

## 在装完docker以后 quantaxis的推荐路径

• 发布于 8 个月前 • 作者 yutiansut (/user/yutiansut) • 4682 次浏览 • 来自 文档

### 1. 首先确认你的docker-compose.yaml

如果你是股票方向的 ==> 选择 qa-service 下的docker-compose.yaml

如果你是期货方向的 ==> 选择 qa-service-future 下的docker-compose.yaml

你可以理解 docker的构成类似搭积木的模式, 你需要这个功能的积木, 就选择他放在你的docker-compose.yaml里面

期货方向的yaml 比股票多一个 QACTPBEE的docker-container [这是用于分发期货的tick行情所需的 股票则无需此积木]

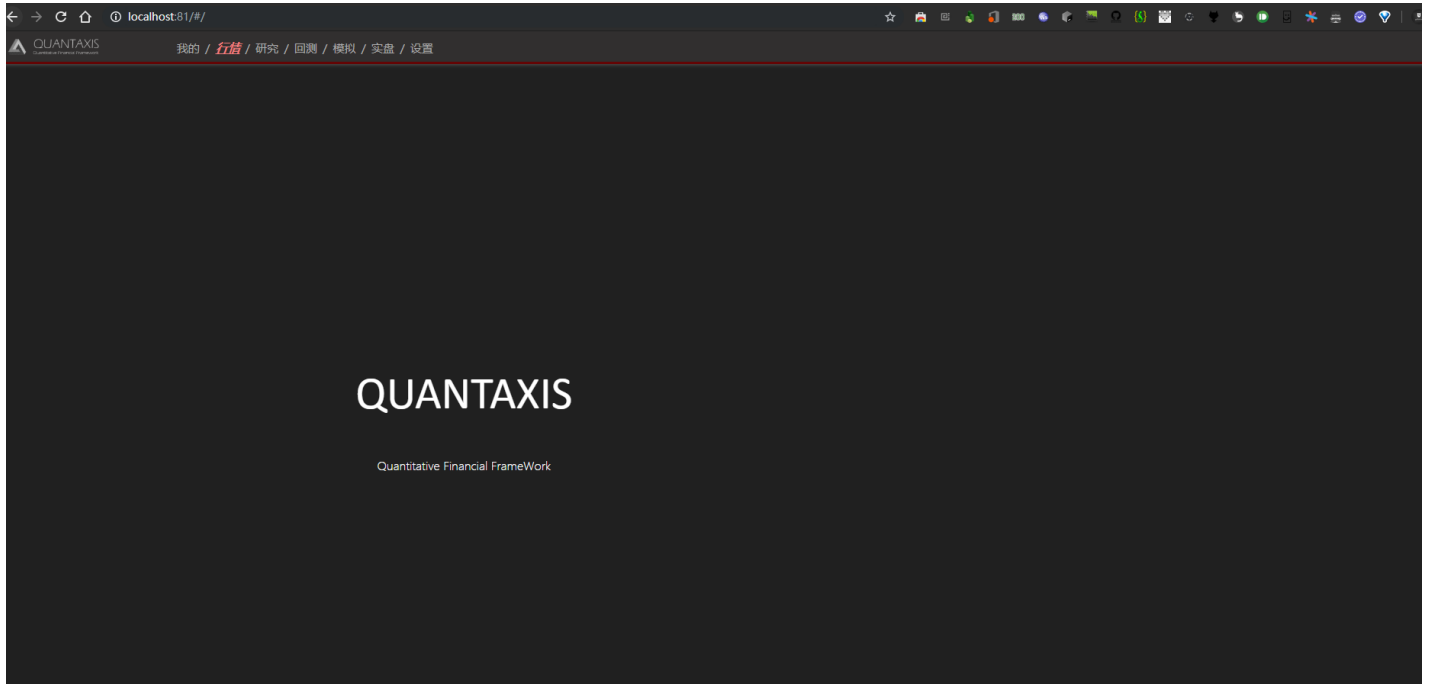
### 2. 下载了docker-compose.yaml以后, cd到这个目录, 输入 docker-compose up -d

docker-compose up -d 的意思是在后台启动这个docker环境, 如果你需要打印输出, 则把-d去掉即可

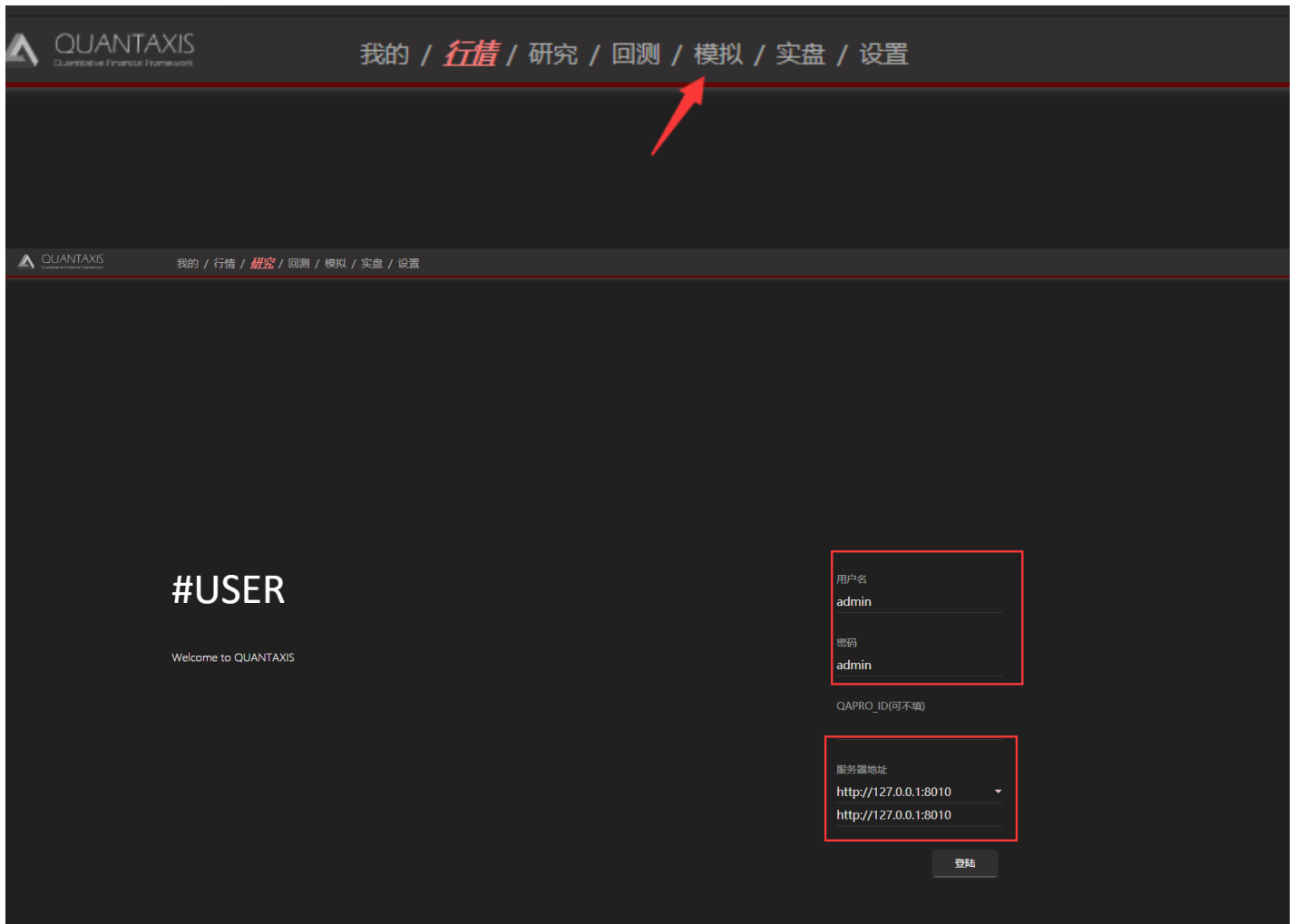
### 3. 在没有什么报错的情况下, 打开浏览器 输入 localhost:81

( 注意: 如果你是 win7/8 以及 win10家庭版用户, 你会遇到 localhost:81是打不开的情况, 因为你们使用的 docker toolbox 没有自动映射端口出来, 详细的步骤参见 dockertoolbox + win7.8.10家庭版的指南 (<http://www.yutiansut.com:3000/topic/5db9ac40d7180005f0020eb8>)

输入了localhost:81 后, 你应该可以看见这个界面



在上方随意点击栏目, 你都可以进入登录界面



登陆界面 的用户名/密码 是 你在做回测的时候的 QAUser的用户密码

一般 使用QAStrategy这个项目的默认密码是 admin/ admin

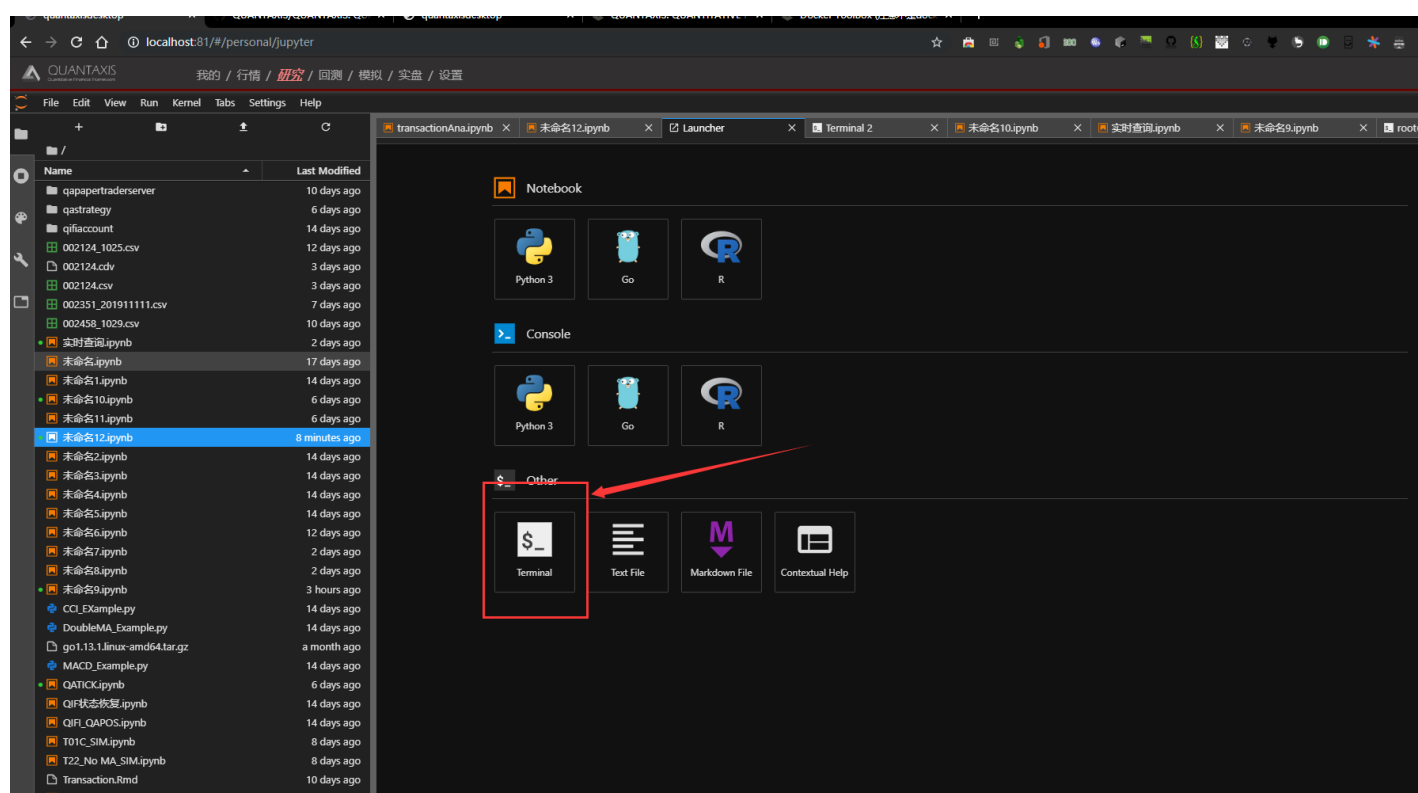
你也可以自由输入

底下的登陆地址, 是你部署了docker/ 或者你想联入的远程地址的8010端口

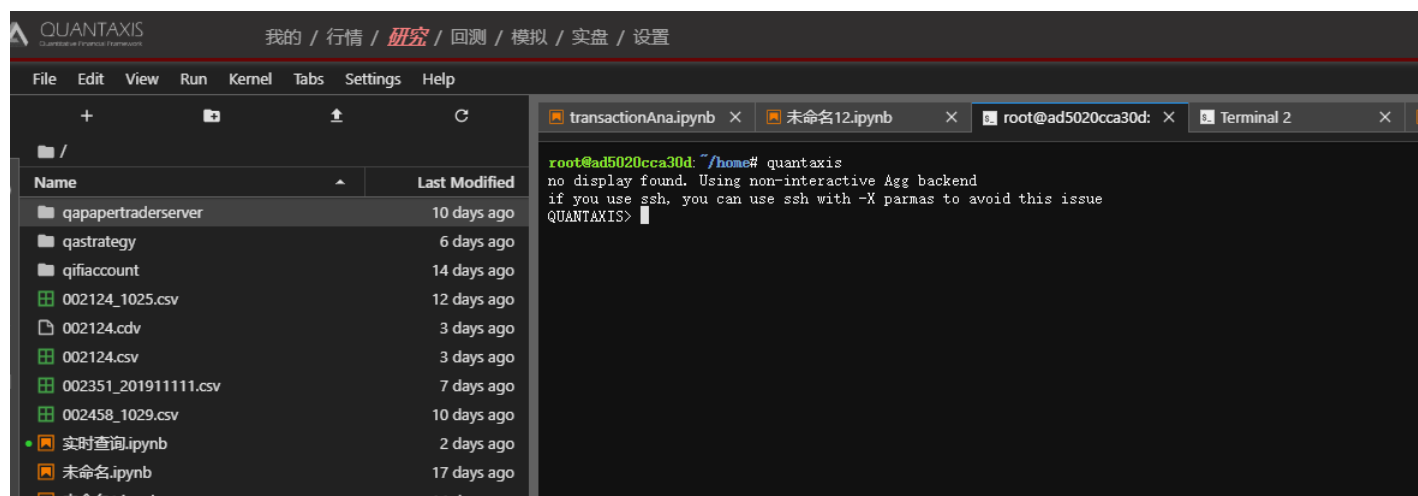
## 4. 当你点击“研究”这一栏, 你会进入到一个jupyter的登陆界面,

在此界面输入 quantaxis 这是默认密码

## 5. 进入这个环境以后, 第一步 是去存储你想要的数据, 在此界面 点击terminal



## 6. 在点开的terminal界面中 , 输入 quantaxis 回车, 进入 quantaxis cli的命令行界面



在命令行界面 输入 save 按回车, 你可以看到许多命令行选项

```

root@ad5020cca30d: /home# quantaxis
no display found. Using non-interactive Agg backend
if you use ssh, you can use ssh with -X parmas to avoid this issue
QUANTAXIS> save
Usage:
    命令格式: save all : save stock_day/xdxr/ index_day/ stock_list/index_list
    命令格式: save X|x : save stock_day/xdxr/min index_day/min etf_day/min stock_list/index_list/block
    命令格式: save day : save stock_day/xdxr index_day etf_day stock_list/index_list
    命令格式: save min : save stock_min/xdxr index_min etf_min stock_list/index_list
    命令格式: save future: save future_day/min/list
    命令格式: save ox: save option_contract_list/option_day/option_min/option_commodity_day/option_commodity_min
    命令格式: save transaction: save stock_transaction and index_transaction (Warning: Large Disk Space Required)

-----
    命令格式: save stock_day : 保存日线数据
    命令格式: save single_stock_day : 保存单个股票日线数据
    命令格式: save stock_xdxr : 保存日除权除息数据
    命令格式: save stock_min : 保存分钟线数据
    命令格式: save single_stock_min : 保存单个股票分钟线数据
    命令格式: save index_day : 保存指数日线数据
    命令格式: save index_min : 保存指数分钟线数据
    命令格式: save single_index_min : 保存单个指数分钟线数据
    命令格式: save future_day : 保存期货日线数据
    命令格式: save future_min : 保存期货分钟线数据
    命令格式: save etf_day : 保存ETF日线数据
    命令格式: save single_etf_day : 保存单个ETF日线数据
    命令格式: save etf_min : 保存ETF分钟数据
    命令格式: save stock_list : 保存股票列表
    命令格式: save stock_block: 保存板块
    命令格式: save stock_info : 保存tushare数据接口获取的股票列表
    命令格式: save financialfiles : 保存高级财务数据(自1996年开始)
    命令格式: save option_contract_list 保存上市的期权合约信息 (不包括已经过期摘牌的数据)
    命令格式: save 50etf_option_day : 保存50ETF期权日线数据 (不包括已经过期摘牌的数据)
    命令格式: save 50etf_option_min : 保存50ETF期权分钟线数据 (不包括已经过期摘牌的数据)
    命令格式: save option_commodity_day : 保存商品期权日线数据 (不包括已经过期摘牌的数据)
    命令格式: save option_commodity_min : 保存商品期权分钟线数据 (不包括已经过期摘牌的数据)
    命令格式: save option_day_all : 保存所有期权日线数据 (不包括已经过期摘牌的数据)
    命令格式: save option_min_all : 保存所有期权分钟数据 (不包括已经过期摘牌的数据)
    命令格式: save index_list : 保存指数列表
    命令格式: save etf_list : 保存etf列表
    命令格式: save future_list : 保存期货列表

-----
    if you just want to save daily data just
        save all+ save stock_block+save stock_info, it about 1G data
    if you want to save the fully data including min level
        save x + save stock_info

    @yutiansut
    @QUANTAXIS
    请访问 https://book.yutiansut.com/
QUANTAXIS>

```

一般来说, 如果你是股票用户

```

save all (股票/指数 的日线数据 | 权息数据 | 板块数据)
save x (股票/指数的 日线/分钟线数据 | 权息数据| 板块数据)

```

如果你是期货用户

```

save future_min_all (期货的全部合约的分钟线数据)
save future_min (q期货主连的分钟线数据)

```

```

save future_day_all (期货全部合约的日线数据)
save future_day (期货主连的日线数据)

```

```

save index_day (指数数据 此处也要存, 因为在做回测的时候, 需要沪深300作为标的对照物)

```

其他的save选项 财务/期权这些 可以仔细看选项

## 7. 存完数据以后, 可以打开一个notebook, 做个回测

```

import QUANTAXIS as QA
import numpy as np
import pandas as pd
import datetime

st1=datetime.datetime.now()

# define the MACD strategy
def MACD_JCSC(dataframe, SHORT=12, LONG=26, M=9):
    """
    1.DIF向上突破DEA，买入信号参考。
    2.DIF向下跌破DEA，卖出信号参考。
    """

    CLOSE = dataframe.close
    DIFF = QA.EMA(CLOSE, SHORT) - QA.EMA(CLOSE, LONG)
    DEA = QA.EMA(DIFF, M)
    MACD = 2*(DIFF-DEA)

    CROSS_JC = QA.CROSS(DIFF, DEA)
    CROSS_SC = QA.CROSS(DEA, DIFF)
    ZERO = 0

    return pd.DataFrame({'DIFF': DIFF, 'DEA': DEA, 'MACD': MACD, 'CROSS_JC': CROSS_JC, 'CROSS_SC': CROSS_SC, 'ZERO': ZERO})


# create account
user = QA.QA_User(username='quantaxis', password='quantaxis')
portfolio = user.new_portfolio('qatestportfolio')

Account = portfolio.new_account(account_cookie='macd_stock', init_cash=1000000)
Broker = QA.QA_BacktestBroker()

QA.QA_SU_save_strategy('MACD_JCSC','Indicator',Account.account_cookie)
# get data from mongodb
QA.QA_SU_save_strategy('MACD_JCSC', 'Indicator',
                        Account.account_cookie, if_save=True)
data = QA.QA_fetch_stock_day_adv(
    ['000001', '000002', '000004', '600000'], '2017-09-01', '2018-05-20')
data = data.to_qfq()

# add indicator
ind = data.add_func(MACD_JCSC)
# ind.xs('000001', level=1)['2018-01'].plot()

data_forbacktest=data.select_time('2018-01-01','2018-05-01')

for items in data_forbacktest.panel_gen:
    for item in items.security_gen:
        #####

```

```
daily_ind=ind.loc[item.index]
```

```
if daily_ind.CROSS_JC.iloc[0]>0:
```

```
    order=Account.send_order(  
        code=item.code[0],  
        time=item.date[0],  
        amount=1000,  
        towards=QA.ORDER_DIRECTION.BUY,  
        price=0,  
        order_model=QA.ORDER_MODEL.CLOSE,  
        amount_model=QA.AMOUNT_MODEL.BY_AMOUNT  
    )
```

```
    #print(item.to_json()[0])
```

```
    Broker.receive_order(QA.QA_Event(order=order,market_data=item))
```

```
    trade_mes=Broker.query_orders(Account.account_cookie,'filled')
```

```
    res=trade_mes.loc[order.account_cookie,order.realorder_id]
```

```
    order.trade(res.trade_id,res.trade_price,res.trade_amount,res.trade_time)
```

```
elif daily_ind.CROSS_SC.iloc[0]>0:
```

```
    #print(item.code)
```

```
    if Account.sell_available.get(item.code[0], 0)>0:
```

```
        order=Account.send_order(  
            code=item.code[0],  
            time=item.date[0],  
            amount=Account.sell_available.get(item.code[0], 0),  
            towards=QA.ORDER_DIRECTION.SELL,  
            price=0,  
            order_model=QA.ORDER_MODEL.MARKET,  
            amount_model=QA.AMOUNT_MODEL.BY_AMOUNT  
        )
```

```
        #print
```

```
        Broker.receive_order(QA.QA_Event(order=order,market_data=item))
```

```
        trade_mes=Broker.query_orders(Account.account_cookie,'filled')
```

```
        res=trade_mes.loc[order.account_cookie,order.realorder_id]
```

```
        order.trade(res.trade_id,res.trade_price,res.trade_amount,res.trade_time)
```

```
Account.settle()
```

```
print('TIME -- {}'.format(datetime.datetime.now()-st1))
```

```
print(Account.history)
```

```
print(Account.history_table)
```

```
print(Account.daily_hold)
```

```
# create Risk analysis
```

```
Risk = QA.QA_Risk(Account)
```

```
Account.save()
```

```
Risk.save()
```



## 8 当然 我们也推荐你使用最新的QAStrategy来做回测/模拟

---

首先 打开terminal (上面有讲), 输入

```
pip install qastrategy
```

然后 新建一个notebook, 输入

```
from QAStrategy import QAStrategyCTABase
import QUANTAXIS as QA
import pprint
```

```
class CCI(QAStrategyCTABase):
```

```
    def on_bar(self, bar):
```

```
        res = self.cci()
```

```
        print(res.iloc[-1])
```

```
        if res.CCI[-1] < -100:
```

```
            print('LONG')
```

```
            if self.positions.volume_long == 0:
```

```
                self.send_order('BUY', 'OPEN', price=bar['close'], volume=1)
```

```
            if self.positions.volume_short > 0:
```

```
                self.send_order('SELL', 'CLOSE', price=bar['close'], volume=1)
```

```
        elif res.CCI[-1] > 100:
```

```
            print('SHORT')
```

```
            if self.positions.volume_short == 0:
```

```
                self.send_order('SELL', 'OPEN', price=bar['close'], volume=1)
```

```
            if self.positions.volume_long > 0:
```

```
                self.send_order('BUY', 'CLOSE', price=bar['close'], volume=1)
```

```
    def cci(self,):
```

```
        return QA.QA_indicator_CCI(self.market_data, 61)
```

```
    def risk_check(self):
```

```
        pass
```

```
        # pprint.pprint(self.qifiacc.message)
```

然后 你可以自由指定回测/模拟

首先实例化这个类

```
strategy =CCI(code='RB2005', frequency='1min',strategy_id='a3916de0-bd28-4b9c-bea1-94d91f1744ac', start='2020-01-01', end='2020-02-07')
```

如果你需要测试这个策略

```
strategy.debug()
```

如果你需要做回测

```
strategy.run_backtest()
```

如果你需要让他直接挂模拟

在挂模拟的时候, 你需要注意一些东西

1. 挂模拟的标的需要和真实标的一致
2. 挂模拟的时候, 你的行情必须是有推送的, 并且申请了你所需要的的分钟线级别的数据

(如何申请行情数据? 你可以看这里 )

[https://github.com/yutiansut/QUANTAXIS\\_RealtimeCollector](https://github.com/yutiansut/QUANTAXIS_RealtimeCollector)

([https://github.com/yutiansut/QUANTAXIS\\_RealtimeCollector](https://github.com/yutiansut/QUANTAXIS_RealtimeCollector))

```
...
# 期货订阅请求
curl -X POST "http://127.0.0.1:8011?action=new_handler&market_type=future_cn&code=au1911"
```bash
# 股票订阅请求
curl -X POST "http://127.0.0.1:8011?action=new_handler&market_type=stock_cn&code=000001"
# 二次采样请求
curl -X POST "http://127.0.0.1:8011?action=new_resampler&market_type=future_cn&code=au1911&frequency=2min"
```

对于小白可能难以理解curl是个啥, 此处给出一个简单易懂的代码

```
import requests
requests.post("http://127.0.0.1:8011?action=new_handler&market_type=future_cn&code={}".format("rb2001"))
以此类推其他的请求都可以这么做
...
```

像 螺纹2001 合约, 你需要改成 rb2001 注意此处是小写

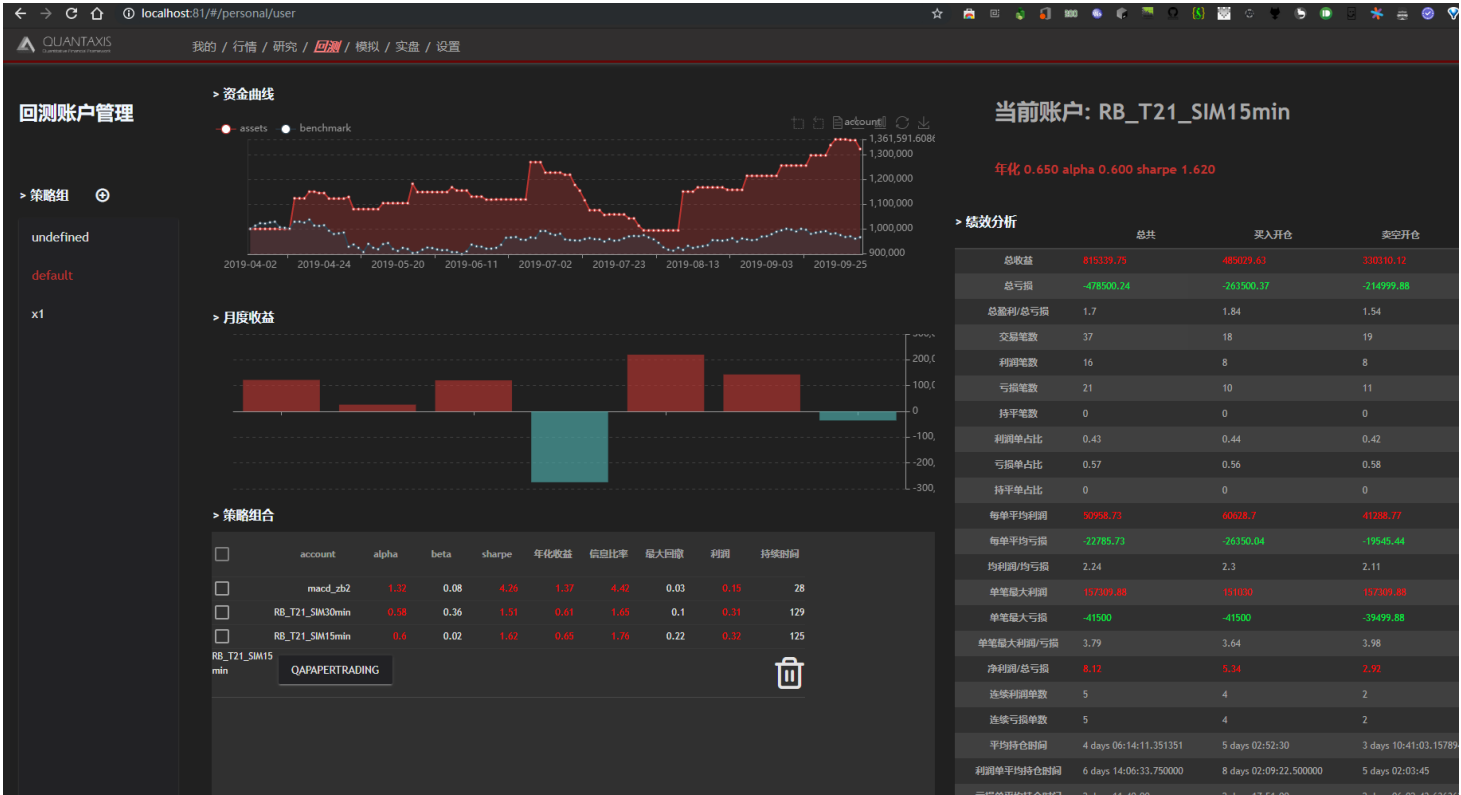
```
strategy =CCI(code='rb2001', frequency='1min',strategy_id='a3916de0-bd28-4b9c-bea1-94d91f1744ac')
strategy.debug_sim()
```



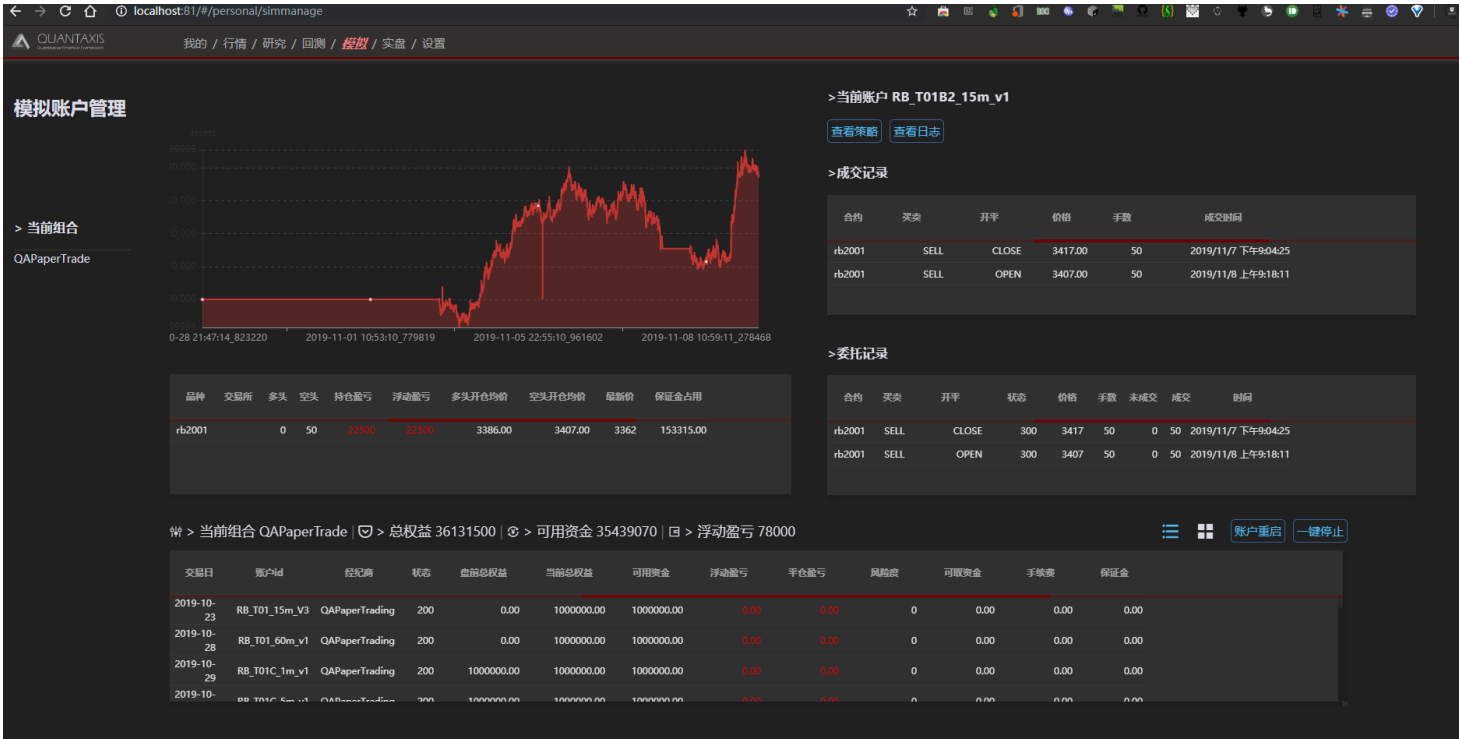
(/user/yutiansut) **yutiansut (/user/yutiansut)** 2楼•8 个月前 作者

1. 做完了这些操作以后, 你可以点击 回测 你就可以看到类似这样的结果






## 点击模拟



(/user/yutiansut) yutiansut (/user/yutiansut) 3楼•8 个月前 作者

对于 行情 界面, 切换则使用键盘精灵来实现




 (/user/walter211) **walter211 (/user/walter211)** 4楼•6 个月前

马

 (/user/richard2k) **richard2k (/user/richard2k)** 5楼•6 个月前

趁这个春节延长假期，好好学习一下，力争用起来

 (/user/yutiansut) **yutiansut (/user/yutiansut)** 6楼•5 个月前 作者

关联贴：

如何修改期货的实盘行情地址 <http://www.yutiansut.com:3000/topic/5dfade9efe01257b44740e70>

(<http://www.yutiansut.com:3000/topic/5dfade9efe01257b44740e70>)

实时如何申请行情 <http://www.yutiansut.com:3000/topic/5dd1be9b0c8e672840f3fea7>


(<http://www.yutiansut.com:3000/topic/5dd1be9b0c8e672840f3fea7>)

如何接入你的实盘期货账户 <http://www.yutiansut.com:3000/topic/5dc865e8c466af76e9e3bdd1>

(<http://www.yutiansut.com:3000/topic/5dc865e8c466af76e9e3bdd1>)

如何实现模拟盘/实盘的跟单 <http://www.yutiansut.com:3000/topic/5ddb5ba8fe01257b4474080a>

(<http://www.yutiansut.com:3000/topic/5ddb5ba8fe01257b4474080a>)

 (/user/yutiansut) **yutiansut (/user/yutiansut)** 7楼•5 个月前 作者

docker 如何访问外部数据库 <https://github.com/QUANTAXIS/QUANTAXIS/issues/1346>

(<https://github.com/QUANTAXIS/QUANTAXIS/issues/1346>)

<http://www.yutiansut.com:3000/topic/5e4531c96d3b182e88b4ebb4>

(<http://www.yutiansut.com:3000/topic/5e4531c96d3b182e88b4ebb4>)



(/user/yutiansut) **yutiansut (/user/yutiansut)** 8楼•5 个月前 作者

docker小白用户的推荐 <http://www.yutiansut.com:3000/topic/5e4cb13f6d3b182e88b4ef64>  
(<http://www.yutiansut.com:3000/topic/5e4cb13f6d3b182e88b4ef64>)

---



(/user/kmmao) **kmmao (/user/kmmao)** 9楼•3 个月前

棒

---



(/user/lovingfantasy) **lovingfantasy (/user/lovingfantasy)** 10楼•3 个月前

mark。节约时间，少走弯路！