

简介

CTP 接口是连接交易柜台的最终门户，PeopleQuant 力图解决交易前的风控、交易后的数据反馈、各种交易场景下的数据查询，解决交易上的“最后一公里问题”，并能嵌入其他交易平台，借助其他平台的优势，以及结合 Python 支持的科学计算、机器学习、AI 等现代技术，开发各种高效策略。

PeopleQuant 基于 openctp 编译的 Python 版 CTP 接口开发。openctp 官网：
<http://openctp.cn/TTS-CTPAPI.html>

PeopleQuant 支持 simnow 模拟、openctp 仿真、实盘交易(期货、期权)、离线数据回测。
欢迎收藏本项目：

github 项目地址: <https://github.com/zhuxueli-cta/PeopleQuant>

gitee 项目地址: <https://gitee.com/zhuxueli-cta/people-quant>

核心优势

CTP 直连：本地部署策略代码更安全，自动选择网络延时最低的柜台前置，让交易快人一步。

指令检查：多维度的指令检查，多方位防范错误交易指令发向交易所。

强大的功能组件：强大高效的业务数据查询、交易函数，使各种策略开发场景更便利。

跨平台：可嵌入其他交易平台，借助其他平台的优势，以及结合 Python 支持的科学计算、机器学习、AI 等现代技术，开发各种高效策略。

支持最新 Python：支持最新 Python，拥抱现代科技，让策略开发有无限可能。

可扩展：作为 Python SDK，可借助 PeopleQuant 开发个性化交易软件。

使用简单：业务函数简单易用，学习曲线平缓

生产级的学习示例：生产级的学习用例可快速上手，也可作为多种策略场景下的开发模板。

.....

QQ 交流群

1016612332



PeopleQuant学习...

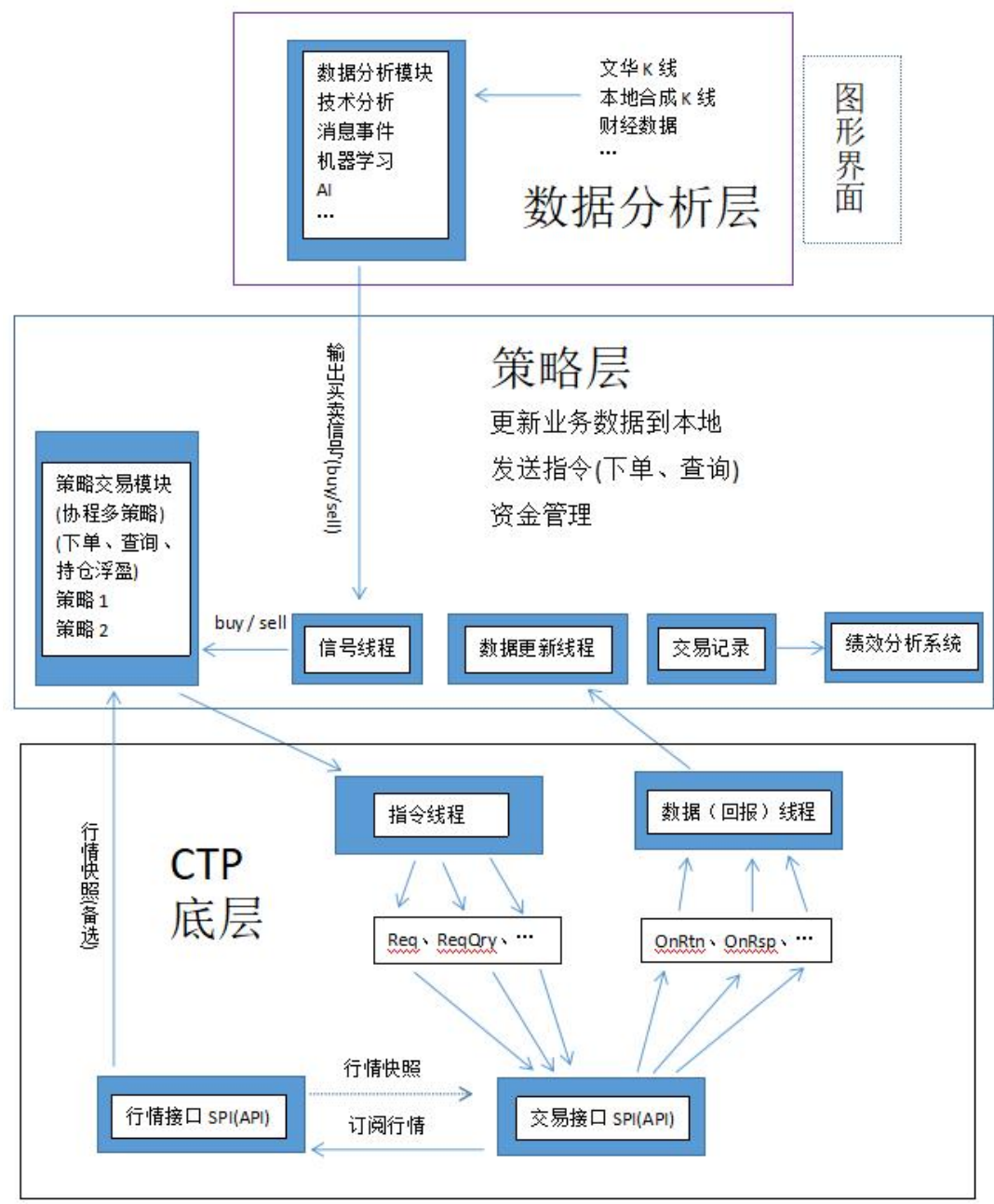
群号: 1016612332



微信公众号



技术架构



安装

安装需 Python3.10 及以上环境，支持 Windows10、11，Linux

```
pip install -U peoplequant
```

安装完成后，项目路径在 Python 扩展库目录下：\site-packages\peoplequant

目录结构

```
└─peoplequant 根目录
  │
  │ example.py    示例
  │ indicators.py 技术指标模块
```

- | | pqctp.py API 接口，编写策略从导入 api 开始
- | | zhuchannel.py 自定义线程模块，编写多线程场景使用
- | | zhustruct.py 业务数据对象模块，可查看数据字段含义及类型
- | | __init__.py
- | |
- | | └─backtest/ 回测模块
- | | └─datastruct_error CTP 接口文件，用于查看业务数据字段含义、取值范围
 - | | 6.7.11_API 接口说明.chm
 - | | error.dtd
 - | | error.xml
 - | | ThostFtdcMdApi.h
 - | | ThostFtdcTraderApi.h
 - | | ThostFtdcUserApiDataType.h 数据字段取值范围
 - | | ThostFtdcUserApiStruct.h 数据结构
- | |
- | | └─example 示例
 - | | example.py
 - | | K 线合成.py
 - | | 交易所标准组合交易.py
 - | | 发邮箱_钉钉_多线程.py
 - | | 多策略隔离_多线程.py
 - | | 无人监控.py
 - | | 智能调仓_协程.py
 - | | 智能调仓_多线程.py
 - | | 移动止损.py
 - | | 远程管理策略.py
- | |
- | | └─libs/ CTP 链接库
- | └─穿透式认证
 - | 穿透式测试流程.docx
 - | 穿透认证_交易指令检查.py
 - | 穿透认证_基础交易功能.py
 - | 穿透认证_应急处置暂停交易.py
 - | 穿透认证_批量撤单.py
 - | 穿透认证_报撤单笔数信息量等监测.py
 - | 穿透认证_系统连接状态异常监测.py
 - | 穿透认证_连通性.py
 - | 穿透认证_重复报单监测.py
 - | 穿透认证_错误防范.py
 - | 穿透认证_阈值设置及预警.py

接口介绍

业务数据类型

持仓（Position）、行情（Quote）、资金（Account）、委托单（Order）、成交单（Trade）的数据类型为类字典结构，可以像 Python 字典一样使用，为动态数据类型，由后台线程自动更新数据，由业务函数（如 `get_position`）返回的这些数据对象，在后续使用中直接为最新值，无需用业务函数重复获取。

业务数据的字段保持和 CTP 原始数据字段一致性，一般为大写字母开头（如 `InstrumentID` 表示合约代码），由 PeopleQuant 新增而 CTP 没有的字段，则以小写字母开头，例如持仓浮动盈亏（`float_profit_long`）。

tick、K 线为 `polars` 类型的数据（值为 `MutableDataHolder.data` 属性），可利用 `polars` 向量化计算多个技术指标，tick、K 线也被封装为动态数据类型（`MutableDataHolder`）。关于 `polars` 的用法推荐使用豆包、千问等 AI 工具查询。

业务数据类型的字段名称及含义参考模块 `zhustruct.py`，主要 CTP 原始数据字段的取值范围参考文件夹 `datastruct_error` 中的 CTP 接口文档和头文件。

业务函数

业务函数（如 `get_quote`、`insert_order`）为阻塞式设计，函数执行成功会返回业务数据，执行失败会返回空值或失败原因。

登录初始化

初始化 `pqapi` 时，会自动查询持仓、资金、委托单、成交单，并对持仓合约订阅行情、查询合约的保证金、手续费，这些基础工作准备完成后（日志输出提示“初始化完成”），才可以进行下一步操作。

技术指标

技术指标库 `indicators.py` 涵盖几十种常用的技术指标，利用 `polars` 向量化计算，比传统 `pandas` 数据计算速度快几倍。

离线数据回测

支持 tick 和 K 线级离线数据回测，高速回测系统大大缩短策略迭代的时间成本，回测代码和实盘高度一致，减少开发复杂度。

传入的离线数据作为基础数据生成 CTP 的行情快照，可再生成其他周期 K 线。（数据不标记品种，因此也可传入股票等其他标的的数据回测）。支持传入多品种离线数据以做多品种复合策略回测。

成交规则：成交价格为下单价格，开仓时更新开仓均价，平仓时均价不变。

由于后台线程推送数据的速度极快，策略端可能刚处理完一条数据的更新，后台就已经推送完几天的数据，为了避免漏掉数据（实盘当中不需要对已经成为历史的数据再参与当下的计算），需要为策略代码当中的合约添加阻塞，只需要加两句代码，在信号循环计算开始前，添加一句：

```
#添加新行情是否被使用记录
```

```
quote_label=pqapi.update_quote_pipe([symbol],add_or_used='add')
```

在对新数据计算完成后，添加一句：

```
#新行情已经被使用完毕
```

```
pqapi.update_quote_pipe([symbol],quote_label,add_or_used='used')
```

具体用法可参考回测示例 `backtest_demo.py` 和公众号文章的说明。

支持期货公司

使用 **PeopleQuant** 已完成厂商穿透式认证的期货公司（如宏源期货），只需要按文档说明输入期货公司名称（如'H 宏源期货_主席'）、期货账户、密码，不需要输入 **APPID**、认证授权码、前置地址。

若需要自行做穿透式测试的期货公司，可按照安装包里穿透式测试示例进行穿透式测试，并获得期货公司授权码，测试时将初始化参数 **ProductionMode** 设置为 **False**（表示测评版，CTP6.7.11 版本）。其他参数参考示例程序中 **simnow** 账户的参数填写。

示例程序

安装包下的示例程序，详尽、实用，可快速帮你学习 **PeopleQuant** 的用法。示例程序更多的文字介绍可通过公众号查看，或通过视频课程进一步学习。

快速入门

1、登录账户

实例化 **PeopleQuantApi** 的过程也是登录账户的过程，