



根据中国证监会《关于进一步加强期货经营机构客户交易终端信息采集有关事项的公告》（证监会公告[2018]27号）和中国期货市场监控中心《期货公司客户交易终端信息采集及接入认证技术规范》要求，为进一步加强期货市场看穿式监管要求，客户自开发交易软件以及第三方开发的交易软件均需通过期货公司的接入认证，期货公司应审查客户软件功能，测试评估其是否已集成符合监管要求的信息采集动态链接库，是否能够准确采集客户终端信息。对于符合监管要求的终端软件，期货公司会在交易系统中配置该软件的 AppID 和授权码，开通准入权限。

根据监管要求，各操作系统需要采集的内容如下：

操作系统	Windows	Linux	Mac OS	移动终端 ios	移动终端 Android
采集指标	信息采集时间 私网 IP 网卡 MAC 地址 设备名 操作系统版本 硬盘序列号 CPU 序列号 BIOS 序列号 系统盘分区信息	信息采集时间 私网 IP 网卡 MAC 地址 设备名 操作系统版本 硬盘序列号 CPU 序列号 BIOS 序列号	信息采集时间 私网 IP 网卡 MAC 地址 设备名 操作系统版本 硬盘序列号 设备序列号	信息采集时间 移动终端 IP 地理位置信息 操作系统版本 设备名 设备类型 网络运营商 通用唯一识别码 (UUID)	信息采集时间 移动终端 IP 地理位置信息 操作系统版本 设备名 设备类型 国际移动设备识别码 (IMEI) 移动设备识别码 (MEID) 设备 MAC 地址 手机号码 设备序列号 国际移动用户识别码 (IMSI) IC 卡的唯一识别号码 (ICCID)

特别注意：1、CTP 交易系统的授权码与客户交易软件的 AppID 或 RelayAppID 绑定，客户交易软件的 AppID 如果发生改变，需要向期货公司重新申请授权码，在不改变 AppID 的情况下，客户可以任意更改交易程序、软件功能等内容。2、用户采集信息由期货公司每日结算后生成加密文件直接报送给中国期货市场监控中心，中国期货市场监控中心共享给期货交易所，期货公司无法获取用户的采集信息。

CTP 主席系统外部接入操作指引


一、 确定 AppID 名称和版本号，获取授权码

客户填写《中信建投期货有限公司外部接入系统客户情况登记表》，确定 AppID 名称和版本号，留存获取授权码的联系方式。个人开发的程序统一命名格式为“client_终端名称_版本号”，其中下划线不得更改，终端名称可以是纯英文字母或纯数字或英文字母和数字的组合，英文字母可区分大小写，不能是中文汉字，字段最大为 10 字节；版本号最大长度为 8 字节。

我司信息技术部将根据客户的 AppID 生成 CTP 评测系统授权码，我部将及时发送给客户，并告知生效日期，**该授权码用于接入 CTP 评测系统、CTP 主席正式实盘系统、CTP 仿真系统。**

二、 使用看穿式监管评测版本 API 接入评测系统进行评测

1、登录上期技术官网 http://www.sfit.com.cn/5_2_DocumentDown_2.htm，下载看穿式监管评测版本 api (v6.7.0_CP)，该版本 api 仅用于接入 CTP 评测系统；


 **看穿式监管生产版本 (版本号: v6.7.0_20230209 9:52:16.3535)** 更新时间: 2023-5-25

简介: 看穿式监管Linux/Windows版本。本次更新内容如下: 此版本支持查询流量压缩lz4算法。适用于6.5.0及以上柜台。仅traderapi, 采集模块(WinDataCollect.dll/ LinuxDataCollect.so)请从“看穿式采集库”中生产版本看穿式采集库中下载。

api下载 md5码: d33a49e120d18cdaa15c3837e8d74ac9 [下载遇到问题](#)

chm下载 (2023.5.25更新) md5码: 6e7875c7bf0658d3e28f1660bf26bad0

demo下载 md5码: 999608d16ee9f473e233d7e0ab67c6e1 [文档下载](#)

 **看穿式监管评测版本 (版本号: v6.7.0_CP_20230303 15:29:48.3535)** 更新时间: 2023-5-25

简介: 看穿式监管Linux/Windows版本。本次更新内容如下: 此版本支持查询流量压缩lz4算法。适用于6.5.0及以上柜台。仅traderapi, 采集模块(WinDataCollect.dll/ LinuxDataCollect.so)请从“看穿式采集库”中评测版本看穿式采集库中下载。

api下载 md5码: 77a9ff067161f273044a88132a05ce6f [下载遇到问题](#)

2、评测系统接入参数；

评测系统 Broker ID：6666

评测系统交易地址：61.186.254.131:42205

3、评测系统帐号，**仅登录即可**；

资金帐号：12345678

登录密码：CS123456 （CS 为大写）

4、客户完成评测系统登录后，请及时告知我部，为您查询评测结果；

5、评测通过认证的范例，通过认证案例如下图：

基本信息	
交易软件商ID:so	公钥版本号:1
终端类型:Windows	信息采集时间:2019-05-30 15:43:57
私网IP1:192	私网IP2:
网卡MAC地址1:60	网卡MAC地址2:
设备名:T/	操作系统版本:6.1
硬盘序列号:11	CPU序列号:B
BIOS序列号:CG	系统盘分区信息:C, 5
补充信息	
异常标识:正常	AppID:P
客户交易终端公网IP:1	客户交易终端公网端口号:5
客户登入时间:2019-05-30 15:44:41	RelayAppID:
中继代理公网IP:	中继代理公网端口号:
中继代理登入时间:	客户内部资金账户:12345678
用户类型:客户	

三、通过评测，客户 AppID 和授权码录入 CTP 主席系统

客户通过评测，待我司录入系统后，可以使用授权码进行接入（获取授权码的下一交易日）。

CTP 主席系统看穿式监管版**互联网接入地址**见下表：

API 接口版本号：v6.7.0（下载地址见下文第四.1 条的截图）
Broker ID：9080 交易端口：42205 行情端口：42213


电信1 (重庆)	61.186.254.135
电信2 (上期张江机房)	180.166.25.24
电信3 (上海来安路农商行机房)	114.80.55.93
电信4 (上海来安路农商行机房)	114.80.55.94
电信5 (上期张江机房)	101.231.128.133
联通1 (上海张江联通)	140.207.150.241
联通2 (重庆)	113.204.105.118
联通3 (北京)	202.130.235.165
联通4 (北京)	202.130.235.168
盘后查询站点 D 交易日 17:40-19:00, 03:00-07:00, 可查询D日全部交易数据, 周末全天24小时, 可查询上一交易日全部交易数据	211.95.60.25
注: 以上站点均支持上期所、能源中心二代五档行情。	

CTP 主席系统看穿式监管版上期张江机房内网接入地址见下表:

API 接口版本号: v6.7.0 (下载地址见下文第四.1 条的截图)	
Broker ID: 9080	
内网地址 1	10.124.34.51 交易: 42205 行情: 42213
内网地址 2	10.124.34.52 交易: 42205 行情: 42213
内网地址 3	10.124.34.53 交易: 42205 行情: 42213
上期二代行情内网 服务器地址	10.124.34.27 行情端口:19213

四、 测试环境：CTP 仿真系统测试（选做）

1、为了保障客户接入无误，建议客户在获取授权码后接入 CTP 仿真系统进行测试验证。登录上期技术官网 http://www.sfit.com.cn/5_2_DocumentDown_2.htm，下载生产版本 api (v6.7.0)，该版本与我司正式实盘系统版本一致；


 **看穿式监管生产版本 (版本号: v6.7.0_20230209 9:52:16.3535)** 更新时间: 2023-5-25

简介: 看穿式监管Linux/Windows版本。本次更新内容如下: 此版本支持查询流量压缩lz4算法。适用于6.5.0及以上柜台。仅traderapi, 采集模块(WinDataCollect.dll/ LinuxDataCollect.so)请从“看穿式采集库”中生产版本看穿式采集库中下载。

api下载 md5码: d33a49e120d18cdaa15c3837e8d74ac9 [下载遇到问题](#)

chm下载 (2023.5.25更新) md5码: 6e7875c7bf0658d3e28f1660bf26bad0

demo下载 md5码: 999608d16ee9f473e233d7e0ab67c6e1 [文档下载](#)

 **看穿式监管评测版本 (版本号: v6.7.0_CP_20230303 15:29:48.3535)** 更新时间: 2023-5-25

简介: 看穿式监管Linux/Windows版本。本次更新内容如下: 此版本支持查询流量压缩lz4算法。适用于6.5.0及以上柜台。仅traderapi, 采集模块(WinDataCollect.dll/ LinuxDataCollect.so)请从“看穿式采集库”中评测版本看穿式采集库中下载。

api下载 md5码: 77a9ff067161f273044a88132a05ce6f [下载遇到问题](#)

2、仿真系统接入参数；

仿真系统 Broker ID：6666

仿真系统交易地址：61.186.254.137:33433

仿真系统行情地址：61.186.254.137:33435

3、仿真系统帐号，任选其一；

资金帐号：50100267，登录密码：CS123456 （CS 为大写）

资金帐号：50100268，登录密码：CS123456

资金帐号：50100282，登录密码：CS123456

五、 签署协议

客户签署《中信建投期货有限公司外部交易系统接入协议》、《中信建投期货有限公司外部接入评测结果确认书》、《中信建投期货有限公司程序化交易客户承诺书》、《中信建投期货有限公司程序化交易系统自测表》。回寄客户经理地址：上海市浦东新区世纪大道 1589 号 810 室。

CTP 次席系统（第三交易中心）外部接入操作指引

接入流程和 API 文件同“CTP 主席系统外部接入操作指引”。

CTP 次席系统（第三交易中心）看穿式监管版互联网接入地址见下表：

API 接口版本号：v6.7.0	
Broker ID：9080 交易端口：42205 行情端口：42213	
电信站点	180.166.25.17
联通站点	27.115.57.105
注：CTP 次席系统的行情站点不支持上期二代行情，如有需要，可选择连接 CTP 主席中的行情站点。	

CTP 次席系统（第三交易中心）看穿式监管版上期张江机房内网接入地址见下表：

API 接口版本号：v6.7.0	
Broker ID：9080	
内网地址	10.124.34.1 交易：42205 行情：42213
上期二代行情内网 服务器地址	10.124.34.27 行情端口:19213

CTP mini2 系统（第七交易中心）外部接入操作指引

接入流程同“CTP 主席系统外部接入操作指引”。

看穿式监管版本 API 文件下载：

登录上期技术官网 http://www.sfit.com.cn/5_2_DocumentDown_4.htm, 下载生产版本 api(v1.6.3)



The screenshot shows a web page with a navigation menu on the left and a main content area. The main content area has a red box highlighting the '看穿式监管生产版本 (版本号: CTPIIMini API V1.6.3)' section. Below this, there is a '简介' (Introduction) section, followed by 'api下载' and 'chm下载' links with their respective md5 hashes. A second section for '看穿式监管评测版本/非看穿式监管生产版本(版本号:v1.2)' is also visible below.

CTP mini2 系统（第七交易中心）看穿式监管版**接入地址**见下表：

API 接口版本号：v1.6.3	
Broker ID：9080	
互联网地址	接入地址：101.231.128.137
(上期一代行情)	交易端口 31803，查询端口 31804，行情端口 31807
上期张江机房	接入地址：10.124.34.92
内网地址	交易端口 31803，查询端口 31804，行情端口：31807
上期二代行情内网 服务器地址	10.124.34.27 行情端口:19213

参考附件 1：终端认证调试 CTP-Linux

1、准备工作

版本说明：

- 1、6.3.13 用于采集客户终端信息，完成客户端的接入认证。
- 2、6.3.15 用于仿真环境和生产环境。

由于 Linux 的发行版众多,不同发行版的 GCC/G++ 编译器的版本和集成度也有一定的区别，但基本编译原理是一致的。本文将以 64 位的 Ubuntu-18.10 做为开发者宿主机环境。

本文中使用的开发 IDE 为 SourceInsight3.5,需要在宿主机上安装 GCC/G++ 编译器。如果程序中使用第三方库请自行完善环境及 Makefile 编译文件。

安装 GCC : `sudo apt-get install gcc`

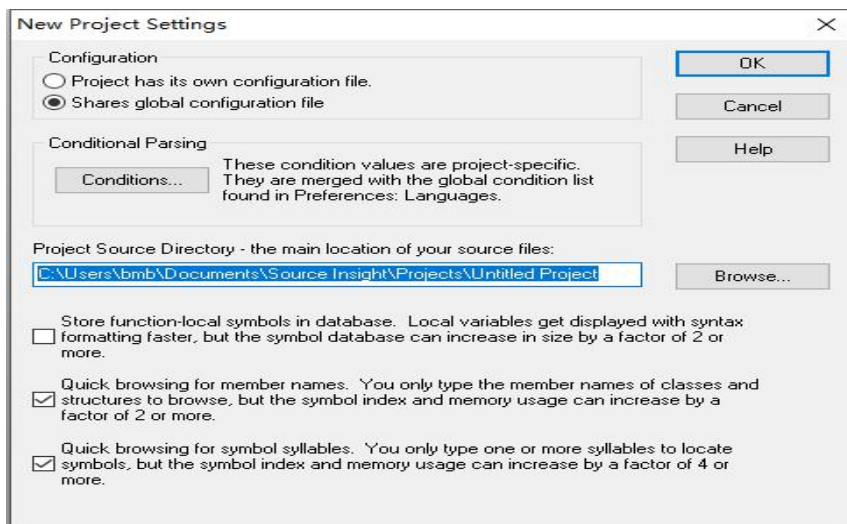
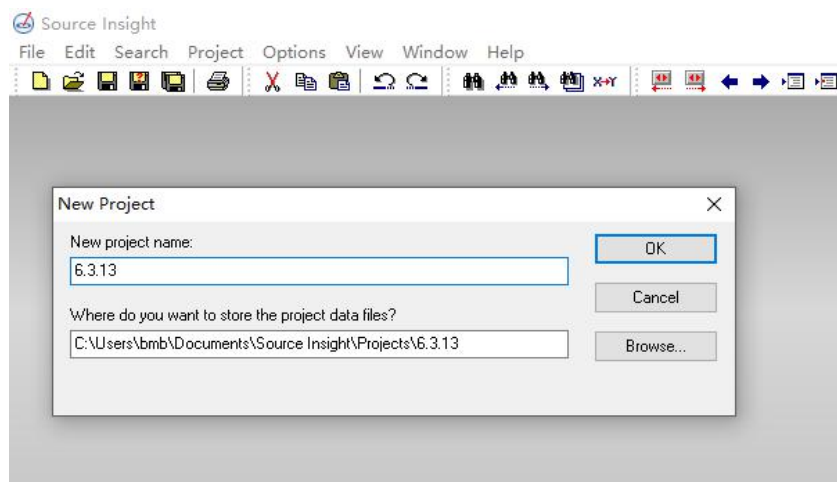
安装 G++ : `sudo apt-get install g++`

2、建立项目解决方案

说明:CTP 6.3.13 版本和 6.3.15 版本在客户端接入及认证部分是一致的，本文以 6.3.15 版本为例进行说明。实际上由于客户接入流程的原因，客户自开发程序需要同时拥有两个版本的解决方案。用户需要先通过 6.3.13 进行客户端数据采集及验证，待期货公司完成数据验证后再通过 6.3.15 版本进行正常交易。

2.1、建立项目工程

打开 SourceInsight，创建 C/C++ 工程解决方案。



2.2、集成 CTP API 到工程中

需要把 CTP API 包拷贝到本地文件系统的工程目录中，拷贝的 API DLL 和头文件位于

v6.3.15_20190220_api_clientdatacollectdll_linux64、v6.3.15_20190220_api_tradeapi_se_linux64 目录中。

v6.3.15_20190220_api_clientdatacollectdll_linux64 用于客户端信息的采集。

v6.3.15_20190220_api_tradeapi_se_linux64 用于行情、交易等功能。

注：带有 64 字符的包表示是用于 64 位 Linux 操作系统的，上期技术没有提供 32 位版本的 linux 库文件。

1、在工程目录中建立 ctplib64 目录，把 CTPAPI 头文件、DLL 拷贝到 ctplib64 目录中。ctplib64 目录名称可以自定义，用于在工程中附加库路径。

citics-flow1	2019/4/26 17:20	文件夹	
ctplib64	2019/4/26 13:54	文件夹	
config.ini	2019/4/29 10:31	配置设置	1 KB
CTraderApi.cpp	2019/4/29 10:32	CPP 文件	76 KB
CTraderApi.h	2019/4/26 15:53	H 文件	15 KB
CTraderApi.o	2019/4/30 8:48	O 文件	100 KB
CTraderSpi.cpp	2019/4/29 9:48	CPP 文件	230 KB
CTraderSpi.h	2019/4/26 16:42	H 文件	21 KB
CTraderSpi.o	2019/4/30 8:48	O 文件	310 KB
Custom.h	2019/4/30 9:12	H 文件	80 KB
getconfig.cpp	2019/4/29 9:16	CPP 文件	2 KB
getconfig.h	2019/4/26 14:51	H 文件	1 KB
getconfig.o	2019/4/30 8:48	O 文件	11 KB
global.h	2019/4/26 16:16	H 文件	1 KB
main	2019/4/30 9:12	文件	283 KB
main.cpp	2019/4/30 9:12	CPP 文件	3 KB
main.o	2019/4/30 9:12	O 文件	177 KB
Makefile	2019/4/26 16:45	文件	1 KB

2、在工程目录中建立两个目录，分别用于交易流和行情流数据的保存。

3、在解决方案中添加 API 包的头文件(.h)到工程中

DataCollect.h	2018/11/19 13:47	H 文件	1 KB
error.dtd	2018/11/19 13:47	XML 文档类型定义	1 KB
error.xml	2018/11/19 13:47	XML 文档	17 KB
LinuxDataCollect.so	2018/11/19 13:47	SO 文件	2,605 KB
ThostFtdcMdApi.h	2018/11/19 13:47	H 文件	6 KB
ThostFtdcTraderApi.h	2018/11/19 13:47	H 文件	35 KB
ThostFtdcUserApiDataType.h	2018/11/19 13:47	H 文件	244 KB
ThostFtdcUserApiStruct.h	2018/11/19 13:47	H 文件	223 KB
thostmduserapi_se.so	2018/11/19 13:47	SO 文件	4,506 KB
thosttraderapi_se.so	2018/11/19 13:47	SO 文件	5,122 KB

4、引入 CTP 库文件与编写编译指令文件(Makefile)

```

SRC = main.cpp
OBJ5 = main.o
CC = g++ -std=c++11

main: getconfig.o CTraderApi.o CTraderSpi.o $(OBJ5)
$(CC) -o main getconfig.o CTraderApi.o CTraderSpi.o $(OBJ5) ctplib64/thostmduserapi_se.so ctplib64/thosttraderapi_se.so
getconfig.o: getconfig.cpp getconfig.h
$(CC) -c getconfig.cpp -o getconfig.o -lcurses
CTraderApi.o: CTraderApi.cpp CTraderApi.h
$(CC) -c CTraderApi.cpp -o CTraderApi.o ctplib64/thostmduserapi_se.so ctplib64/thosttraderapi_se.so
CTraderSpi.o: CTraderSpi.cpp CTraderSpi.h
$(CC) -c CTraderSpi.cpp -o CTraderSpi.o ctplib64/thostmduserapi_se.so ctplib64/thosttraderapi_se.so
main.o: $(SRC)
$(CC) -c $(SRC) -o main.o ctplib64/thostmduserapi_se.so ctplib64/thosttraderapi_se.so ctplib64/L:

clean:
-rm -f *.o main
    
```

说明：上图中的内容仅作为一个 Makefile 参考模板，编译文件中的文件数量以及链接库需要根据实际情况进行编写。

上图中的四个主要文件作用：

Main.c :程序主线程完成交易类和行情类的初始化工作。

Getconfig.c : 程序的配置选项。保存交易前置、行情前置地址、投资者账号、密码等信息。

CTraderApi.c : 继承 CTP 交易类 API 的自定义实现。

CTraderSpi.c : 继承 CTP 交易类 SPI 的自定义实现。

Makefile 文件的\$(CC)标记处都链接了.so 库和 lpthread 库。这里仅作为文件编写参考，实际上不需要每个文件都重复链接相同的库。

5、至此 API 包已经和工程集成了。

2.3、接入相关的主要代码

1、创建交易类

```
CTraderApi* pUserApi = new CTraderApi;
```

```
/*
```

```
CTraderApi 继承 CThostFtdcTraderApi。
```

```
CThostFtdcTraderApi类提供了交易api的初始化、登录、报单和查询等功能。
```

```
由于CTP定义的交易和行情类都是抽象的虚函数，所以在CTraderApi.cpp需要全部实现但可以不 用重写。
```

```
详情请参考CTP技术手册
```

```
*/
```

2、创建交易流目录

```
pUserApi->CreateFtdcTraderApi(".*\\citics-flow1\\");
```

```
/*
```

```
pUserApi->CreateFtdcTraderApi(".*\\citics-flow1\\");
```

```
程序可以通过CThostFtdcTraderApi发出操作请求，通过继承CThostFtdcTraderSpi并重载回调函数来处理后台服务的响应。
```

```
如果创建多个api实例，则每个实例的flow目录都要区分开，否则可能会导致报单回报丢失。
```

```
程序运行之前，流目录必须提前创建好，否则会报错"RuntimeError:can not open CFlow file in line 279 of
```

```
file ..\\.\\source\\userapi\\ThostFtdcUserApiImplBase.cpp"。
```

```
这里仅仅为了接入演示，所以手动在文件夹中建立好"citics-flow1"目录
```

```
*/
```

3、初始化交易线程

```
CustomTraderHandler sh(pUserApi);
```

```
pUserApi->RegisterSpi(&sh);
```

```
pUserApi->SubscribePrivateTopic(THOST_TERT_QUICK);
```

```
pUserApi->SubscribePublicTopic(THOST_TERT_QUICK);
```

```
pUserApi->RegisterFront("交易前置地址");
```

```
pUserApi->Init();
/*
CustomTraderHandler 继承 CTraderSpi, 用于客户的自定义业务逻辑
*/
```

4、信息采集及认证

```
m_pUserApi->ReqAuthenticate(&pReqAuthenticateField, 1);
/*
ReqAuthenticate 函数原型为
virtual int ReqAuthenticate(CHostFtdcReqAuthenticateField *pReqAuthenticateField, int nRequestID) = 0;
参数及返回值请参考 CTP 手册
*/
```

5、账号登录

```
pUserApi->ReqUserLogin(&reqUserLogin, nRequestID++);
```

3、Linux 接入程序中的注意事项

3.1、字符串长度(重要)

字符串是连续的字符序列，最后以空字符'\0'作为终止符。一个字符串的长度指所有字符的数量，但不包括终止符。在 C 语言中，没有字符串类型，自然也就没有运算符以字符串为操作数。

字符串被存储在元素类型为 char 或宽字符类型数组中（宽字符类型指 wchar_t、char16_t 或 char32_t）。存储字符串的数组一定比字符串长度多一个元素，以容纳下字符串终止符（空字符'\0'）。

Linux CTP 接入中要注意字符串长度，长度不正确 CTP 后台会返回错误信息。

例如：假设投资者账号为“123456”。在 c 语言中用 char 数组保存这个字符串实际上需要占用 7 字节加上末尾的'\0'结束符。如果直接使用这个 char 数组用于登录接口调用，CTP 会返回登录失败。

Linux 中 CTP 能正确识别的字符串需要去掉字符数组最后的结束符。如果投资者账号为“123456”，在调用 CTP 接口时只能把 6 字节的“123456”作为参数传入。

3.2、线程同步

Linux 中设计线程同步方案可使用锁、信号量等机制。下面给出线程锁的同步方案作为参考，关于锁的技术原理请参考相关资料。

1、初始化静态锁

```
/*初始化静态方式锁*/
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
pthread_cond_t flag = PTHREAD_COND_INITIALIZER;
```

2、线程加锁

```
/*初始化自定义交易类--start*/
pthread_mutex_lock(&mutex); //加锁
```

```
CustomTraderHandler sh(pUserApi);
pUserApi->RegisterSpi(&sh);
pUserApi->SubscribePrivateTopic(THOST_TERT_QUICK);
pUserApi->SubscribePublicTopic(THOST_TERT_QUICK);
pUserApi->RegisterFront(const_cast<char*>(g_chFrontaddr.c_str()));
pUserApi->Init();
```



```
pthread_cond_wait(&flag,&mutex); //等待 flag 置位  
pthread_mutex_unlock(&mutex); //解锁
```

3、对应的线程回调中置位 flag 标志

*/*前置连接后的回调中置位锁*/*

```
virtual void OnFrontConnected()  
{  
    /*用户自定义实现的业务逻辑*/  
    /*置位锁*/  
    pthread_mutex_lock(&mutex);  
    pthread_mutex_unlock(&mutex);  
    pthread_cond_signal(&flag);  
}
```

参考附件 2：终端认证调试 CTP-windows

1、准备工作

版本说明：

- 1、6.3.13 用于采集客户终端信息，完成客户端的接入认证。
- 2、6.3.15 用于仿真环境和生产环境。

在开发者宿主机上安装 C/C++ IDE 及开发环境,如 VisualStudio、QtCreator.

本说明文档采用 VisualStudio 作为开发环境 IDE 进行后续项目工程的开发。

2、建立项目解决方案

说明:CTP 6.3.13 版本和 6.3.15 版本在客户端接入及认证部分是一致的，本文以 6.3.15 版本为例进行说明。实际上由于客户接入流程的原因，客户自开发程序需要同时拥有两个版本的解决方案。用户需要先通过 6.3.13 进行客户端数据采集及验证，待期货公司完成数据验证后再通过 6.3.15 版本进行正常交易。

2.1、建立项目工程

打开 VisualStudio,创建 C++/win32 工程解决方案。本文使用的 VisualStudio 是 2019 版本，其他版本在功能上是一致的界面上会有一些的区别。(建议使用 VisualStudio2012 以上的版本进行开发)



2.2、集成 CTP API 到工程中

需要把 CTP API 包拷贝到本地文件系统的工程目录中，拷贝的 API DLL 和头文件位于 6.3.15_20190220_clientdll64_windows、6.3.15_20190220_tradeapi64_se_windows 目录中。

6.3.15_20190220_clientdll64_windows 用于客户端信息的采集。

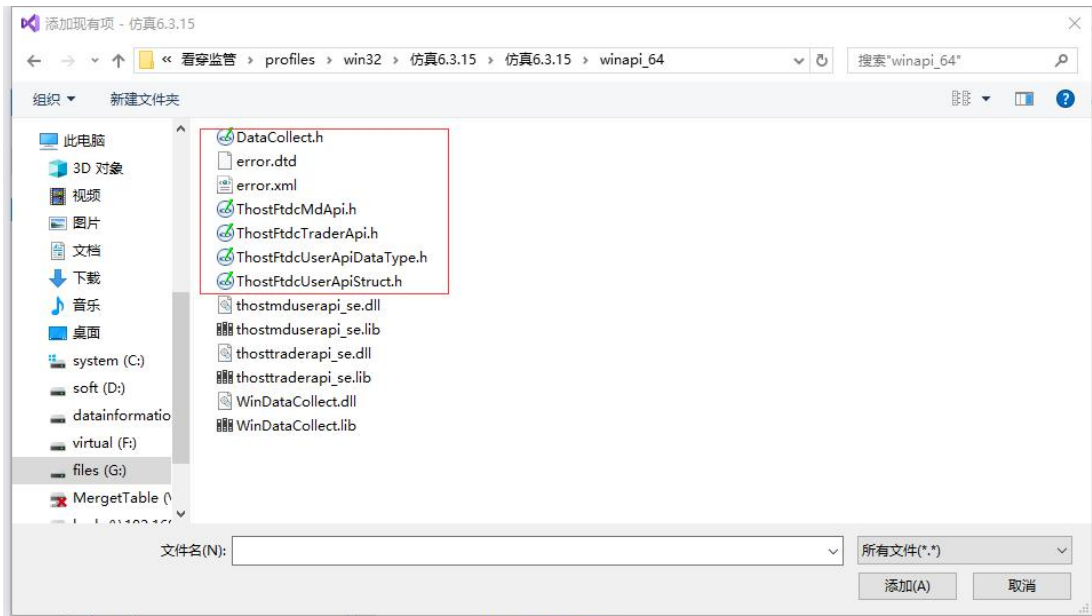
6.3.15_20190220_tradeapi64_se_windows 用于行情、交易等功能。

注：带有 64 字符的包表示是用于 windows64 位操作系统的，没有 64 字符的包表示用于 windows32 位操作系统。

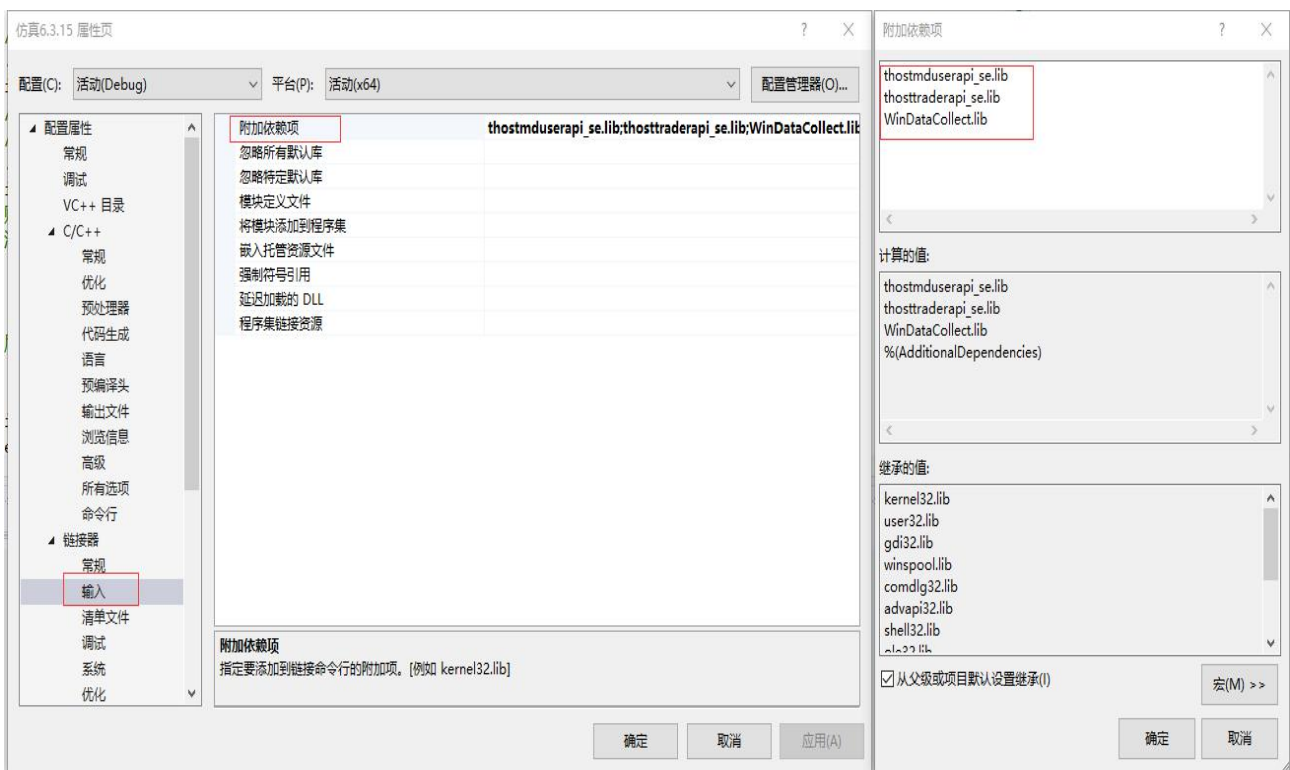
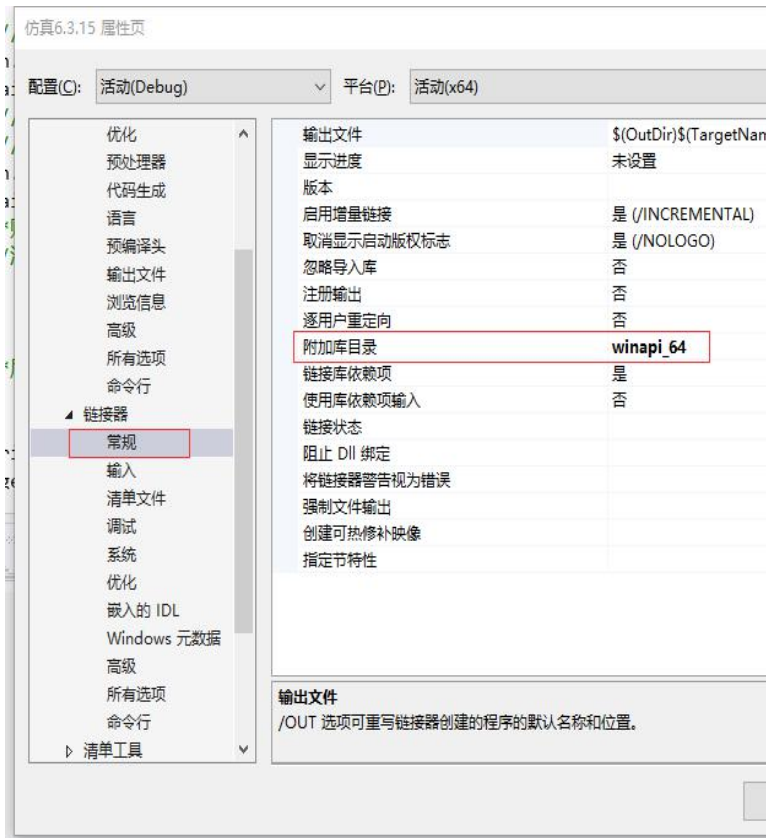
- 1、在工程目录中建立 winapi_64 目录，把 CTPAPI 头文件、DLL 拷贝到 winapi_64 目录中。Winapi_64 目录名称可以自定义，用于在工程中附加库路径。

名称	修改日期	类型
DataCollect.h	2019/2/15 11:09	H 文件
error.dtd	2019/2/15 11:08	XML 文件
error.xml	2019/2/15 11:08	XML 文件
ThostFtdcMdApi.h	2019/2/20 20:47	H 文件
ThostFtdcTraderApi.h	2019/2/20 20:46	H 文件
ThostFtdcUserApiDataType.h	2019/2/20 20:47	H 文件
ThostFtdcUserApiStruct.h	2019/2/20 20:47	H 文件
thostmduserapi_se.dll	2019/2/20 20:49	应用程序
thostmduserapi_se.lib	2019/2/20 20:49	对象库
thosttraderapi_se.dll	2019/2/20 20:47	应用程序
thosttraderapi_se.lib	2019/2/20 20:47	对象库
WinDataCollect.dll	2019/2/20 20:49	应用程序
WinDataCollect.lib	2019/2/20 20:49	对象库

- 在工程目录中建立两个目录，分别用于交易流和行情流数据的保存。
- 在解决方案中添加 API 包的头文件(.h)到工程中



- 在工程中引用 DLL 文件



5、至此 API 包已经和工程集成了。

2.3、接入相关的主要代码

1、创建交易类

```
CTraderApi* pUserApi = new CTraderApi;
/*
CTraderApi 继承 CHostFtdcTraderApi。
```

CThostFtdcTraderApi类提供了交易api的初始化、登录、报单和查询等功能。

由于CTP定义的交易和行情类都是抽象的虚函数，所以在CTraderApi.cpp需要全部实现但可以不 用重写。
详情请参考CTP技术手册

*/

2、创建交易流目录

```
pUserApi->CreateFtdcTraderApi(".\\citics-flow1\\");
```

/*

```
pUserApi->CreateFtdcTraderApi(".\\citics-flow1\\");
```

程序可以通过CThostFtdcTraderApi发出操作请求，通过继承CThostFtdcTraderSpi并重载回调函数来处理后台服务的响应。

如果创建多个api实例，则每个实例的flow目录都要区分开，否则可能会导致报单回报丢失。

程序运行之前，流目录必须提前创建好，否则会报错“RuntimeError:can not open CFlow file in line 279 of file ..\\.\\source\\userapi\\ThostFtdcUserApiImplBase.cpp”。

这里仅仅为了接入演示，所以手动在文件夹中建立好“citics-flow1”目录

*/

3、初始化交易线程

```
CustomTraderHandler sh(pUserApi);
```

```
pUserApi->RegisterSpi(&sh);
```

```
pUserApi->SubscribePrivateTopic(THOST_TERT_QUICK);
```

```
pUserApi->SubscribePublicTopic(THOST_TERT_QUICK);
```

```
pUserApi->RegisterFront("交易前置地址");
```

```
pUserApi->Init();
```

/*

CustomTraderHandler 继承 CTraderSpi，用于客户的自定义业务逻辑

*/

4、信息采集及认证

```
m_pUserApi->ReqAuthenticate(&pReqAuthenticateField, 1);
```

/*

ReqAuthenticate 函数原型为

```
virtual int ReqAuthenticate(CThostFtdcReqAuthenticateField *pReqAuthenticateField, int nRequestID) = 0;
```

参数及返回值请参考 CTP 手册

*/

5、账号登录

```
pUserApi->ReqUserLogin(&reqUserLogin, nRequestID++);
```