交易员应用程序
接口

中国金融期货交易所
China Financial Futures Exchange

2011 年 6 月 21 日

文件版本号	修正日期	备　注
V1.5	2011-06-21	首次发布

目 录

第1章 介绍

1.1. TraderAPI 简介

交易系统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全部的交易功能，包括报单录入、报单撤销、报单查询、成交单查询、会员客户查询、会员持仓查询、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。该类库包含以下5个文件：

文件名	版本	文件描述
CFFEXFtdcTraderApi.h	V1.5 L300	交易接口头文件
CFFEXFtdcUserApiStruct.h	V1.5 L300	定义了 UserAPI 所需的一系列数据类型的头文件
CFFEXFtdcUserApiDataType.h	V1.5 L300	定义了一系列业务相关的数据结构的头文件
CFFEXtraderapi.dll	V1.5 L300	Windows 动态链接库二进制文件
CFFEXtraderapi.lib	V1.5 L300	导入库文件
CFFEXtraderapi.so	V1.5 L300	Linux 动态库

Windows 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。

注意： V1.5 交易系统支持许多新的报单指令（比如市价、最优价、组合交易等），还支持新的交易品种（比如期权及其报价），但受现有交易规则限制，V1.5 交易系统并没有开放上述功能。会员系统在开发过程中需注意“**当前版本不开放的业务**”，同时在每个功能描述中的具体描述。

1.2. TraderAPI 发行的平台

目前发布了以下操作系统平台的版本：

- Intel X86/WindowsXP：包括.h 文件、.dll 文件和.lib 文件。
- Linux RedHat5.5：包括.h 文件和.so 文件。

1.3. 修改历史

1.3.1. 版本 1.5

本版本基于《TraderAPI V1.2 L300》修改。主要有以下变更：

- 本版本提供了灾备功能：
  - 增加【灾备接口】，简要说明了灾备原理。
  - 由于登录报文中增加了数据中心代码，TraderAPI 修改了 ReqUserLogin 和 OnRspUserLogin 方法的参数。
  - TraderAPI 中增加了“数据流回退通知”OnRtnFlowMessageCancel 方法的说明。
- 本版本提供了数据流长度的查询功能：
  - 登录交易系统时，应答中会返回当前会员私有流长度和交易员私有流长度。
  - TraderAPI 增加了对 ReqQryTopic 和 RspQryTpoic 方法说明，用于查询流长度。
- 对以前版本发现问题的修正：
  - TraderAPI 增加了 GetVersion 方法说明，之前版本虽提供功能但无文档说明。
- 使用限制
  - 为提高 API 的性能，在 WIN32 环境下，TraderAPI 初始化时随机使用了 39901-39950 的 Tcp 端口。Linux 版本没有这个限制。

第2章 体系结构

交易员 API 使用建立在 TCP 协议之上 FTD 协议与交易所的交易前置系统进行通讯。交易前置系统负责会员系统的交易业务，而不会发出行情。接收行情需要使用另外的“行情 API”。

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会员端主动发起的通讯请求。该请求被交易所端接收和处理，并给予响应。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，向某个特定的会员发出的信息。例如成交回报等。

广播通讯模式是指交易所端主动，向市场中的所有会员都发出相同的信息。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：

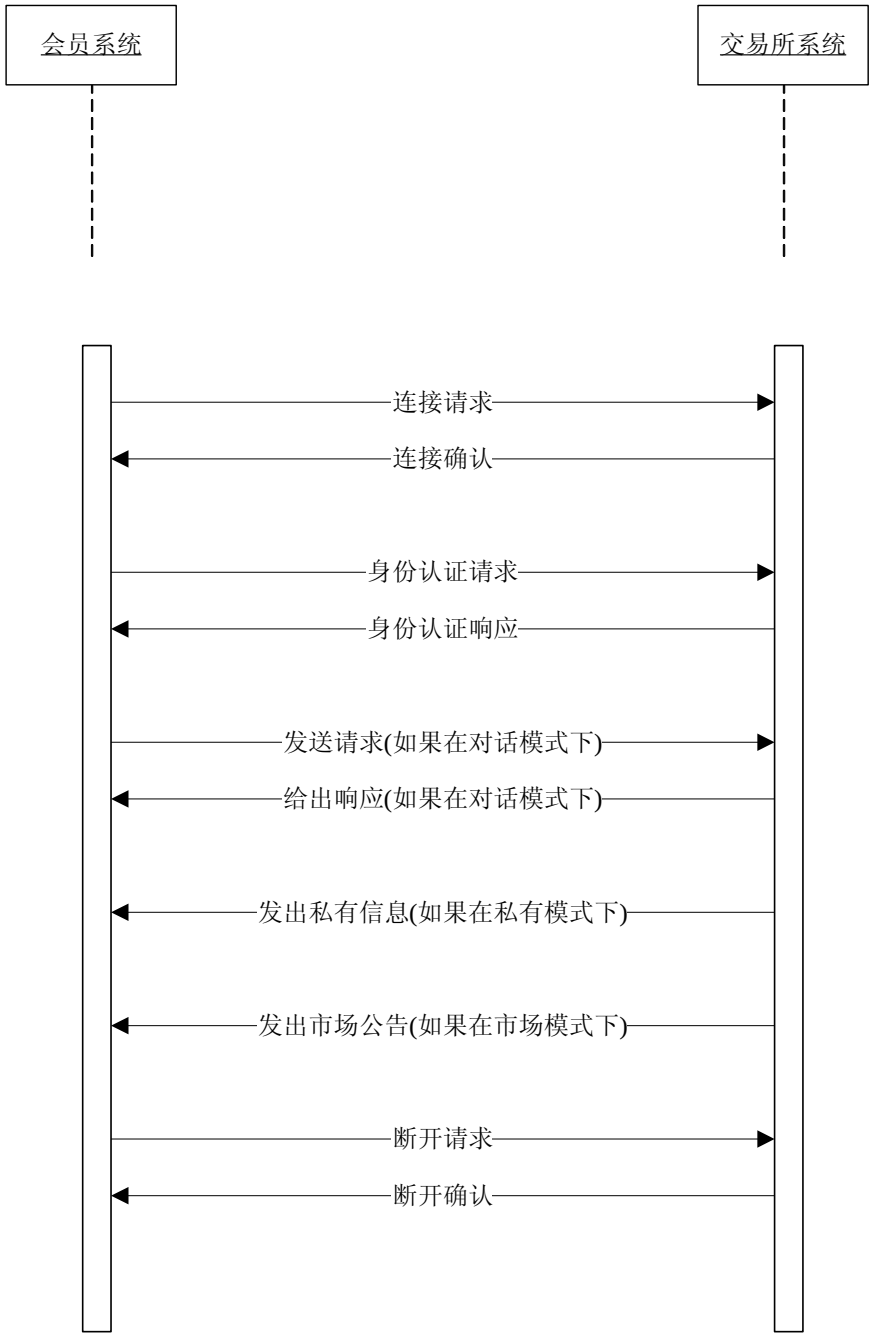


图1) 各通讯模式的工作过程

2.2 数据流

交易前置支持对话通讯模式、私有通讯模式、广播通讯模式：

1、对话通讯模式

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，会员系统发送交易请求，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，会员系统发送查询请求，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。

2、私有通讯模式

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，由交易系统发向会员系统，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，交易系统维护每个会员系统的私有流，在一个交易日内，会员系统断线后恢复连接时，可以请求交易系统发送指定序号之后的私有流数据。私有数据流向会员系统提供报单状态报告、成交回报更等信息。

3、广播通讯模式

广播通讯模式下支持公共数据流：

公共数据流是一个单向数据流，由交易系统发向会员系统，用于发送市场公共信息；公共数据流也是一个可靠的数据流，交易系统维护整个系统的公共数据流，在一个交易日内，会员系统断线恢复连接时，可以请求交易系统发送指定序号之后的公共数据流数据。

第3章 接口模式

交易员API提供了二个接口，分别为CFfexFtdcTraderApi和CFfexFtdcTraderSpi。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CFfexFtdcTraderApi发出操作请求，通继承CFfexFtdcTraderSpi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 CFfexFtdcTraderApi::ReqXXX(
                CFfexFtdcXXXField *pReqXXX,
                int nRequestID)
////响应：
void CFfexFtdcTraderSpi::OnRspXXX(
                CFfexFtdcXXXField *pRspXXX,
                CFfexFtdcRspInfoField *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。该参数根据请求命令的不同使用具有不同的类，该类的成员变量的类型和合法的数值请参阅附录中的“枚举值列表”和“数据类型列表”。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。在接收交易后台的响应时，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，CFfexFtdcTraderSpi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

私有流中的数据中会员的私有信息，包括报单回报、成交回报、报价回报、执行宣告回报等。

通过私有流接收回报的编程接口通常如下：

```
void CFfexFtdcTraderSpi::OnRtnXXX(CFfexFtdcXXXField *pXXX)
//// 或
void CFfexFtdcTraderSpi::OnErrRtnXXX(CFfexFtdcXXXField *pXXX,
    CFfexFtdcRspInfoField *pRspInfo)
```

当收到交易后台通过私有流发布的回报数据时，CFfexFtdcTraderSpi 的回调函数会被调用。回调函数的参数为回报的具体内容。

3.3 公共流编程接口

公共流中的数据中交易所的公共信息，包括合约、公告等。

通过公共流接收回报的编程接口通常如下：

```
void CFfexFtdcTraderSpi::OnRtnXXX(CFfexFtdcXXXField *pXXX)
```

当收到交易后台通过公共流发布的回报数据时，CFfexFtdcTraderSpi 的回调函数会被调用。回调函数的参数为通知的具体内容。

第4章 运行模式

4.1 工作流程

会员系统和交易系统的交互过程分为 2 个阶段：初始化阶段和功能调用阶段。

4.1.1 初始化阶段

在初始化阶段，会员系统的程序必须完成如下步骤（具体代码请参考开发实例）：

顺序	会员系统
1	产生一个 CFfexFtdcTraderApi 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； 订阅公共流；
5	设置交易前置 NameServer 的网络地址。[1]
6	**初始化**

[1] 为了保持与上一版的兼容性，API 仍然提供了注册交易前置（行情服务）的接口，但交易所建议不要使用这些接口，这些接口将在下一版本中取消。有关 NameServer 的说明参见**错误！未找到引用源。**前置机列表。

4.1.2 功能调用阶段

在功能调用阶段，会员系统可以任意调用交易接口中的请求方法，如 ReqUserLogin、ReqOrderInsert 等，同时提供回调函数以响应回报信息。注意事项：

- API 请求的输入参数不能为 NULL。
- 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。

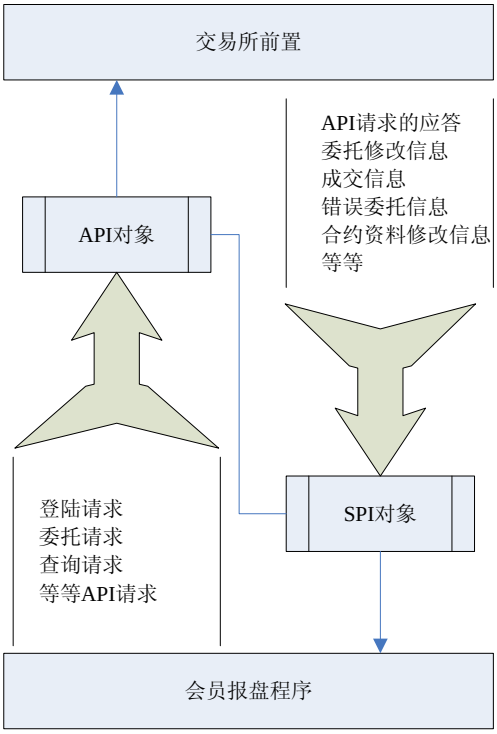
4.2 工作线程

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。

CFfexFtdcTraderApi 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。

CFfexFtdcTraderSpi 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，可以从交易前置收取所需数据。

如果重载的某个回调函数阻塞，则等于阻塞了 TraderAPI 工作线程，TraderAPI 与交易系统的通讯会停止。因此，在 CFfexFtdcTraderSpi 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。



4.3 会员系统使用 TraderAPI 与交易系统的交互

会员系统通过 TraderAPI 与交易系统交互。会员系统的请求通过 TraderAPI 发送到交易系统；交易系统返回的应答和回报通过 TraderAPI 返回给会员系统。

TraderAPI 的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(包括全部成交和部分成交)，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会员下其他的交易员接收到（如果该用户不使用仅接收交易员私有流方式登录）。

以交易员日常交易为例，有两个会员系统 A 和 B，发生的事件包括：

1．交易员 A 报单，IF0911，买，10 手，900000 元；

- CFfexFtdcTraderApi::ReqOrderInsert：报单录入请求。本函数由会员系统的应用主线程调用，通过对话流发送到 V1.5 交易系统前置机。
- 交易系统报单处理：报单系统编号为 1；由于此时撮合队列中无对手，报单状态为“未成交还在队列中”。V1.5 交易系统的前置机发出报单响应给交易员 A 的对话流；发出报单回报给交易员 A 的私有流和交易员 A 所属会员的私有流。响应和回报的报文由 TraderAPI 工作线程处理并调用 Spi 对象的方法。
- CFfexFtdcTraderSpi::OnRsqOrderInsert：交易所的交易前置给出请求的应答，内容为：录入成功，本地编号为 1 的报单的系统编号为 1。本函数由 TraderAPI 工作线程在收到交易前置应答后调用。
- CFfexFtdcTraderSpi::OnRtnOrder：交易所的交易前置立即在交易员 A 所属会员的私有流中或者交易员 A 的私有流中给出报单回报，因其他席位无法获得报单的具体信息，为保持信息同步，因此回报内容包括报单的状态等报单的全部内容。本函数由 TraderAPI 工作线程在收到交易前置的报单回报后调用。如果会员 A 还有其他交易员连接并登录到交易系统，并且接收会员私有流，将收到相同的报单回报（以下同）。

2．交易员 B 报单，IF0911，卖，5 手，450000 元；

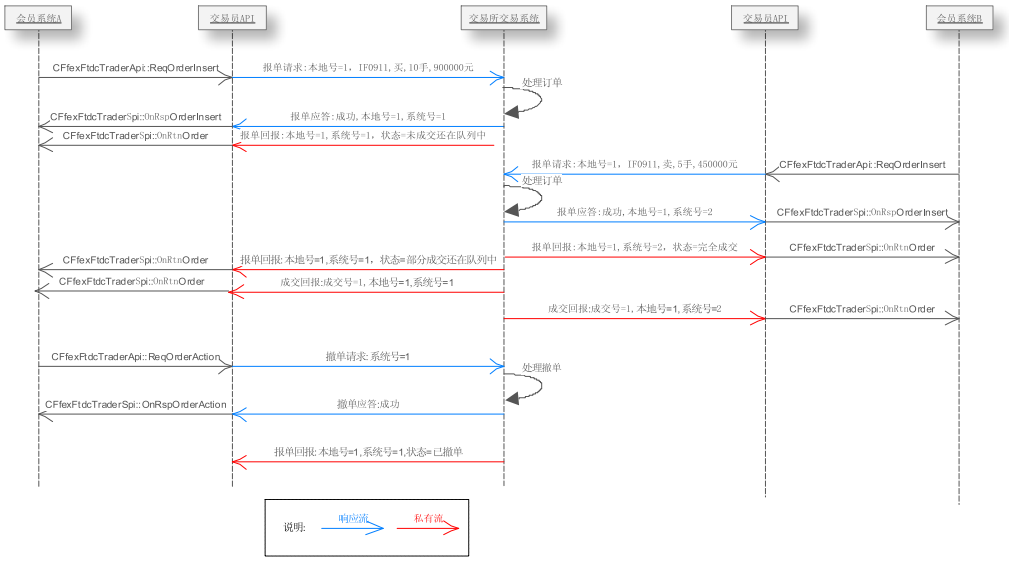
- CFfexFtdcTraderApi::ReqOrderInsert：报单录入请求。
- 交易系统报单处理：报单系统编号为 2；由于此时撮合队列中无对手，报单状态为“未成交还在队列中”。
- 交易系统报单处理：报单系统编号为 2；尝试撮合并能成交，因此报单状态为“完全成交”。V1.5 交易系统的前置机发出报单响应给交易员 B 的对话流；发出报单回报给交易员 B 的私有流和交易员 B 所属的会员私有流；发出报单回报给交易员 A 的私有流和交易员 A 所属会员的私有流，提示系统编号为 1 的报单状态交易系统已经修改为“部分成交还在队列中”，报单的“剩余数量”调整为 5 手；发出成交回报给交易员 B 的私有流和交易员 B 所属的会

员私有流；发出成交回报给交易员 B 的私有流和交易员 B 所属的会员私有流。**v1.5 交易系统将确保报单的回报在成交回报之前送达会员系统，并且报单回报中的“剩余数量”字段已经反映交易系统报单簿中的最新数量，没有必要再根据成交回报的数量再调整。**

- CFfexFtdcTraderSpi::OnRsqOrderInsert：交易所的交易前置给出请求的应答，内容为：录入成功，本地编号为 1 的报单的系统编号为 2。
- CFfexFtdcTraderSpi::OnRtnOrder：交易所的交易前置立即在交易员 B 所属会员的私有流中或者交易员 B 的私有流中给出报单回报，报单状态为“完全成交”。
- CFfexFtdcTraderSpi::OnRtnOrder：交易所的交易前置立即在交易员 A 所属会员的私有流中或者交易员 A 的私有流中给出报单回报，报单状态为“部分成交还在队列中”，剩余数量为 5 手。
- CFfexFtdcTraderSpi::OnRtnTrade：交易所的交易前置立即在交易员 A 所属会员的私有流中或者交易员 A 的私有流中给出成交回报。
- CFfexFtdcTraderSpi::OnRtnTrade：交易所的交易前置立即在交易员 B 所属会员的私有流中或者交易员 B 的私有流中给出成交回报。

3. 交易员 A 撤单；

下图说明了会员系统、TraderAPI 和交易系统之间的 UML 交互图。



4.4 与交易所前置系统的连接

TraderAPI 使用建立在 TCP 协议之上的 FTD 协议与交易所的交易/行情前置系统进行通信。TraderAPI 使用 CFfexFtdcTraderSpi::RegisterFront 方法注册交易所前置系统的网络地址。

交易所拥有多个交易前置系统，用于负载均衡且互为备份，从而提高系统的

性能和可靠性。为保证交易时通信的可靠性，TraderAPI 可以注册多个前置。API 在初始化后，会从已注册的前置中随机挑选一个前置，尝试建立网络连接，如果不成功，则依次逐个尝试其它前置，直到连接成功为止。如果在交易过程中网络连接出现故障， API 依然使用上述过程，尝试连接其它前置。

中国金融期货交易所将公布至少两台前置机的网络地址，因此会员系统应该至少注册两个前置机的网络地址以防止所连交易前置发生故障从而引发单点隐患。鉴于 FTD 协议的带宽要求比较高，建议会员使用 128K 以上 DDN 专线或者 2M SDH 数字专线。中国金融期货交易所和上海期货交易所将共享会员远程交易网络接入：直接连接中国金融期货交易所的链路可以作为备份链路接入上海期货交易所；反之亦然。

中国金融期货交易所将启用 NameServer，将来只公布 NameServer 的地址而不公布前置地址，TraderAPI 使用 CFfexFtdcTraderApi:: RegisterNameServer 方法注册交易所名字服务器网络地址。可以多次调用该函数注册多个名字服务器网络地址。TraderAPI 会自动连接 NameServer 获取一组前置地址然后连接前置。

4.5 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"。 交易 API 与行情 API 的本地文件必须分目录存放。

4.6 请求/应答日志文件

TraderAPI 提供了两个日志接口，用于记录通信日志。OpenRequestLog 用于打开请求日志，OpenResponseLog 用于打开应答日志。日志打开后，所有的业务请求将记入请求日志，所有的业务应答和回报将记入应答日志。注意，为保密和节约存储空间，登录请求/应答和查询请求/应答不记日志。

请求格式为：

日期 时间, 请求名称,请求结果, [请求参数名, 请求参数内容]

应答格式为：

日期 时间, 应答名称,响应代码,响应信息, [应答参数名, 应答参数内容]

回报格式为：

日期 时间, 回报名称, [回报参数名,回报参数内容]

4.7 可靠数据流的订阅方式

FTD 协议中私有流、公共流、行情流等可以保证数据可靠、有序地传送到对方，称为可靠数据流。可靠数据流对于保证会员系统的数据正确性和完整性非常重要。例如，会员系统通过交易系统发出的会员私有流中的各种回报，就可以得到足够的信息，从而完成会员端的业务操作。为了保证会员系统业务的正确性，就需要可靠、有序、唯一地接收私有流中的报文。

可靠数据流是依靠重传来保证数据可靠、有序地传送的。即客户端负责管理数据流的序号，如果出现传输中断，可以从指定序号开始，重新订阅数据流，来保证数据的完整性。

对话流和查询流不支持重传，是不可靠的数据流。

交易系统接口提供了两种管理可靠数据流的方式：API 维护重传报文序号和会员系统维护重传报文的序号。

4.7.1 可靠数据流的订阅方式

每当 API 收到一个可靠数据流报文，则（a）先调用 SPI 的回调函数通知会员系统，（b）将该报文的序号记入本地文件中（扩展名为.con）。如果会员系统退出后再重新订阅数据流，则可以使用本地文件中记录的报文序号来订阅数据流。

CFfexFtdcTraderApi 的 SubscribePrivateTopic 方法、SubscribePublicTopic 方法、SubscribeUserTopic 方法用于订阅可靠数据流。

通过接口参数可以指定重传方式，重传方式有三种：重发（RESTART）、续传（RESUME）和快照（QUICK）。

- 重发（RESTART）方式从数据流中的第一个报文开始传输，这时 API

将忽略本地文件中记录的数据流报文序号。

- 续传（RESUME）方式从本地文件中记录的数据流报文序号之后开始传输。为保持会员交易数据的完整性，对于会员或者交易员私有流，交易所建议使用 RESUME 重传方式。
- 快照（QUICK）方式从订阅这一时刻数据流最大序号开始传输。快照方式主要用于不需要保证数据完整性的场合，对于会员或者交易员私有流，交易所不建议使用 QUICK 方式。

API 维护重传报文的序号存在一定数据一致性的风险，例始，如果（a）完成后，（b）未完成，则会造成同一个报文两次回调到会员系统，从而导致会员系统处理上比较困难。另外，如果记录数据流序号的本地文件损坏，则不得不重传数据流，可能会影响会员系统的效率。

如果使用 API 维护重传报文的序号，则 API 会将上次登录返回的 TradingDay 和 DataCenterID 这两个字段记录在文件中（resume.con）；登录时 API 会使用文件中的值覆盖会员系统所填的这两个字段。

4.7.2 会员系统维护重传报文的序号

每当 API 收到一个可靠数据流的报文，则（a）先调用 SPI 的 OnPackageStart 函数通知会员系统收到一个报文，（b）调用 SPI 的回调函数通知会员系统业务数据，（c）最后调用 SPI 的 OnPackageEnd 函数通知会员系统该报文的回调结束。在 OnPackageStart 和 OnPackageEnd 接口中，会员系统可以得到当前回调的报文的序号。会员系统可以记录该序号，在重传可靠数据流时，将该序号做为 CFfexFtdcTraderApi::ReqSubscribeTopic 方法的参数（与 RESUME 方式类似）。

通过 CFfexFtdcTraderApi::ReqSubscribeTopic 方法，会员系统可以指定数据流重传报文的序号。如果指定序号为 0，则重发整个数据流（与 RESTART 方式类似）；如果指定序号为-1，则从订阅这一时刻数据流最大序号开始传输，（与 QUICK 方式类似）。

会员系统重传维护报文的序号比 API 维护重传报文的序号有更好的一致性和可靠性，对事务完整性要求高的会员系统应尽量使用这个方法。

注意登录时，TradingDay 和 DataCenterID 应填写为上次登录应答的返回值。

如果是第一次登录或不需要续传，则 TradingDay 可以填空字符串（””），DataCenterID 可以填 0 或交易所公布的主数据中心代码。

4.8 心跳机制

会员系统和交易所的前置机之间使用 TCP 虚链路进行通信。假设通讯链路发生故障，并且在这段时间内会员系统和前置机没有数据通信，或者更确切地说，双方都没有调用 Socket recv()和 Socket send()函数，则双方无法辨别目前系统的工作状态，需等待 Socket 超时，一般而言操作系统定义的超时时间较长，不利于通信双方进行监测以提高响应速度和自动恢复处理。

可以通过增加附加的心跳信息来监测通信双方的工作状态，其原理非常简单，并且不会增加双方的通讯成本：在有业务数据传送时，双方能够检测出链路和通信状态；如果没有业务数据传送，则需给对方发送心跳（Heartbeat）信息（此时链路上无传输数据，增加的心跳信息不会对带宽带来压力和成本）。虽然没有增加通讯成本，但对于服务器（比如交易所前置机）而言，随着连接数量增加，其巡检成本（每秒监测是否需要发送心跳信息和维护连接表）也将线性增加。

增设心跳报文用于检测连接是否有效。连接的一端如果在规定超时时间（timeout）内未收到对方的任何报文，则认为 TCP 虚链路失效，应主动断开；如果一方在一定的时间间隔（interval）内未向对方发送任何报文，则应向对方发送心跳报文，以维持 TCP 虚链路的正常状态。通常，timeout 是 interval 的三倍。

API 提供了 void SetHeartbeatTimeout(unsigned int timeout)方法，用于设定会员系统检测 TCP 虚链路有效性的超时时间：交易系统在空闲时会每隔(timeout-1)/3 秒向 API 发送心跳报文；若超过 timeout/2 秒未收到交易系统的任何报文时，将触发回调 CFfexFtdcTraderApi::OnHeartBeatWarning()；若超过 timeout 秒未收到交易系统的任何报文时，TCP 连接将会中断，并且触发回调 CFfexFtdcTraderApi ::OnFrontDisconnected()。

例如，会员端将心跳超时设置为 16 秒，则交易系统在空闲时会每 5 秒向 API 发送一个心跳报文。如果 API 在 8 秒内未收到交易系统的任何报文，则触发回调 CFfexFtdcTraderApi::OnHeartBeatWarning()。如果超过 16 秒未收到交易系统的任何报文，API 会主动断开网络连接，并且触发回调

CFfexFtdcTraderApi::OnFrontDisconnected()。此时会员端可以选用备用专线链路重连交易所前置机。

交易所前置机也通过心跳方式监测会员端系统的 TCP 连接：如果会员系统未调用 SetHeartbeatTimeout 方法，则心跳超时目前固定设置为 120 秒；如果会员系统调用 SetHeartbeatTimeout 方法，timeout 参数将同步用于前置机对会员端的监测。这个机制非常有用：在链路中断后前置机将可在大致可用时间内（timeout 参数+5 秒左右）主动断开与会员端 TCP 连接，这样会员可以使用备用线路（不同 IP 地址）登录，否则前置机将认为原地址的 TCP 连接仍然有效并拒绝备用地址登录。

若会员系统未调用过 SetHeartbeatTimeout 方法，在 V1.20 之前（不含）版本 API 的缺省的 timeout 设置为 120 秒，缺省的警告时间为 80 秒。为加快会员端交易系统监测专线链路中断的速度，从 V1.20 版本开始，API 将在初始化建立与前置机 TCP 连接后将主动调用 SetHeartbeatTimeOut()方法将 timeout 设置为 10 秒。参数 timeout 允许的最小值为 4。参数 timeout 设置过高，发生链路中断情况下会员端系统切换时间将很长；设置过小，将可能发生非预期切换，需要综合考虑会员端应用情况和网络情况合理设置。

交易所建议会员系统将 timeout 值设置为 10 至 30 秒之间。

4.9 前置机列表

会员系统连接交易所前置机后方能接入 V1.5 交易系统。为了容错和负载均衡，交易所将在主用数据中心和备用数据中心部署两组、每组多台前置机。交易所将公布各前置机的网络地址的列表，会员系统从列表中随机选出一个尝试与其建立连接；会员系统在某一时刻只与一个前置机相连，如果由该前置机出现故障导致连接中断或者超时，则会员系统需尝试连接列表中的其它前置机。

会员系统得到前置列表的方式有两种：

- 交易所公布前置列表，会员系统通过 API 的 RegisterFront 接口，将前置机逐一注册进 API。
- 交易系统提供 NameServer，其功能是向 API 公布前置机列表。交易所首先公布 NameServer 列表，会员系统通过 API 的 RegisterNameServer 接

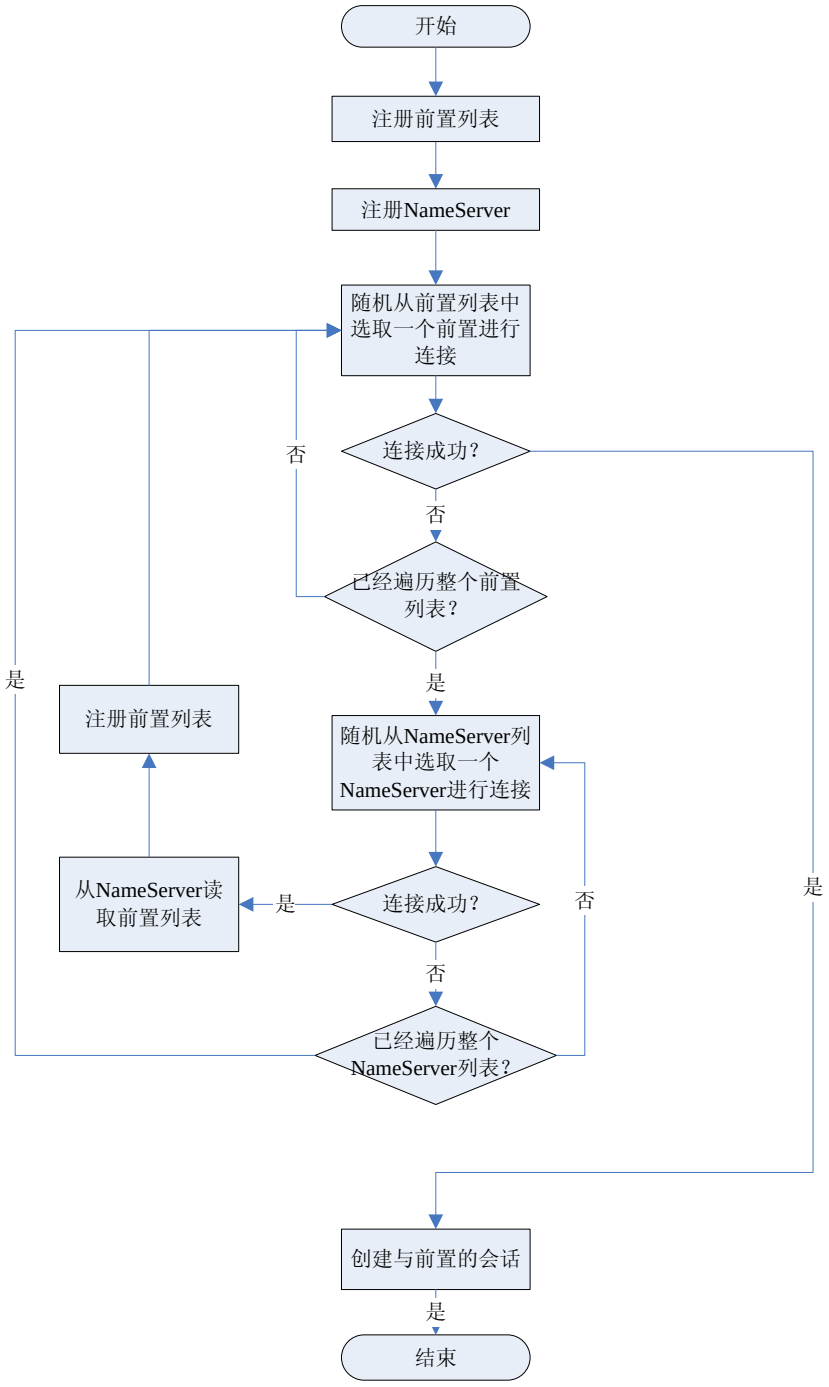
口，将 NameServer 注册进 API。API 先尝试从 NameServer 中得到前置列表，再尝试根据前置列表来连接前置机。

使用 NameServer 的优点在于：

- 增加交易所前置机部署的灵活性，可以根据业务需要和负载短期内新增前置机并无需对会员系统做任何修改。
- NameServer 可以较好地完成主用系统和灾备系统的切换。
- NameServer 功能单一，结构简单，负载也很低，不用考虑负载均衡，可以灵活部署。

会员系统可以同时使用 RegisterFront()方法注册前置机列表和使用 RegisterNameServer()方法注册 NameServer 列表。API 会首先尝试连接已注册的前置，如果连接不成功，再尝试连接 NameServer。

API 连接前置的流程图：



4.10 灾备接口

交易所建有期货大厦（大厦数据中心，使用此接口之前的主用数据中心）和外高桥两个数据中心，两个中心使用高速光纤实现系统的互联互通。V1.5 交易系统同时在两个数据中心运行，主中心进行业务处理；备中心异步接收主中心的数据并同步业务；备中心处于待命状态。

当主数据中心发生灾难性事件时，可以切换到备中心；备数据中心接管主数

据中心工作，在主数据中心处理的业务基础上继续进行业务处理。数据中心切换可能出现部分业务数据丢失。会员端系统需要通过 API 接口，获知要取消的交易。

一、API 用户登录请求接口中增加“数据中心代码”字段，标识上次登录的数据中心代码。在用户登录应答接口中，同样增加“数据中心代码”字段，交易系统返回当前使用的数据中心代码。会员端系统应保存交易系统返回的数据中心代码，在下次登录时填入登录请求中。

二、API 增加“取消交易”（OnRtnFlowMessageCancel）接口，在会员端发出订阅请求后，通知订阅的主题中要取消的报文。会员端系统可根据该接口，得到被取消的报文的序号，从而找到原始报文。原始报文的序号可通过 OnPackageStart 和 OnPackageEnd 接口获得。

第5章 TraderAPI 接口分类

5.1 管理接口

管理接口是对 API 的生命周期和运行参数进行控制。

接口类型	接口名称	说明
生命周期管理接口	CFfexFtdcTraderApi:: CreateFtdcTraderApi	创建 TraderApi 实例
	CFfexFtdcTraderApi:: GetVersion	获取 API 版本
	CFfexFtdcTraderApi:: Release	删除接口实例
	CFfexFtdcTraderApi:: Init	初始化
	CFfexFtdcTraderApi:: Join	等待接口线程结束运行
参数管理接口	CFfexFtdcTraderApi:: RegisterSpi	注册回调接口
	CFfexFtdcTraderApi:: RegisterFront	注册前置机网络地址
	CFfexFtdcTraderApi:: RegisterNameServer	注册 NameServer 网络地址
	CFfexFtdcTraderApi:: RegisterCertificateFile	加载证书
	CFfexFtdcTraderApi:: SetHeartbeatTimeout	设置心跳超时时间
订阅接口	CFfexFtdcTraderApi:: SubscribePrivateTopic	订阅私有流
	CFfexFtdcTraderApi:: SubscribePublicTopic	订阅公共流
	CFfexFtdcTraderApi:: SubscribeUserTopic	订阅交易员流
日志接口	CFfexFtdcTraderApi:: OpenRequestLog	打开请求日志文件
	CFfexFtdcTraderApi:: OpenResponseLog	打开应答日志文件
通信状态接口	CFfexFtdcTraderSpi:: OnFrontConnected	与交易系统建立起通信连接时（还未登录前），该方法被调用
	CFfexFtdcTraderSpi:: OnFrontDisconnected	与交易系统通信连接断开时，该方法被调用
	CFfexFtdcTraderSpi:: OnHeartBeatWarning	当长时间未收到报文时，该方法被调用
	CFfexFtdcTraderSpi:: OnPackageStart	报文回调开始通知
	CFfexFtdcTraderSpi:: OnPackageEnd	报文回调结束通知
灾备接口	CFfexFtdcTraderSpi:: OnRtnFlowMessageCancel	数据流回退通知

5.2 业务接口

业务类型	业务	请求接口 / 响应接口	数据流
登录	登录	CFfexFtdcTraderApi:: ReqUserLogin CFfexFtdcTraderSpi::OnRspUserLogin	N/A
	登出	CFfexFtdcTraderApi::ReqUserLogout CFfexFtdcTraderSpi::OnRspUserLogout	对话流
	修改用户口令	CFfexFtdcTraderApi::ReqUserPasswordUpdate CFfexFtdcTraderSpi::OnRspUserPasswordUpdate	对话流
订阅	订阅主题	CFfexFtdcTraderApi::ReqSubscribeTopic CFfexFtdcTraderSpi::OnRspSubscribeTopic	对话流
	查询主题	CFfexFtdcMduserApi::ReqQryTopic CFfexFtdcMduserSpi::OnRspQryTopic	查询流
交易	报单录入	CFfexFtdcTraderApi::ReqOrderInsert CFfexFtdcTraderSpi::OnRspOrderInsert	对话流
	报单操作	CFfexFtdcTraderApi::ReqOrderAction CFfexFtdcTraderSpi::OnRspOrderAction	对话流
	组合报单录入	CFfexFtdcTraderApi::ReqCombOrderInsert CFfexFtdcTraderSpi::OnRspCombOrderInsert	对话流
	报价录入	CFfexFtdcTraderApi::ReqQuoteInsert CFfexFtdcTraderSpi::OnRspQuoteInsert	对话流
	报价操作	CFfexFtdcTraderApi::ReqQuoteAction CFfexFtdcTraderSpi::OnRspQuoteAction	对话流
	执行宣告录入	CFfexFtdcTraderApi::ReqExecOrderInsert CFfexFtdcTraderSpi::OnRspExecOrderInsert	对话流
	执行宣告操作	CFfexFtdcTraderApi::ReqExecOrderAction CFfexFtdcTraderSpi::OnRspExecOrderAction	对话流
私有回报	成交回报	CFfexFtdcTraderSpi::OnRtnTrade	私有流
	报单回报	CFfexFtdcTraderSpi::OnRtnOrder	私有流
	组合报单回报	CFfexFtdcTraderSpi::OnRtnCombOrder	私有流
	报价回报	CFfexFtdcTraderSpi::OnRtnQuote	私有流
	执行宣告回报	CFfexFtdcTraderSpi::OnRtnExecOrder	私有流
	报单录入错误回报	CFfexFtdcTraderSpi::OnErrRtnOrderInsert	私有流
	报单操作错误回报	CFfexFtdcTraderSpi::OnErrRtnOrderAction	私有流
	组合报单录入错误回报	CFfexFtdcTraderSpi::OnErrRtnCombOrderInsert	私有流
	报价录入错误回报	CFfexFtdcTraderSpi::OnErrRtnQuoteInsert	私有流
	报价操作错误回报	CFfexFtdcTraderSpi::OnErrRtnQuoteAction	私有流
	执行宣告录入错误回报	CFfexFtdcTraderSpi::OnErrRtnExecOrderInsert	私有流
	执行宣告操作错误回报	CFfexFtdcTraderSpi::OnErrRtnExecOrderAction	私有流
公共通知	合约交易状态通知	CFfexFtdcTraderSpi::OnRtnInstrumentStatus	公共流
	增加合约通知	CFfexFtdcTraderSpi::OnRtnInsInstrument	公共流
	删除合约通知	CFfexFtdcTraderSpi::OnRtnDelInstrument	公共流

业务类型	业务	请求接口 / 响应接口	数据流
	增加组合合约单腿通知	CFfexFtdcTraderSpi::OnRtnInsCombinationLeg	公共流
	删除组合合约单腿通知	CFfexFtdcTraderSpi::OnRtnDelCombinationLeg	公共流
	别名定义通知	CFfexFtdcTraderSpi::OnRtnAliasDefine	公共流
	公告通知	CFfexFtdcTraderSpi::OnRtnBulletin	公共流
查询	资金查询	CFfexFtdcTraderApi::ReqQryPartAccount CFfexFtdcTraderSpi::OnRspQryPartAccount	查询流
	报单查询	CFfexFtdcTraderApi::ReqQryOrder CFfexFtdcTraderSpi::OnRspQryOrder	查询流
	组合报单查询	CFfexFtdcTraderApi::ReqQryCombOrder CFfexFtdcTraderSpi::OnRspQryCombOrder	查询流
	报价查询	CFfexFtdcTraderApi::ReqQryQuote CFfexFtdcTraderSpi::OnRspQryQuote	查询流
	成交查询	CFfexFtdcTraderApi::ReqQryTrade CFfexFtdcTraderSpi::OnRspQryTrade	查询流
	客户查询	CFfexFtdcTraderApi::ReqQryClient CFfexFtdcTraderSpi::OnRspQryClient	查询流
	会员持仓查询	CFfexFtdcTraderApi::ReqQryPartPosition CFfexFtdcTraderSpi::OnRspQryPartPosition	查询流
	客户持仓查询	CFfexFtdcTraderApi::ReqQryClientPosition CFfexFtdcTraderSpi::OnRspQryClientPosition	查询流
	合约查询	CFfexFtdcTraderApi::ReqQryInstrument CFfexFtdcTraderSpi::OnRspQryInstrument	查询流
	合约交易状态查询	CFfexFtdcTraderApi::ReqQryInstrumentStatus CFfexFtdcTraderSpi::OnRspQryInstrumentStatus	查询流
	保值额度查询	CFfexFtdcTraderApi::ReqQryHedgeVolume CFfexFtdcTraderSpi::OnRspQryHedgeVolume	查询流
	行情查询	CFfexFtdcTraderApi::ReqQryMarketData CFfexFtdcTraderSpi::OnRspQryMarketData	查询流
	公告查询	CFfexFtdcTraderApi::ReqQryBulletin CFfexFtdcTraderSpi::OnRspQryBulletin	查询流
	合约价位查询	CFfexFtdcTraderApi::ReqQryMBLMarketData CFfexFtdcTraderSpi::OnRspQryMBLMarketData	查询流

5.3 当前版本不开放的接口

业务类型	业务	请求接口 / 响应接口	开放情况
交易	报单录入	CFfexFtdcTraderApi::ReqOrderInsert CFfexFtdcTraderSpi::OnRspOrderInsert	部分开放
	报单操作	CFfexFtdcTraderApi::ReqOrderAction CFfexFtdcTraderSpi::OnRspOrderAction	部分开放
	组合报单录入	CFfexFtdcTraderApi::ReqCombOrderInsert CFfexFtdcTraderSpi::OnRspCombOrderInsert	不开放
	报价录入	CFfexFtdcTraderApi::ReqQuoteInsert CFfexFtdcTraderSpi::OnRspQuoteInsert	不开放
	报价操作	CFfexFtdcTraderApi::ReqQuoteAction CFfexFtdcTraderSpi::OnRspQuoteAction	不开放
	执行宣告录入	CFfexFtdcTraderApi::ReqExecOrderInsert CFfexFtdcTraderSpi::OnRspExecOrderInsert	不开放
	执行宣告操作	CFfexFtdcTraderApi::ReqExecOrderAction	不开放

业务类型	业务	请求接口 / 响应接口	开放情况
		CFfexFtdcTraderSpi::OnRspExecOrderAction	
回报	组合报单回报	CFfexFtdcTraderSpi::OnRtnCombOrder	不开放
	报价回报	CFfexFtdcTraderSpi::OnRtnQuote	不开放
	执行宣告回报	CFfexFtdcTraderSpi::OnRtnExecOrder	不开放
	组合报单录入错误回报	CFfexFtdcTraderSpi::OnErrRtnCombOrderInsert	不开放
	报价录入错误回报	CFfexFtdcTraderSpi::OnErrRtnQuoteInsert	不开放
	报价操作错误回报	CFfexFtdcTraderSpi::OnErrRtnQuoteAction	不开放
	执行宣告录入错误回报	CFfexFtdcTraderSpi::OnErrRtnExecOrderInsert	不开放
	执行宣告操作错误回报	CFfexFtdcTraderSpi::OnErrRtnExecOrderAction	不开放
公共通知	增加组合合约单腿通知	CFfexFtdcTraderSpi::OnRtnInsCombinationLeg	不开放
	删除组合合约单腿通知	CFfexFtdcTraderSpi::OnRtnDelCombinationLeg	不开放
查询	组合报单查询	CFfexFtdcTraderApi::ReqQryCombOrder CFfexFtdcTraderSpi::OnRspQryCombOrder	不开放

第6章 TraderAPI 参考手册

6.1 CFfexFtdcTraderSpi 接口

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

6.1.1 OnFrontConnected 方法

当会员系统与交易系统的交易前置机建立起 TCP 虚链路（连接）后，该方法被调用。该连接是由 API 自动建立的。

函数原型：

```
void OnFrontConnected();
```

注意： OnFrontConnected 被调用仅说明 TCP 连接成功，会员系统必须自行登录，才能进行后续的业务操作。登录失败不会回调该方法。

6.1.2 OnFrontDisconnected 方法

当会员系统与交易系统的交易前置机的 TCP 虚链路断开时，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。

函数原型：

```
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

6.1.3 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。缺省超时告警时段设置为 80 秒。如果调用过 SetHeartbeatTimeout(unsigned int timeout)自行设定心跳超时，则警告时间为 timeout/2。

函数原型：

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

参数：

nTimeLapse：距离上次接收报文的时间

6.1.4 OnPackageStart 方法

报文回调开始通知。当 API 收到一个报文后，首先调用本方法，然后是各数据域的回调，最后是报文回调结束通知。

函数原型：

```
void OnPackageStart (int nTopicID, int nSequenceNo);
```

参数：

nTopicID: 主题代码（如私有流、公共流、行情流等）。

nSequenceNo: 报文序号。

6.1.5 OnPackageEnd 方法

报文回调结束通知。当 API 收到一个报文后，首先调用报文回调开始通知，然后是各数据域的回调，最后调用本方法。

函数原形：

```
void OnPackageEnd (int nTopicID, int nSequenceNo);
```

参数：

nTopicID: 主题代码（如私有流、公共流、行情流等）。

nSequenceNo：报文序号。

6.1.6 OnRspUserLogin 方法

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

函数原型：

```
void OnRspUserLogin(
CFfexFtdcRspUserLoginField *pRspUserLogin,
CFfexFtdcRspInfoField *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 CFfexFtdcRspUserLoginField
{
    ///交易日
    TFfexFtdcDateType   TradingDay;
    ///登录成功时间
    TFfexFtdcTimeType   LoginTime;
    ///最大本地报单号
    TFfexFtdcOrderLocalIDType  MaxOrderLocalID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///会员代码
    TFfexFtdcParticipantIDType  ParticipantID;
    ///交易系统名称
    TFfexFtdcTradingSystemNameType  TradingSystemName;
    ///数据中心代码
    TFfexFtdcDataCenterIDType  DataCenterID;
    ///会员私有流当前长度
    TFfexFtdcSequenceNoType PrivateFlowSize;
    ///交易员私有流当前长度
    TFfexFtdcSequenceNoType  UserFlowSize;
};
```

注意：如果会员系统自行维护重传序号，则应保存好返回的 TradingDay 和 DataCenterID，以便下次登录时填入登录请求中。

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 CFfexFtdcRspInfoField{
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
};
```

可能出现的错误：

错误代码	错误提示	可能的原因
3	会员找不到	登录的会员代码错误
45	结算组初始化状态不对	交易系统尚未初始化完毕，可以稍后再重试
59	用户重复登录	该交易用户已经登录过了
60	用户名或口令错误	登录的交易用户代码或口令错误
62	用户不活跃	交易系统不允许该交易员登录
64	用户不属于此会员	登录的会员代码错误
65	错误的登录 IP 地址	发起登录的计算机没有交易所允许 IP 地址
100	错误的用户类型	非交易用户登录交易系统

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。

6.1.7 OnRspUserLogout 方法

当会员系统发出退出请求之后，交易后台返回响应时，该方法会被调用，通知会员系统退出是否成功。

函数原型：

```
void OnRspUserLogout(CFfexFtdcRspUserLogoutField *pRspUserLogout,
CFfexFtdcRspInfoField *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 CFfexFtdcRspUserLogoutField
{
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///会员代码
    TFfexFtdcParticipantIDType  ParticipantID;
};
```

pRspInfo：返回用户响应信息的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField{
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
};
```

可能出现的错误：

错误代码	错误提示	可能的原因
66	用户尚未登录	尚未登录
67	并没有以此用户登录	登出时的交易员和登录时不同

68	并没有以此会员登录	登出时的会员和登录时不同

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。

6.1.8 OnRspUserPasswordUpdate 方法

用户密码修改应答。当会员系统发出用户密码修改指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspUserPasswordUpdate(
CFfexFtdcUserPasswordUpdateField *pUserPasswordUpdate,
CFfexFtdcRspInfoField *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 CFfexFtdcUserPasswordUpdateField
    {
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///旧密码
        TFfexFtdcPasswordType   OldPassword;
        ///新密码
        TFfexFtdcPasswordType   NewPassword;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
58	用户不匹配	要修改口令的交易员和登录时不同
60	用户名或口令错误	原口令错误
66	用户尚未登录	尚未登录
68	并没有以此会员登录	修改口令的会员号与登录时不同

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。

6.1.9 OnRspSubscribeTopic 方法

订阅主题应答。当会员系统发出订阅主题指令后，交易系统返回响应时，该方法会被调用。

函数原形：

```
void OnRspSubscribeTopic (
    CFfexFtdcDisseminationField *pDissemination,
    CFfexFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast);
```

参数：

pDissemination：指向订阅主题结构的地址，包含了要订阅的主题和起始报文的序号。订阅主题结构：

```
struct CFfexFtdcDisseminationField {
    ///序列系列号
    TFfexFtdcSequenceSeriesType SequenceSeries;
    ///序列号
    TFfexFtdcSequenceNoType SequenceNo;
};
```

pRspInfo：指向响应信息结构的地址。响应信息结构：

```
struct CFfexFtdcRspInfoField {
    ///错误代码
    TFfexFtdcErrorIDType    ErrorID;
    ///错误信息
    TFfexFtdcErrorMsgType   ErrorMsg;
};
```

可能出现的错误：

错误代码	错误提示	可能的原因
66	用户尚未登录	尚未登录

nRequestID：返回订阅主题请求的 ID，该 ID 由用户在订阅主题时指定。

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

6.1.10 OnRspQryTopic 方法

查询主题应答。当会员系统发出查询主题指令后，交易系统返回响应时，该方法会被调用。

函数原形：

```
void OnRspQryTopic (
    CFfexFtdcDisseminationField *pDissemination,
    CFfexFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast);
```

参数：

pDissemination：指向查询主题结构的地址，包含了要查询的主题和该主题报文个数。查询主题结构：

```
struct CFfexFtdcDisseminationField {
    ///序列系列号
    TFfexFtdcSequenceSeriesType SequenceSeries;
    ///序列号
    TFfexFtdcSequenceNoType SequenceNo;
};
```

pRspInfo：指向响应信息结构的地址。响应信息结构：

```
struct CFfexFtdcRspInfoField {
    ///错误代码
    TFfexFtdcErrorIDType    ErrorID;
    ///错误信息
    TFfexFtdcErrorMsgType   ErrorMsg;
};
```

可能出现的错误：

错误代码	错误提示	可能的原因
66	用户尚未登录	尚未登录

nRequestID：返回查询主题请求的 ID，该 ID 由用户在订阅主题时指定。

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

6.1.11 OnRspError 方法

针对用户请求的出错通知。

函数原型：

```
void OnRspError(
CFfexFtdcRspInfoField *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 CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
```

```
        TFfexFtdcErrorMsgType   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。

6.1.12 OnRspOrderInsert 方法

报单录入应答。当会员系统发出过报单录入指令后，交易后台返回响应时，该方法会被调用。

函数原型：

```
void OnRspOrderInsert(
CFfexFtdcInputOrderField *pInputOrder,
CFfexFtdcRspInfoField *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，和交易系统返回的报单编号。注意：结构中的某些字段与报单录入时不同，交易系统返回为空值。

输入报单结构：

```
struct CFfexFtdcInputOrderField
    {
        ///报单编号，该字段由交易系统返回。
        TFfexFtdcOrderSysIDType OrderSysID;
        ///会员代码，未使用[1]
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码，未使用
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码，未使用
        TFfexFtdcUserIDType UserID;
        ///合约代码，未使用
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///报单价格条件，未使用
        TFfexFtdcOrderPriceTypeType OrderPriceType;
        ///买卖方向，未使用
        TFfexFtdcDirectionType  Direction;
        ///组合开平标志，未使用
        TFfexFtdcCombOffsetFlagType CombOffsetFlag;
        ///组合投机套保标志，未使用
        TFfexFtdcCombHedgeFlagType  CombHedgeFlag;
        ///价格，未使用
        TFfexFtdcPriceType  LimitPrice;
        ///数量，未使用
        TFfexFtdcVolumeType VolumeTotalOriginal;
        ///有效期类型，未使用
```

[1] 为保持与后续的 V1.5 交易系统的版本兼容，保留此数据项，但其内容无含义。会员系统不应对其内容有任何假设。在网络底层通信的实现中，TraderAPI 通过压缩算法以基本抵消通信带宽的开销，但保持了协议和 TraderAPI 的兼容性和可扩展性。以下同。

```
        TFfexFtdcTimeConditionType  TimeCondition;
        ///GTD 日期，未使用
        TFfexFtdcDateType   GTDDate;
        ///成交量类型，未使用
        TFfexFtdcVolumeConditionType    VolumeCondition;
        ///最小成交量，未使用
        TFfexFtdcVolumeType MinVolume;
        ///触发条件，未使用
        TFfexFtdcContingentConditionType    ContingentCondition;
        ///止损价，未使用
        TFfexFtdcPriceType  StopPrice;
        ///强平原因，未使用
        TFfexFtdcForceCloseReasonType  ForceCloseReason;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType  OrderLocalID;
        ///自动挂起标志，未使用
        TFfexFtdcBoolType   IsAutoSuspend;
        ///业务单元，未使用
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
2	合约找不到	找不到报单中的合约
3	会员找不到	找不到报单中的会员
4	客户找不到	找不到报单中的客户
6	报单字段错误	报单中有字段值不合法（枚举值越界）或发现非强平单中设置了强平原因
12	重复的报单	报单中的本地报单号重复
15	客户没有在该会员开户	报单中的客户并没有在指定的会员中开户
16	IOC 需在连续交易阶段	在非连续交易阶段企图报入 IOC 单
17	GFA 需在集合竞价阶段	在非集合竞价阶段企图报入 GFA 的报单
18	市价单不能排队	市价单的时间条件不是 IOC
19	数量约束应在 IOC 单上	数量约束不是任意数量的报单的时间条件不是 IOC
20	GTD 报单过期了	GTD 报单中的 GTD 日期已经过期了
21	最小数量大于报单数量	有最小数量条件的报单的最小数量大于报单数量
22	交易所数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
23	结算组数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
26	当前状态禁止此项操作	合约交易状态不是连续交易、集合竞价报单或者集合竞价平衡
31	平仓时客户持仓不足	报入平仓报单时，客户的持仓不足
32	超出客户限仓	报入开仓报单时，超过了客户的投机限仓
34	超出会员限仓	报入开仓报单时，超过了会员限仓
35	找不到帐号	找不到报单使用的资金帐号
36	资金不足	资金帐号中没有足够的资金
37	不合法的数量	报单数量不是最小报单数量要求的正整数倍，或者超过最大报单数量
48	价格非最小单位的倍数	报单的价格不是合约的最小变动单位的整数倍
49	价格超出涨停板	报单的价格高于合约的涨停板
50	价格跌破跌停板	报单的价格低于合约的跌停板

51	没有交易权限	对指定合约、或者客户对指定的合约、或者交易员没有的交易权限
52	只能平仓	会员对指定合约、或者客户对指定的合约、或者交易员只有平仓的权限
53	没有此项交易角色	会员在指定合约上不具有该客户对应的交易角色
57	不能为其他会员操作	交易员给非其所属的会员进行操作
58	用户不匹配	报单中的交易员和登录时的交易员不匹配
66	用户尚未登录	用户尚未登录
78	GTD 报单没有设定日期	GTD 报单没有指定 GTD 日期
79	不被支持的报单类型	交易所不支持此种报单类型
83	止损单仅用于连续交易	在非连续交易阶段报入止损单
84	止损单需是 IOC 或 GFD	止损单时的时间条件既不是 IOC，也不是 GFD
95	止损报单需说明止损价	止损单没有指定止损价
96	保值额度不足	报入保值报单时，客户的套期保值额度不足
103	当日套保仓位不能平仓	套保仓位不应使用平今仓报单进行平仓
114	最优价单不能排队	最优价单的时间条件不是 IOC

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。

注意：

CFfexFtdcRspInfoField.ErrorID 为零（意味着本次报单录入成功），CFfexFtdcInputOrderField *pInputOrder 中仅报单编号（交易系统给出的系统编号）和本地报单编号有意义，用于交易系统与会员系统之间报单的关联。该报单的具体内容需要从私有流获得。

CFfexFtdcInputOrderField 各数据字段的说明请参阅 OnRtnOrder 方法。

6.1.13 OnRspOrderAction 方法

当前版本不开放报单修改功能。

报单操作应答。报单操作包括报单的撤销、报单的挂起、报单的激活、报单的修改。当客户端发出过报单操作指令后，交易后台返回响应时，该方法会被调用。

函数原型：

```
void OnRspOrderAction(
CFfexFtdcOrderActionField *pOrderAction,
CFfexFtdcRspInfoField *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，和交易系统返回的报单编号。

报单操作结构：

```
struct CFfexFtdcOrderActionField
    {
        ///报单编号，该字段由交易系统返回。
        TFfexFtdcOrderSysIDType OrderSysID;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType  OrderLocalID;
        ///报单操作标志
        TFfexFtdcActionFlagType ActionFlag;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///价格，未使用的保留字段，其值为空
        TFfexFtdcPriceType  LimitPrice;
        ///数量变化，未使用的保留字段，其值为空
        TFfexFtdcVolumeType VolumeChange;
        ///操作本地编号
        TFfexFtdcOrderLocalIDType  ActionLocalID;
        ///业务单元，未使用的保留字段，其值为空
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
3	会员找不到	报单操作中的会员找不到
4	客户找不到	报单操作中的客户找不到
8	报单操作字段错误	报单操作中有字段值不合法（枚举值越界）
15	客户没有在该会员开户	客户并没有在指定的会员中开户
22	交易所数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
23	结算组数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
24	报单找不到	找不到要操作的报单
26	当前状态禁止此项操作	对于激活操作，合约交易状态不是连续交易、集合竞价报单或者集合竞价平衡 对于其他操作，交易状态不是连续交易或者集合竞价报单
28	报单已经全部成交	报单已经全部成交了
29	报单已经撤销	报单已经撤销了
32	超出客户限仓	修改报单时，超过了客户的投机限仓
34	超出会员限仓	修改报单时，超过了会员限仓
35	找不到帐号	找不到应当使用的资金帐号
36	资金不足	资金帐号中没有足够的资金
37	不合法的数量	修改后报单的数量不是最小报单数量要求的正整数倍，或者超过最大报单数量
48	价格非最小单位的倍数	修改后报单的价格不是合约的最小变动单位的整数倍
49	价格超出涨停板	修改后报单的价格价格高于合约的涨停板
50	价格跌破跌停板	修改后报单的价格价格低于合约的跌停板

57	不能为其他会员操作	交易员给非其所属的会员进行操作
58	用户不匹配	报单操作中的交易员和登录时的交易员不匹配
66	用户尚未登录	用户尚未登录
76	报单已经被挂起	在报单挂起时，报单已经被挂起了
77	报单已经被激活	在报单激活时，报单已经被激活了
96	保值额度不足	修改后报单时，客户的套期保值额度不足
97	重复的操作	报单操作的本地操作号重复
99	不能为其他用户操作	非授权交易员操作同会员的其他交易员报入的报单

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。

6.1.14 OnRspQuoteInsert 方法

当前版本不开放本方法。

报价录入应答。当会员系统发出过报价录入指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQuoteInsert(
CFfexFtdcInputQuoteField *pInputQuote,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pInputQuote：指向输入报价结构的地址，包含了报价录入操作的输入数据，和交易系统返回的报价编号。

输入报价结构：

```
struct CFfexFtdcInputQuoteField
    {
        ///报价编号，该字段由交易后台返回。
        TFfexFtdcQuoteSysIDType QuoteSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///本地报价编号
        TFfexFtdcQuoteLocalIDType   QuoteLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType   BusinessUnit;
        ///买方组合开平标志
        TFfexFtdcCombOffsetFlagType BidCombOffsetFlag;
        ///买方组合套保标志
        TFfexFtdcCombHedgeFlagType  BidCombHedgeFlag;
        ///买方价格
```

```
        TFfexFtdcPriceType  BidPrice;
        ///卖方组合开平标志
        TFfexFtdcCombOffsetFlagTypeAskCombOffsetFlag;
        ///卖方组合套保标志
        TFfexFtdcCombHedgeFlagType  AskCombHedgeFlag;
        ///卖方价格
        TFfexFtdcPriceType  AskPrice;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
2	合约找不到	找不到报价中的合约
3	会员找不到	找不到报价中的会员
4	客户找不到	找不到报价中的客户
7	报价字段错误	报价中有字段值不合法（枚举值越界）或
13	重复的报价	报价中的本地报价号重复
15	客户没有在该会员开户	报价中的客户并没有在指定的会员中开户
22	交易所数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
23	结算组数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
26	当前状态禁止此项操作	合约交易状态不是连续交易、集合竞价报单或者集合竞价平衡
31	平仓时客户持仓不足	客户的持仓不足
32	超出客户限仓	该报价导致客户的投机持仓超过了限仓
34	超出会员限仓	该报价导致会员的持仓超过了限仓
35	找不到帐号	找不到报价使用的资金帐号
36	资金不足	资金帐号中没有足够的资金
37	不合法的数量	报价数量不是最小报单数量要求的正整数倍，或者超过最大报单数量
48	价格非最小单位的倍数	报价的价格不是合约的最小变动单位的整数倍
49	价格超出涨停板	报价的价格高于合约的涨停板
50	价格跌破跌停板	报价的价格低于合约的跌停板
51	没有交易权限	对指定合约、或者客户对指定的合约、或者交易员没有的交易权限
52	只能平仓	会员对指定合约、或者客户对指定的合约、或者交易员只有平仓的权限
53	没有此项交易角色	会员在指定合约上不具有该客户对应的交易角色
57	不能为其他会员操作	交易员给非其所属的会员进行操作
58	用户不匹配	报价中的交易员和登录时的交易员不匹配
66	用户尚未登录	用户尚未登录
79	不被支持的报单类型	交易所不支持此种报单类型
96	保值额度不足	报入保值报价时，客户的套期保值额度不足
103	当日套保仓位不能平仓	套保仓位不应使用平今仓报价进行平仓

nRequestID：返回用户报价录入操作请求的 ID，该 ID 由用户在报价录入时指定。

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

6.1.15 OnRspQuoteAction 方法

当前版本不开放本方法。

报价操作应答。包括报价的撤销、报价的挂起、报价的激活、报价的修改。当会员系统发出过报价操作指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQuoteAction(
CFfexFtdcQuoteActionField *pQuoteAction,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pQuoteAction：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，和交易系统返回的报价编号。

报价操作结构：

```
struct CFfexFtdcQuoteActionField
    {
        ///报价编号，该字段由交易系统返回。
        TFfexFtdcQuoteSysIDType QuoteSysID;
        ///本地报价编号
        TShfeFtdcOrderLocalIDType  QuoteLocalID;
        ///报单操作标志
        TFfexFtdcActionFlagType ActionFlag;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///操作本地编号
        TFfexFtdcOrderLocalIDType  ActionLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;

    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
3	会员找不到	报价操作中的会员找不到
4	客户找不到	报价操作中的客户找不到
8	报单操作字段错误	报价操作中有字段值不合法（枚举值越界）
15	客户没有在该会员开户	客户并没有在指定的会员中开户

22	交易所数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
23	结算组数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
25	报价找不到	找不到要操作的报价
26	当前状态禁止此项操作	对于激活操作，合约交易状态不是连续交易、集合竞价报单或者集合竞价平衡 对于其他操作，交易状态不是连续交易或者集合竞价报单
28	报单已经全部成交	报价派生的报单已经全部成交了
29	报单已经撤销	报价派生的报单已经撤销了
35	找不到帐号	找不到应当使用的资金帐号
36	资金不足	资金帐号中没有足够的资金
57	不能为其他会员操作	交易员给非其所属的会员进行操作
58	用户不匹配	报价操作中的交易员和登录时的交易员不匹配
66	用户尚未登录	用户尚未登录
70	报价已经被取消	报价已经被取消
97	重复的操作	报价操作的本地操作号重复
99	不能为其他用户操作	非授权交易员操作同会员的其他交易员报入的报价

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

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

6.1.16 OnRspExecOrderInsert 方法

当前版本不开放该方法。

执行宣告录入应答。当会员系统执行宣告录入后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspExecOrderInsert(
CFfexFtdcInputExecOrderField *pInputExecOrder,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID, bool bIsLast);
```

参数：

pInputExecOrder：指向宣告录入结构的地址。

输入执行宣告结构：

```
struct CFfexFtdcInputExecOrderField
    {
        ///合约编号
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///本地执行宣告编号
        TFfexFtdcOrderLocalIDType  ExecOrderLocalID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
```

```
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
2	合约找不到	找不到执行宣告中的合约
3	会员找不到	执行宣告中的会员找不到
4	客户找不到	执行宣告中的客户找不到
15	客户没有在该会员开户	执行宣告中的客户并没有在指定的会员中开户
22	交易所数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
23	结算组数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
26	当前状态禁止此项操作	合约交易状态处于收市状态
51	没有交易权限	对指定合约、或者客户对指定的合约、或者交易员没有的交易权限
53	没有此项交易角色	会员在指定合约上不具有该客户对应的交易角色
57	不能为其他会员操作	交易员给非其所属的会员进行操作
58	用户不匹配	执行宣告中的交易员和登录时的交易员不匹配
66	用户尚未登录	用户尚未登录
79	不被支持的报单类型	交易所不支持此种报单类型
89	执行宣告字段错误	执行宣告中有字段值不合法（枚举值越界）
91	重复的执行宣告	执行宣告中的本地执行宣告号重复
94	执行宣告只能用于期权	执行宣告中的合约是非期权合约

nRequestID：返回执行宣告录入请求的 ID，该 ID 由用户在执行宣告录入时指定。

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

6.1.17 OnRspExecOrderAction 方法

当前版本不开放本方法。

执行宣告操作应答。当客户端执行宣告操作后，交易后台返回响应时，该方法会被调用。

函数原型：

```
void OnRspExecOrderAction(
CFfexFtdcExecOrderActionField *pExecOrderAction,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pInputExecAction：指向宣告操作结构的地址。

宣告操作结构：

```
struct CFfexFtdcExecOrderActionField
    {
        ///执行宣告编号
        TFfexFtdcExecOrderSysIDType ExecOrderSysID;
        ///本地执行宣告编号
        TFfexFtdcOrderLocalIDType  ExecOrderLocalID;
        ///报单操作标志
        TFfexFtdcActionFlagType ActionFlag;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///操作本地编号
        TFfexFtdcOrderLocalIDType  ActionLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;

    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
2	合约找不到	找不到执行宣告中的合约
3	会员找不到	执行宣告中的会员找不到
4	客户找不到	执行宣告中的客户找不到
15	客户没有在该会员开户	执行宣告中的客户并没有在指定的会员中开户
22	交易所数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
23	结算组数据没有同步	交易系统尚未初始化完毕，可以稍后再重试
26	当前状态禁止此项操作	合约交易状态处于收市状态
51	没有交易权限	对指定合约、或者客户对指定的合约、或者交易员没有的交易权限
53	没有此项交易角色	会员在指定合约上不具有该客户对应的交易角色
57	不能为其他会员操作	交易员给非其所属的会员进行操作
58	用户不匹配	执行宣告中的交易员和登录时的交易员不匹配
66	用户尚未登录	用户尚未登录
79	不被支持的报单类型	交易所不支持此种报单类型
90	执行宣告操作字段错误	执行宣告操作中有字段值不合法（枚举值越界）
92	执行宣告已经取消	要操作的执行宣告已经取消
93	执行宣告找不到	要操作的执行宣告没有找到
97	重复的操作	执行宣告操作的本地操作号重复

nRequestID：返回执行宣告操作请求的 ID，该 ID 由用户在执行宣告操作时指定。

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

6.1.18 OnRspQryPartAccount 方法

会员资金查询应答。当客户端发出会员资金查询指令后，交易后台返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryPartAccount(
CFfexFtdcRspPartAccountField *pRspPartAccount,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pRspPartAccount：指向会员资金应答结构的地址。

会员资金应答结构：

```
struct CFfexFtdcRspPartAccountField
{
    ///交易日
    TFfexFtdcDateType  TradingDay;
    ///结算组代码
    TFfexFtdcSettlementGroupIDType SettlementGroupID;
    ///结算编号
    TFfexFtdcSettlementIDType  SettlementID;
    ///上次结算准备金
    TFfexFtdcMoneyType  PreBalance;
    ///当前保证金总额
    TFfexFtdcMoneyType  CurrMargin;
    ///平仓盈亏
    TFfexFtdcMoneyType  CloseProfit;
    ///期权权利金收支
    TFfexFtdcMoneyType  Premium;
    ///入金金额
    TFfexFtdcMoneyType  Deposit;
    ///出金金额
    TFfexFtdcMoneyType  Withdraw;
    ///期货结算准备金
    TFfexFtdcMoneyType  Balance;
    ///可提资金
    TFfexFtdcMoneyType  Available;
    ///资金帐号
    TFfexFtdcAccountIDType  AccountID;
    ///冻结的保证金
    TFfexFtdcMoneyType  FrozenMargin;
    ///冻结的权利金
    TFfexFtdcMoneyType  FrozenPremium;
    ///基本准备金
    TFfexFtdcMoneyType  BaseReserve;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
```

```
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

nRequestID：返回用户资金查询请求的 ID，该 ID 由用户在资金查询时指定。

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

6.1.19 OnRspQryOrder 方法

报单查询请求。当会员系统发出报单查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryOrder(
CFfexFtdcOrderField *pOrder,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pOrder：指向报单信息结构的地址。

报单信息结构：

```
struct CFfexFtdcOrderField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///报单编号
        TFfexFtdcOrderSysIDType OrderSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///报单价格条件
        TFfexFtdcOrderPriceTypeType OrderPriceType;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///组合开平标志
        TFfexFtdcCombOffsetFlagType CombOffsetFlag;
        ///组合投机套保标志
        TFfexFtdcCombHedgeFlagType  CombHedgeFlag;
        ///价格
```

```
        TFfexFtdcPriceType  LimitPrice;
        ///数量
        TFfexFtdcVolumeType VolumeTotalOriginal;
        ///有效期类型
        TFfexFtdcTimeConditionType  TimeCondition;
        ///GTD 日期，未使用
        TFfexFtdcDateType   GTDDate;
        ///成交量类型
        TFfexFtdcVolumeConditionType    VolumeCondition;
        ///最小成交量
        TFfexFtdcVolumeType MinVolume;
        ///触发条件
        TFfexFtdcContingentConditionType    ContingentCondition;
        ///止损价，未使用
        TFfexFtdcPriceType  StopPrice;
        ///强平原因
        TFfexFtdcForceCloseReasonType  ForceCloseReason;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType  OrderLocalID;
        ///自动挂起标志
        TFfexFtdcBoolType   IsAutoSuspend;
        ///报单来源
        TFfexFtdcOrderSourceType    OrderSource;
        ///报单状态
        TFfexFtdcOrderStatusType    OrderStatus;
        ///报单类型
        TFfexFtdcOrderTypeType  OrderType;
        ///今成交数量
        TFfexFtdcVolumeType VolumeTraded;
        ///剩余数量
        TFfexFtdcVolumeType VolumeTotal;
        ///报单日期
        TFfexFtdcDateType   InsertDate;
        ///插入时间
        TFfexFtdcTimeType   InsertTime;
        ///激活时间
        TFfexFtdcTimeType   ActiveTime;
        ///挂起时间
        TFfexFtdcTimeType   SuspendTime;
        ///最后修改时间
        TFfexFtdcTimeType   UpdateTime;
        ///撤销时间
        TFfexFtdcTimeType   CancelTime;
        ///最后修改交易用户代码
        TFfexFtdcUserIDType ActiveUserID;
        ///优先权，未使用
        TFfexFtdcPriorityType   Priority;
        ///按时间排队的序号，未使用
        TFfexFtdcTimeSortIDType TimeSortID;
        ///结算会员编号，未使用
        TFfexFtdcParticipantIDType  ClearingPartID;
        ///业务单元，未使用
        TFfexFtdcBusinessUnitType   BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
```

```
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

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。

6.1.20 OnRspQryQuote 方法

当前版本不开放本方法。

报价查询应答。当会员系统发出报价查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryQuote(
CFfexFtdcQuoteField *pQuote,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pQuote：指向报价信息结构的地址。

报价信息结构：

```
struct CFfexFtdcQuoteField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType  SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///报价编号
        TFfexFtdcQuoteSysIDType QuoteSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///本地报价编号
        TFfexFtdcQuoteLocalIDType   QuoteLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType   BusinessUnit;
        ///买方组合开平标志
        TFfexFtdcCombOffsetFlagType BidCombOffsetFlag;
```

```
        ///买方组合套保标志
        TFfexFtdcCombHedgeFlagType  BidCombHedgeFlag;
        ///买方价格
        TFfexFtdcPriceType  BidPrice;
        ///卖方组合开平标志
        TFfexFtdcCombOffsetFlagTypeAskCombOffsetFlag;
        ///卖方组合套保标志
        TFfexFtdcCombHedgeFlagType  AskCombHedgeFlag;
        ///卖方价格
        TFfexFtdcPriceType  AskPrice;
        ///插入时间
        TFfexFtdcTimeType   InsertTime;
        ///撤销时间
        TFfexFtdcTimeType   CancelTime;
        ///成交时间
        TFfexFtdcTimeType   TradeTime;
        ///买方报单编号
        TFfexFtdcOrderSysIDTypeBidOrderSysID;
        ///卖方报单编号
        TFfexFtdcOrderSysIDTypeAskOrderSysID;
        ///结算会员编号
        TFfexFtdcParticipantIDType ClearingPartID;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

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

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

6.1.21 OnRspQryTrade 方法

成交单查询应答。当会员系统发出成交单查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryTrade(
CFfexFtdcTradeField *pTrade,
CFfexFtdcRspInfoField *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 CFfexFtdcTradeField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType  SettlementID;
        ///成交编号
        TFfexFtdcTradeIDType    TradeID;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///报单编号
        TFfexFtdcOrderSysIDType OrderSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易角色
        TFfexFtdcTradingRoleType    TradingRole;
        ///资金帐号
        TFfexFtdcAccountIDType  AccountID;
        ///合约代码
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///开平标志
        TFfexFtdcOffsetFlagType OffsetFlag;
        ///投机套保标志
        TFfexFtdcHedgeFlagType  HedgeFlag;
        ///价格
        TFfexFtdcPriceType  Price;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///成交时间
        TFfexFtdcTimeType   TradeTime;
        ///成交类型
        TFfexFtdcTradeTypeType  TradeType;
        ///成交价来源
        TFfexFtdcPriceSourceType    PriceSource;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType  OrderLocalID;
        ///结算会员编号
        TFfexFtdcParticipantIDType ClearingPartID;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

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。

6.1.22 OnRspQryClient 方法

会员客户查询应答。当会员系统发出会员客户查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryClient(
CFfexFtdcRspClientField*pClient,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pClient：指向客户信息结构的地址。

客户信息结构：

```
struct CFfexFtdcRspClientField
{
 ///客户代码
 TFfexFtdcClientIDType  ClientID;
 ///客户名称
 TFfexFtdcPartyNameType  ClientName;
 ///证件类型
 TFfexFtdcIdCardTypeType IdentifiedCardType;
 ///原证件号码
 TFfexFtdcIdentifiedCardNoV1TypeUseLess;
 ///交易角色
 TFfexFtdcTradingRoleType   TradingRole;
 ///客户类型
 TFfexFtdcClientTypeType ClientType;
 ///是否活跃
 TFfexFtdcBoolType  IsActive;
 ///会员号
 TFfexFtdcParticipantIDType  ParticipantID;
 ///证件号码
 TFfexFtdcIdentifiedCardNoType  IdentifiedCardNo;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
```

```
    TFfexFtdcErrorMsgType   ErrorMsg;
};
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

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。

6.1.23 OnRspQryPartPosition 方法

会员持仓查询应答。当会员系统发出会员持仓查询指令后，后交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryPartPosition(
CFfexFtdcRspPartPositionField *pRspPartPosition,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pRspPartPosition：指向会员持仓应答结构的地址。

会员持仓应答结构：

```
struct CFfexFtdcRspPartPositionField
{
    ///交易日
    TFfexFtdcDateType   TradingDay;
    ///结算组代码
    TFfexFtdcSettlementGroupIDType SettlementGroupID;
    ///结算编号
    TFfexFtdcSettlementIDType   SettlementID;
    ///投机套保标志
    TFfexFtdcHedgeFlagType  HedgeFlag;
    ///持仓多空方向
    TFfexFtdcPosiDirectionType  PosiDirection;
    ///上日持仓
    TFfexFtdcVolumeType YdPosition;
    ///今日持仓
    TFfexFtdcVolumeType Position;
    ///多头冻结
    TFfexFtdcVolumeType LongFrozen;
    ///空头冻结
    TFfexFtdcVolumeType ShortFrozen;
    ///昨日多头冻结
    TFfexFtdcVolumeType YdLongFrozen;
    ///昨日空头冻结
    TFfexFtdcVolumeType YdShortFrozen;
    ///合约代码
    TFfexFtdcInstrumentIDType   InstrumentID;
```

```
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///交易角色
        TFfexFtdcTradingRoleType    TradingRole;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

nRequestID：返回会员持仓查询请求的 ID，该 ID 由用户在会员持仓查询时指定。

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

6.1.24 OnRspQryClientPosition 方法

客户持仓查询应答。当会员系统发出客户持仓查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryClientPosition(
CFfexFtdcRspClientPositionField *pRspClientPosition,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pRspClientPosition：指向客户持仓应答结构的地址。

客户持仓应答结构：

```
    struct CFfexFtdcRspClientPositionField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///投机套保标志
        TFfexFtdcHedgeFlagType  HedgeFlag;
        ///持仓多空方向
        TFfexFtdcPosiDirectionType  PosiDirection;
```

```
        ///上日持仓
        TFfexFtdcVolumeType YdPosition;
        ///今日持仓
        TFfexFtdcVolumeType Position;
        ///多头冻结
        TFfexFtdcVolumeType LongFrozen;
        ///空头冻结
        TFfexFtdcVolumeType ShortFrozen;
        ///昨日多头冻结
        TFfexFtdcVolumeType YdLongFrozen;
        ///昨日空头冻结
        TFfexFtdcVolumeType YdShortFrozen;
        ///当日买成交量
        TFfexFtdcVolumeType BuyTradeVolume;
        ///当日卖成交量
        TFfexFtdcVolumeType SellTradeVolume;
        ///持仓成本
        TFfexFtdcMoneyType  PositionCost;
        ///昨日持仓成本
        TFfexFtdcMoneyType  YdPositionCost;
        ///占用的保证金
        TFfexFtdcMoneyType  UseMargin;
        ///冻结的保证金
        TFfexFtdcMoneyType  FrozenMargin;
        ///多头冻结的保证金
        TFfexFtdcMoneyType  LongFrozenMargin;
        ///空头冻结的保证金
        TFfexFtdcMoneyType  ShortFrozenMargin;
        ///冻结的权利金
        TFfexFtdcMoneyType  FrozenPremium;
        ///合约代码
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///会员代码
        TFfexFtdcParticipantIDType ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

nRequestID：返回客户持仓查询请求的 ID，该 ID 由用户在客户持仓查询时指定。

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

6.1.25 OnRspQryInstrument 方法

合约查询应答。当会员系统发出合约查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryInstrument(
CFfexFtdcRspInstrumentField *pRspInstrument,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pRspInstrument：指向合约结构的地址。

合约结构：

```
struct CFfexFtdcRspInstrumentField
    {
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///产品代码
        TFfexFtdcProductIDType ProductID;
        ///产品组代码
        TFfexFtdcProductGroupIDType ProductGroupID;
        ///基础商品代码
        TFfexFtdcInstrumentIDType  UnderlyingInstrID;
        ///产品类型
        TFfexFtdcProductClassType  ProductClass;
        ///持仓类型
        TFfexFtdcPositionTypeType  PositionType;
        ///执行价
        TFfexFtdcPriceType StrikePrice;
        ///期权类型
        TFfexFtdcOptionsTypeType   OptionsType;
        ///合约数量乘数
        TFfexFtdcVolumeMultipleType VolumeMultiple;
        ///合约基础商品乘数
        TFfexFtdcUnderlyingMultipleTypeUnderlyingMultiple;
        ///合约代码
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///合约名称
        TFfexFtdcInstrumentNameType InstrumentName;
        ///交割年份
        TFfexFtdcYearType  DeliveryYear;
        ///交割月
        TFfexFtdcMonthType DeliveryMonth;
        ///提前月份
        TFfexFtdcAdvanceMonthType  AdvanceMonth;
        ///当前是否交易
        TFfexFtdcBoolType  IsTrading;
        ///创建日
        TFfexFtdcDateType  CreateDate;
        ///上市日
        TFfexFtdcDateType  OpenDate;
        ///到期日
        TFfexFtdcDateType  ExpireDate;
        ///开始交割日
```

```
        TFfexFtdcDateType   StartDelivDate;
        ///最后交割日
        TFfexFtdcDateType   EndDelivDate;
        ///挂牌基准价
        TFfexFtdcPriceType  BasisPrice;
        ///市价单最大下单量
        TFfexFtdcVolumeType MaxMarketOrderVolume;
        ///市价单最小下单量
        TFfexFtdcVolumeType MinMarketOrderVolume;
        ///限价单最大下单量
        TFfexFtdcVolumeType MaxLimitOrderVolume;
        ///限价单最小下单量
        TFfexFtdcVolumeType MinLimitOrderVolume;
        ///最小变动价位
        TFfexFtdcPriceType  PriceTick;
        ///交割月自然人开仓
        TFfexFtdcMonthCountType AllowDelivPersonOpen;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

nRequestID：返回合约查询请求的 ID，该 ID 由用户在合约查询时指定。

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

6.1.26 OnRspQryInstrumentStatus 方法

合约交易状态查询应答。当会员系统发出合约交易状态查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryInstrumentStatus(
CFfexFtdcInstrumentStatusField *pInstrumentStatus,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pInstrumentStatus：指向合约交易状态结构的地址。

合约交易状态结构：

```
struct CFfexFtdcInstrumentStatusField
{
    ///结算组代码
    TFfexFtdcSettlementGroupIDType SettlementGroupID;
    ///合约代码
    TFfexFtdcInstrumentIDType  InstrumentID;
    ///合约交易状态
    TFfexFtdcInstrumentStatusType  InstrumentStatus;
    ///交易阶段编号
    TFfexFtdcTradingSegmentSNType  TradingSegmentSN;
    ///进入本状态时间
    TFfexFtdcTimeType  EnterTime;
    ///进入本状态原因
    TFfexFtdcInstStatusEnterReasonType EnterReason;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

nRequestID：返回合约交易状态查询请求的 ID，该 ID 由用户在合约交易状态查询时指定。

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

6.1.27 OnRspQryBulletin 方法

交易所公告查询请求应答。当会员系统发出交易所公告查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryBulletin(
CFfexFtdcBulletinField *pBulletin,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pBulletin：指向交易所公告结构的地址。

交易所公告结构：

```
struct CFfexFtdcBulletinField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///公告编号
        TFfexFtdcBulletinIDType BulletinID;
```

```
        ///序列号
        TFfexFtdcSequenceNoType SequenceNo;
        ///公告类型
        TFfexFtdcNewsTypeType   NewsType;
        ///紧急程度
        TFfexFtdcNewsUrgencyType    NewsUrgency;
        ///发送时间
        TFfexFtdcTimeType   SendTime;
        ///消息摘要
        TFfexFtdcAbstractType   Abstract;
        ///消息来源
        TFfexFtdcComeFromType   ComeFrom;
        ///消息正文
        TFfexFtdcContentType    Content;
        ///WEB 地址
        TFfexFtdcURLLinkType    URLLink;
        ///市场代码
        TFfexFtdcMarketIDType   MarketID;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

nRequestID：返回交易所公告查询请求的 ID，该 ID 由用户在交易所公告查询时指定。

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

6.1.28 OnRspQryMarketData 方法

普通行情查询应答，当客户端发出过普通行情查询，交易后台返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryMarketData(
CFfexFtdcMarketDataField *pMarketData,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pMarketData：返回的市场行情信息的地址。

市场行情信息结构：

```
struct CFfexFtdcMarketDataField
    {
        ///交易日
```

```
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///最新价
        TFfexFtdcPriceType  LastPrice;
        ///昨结算
        TFfexFtdcPriceType  PreSettlementPrice;
        ///昨收盘
        TFfexFtdcPriceType  PreClosePrice;
        ///昨持仓量
        TFfexFtdcLargeVolumeType    PreOpenInterest;
        ///今开盘
        TFfexFtdcPriceType  OpenPrice;
        ///最高价
        TFfexFtdcPriceType  HighestPrice;
        ///最低价
        TFfexFtdcPriceType  LowestPrice;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///成交金额
        TFfexFtdcMoneyType  Turnover;
        ///持仓量
        TFfexFtdcLargeVolumeType    OpenInterest;
        ///今收盘
        TFfexFtdcPriceType  ClosePrice;
        ///今结算
        TFfexFtdcPriceType  SettlementPrice;
        ///涨停板价
        TFfexFtdcPriceType  UpperLimitPrice;
        ///跌停板价
        TFfexFtdcPriceType  LowerLimitPrice;
        ///昨虚实度
        TFfexFtdcRatioType  PreDelta;
        ///今虚实度
        TFfexFtdcRatioType  CurrDelta;
        ///最后修改时间
        TFfexFtdcTimeType   UpdateTime;
        ///最后修改毫秒
        TFfexFtdcMillisecType   UpdateMillisec;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
    };
```

pRspInfo：返回用户响应信息的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
            ///错误代码
            TFfexFtdcErrorIDType    ErrorID;
            ///错误信息
            TFfexFtdcErrorMsgType   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。

6.1.29 OnRspQryMBLMarketData 方法

合约价位查询应答。当会员系统合约价位查询指令后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void    OnRspQryMBLMarketData(
CFfexFtdcMBLMarketDataField *pMBLMarketData,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pMBLMarketData：指向分价表结构的地址。

分价表结构：

```
struct CFfexFtdcMBLMarketDataField
    {
        ///合约代码
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///价格
        TFfexFtdcPriceType  Price;
        ///数量
        TFfexFtdcVolumeType Volume;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

nRequestID：返回合约价位查询请求的 ID，该 ID 由用户在合约价位查询时指定。

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

6.1.30 OnRspQryHedgeVolume 方法

保值额度应答。当会员系统执行保值额度查询操作后，交易系统返回响应时，该方法会被调用。

函数原型：

```
void OnRspQryHedgeVolume(
CFfexFtdcHedgeVolumeField *pHedgeVolume,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pHedgeVolume：指向保值额度量结构的地址。

保值额度量结构：

```
struct CFfexFtdcHedgeVolumeField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType  SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///多头保值额度最初申请量，单位为手。
        TFfexFtdcVolumeType LongVolumeOriginal;
        ///空头保值额度最初申请量，单位为手。
        TFfexFtdcVolumeType ShortVolumeOriginal;
        ///多头保值额度，单位为手。
        TFfexFtdcVolumeType LongVolume;
        ///空头保值额度，单位为手。
        TFfexFtdcVolumeType ShortVolume;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

可能出现的错误：

错误代码	错误提示	可能的原因
80	用户无此权限	只能查询本会员下的情况
57	不能为其他会员操作	不能查询其它会员下的情况

nRequestID：返回执行保值额度查询的 ID，该 ID 由用户在执行保值额度查询时指定。

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

6.1.31 OnRtnTrade 方法

成交回报。当发生成交时交易系统会通知会员系统，该方法会被调用。

函数原型：

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

参数：

pTrade：指向成交信息结构的地址。注意：成交回报中的某些字段未使用，交易系统返回为空值。

成交信息结构：

```
struct CFfexFtdcTradeField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///成交编号
        TFfexFtdcTradeIDType    TradeID;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///报单编号
        TFfexFtdcOrderSysIDType OrderSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易角色，未使用
        TFfexFtdcTradingRoleType    TradingRole;
        ///资金帐号，未使用
        TFfexFtdcAccountIDType  AccountID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///开平标志
        TFfexFtdcOffsetFlagType OffsetFlag;
        ///投机套保标志
        TFfexFtdcHedgeFlagType  HedgeFlag;
        ///价格
        TFfexFtdcPriceType  Price;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///成交时间
        TFfexFtdcTimeType   TradeTime;
        ///成交类型，未使用
        TFfexFtdcTradeTypeType  TradeType;
        ///成交价来源，未使用
        TFfexFtdcPriceSourceType    PriceSource;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType   OrderLocalID;
        ///结算会员编号，未使用
        TFfexFtdcParticipantIDType  ClearingPartID;
        ///业务单元，未使用
```

```
        TFfexFtdcBusinessUnitType   BusinessUnit;
    };
```

6.1.32 OnRtnOrder 方法

报单回报。当会员系统进行报单录入、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，交易系统会主动通知客户端，该方法会被调用。

函数原型：

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

参数：

pOrder：指向报单信息结构的地址。注意：报单回报中的某些字段未使用，交易系统返回为空值。

报单信息结构：

```
struct CFfexFtdcOrderField
    {
        ///交易日，未使用
        TFfexFtdcDateType   TradingDay;
        ///结算组代码，未使用
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号，未使用
        TFfexFtdcSettlementIDType   SettlementID;
        ///报单编号
        TFfexFtdcOrderSysIDType OrderSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///报单价格条件
        TFfexFtdcOrderPriceTypeType OrderPriceType;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///组合开平标志
        TFfexFtdcCombOffsetFlagType CombOffsetFlag;
        ///组合投机套保标志
        TFfexFtdcCombHedgeFlagType  CombHedgeFlag;
        ///价格
        TFfexFtdcPriceType  LimitPrice;
        ///数量
        TFfexFtdcVolumeType VolumeTotalOriginal;
        ///有效期类型
        TFfexFtdcTimeConditionType  TimeCondition;
        ///GTD 日期
        TFfexFtdcDateType   GTDDate;
        ///成交量类型
        TFfexFtdcVolumeConditionType    VolumeCondition;
        ///最小成交量
        TFfexFtdcVolumeType MinVolume;
        ///触发条件
        TFfexFtdcContingentConditionType    ContingentCondition;
```

```
        ///止损价
        TFfexFtdcPriceType  StopPrice;
        ///强平原因
        TFfexFtdcForceCloseReasonType  ForceCloseReason;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType  OrderLocalID;
        ///自动挂起标志
        TFfexFtdcBoolType  IsAutoSuspend;
        ///报单来源，未使用
        TFfexFtdcOrderSourceType  OrderSource;
        ///报单状态
        TFfexFtdcOrderStatusType  OrderStatus;
        ///报单类型，未使用
        TFfexFtdcOrderTypeType  OrderType;
        ///今成交数量，未使用
        TFfexFtdcVolumeType VolumeTraded;
        ///剩余数量
        TFfexFtdcVolumeType VolumeTotal;
        ///报单日期
        TFfexFtdcDateType  InsertDate;
        ///插入时间，未使用
        TFfexFtdcTimeType  InsertTime;
        ///激活时间，未使用
        TFfexFtdcTimeType  ActiveTime;
        ///挂起时间，未使用
        TFfexFtdcTimeType  SuspendTime;
        ///最后修改时间
        TFfexFtdcTimeType  UpdateTime;
        ///撤销时间，未使用
        TFfexFtdcTimeType  CancelTime;
        ///最后修改交易用户代码
        TFfexFtdcUserIDType ActiveUserID;
        ///优先权，未使用
        TFfexFtdcPriorityType  Priority;
        ///按时间排队的序号，未使用
        TFfexFtdcTimeSortIDType TimeSortID;
        ///结算会员编号，未使用
        TFfexFtdcParticipantIDType ClearingPartID;
        ///业务单元，未使用
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

6.1.33 OnRtnQuote 方法

报价回报。当会员系统进行报价录入、报价操作导致报价状态发生变化时，由交易系统服务主动通知客户端，该方法会被调用。

函数原型：

```
void OnRtnQuote(CFfexFtdcQuoteField *pQuote);
```

参数：

pQuote：指向报价结构的地址。

报价结构：

```
struct CFfexFtdcQuoteField
    {
        ///交易日
```

```
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///报价编号
        TFfexFtdcQuoteSysIDType QuoteSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///本地报价编号
        TFfexFtdcOrderLocalIDType   QuoteLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType   BusinessUnit;
        ///买方组合开平标志
        TFfexFtdcCombOffsetFlagType BidCombOffsetFlag;
        ///买方组合套保标志
        TFfexFtdcCombHedgeFlagType  BidCombHedgeFlag;
        ///买方价格
        TFfexFtdcPriceType  BidPrice;
        ///卖方组合开平标志
        TFfexFtdcCombOffsetFlagType AskCombOffsetFlag;
        ///卖方组合套保标志
        TFfexFtdcCombHedgeFlagType  AskCombHedgeFlag;
        ///卖方价格
        TFfexFtdcPriceType  AskPrice;
        ///插入时间
        TFfexFtdcTimeType   InsertTime;
        ///撤销时间
        TFfexFtdcTimeType   CancelTime;
        ///成交时间
        TFfexFtdcTimeType   TradeTime;
        ///买方报单编号
        TFfexFtdcOrderSysIDType BidOrderSysID;
        ///卖方报单编号
        TFfexFtdcOrderSysIDType AskOrderSysID;
        ///结算会员编号
        TFfexFtdcParticipantIDType  ClearingPartID;

    };
```

6.1.34 OnRtnExecOrder 方法

执行宣告回报。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnRtnExecOrder(CFfexFtdcExecOrderField *pExecOrder);
```

参数：

pExecOrder：指向执行宣告结构的地址。

执行宣告结构：

```
struct CFfexFtdcExecOrderField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType   SettlementID;
        ///合约编号
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///本地执行宣告编号
        TFfexFtdcOrderLocalIDType   ExecOrderLocalID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///业务单元
        TFfexFtdcBusinessUnitType   BusinessUnit;
        ///执行宣告编号
        TFfexFtdcExecOrderSysIDType ExecOrderSysID;
        ///报单日期
        TFfexFtdcDateType   InsertDate;
        ///插入时间
        TFfexFtdcTimeType   InsertTime;
        ///撤销时间
        TFfexFtdcTimeType   CancelTime;
        ///执行结果
        TFfexFtdcExecResultType ExecResult;
        ///结算会员编号
        TFfexFtdcParticipantIDType  ClearingPartID;
    };
```

6.1.35 OnRtnInstrumentStatus 方法

合约交易状态改变回报。当合约交易状态发生变化时，交易系统会主动通知会员系统，该方法会被调用。

函数原型：

```
void OnRtnInstrumentStatus(CFfexFtdcInstrumentStatusField *pInstrumentStatus);
```

参数：

pInstrumentStatus：指向合约状态结构的地址。

合约状态结构：

```
struct CFfexFtdcInstrumentStatusField
    {
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///合约交易状态
        TFfexFtdcInstrumentStatusType   InstrumentStatus;
        ///交易阶段编号
```

```
        TFfexFtdcTradingSegmentSNType  TradingSegmentSN;
        ///进入本状态时间
        TFfexFtdcTimeType  EnterTime;
        ///进入本状态原因
        TFfexFtdcInstStatusEnterReasonType EnterReason;
    };
```

6.1.36 OnRtnInsInstrument 方法

增加合约通知。当会员系统登录成功后时，交易系统会把系统中的增加的合约通过公共流通知给客户端。

函数原型：

```
void OnRtnInsInstrument(CFfexFtdcInstrumentField *pInstrument);
```

参数：

pInstrument：指向合约结构的地址。

合约结构：

```
struct CFfexFtdcInstrumentField
    {
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///产品代码
        TFfexFtdcProductIDType  ProductID;
        ///产品组代码
        TFfexFtdcProductGroupIDType ProductGroupID;
        ///基础商品代码
        TFfexFtdcInstrumentIDType  UnderlyingInstrID;
        ///产品类型
        TFfexFtdcProductClassType  ProductClass;
        ///持仓类型
        TFfexFtdcPositionTypeType  PositionType;
        ///执行价
        TFfexFtdcPriceType  StrikePrice;
        ///期权类型
        TFfexFtdcOptionsTypeType   OptionsType;
        ///合约数量乘数
        TFfexFtdcVolumeMultipleType VolumeMultiple;
        ///合约基础商品乘数
        TFfexFtdcUnderlyingMultipleType UnderlyingMultiple;
        ///合约代码
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///合约名称
        TFfexFtdcInstrumentNameType InstrumentName;
        ///交割年份
        TFfexFtdcYearType  DeliveryYear;
        ///交割月
        TFfexFtdcMonthType  DeliveryMonth;
        ///提前月份
        TFfexFtdcAdvanceMonthType  AdvanceMonth;
        ///当前是否交易
        TFfexFtdcBoolType  IsTrading;
    };
```

6.1.37 OnRtnDelInstrument 方法

删除合约通知。当会员系统登录成功后时，交易系统会把系统中删除的合约通过公共流通知给会员系统。

函数原型：

```
void OnRtnDelInstrument(CFfexFtdcInstrumentField *pInstrument);
```

参数：

pInstrument：指向合约结构的地址。

合约结构：

```
struct CFfexFtdcInstrumentField
    {
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///产品代码
        TFfexFtdcProductIDType ProductID;
        ///产品组代码
        TFfexFtdcProductGroupIDType ProductGroupID;
        ///基础商品代码
        TFfexFtdcInstrumentIDType  UnderlyingInstrID;
        ///产品类型
        TFfexFtdcProductClassType  ProductClass;
        ///持仓类型
        TFfexFtdcPositionTypeType  PositionType;
        ///执行价
        TFfexFtdcPriceType StrikePrice;
        ///期权类型
        TFfexFtdcOptionsTypeType   OptionsType;
        ///合约数量乘数
        TFfexFtdcVolumeMultipleType VolumeMultiple;
        ///合约基础商品乘数
        TFfexFtdcUnderlyingMultipleType UnderlyingMultiple;
        ///合约代码
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///合约名称
        TFfexFtdcInstrumentNameType InstrumentName;
        ///交割年份
        TFfexFtdcYearType  DeliveryYear;
        ///交割月
        TFfexFtdcMonthType DeliveryMonth;
        ///提前月份
        TFfexFtdcAdvanceMonthType  AdvanceMonth;
        ///当前是否交易
        TFfexFtdcBoolType  IsTrading;
    };
```

6.1.38 OnRtnInsCombinationLeg 方法

当前版本不支持该方法。

增加合约单腿通知。会员系统登录成功后时，交易系统会把系统中的增加组合合约的单腿通过公共流通知给会员系统。

函数原型：

```
void OnRtnInsCombinationLeg(CFfexFtdcCombinationLegField *pCombinationLeg);
```

参数：

pCombinationLeg：指向组合交易合约单腿结构的地址。

组合交易合约单腿结构：

```
struct CFfexFtdcCombinationLegField
    {
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///组合合约代码
        TFfexFtdcInstrumentIDType  CombInstrumentID;
        ///单腿编号
        TFfexFtdcLegIDType  LegID;
        ///单腿合约代码
        TFfexFtdcInstrumentIDType  LegInstrumentID;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///单腿乘数
        TFfexFtdcLegMultipleType   LegMultiple;
        ///推导层数
        TFfexFtdcImplyLevelType ImplyLevel;
    };
```

6.1.39 OnRtnDelCombinationLeg 方法

当前版本不支持该方法。

删除合约单腿通知。会员系统登录成功后时，交易系统会把系统中的删除组合合约的单腿通过公共流通知给会员系统。

函数原型：

```
void OnRtnDelCombinationLeg(CFfexFtdcCombinationLegField *pCombinationLeg);
```

参数：

pCombinationLeg：指向组合交易合约单腿结构的地址。

组合交易合约单腿结构：

```
struct CFfexFtdcCombinationLegField
    {
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///组合合约代码
        TFfexFtdcInstrumentIDType  CombInstrumentID;
        ///单腿编号
        TFfexFtdcLegIDType  LegID;
        ///单腿合约代码
        TFfexFtdcInstrumentIDType  LegInstrumentID;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///单腿乘数
        TFfexFtdcLegMultipleType   LegMultiple;
        ///推导层数
```

```
        TFfexFtdcImplyLevelType ImplyLevel;
    };
```

6.1.40 OnRtnBulletin 方法

公告通知。交易所通过交易系统发布公告时，由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnRtnBulletin(CFfexFtdcBulletinField *pBulletin);
```

参数：

pBulletin：指向公告结构的地址。

公告结构：

```
struct CFfexFtdcBulletinField
    {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///公告编号
        TFfexFtdcBulletinIDType BulletinID;
        ///序列号
        TFfexFtdcSequenceNoType SequenceNo;
        ///公告类型
        TFfexFtdcNewsTypeType   NewsType;
        ///紧急程度
        TFfexFtdcNewsUrgencyType    NewsUrgency;
        ///发送时间
        TFfexFtdcTimeType   SendTime;
        ///消息摘要
        TFfexFtdcAbstractType   Abstract;
        ///消息来源
        TFfexFtdcComeFromType   ComeFrom;
        ///消息正文
        TFfexFtdcContentType    Content;
        ///WEB 地址
        TFfexFtdcURLLinkType    URLLink;
        ///市场代码
        TFfexFtdcMarketIDType   MarketID;
    };
```

6.1.41 OnRtnAliasDefine 方法

别名定义通知。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnRtnAliasDefine(CFfexFtdcAliasDefineField *pAliasDefine);
```

参数：

pAliasDefine：指向别名定义结构的地址。

别名定义结构：

```
struct CFfexFtdcAliasDefineField
{
    ///起始位置
    TFfexFtdcStartPosType  StartPos;
    ///别名
    TFfexFtdcAliasType  Alias;
    ///原文
    TFfexFtdcOriginalTextType  OriginalText;
};
```

6.1.42 OnRtnFlowMessageCancel 方法

数据流回退通知。由交易系统发生灾备切换之后，用户重新登录交易系统并订阅某个数据流（私有流或公共流）时，交易系统主动通知会员系统该数据流某些报文被取消了，该方法会被调用。

函数原型：

```
void OnRtnFlowMessageCancel(
CFfexFtdcFlowMessageCancelField *pFlowMessageCancel);
```

参数：

pFlowMessageCancel：指向数据流回退结构的地址。别名定义结构：

```
struct CFfexFtdcFlowMessageCancelField
{
    ///序列系列号
    TFfexFtdcSequenceSeriesType    SequenceSeries;
    ///交易日
    TFfexFtdcDateType  TradingDay;
    ///数据中心代码
    TFfexFtdcDataCenterIDType  DataCenterID;
    ///回退起始序列号
    TFfexFtdcSequenceNoType StartSequenceNo;
    ///回退结束序列号
    TFfexFtdcSequenceNoType    EndSequenceNo;
};
SequenceSeries：发生回退的数据流代码（私有流或公共流）
回退的报文区间为：（StartSequenceNo，EndSequenceNo]
```

6.1.43 OnErrRtnOrderInsert 方法

报单录入错误回报。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnErrRtnOrderInsert(
    CFfexFtdcInputOrderField *pInputOrder,
    CFfexFtdcRspInfoField *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 CFfexFtdcInputOrderField
    {
        ///报单编号，该字段由交易后台返回。
        TFfexFtdcOrderSysIDType OrderSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///报单价格条件
        TFfexFtdcOrderPriceTypeType OrderPriceType;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///组合开平标志
        TFfexFtdcCombOffsetFlagType CombOffsetFlag;
        ///组合投机套保标志
        TFfexFtdcCombHedgeFlagType  CombHedgeFlag;
        ///价格
        TFfexFtdcPriceType  LimitPrice;
        ///数量
        TFfexFtdcVolumeType VolumeTotalOriginal;
        ///有效期类型
        TFfexFtdcTimeConditionType  TimeCondition;
        ///GTD 日期
        TFfexFtdcDateType   GTDDate;
        ///成交量类型
        TFfexFtdcVolumeConditionType    VolumeCondition;
        ///最小成交量
        TFfexFtdcVolumeType MinVolume;
        ///触发条件
        TFfexFtdcContingentConditionType    ContingentCondition;
        ///止损价
        TFfexFtdcPriceType  StopPrice;
        ///强平原因
        TFfexFtdcForceCloseReasonType   ForceCloseReason;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType   OrderLocalID;
        ///自动挂起标志
        TFfexFtdcBoolType   IsAutoSuspend;
        ///业务单元
        TFfexFtdcBusinessUnitType   BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

6.1.44 OnErrRtnOrderAction 方法

报单操作错误回报。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnErrRtnOrderAction (
     CFfexFtdcOrderActionField *pOrderAction,
     CFfexFtdcRspInfoField *pRspInfo);
```

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，和交易系统返回的报单编号。

报单操作结构：

```
struct CFfexFtdcOrderActionField
    {
        ///报单编号，该字段由交易后台返回。
        TFfexFtdcOrderSysIDType OrderSysID;
        ///报单操作标志
        TFfexFtdcActionFlagType ActionFlag;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///价格
        TFfexFtdcPriceType  LimitPrice;
        ///数量变化
        TFfexFtdcVolumeType VolumeChange;
        ///操作本地编号
        TFfexFtdcOrderLocalIDType   ActionLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType   BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

6.1.45 OnErrRtnQuoteInsert 方法

当前版本不支持该方法。

报价录入错误回报。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnErrRtnQuoteInsert (
     CFfexFtdcInputQuoteField *pInputQuote,
     CFfexFtdcRspInfoField *pRspInfo);
```

参数：

pInputQuote：指向输入报价结构的地址，包含了报价录入操作的输入数据，和交易系统返回的报价编号。

输入报价结构：

```
struct CFfexFtdcInputQuoteField
{
    ///报价编号，该字段由交易后台返回。
    TFfexFtdcQuoteSysIDType QuoteSysID;
    ///会员代码
    TFfexFtdcParticipantIDType  ParticipantID;
    ///客户代码
    TFfexFtdcClientIDType   ClientID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///数量
    TFfexFtdcVolumeType Volume;
    ///合约代码
    TFfexFtdcInstrumentIDType   InstrumentID;
    ///本地报价编号
    TFfexFtdcQuoteLocalIDType   QuoteLocalID;
    ///业务单元
    TFfexFtdcBusinessUnitType   BusinessUnit;
    ///买方组合开平标志
    TFfexFtdcCombOffsetFlagType BidCombOffsetFlag;
    ///买方组合套保标志
    TFfexFtdcCombHedgeFlagType  BidCombHedgeFlag;
    ///买方价格
    TFfexFtdcPriceType  BidPrice;
    ///卖方组合开平标志
    TFfexFtdcCombOffsetFlagType AskCombOffsetFlag;
    ///卖方组合套保标志
    TFfexFtdcCombHedgeFlagType  AskCombHedgeFlag;
    ///卖方价格
    TFfexFtdcPriceType  AskPrice;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

6.1.46 OnErrRtnQuoteAction 方法

当前版本不支持该方法。

报价操作错误回报。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnErrRtnQuoteAction (
     CFfexFtdcQuoteActionField *pQuoteAction,
     CFfexFtdcRspInfoField *pRspInfo);
```

参数：

pQuoteAction：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，和交易系统返回的报价编号。

报价操作结构：

```
struct CFfexFtdcQuoteActionField
{
    ///报价编号，该字段由交易后台返回。
    TFfexFtdcQuoteSysIDType QuoteSysID;
    ///本地报价编号
    TFfexFtdcOrderLocalIDType QuoteLocalID;
    ///报单操作标志
    TFfexFtdcActionFlagType ActionFlag;
    ///会员代码
    TFfexFtdcParticipantIDType ParticipantID;
    ///客户代码
    TFfexFtdcClientIDType  ClientID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///操作本地编号
    TFfexFtdcOrderLocalIDType  ActionLocalID;
    ///业务单元
    TFfexFtdcBusinessUnitType  BusinessUnit;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
{
    ///错误代码
    TFfexFtdcErrorIDType    ErrorID;
    ///错误信息
    TFfexFtdcErrorMsgType   ErrorMsg;
};
```

6.1.47 OnErrRtnExecOrderInsert 方法

当前版本不支持该方法。

执行宣告录入错误回报。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnErrRtnExecOrderInsert (
     CFfexFtdcInputExecOrderField *pInputExecOrder,
     CFfexFtdcRspInfoField *pRspInfo);
```

参数：

pInputExecOrder：指向宣告录入结构的地址。

输入执行宣告结构：

```
struct CFfexFtdcInputExecOrderField
    {
        ///合约编号
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///本地执行宣告编号
        TFfexFtdcOrderLocalIDType  ExecOrderLocalID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
    struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType   ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType  ErrorMsg;
    };
```

6.1.48 OnErrRtnExecOrderAction 方法

当前版本不支持该方法。

执行宣告操作错误回报。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnErrRtnExecOrderAction (
     CFfexFtdcExecOrderActionField *pExecOrderAction,
     CFfexFtdcRspInfoField *pRspInfo);
```

参数：

pInputExecAction：指向宣告操作结构的地址。

宣告操作结构：

```
struct CFfexFtdcExecOrderActionField
    {
        ///执行宣告编号
        TFfexFtdcExecOrderSysIDType ExecOrderSysID;
        ///本地执行宣告编号
        TFfexFtdcOrderLocalIDType  ExecOrderLocalID;
        ///报单操作标志
        TFfexFtdcActionFlagType ActionFlag;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
```

```
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///操作本地编号
        TFfexFtdcOrderLocalIDType  ActionLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
    {
        ///错误代码
        TFfexFtdcErrorIDType    ErrorID;
        ///错误信息
        TFfexFtdcErrorMsgType   ErrorMsg;
    };
```

6.1.49 OnRspQryCombOrder 方法

当前版本不支持该方法。

非常规组合报单查询应答。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
    void OnRspCombOrderInsert (
        CShfeFtdcCombOrderField *pCombOrder,
        CShfeFtdcRspInfoField *pRspInfo,
        int nRequestID,
        bool bIsLast);
```

参数：

pCombOrder：指向非常规组合报单结构的地址，非常规组合报单结构：

```
    struct CFfexFtdcCombOrderField {
        ///交易日
        TFfexFtdcDateType   TradingDay;
        ///结算组代码
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///结算编号
        TFfexFtdcSettlementIDType  SettlementID;
        ///组合报单编号
        TFfexFtdcOrderSysIDType CombOrderSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///价格
        TFfexFtdcPriceType  LimitPrice;
        ///数量
        TFfexFtdcVolumeType VolumeTotalOriginal;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType  CombOrderLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
        ///合约代码 1
```

```
    TFfexFtdcInstrumentIDType  InstrumentID1;
    ///买卖方向1
    TFfexFtdcDirectionType  Direction1;
    ///分腿乘数1
    TFfexFtdcLegMultipleType   LegMultiple1;
    ///开平标志1
    TFfexFtdcOffsetFlagType OffsetFlag1;
    ///投机套保标志1
    TFfexFtdcHedgeFlagType  HedgeFlag1;
    ///合约代码2
    TFfexFtdcInstrumentIDType  InstrumentID2;
    ///买卖方向2
    TFfexFtdcDirectionType  Direction2;
    ///分腿乘数2
    TFfexFtdcLegMultipleType   LegMultiple2;
    ///开平标志2
    TFfexFtdcOffsetFlagType OffsetFlag2;
    ///投机套保标志2
    TFfexFtdcHedgeFlagType  HedgeFlag2;
    ///合约代码3
    TFfexFtdcInstrumentIDType  InstrumentID3;
    ///买卖方向3
    TFfexFtdcDirectionType  Direction3;
    ///分腿乘数3
    TFfexFtdcLegMultipleType   LegMultiple3;
    ///开平标志3
    TFfexFtdcOffsetFlagType OffsetFlag3;
    ///投机套保标志3
    TFfexFtdcHedgeFlagType  HedgeFlag3;
    ///合约代码4
    TFfexFtdcInstrumentIDType  InstrumentID4;
    ///买卖方向4
    TFfexFtdcDirectionType  Direction4;
    ///分腿乘数4
    TFfexFtdcLegMultipleType   LegMultiple4;
    ///开平标志4
    TFfexFtdcOffsetFlagType OffsetFlag4;
    ///投机套保标志4
    TFfexFtdcHedgeFlagType  HedgeFlag4;
    ///报单来源
    TFfexFtdcOrderSourceType   OrderSource;
    ///今成交数量
    TFfexFtdcVolumeType VolumeTraded;
    ///剩余数量
    TFfexFtdcVolumeType VolumeTotal;
    ///报单日期
    TFfexFtdcDateType  InsertDate;
    ///插入时间
    TFfexFtdcTimeType  InsertTime;
    ///结算会员编号
    TFfexFtdcParticipantIDType  ClearingPartID;
};
```

pRspInfo：指向响应信息结构的地址。响应信息结构：

```
struct CFfexFtdcRspInfoField {
    ///错误代码
    TFfexFtdcErrorIDType    ErrorID;
    ///错误信息
    TFfexFtdcErrorMsgType  ErrorMsg;
};
```

nRequestID：非常规组合报单查询请求的 ID，该 ID 由用户指定，管理。

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

6.1.50 OnRtnCombOrder 方法

当前版本不支持该方法。

非常规组合报单回报。由交易系统主动通知会员系统，该方法会被调用。

函数原形：

```
void OnRtnCombOrder (CShfeFtdcCombOrderField *pCombOrder);
```

参数：

pCombOrder：指向非常规组合报单结构的地址，非常规组合报单结构：

```
struct CFfexFtdcCombOrderField {
    ///交易日
    TFfexFtdcDateType  TradingDay;
    ///结算组代码
    TFfexFtdcSettlementGroupIDType SettlementGroupID;
    ///结算编号
    TFfexFtdcSettlementIDType  SettlementID;
    ///组合报单编号
    TFfexFtdcOrderSysIDType CombOrderSysID;
    ///会员代码
    TFfexFtdcParticipantIDType ParticipantID;
    ///客户代码
    TFfexFtdcClientIDType  ClientID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///价格
    TFfexFtdcPriceType  LimitPrice;
    ///数量
    TFfexFtdcVolumeType VolumeTotalOriginal;
    ///本地报单编号
    TFfexFtdcOrderLocalIDType  CombOrderLocalID;
    ///业务单元
    TFfexFtdcBusinessUnitType  BusinessUnit;
    ///合约代码 1
    TFfexFtdcInstrumentIDType  InstrumentID1;
    ///买卖方向 1
    TFfexFtdcDirectionType  Direction1;
    ///分腿乘数 1
    TFfexFtdcLegMultipleType   LegMultiple1;
    ///开平标志 1
    TFfexFtdcOffsetFlagType OffsetFlag1;
    ///投机套保标志 1
    TFfexFtdcHedgeFlagType  HedgeFlag1;
    ///合约代码 2
    TFfexFtdcInstrumentIDType  InstrumentID2;
    ///买卖方向 2
    TFfexFtdcDirectionType  Direction2;
    ///分腿乘数 2
    TFfexFtdcLegMultipleType   LegMultiple2;
    ///开平标志 2
    TFfexFtdcOffsetFlagType OffsetFlag2;
    ///投机套保标志 2
```

```
    TFfexFtdcHedgeFlagType  HedgeFlag2;
    ///合约代码 3
    TFfexFtdcInstrumentIDType  InstrumentID3;
    ///买卖方向 3
    TFfexFtdcDirectionType  Direction3;
    ///分腿乘数 3
    TFfexFtdcLegMultipleType   LegMultiple3;
    ///开平标志 3
    TFfexFtdcOffsetFlagType OffsetFlag3;
    ///投机套保标志 3
    TFfexFtdcHedgeFlagType  HedgeFlag3;
    ///合约代码 4
    TFfexFtdcInstrumentIDType  InstrumentID4;
    ///买卖方向 4
    TFfexFtdcDirectionType  Direction4;
    ///分腿乘数 4
    TFfexFtdcLegMultipleType   LegMultiple4;
    ///开平标志 4
    TFfexFtdcOffsetFlagType OffsetFlag4;
    ///投机套保标志 4
    TFfexFtdcHedgeFlagType  HedgeFlag4;
    ///报单来源
    TFfexFtdcOrderSourceType   OrderSource;
    ///今成交数量
    TFfexFtdcVolumeType VolumeTraded;
    ///剩余数量
    TFfexFtdcVolumeType VolumeTotal;
    ///报单日期
    TFfexFtdcDateType  InsertDate;
    ///插入时间
    TFfexFtdcTimeType  InsertTime;
    ///结算会员编号
    TFfexFtdcParticipantIDType  ClearingPartID;
};
```

6.1.51 OnErrRtnCombOrderInsert 方法

当前版本不支持该方法。

报单录入组合错误回报。由交易系统主动通知会员系统，该方法会被调用。

函数原形：

```
void OnErrRtnCombOrderInsert (
    CFfexFtdcInputCombOrderField *pInputCombOrder,
    CFfexFtdcRspInfoField *pRspInfo);
```

参数：

pInputCombOrder：指向非常规组合报单录入结构的地址，非常规组合报单录入结构：

```
struct CFfexFtdcInputCombOrderField {
    ///组合报单编号
    TFfexFtdcOrderSysIDType CombOrderSysID;
    ///会员代码
    TFfexFtdcParticipantIDType  ParticipantID;
    ///客户代码
```

```
    TFfexFtdcClientIDType  ClientID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///价格
    TFfexFtdcPriceType  LimitPrice;
    ///数量
    TFfexFtdcVolumeType VolumeTotalOriginal;
    ///本地报单编号
    TFfexFtdcOrderLocalIDType  CombOrderLocalID;
    ///业务单元
    TFfexFtdcBusinessUnitType  BusinessUnit;
    ///合约代码 1
    TFfexFtdcInstrumentIDType  InstrumentID1;
    ///买卖方向 1
    TFfexFtdcDirectionType  Direction1;
    ///分腿乘数 1
    TFfexFtdcLegMultipleType   LegMultiple1;
    ///开平标志 1
    TFfexFtdcOffsetFlagType OffsetFlag1;
    ///投机套保标志 1
    TFfexFtdcHedgeFlagType  HedgeFlag1;
    ///合约代码 2
    TFfexFtdcInstrumentIDType  InstrumentID2;
    ///买卖方向 2
    TFfexFtdcDirectionType  Direction2;
    ///分腿乘数 2
    TFfexFtdcLegMultipleType   LegMultiple2;
    ///开平标志 2
    TFfexFtdcOffsetFlagType OffsetFlag2;
    ///投机套保标志 2
    TFfexFtdcHedgeFlagType  HedgeFlag2;
    ///合约代码 3
    TFfexFtdcInstrumentIDType  InstrumentID3;
    ///买卖方向 3
    TFfexFtdcDirectionType  Direction3;
    ///分腿乘数 3
    TFfexFtdcLegMultipleType   LegMultiple3;
    ///开平标志 3
    TFfexFtdcOffsetFlagType OffsetFlag3;
    ///投机套保标志 3
    TFfexFtdcHedgeFlagType  HedgeFlag3;
    ///合约代码 4
    TFfexFtdcInstrumentIDType  InstrumentID4;
    ///买卖方向 4
    TFfexFtdcDirectionType  Direction4;
    ///分腿乘数 4
    TFfexFtdcLegMultipleType   LegMultiple4;
    ///开平标志 4
    TFfexFtdcOffsetFlagType OffsetFlag4;
    ///投机套保标志 4
    TFfexFtdcHedgeFlagType  HedgeFlag4;
};
```

pRspInfo：指向响应信息结构的地址。 响应信息结构：

```
struct CFfexFtdcRspInfoField {
    ///错误代码
    TFfexFtdcErrorIDType    ErrorID;
    ///错误信息
    TFfexFtdcErrorMsgType   ErrorMsg;
};
```

6.1.52 OnRspAdminOrderInsert 方法

管理报单录入应答。由交易系统主动通知会员系统，该方法会被调用。

函数原型：

```
void OnRspAdminOrderInsert(
CFfexFtdcInputAdminOrderField *pInputAdminOrder,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pInputAdminOrder:指向输入管理员报单结构的地址。

输入管理员报单结构：

```
struct CFfexFtdcInputAdminOrderField
{
///合约代码
TFfexFtdcInstrumentIDType  InstrumentID;
///管理报单命令
TFfexFtdcAdminOrderCommandFlagType AdminOrderCommand;
///结算会员编号
TFfexFtdcParticipantIDType ClearingPartID;
///交易会员编号
TFfexFtdcParticipantIDType ParticipantID;
///金额
TFfexFtdcMoneyType  Amount;
///结算组代码
TFfexFtdcSettlementGroupIDType SettlementGroupID;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
{
    ///错误代码
    TFfexFtdcErrorIDType    ErrorID;
    ///错误信息
    TFfexFtdcErrorMsgType   ErrorMsg;
};
```

nRequestID：输入管理员报单请求的 ID，该 ID 由用户指定，管理。

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

6.1.53 OnRspQryCreditLimit 方法

信用额度查询应答。由交易后台主动通知会员系统，该方法会被调用。

函数原型：

```
void OnRspQryCreditLimit(
CFfexFtdcCreditLimitField *pCreditLimit,
CFfexFtdcRspInfoField *pRspInfo,
int nRequestID,
bool bIsLast);
```

参数：

pQryCreaditLimit：指向信用额度结构的地址

信用额度结构

```
struct CFfexFtdcCreditLimitField
{
///交易日
TFfexFtdcDateType  TradingDay;
///结算组代码
TFfexFtdcSettlementGroupIDType SettlementGroupID;
///结算编号
TFfexFtdcSettlementIDType  SettlementID;
///上次结算准备金
TFfexFtdcMoneyType  PreBalance;
///当前保证金总额
TFfexFtdcMoneyType  CurrMargin;
///平仓盈亏
TFfexFtdcMoneyType  CloseProfit;
///期权权利金收支
TFfexFtdcMoneyType  Premium;
///入金金额
TFfexFtdcMoneyType  Deposit;
///出金金额
TFfexFtdcMoneyType  Withdraw;
///期货结算准备金
TFfexFtdcMoneyType  Balance;
///可提资金
TFfexFtdcMoneyType  Available;
///交易会员编号
TFfexFtdcParticipantIDType  ParticipantID;
///结算会员编号
TFfexFtdcParticipantIDType  ClearingPartID;
///冻结的保证金
TFfexFtdcMoneyType  FrozenMargin;
///冻结的权利金
TFfexFtdcMoneyType  FrozenPremium;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CFfexFtdcRspInfoField
{
    ///错误代码
    TFfexFtdcErrorIDType    ErrorID;
    ///错误信息
    TFfexFtdcErrorMsgType   ErrorMsg;
};
```

nRequestID：信用额度查询请求的 ID，该 ID 由用户指定，管理。

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

6.2 CFfexFtdcTraderApi 接口

CFfexFtdcTraderApi 接口提供给用户的功能包括，报单与报价的录入、报单与报价的撤销、报单与报价的挂起、报单与报价的激活、报单与报价的修改、报单与报价的查询、成交单查询、会员客户查询、会员持仓查询、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。

系统对每个席位发送的指令速度（每秒发送的指令条数）有限制，超过限额，发送的指令将会被阻塞在网络上。具体限额数，请咨询交易所相关部门。

6.2.1 CreateFtdcTraderApi 方法

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

函数原型：

```
static CFfexFtdcTradeApi *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交易后台发布消息的状态。 默认值代表当前目录。

返回值：

返回一个指向 CFfexFtdcTradeApi 实例的指针。

6.2.2 GetVersion 方法

获取 API 版本号。

函数原型：

```
    const char *GetVersion(int &nMajorVersion, int &nMinorVersion) ;
```

参数：

nMajorVersion：返回主版本号

nMajorVersion：返回次版本号

返回值：

返回一个指向版本标识字符串的常量指针。

6.2.3 Release 方法

释放一个 CFfexFtdcTradeApi 实例。不能使用 delete 方法

函数原型：

```
void Release();
```

6.2.4 Init 方法

使会员系统开始与交易后台建立连接，连接成功后可以进行登陆。

函数原型：

```
void Init();
```

6.2.5 Join 方法

会员系统等待一个接口实例线程的结束。

函数原型：

```
void Join();
```

6.2.6 GetTradingDay 方法

获得当前交易日。只有当与交易后台连接建立后才会取到正确的值。

函数原型：

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

返回值：

返回一个指向日期信息字符串的常量指针。

6.2.7 RegisterSpi 方法

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

函数原型：

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

参数：

pSpi：实现了 CFfexFtdcTraderSpi 接口的实例指针。

6.2.8 RegisterFront 方法

设置交易系统前置的网络通讯地址，交易系统拥有多个通信地址,，用户可以同时注册多个前置系统的网络通讯地址。

函数原型：

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

参数：

pszFrontAddress：指向交易所前置系统网络通讯地址的指针。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。

6.2.9 RegisterNameServer 方法

设置交易所 NameServer 的网络通讯地址，用于获取前置列表。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：指向交易所 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。

6.2.10 SetHeartbeatTimeout 方法

设置网络通信心跳的超时时间。当 TraderAPI 与交易系统的 TCP 连接建立后，连接会定时发送心跳，用以检测连接是否正常。该方法用于设置检测心跳超时的时间。**交易所建议会员系统将 timeout 值设置为 10 至 30 秒之间。**

函数原型：

```
virtual void SetHeartbeatTimeout(unsigned int timeout);
```

参数：

timeout：心跳超时时间（秒）。若超过 timeout/2 秒未收到交易系统的任何信

息时，将触发 CFfexFtdcTraderApi::OnHeartBeatWarning()。若超过 timeout 秒未收到交易系统的任何信息时，连接将会中断，将触发 CFfexFtdcTraderApi ::OnFrontDisconnected()。

参见第一部分**错误！未找到引用源。错误！未找到引用源。**

6.2.11 OpenRequestLog 方法

打开请求日志文件。调用该方法之后，所有向交易系统发出的请求信息将记录在指定的日志文件中。

函数原型：

```
virtual int OpenRequestLog(const char *pszReqLogFileName);
```

参数：

pszReqLogFileName：请求日志文件名。

6.2.12 OpenResponseLog 方法

打开应答日志文件。调用该方法之后，所有交易系统返回的信息将记录在指定的日志文件中，包括应答信息和回报信息。

函数原型：

```
virtual int OpenResponseLog(const char *pszRspLogFileName);
```

参数：

pszRspLogFileName：应答日志文件名。

6.2.13 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。

函数原型：

```
void SubscribePrivateTopic(TE_RESUME_TYPE nResumeType);
```

参数：

nResumeType： 私有流重传方式

- TERT_RESTART:从本交易日开始重传

- TERT_RESUME:从上次收到的续传**为保证会员交易数据的完整性，交易所建议使用此方式接收私有流，待会员当日交易数据恢复后再处理后续报单业务。**
- TERT_QUICK:只传送登录后私有流的内容**为保证会员交易数据的完整性，交易所不建议使用此方式接收私有流。**

6.2.14 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。

函数原型：

```
void SubscribePublicTopic(TE_RESUME_TYPE nResumeType);
```

参数：

nResumeType： 公共流重传方式

- TERT_RESTART:从本交易日开始重传
- TERT_RESUME:从上次收到的续传
- TERT_QUICK:只传送登录后公共流的内容

6.2.15 SubscribeUserTopic 方法

订阅交易员私有流。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。

函数原型：

```
void SubscribeUserTopic(TE_RESUME_TYPE nResumeType);
```

参数：

nResumeType：公共流重传方式，

- TERT_RESTART：从本交易日开始重传；
- TERT_RESUME：从上次收到的续传；**为保证会员交易数据的完整性，交易所建议使用此方式接收私有流，待会员和交易员当日交易数据恢复后再处理后续报单业务。**
- TERT_QUICK：只传送登录后会员私有流的内容；**为保证会员交易数据的完整性，交易所不建议使用此方式接收私有流。**

6.2.16 ReqUserLogin 方法

用户发出登陆请求。

函数原型：

```
int ReqUserLogin(
CFfexFtdcReqUserLoginField *pReqUserLoginField,
int nRequestID);
```

参数：

pReqUserLoginField：指向用户登录请求结构的地址。

用户登录请求结构：

```
struct CFfexFtdcReqUserLoginField
    {
///交易日
TFfexFtdcDateType   TradingDay;
///交易用户代码
TFfexFtdcUserIDType UserID;
///会员代码
TFfexFtdcParticipantIDType  ParticipantID;
///密码
TFfexFtdcPasswordType   Password;
        ///用户端产品信息
        TFfexFtdcProductInfoType    UserProductInfo;
        ///接口端产品信息
        TFfexFtdcProductInfoType    InterfaceProductInfo;
        ///协议信息
        TFfexFtdcProtocolInfoType   ProtocolInfo;
        ///数据中心代码
        TFfexFtdcDataCenterIDType   DataCenterID;
    };
```

用户需要填写 UserProductInfo 字段，即会员系统的产品信息，如软件开发商、版本号等，例如：”Ffex Trader V100”代表中国金融期货交易所开发的交易程序和版本号。

如果会员系统自行维护重传序号，则注意 TradingDay 和 DataCenterID 填写为上次登录应答的返回值。如果是第一次登录或不需要续传，则 TradingDay 可以填空字符串（””），DataCenterID 可以填 0 或交易所公布的主数据中心代码。

nRequestID：用户登录请求的 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。

6.2.17 ReqUserLogout 方法

用户发出登出请求。

函数原型：

```
int ReqUserLogout(
CFfexFtdcReqUserLogoutField *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 CFfexFtdcReqUserLogoutField
{
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///会员代码
    TFfexFtdcParticipantIDType  ParticipantID;
};
```

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；
- -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。

6.2.18 ReqUserPasswordUpdate 方法

用户密码修改请求。

函数原型：

```
int ReqUserPasswordUpdate(
CFfexFtdcUserPasswordUpdateField *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 CFfexFtdcUserPasswordUpdateField
    {
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///会员代码
```

```
    TFfexFtdcParticipantIDType  ParticipantID;
    ///旧密码
    TFfexFtdcPasswordType   OldPassword;
    ///新密码
    TFfexFtdcPasswordType   NewPassword;
};
```

nRequestID：用户操作请求的 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。

6.2.19 ReqSubscribeTopic 方法

订阅主题请求请求。应当在登录完成后调用。

函数原型：

```
int ReqSubscribeTopic (
    CFfexFtdcDisseminationField * pDissemination,
    int nRequestID);
```

参数：

pDissemination：指向订阅主题结构的地址，包含了要订阅的主题和起始报文的序号。订阅主题结构：

```
struct CFfexFtdcDisseminationField {
    ///序列系列号
    TFfexFtdcSequenceSeriesType SequenceSeries;
    ///序列号
    TFfexFtdcSequenceNoType SequenceNo;
};
```

SequenceSeries：要订阅的主题，1 表示对话流，2 表示会员私有流，3 表示公共流，4 表示查询，5 表示交易员私有流

SequenceNo：=-1 表示使用快照方式，其它值表示从该序号开始续传

nRequestID：用户操作请求的 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。

6.2.20 ReqQryTopic 方法

查询主题请求。应当在登录完成后调用。

函数原型：

```
int ReqQryTopic (
    CFfexFtdcDisseminationField * pDissemination,
    int nRequestID);
```

参数：

pDissemination：指向查询主题结构的地址，包含了要查询的主题。订阅主题结构：

```
struct CFfexFtdcDisseminationField {
    ///序列系列号
    TFfexFtdcSequenceSeriesType SequenceSeries;
    ///序列号
    TFfexFtdcSequenceNoType SequenceNo;
};
SequenceSeries：要查询的主题
SequenceNo：不用填写
```

nRequestID：用户操作请求的 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。

6.2.21 ReqOrderInsert 方法

会员系统发出报单录入请求。

函数原型：

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

参数：

pInputOrder：指向输入报单结构的地址。

输入报单结构：

```
struct CFfexFtdcInputOrderField
{
```

```
        ///报单编号，该字段由交易后台返回。
        TFfexFtdcOrderSysIDType  OrderSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType     UserID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///报单价格条件，只支持”限价”
        TFfexFtdcOrderPriceTypeType     OrderPriceType;
        ///买卖方向
        TFfexFtdcDirectionType  Direction;
        ///组合开平标志，只有第一个标志有效
        TFfexFtdcCombOffsetFlagType CombOffsetFlag;
        ///组合投机套保标志，只有第一个标志有效
        TFfexFtdcCombHedgeFlagType  CombHedgeFlag;
        ///价格
        TFfexFtdcPriceType  LimitPrice;
        ///数量
        TFfexFtdcVolumeType VolumeTotalOriginal;
        ///有效期类型
        TFfexFtdcTimeConditionType  TimeCondition;
        ///GTD 日期，未使用
        TFfexFtdcDateType   GTDDate;
        ///成交量类型，只支持”任意数量”
        TFfexFtdcVolumeConditionType    VolumeCondition;
        ///最小成交量，未使用
        TFfexFtdcVolumeType MinVolume;
        ///触发条件，只支持”立即”
        TFfexFtdcContingentConditionType    ContingentCondition;
        ///止损价，未使用
        TFfexFtdcPriceType  StopPrice;
        ///强平原因，只支持”非强平”
        TFfexFtdcForceCloseReasonType   ForceCloseReason;
        ///本地报单编号
        TFfexFtdcOrderLocalIDType   OrderLocalID;
        ///自动挂起标志
        TFfexFtdcBoolType   IsAutoSuspend;
        ///业务单元
        TFfexFtdcBusinessUnitType   BusinessUnit;
    };
    * OrderLocalID：本地报单编号，只能单调递增。每次登入成功后，可以从 OnRspUserLogin
的输出参数 CFfexFtdcRspUserLoginField 中获得上次登入用过的最大 OrderLocalID，
MaxOrderLocalID。因为交易系统按照字符串比较 OrderLocalID 的大小，所以在设置 OrderLocalID
时要填满 TFfexFtdcOrderLocalIDType 的全部空间。
```

nRequestID：用户报单请求的 ID，该 ID 由用户指定，管理。在一次会话中，该 ID 不能重复。

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。

6.2.22 ReqOrderAction 方法

会员系统发出报单操作请求，包括报单的撤销、报单的挂起、报单的激活、报单的修改。

函数原型：

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

参数：

pOrderAction：指向报单操作结构的地址。

报单操作结构：

```
struct CFfexFtdcOrderActionField
    {
        ///报单编号,，该字段由交易后台返回。
        TFfexFtdcOrderSysIDType OrderSysID;
        ///本地报单编号*
        TFfexFtdcOrderLocalIDType  OrderLocalID;
        ///报单操作标志
        TFfexFtdcActionFlagType ActionFlag;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///价格，未使用
        TFfexFtdcPriceType  LimitPrice;
        ///操作本地编号*
        TFfexFtdcOrderLocalIDType  ActionLocalID;
        ///数量变化
        TFfexFtdcVolumeType VolumeChange;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
    * OrderSysID，OrderLocalID 是指要进行操作的目标报单，可以任填一个。
    * ActionLocalID：操作本地编号，只能单调递增。每次登入成功后，可以从 OnRspUserLogin 的输出参数 CFfexFtdcRspUserLoginField 中获得上次登入用过的最大 OrderLocalID，MaxOrderLocalID。因为交易系统按照字符串比较 ActionLocalID 的大小，所以在设置 ActionLocalID 时要填满 TFfexFtdcOrderLocalIDType 的全部空间。
```

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。

6.2.23 ReqQuoteInsert 方法

当前版本不支持该方法。

会员系统发出报价录入请求。

函数原型：

```
int ReqQuoteInsert(
CFfexFtdcInputQuoteField *pInputQuote,
int nRequestID);
```

参数：

pInputQuote：指向输入报价结构的地址。

输入报价结构：

```
struct CFfexFtdcInputQuoteField
    {
        ///报价编号，该字段由交易后台返回。
        TFfexFtdcQuoteSysIDType QuoteSysID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///本地报价编号
        TFfexFtdcQuoteLocalIDType   QuoteLocalID;
        ///业务单元，未使用
        TFfexFtdcBusinessUnitType   BusinessUnit;
        ///买方组合开平标志
        TFfexFtdcCombOffsetFlagType BidCombOffsetFlag;
        ///买方组合套保标志
        TFfexFtdcCombHedgeFlagType  BidCombHedgeFlag;
        ///买方价格
        TFfexFtdcPriceType  BidPrice;
        ///卖方组合开平标志
        TFfexFtdcCombOffsetFlagType AskCombOffsetFlag;
        ///卖方组合套保标志
        TFfexFtdcCombHedgeFlagType  AskCombHedgeFlag;
        ///卖方价格
        TFfexFtdcPriceType  AskPrice;
    };
```

nRequestID：用户报价请求的 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。

6.2.24 ReqQuoteAction 方法

当前版本不支持该方法。

会员系统发出报价操作请求，包括报价的撤销、报价的挂起、报价的激活、报价的修改。

函数原型：

```
int ReqQuoteAction(
CFfexFtdcQuoteActionField *pQuoteAction,
int nRequestID);
```

参数：

pQuoteAction：指向报价操作结构的地址。

报价操作结构：

```
    struct CFfexFtdcQuoteActionField
{
    ///报价编号
    TFfexFtdcQuoteSysIDType QuoteSysID;
    ///本地报价编号
    TFfexFtdcOrderLocalIDType  QuoteLocalID;
    ///报单操作标志
    TFfexFtdcActionFlagType ActionFlag;
    ///会员代码
    TFfexFtdcParticipantIDType  ParticipantID;
    ///客户代码
    TFfexFtdcClientIDType  ClientID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
     ///操作本地编号
    TFfexFtdcOrderLocalIDType  ActionLocalID;
    ///业务单元
    TFfexFtdcBusinessUnitType  BusinessUnit;
};
```

nRequestID：用户报价操作请求的 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。

6.2.25 ReqExecOrderInsert 方法

当前版本不支持该方法。

执行宣告录入请求。

函数原型：

```
int ReqExecOrderInsert(
CFfexFtdcInputExecOrderField *pInputExecOrder,
int nRequestID);
```

参数：

pInputExecOrder：指向执行宣告结构的地址。

执行宣告结构：

```
struct CFfexFtdcInputExecOrderField
    {
        ///合约编号
        TFfexFtdcInstrumentIDType  InstrumentID;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///本地执行宣告编号
        TFfexFtdcOrderLocalIDType  ExecOrderLocalID;
        ///数量
        TFfexFtdcVolumeType Volume;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

nRequestID：宣告录入请求的 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。

6.2.26 ReqExecOrderAction 方法

当前版本不支持该方法。

执行宣告操作请求。

函数原型：

```
int ReqExecOrderAction(
CFfexFtdcExecOrderActionField *pExecOrderAction,
int nRequestID);
```

参数：

pExecOrderAction：指向执行宣告操作结构的地址。

执行宣告操作结构：

```
struct CFfexFtdcExecOrderActionField
    {
        ///执行宣告编号
        TFfexFtdcExecOrderSysIDType ExecOrderSysID;
        ///本地执行宣告编号
        TFfexFtdcOrderLocalIDType  ExecOrderLocalID;
        ///报单操作标志
        TFfexFtdcActionFlagType ActionFlag;
        ///会员代码
        TFfexFtdcParticipantIDType  ParticipantID;
        ///客户代码
        TFfexFtdcClientIDType  ClientID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
        ///操作本地编号
        TFfexFtdcOrderLocalIDType  ActionLocalID;
        ///业务单元
        TFfexFtdcBusinessUnitType  BusinessUnit;
    };
```

nRequestID：执行宣告操作请求的 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。

6.2.27 ReqQryPartAccount 方法

会员资金查询请求。所有超时没有完成的查询请求都会被清除（以下查询方法均如此）。

函数原型：

```
int ReqQryPartAccount(
CFfexFtdcQryPartAccountField *pQryPartAccount,
int nRequestID);
```

参数：

pQryPartAccount：指向会员资金查询结构的地址。

会员资金查询结构：

```
    struct CFfexFtdcQryPartAccountField
    {
        ///起始会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///资金帐号，可选
        TFfexFtdcAccountIDType  AccountID;
    };
```

nRequestID：用户操作请求的 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。

6.2.28 ReqQryOrder 方法

报单查询请求。

函数原型：

```
int ReqQryOrder(
CFfexFtdcQryOrderField *pQryOrder,
int nRequestID);
```

参数：

pQryOrder：指向报单查询结构的地址。

报单查询结构：

```
struct CFfexFtdcQryOrderField
    {
        ///起始会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///报单编号，可选
        TFfexFtdcOrderSysIDType OrderSysID;
        ///合约代码，可选
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///客户代码，可选
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码，可选
        TFfexFtdcUserIDType UserID;
        ///开始时间，可选
        TFfexFtdcTimeType   TimeStart;
        ///结束时间，可选
        TFfexFtdcTimeType   TimeEnd;

    };
```

nRequestID：用户报单查询请求的 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。

6.2.29 ReqQryQuote 方法

当前版本不支持该方法。

报价查询请求。

函数原型：

```
int ReqQryQuote(
CFfexFtdcQryQuoteField *pQryQuote,
int nRequestID);
```

参数：

pQryQuote：指向报价查询结构的地址。

报价查询结构：

```
struct CFfexFtdcQryQuoteField
    {
        ///起始会员代码
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///报价编号
        TFfexFtdcQuoteSysIDType QuoteSysID;
        ///客户代码
        TFfexFtdcClientIDType   ClientID;
        ///合约代码
        TFfexFtdcInstrumentIDType   InstrumentID;
        ///交易用户代码
        TFfexFtdcUserIDType UserID;
    };
```

nRequestID：用户报价查询请求的 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。

6.2.30 ReqQryTrade 方法

成交单查询请求。

函数原型：

```
int ReqQryTrade(
CFfexFtdcQryTradeField *pQryTrade,
int nRequestID);
```

参数：

pQryTrade：指向成交查询结构的地址。

成交查询结构：

```
struct CFfexFtdcQryTradeField
    {
        ///起始会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///起始合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDStart;
        ///结束合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDEnd;
        ///成交编号，可选
        TFfexFtdcTradeIDType    TradeID;
        ///客户代码，可选
        TFfexFtdcClientIDType   ClientID;
        ///交易用户代码，可选
        TFfexFtdcUserIDType UserID;
        ///开始时间，可选
        TFfexFtdcTimeType   TimeStart;
        ///结束时间，可选
        TFfexFtdcTimeType   TimeEnd;

    };
```

nRequestID：用户成交单查询请求的 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。

6.2.31 ReqQryClient 方法

会员客户查询请求。

函数原型：

```
int ReqQryClient(
CFfexFtdcQryClientField *pQryClient,
int nRequestID);
```

参数：

pQryClient：指向客户查询结构的地址。

客户查询结构：

```
struct CFfexFtdcQryClientField
    {
        ///起始会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///起始客户代码，可选
```

```
    TFfexFtdcClientIDType   ClientIDStart;
    ///结束客户代码，可选
    TFfexFtdcClientIDType   ClientIDEnd;
};
```

nRequestID：用户客户查询请求的 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。

6.2.32 ReqQryPartPosition 方法

会员持仓查询请求。

函数原型：

```
int ReqQryPartPosition(
CFfexFtdcQryPartPositionField *pQryPartPosition,
int nRequestID);
```

参数：

pQryPartPosition：指向会员持仓查询结构的地址。

会员持仓查询结构：

```
struct CFfexFtdcQryPartPositionField
    {
        ///起始会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///起始合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDStart;
        ///结束合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDEnd;
    };
```

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。

6.2.33 ReqQryClientPosition 方法

客户持仓查询请求。

函数原型：

```
int ReqQryClientPosition(
CFfexFtdcQryClientPositionField *pQryClientPosition,
int nRequestID);
```

参数：

pQryClientPosition：指向客户持仓查询结构的地址。

客户持仓查询结构：

```
struct CFfexFtdcQryClientPositionField
    {
        ///起始会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///起始客户代码，可选
        TFfexFtdcClientIDType   ClientIDStart;
        ///结束客户代码，可选
        TFfexFtdcClientIDType   ClientIDEnd;
        ///起始合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDStart;
        ///结束合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDEnd;
        ///客户类型，可选
        TFfexFtdcClientTypeType ClientType;
    };
```

nRequestID：客户持仓查询请求的 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。

6.2.34 ReqQryInstrument 方法

合约查询请求。

函数原型：

```
int ReqQryInstrument(
CFfexFtdcQryInstrumentField *pQryInstrument,
int nRequestID);
```

参数：

pQryInstrument：指向合约查询结构的地址。

合约查询结构：

```
struct CFfexFtdcQryInstrumentField
    {
        ///结算组代码，可选
        TFfexFtdcSettlementGroupIDType SettlementGroupID;
        ///产品组代码，可选
        TFfexFtdcProductGroupIDType ProductGroupID;
        ///产品代码，可选
        TFfexFtdcProductIDType ProductID;
        ///合约代码，可选
        TFfexFtdcInstrumentIDType InstrumentID;
    };
```

nRequestID：合约查询请求的 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。

6.2.35 ReqQryInstrumentStatus 方法

合约交易状态查询请求。

函数原型：

```
int ReqQryInstrumentStatus(
CFfexFtdcQryInstrumentStatusField *pQryInstrumentStatus,
int nRequestID);
```

参数：

pQryInstrumentStatus：指向合约状态查询结构的地址。

合约状态查询结构：

```
struct CFfexFtdcQryInstrumentStatusField
    {
        ///起始合约代码，可选
        TFfexFtdcInstrumentIDType InstIDStart;
        ///结束合约代码，可选
        TFfexFtdcInstrumentIDType InstIDEnd;
    };
```

nRequestID：合约状态查询请求的 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。

6.2.36 ReqQryMarketData 方法

会员系统发出普通行情查询请求。

函数原型：

```
int ReqQryMarketData(
CFfexFtdcQryMarketDataField *pQryMarketData,
int nRequestID);
```

参数：

pQryMarketData：指向行情查询结构的地址。

行情查询结构：

```
struct CFfexFtdcQryMarketDataField
    {
            ///产品代码，可选
            TFfexFtdcProductIDType  ProductID;
            ///合约代码，可选
            TFfexFtdcInstrumentIDType   InstrumentID;
    };
```

nRequestID：用户查询请求的 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。

6.2.37 ReqQryBulletin 方法

交易所公告查询请求。

函数原型：

```
int ReqQryBulletin(
CFfexFtdcQryBulletinField *pQryBulletin,
int nRequestID);
```

参数：

pQryBulletin：指向公告查询结构的地址。

公告查询结构：

```
struct CFfexFtdcQryBulletinField
    {
        ///交易日，可选
```

```
        TFfexFtdcDateType   TradingDay;
        ///市场代码，可选
        TFfexFtdcMarketIDType   MarketID;
        ///公告编号，可选
        TFfexFtdcBulletinIDType BulletinID;
        ///公告类型，可选
        TFfexFtdcNewsTypeType   NewsType;
        ///紧急程度，可选
        TFfexFtdcNewsUrgencyType    NewsUrgency;
    };
```

nRequestID：公告查询请求的 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。

6.2.38 ReqQryMBLMarketData 方法

合约价位查询请求。

函数原型：

```
int ReqQryMBLMarketData(
CFfexFtdcQryMBLMarketDataField *pQryMBLMarketData,
int nRequestID);
```

参数：

pQryMBLMarketData：指向合约价位查询结构的地址。

合约价位查询结构：

```
struct CFfexFtdcQryMBLMarketDataField
    {
///起始合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDStart;
        ///结束合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDEnd;
        ///买卖方向，可选
        TFfexFtdcDirectionType  Direction;
    };
```

nRequestID：合约价位查询请求的 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。

6.2.39 ReqQryHedgeVolume 方法

保值额度查询请求。

函数原型：

```
int ReqQryHedgeVolume(
CFfexFtdcQryHedgeVolumeField *pQryHedgeVolume,
int nRequestID);
```

参数：

pQryHedgeVolume：指向保值额度查询结构的地址。

保值额度查询结构：

```
struct CFfexFtdcQryHedgeVolumeField
    {
        ///起始会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDStart;
        ///结束会员代码，只能是本会员
        TFfexFtdcParticipantIDType  PartIDEnd;
        ///起始客户代码，可选
        TFfexFtdcClientIDType   ClientIDStart;
        ///结束客户代码，可选
        TFfexFtdcClientIDType   ClientIDEnd;
        ///起始合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDStart;
        ///结束合约代码，可选
        TFfexFtdcInstrumentIDType   InstIDEnd;
    };
```

nRequestID：保值额度查询请求的 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。

6.2.40 ReqCombOrderInsert 方法

当前版本不支持该方法。

会员系统发出非常规组合录入请求。

函数原型：

```
int ReqCombOrderInsert (
CFfexFtdcInputCombOrderField *pInputCombOrder,
 int nRequestID
);
```

参数：

pInputCombOrder：指向非常规组合报单录入结构的地址。非常规组合报单录入结构：

```
struct CFfexFtdcInputCombOrderField {
    ///组合报单编号
    TFfexFtdcOrderSysIDType CombOrderSysID;
    ///会员代码
    TFfexFtdcParticipantIDType  ParticipantID;
    ///客户代码
    TFfexFtdcClientIDType   ClientID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///价格
    TFfexFtdcPriceType  LimitPrice;
    ///数量
    TFfexFtdcVolumeType VolumeTotalOriginal;
    ///本地报单编号
    TFfexFtdcOrderLocalIDType   CombOrderLocalID;
    ///业务单元
    TFfexFtdcBusinessUnitType   BusinessUnit;
    ///合约代码 1
    TFfexFtdcInstrumentIDType   InstrumentID1;
    ///买卖方向 1
    TFfexFtdcDirectionType  Direction1;
    ///分腿乘数 1
    TFfexFtdcLegMultipleType    LegMultiple1;
    ///开平标志 1
    TFfexFtdcOffsetFlagType OffsetFlag1;
    ///投机套保标志 1
    TFfexFtdcHedgeFlagType  HedgeFlag1;
    ///合约代码 2
    TFfexFtdcInstrumentIDType   InstrumentID2;
    ///买卖方向 2
    TFfexFtdcDirectionType  Direction2;
    ///分腿乘数 2
    TFfexFtdcLegMultipleType    LegMultiple2;
    ///开平标志 2
    TFfexFtdcOffsetFlagType OffsetFlag2;
    ///投机套保标志 2
    TFfexFtdcHedgeFlagType  HedgeFlag2;
    ///合约代码 3
    TFfexFtdcInstrumentIDType   InstrumentID3;
    ///买卖方向 3
    TFfexFtdcDirectionType  Direction3;
    ///分腿乘数 3
    TFfexFtdcLegMultipleType    LegMultiple3;
    ///开平标志 3
    TFfexFtdcOffsetFlagType OffsetFlag3;
    ///投机套保标志 3
    TFfexFtdcHedgeFlagType  HedgeFlag3;
    ///合约代码 4
    TFfexFtdcInstrumentIDType   InstrumentID4;
    ///买卖方向 4
    TFfexFtdcDirectionType  Direction4;
    ///分腿乘数 4
    TFfexFtdcLegMultipleType    LegMultiple4;
    ///开平标志 4
    TFfexFtdcOffsetFlagType OffsetFlag4;
    ///投机套保标志 4
    TFfexFtdcHedgeFlagType  HedgeFlag4;
};
```

nRequestID：用户非常规报单录入请求的 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。

6.2.41 ReqQryCombOrder 方法

当前版本不支持该方法。

报价查询请求。

函数原型：

```
int ReqQryCombOrder (
    CFfexFtdcQryCombOrderField *pQryCombOrder,
    int nRequestID);
```

参数：

pCombOrder：指向非常规组合报单结构的地址。非常规组合报单结构：

```
struct CFfexFtdcCombOrderField {
    ///交易日
    TFfexFtdcDateType  TradingDay;
    ///结算组代码
    TFfexFtdcSettlementGroupIDType SettlementGroupID;
    ///结算编号
    TFfexFtdcSettlementIDType  SettlementID;
    ///组合报单编号
    TFfexFtdcOrderSysIDType CombOrderSysID;
    ///会员代码
    TFfexFtdcParticipantIDType ParticipantID;
    ///客户代码
    TFfexFtdcClientIDType  ClientID;
    ///交易用户代码
    TFfexFtdcUserIDType UserID;
    ///价格
    TFfexFtdcPriceType LimitPrice;
    ///数量
    TFfexFtdcVolumeType VolumeTotalOriginal;
    ///本地报单编号
    TFfexFtdcOrderLocalIDType  CombOrderLocalID;
    ///业务单元
    TFfexFtdcBusinessUnitType  BusinessUnit;
    ///合约代码 1
    TFfexFtdcInstrumentIDType  InstrumentID1;
    ///买卖方向 1
    TFfexFtdcDirectionType Direction1;
    ///分腿乘数 1
    TFfexFtdcLegMultipleType   LegMultiple1;
    ///开平标志 1
    TFfexFtdcOffsetFlagType OffsetFlag1;
    ///投机套保标志 1
```

```
    TFfexFtdcHedgeFlagType  HedgeFlag1;
    ///合约代码 2
    TFfexFtdcInstrumentIDType   InstrumentID2;
    ///买卖方向 2
    TFfexFtdcDirectionType  Direction2;
    ///分腿乘数 2
    TFfexFtdcLegMultipleType    LegMultiple2;
    ///开平标志 2
    TFfexFtdcOffsetFlagType OffsetFlag2;
    ///投机套保标志 2
    TFfexFtdcHedgeFlagType  HedgeFlag2;
    ///合约代码 3
    TFfexFtdcInstrumentIDType   InstrumentID3;
    ///买卖方向 3
    TFfexFtdcDirectionType  Direction3;
    ///分腿乘数 3
    TFfexFtdcLegMultipleType    LegMultiple3;
    ///开平标志 3
    TFfexFtdcOffsetFlagType OffsetFlag3;
    ///投机套保标志 3
    TFfexFtdcHedgeFlagType  HedgeFlag3;
    ///合约代码 4
    TFfexFtdcInstrumentIDType   InstrumentID4;
    ///买卖方向 4
    TFfexFtdcDirectionType  Direction4;
    ///分腿乘数 4
    TFfexFtdcLegMultipleType    LegMultiple4;
    ///开平标志 4
    TFfexFtdcOffsetFlagType OffsetFlag4;
    ///投机套保标志 4
    TFfexFtdcHedgeFlagType  HedgeFlag4;
    ///报单来源
    TFfexFtdcOrderSourceType    OrderSource;
    ///今成交数量
    TFfexFtdcVolumeType VolumeTraded;
    ///剩余数量
    TFfexFtdcVolumeType VolumeTotal;
    ///报单日期
    TFfexFtdcDateType   InsertDate;
    ///插入时间
    TFfexFtdcTimeType   InsertTime;
    ///结算会员编号
    TFfexFtdcParticipantIDType  ClearingPartID;
};
```

nRequestID：用户报价查询请求的 ID，该 ID 由用户指定和管理。

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。

6.2.42 ReqAdminOrderInsert 方法

通过 AdminOrder，初始化、调整以及取消信用限额。在初始化以前，不能进行信用额度的调整。允许不取消信用额度而直接重新设置。额度调整的额度分为正负值，正值表示增加额度，负值表示减少额度。取消信用额度相当于将信用额度清零，取消信用额度总是允许操作的。

函数原型：

```
int ReqAdminOrderInsert(
CFfexFtdcInputAdminOrderField *pInputAdminOrder,
int nRequuestID);
```

参数：

pInputAdminOrder：指向输入管理员报单结构的地址。结构中的 InstrumentID 字段是不用填的。

输入管理员报单结构：

```
struct CFfexFtdcInputAdminOrderField
{
 ///合约代码
 TFfexFtdcInstrumentIDType  InstrumentID;
 ///管理报单命令
 TFfexFtdcAdminOrderCommandFlagType AdminOrderCommand;
 ///结算会员编号
 TFfexFtdcParticipantIDType ClearingPartID;
 ///交易会员编号
 TFfexFtdcParticipantIDType ParticipantID;
 ///金额
 TFfexFtdcMoneyType  Amount;
 ///结算组代码
 TFfexFtdcSettlementGroupIDType SettlementGroupID;
};
```

nRequestID：输入管理员报单请求的 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。

6.2.43 ReqQryCreditLimit 方法

信用额度查询请求。

函数原型：

```
int ReqQryCreditLimit(
CFfexFtdcQryCreditLimitField *pQryCreaditLimit,
int nRequestID);
```

参数：

pQryCreaditLimit：指向信用额度查询结构的地址

信用额度查询结构

```
struct CFfexFtdcQryCreditLimitField
{
///交易会员编号
TFfexFtdcParticipantIDType ParticipantID;
///结算会员编号
TFfexFtdcParticipantIDType ClearingPartID;
};
```

nRequestID：信用额度查询请求的 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。

第7章 开发示例

```
// tradeapitest.cpp :
// 一个简单的例子，介绍CFfexFtdcTraderApi和CFfexFtdcTraderSpi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 "FtdcTraderApi.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);
// 会员代码
TFfexFtdcParticipantIDType g_chParticipantID;
// 交易用户代码
TFfexFtdcUserIDType g_chUserID;

class CSimpleHandler : public CFfexFtdcTraderSpi
{
public: // 构造函数，需要一个有效的指向CFfexFtdcMduserApi实例的指针
    CSimpleHandler(CFfexFtdcTraderApi *pUserApi) : m_pUserApi(pUserApi) {}

    ~CSimpleHandler() {}

    // 当会员系统与交易系统建立起通信连接，会员系统需要进行登录
    virtual void OnFrontConnected()
    {
        CFfexFtdcReqUserLoginField reqUserLogin;
        // get ParticipantID
        printf("participantid:");
        scanf("%s", &g_chParticipantID);
        strcpy(reqUserLogin.ParticipantID, g_chParticipantID);
        // 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);
    }

    // 当会员系统与交易系统通信连接断开时，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，会员系统可不做处理
        printf("OnFrontDisconnected.\n");
    }

    // 当会员系统发出登录请求之后，该方法会被调用，通知会员系统登录是否成功
    virtual void OnRspUserLogin(CFfexFtdcRspUserLoginField *pRspUserLogin,
CFfexFtdcRspInfoField *pRspInfo, int nRequestID, bool bIsLast)
    {
        printf("OnRspUserLogin:\n");
        printf("ErrorCode=[%d], ErrorMsg=[%s]\n", pRspInfo->ErrorID,
pRspInfo->ErrorMsg);
        printf("RequestID=[%d], Chain=[%d]\n", nRequestID, bIsLast);
```

```
        if (pRspInfo->ErrorID != 0) {
    // 登录失败，会员系统需进行错误处理
            printf("Failed to login, errorcode=%d errormsg=%s requestid=%d
chain=%d", pRspInfo->ErrorID, pRspInfo->ErrorMsg, nRequestID, bIsLast);
            exit(-1);
        }

        // 登录成功,发出报单录入请求
        CFfexFtdcInputOrderField ord;
        memset(&ord, 0, sizeof(ord));

        // 会员代码
        strcpy(ord.ParticipantID, g_chParticipantID);
        // 客户代码
        strcpy(ord.ClientID, "12345");
        // 交易用户代码
        strcpy(ord.UserID, g_chUserID);
        // 合约代码
        strcpy(ord.InstrumentID, "cn0601");
        // 报单价格条件
        ord.OrderPriceType = FFEX_FTDC_OPT_LimitPrice;
        // 买卖方向
        ord.Direction = FFEX_FTDC_D_Buy;
        // 组合开平标志
        strcpy(ord.CombOffsetFlag, "0");
        // 组合投机套保标志
        strcpy(ord.CombHedgeFlag, "1");
        // 价格
        ord.LimitPrice = 50000;
        // 数量
        ord.VolumeTotalOriginal = 10;
        // 有效期类型
        ord.TimeCondition = FFEX_FTDC_TC_GFD;
        // GTD日期
        strcpy(ord.GTDDate, "");
        // 成交量类型
        ord.VolumeCondition = FFEX_FTDC_VC_AV;
        // 最小成交量
        ord.MinVolume = 0;
        // 触发条件
        ord.ContingentCondition = FFEX_FTDC_CC_Immediately;
        // 止损价
        ord.StopPrice = 0;
        // 强平原因
        ord.ForceCloseReason = FFEX_FTDC_FCC_NotForceClose;
        // 本地报单编号
        strcpy(ord.OrderLocalID, "0000000001");
        // 自动挂起标志
        ord.IsAutoSuspend = 0;

        m_pUserApi->ReqOrderInsert(&ord, 1);
    }

    // 报单录入应答
    virtual void OnRspOrderInsert(CFfexFtdcInputOrderField *pInputOrder,
CFfexFtdcRspInfoField *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(CFfexFtdcOrderField *pOrder)
    {
        printf("OnRtnOrder:\n");
        printf("OrderSysID=[%s]\n", pOrder->OrderSysID);
    }

    // 针对用户请求的出错通知
    virtual void OnRspError(CFfexFtdcRspInfoField *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);
        // 会员系统需进行错误处理
        exit(-1);
    }

private:
    // 指向CFfexFtdcMduserApi实例的指针
    CFfexFtdcTraderApi *m_pUserApi;
};

int main()
{
    // 产生一个CFfexFtdcTraderApi实例
    CFfexFtdcTraderApi *pUserApi = CFfexFtdcTraderApi::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
    //       TERT_RESTART:从本交易日开始重传
    //       TERT_RESUME:从上次收到的续传
    //       TERT_QUICK:只传送登录后私有流的内容
    pUserApi->SubscribePrivateTopic(TERT_RESUME);

    // 订阅公共流
    //       TERT_RESTART:从本交易日开始重传
    //       TERT_RESUME:从上次收到的续传
    //       TERT_QUICK:只传送登录后公共流的内容
    pUserApi->SubscribePublicTopic(TERT_RESUME);

    //设置心跳超时时间
    pUserApi->SetHeartbeatTimeout(19);

    // 设置交易系统前置NameServer的地址
    pUserApi->RegisterNameServer("tcp://172.16.0.31:17001");
    // 使会员系统开始与交易系统建立连接
    pUserApi->Init();

    // 会员系统等待报单操作完成
    WaitForSingleObject(g_hEvent, INFINITE);

    // 释放API实例
    pUserApi->Release();

    return 0;
}
```

第8章 附录

8.1 错误编码列表

错误编号	错误提示	错误原因
1	会话不正确	在各项操作中发现会话不合法
2	合约找不到	插入报单、报价、OTC 报单或者执行宣告时，找不到合约
3	会员找不到	在各项操作中发现会员找不到
4	客户找不到	在各项操作中发现客户找不到
6	报单字段错误	插入报单时，发现报单中有字段值不合法（枚举值越界）
		插入报单时，发现非强平单中设置了强平原因
7	报价字段错误	插入报价时，发现报价中有字段值不合法（枚举值越界）
8	报单操作字段错误	报单操作时，发现报单操作中有字段值不合法（枚举值越界）
9	报价操作字段错误	报价操作时，发现报价操作中有字段值不合法（枚举值越界）
12	重复的报单	插入报单或者非标准组合报单时，本地报单号重复
13	重复的报价	插入报价时，本地报价号重复
15	客户没有在该会员开户	在各项操作时，发现指定的客户并没有在指定的会员中开户
16	IOC 需在连续交易阶段	在非连续交易阶段企图插入 IOC 单
17	GFA 需在集合竞价阶段	在非集合竞价阶段企图插入 GFA 的报单
18	市价单不能排队	在插入市价单时，发现时间条件不是 IOC
19	数量约束应在 IOC 单上	在插入数量约束不是任意数量的报单时，发现时间条件不是 IOC
20	GTD 报单过期了	在插入 GTD 报单时，发现 GTD 日期已经过期了
21	最小数量大于报单数量	在插入有最小数量条件的报单时，发现最小数量大于报单数量
22	交易所数据没有同步	进行各类业务操作时，发现交易所的数据还没有同步
23	结算组数据没有同步	进行各类业务操作时，发现结算组的数据还没有同步
24	报单找不到	进行报单操作时，发现要操作的报单找不到
25	报价找不到	进行报价操作时，发现要操作的报价找不到
26	当前状态禁止此项操作	在插入报单时，发现合约交易状态不是连续交易、集合竞价报单或者集合竞价平衡
		在报单操作时， 对于激活操作，发现合约交易状态不是连续交易、集合竞价报单或者集合竞价平衡 对于其他操作对于非管理用户，发现合约交易状态不是连续交易或者集合竞价报单 对于管理用户 对于撤单或者挂起操作，发现合约交易状态是已收盘

		对于其他操作，发现合约交易状态不是连续交易或者集合竞价报单
		在插入 OTC 报单时，发现合约的交易状态不是连续交易
27	不合法的合约状态迁移	在切换合约交易状态时，发现此迁移不符合交易状态迁移规定
28	报单已经全部成交	在报单操作时，发现报单已经全部成交了
29	报单已经撤销	在报单操作时，发现报单已经撤销了
31	平仓时客户持仓不足	在各类可能造成平仓的操作时，发现客户的持仓不足
32	超出客户限仓	在各类可能开仓操作时，发现超过了客户的投机限仓
34	超出会员限仓	在各类可能开仓操作时，发现超过了会员限仓
35	找不到帐号	在各类操作时，发现找不到该操作应当使用的帐号
36	资金不足	在各类操作时，发现帐号中没有足够的资金
37	不合法的数量	在插入报单、报单操作、插入 OTC 报单和报单操作时，数量不是最小报单数量要求的正整数倍，或者超过最大报单数量
45	结算组初始化状态不对	在用户登录时，发现没有任何一个结算组数据已经完成过同步
48	价格非最小单位的倍数	在各类操作时，价格不是合约的最小变动单位的整数倍
49	价格超出涨停板	在各类操作时，价格高于合约的涨停板
50	价格跌破跌停板	在各类操作时，价格低于合约的跌停板
51	没有交易权限	在进行各类操作时，发现会员对指定合约、或者客户对指定的合约、或者交易员没有的交易权限
52	只能平仓	在进行各类可能造成开仓的操作时，发现会员对指定合约、或者客户对指定的合约、或者交易员只有平仓的权限
53	没有此项交易角色	在插入报单、插入 OTC 报单或插入组合报单时，该会员在指定合约上不具有该客户对应的交易角色
57	不能为其他会员操作	在各项操作中，发现用户给非其所属的会员进行操作
58	用户不匹配	在各项操作时，发现操作的用户和会话的用户不匹配
59	用户重复登录	在用户登录时，发现此用户已经登录过了
60	用户名或口令错误	在用户登录或者修改口令时，发现用户名找不到或者口令错误
62	用户不活跃	在用户登录时，发现该用户不活跃
64	用户不属于此会员	在用户登录时，发现用户不属于指定的会员
65	错误的登录 IP 地址	在用户登录时，发现用户的 IP 地址不合法
67	并没有以此用户登录	在用户登出时，发现用户并不是以此用户登录的
66	用户尚未登录	在各项操作中，发现用户尚未登录
68	并没有以此会员登录	在用户登出、强制用户登出或者修改口令时，发现用户并不是以此会员登录的
70	报价已经被取消	在报价操作时，发现报价已经被取消了
76	报单已经被挂起	在报单挂起时，发现报单已经被挂起了
77	报单已经被激活	在报单激活时，发现报单已经被激活了
78	GTD 报单没有设定日期	在插入 GTD 报单时，没有指定 GTD 日期
79	不被支持的报单类型	在插入各类报单时，发现目前本交易所还不支持此种报单

		类型
80	用户无此权限	使用普通用户，进行各项需要管理用户才能进行的操作
83	止损单仅用于连续交易	在非连续交易阶段企图插入止损单
84	止损单需是 IOC 或 GFD	在插入止损单时，发现时间条件既不是 IOC，也不是 GFD
89	执行宣告字段错误	插入执行宣告时，发现执行宣告中有字段值不合法（枚举值越界）
90	执行宣告操作字段错误	执行宣告操作时，发现执行宣告操作中有字段值不合法（枚举值越界）
91	重复的执行宣告	插入执行宣告时，本地执行宣告编号重复
92	执行宣告已经取消	在执行宣告操作时，发现执行宣告已经取消了
93	执行宣告找不到	进行执行宣告操作时，发现要操作的执行宣告找不到
94	执行宣告只能用于期权	插入执行宣告时，发现合约是非期权合约
95	止损报单需说明止损价	在插入止损单时，发现没有指定止损价
96	保值额度不足	在各类可能开仓操作时，发现客户的套期保值额度不足
97	重复的操作	报单操作、报价操作或者执行宣告操作时，本地操作号重复
99	不能为其他用户操作	在报单操作时，发现非授权用户企图操作同会员的其他用户插入的报单
100	错误的用户类型	在交易员登录时，发现用户类型是行情用户
103	当日套保仓位不能平仓	企图对套保仓位进行插入平今仓报单
104	不明管理指令	在收到管理指令时，无法认出指令类型
114	最优价单不能排队	在插入最优价单时，发现时间条件不是 IOC

8.2 枚举编码列表

序号	枚举描述	枚举前缀	枚举名称	编码描述	编码名称	编码数值
1	交易角色	ER	TradingRole	代理	Broker	1
				自营	Host	2
				做市商	MarketMaker	3
2	交易用户类型	UT	UserType	交易员	Trader	1
				交易控制员	TradeManager	2
				行情商用户	MDUser	3
				无授权交易员	SingleTrader	4
3	产品类型	PC	ProductClass	期货	Futures	1
				期权	Options	2
				组合	Combination	3
				即期	Spot	4
				期转现	EFP	5
4	期权类型	OT	OptionsType	非期权	NotOptions	0
				看涨	CallOptions	1
				看跌	PutOptions	2
5	合约交易状态	IS	InstrumentStatus	开盘前	BeforeTrading	0
				非交易	NoTrading	1
				连续交易	Continous	2

序号	枚举描述	枚举前缀	枚举名称	编码描述	编码名称	编码数值
				集合竞价报单	AuctionOrdering	3
				集合竞价价格平衡	AuctionBalance	4
				集合竞价撮合	AuctionMatch	5
				收盘	Closed	6
6	买卖方向	D	Direction	买	Buy	0
				卖	Sell	1
7	持仓类型	PT	PositionType	净持仓	Net	1
				综合持仓	Gross	2
8	持仓多空方向	PD	PosiDirection	净	Net	1
				多头	Long	2
				空头	Short	3
9	交易所数据同步状态	EDS	ExchangeDataSyncStatus	未同步	Asynchronous	1
				同步中	Synchronizing	2
				已同步	Synchronized	3
10	结算组数据同步状态	SGDS	SGDataSyncStatus	未同步	Asynchronous	1
				同步中	Synchronizing	2
				已同步	Synchronized	3
11	投机套保标志	HF	HedgeFlag	投机	Speculation	1
				套保	Hedge	3
12	客户类型	CT	ClientType	自然人	Person	0
				法人	Company	1
				投资基金	Fund	2
13	品种进入交易状态原因	IER	InstStatusEnterReason	自动切换	Automatic	1
				手动切换	Manual	2
				熔断	Fuse	3
				熔断手动	FuseManual	4
14	报单价格条件	OPT	OrderPriceType	任意价	AnyPrice	1
				限价	LimitPrice	2
				最优价	BestPrice	3
15	开平标志	OF	OffsetFlag	开仓	Open	0
				平仓	Close	1
				强平	ForceClose	2
				平今	CloseToday	3
				平昨	CloseYesterday	4
16	强平原因	FCC	ForceCloseReason	非强平	NotForceClose	0
				资金不足	LackDeposit	1
				客户超仓	ClientOverPositionLimit	2
				会员超仓	MemberOverPositionLimit	3
				持仓非整数倍	NotMultiple	4
				违规	Violation	5
				其它	Other	6

序号	枚举描述	枚举前缀	枚举名称	编码描述	编码名称	编码数值
17	报单状态	OST	OrderStatus	全部成交	AllTraded	0
				部分成交还在队列中	PartTradedQueueing	1
				部分成交不在队列中	PartTradedNotQueueing	2
				未成交还在队列中	NoTradeQueueing	3
				未成交不在队列中	NoTradeNotQueueing	4
				撤单	Canceled	5
18	报单类型	ORDT	OrderType	正常	Normal	0
				报价衍生	DeriveFromQuote	1
				组合衍生	DeriveFromCombination	2
19	OTC 报单状态	OOS	OTCOrderStatus	一方输入	Inputed	0
				已经确认	Confirmed	1
				已经撤销	Canceled	2
				已经拒绝	Refused	3
20	有效期类型	TC	TimeCondition	立即完成，否则撤销	IOC	1
				本节有效	GFS	2
				当日有效	GFD	3
				指定日期前有效	GTD	4
				撤销前有效	GTC	5
				集合竞价有效	GFA	6
21	成交量类型	VC	VolumeCondition	任何数量	AV	1
				最小数量	MV	2
				全部数量	CV	3
22	触发条件	CC	ContingentCondition	立即	Immediately	1
				止损	Touch	2
23	操作标志	AF	ActionFlag	删除	Delete	0
				挂起	Suspend	1
				激活	Active	2
				修改	Modify	3
24	报单来源	OSRC	OrderSource	来自参与者	Participant	0
				来自管理员	Administrator	1
25	成交类型	TRDT	TradeType	普通成交	Common	0
				期权执行	OptionsExecution	1
				OTC 成交	OTC	2
				期转现衍生成交	EFPDerived	3
				组合衍生成交	CombinationDerived	4
26	成交价来	PSR	PriceSource	前成交价	LastPrice	0

序号	枚举描述	枚举前缀	枚举名称	编码描述	编码名称	编码数值
	源	C		买委托价	Buy	1
				卖委托价	Sell	2
27	帐户状态	ACCS	AccountStatus	激活状态	Enable	0
				停止状态	Disable	1
28	会员类型	MT	MemberType	交易会员	Trading	0
				结算会员	Settlement	1
				综合会员	Compositive	2
29	执行结果	OER	ExecResult	没有执行	NoExec	n
				已经取消	Canceled	c
				执行成功	OK	0
				期权持仓不够	NoPosition	1
				资金不够	NoDeposit	2
				会员不存在	NoParticipant	3
				客户不存在	NoClient	4
				合约不存在	NoInstrument	6
				没有执行权限	NoRight	7
				不合理的数量	InvalidVolume	8
				没有足够的历史成交	NoEnoughHistoryTrade	9
30	管理报单指令	AOC	AdminOrderCommandFlag	交割月持仓非整数倍强平	NotMultipleForceClose	1
				初始化交易会员信用限额	InitCreditLimit	2
				调整交易会员信用限额	AlterCreditLimit	3
				取消交易会员信用限额	CancelCreditLimit	4

8.3 数据类型列表

数据类型名	基本数据类型	数据类型说明
TFfexFtdcErrorIDType	int	错误代码
TFfexFtdcPriorityType	int	优先权
TFfexFtdcSettlementIDType	int	结算编号
TFfexFtdcMonthCountType	int	月份数量
TFfexFtdcTradingSegmentSNType	int	交易阶段编号
TFfexFtdcVolumeType	int	数量
TFfexFtdcTimeSortIDType	int	按时间排队的序号
TFfexFtdcFrontIDType	int	前置编号
TFfexFtdcSessionIDType	int	会话编号
TFfexFtdcSequenceNoType	int	序列号

数据类型名	基本数据类型	数据类型说明
TFfexFtdcBulletinIDType	int	公告编号
TFfexFtdcInformationIDType	int	信息编号
TFfexFtdcMillisecType	int	时间（毫秒）
TFfexFtdcVolumeMultipleType	int	合约数量乘数
TFfexFtdcImplyLevelType	int	派生层数
TFfexFtdcStartPosType	int	起始位置
TFfexFtdcAliasType	char[3]	别名
TFfexFtdcOriginalTextType	char[3]	原文
TFfexFtdcParticipantIDType	char[11]	会员代码
TFfexFtdcParticipantNameType	char[51]	会员名称
TFfexFtdcParticipantAbbrType	char[9]	会员简称
TFfexFtdcUserIDType	char[16]	交易用户代码
TFfexFtdcPasswordType	char[41]	密码
TFfexFtdcClientIDType	char[11]	客户代码
TFfexFtdcInstrumentIDType	char[31]	合约代码
TFfexFtdcProductIDType	char[9]	产品代码
TFfexFtdcProductNameType	char[21]	产品名称
TFfexFtdcExchangeIDType	char[9]	交易所代码
TFfexFtdcDateType	char[9]	日期
TFfexFtdcTimeType	char[9]	时间
TFfexFtdcInstrumentNameType	char[21]	合约名称
TFfexFtdcProductGroupIDType	char[9]	产品组代码
TFfexFtdcProductGroupNameType	char[21]	产品组名称
TFfexFtdcMarketIDType	char[9]	市场代码
TFfexFtdcSettlementGroupIDType	char[9]	结算组代码
TFfexFtdcOrderSysIDType	char[13]	报单编号
TFfexFtdcOTCOrderSysIDType	char[13]	OTC 报单编号
TFfexFtdcExecOrderSysIDType	char[13]	执行宣告系统编号
TFfexFtdcQuoteSysIDType	char[13]	报价编号
TFfexFtdcTradeIDType	char[13]	成交编号
TFfexFtdcOrderLocalIDType	char[13]	本地报单编号
TFfexFtdcComeFromType	char[21]	消息来源
TFfexFtdcAccountIDType	char[13]	资金帐号
TFfexFtdcNewsTypeType	char[3]	公告类型
TFfexFtdcAdvanceMonthType	char[4]	提前月份
TFfexFtdcCommodityIDType	char[9]	商品代码
TFfexFtdcIPAddressType	char[16]	IP 地址
TFfexFtdcProductInfoType	char[41]	产品信息
TFfexFtdcProtocolInfoType	char[41]	协议信息
TFfexFtdcBusinessUnitType	char[21]	业务单元
TFfexFtdcTradingSystemNameType	char[61]	交易系统名称
TFfexFtdcTradingRoleType	char	交易角色
TFfexFtdcUserTypeType	char	交易用户类型

数据类型名	基本数据类型	数据类型说明
TFfexFtdcProductClassType	char	产品类型
TFfexFtdcOptionsTypeType	char	期权类型
TFfexFtdcInstrumentStatusType	char	合约交易状态
TFfexFtdcDirectionType	char	买卖方向
TFfexFtdcPositionTypeType	char	持仓类型
TFfexFtdcPosiDirectionType	char	持仓多空方向
TFfexFtdcExchangeDataSyncStatusType	char	交易所数据同步状态
TFfexFtdcSGDataSyncStatusType	char	结算组数据同步状态
TFfexFtdcHedgeFlagType	char	投机套保标志
TFfexFtdcClientTypeType	char	客户类型
TFfexFtdcInstStatusEnterReasonType	char	品种进入交易状态原因
TFfexFtdcOrderPriceTypeType	char	报单价格条件
TFfexFtdcOffsetFlagType	char	开平标志
TFfexFtdcForceCloseReasonType	char	强平原因
TFfexFtdcOrderStatusType	char	报单状态
TFfexFtdcOrderTypeType	char	报单类型
TFfexFtdcOTCOrderStatusType	char	OTC 报单状态
TFfexFtdcTimeConditionType	char	有效期类型
TFfexFtdcVolumeConditionType	char	成交量类型
TFfexFtdcContingentConditionType	char	触发条件
TFfexFtdcActionFlagType	char	操作标志
TFfexFtdcOrderSourceType	char	报单来源
TFfexFtdcTradeTypeType	char	成交类型
TFfexFtdcPriceSourceType	char	成交价来源
TFfexFtdcAccountStatusType	char	帐户状态
TFfexFtdcMemberTypeType	char	会员类型
TFfexFtdcExecResultType	char	执行结果
TFfexFtdcYearType	int	年份
TFfexFtdcMonthType	int	月份
TFfexFtdcLegMultipleType	int	单腿乘数
TFfexFtdcLegIDType	int	单腿编号
TFfexFtdcBoolType	int	布尔型
TFfexFtdcUserActiveType	int	交易员活跃情况
TFfexFtdcPriceType	double	价格
TFfexFtdcUnderlyingMultipleType	double	合约基础商品乘数
TFfexFtdcCombOffsetFlagType	char[5]	组合开平标志
TFfexFtdcCombHedgeFlagType	char[5]	组合投机套保标志
TFfexFtdcRatioType	double	比率
TFfexFtdcMoneyType	double	资金
TFfexFtdcLargeVolumeType	double	大额数量
TFfexFtdcNewsUrgencyType	char	紧急程度
TFfexFtdcSequenceSeriesType	short	序列系列号
TFfexFtdcCommPhaseNoType	short	通讯时段号

数据类型名	基本数据类型	数据类型说明
TFfexFtdcContentLengthType	int	正文长度
TFfexFtdcErrorMsgType	char[81]	错误信息
TFfexFtdcAbstractType	char[81]	消息摘要
TFfexFtdcContentType	char[501]	消息正文
TFfexFtdcURLLinkType	char[201]	WEB 地址
TFfexFtdcIdentifiedCardNoType	char[51]	证件号码
TFfexFtdcIdentifiedCardNoV1Type	char[21]	原证件号码
TFfexFtdcPartyNameType	char[81]	参与人名称
TFfexFtdcIdCardTypeType	char[16]	证件类型
TFfexFtdcAdminOrderCommandFlagType	char	管理报单指令
TFfexFtdcDataCenterIDType	int	数据中心代码