1.入币圈

我是上个世纪山财的金融本科大学生，当时的互联网正经历着快速崛起的年代，我接触计算机也就是在那个时候，自从在大学里创办网络协会之后便走上了一条离经叛道的“不归路”。毕业那年，同学们都在憧憬着去哪家银行或者证券交易所的时候，我却选择放弃本专业去做一个苦逼的程序员，那全是因为热爱。

随着后来自己创办软件公司，有了一部分资金之后便又重新捡起了本专业—金融，开始是用空余时间做做股票、接着是期货、然后是外汇、黄金。之后出现了比特币，那时候我是不看好这玩意的，直到后来，随着投资酒店、电商等产业的失败，资金已经捉襟见肘，茫然无措的时候才发现比特币已如日中天，当年公司里的小伙子已经靠这东西成功实现了财富自由，没办法四十几岁的年纪才重新开始认真的审视币圈，并决定加入它。

2.搞量化

在经历了很多个盯盘的不眠之夜后，作为一个老程序员，我开始思考有没有办法让程序去替我交易，这样我就可以安心睡觉了。其实那个时候我就找到fmz了，但是时代变了，我所熟悉的C++已经没多少人在用了，在新的平台上一切需要重新开始学习，只不过这对老年人有点困难，开始的时间我完全没有搞懂怎么用，然后就暂时放弃了，然后就自己对接API写机器人，写了些双均线、网格之类的策略，虽然并没有怎么赚到钱，但是也算是一脚踏入了量化大门，学会了用talib库。到后来因为因为一些事情就没有继续做下去，机器人也停了，又回归了手动看盘操作模式，但是频率已经很低了，时不时操作一两笔。

3.用FMZ

兜兜转转一圈之后，我还是回到了fmz，但是因为已经有了自己手写策略的经验，这次上手fmz就非常快速了。我发现fmz对接了几乎所有主流交易所，把交易、账户等等操作都统一封装起来，又方便又好用，写一份代码就可以到处跑了，对于一个花了老大力气手动封装过交易所api的人来说，简直就是找到了救星。

我在GitHub上找到了fmz公开的策略仓库，就像找到了宝藏一样，花了一两个晚上翻遍了所有的策略，复制过来回测，读策略逻辑，尝试改进，终于吃透了各种策略逻辑，在大势判断正确的前提下，已经可以完全放弃手动操作了。

4、布道

既然现在有这么方便的工具，我也乐得跟大家一起来分享，所以我有个想法就是在

这里开一个专栏去给大家讲解各种策略，让大家可以都实现交易的量化，不用再去为没日没夜的盯盘而烦恼了。不过开专栏好像有粉丝数的限制，我还是希望大家多多支持，让我们的专栏可以顺利开起来。

最后分享一个简易的对冲策略

```
var preSumBalance = 0
var initSumBalance = 0
function UpdateAccount(isFirst){
    var msg = ""
    var sumStocks = 0
    var sumBalance = 0
    for(var i = 0; i < exchanges.length; i++){
        if(exchanges[i].needUpdate == true || isFirst == true){
            exchanges[i].account = _C(exchanges[i].GetAccount)
            exchanges[i].needUpdate = false
            if(isFirst == true){
                initSumBalance += (exchanges[i].account.Balance + exchanges[i].account.FrozenBalance)
                exchanges[i].SetPrecision(_CurrencyPrecision, _BaseCurrencyPrecision)
            }
        }
        sumStocks += (exchanges[i].account.Stocks + exchanges[i].account.FrozenStocks)
        sumBalance += (exchanges[i].account.Balance + exchanges[i].account.FrozenBalance)
        msg += exchanges[i].GetName() + "币:" + exchanges[i].account.Stocks + "冻币:" + exchanges[i].account.FrozenStocks + "钱:" +
exchanges[i].account.Balance + "冻钱:" + exchanges[i].account.FrozenBalance + "\n"
    }
    LogStatus(_D(), "总币, " + sumStocks, "总钱, " + sumBalance, "\n", msg)
    if(preSumBalance != sumBalance){
        LogProfit(sumBalance - initSumBalance, preSumBalance = sumBalance)
    }
}
function main(){
    UpdateAccount(true)
    while(1){
        for(var i = 0; i < exchanges.length; i++){
            for(var j = 0; j < exchanges.length; j++){
                if(i == 0 && j == 0){
                    for(var m = 0; m < exchanges.length; m++){
                        exchanges[m].thread = exchanges[m].Go("GetTicker")
                    }
                    for(var n = 0; n < exchanges.length; n++){
                        exchanges[n].ticker = exchanges[n].thread.wait()
                    }
                }
                if(exchanges[i].GetName() != exchanges[j].GetName() && exchanges[i].ticker && exchanges[j].ticker && exchanges
[i].ticker.Buy - exchanges[j].ticker.Sell > _HedgePrice){
                    if(exchanges[i].account.Stocks > _HedgeAmount && exchanges[j].account.Balance / ((exchanges[i].ticker.Buy +
exchanges[j].ticker.Sell) / 2) > _HedgeAmount){
                        var sellId_i = exchanges[i].Sell((exchanges[i].ticker.Buy + exchanges[j].ticker.Sell) / 2, _HedgeAmount,
exchanges[i].GetName())
                        var buyId_j = exchanges[j].Buy((exchanges[i].ticker.Buy + exchanges[j].ticker.Sell) / 2, _HedgeAmount,
exchanges[i].GetName())
                        exchanges[i].needUpdate = exchanges[j].needUpdate = true
                    }
                }
            }
```