

# 雷达星 6.0.2 交易指令接口使用说明

提供基于 TCP 端口接收\*\*基金策略服务器发出的现货普通交易下单指令【信用交易指令暂不提供】。

为保障数据私密安全，采取如下 4 种措施：

- 1、服务器端支持 IP 白名单控制 只有在白名单中的 IP 机器发送的数据才能被接收，非白名单的会断开网络连接。不设置白名单则不控制。
- 2、通讯数据加密后传送 调用 API 函数可选择使用加密传送，服务器收到加密数据后解密。
- 3、验证客户信息-客户号。[注 1](#)
- 4、当日握手密钥机制，在 NetSend 函数中送入交易密码。[注 2](#)

为保证合规使用，采用 2 条限制：

- 1、仅一个有效接入【白名单仅支持一个 IP】。
- 2、每日 TCP 买入/卖出指令委托笔数不能超过 300 笔。

【[注 1](#),[注 2](#)】强制加密传送。

## API 使用说明

提供 WIN32 及 X64 两种版本接口。

名称	修改日期	类型	大小
WIN32	2024/9/15 10:56	文件夹	
StockLib.dll	2025/4/1 12:54	应用程序扩展	272 KB
stocklib.h	2025/3/26 15:51	C/C++ Header	2 KB
StockLib.lib	2025/4/1 12:54	Object File Library	21 KB

## 交易指令信息

```
struct stTradeInfo
{
//20240227交易指令
int nZHA6;//客户号前 6 位-20241111
int nZHB6;//客户号后6位-2024111
int nSymbol;//atoi(证券代码)
int nTriggle_Volume;//触发交易的卖一量, 小于此量才进行买入--打板单有值
```

```

int nBuySell_Volume;//触发交易后. 买入卖出的量
int nBuySell;//1-买, 2-卖, 3-撤单, 4-盯盘
int nPrice;//普通交易价格*1000
int nFID;//撤单字段1
int nSID;//撤单字段2
int nOrdRef;//撤单字段3
char cEnCode[16];//当日握手密钥-由调用的函数传入账户交易密码参数-加密
int nELen;//20250326-增加密码长度传送
};
//20240206指令传送计数
int g_nTcpTansCnt = 0;
//20240206交易指令传送
STOCKTEK_API int NetSend(void* pSocket/*TCP socket*/, stTradeInfo* pTInfo/*交易指令*/, const char* szKhh/*客户号*/, const char* pJymm/*交易密码*/, int nSMode = 1/*1 加密 0 不加密*/);

```

调用示例

客户号 szKHH;

交易密码 szJymm;

## 盯盘

```

stTradeInfo* pT2 = new stTradeInfo;
memset(pT2, 0, sizeof(stTradeInfo));

pT2->nSymbol = atoi("000881");//代码000881
pT2->nBuySell_Volume = 100;//买入数量100股
pT2->nTriggle_Volume = 15000;//卖一盘低于15000股
pT2->nBuySell = 4;//盯盘

```

## 买入

### (1) 普通买入

```

stTradeInfo* pT2 = new stTradeInfo;
memset(pT2, 0, sizeof(stTradeInfo));

pT2->nSymbol = atoi("000881");//代码000881
pT2->nBuySell_Volume = 100;//买入数量100股
pT2->nBuySell = 1;//买入
pT2->nPrice = (int)1000*6.86;//价格
pT2->nTriggle_Volume = 0;

```

### (2) 打板买入后盯盘

```

stTradeInfo* pT2 = new stTradeInfo;
memset(pT2, 0, sizeof(stTradeInfo));

```

```

pT2->nSymbol = atoi("000881");//代码000881
pT2->nBuySell_Volume = 100;//买入数量100股
pT2->nBuySell = 1;//买入
pT2->nPrice = (int)1000*6.86;//价格-涨停价
pT2->nTriggle_Volume = 15000;

```

### 卖出

```

stTradeInfo* pT2 = new stTradeInfo;
memset(pT2, 0, sizeof(stTradeInfo));

pT2->nSymbol = atoi("000881");//代码000881
pT2->nBuySell_Volume = 100;//卖出数量100股
pT2->nBuySell = 2;//卖出
pT2->nPrice = (int)1000*6.86;//价格

```

### 撤单

```

stTradeInfo* pT2 = new stTradeInfo;
memset(pT2, 0, sizeof(stTradeInfo));

pT2->nSymbol = atoi("000881");//代码000881
pT2->nBuySell = 3;//撤单
pT2->nFID = 5689;
pT2->nSID = 2202;
Pt2->nOrdRef = 89;

```

撤(nFID = 5689; nSID = 2202; nOrdRef = 89)的委托

### 查询全部持仓

```

stTradeInfo* pT2 = new stTradeInfo;
memset(pT2, 0, sizeof(stTradeInfo));

pT2->nSymbol = 999999;//6个9

int nRet = NetSend((void*)g_pSocket->m_hSocket, pT2, szKhh, szJymm);
//执行加密传送
int nRet = NetSend((void*)g_pSocket->m_hSocket, pT2, szKhh, szJymm, 0);
//执行不加密传送

```

其中，g\_pSocket 是你自己已建立的 TCP 连接。

在 PY 或 C++ 中 创建 SOCKECT，建立 SOCKET 连接，调用 NetSend 函数进行加密传送。  
数据信息见 stTradeInfo

## 回报数据接口

```
struct stOrderInfo
{
    //20240227委托回报
    int nSymbol;//atoi(证券代码)
    int nBuySell;//1-买, 2-卖
    int nPrice;//普通交易价格*1000
    int nBuySell_Volume;//触发交易后. 买入卖出的量
    int nTrade_Volume;//成交数量
    int nCancel_Volume;//撤单数量
    int nFID;//撤单用字段1
    int nSID;//撤单用字段2
    int nOrdRef;//撤单用字段3
    int nOrderStatus;//订单状态2-未成交、3-部分成交、4-全部成交、5-部成部撤、6-全部撤单、7-废单
    int nTotalPos;//总的持仓数量
    int nAvlPos;//可卖数量
    double dAvlJE;//可用金额
};
```

## 接收委托回报函数

```
//20240227
STOCKTEK_API int NetRecv(void* pSocket/*TCP socket*/, stOrderInfo* p0Info/*委托数据*/,
int nSMODE = 1/*1加密0不加密*/);
```

当服务器有数据推送过来时调用本函数接收委托

查持仓通过委托回报方式返回，  
除 代码、总数量、可用数量、可用金额有值外，其他为0

## 调用示例

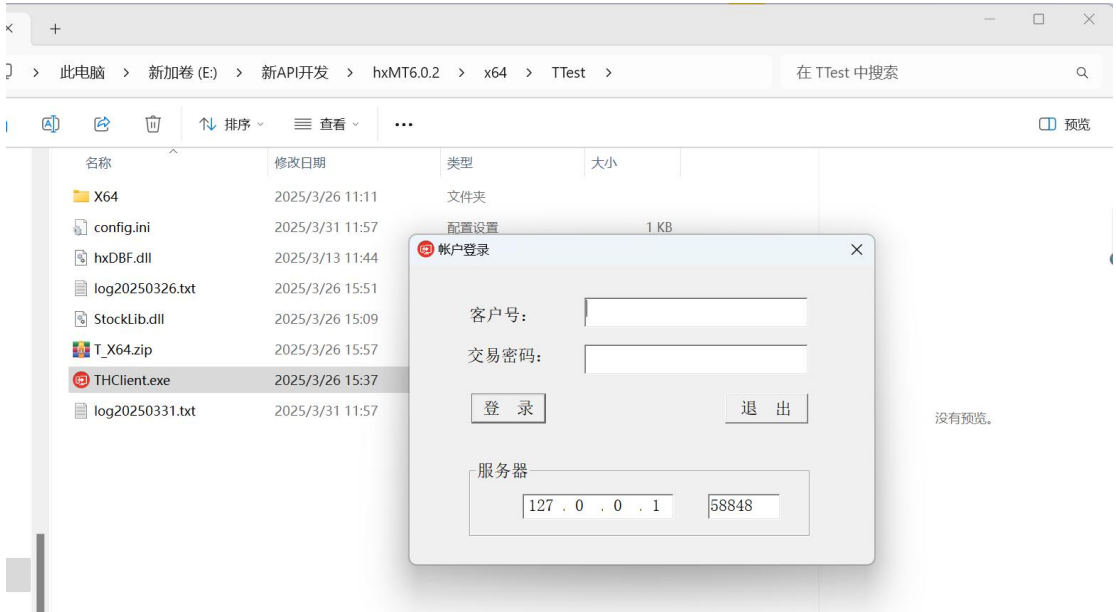
```
stOrderInfo* pOrder = new stOrderInfo();
memset(pOrder, 0, sizeof(stOrderInfo));

int nRet = NetRecv((void*)g_pSocket->m_hSocket, pOrder);
返回值,
-1, 网络错误
0, 无数据
>0, 正常收到数据
```

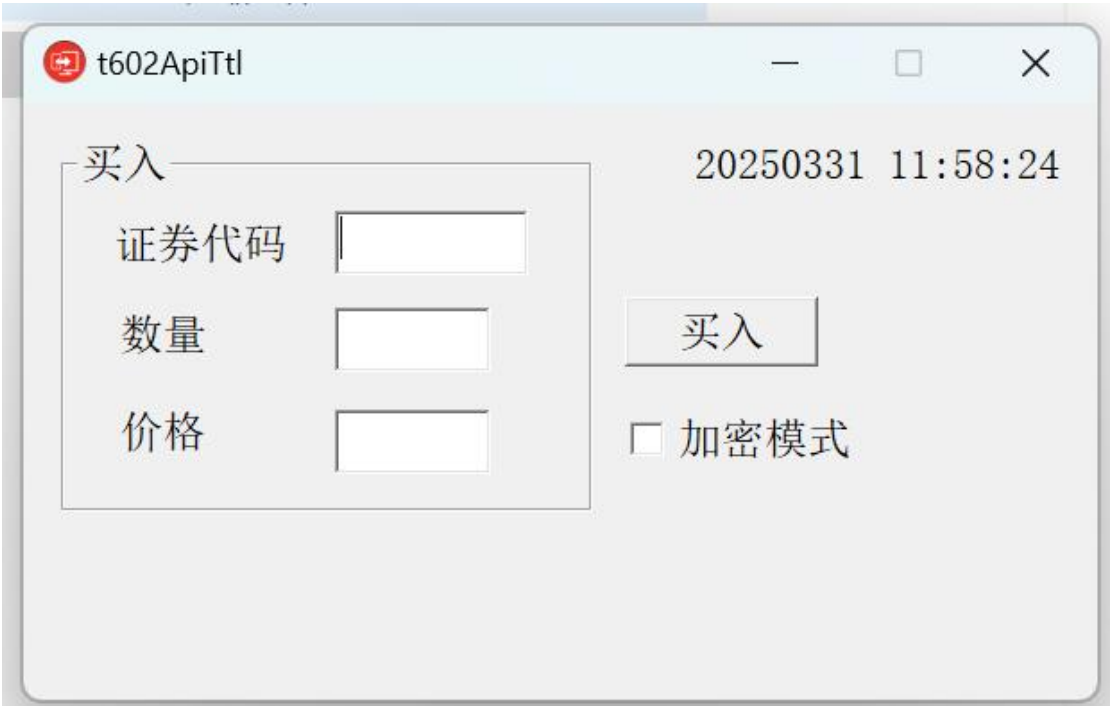
## 委托推送的三个场景

- 1、报单进入交易所-有效委托、或者废单
- 2、委托单有成交含部分成交和全部成交
- 3、撤销委托含部成部撤和全部撤单

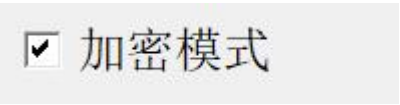
附：模拟测试程序 THClient.EXE



比如雷达星 在本地开启 58848 端口  
输入客户号、交易密码 接入



提供 简单买入指令测试。



当服务器使用加密模式通讯时 勾选

;通讯加密

---

Encrypt=1

详见 《托管部署说明》