CTP 客户开发 指南

**利用高科技改善人们的生活**

更新：2017-10-9

版本：2.0

前言

本指南可帮助新开发人员使用 CTP 的 API。它提供了 API 的概述，解释了其基础机制，并概述了开发客户端程序所需的一般步骤。它还回答了一些开发人员最经常 asked 问的问题 developers。

本指南总结了现有文档的内容，并补充了最新版本 API 中新功能的内容。为使本文档的内容完整、准确且易于理解，我们欢迎s错误或遗漏报告，else 或您认为可能有助于改进文档的任何内容。请随时与我们联系。

联系信息

SFIT 财务业务部

[apisupport@sfit.shfe.com.cn](mailto:qiao.yu@sfit.shfe.com.cn)

请 订阅 微信公共帐户，从 SFIT 获得更多实时信息：



目录

[1. CTP 5](#_Toc26298)

[1.1. 简介 5](#_Toc5322)

[1.2. FTD 通信协议 6](#_Toc15251)

[1.2.1. 通信模式 6](#_Toc1855)

[1.2.2. 数据流 7](#_Toc11091)

[1.3. 两种数据交换 模式 8](#_Toc5182)

[1.3.1. 请求/ 回复模式](#_Toc23220)  [8 ８](#_Toc23220)

[1.3.2. 发布/订阅模式 8](#_Toc16799)

[1.4. CTP 架构 8](#_Toc25643)

[2. API 11](#_Toc9233)

[2.1. 简介 11](#_Toc9585)

[2.1.1. 接口文件 12](#_Toc30476)

[2.2. 一般规则 13](#_Toc30265)

[2.2.1. 命名公约 13](#_Toc32042)

[2.2.2. 接口类 13](#_Toc31135)

[2.2.3. 一般参数 13](#_Toc13398)

[2.2.4.](#_Toc2077)  [t rade API 14 的一般初始 化步骤](#_Toc2077)

[3. 收购报价演示 15](#_Toc1145)

[3.1. 准备 15](#_Toc24062)

[3.2. 报价 API 的初始化 16](#_Toc1166)

[3.3. 登录 17](#_Toc23219)

[3.4. 订阅报价数据 18](#_Toc4411)

[3.5. 订阅和接收报价请求 19](#_Toc16236)

[4. 交易演示 21](#_Toc2411)

[4.1. 贸易](#_Toc4072)  [API](#_Toc4072)  [的初始化 21](#_Toc4072)

[4.2. 登录 22](#_Toc26745)

[4.3. 确认结算信息 22](#_Toc3673)

[4.4. 订单 处理 流程 23](#_Toc29420)

[4.5. 处理订单的方法 25](#_Toc21452)

[4.6. 下订单 26](#_Toc4562)

[4.7. 放置 停车令 32](#_Toc20603)

[4.8. 取消订单 33](#_Toc6719)

[4.9. 询 价和 报价 33](#_Toc7145)

[4.10. Exercise 行使备 选 案文35](#_Toc27003)

[5. 附录 37](#_Toc31427)

[5.1. 数据流文件 37](#_Toc17378)

[5.2. F 低控制 38](#_Toc28884)

[5.2.1. 查询流量控制 38](#_Toc4996)

[5.2.2. 订单流量控制 38](#_Toc10127)

[5.3. Dis 取消连接 38](#_Toc19720)

# Ctp

## 介绍

综合交易平台，称为CTP，,是专门为期货公司开发的未来经纪管理系统。它由交易a 系统、风险管理系统和结算信息管理系统组成。

交易系统负责订单处理、报价转发和银行期货转账。结算信息管理系统s负责交易、账户、经纪人、资金、, rate费率、日常结算、信息查询、报表s,等。风险管理系统提供高速、实时的计算，以揭示和控制风险。 CTP 可以连接四个（五个？国内期货交易所同时支持国内商品期货和股指期货的交易结算业务。系统,可自动生成和提交保证金监控和反洗钱监控文件。

CTP继承了S the experience会"新一代交易所系统（NGES）"的经验，代表了level of 国际期货衍生品交易系统的最高水平s。它采用创新的分布式架构，可以精确重播历史并记录每个操作。

纯基于the RAM，CTP 支持 7\*24 连续交易。由于"一键, because of the 操作和维护"功能，manually every day操作人员无需每天手动停止和启动系统feature。此功能允许添加交易中心以扩大业务，而无需任何额外的操作或维护成本。

目前，只有CTP实现了支持FENS机制的多个活动交易中心的"一键开关"。如果case 一个交易中心崩溃，系统可以立即切换到备用交易中心，以达到真正的连续交易的目标。

CTP 已公开并发布了交易 API，用户可以通过该 API 从交易所接收报价并提交submit 交易说明。该接口使用开放接口 （API），该接口已成为期货行业事实上的标准。

CTP 迷你版 （CTP mini） 是一个更快、更轻巧的系统。与CTP相比，它追求更紧凑的配置和更少的资源和设备消耗的目标。C通过 CTP API 开发的the C lient 程序与 CTP 迷你系统完全兼容。

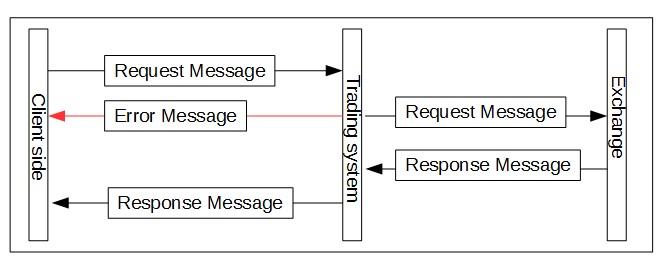
## FTD 通信协议

期货交易数据交换协议（FTD）用于期货交易系统与其下属交易客户 and 之间的数据交换和通信s。本章介绍了FTDs协议中的通信s模式和数据流。

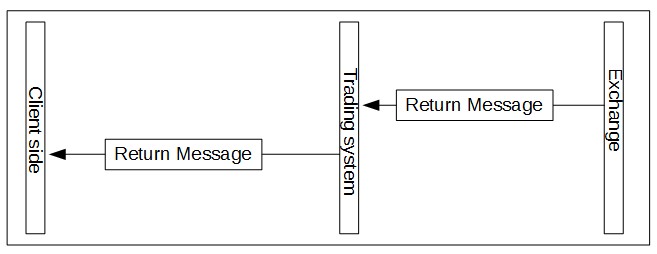
### 通信模式

FTD 协议内的所有通信都基于某些通信模式。每种通信模式都是双方相互合作的一种方式。 FTD 协议使用以下三种通信模式：

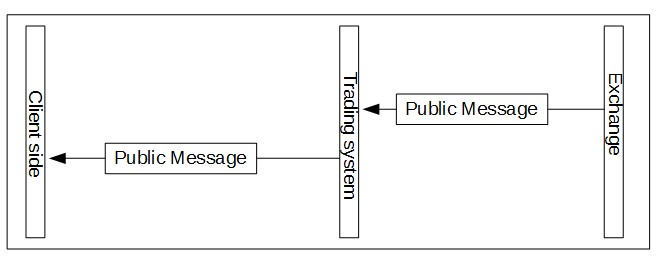
**对话框通信** 模式是 通信请求由 客户端程序启动。 请求 （例如工具 查询）由交易系统接收和处理，响应将发送回 客户端程序。 此 通信模式类似于通常的客户端/服务器模式。



**私人通信**模式 that 是交易系统向特定客户发送信息（例如交易ly回报）主动。



**广播通信**模式是the 交易系统向所有连接的客户端发送相同的信息（例如报价数据）。



通常，在对话通信模式下返回的消息称为**响应，,** 而在私有通信模式下返回的消息称为 **Returns。**.

### 数据流

在FTD协议中，我们需要区分两个重要概念，即通信模式和数据流。数据流是一个单向或双向、连续、唯一、datagrams完整的数据报序列，而通信模式是数据流的交互式工作模式。每个数据流对应于一个通信模式，但一个通信模式可以有多个数据流。

一种类型的通信模式只能由一个 数据流 构造 ， 从而生成对话流、专用流和广播流。一 种类型的通信模式也可以 由 多个数据流构建。例如，用户可以在对话通信模式下建立查询流和交易流，或在广播通信模式下建立通知流和报价流。FTD 仅指定 哪个数据报在哪种通信模式下工作，但未指定数据流的划分。

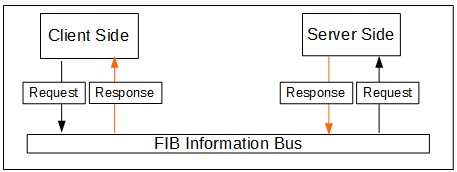
不同的通信模式 以不同的方式 管理数据流。在对话通信模式下，一个数据流是一个连接。消息的完整性和顺序在连接中保持。断开连接后 ， 重新连接将启动新的数据流， 该 数据 流 与原始数据 流完全不同。如果一个客户端在收到响应 之前发送请求并断开连接，则重新连接后不会通过新数据流接收响应。

对于私有通信和广播通信模式，一个数据流对应于特定功能在一个交易日的所有连接。除非指定，否则客户端将从 will 上次传输恢复，而不是从开始恢复。

## 两种数据交换 模式

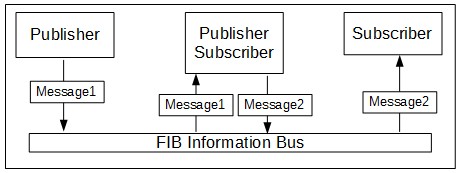
### 请求/响应 模式

客户端程序（客户端）向服务器（服务器）发送请求。服务器在收到请求后处理该请求， 并将 结果发送 回客户端。

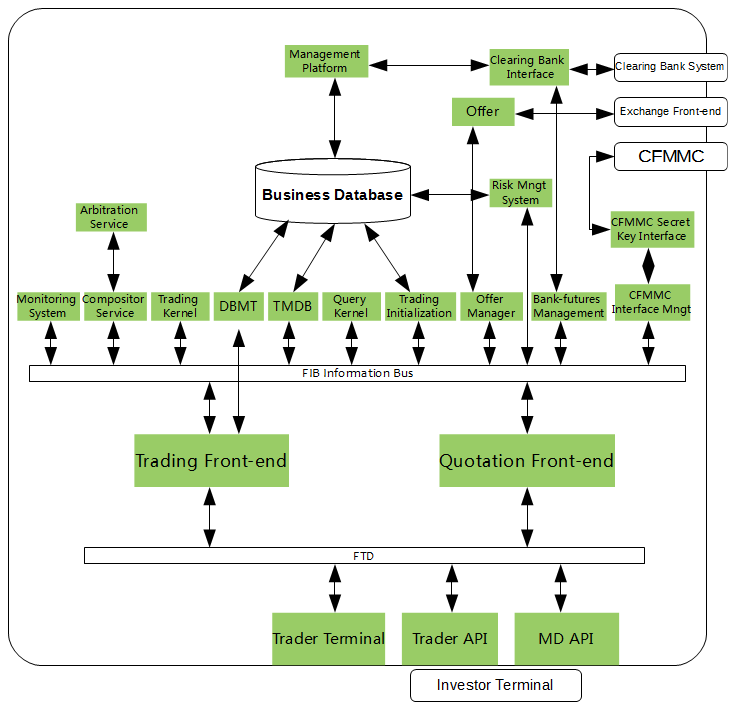


### 发布/订阅模式

发布/订阅模式是异步传输模式。发布者将消息发布到主题。订阅者订阅来自th ose 主题 的消息。发布商和订阅者相对独立。它们无需直接 相互连接 来传输消息。一个 FIB 应用程序可以是发布者和订阅者。



## CTP 架构



**投资者终端**是一个交易客户。它使用 CTP的 trade API 和报价 API。它提供接收报价、交易、银行期货转账等功能。

**贸易r**  **Terminal**终端使用 CTP 的s UserAPI。它为期货公司提供订购、银行期货转账、查询交易数据等功能。T他用户APIfor 是期货公司，它可以控制的数据范围大于交易API。考虑到数据安全性和特权划分the ，UserAPI目前不会向期货公司和投资者披露。（观众不应该知道交易者终端或用户API）)

**FTD** 是期货交易数据交易所协议。

**交易前端**通过TCPto 协议连接到交易客户端，通过FIB to连接到其他后台服务。交易前端负责沟通，与业务irrelevant to 无关，因此，我能够平衡负载，降低the 复杂性，提高the 交易系统的安全性。它主要有三个功能：链路管理、协议转换和数据路由。

**报价前端通过FIB**从报价the Offer Manager 管理器订阅所有报价数据，并转发报价数据给交易客户，交易have 客户通过TCP连接订阅了某些合同的报价。

**FIB（期货交易所信息B我们**）是交易系统的通信基础设施。它为上层应用程序提供数据封装、请求/响应通信模式和发布/订阅通信application模式。

, and监控系统从交易系统获取部分 the 交易数据，以监控data应用数据，并监控物理组件的容量、性能等。

**仲裁**服务管理合成器the 服务的状态开关。

**Compositor** 服务负责序列化事务请求和将事务序列发布为交易 内核的 数据源。

**交易**内核根据投资者的头寸、订单、交易和基金存款和提取计算实时基金和头寸，用于交易前风险管理。 同时，它验证所有订单，驱动交易所报价，并发布实时交易结果到FIB。

**查询内核**具有与交易内核相同的内存logic 数据库结构和业务逻辑实现。基于投资者的实时计算，查询内核更新内存the 数据库，通过FIB为交易客户端提供s查询服务。

**DBMT**与业务the 数据库实时交互。它通过交易that 前端向交易内核发送需要更新的业务数据。

results Tmdb 通过 Fib订阅 trading kernel 的结果。它实时将相对业务back 数据写回物理数据库中，在交易时间后进行结算使用。

**交易初始**化有major 两个主要功能：1.GG为the 基于数据库的交易内核’初始化提供必要的数据。2. S结束对系统的交易准备指令，开始新的交易会话。

**报价经理**管理交易和报价。它允许allows 交易内核避免交易和交换前端Offers 之间的复杂通信过程，从而简化交易内核的处理逻辑。

**报价**实现the 交易所的报价API和贸易API。. It 它通过交易所提供的远程, 交易席位下订单、接收订单退货、交易the 回报、报价数据等。

**风险管理系统**获取交易内核发布的交易the 序列和交易s结果，实时监控交易数据，同时提供试值风险risk 计算和强制仓位清算功能。

**银行期货**管理用于管理 manag银行期货转账接口。

**清算银行接口（银行期货接口**）实现银行期货转账API，from each bank, and 并在交易系统和银行系统之间提供数据交换通道。

**期货保证金监控中心（CFMMC） Secret Key Interface**密钥接口用于从期货保证金监控中心查询期货公司的秘密密钥。

**期货保证金**the **监控中心(（CFMMC） Interface Management** 接口管理用于管理期货保证金监控中心提供的密钥接口。

**业务** 数据库为整个系统提供 物理数据存储。

**管理** 平台为期货公司 提供 各种业务运营条目。

# Api

## 介绍

我们的API由the Trading 系统 API、风险the API 和结算the API （CSV） 组成。通过使用三个 API，用户可以与 CTP the 系统交互，to以处理交易、风险控制和结算数据存储。

Trading系统API用于接收交易所的报价数据和发送交易 subscribing quotations指令，如订阅报价、an order, canceling an 下单、取消订单、下a parked order, 单、处理期货银行转账querying 、查询信息等。贸易 API 是使用最广泛的接口。它的目标用户主要是客户软件提供商providers （如Q7）和个人，机构和自营投资者 investors ，对交易客户有特殊要求。

风险 API用于获取 to 实时数据（如资金、头寸、保证金等），以便对投资者的风险进行试验计算，并提供必要的强制头寸清算功能，以减轻风险。为期货公司的风险控制人员提供风险披露和风险管理。

由SFIT 为期货公司风险控制人员提供的 RcWin平台是使用风险 API 开发的。由于风险 API 管理的数据涉及所有投资者，因此风险 API的目标用户是具有风险管理特殊要求的期货公司，而不是个人投资者。

结算 API由一系列数据库指令组成。它部署在可以访问 computers CTP 物理the 数据库的计算机上。通过执行相应的指令，运营商可以获取相关数据，如投资者信息、交易、仓位等。主要目标用户也是期货公司，而不是个人投资者。

（此处应提及风险 API 和结算 API？）

### 接口文件

开发人员可以向期货公司咨询贸易 API，或从我们的网站下载官方贸易 API。这两个版本可能不同。

官方网站地址： [*http://www.sfit.com.cn/*](http://www.sfit.com.cn/)

API 文件列表：

|  |  |
| --- | --- |
| **文件名** | **文件描述** |
| Thostftdctraderapi. h | 贸易界面C++头er 文件 |
| Thostftdcmdapi. h | 报价接口C++头 erer文件 |
| Thostftdcuserapistruct. h | 定义所有数据结构 |
| Thostftdcuserapidatatype. h | 定义所有数据类型 |
| 托斯特拉特拉皮. dll | 交易界面的动态链接Trade库。 |
| 托斯特拉特拉皮. 利布 |
| thostmduserapi. dll | 报价接口的动态 链接 库。 |
| thostmduserapi. lib |
| 错误. dtd | API 错误代码和信息（xml 格式）。 |
| 错误. xml |

## 一般规则

### 命名约定

|  |  |  |
| --- | --- | --- |
| **规则** | **格式** | **例子** |
| 请求 | 要求---- | ReqUserLogin |
| 响应 | OnRsp---- | OnRspuserLogin |
| 查询 | ReqQry---- | 要求解释 |
| 查询 的响应 | OnRsqry---- | OnRsqry 仪器 |
| 返回 | Onrtn---- | Onrtnorder |
| 错误返回 | Onerrtn---- | Onerrtnorderinsert |

### 接口类

**Spi**  **------Spi（例如CThostFtdcTraderSpi）)**包括所有响应和返回函数。开发人员应派生接口并实现相应的虚拟函数。

**------Api（例如CThostFtdcTraderApi）)** 包含用于主动发送请求和订阅的接口。开发人员可以直接调用它。

### 一般参数

**n请求 ID：**客户端在发送请求时应指定请求 ID。交易trade API 将在响应或返回中返回相同的请求 ID。当客户端频繁操作时，可以连续多次调用一个响应方法。在这种情况下，请求和响应与请求correlated with the ID 相关。

**IsLast**指示当前the 数据包是否是与 nRequestID有关的最后一个数据包，如果将大数据包划分为多个数据包。在这种情况下，将多次调用响应方法。例如，如果有一个数据包，则为true的响应为最后。如果有两个数据包 packets，则第一个响应的 IsLast 为 false，而第二个响应的IsLast 为 true。

RspInfo 指示执行过程中是否有错误。ErrorId 0 表示交易内核已成功处理请求。否则，此参数指示交易内核返回的错误类型。

错误类型的所有可能 值都在 错误.xml **中描述。**

### trade API 的一般初始化步骤

以下步骤描述了创建the 和启动the 交易系统的 trade API 和报价 API of the trading system 的过程，并启动交易API的工作线程。

目前，文档中未涵盖风险 API 和 结算 API。

1. 创建"SPI"和"API"实例

"SPI"是一个用户定义的类，它派生SPI接口（CThostFtdcTraderSpi或CThostFtdcMdSpi）。 此处的 "API" here是界面中的 CThostFtdcmdApi or 或CThostftdcTraderApi。

1. 将 SPI实例注册到 API 实例。
2. 注册相应的API的不同前端网络corresponding API地址。请注册交易 API 的交易前端地址，而报价 API 的前端地址。
3. 订阅公共流（ 仅适用于 交易 API，不需要报价API）以接收公共数据，如市场 交易状态。默认情况下，由交换发布的数据从 上次 断开连接（恢复模式）恢复。开发人员还可以指定从开始（ 重新启动 模式）或从登录时间（快速模式）接收。
4. 订阅专用流（仅适用于交易 API，不需要报价 API）以接收私有数据，如订单回报s。默认情况下，由交换发布的数据从上次断开连接（恢复模式）恢复。开发人员还可以指定从开始（重新启动模式）或从登录时间（快速模式）接收。
5. 初始化（初始化）。
6. 等待 接口 线程终止（加入）。

有关充足的ample代码和更多详细信息，请参阅"开发演示"章ter。

开发演示

下一部分将介绍使用交易API和交易系统the 报价API的步骤和要点。附加的代码片段只是供参考 your。

本文档不解释编程详细信息，也不包含 图形用户界面 开发指南。

# 收购报价演示

报价 API 易于使用。通过使用它，开发人员可以快速了解 CTP API 的基本用法。

## 制备

到目前为止s tthe ，, 我们已经发布了CTP的tradeAPI API 的Windows，Linux， Windows, Linux安卓和iOS。本指南以the C/C++的 Windows as an 版本为例。

以下是三个推荐的 C/C++ IDEs：

* **Visualstudio**
* **代码块**
* **QT 创建者**

**导入 API 文件**

开发人员应将前面描述的所有 API 文件复制到其项目的目录中，并导入所有头 er 文件和静态或动态链接库。

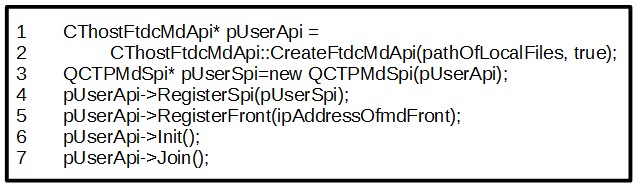
**实现 SPI 接口**

开发人员应派生**CThostFtdcMdSpi** 接口并实现必要的虚拟函数。

|  |  |
| --- | --- |
| 正面连接 | 此函数在 建立到服务器的连接建立 后调用 ，但在 登录到交易系统之前调用。  通常，它是 您将登录代码放在的地方。 |
| 正面断开连接 | 客户端断开服务器后调用此函数。  API 将尝试自动重新连接。  此函数 的参数 指示连接 失败 的原因。 |
| OnRspuserLogin | 从 the 登录请求返回 响应。  错误消息，如"il合法 登录"通常意味着错误的密码。 |
| OnRspSub 市场数据 | 从订阅 市场 数据的请求返回响应。  通常，当 订阅 市场数据的请求发生错误时，会调用它。  误差原因和仪器可以在参数中找到。 |
| OnRsunSub 市场数据 | 从取消 订阅 市场数据的请求返回响应。 |
| OnRtnDepthMarketData | 返回特定 工具 的报价。  如果请求没有发生错误，则调用此函数，而不是 OnRssubMarketData |
| OnRserror | 当 CTP 无法识别客户端发送的请求时，将调用此函数。  例如，如果开发人员将无效的 IP 注册到 前端 ，他们可能会收到通知。 |
| OnheartBeat 警告 | 如果在预设的一段时间内 未 收到消息，则调用此函数。 它可用于发送心跳，以表明客户端和服务器之间的连接仍然存在。  接口不再有效。 |

（缺行情接口工作原理简图）

## 报价 API 的初始化



**第 1 行和 2 号线**

使用 CreateFtdcMdApi 方法创建 the **CThostFtdcMdApi** **CThostFtdcMdApi**的实例。第一个参数是本地流文件的目录。流文件是由报价 API 或交易 API生成的本地文件，使用.con进行后缀，which record 它记录客户端接收的所有数据包的数量。第二个参数指示传输模式为 TCP（false）或 UDP （truetrue）。

如果开发人员需要使用多播报价，他们也应该设置第三个参数 **bIsMulticast**。只有当第二个和第三个参数都设置为 true 时，报价数据才能通过多播传输。

**注意：多播报价只能在 Intranet 内 使用。**

有关流程文件的详细介绍，请参阅"附录"一章。

**第 3 行**

创建 nSPI 实例。 QCTPMDSpi 是派生sample **CThostFtdcmdSpi 的样本实体类。**

**第 4 行**

将SPI 实例注册到API实例。

**第 5 行**

将前端地址注册到API 实例。前端地址就像tcp://127.0.0.1:17001。 "tcp" 是起始字符串， 不是通信模式， 无法更改。"127.0.0.1"” is the address 是报价前端的地址。"17001" 是报价前端的端口号。

**注意：使用CTP API，报价数据可以通过三种方式传输：TCP、UDP和多播。一个前端地址可以使用三种方法之一。不需要为to每个e前端地址设计一个传输地址。**

**第 6 行和 7 号线**

这是 用于初始化报价 API 的工作线程。初始化后，线程将自动启动，并使用最后一步中注册的地址 连接到 服务器。

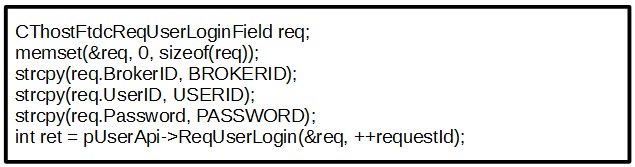
所有 CTP API 都有独立的工作线程。在开发GUIprogram程序时，开发人员应注意线程之间的潜在冲突。

## 登录

在fter初始化the 中，报价工作线程将使用已注册的前端地址与服务器建立无n 身份验证the 的连接。连接后，**将调用 OnFrontConnect。** 如果在程序执行期间未连接或断开连接，**将调用 OnFront** 断开连接。

连接后，客户端程序可以使用**ReqUserLogin（或****(or**  **ReqUserLogin2）请求登录**。关键数据结构是**CThostFtdcReqUserLoginField。**.

注意： ReqUserLogin2（登录请求enhanced2）d request ReqUserPasswordUpdate2（用户密码更新请求2）是两个增强的 APIs，它要求 CTP后端版本为 P20170915 或更晚。



**BrokerID**是期货公司的成员 ID。

**UserID**是期货for公司投资者的客户代码。

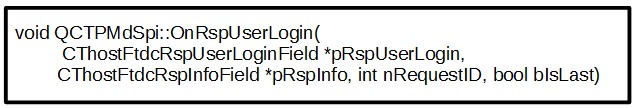
密码是投资者的密码。

此函数返回的整数值 （示例代码中的 in the sample codewhether whether was was **ret）** 指示请求是否已成功发送，而不是服务器是否处理请求。

**返回值**

* 0， 已成功 发送。
* -1， 发送失败;
* -2， 未处理的请求超过允许 的 卷;
* -3， 秒发送的请求数超过允许 的量。

**CTP API 中大多数请求的返回值与上述值相同。**



s the response via the 成功接收登录请求后，服务器验证 BrokerID、UserID UserID 和密码，然后通过 **OnRspUserLogin**方法返回响应。

开发人员可以通过第二个参数successful via the **pRspInfo**中的 ErrorID判断**登录是否成功**。如果 ErrorID 为 0，则登录成功，否则失败。

通常，错误消息"il合法 登录"将返回密码错误。

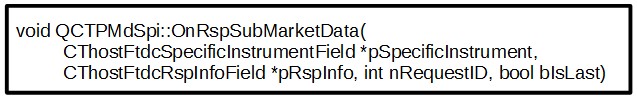
成功登录后，第一个**参数 pRspUserLogin**包含来自服务器的一些基本数据，’s如会话SessionID, ID、frontID、maxOrderRef 和每个交换服务器上frontID, 的时间。

## 订阅报价数据



成功 登录 后，客户端 现在可以 订阅报价单。

订阅 **市场数据被** 调用来 订阅报价。第一个参数是要订阅 的所有协定的数组。第二个参数是此数组的长度。





订阅报价后 may ，可以调用 OnRsSubMarketData 或 OnRtnDepMarketData。 s be invoked.

**OnRspSub 市场数据**

如果订阅请求是合法的legitimate，则通过此方法将返回错误消息 （ rspInfo ）。即使请求是合法的，也会调用此函数，返回的消息将是"CTP：无错误"。

**OnRtnDepthMarketData**

如果订阅请求 是合法的，服务器将 返回 报价单， 频率 为每天 **两次。**

ReqUserLogout and 和取消订阅市场数据to log用于注销和取消订阅e 报价。 用法与登录和订阅ing报价单 quotation相同。

**注意：目前，如果通过**  **ReqUserLogout**注销**，则 APIging 将断开当前连接。重新登录后， API 将建立新的连接.T他会话 ID** **restart将被重置，** **MaxorderRef** **将从 0 重新启动。**

## 订阅和接收报价请求

做市商可以通过报价API认购投资者的报价请求s（ **询**价）。

目前，交易所采取不同的方式处理投资者的询价和来自做市商的认购。在处理报价、询价等函数时，它们实现的方式不同。考虑到开发效率，CTP实现了投资者the functions for使to make 询价和s做市商在报价API中接收询s in the 价的所有功能。

**四个交易所 的询 价实施**

**SHFE &CFFEX**

询价流通过交易API传输。交易API为投资者提供了接口，使to make 询价和做市商获得询价s。做市商无需认购，即无需认购，即获得询价s。当做市商使用交易 API 登录时，服务器会识别他们的账户，并根据他们在交易所中的特权发送相应的合约的询价。

**德塞和捷克**

询价流通过报价API 传输。投资者仍通过交易API进行 via the 询价，而做市商应在to 服务器传输询价s流之前认购投资者的询价。

CTP API 坚持与不同 交换保持一 致 的原则。如果交易所改进了处理询价的实现，CTP API将 相应地 进行调整。

**API 方法**

报价 API

|  |  |  |
| --- | --- | --- |
| **请求** | **响应/返回** | **描述** |
| 订阅报价 | OnRspSubforQuoteRsp | 做市商认购投资者的询价s |
| 取消订阅报价 | OnRsunsubforQuoteRsp | 做市商取消认购投资者的询价s |
|  | OnrtnForQuotersp | 做市商获得投资者的询价 |

贸易API

|  |  |
| --- | --- |
| **请求/返回** | **描述** |
| OnrtnForQuotersp | 做市商获得投资者的询价 |
| ReqFor报价因特因特 | 投资者发送 询 价请求 |

报价 API 中订阅询价 in the 的用法与订阅报价单相同。在贸易 API 中，无需订阅。

如何为投资者制作询价将在下一章中the next “Demo of Trading” Chapter详细描述。

## 报价推规则的解释

CTP 报价 规则 为：

√两次报价snapshot快照s

√报价更新时才进行推送 happens

√ 连接后推送初始报价

CTP 报价推送的基项是按秒推两次，但推送时间（in millisecond以毫秒为单位）并不严格ly 000=500。也就是说，推送时间可能是300~800 。

原因有二：一是交换的引文（quotations from exchanges (以毫秒milliseconds为单位）不是常数;二是换方的引文不是常数。;另一种是，推送仅在报价更新发生时发生。

此外，在客户端通过connected toCTP API 连接到CTP 后，CTP将按常规the推送最新报价，以便在当前市场有任何报价时通知any 客户。

**[通知]：交易所在开盘前的拍卖s和匹配期内也会推送报价。开盘时报价的时间戳可能等于could probably** 或**大于equal to**  **500，或小于 500。**

**[通知]**  **：在互联网**环境中，**is update在打开 ing s** **后，CTP**将推动**合约**的状态**（例如，a 合约的状态**更新**为 to** **continuous trading after open**开盘后连续**交易），并且此消息的推送时间**与开盘后的第一个报价时间几乎**相同。**

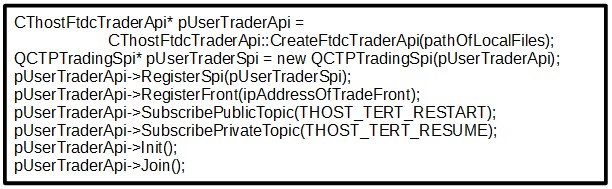
# 交易演示

与with 报价 API 相比，the 交易 API 更强大、更复杂。交易 API 提供在that 投资者交易过程中需要调用的方法，例如。g.下订单 an、取消订单、进行making 银行期货转账、查询信息等。

其用法与报价the API 类似。

## 贸易forAPI的初始化

就像报价 API 一样，开发人员应该从 from the **CThostFtdcTraderSpi 类派生**，并实现必要的虚拟函数。



步骤和代码几乎与报价 API（3.2as those of the 节）的步骤和代码相同。

**差异：**

1. 创建an API 实例时，不需要指定传输s协议，因此第 2 行中的方法只需要一个参数（流文件目录）。
2. 需要订阅公共流和私有流。

Public Stream 包含交易所发布的所有客户信息，例如合约在市场上的交易状态。

专用流包含交易所发送给指定客户的信息，例如订单退货、交易回报。

**T有三种订阅模式：**

* 重新启动：从交易所在当前交易日发送的同一类型的第一s条消息重新传输。
* 恢复：通过恢复和从上次传输继续 进行重新 传输;
* 快速：从当前登录进行传输。

1. 要注册的前端地址是the 交易前端地址。

## 登录

**认证**

在 登录到 交易系统之前， 如果配置， 需要身份验证。只有在成功进行身份验证后才能请求登录。

认证可以在期货公司业务人员使用的结算平台中进行配置。可以将其配置为be 关闭ed，以便不需要对客户端to 进行身份验证。否则，期货公司需要维护结算平台中客户程序的身份验证代码 （ AuthCode .

身份验证使用 ReqAuthenate（ 请求身份验证） 和 OnRsAuththate（ 从服务器返回身份验证响应）方法。

**登录**

请求在交易 API 中登录与报价 API 中的登录相同。

and the 成功登录后 parameter in successfully, the ，OnRspUserLogin 方法中的 pRspUserLogin 参数将包含前 ID （FrontID）、会话 ID （会话ID） 和最大订单引用号（MaxOrderRef）。

* FrontID是客户端连接到的前端的toID。
* 会话ID 是客户端和前端服务器之间连接的会话 ID。
* MaxOrderRef是当前会话的最大 OrderRef 编号。OrderRef OrderRef是一个会话中订单的唯一标识号。如果客户端未将值分配给 OrderRef，, 服务器将自动分配它。如果客户端分配一个值，则该值将从 MaxOrderRef 递增，以避免与其他订单重复。

**注意：目前，如果用户使用ReqUserLogout** **注销，则当前连接将断开连接。重新登录后，将建立新连接，重置会话 ID， will be reset and the** 从 0 开始**MaxOrderRef。**

## ****确认结算信息****

stors to confirm the settlement info of the prior 为了让投资者及时、准确地了解其交易statuses状态（如可用资金、头寸、保证金占用等），并了解其风险状况，CTP要求投资者在首次登录当前交易日时，确认前一个交易日的trading day, when they log in on current trading day结算信息。

**结算确认方法**

|  |  |  |
| --- | --- | --- |
| **请求** | **答复** | **描述** |
| ReqQrySettlementInfo | OnRsqrySettlementInfo | 查询结算信息 |
| ReqSettlementInfo 确认 | OnRspSetlementInfo 确认 | 确认结算信息 |
| ReqQrySettlementInfo 确认 | OnRsqrySetlementInfo 确认 | 查询确认日期 |

## 

## 位置计算

查询位置的sAPI 如下：

|  |  |  |
| --- | --- | --- |
| **请求** | **答复** | **描述** |
| ReqQry投资者职位 | OnRspQry投资者位置 | 查询 位置 （摘要） 信息 |
| 要求QryInvestor 位置Detail | OnRspQryInvestor 位置Detail | 查询位置详细信息 info |
| ReqQry投资者位置组合Detail | OnRspQry投资者位置组合Detail | 查询 位置组合详细信息 信息 |

**[位置详细信息]**

CTP根据交易所的犯罪记录生成仓位详细信息 trading recordsrecorsd s; ; 每个记录对应于一个仓位细节 record。

**[位置 摘要]**

CTP根据合约 the 、仓位方向、开仓日期for（仅适用于SHFE，旧仓位和新仓位不同）汇总仓位详细记录。

在位置 摘要中：

**Yd仓位**量在昨天收盘时间（≠当前昨天仓位量，静态，不因当天开盘和收盘而变化）

**定位** 当前 仓位金额

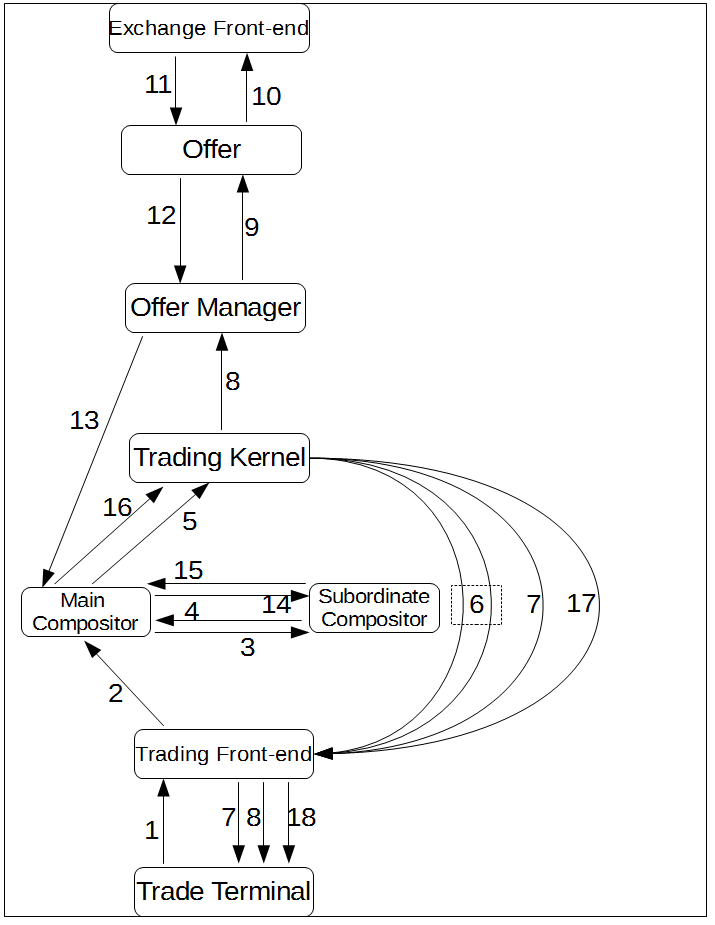
**今天仓位** today今天的新s开仓量

Current 昨天的位置 = ∑位置 - ∑今天位置

**[位置 组合详细信息]**

仅适用于组合合同的' 位置详细信息记录s

## 订单 处理 流程



1. 交易终端通过交易API向 Trading Front-End 提交订单请求。
2. 主合成器从交易前端 订阅交易请求。
3. 主合成器将事务序列发送到下级合成器进行确认。
4. 收到要确认的数据包后，下级合成器将 相关 数据包写入流文件并立即返回确认消息。
5. 交易内核从主合成器订阅交易序列。
6. 交易 内核 在收到交易顺序后对其进行验证。如果出现任何错误 ，交易 内核将返回一个订单响应，并收到错误消息到交易前端。然后交易前端立即将消息转移到交易终端。如果 请求 是 合法 交易请求， 交易内核 也将返回对交易前端的响应，但此响应不会返回到交易终端。
7. 此步骤有两种情况： 1. 如果交易前端收到错误订单响应， 它将通过对话框模式将包发送到交易终端。2. 如果响应消息正确，交易前端通过专用模式立即返回相应的订单到交易终端（步骤8，从交易前端到交易终端）。
8. 此步骤有两个过程：1。收到来自交易内核的订单返回后，交易前端通过专用模式将订单返回发送到交易终端。2. 将第一个订单返回到交易前端后，交易内核将生成a请求包，请求订单插入离子并将其发送到to the 交易所。包裹将过帐到优惠管理器。
9. 收到订单插入离子ion请求后，报价管理器将包裹转发到相应的要约Offer。
10. 从报价经理那里收到包裹的fter，即他通过交易所the 的贸易API将订单发送到s 交易所。
11. 优惠通过交易所的交易 API 从交易所的交易前端接收订单回报、 交易回报或错误订单响应。
12. 优惠收集所有订单回报，交易者回报和订单响应 从 交易所，并感性他们 到报价经理。
13. 主合成器从receives 产品/服务管理器接收订单包。
14. 序列化包后，主合成器将事务序列发送到下级合成器进行确认。
15. 收到要确认的数据包后，下级合成器将数据包写入流文件并立即返回确认消息
16. 交易内核从主合成器订阅所有订单和交易回报。
17. 交易前端订阅所有交易结果从交易 内核。
18. 交易前端将交易结果分发到每个交易终端。

## 处理订单的方法

**请求订单插入**离子（ion上面的步骤1）。 ).

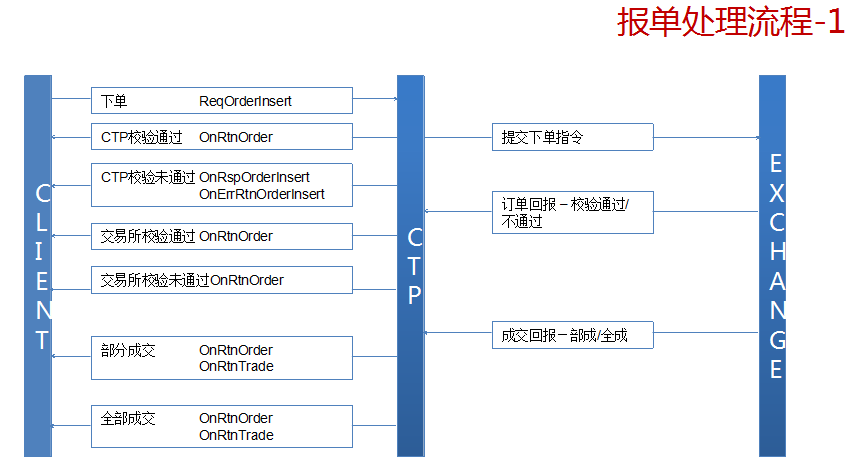
**OnRsorderInsert：**：返回包含交易内核an 返回的错误消息的订单响应（步骤 7 中的第一个案例）。

**OnRtnOrder**： 从交换返回订单状态。每次更改订单状态时，此方法将调用一次。在一个订单过程中，例如，当交易系统向交易所提交the 订单时（步骤 8中的第二个流程），当交易所取消或接受订单时，当订单被填满时，当订单被填满时，当订单被填满时，当交易系统被调用几次时，当交易系统将订单提交到交易所时，当交易所取消或接受订单时，当订单被填满时，等等etc。

**OnErRtnOrderInsert：**当交易所收到交易内核从要约验证 from 的订单 Offer时，交易所将再次验证订单。如果订单是合法的il，交易所将取消订单，向报价发送错误消息并返回最新的the订单状态。客户端收到错误消息后，将the 调用 OnErRtnOrderInsert 方法，并将通过 order status (canceled OnRtnOrder 方法返回最新的订单状态 （已取消）。如果订单是合法的legitimate，则只返回订单状态（未触发）。

**OnRtnTrade**：：如果订单已满，交易所将返回 the 已填充的订单状态，并将通过此方法返回交易信息。

订单填写后，订单退货 （OnRtnOrder） 和交易退货 （OnRtnTrade） 将发送给客户。订单返回中的订单状态将被"填充"。但由于CTP的交易内核只only 在收到交易回报后更新订单状态，我们建议clients 客户使用交易回报来判断订单是否被填充。例如，如果客户端收到具有"填充"状态的订单返回并立即发送平仓指令，则平仓仓位操作可能会失败。原因是交易内核在交易回报前收到平仓指令，且交易内核中尚未“"”填充"订单状态。

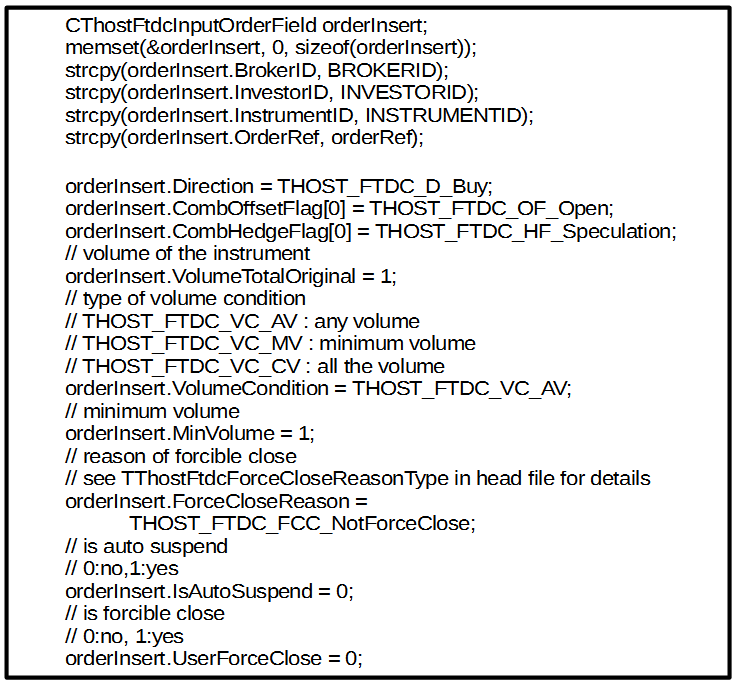




## 下订单

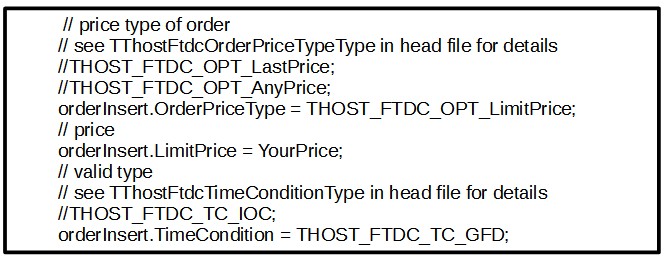
ReqOrderInsert 方法用于下订单。主要数据结构是CThostFtdcInputOrderField。.

一般分配代码如下：

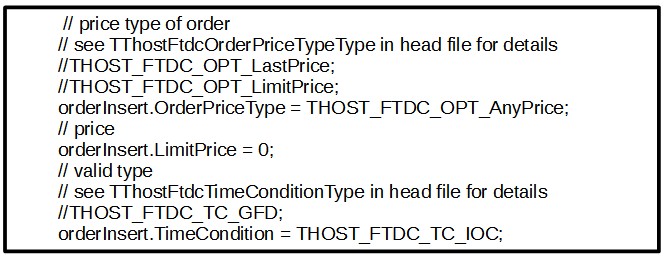


应为相应的字段指定具有不同值的不同订单。

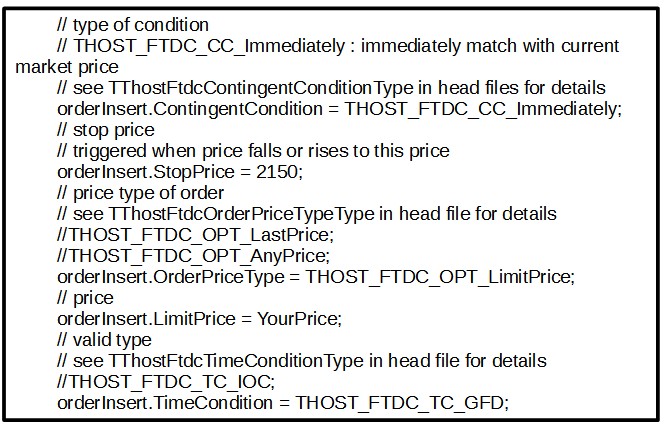
**限价单**



**市场订单**



**条件顺序**



**条件订单介绍**

条件订单是一种订单，仅在满足指定条件时才执行。条件可以基于最新报价单或the 时间。例如，如果投资者有一个 IF1410 的空and 头头寸，并且他希望买入一个空头头寸，以便在价格低于2200时结束仓位，那么他可以下有条件的订单。当the 报价数据波动且满足条件时，将自动发送订单。通过使用条件顺序，他不需要盯着电脑来监控报价。为了更有效地使用有条件订单，投资者可以提交止损和限价订单和市场交易。

### FOK & FAK

**FOK（填充或终止）**是一种特殊的条件顺序.交换部门接受订单后，如果订单目前能够完成，订单将立即完成;否则整个订单将 被取消。

**FAK（填充和杀死）**也是一种特殊的条件顺序.交换接受订单后，如果此时可以部分filled填写订单，则订单的一部分将被填写the ，订单的剩余数量将立即取消。

CTP API 通过字段组合实现 FAK和 FOK 指令。

|  |  |  |
| --- | --- | --- |
|  | Fak | 育种 |
| TthostFtdcorder 价格类型 | THOST\_FTDC\_OPT\_LimitPrice | THOST\_FTDC\_OPT\_LimitPrice |
| Tthostftdctime 条件类型 | THOST\_FTDC\_TC\_IOC | THOST\_FTDC\_TC\_IOC |
| TThostFtdcVolume 条件类型 | THOST\_FTDC\_VC\_AV /  THOST\_FTDC\_VC\_MV | THOST\_FTDC\_VC\_CV |

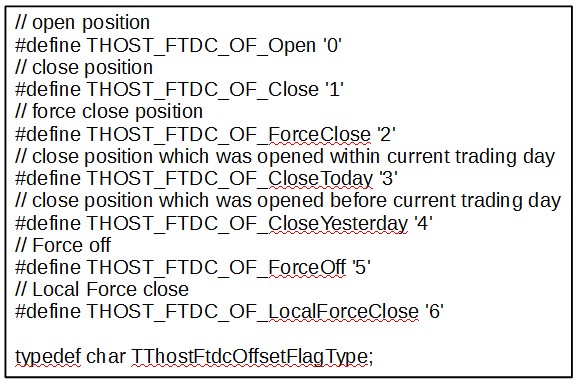
**最小卷 字段**

对于 FAK指令，THOST\_FTDC\_VC\_AV表示任何卷，THOST\_FTDC\_VC\_MV表示最小卷。如果THOST\_FTDC\_VC\_MV，投资者应指定需要填充的最小交易量。如果可以填充的卷小于 MinVolume，则整个订单将被取消，并且不会填充任何卷。

**组合集火焰 字段**

CombOffsetFlag 用于指定订单的偏移属性（开盘、收盘、关闭）。

IMG_256

该字段是一个 5 个字符的数组。它可用于描述常见订单和组合 al 订单的订单偏移属性。对于通用订单，只应指定第一个项目，而对于组合订单，应指定第一个和第二个项目。有关允许的值，请参阅 ThostFtdcuserApistruct.h。 

s SHFE的仓位分为今天仓位s（s今天开盘）和昨天仓位s（历史仓位）。平 closing仓时，必须指定指令whether 是今天平仓还是昨天平仓s。

如果使用THOST\_FTDC\_OF\_Close THOST\_FTDC\_OF\_CloseYesterday SHFE 的位置，则结果与THOST\_FTDC\_OF\_CloseYesterday。如果将THOST\_FTDC\_OF\_CloseTHOST\_FTDC\_OF\_CloseToday or THOST\_FTDC\_OF\_CloseToday THOST\_FTDC\_OF\_CloseYesterday或数据用于其他交换，则结果与THOST\_FTDC\_OF\_Close。 .

**注：目前，四家交易所都是先开，平仓时先平仓。除了第一个 and 开盘规则和第一个**收盘规则**外，DCE 和 CZCE**   **还有**先**关闭 the 普通订单的规则，然后是the 组合的 al** **订单。**

### 订单序列号

在CTP和交换系统中，每个订单有3组唯 一 的序列号， 以避免重复。

**正面= 会话** **ID**  **= 订单参考**

登录后，交易内核将返回 FrontID and 和当前连接的s SessionID。这两个值在这方面是不变this的。

OrderRef 是the CThostFtdcInputorder 字段结构中的一个字段。为了确保唯一性，开发人员可以在一个登录会话中，从 MaxOrderRef 对每个订单进行一次递增。如果未指定，交易内核将自动为其分配唯一值。

客户端程序可以维护自己的序列号，并可以随时 **通过这组** 序列号取消订单。

**交换ID +交易者**ID **+ 订单本地 ID**

交易内核将订单提交到要约管理器后，交易内核将生成 OrderLocalID 并返回给客户端程序。 ExchangeID 是订单合同 that 所属的交易所代码。交易者ID是由交易内核指定的。客户端程序还可以使用此一组序列号来取消订单。

与第一组不同，此组由 交易内核 维护。

**交换 ID = 订单系统 ID**

收到订单后，交易所将为订单生成 OrderSysID。 这组序列号将通过 CTP 返回到客户端程序。客户端程序还可以使用此一组序列号来取消订单。

此序列号组由交换保持。

### 订单类型组合

API 支持五 个交换的所有订单指令。 Various 指令通过各种组合生效。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | 限价单 | 市场订单 | 条件顺序 |
| 订单价格类型 | 价格类型 | THOST\_FTDC\_OPT\_LimitPrice | •任何价格 | 限价价格 |
| 限价价格 | 价格 | DIY 价格 | 0 | DIY 价格 |
| T ime 条件 | 有效日期 | THOST\_FTDC\_TC\_Gfd | 国际 奥委会 | Gfd |
| 音量条件 | 卷类型 | THOST\_FTDC\_ VVC\_AV | Av | Av |
| 应急条件 | 或有条件 | THOST\_FTDC\_ CcCC\_立即 | 立即 | 分配或有条件 |
| 一个条件 | |  |  | 停止价格 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | Fak | 育种 | 停止 - Dce |
| 订单价格类型 | 价格类型 | THOST\_FTDC\_OPT\_LimitPrice | 限价Price | 限价价格 |
| 限价价格 | 价格 | DIY 价格 | DIY 价格 | DIY 价格 |
| T ime 条件 | 有效日期 | THOST\_FTDC\_TC\_IOC | 国际 奥委会 | Gfd |
| 音量条件 | 卷类型 | THOST\_FTDC\_VC\_AV/MV | CV | Av |
| 应急条件 | 或有条件 | THOST\_FTDC\_ CcCC\_立即 | 立即 | 触摸/触摸利润 |
| 一个条件 | | 明卷 |  | 停止价格 |

CFFEX 五范围指令的组合规则：

CFFEX -限制价格 to的任何价格

* THOST\_FTDC\_OPT\_价格Price
* THOST\_FTDC\_Tc\_Gfd
* THOST\_FTDC\_Vc\_Av

CFFEX- 五档 市场价格

* THOST\_FTDC\_OPT\_五级价格
* THOST\_FTDC\_Tc\_Ioc
* THOST\_FTDC\_Vc\_Av

CFFEX- 五档市场价格 以限价

* THOST\_FTDC\_OPT\_五级价格
* THOST\_FTDC\_Tc\_Gfd
* THOST\_FTDC\_Vc\_Av

Cffex - 最佳价格

* THOST\_FTDC\_OPT\_价格
* THOST\_FTDC\_Tc\_Gfd
* THOST\_FTDC\_Vc\_Av

CFFEX - 限价 的最佳 价格

* THOST\_FTDC\_OPT\_价格
* THOST\_FTDC\_Tc\_Gfd
* THOST\_FTDC\_Vc\_Av

### 订单退货

方法onRtnOrder 用于返回订单返回。数据结构是CThostFtdcOrderField。.

订单退货用于通知客户程序订单的最新状态，例如已提交、 取消、不排队和填写等。每次更改订单状态时，将调用此方法一次。

**总成交量和成交量 &** 交易及**成交量总**

以上三个字段是 订单的原始交易量、填充量和剩余未填充的卷。

如果一个订单被 填充多次，则每个交易记录将有一个订单退货。

执行条件订单时，交易内核将验证订单。如果验证失败，将调用 OnRtnError 条件顺序 will be invoked and ，并将返回一条错误消息。

**订单状态**

* 0 完全 填充
* 1 部分填充 ，订单仍在匹配队列中换
* 3 未填写，订单仍在匹配队列中换
* 5 已 取消
* 6 -个未知 - 订单已提交the 到交易所，但尚未收到received确认信息

### 交易回报

方法onRtnTrade 用于返回交易回报。每次部分填充订单或完全填充订单时，将调用此方法一次。交易退货仅包含合约、填充量、价格等信息。它只包含有关交易的信息，但交易后没有投资者的头寸和资金。

ReqQryTradingAccount 方法用于查询投资者资金的最新状态，如保证金、交易费用、头寸利润、可用资金等。

我们建议客户使用交易回报来判断是否填写了n订单，如果是，数量和价格。 如果使用订单回报（状态为"部分填充"或"完全填充"），则有可能发生关闭可能失败，因为订单回报与交易回报之间有理论上的时差（非常边际），而 CTP 的端端会根据交易回报更新订单状态。

**查询保证金率**

查询保证金率的方法是 ReqQry 计算结果。.

它仅用于普通合同。对于组合al 合约，用户可以查询组合the al 合约的两条腿的费率，然后根据the 交易所规则计算’费率。

## 下 停车订单

保留订单是唯一可以在非交易期间（在呼叫拍卖之前或在交易时段之间的 the 休息时间）提交到交易所的订单类型。下一个交易日开始时，已停放的订单the 将重新交易。用户可以同时放置停放订单s 和停车取消订单。.

触发已停放订单（已停放订单插入）时，新订单将提交到交易所。

当触发已停放的取消订单（停车订单操作）时，订单取消将提交给交易所以取消现有的订单。

ReqParkedOrderInsert 方法用于下停放的订单。主要数据结构是CThostFtdcParkedOrderField。. The usage of ReqParkedOrderInsert 的用法与the ReqOrderInsert 类似。其响应方法是 OnRsparkedOrderInsert，它返回来自交易内核的响应。

ReqParkedOrderAction 方法用于放置停放的取消订单，相应的响应方法是 OnRsparkedOrderAction。

ReqRemoveParkedOrder 方法用于删除已提交但未触发的停车顺序。

ReqRemoveParkedOrderAction 方法用于删除已提交但未触发的停车取消订单。

ReqQQryParkedOrder 和 and ReqQQryParkedOrder 操作方法用于查询已停放的订单和停放的取消订单。他们的响应方法是 OnRsqryParkedorder 和 OnRsqryParkorder 操作。

触发regular停放订单或停放取消订单后，订单将变成常规订单。T他处理与常规订单插入或订单取消完全相同。到目前为止，在触发已停放订单后，提交订单的订单引用由 CTP 生成，并且不能由客户端程序控制。

## 取消订单

订单取消方法是请求操作。数据结构是CThostFtdcInputorderActionField。. 取消订单和下订单非常相似，但有两点需要注意：

1. 操作标志 字段

目前，国内交易所只支持取消 and do not support amending an 订单 for the ，不支持修改订单，the 因此 ReqOrderAction 方法仅支持canceling an 取消a订单，而 ActionFlag字段的可用值只能THOST\_FTDC\_AF\_Delete。

1. 序列号

订单取消需要通过序列号找到原始订单。最后一节中介绍的序列号的三组都可用于取消订单。

**取消订单 的响应和 退货**

OnRsorder操作：返回订单取消the的响应，whi which 包含交易内核返回的错误消息。

OnRtnOrder ： 交易内核验证订单取消指令后，该指令将提交到交易所，并返回相关订单的新订单状态。

OnErRtnOrder 操作：交易所将再次验证订单取消指令。如果交换发现它是合法的legitimate，将调用此方法，并将返回一条错误消息。如果订单有效，则相关订单的新订单状态也将返回（OnRtnOrder）。

## 询价和 报价

**询价s**

ReqForQuoteInsert 方法用于为投资者提交n 询价。主要数据结构是the CThostFtdcInputForQuoteField。只需要传递合同代码和报价参考号。

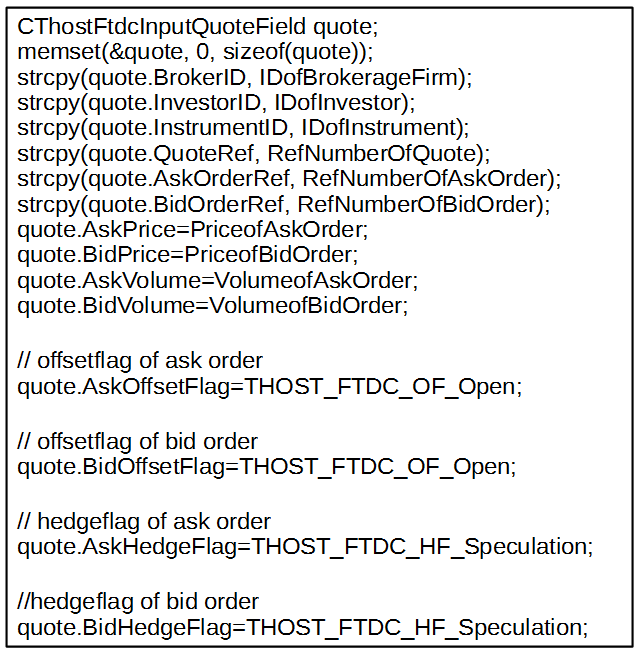
如果询价未通过验证，将调用 OnRsforQuoteInsert 返回错误消息。如果询价有效，则不会返回任何消息。

**做市商获得询价**

在交易 API 中，具有做市商特权的用户将在登录后自动收到投资者的询价。不需要or 请求或订阅the询价。 OnRtnForQuoteRsp 用于接收询价。.

**做市商插入报价**

要插入引号，使用 ReqQuoteInsert 方法，主要数据结构是 CThostFtdcInputQuoteField。



当报价失败验证并且交易内核返回错误消息quote 时，an error message is returned by the 将调用OnRsquoteInsert 方法。

当报价通过验证，交易内核将报价提交到交易所时，将调用 OnRtnQuote。 T wo 常规订单将从报价单派生，quote并连同报价一起提交到交易所。在这种情况下，还将调用 onRtnOrder。

从版本 6.3.0 中，将两个字段 AskOrderRef and 和BidOrderRef 添加到API中。这两个值将分别分配给两个派生订单的 OrderRef 字段。如果客户端程序不指定se值，交易内核将自动分配它们。

从版本 6.3.6，the 字段 ForQuoteSysID is added 添加到 API。连接到 from the quote response DCE 和and CZCE 的做市商应使用报价响应（OnRtnForQuoteRsp）中的报价ID (（ForquoteSysID）填写出价区域。交易所s will 将根据这一领域计算做市商的出价完成率。

**注意：**对于做市商连接到market makers connecting CFFEX，此字段是不需要的to。CFFEX 确定做市商 is that maker bids within a specific period of time after the 的整理率的规则是，如果做市商在交易所向做市商发送请求后的特定时间段（由交易所指定，例如 1分钟exchange sends the ask request to the ）出价。 finishing rate of market maker

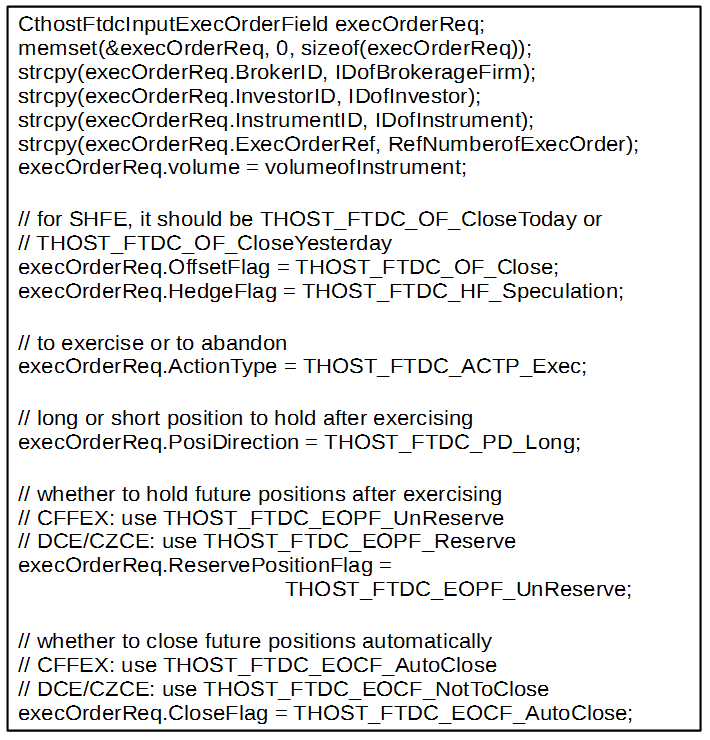
**取消报价**

ReqQuoteAction 方法用于取消报价。其用法与the "要求操作"类似。报价参考+会话 ID=前端 ID 用于取消报价。

取消报价将取消报价单和剩余的remaining 未填写派生订单。做市商也可以直接取消派生order 订单。该方法与取消上一节中的订单相同。

## 行使选项

ReqExecOrderInsert 用于声明投资者打算行使其选择权。主要数据结构是CThostFtdcInputExecOrderField。.



如果请求未通过验证，将调用OnRsExExecOrderInsert 方法。如果请求通过验证，将调用 OnRtnExecorder。

执行 选项后 ，如果客户端收到未执行的消息，则表示交换已接受请求。由于 期权执行是在结算期间处理 的， 所以未执行的消息将在交易 时间由交易所 返回。

**四个 交易所行使 期权的规则**

CffeX

货币期权将自动行使，资金外期权将自动放弃。Forced 练习不允许用于资金外选项。行使期权后，头寸将立即关闭（因为期权行使日期与相应的as 基础未来交割日期相同）。

什费

货币期权将自动行使，资金外期权将自动放弃。投资者可以选择执行或不执行资金 外期权。执行后，保留期货头寸或不守仓是可选的。

Dce

如果期货公司在 DCE 官方网站上设置自动行使标志，期权将自动行使。否则，投资者需要申请行使或放弃自己。

CZCE

货币期权将自动行使，资金外期权将自动放弃。投资者可申请自己行使或放弃。

## Ma） 组合合同

组合 合同保证金规则：

CFFEX rule 将实现s的组合 rule合约的保证金规则是 is 不同于大reduc边规则的一种保证金减值规则。交易所规定了期货与期货、期货与期权或期权与期货之间的匹配比率futures&futures, futures&option, （例如1：3）。 . Investor拥有 who 组合的投资者可向CFFEXreduc申请保证金减息。申请通过后，两条单s腿将绑定到组合合同，这可以从benefit from the 杂志减少规则中受益。关闭组合合约时，应先在结束之前分解组合。

is added to the 具体到组合合同的保证金, ，ReqCombActionInsert 被添加到 API 以支持规则，核心结构是 CThostFtDInputCombActionField。

CTP大大简化了CFFEX组合rules合同规则保证金的适用和中断流程，开发人员只需稍微更改代码the code ，实现rules CFFEX中组合合同的保证金规则。

一个 pply / 休息：

Cthostftdinputcombactionfield

{

经纪人

TThostFtdcBrokerIDType 经纪商 ID;

投资者 ID

TThostFtdc 投资者 类型投资者 IDIDD;

仪器 ID

开发人员需要获得像 a&b 这样的组合合同，然后通过 查询合同 （ReqQryInstrument）来填写此字段。

TThostFtdc 仪器 ID;

组合引用

TThostFtdcOrderReftype CombActionRef;

用户 ID

TThostFtdcUserIDDid ;

方向

开发人员需要填写此字段，在first leg in a&b 中的第一段（contract(如组合合同（即合同 a）。

TThostFtdc 方向 方向;

体积

开发人员 需要填写此字段，该字段应用 了 卷。

例如，对于比率a&b 1：3，a 的位置为is 2 & b 为 6，填充的体积为 2。

TThostFtdc卷 卷;

合并合同方向

填写THOST\_FTDC\_CMDR\_Comb THOST\_FTDC\_CMDR\_Comb （申请）

填写THOST\_FTDC\_CMDR\_UnComb THOST\_FTDC\_CMDR\_UnComb （用于破碎）

TThostFtdcComb 方向 类型组合方向;

对冲旗

TThostFtdcHedgeFlagType 对冲旗;

};

## 数据流文件

启动时，CTP API 可能会在本地计算机上生成the 一些流文件，以保存of received packets in当前交易日的公共、对话框和专用流中接收的数据包数。

数据流文件主要用于恢复模式下的重新传输数据。使用 CTP 风险 API 时，数据流 文件也 用于批处理查询。

D 示例需要注意以下问题：

1. 客户端程序进行大量的读取，s并写入数据 to 流文件。如果客户端程序不能很好地管理系统中的文件句柄数， well文件句柄可能会被使用。
2. 为多个帐户开发程序时，请注意不要将to 所有帐户的流which 文件放在一个目录中，这可能会导致只有一个帐户可以接收退货，而其他帐户不能接收。

数据流Files 由 Quotation API 生成

|  |  |
| --- | --- |
| **报价 API** | |
| 对话Rsp. con | 接收对话框响应数据流 |
| 查询Rsp. con | 接收查询响应数据流 |
| 交易日. con | 交易日 |

数据流Files G由 the Trade API 激活

|  |  |
| --- | --- |
| **贸易API** | |
| 对话Rsp. con | 接收对话框响应数据流 |
| 查询Rsp. con | 接收查询响应数据流 |
| 交易日. con | 交易日 |
| Public. con | 收到的公共返回数据流 |
| 私人. con | 接收的专用返回数据流 |

## F低控制

### 查询流控制

交易 API 中查询操作的限制are 如下：

* 用户 最多只能发送 one 查询操作/秒。
* 最多允许ne同时进行query 进程内查询。如果发出查询操作，但尚未收到响应，则查询操作正在处理中。

以上限制仅适用于交易 API 中的 查询 操作。 下订单、 取消订单、提交报价 和 询价没有这些限制。

### 订单流量控制

为了订单流控制，需要在期货公司need to be系统中配置一些相关参数s。

如果未配置这些参数，则默认限制如下：

* 在一个会话中，每个客户最多只能发送 6 个交易指令（例如下单、取消订单等）。
* 一个帐户 最多只能 同时建立6个会话。

**注意： 如果 操作超出限制，则不会返回任何错误。订单将按队列状态等待处理。**

## 不连接

如果发生断开连接，将调用 OnFrontDisconnect 的方法occurs。断开原因将在 nReason 参数中说明。

**可能的原因：**

|  |  |  |
| --- | --- | --- |
| **十进制系统（10D）** | **十六进制（16H）** | **解释** |
| 4097 | 0 x 1001 | 无法从网络读取network。 |
| 4098 | 0 x 1002 | 无法写入to网络 network。 |
| 8193 | 0x2001 | 时间d 出接受心跳。 |
| 8194 | 0x 2002 | 时间d 出发送心跳。 |
| 8195 | 0x 2003 | 收到错误消息。 |

客户端 和 服务器之间的断开连接有两个原因：

* 网络问题导致 d 断开连接。
* 服务器 主动 断开连接。

服务器主动断开连接有两个原因。

* 客户端很久没有从服务器收到数据包了。超时 发生。
* 连接量超过exceeded 限制。

**心跳机制**

CTP使用检测信号机制来判断客户端和服务器之间的连接是否有效。如果服务器没有要发送到客户端的消息，服务器将及时向客户端发送检测信号数据包。目前，检测信号机制仅在 APIand 中实现，clients 客户端不知道它，因此 the 不会调用 OnHeartBeat 警告方法。

**超时**

网络延迟 可能导致服务器 主动 断开 与客户端的连接。

默认超时为读取 120 秒，写入 60 秒，检测信号 80 秒s。

当客户端和服务器断开连接时，交易 API 将尝试每 5 秒自动重新连接一次。