Thsrobot使用指南

一、程序简介

Thsrobot 是一款专为同花顺设计的自动化交易工具，支持所有同花顺合作券商，具有以下特点：

- **轻量高效**：程序体积小，下单任务响应速度极快（1秒内完成）
- **安全可靠**：完全本地运行
- **功能全面**：支持 账户资金查询、 持仓查询、 委托查询、 成交查询、 股票买入、 股票卖出等操作。

二、安装准备

1. 系统要求

- **操作系统**：使用 Windows 10及以上版本，云服务使用 Window Server2019 及以上（64位）
- **必要软件**：同花顺官方PC版
- **Microsoft Visual C++**：系统上需要安装 Microsoft Visual C++ 可再发行程序包，这个需要你更新到最新版本状态； window 64下载、window 32下载，更多查看Microsoft Visual C++

2. 环境配置

（1）安装同花顺

1. 下载地址：同花顺官网下载
2. 安装步骤：
    - 默认安装即可，不同版本安装目录结构可能有所差异，一般安装在目录 `C:\同花顺软件\同花顺\同花顺`
    - 首次手动登录并添加券商账户 (打开同花顺，在右上角，点击“委托”，添加券商账户，选择你的证券公司，完成账号登陆)
    - 完成交易界面测试（确保能正常打开下单界面）

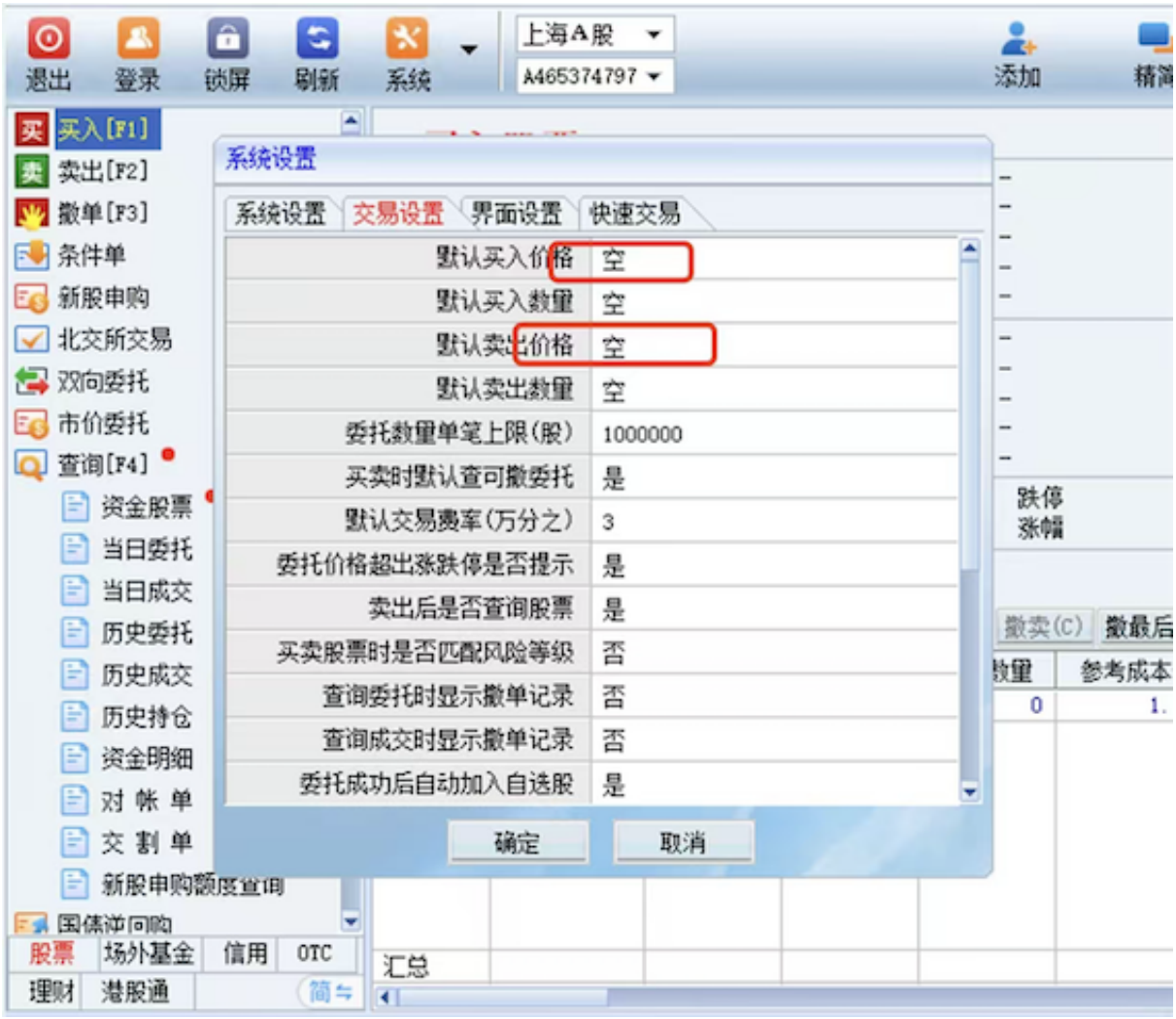
（2）安装最新 Microsoft Visual C++

1. 下载地址：window 64下载、window 32下载，更多查看Microsoft Visual C++
2. 安装步骤：
    - 默认安装即可

三、软件配置

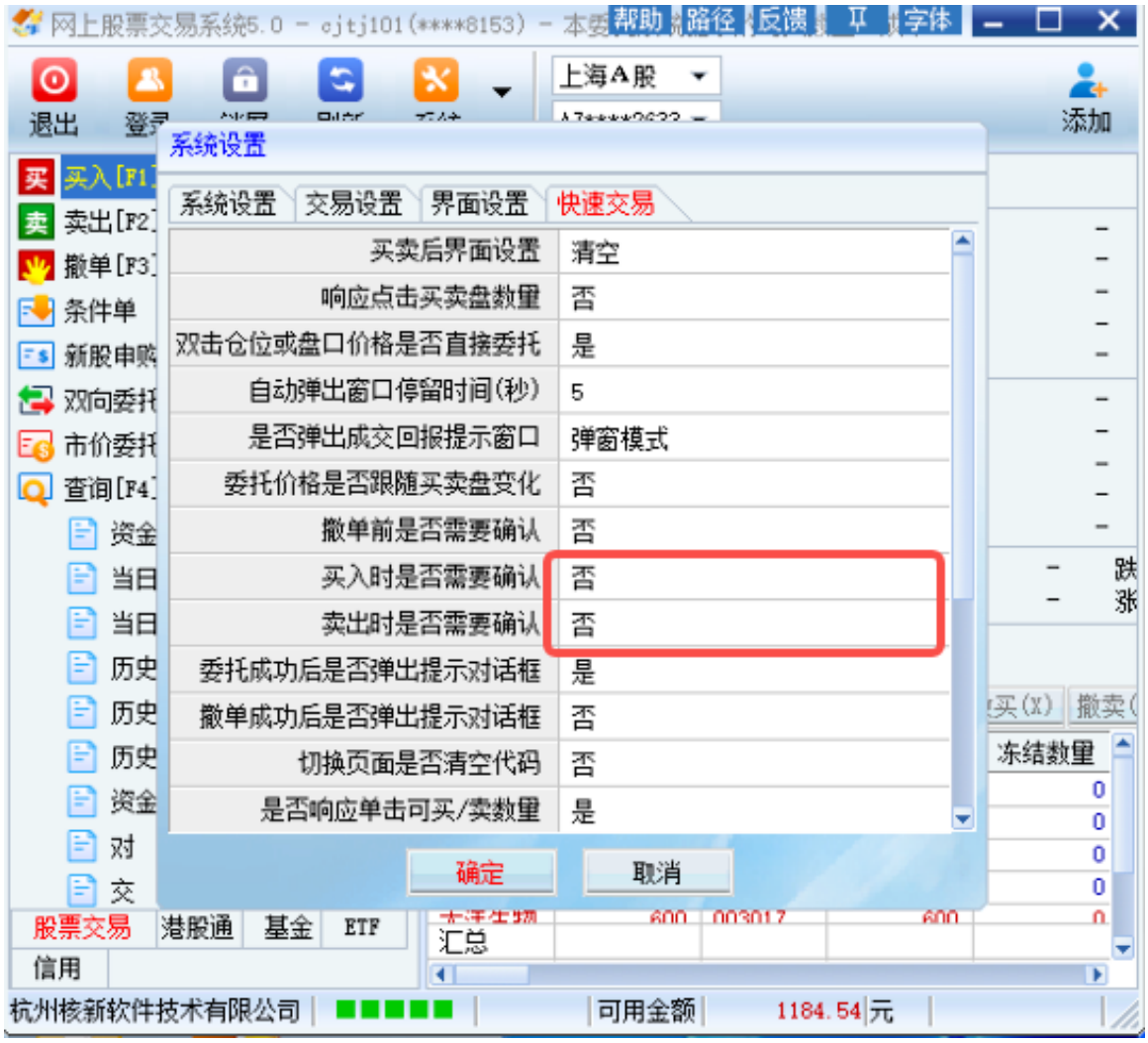
1. 同花顺关键设置

- **价格设置**（重要！！！）：
  - `系统 > 交易设置 >`
    - `默认买入价格` → 设置"空"
    - `默认卖出价格` → 设置"空"
  - 示例截图：



- **快捷交易**：
  - `系统 > 快速交易 >`
    - `买入时是否需要确认` → 设为"否"
    - `卖出时是否需要确认` → 设为"否"

- 示例截图：



2. THSRobot 部署

1. 创建专用目录（推荐 `C:\app`）
2. 将 `thsrobot.exe` 放入目录
3. **首次运行**：右键选择 `以管理员身份运行`

四、配置文件说明

首次运行后自动生成 `config.yaml`，需修改以下关键参数，修改完成再次运行thsrobot.exe 程序：

- 备注：你也可以提前手动创建配置

```
login-auth:
  account: 资金账户
  company: 证券公司
  credit-account: 你的信用账户
  credit-password: 你的信用密码
  password: 资金密码
  ths-account: 同花顺账户
```

```
  ths-password: 同花顺密码
  transaction-mode: stock
mode: prod
platform:
  password: 量子象限平台密码
  url: https://dev.liangzxx.com/api/thsrobot
  username: 量子象限平台账户
port: 80
web-auth:
  disable: false
  secret-id: Do1W7EmBM3WxofnYQR_05A==
  secret-key: n50OX6RFDkWyDjORmzm_z45iTcNQw04WmnZkrIBqqEI=
```

五、API 接口速查

基础路径：`http://localhost:80/api/v1`

普通账户接口

功能	请求方式	接口路径	请求参数示例（JSON）	说明
同步买入	POST	`/stock/buy`	`{"code":"002188","price":1.22,"volume":100,"acceptRisk":false}`	直接返回操作结果，可能包含失败提示
异步买入	POST	`/stock/sync/buy`	`{"code":"002188","price":1.22,"volume":100,"acceptRisk":false}`	提交后立即返回"提交成功"，实际结果需通过其他接口查询
同步卖出	POST	`/stock/sell`	`{"code":"002188","price":1.22,"volume":100,"acceptRisk":false}`	同同步买入逻辑
异步卖出	POST	`/stock/sync/sell`	`{"code":"002188","price":1.22,"volume":100,"acceptRisk":false}`	同异步买入逻辑
同步撤单	POST	`/stock/cancel`	`{"cancelType": 0}`	`cancelType`：0=全部撤单，1=撤买入单，2=撤卖出单
异步撤单	POST	`/stock/sync/cancel`	`{"cancelType": 0}`	同同步撤单逻辑
查询资金信息	GET	`/stock/assets`	-	返回可用金额、总资产、持仓盈亏等数据
查询委托记录	GET	`/stock/order`	-	返回当前所有委托单的详细信息（委托价格、数量、状态等）
查询持仓	GET	`/stock/position`	-	返回持仓股票的代码、数量、成本价、盈亏等
查询成交记录	GET	`/stock/trade`	-	返回历史成交记录（成交时间、价格、数量等）

接口响应通用格式

```
{
    "status": 0,        // 0=成功，非0=失败
    "message": "提示信息",
    "data": {}          // 具体数据（查询类接口返回）
}
```

六、安全认证

所有API请求需添加鉴权头，Python示例代码：

```python
# coding:utf-8

import hashlib
import hmac
import json
import time
import uuid
from urllib.parse import urlparse, parse_qs, urlencode

import requests
from requests.auth import AuthBase

your_server_addr ="http://localhost:8080/api/v1/stock/"
secretId ="bWYgDxe1ZBiQK4Tt4XCP6vYCWY3QuYxm"
secretKey ="bWYgDxe1ZBiQK4Tt4XCP6vYCWY3QuYxm"

# 认证签名
class SignAuth(AuthBase):
    def __init__(self, secret_id:str = secretKey, secret_key:str = secretKey):
        self.secret_id = secret_id
        self.secret_key = secret_key

    def __call__(self, r):
        # 获取当前时间戳和nonce
        timestamp = str(int(time.time()))
        nonce = str(uuid.uuid4())
        body = r.body or b""

        parsed_url = urlparse(r.url)
        query_params = parse_qs(parsed_url.query)  # 获取查询参数字典
        # 对查询参数进行排序
        sorted_query_params = dict(sorted(query_params.items()))
        # 将排序后的查询参数重新编码为字符串
        sorted_params_str = urlencode(sorted_query_params, doseq=True)

        # 构造待签名字符串
        sign_data = [
            r.method,
            r.path_url.split("?")[0],
            sorted_params_str,
```

```
            timestamp,
            nonce,
            body.decode('utf-8') if isinstance(body, bytes) else body
        ]

        sign_data = '\n'.join(sign_data)

        # 使用HMAC算法和SHA256哈希函数创建签名
        signature = hmac.new(self.secret_key.encode('utf-8'), sign_data.encode('utf-8'),
hashlib.sha256)

        # 将签名转换为Base64编码的字符串
        signature = signature.digest().hex()

        # 添加必要的认证头
        authorization = f"hmac id=\"{self.secret_id}\", ts=\"{timestamp}\", nonce=\"
{nonce}\", sig=\"{signature}\""

        r.headers['Authorization'] = authorization
        return r




#股票买入方法
def buy_stock(stock_code,price,vol):
    start_time = time.time()
    print('开始买入:'+stock_code+' 价格: '+str(price)+' 数量: '+str(vol))
    result=requests.post(your_server_addr+"buy", json={
        "code": stock_code,
        "price": price,
        "volume": vol
    }, auth=SignAuth())
    print(result.json())
    end_time = time.time()
    elapsed_time = end_time - start_time
    print(f"买入执行耗时: {elapsed_time} 秒")

#股票卖出方法
def sell_stock(stock_code,price,vol):
    start_time = time.time()
    print('开始卖出:'+stock_code+' 价格: '+str(price)+' 数量: '+str(vol))
    stock_code=stock_code[:6]
    result=requests.post(your_server_addr+"sell", json={
        "code": stock_code,
        "price": price,
        "volume": vol
    }, auth=SignAuth())
    print(result.json)
    end_time = time.time()
    elapsed_time = end_time - start_time
    print(f"卖出执行耗时: {elapsed_time} 秒")
```

```
#委撤撤单方法
def cancel_stock(cancelType):
    start_time = time.time()
    print('开始全部撤单')
    result=requests.post(your_server_addr+"cancel", json={
        "cancelType": cancelType
    }, auth=SignAuth())
    print(result.json)
    end_time = time.time()
    elapsed_time = end_time - start_time
    print(f"卖出执行耗时: {elapsed_time} 秒")

#获取账户资金方法
def get_account():
    result=requests.get(your_server_addr+"funding", auth=SignAuth()).text
    data_dict=json.loads(result)
    return data_dict['data']

#获取委托信息方法
def get_order():
    result=requests.get(your_server_addr+"order", auth=SignAuth()).text
    data_dict=json.loads(result)
    if 'data' in data_dict.keys():
        return data_dict['data']
    else:
        return

#获取持仓方法
def get_position():
    result=requests.get(your_server_addr+"position", auth=SignAuth()).text
    data_dict=json.loads(result)
    return data_dict['data']


############################以下为方法的调用示例#################################
if 1:
    #获取账号账户信息
    print('开始获取账户信息')
    account1=get_account()
    print(account1)
    account_total1=float(account1['total'])
    print("账号总资金: "+str(account_total1))

if 1:
    #获取持仓信息
    print('开始获取持仓信息')
    holdings_dict={}
    holdings = get_position()
    if holdings and len(holdings)>0:
        for i in holdings:
            if int(i['可用余额'])>0:
```

```
                stock_code=i['证券代码']
                holdings_dict[stock_code]=int(i['可用余额'])
    print('账号持仓')
    print(holdings_dict)

if 1:
    #股票买入,注：此处买入的价格，必须为现价的+-2%以内（不然会被交易所废单 ），买入数量必须是100的整数倍
    buy_stock('002936',2.01,100)

if 1:
    #获取委托信息
    print('开始获取委托信息')
    print(get_order())


if __name__ == '__main__':
    print(get_order())
    print(get_account() )
    print(get_position() )
    print(get_position() )
```

七、注意事项

1. 客户端状态管理

- **保持同花顺登录**
  - 程序运行期间，同花顺客户端必须保持登录状态
  - 程序运行期间，禁止对桌面做任何操作

2. 交易操作规范

- **测试验证**
  - 首次使用前，**必须通过小额订单（如1手）验证功能正常性**
  - 检查成交记录是否与预期一致（通过 `/stock/trade` 接口）
- **价格输入规范**
  - 价格参数需精确到小数点后两位（如 `1.22`）

3. 系统兼容性

- **运行环境限制**
  - 仅支持Windows系统（推荐Windows 10及以上版本， Window Server2019及以上）

4. 异常处理

- **验证码识别失败**
  - 若频繁失败，手动登录同花顺清除验证码缓存
- **接口调用错误**
  - `code` 非0时，根据 `message` 字段排查：

八、详细视频教程

实盘自动化工具安装、配制、使用，以及聚宽策略接入示例及讲解，见《视频教程》

工具使用以及量化学习方面的问题，可联系量子象限客服交流