# 群益Python API

**群益Python API支援3大功能**

**報價**

**歷史K**

**下單**

**回報**

以下每一個步驟解說，皆包含簡短程式碼，最後附上完整程式碼讓你可以直接複製貼上，即可開始使用，一點都不麻煩。

下面是解釋透過Python開發使用群益API時，需要了解有那些程序：

1.初始化群益COM元件-工欲善其事，必先利其器。

2.使用Python建立windows視窗-規劃一個美美的個人化視窗吧。

3.啟動報價/抓歷史K-行情波動閃爍真迷人,心跳開始加速了嗎。

4.串接下單功能-配合報價資料，就可以來下單囉。

5.成果回報-立即來檢視一下獲利吧。

更版記錄：

|  |  |  |
| --- | --- | --- |
| 修改日期 | 元件版本編號 | 說明 |
| 2018/06/13 | 2.13.12 | 初版建立。 |
| 2019/11/02 | 2.13.19 | * 增加Python範例使用說明   (國內證券帳務、國內期選下單及帳務、國內證期選智慧單功能、海外期選下單及帳務, 回報含智慧單、配合非同步委託回傳架構調整   * 增加換版注意事項 |
| 2020/11/03 | 2.13.25 | 獨立註冊公告、SKCOM換版注意事項單元 |

# 如何使用python 串接SKCOM及建立介面

1. [初使化群益元件](#初使化群益元件)
2. [建立視窗介面](#建立windows畫面)
3. [環境設定](#環境設定)
4. 註冊公告
5. [換版注意事項](#換版注意事項)
6. [啟動報價](#啟動報價)
7. [群益下單功能](#群益下單功能)
8. [回報功能介紹](#回報功能介紹)

# *初始化群益元件*

**群益API COM元件由群益金融網下載，內含說明文件－教學你如何註冊。使用前及更新版本，需先註冊COM元件。**

群益API COM底層是由C++編寫而成，有強大運算能力且擁有速度上的優勢，而使用不同的程式語言必須要透過相關引用([安裝](#環境設定))，才能在Python範例程式碼直接介接群益API COM元件，以下為安裝套件後，介接x64 COM範例：

# 先把API com元件初始化

import comtypes.client

comtypes.client.GetModule(r'./x64/SKCOM.dll')

import comtypes.gen.SKCOMLib as sk

# 第一種讓群益API元件可導入讓Python code使用的方法

import win32com.client

from ctypes import WinDLL,byref

from ctypes.wintypes import MSG

SKCenterLib = win32com.client.Dispatch("{AC30BAB5-194A-4515-A8D3-6260749F8577}")

SKOrderLib = win32com.client.Dispatch("{54FE0E28-89B6-43A7-9F07-BE988BB40299}")

# 第二種讓群益API元件可導入Python code內用的物件宣告

skC = comtypes.client.CreateObject(sk.SKCenterLib,interface=sk.ISKCenterLib)

skO = comtypes.client.CreateObject(sk.SKOrderLib,interface=sk.ISKOrderLib)

skQ = comtypes.client.CreateObject(sk.SKQuoteLib,interface=sk.ISKQuoteLib)

skR = comtypes.client.CreateObject(sk.SKReplyLib,interface=sk.ISKReplyLib)

# 畫視窗用物件

from tkinter import \*

from tkinter.ttk import \*

from tkinter import messagebox,colorchooser,font,Button,Frame,Label

# 數學計算用物件

import math

# 顯示各功能狀態用的function

def WriteMessage(strMsg,listInformation):

listInformation.insert('end', strMsg)

listInformation.see('end')

def SendReturnMessage(strType, nCode, strMessage,listInformation):

GetMessage(strType, nCode, strMessage,listInformation)

def GetMessage(strType,nCode,strMessage,listInformation):

strInfo = ""

if (nCode != 0):

strInfo ="【"+ skC.SKCenterLib\_GetLastLogInfo()+ "】"

WriteMessage("【" + strType + "】【" + strMessage + "】【" + skC.SKCenterLib\_GetReturnCodeMessage(nCode) + "】" + strInfo,listInformation)

以上程式碼的部分為初始化元件與主要相關功能宣告，未初始化是無法讓程式順利運作的，以上初始化的程式，不管在報價、下單或是回報都會用到，看不懂沒關係，別給自己太大壓力，[有完整程式](#報價)範例與環境設定，直接打開複製貼上或直接執行範例。

# *使用Python建立視窗介面*

大部分Python程式寫出來都是console畫面，為了要讓各位看官方便使用，我們開發了一個視窗介面的範例，讓操作更容易，顯示資料較直接等等。

以下僅列出部分UI介面設計相關的程式碼，方便大家參考。

#上半部登入框

class FrameLogin(Frame):

def \_\_init\_\_(self, master = None):

Frame.\_\_init\_\_(self, master)

self.grid()

#self.pack()

self.place()

self.FrameLogin = Frame(self)

self.master["background"] = "#ffecec"

self.FrameLogin.master["background"] = "#ffecec"

self.createWidgets()

def createWidgets(self):

#帳號

self.labelID = Label(self)

self.labelID["text"] = "帳號："

self.labelID["background"] = "#ffecec"

self.labelID["font"] = 20

self.labelID.grid(column=0,row=0)

#輸入框

self.textID = Entry(self)

self.textID["width"] = 50

self.textID.grid(column = 1, row = 0)

#密碼

self.labelPassword = Label(self)

self.labelPassword["text"] = "密碼："

self.labelPassword["background"] = "#ffecec"

self.labelPassword["font"] = 20

self.labelPassword.grid(column = 2, row = 0)

#輸入框

self.textPassword = Entry(self)

self.textPassword["width"] = 50

self.textPassword['show'] = '\*'

self.textPassword.grid(column = 3, row = 0)

#按鈕

self.buttonLogin = Button(self)

self.buttonLogin["text"] = "Login"

self.buttonLogin["background"] = "#ff9797"

self.buttonLogin["foreground"] = "#000000"

self.buttonLogin["highlightbackground"] = "#ff0000"

self.buttonLogin["font"] = 20

self.buttonLogin["command"] = self.buttonLogin\_Click

self.buttonLogin.grid(column = 4, row = 0)

#ID

self.labelID = Label(self)

self.labelID["text"] = "<<ID>>"

self.labelID["background"] = "#ffecec"

self.labelID["font"] = 20

self.labelID.grid(column = 5, row = 0)

#訊息欄

self.listInformation = Listbox(root, height=5)

self.listInformation.grid(column = 0, row = 1, sticky = E + W)

global GlobalListInformation,Global\_ID

GlobalListInformation = self.listInformation

Global\_ID = self.labelID

# 這裡是登入按鈕,使用群益API不管要幹嘛你都要先登入才行

def buttonLogin\_Click(self):

try:

m\_nCode = skC.SKCenterLib\_Login(self.textID.get().replace(' ',''),self.textPassword.get().replace(' ',''))

if(m\_nCode==0):

Global\_ID["text"] = self.textID.get().replace(' ','')

WriteMessage("登入成功",self.listInformation)

else:

WriteMessage(m\_nCode,self.listInformation)

except Exception as e:

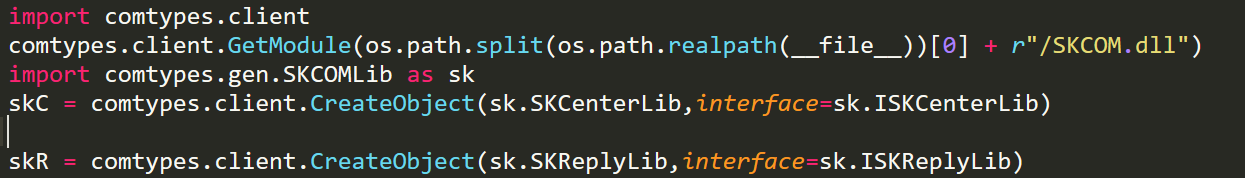
messagebox.showerror("error！",e)

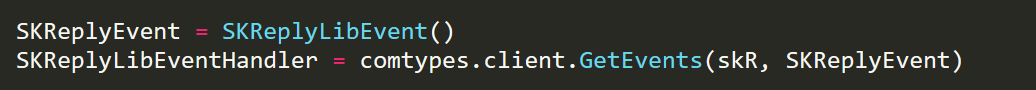
上面黃色框線部分，是很重要的登入功能，黃色字體部份是關鍵的部分－登入函式，其他都只是畫面顯示用的設定，看不懂沒關係，別給自己太大壓力，有完整程式範例與環境設定，直接打開複製貼上或直接執行範例。

## *註冊公告(必要項目)*

在登入SKCenterLib\_Login or SKCenterLib\_LoginSetQuote前，必須先註冊回報中的公告。

先建立SKReplyLib 物件及其事件SKReplyLib OnNotifyEvent

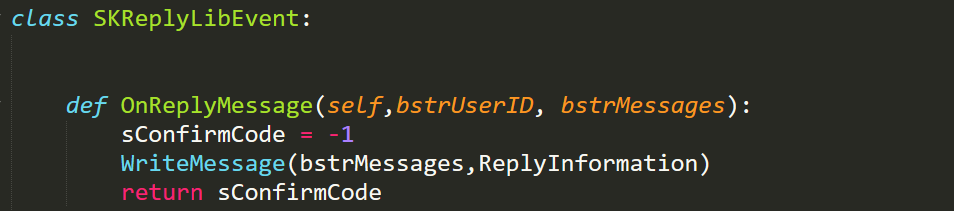




【策略王COM元件使用說明】4-3-e OnReplyMessage

▼下方是SKReplyLibEvent class中，處理公告通知OnReplyMessage函式 的部分程式碼，

並指定回傳-1



# *啟動報價*

取得報價資訊分為兩個步驟：

1. 呼叫一個叫SKQuoteLib\_EnterMonitor()的函式，這部分是連接群益報價主機，在範例程式大概是下面片段：

def btnConnect\_Click(self):

try:

m\_nCode = skQ.SKQuoteLib\_EnterMonitor()

SendReturnMessage("Quote", m\_nCode, "startQuote",GlobalListInformation)

except Exception as e:

messagebox.showerror("error！",e)

2. 接下來，想要什麼股票的報價，則必須透過SKQuoteLib\_RequestStocks函式送出商品的代號給API COM元件：

….

def btnQueryStocks\_Click(self):

try:

if(self.txtPageNo.get().replace(' ','') == ''):

pn = 0

else:

pn = int(self.txtPageNo.get())

skQ.SKQuoteLib\_RequestStocks(pn,self.txtStocks.get().replace(' ',''))

except Exception as e:

messagebox.showerror("error！",e)

….

由於在前端我們會提供一個輸入框讓你輸入一串(一個以上)股票代號self.txtStocks.get().replace(' ','') 代表是從輸入端取得的股票代碼，例如：2330，其中pn的作用說明如下，由於群益API可在同一時間提供非常多商品報價，非常非常多，為了有效處理這麼多檔商品的報價資料進而方便使用者分類顯示，類似頁簽的概念，在自己寫的程式裡可以將報價分別放在不同分頁；假設以產業分類，在第一頁想看食品類20檔報價，第二頁為光電類50檔，這個pn你可以填1跟2，但初學者可以填1。

ex: skQ.SKQuoteLib\_RequestStocks(1, "2330") # 這樣就完成跟群益要台積電報價的程序了

3. 註冊完報價呢? 已註冊的股票報價資料會從下面片段程式，進行處理：

……

def OnNotifyQuote(self, sMarketNo, sStockidx):

pStock = sk.SKSTOCK()

skQ.SKQuoteLib\_GetStockByIndex(sMarketNo, sStockidx, pStock)

strMsg = '代碼:',pStock.bstrStockNo,'--名稱:',pStock.bstrStockName,'--開盤價:',pStock.nOpen/math.pow(10,pStock.sDecimal),'--最高:',pStock.nHigh/math.pow(10,pStock.sDecimal),'--最低:',pStock.nLow/math.pow(10,pStock.sDecimal),'--成交價:',pStock.nClose/math.pow(10,pStock.sDecimal),'--總量:',pStock.nTQty

WriteMessage(strMsg,Gobal\_Quote\_ListInformation)

……

使用者可以在OnNotifyQuote()加入中斷點，確認報價資料是如何由指定的事件送進來，這就是個callback function的概念，callback又是什麼概念，這大概像是你想看報紙，於是你到報社去訂報紙，你當然可以在報社等當天的報紙下來，然後每天去報社拿，但這樣太累，何不告訴報社你家住址讓報社每天送報紙給你呢? 宣告一個callback function就是告訴報社你家住址在哪，報社一有報紙(報價)就丟給你，你只需要到信箱收報紙(報價)即可。

另外解釋一下，SKQuoteLib\_GetStockByIndex是拿來做什麼的，由於OnNotifyQuote()由群益主機送下來的資訊只有群益用的商品編號跟市場編號，這兩個編號用途是取得SKSTOCKS完整的內容，群益報價主機為各市場及其商品自訂的索引。因此註冊報價後，要如何知道自己註冊的股票完整資訊，必須透過這個索引編號SKQuoteLib\_GetStockByIndex才能真正拿到該檔股票的詳細資訊。

群益也同時提供了歷史行情讓使用者計算、回測等等，相關的功能可從範例程式碼取得，很簡單，也容易上手。

看不懂沒關係，別給自己太大壓力，有完整程式範例與環境設定，直接打開複製貼上或直接執行範例。

# *群益下單功能*

下單步驟比報價複雜一點，因為這是要去外面跟人家拚輸贏，所以要謹慎檢核送出的委託內容：

這邊是簡單介紹下單步驟，讓大家有印象。

首先介紹下單初始函式，下單需要先進行相關的設定：

#1.下單物件初始

def btnInitialize\_Click(self):

try:

m\_nCode = skO.SKOrderLib\_Initialize()

SendReturnMessage("Order", m\_nCode, "Order\_Initialize",GlobalListInformation)

except Exception as e:

messagebox.showerror("error！",e)

第二部分是檢查憑證，讀取憑證是為了讓使用者的交易更安全，確保本人交易：

#2.讀取憑證

def btnReadCert\_Click(self):

try:

m\_nCode = skO.ReadCertByID(Global\_ID["text"])

SendReturnMessage("Order", m\_nCode, "ReadCertByID",GlobalListInformation)

except Exception as e:

messagebox.showerror("error！",e)

第三部分是讀取下單帳號，你一定會記得你的身分證，但你可能不清楚你下單帳號是什麼，如果你開多個戶頭的話，則需要選擇交易帳號，下單時，這一定要代入的參數：

#3.取得下單帳號

def btnGetAccount\_Click(self):

try:

m\_nCode = SKOrderLib.GetUserAccount()

SendReturnMessage("Order", m\_nCode, "GetAccount",GlobalListInformation)

except Exception as e:

messagebox.showerror("error！",e)

呼叫GetUserAccount()之後的帳號從哪裡接收呢?這裡需要另一個callback function，來接收由群益主機傳送過來的帳戶資訊：

def OnAccount(self,bstrLogInID,bstrAccountData):

strValues = bstrAccountData.split(',')

strAccount = strValues[1] + strValues[3]

if strValues[0] == 'TS':

GlobalboxStockAccount['values'] = (strAccount)

上面這段code會接收你登入ID的所有帳號資訊，經過字串處理之後就可以拿來使用了。

第四部分就是下單-委託單內容相關的函式：

#4.下單送出

def btnSendStockOrder\_Click(self):

try:

if GlobalboxPrime.get()=="上市櫃":

sPrime=0

elif GlobalboxPrime.get()=="興櫃":

sPrime=1

if GlobalboxPeriod.get()=="盤中":

sPeriod=0

elif GlobalboxPeriod.get()=="盤後":

sPeriod=1

elif GlobalboxPeriod.get()=="零股":

sPeriod=2

if GlobalboxFlag.get()=="現股":

sFlag=0

elif GlobalboxFlag.get()=="融資":

sFlag=1

elif GlobalboxFlag.get()=="融券":

sFlag=2

elif GlobalboxFlag.get()=="無券":

sFlag=3

if GlobalboxBidAsk.get()=="買進":

sBuySell=0

elif GlobalboxBidAsk.get()=="賣出":

sBuySell=1

# 建立下單用的參數物件

oStock=sk.STOCKORDER()

# 填入完整帳號

oStock.bstrFullAccount = GlobalboxStockAccount.get()

# 填入股票代號

oStock.bstrStockNo = GlobaltxtStockNo.get()

# 上市、上櫃、興櫃

oStock.sPrime = sPrime

# 盤中、盤後、零股

oStock.sPeriod = sPeriod

# 現股、融資、融券

oStock.sFlag = sFlag

# 買賣別

oStock.sBuySell = sBuySell

# 委託價

oStock.bstrPrice = GlobaltxtPrice.get()

# 委託數量

oStock.nQty = int(GlobaltxtQty.get())

message,m\_nCode = skO.SendStockOrder(Global\_ID["text"], False, oStock)

SendReturnMessage("Order", m\_nCode, "GetUserAccount",GlobalListInformation)

except Exception as e:

messagebox.showerror("error！",e)

上面的程式碼，涵括了下單時要必須填入的所有參數，例如：股號，買賣別，現股還是融資、融券，盤中單、盤後單等等，黃色的部分是最後送出委託單的函式，執行完這行，代表委託單送到實際的股票市場進行條件的搓合；委託單送出前，需自行進行委託條件的檢查！看不懂沒關係，別給自己太大壓力，[[有完整程式範例](#報價)](#下單)與環境設定，直接打開複製貼上或直接執行範例。

# *回報功能介紹*

下單後，一定要確認回報，尤其在程式交易裡面，要確認下單指令到底有沒有正確的執行，下面簡短的介紹回報功能：

def btnConnect\_Click(self):

try:

m\_nCode = skR.SKReplyLib\_ConnectByID("F123456789")

except Exception as e:

messagebox.showerror("error！",e)

上面的程式碼，是為了建立一個ID:F123456789與群益回報主機的連線，透過群益主機，把回報發送到使用者本機端；回報連線成功後，回報資料可以從下面的通知函式接收，這個callback function是接收登入ID所有回報資料：

def OnNewData(self,strUserID,StrData):

WriteMessage(strUserID + " " +StrData ,ReplyInformation)

回報功能除了上述的連線及接收資料，還有後續回報資料的處理，可參考策略王COM元件使用說明文件裡的回報欄位介紹，依你的設計需求,進行字串的處理。

看不懂沒關係，別給自己太大壓力，[這裡備有整份程式碼](#回報)與環境設定，直接打開複製貼上或直接執行就會動。

***總結***

上面我們提到最重要的function以下做一個總整理：

SKCenterLib\_Login() #使用群益API不管報價、下單、回報一定都要先登入

SKQuoteLib\_EnterMonitor() #使用報價前要先呼叫的function

SKQuoteLib\_RequestStocks() #告訴群益API你想要什麼股票的報價

OnNotifyQuote() #SKQuoteLib\_RequestStocks()的callback function

SKQuoteLib\_GetStockByIndex() #取得商品的詳細資訊

SKOrderLib\_Initialize() #下單要先初始的函式

ReadCertByID() #讀取憑證

GetUserAccount() #讀取帳戶基本資料

OnAccount() #GetUserAccount()的callback function

SendStockOrder() #下單送單函式

SKReplyLib\_ConnectByID() #連接群益回報主機

OnNewData() #SKReplyLib\_ConnectByID() 的callback function

以上Function都可以在 策略王COM元件使用說明XXXX.docx 文件裡面，找到相關功能及詳細參數說明，文件附在群益官網下載及API套件壓縮檔裡。

**安裝套件及環境設定:**

透過Python開發，使用群益API COM需要安裝下面幾項Python套件（非群益提供）：

以下為建議使用版本，可以個人需求更新。

comtypes(1.1.4)

pip(9.0.1)

pytz(2018.3)

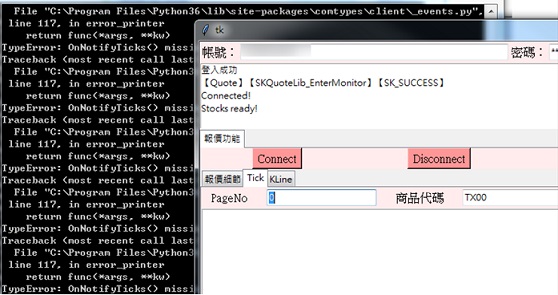
pywin32(222)

其中comtypes跟pywin32是一定需要的，另外使用的IDE是

Visual Studio Code。

## SKAPI換版注意事項

當使用comtypes時，換版後，部分功能應該會遇到以下問題，發生一直跳出錯誤訊息的情況，但程式卻又沒有錯誤！

※此例為OnNotifyTicks修改後發生錯誤▼

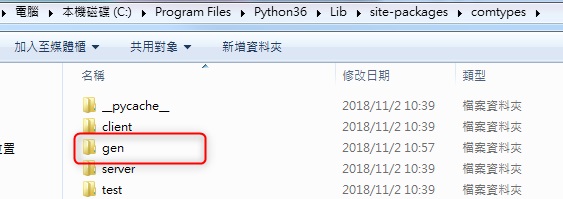
這是由於comtypes設計所致，當COM元件換版，若function或callback有修改的話，comtypes並不會同步更新相關檔案，python程式在執行的時候還是依據舊版本API的參數去處理，而無法正確對應API實際使用的參數，導致此問題發生！

因此我們需要將comtypes所產生的相關檔案作刪除動作，才能讓他重新產生新版COM元件所需的檔案。

1. 從錯誤訊息中我們可以確認comtypes實際路徑如下



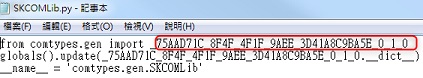
1. 使用檔案總管至該目錄下可看到gen資料夾

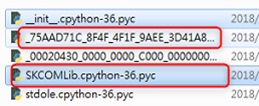
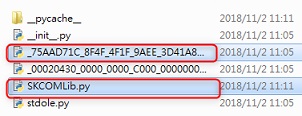


1. 刪除相關檔案，下方兩種方式擇一

3-1.若comtypes相關程式只有SKAPI的話，可以直接砍掉整個gen資料夾！

3-2.若要個別刪除，不是只有SKCOMLib.py要刪除，需要先查看該檔內容

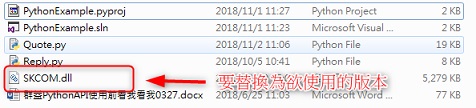


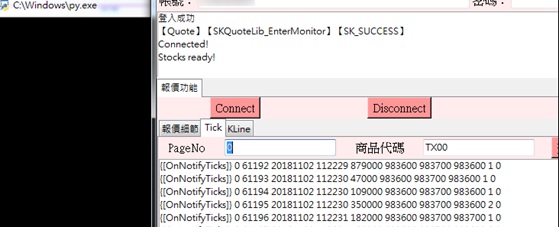
需要連這個GUID名稱的檔案也一起刪除掉，稍後重新產生才不會有問題

1. 產生相關檔案，請注意將此段code加入執行，這段code會產生comtypes相關檔案，產生後即可註解該行，因為產生檔案之後就不會再作用了。

comtypes.client.GetModule(os.path.split(os.path.realpath(\_\_file\_\_))[0] + r'\SKCOM.dll') #加此行需將API放與py同目錄

※注意python程式目錄也要放SKCOM.dll，這是要產生comtypes相關檔案所使用的。



1. 執行後，已可正常處理資料！

### 換版常見問題

1. 關於是註冊&執行同一SKCOM.dll  
   comtypes.client.getModule的SKCOM版本(這裡是設絕對路徑，跟install的版本無關)，跟實際install的SKCOM版本(實際執行function時是依註冊表設定執行對應版本)，兩者不一樣，出現問題，請檢查是否一致。

API討論區原文：

* 如果執行過舊版本的 getMedule(SKCOM.dll)，對應產生的py檔會存放在 **comtypes.client.gen\_dir** 的路徑上。
* comtypes.client.getModule會優先讀取comtypes.client.gen\_dir上已有應對的檔案，資料夾內沒有，才會用io生成新的。

若優先讀取用舊版COM產生的.py(暫存檔)會和註冊的新版SKCOM不一致，因為舊版.py檔在並沒有寫回傳sConfirmCode(-1)。

* 刪了舊py檔(路徑**comtypes.client.gen\_dir** )，會產生出新版SKCOM的py檔。
* 透過 **print(comtypes.client.gen\_dir)** 就可以找到py檔路徑。

如果是Visual studio的可供選安裝環境，就不會在python環境中的comtypes裡找到gen資料夾，因為是放在另外一個cache folder。

可在自行開發的py檔中，加入remove指定py能確保getModule總是從指定的dll生成出的py檔。(運行正常之後就可以拿掉remove程式片段)

remove 範例:

os.remove(comtypes.client.gen\_dir + r"\\_E7BCB8BB-E1F0-4F6F-A944-2679195E5807\_0\_1\_0.py")

# Python範例使用說明

# 範例目錄

* Python 範例[新增檔案說明](#python新增檔案說明)

1. 國內報價
   1. [Tick&Best5](#_Tick_&_Best5)
   2. [主機時間](#_主機時間)
   3. [上市上櫃市場資訊Market Info](#_上市上櫃市場資訊Market_Info)
   4. [Stock List 國內證期選商品檔](#_Stock_List_國內證期選商品檔)
2. 下單
   1. 證券
      1. [證券委託下單](#_證券委託下單)
      2. [證券委託減量](#_證券委託減量)
      3. [證券取消委託](#_證券取消委託)
      4. [證券即時庫存](#_證券即時庫存)
      5. [證券即時損益](#_證券即時庫存)
      6. [證券配額查詢](#_證券配額查詢)
      7. [證券集保庫存](#_證券集保庫存)

* 1. [期貨](#_1.1_期貨)
     1. [委託下單](#_1.1.1_委託下單)
     2. [委託減量](#_1.1.2_委託減量)
     3. [取消委託](#_1.1.3_取消委託)
     4. [國內期權未平倉查詢(含格式)](#_1.1.4_國內期權未平倉查詢(含格式))
     5. [權益數查詢](#_1.1.5_權益數查詢)
     6. [委託改價](#_1.1.6_委託改價)
  2. [選擇權](#_1.2_選擇權)
     1. [委託下單](#_1.2.1_委託下單)
     2. [複式單下單](#_1.2.2_複式單下單)
     3. [委託改價](#_1.2.3_選擇權改價)
  3. [海期](#_1.3_海期)
     1. [委託下單(含OLID委託、價差委託、價差OLID委託)](#_1.3.1_委託下單(含OLID委託、價差委託、價差OLID委託))
     2. [委託減量](#_1.3.2_委託減量)
     3. [取消委託](#_1.3.3_取消委託)
     4. [海外期權未平倉查詢](#_1.3.4_海外期權未平倉查詢)
     5. [海外期權權益數查詢](#_1.3.5_海外期權權益數查詢)
     6. [海外期貨商品清單查詢](#_1.3.6_海外期貨商品清單查詢)
     7. [委託改價](#_1.3.7_委託改價)
  4. [海選](#_1.4_海選)
     1. [委託下單](#_1.4.1_委託下單)
     2. [委託改價](#_1.4.2_委託改價)
     3. [海外選擇權商品清單查詢](#_1.4.3_海外選擇權商品清單查詢)
  5. 期選[智慧單](#_1.5_智動單-停損)
     1. [期貨停損委託](#_1.5.1_期貨停損委託)
     2. [期貨移動停損委託](#_1.5.2_期貨移動停損委託)
     3. [選擇權停損委託](#_1.5.3_選擇權停損委託)
     4. [期貨OCO委託](#_1.5.4_期貨OCO委託)
     5. [智慧單刪單](#_1.5.5_智動單刪單)
     6. [期貨MIT](#_期貨MIT委託)
     7. [選擇權MIT](#_選擇權MIT委託)
     8. [MIT刪單](#_MIT刪單)
     9. [智慧單查詢](#_1.5.6_智動單查詢_1)
  6. 證券智慧單
     1. 現股當沖組合
     2. [出清策略](#_1.6.1_出清策略)
     3. [證券智慧單查詢](#_1.6.2_證券智慧單查詢)
     4. [證券智慧單刪單](#_1.6.3_證券智慧單刪單)

1. 回報
   1. [一般回報](#_2.1_一般回報)
   2. [智慧單回報](#_2.2_智慧單回報)

新增檔案說明

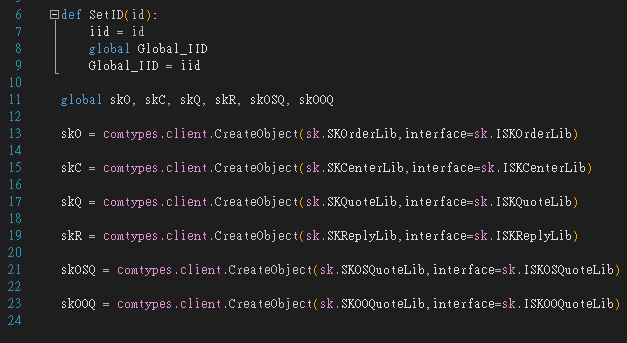
## Global.py

該檔案位於Order\_service 資料夾中

由於過去Order架構的關係，當建立多個SKOrderLib，會導致Callback未回傳。

因此SKOrderLib改為Global全域性變數的方式，避免重複建構，造成可能Callback事件的註冊會被覆蓋掉，產生部分function之對應Callback未回傳；SetID的部分，一樣使用Global呼叫的方式。

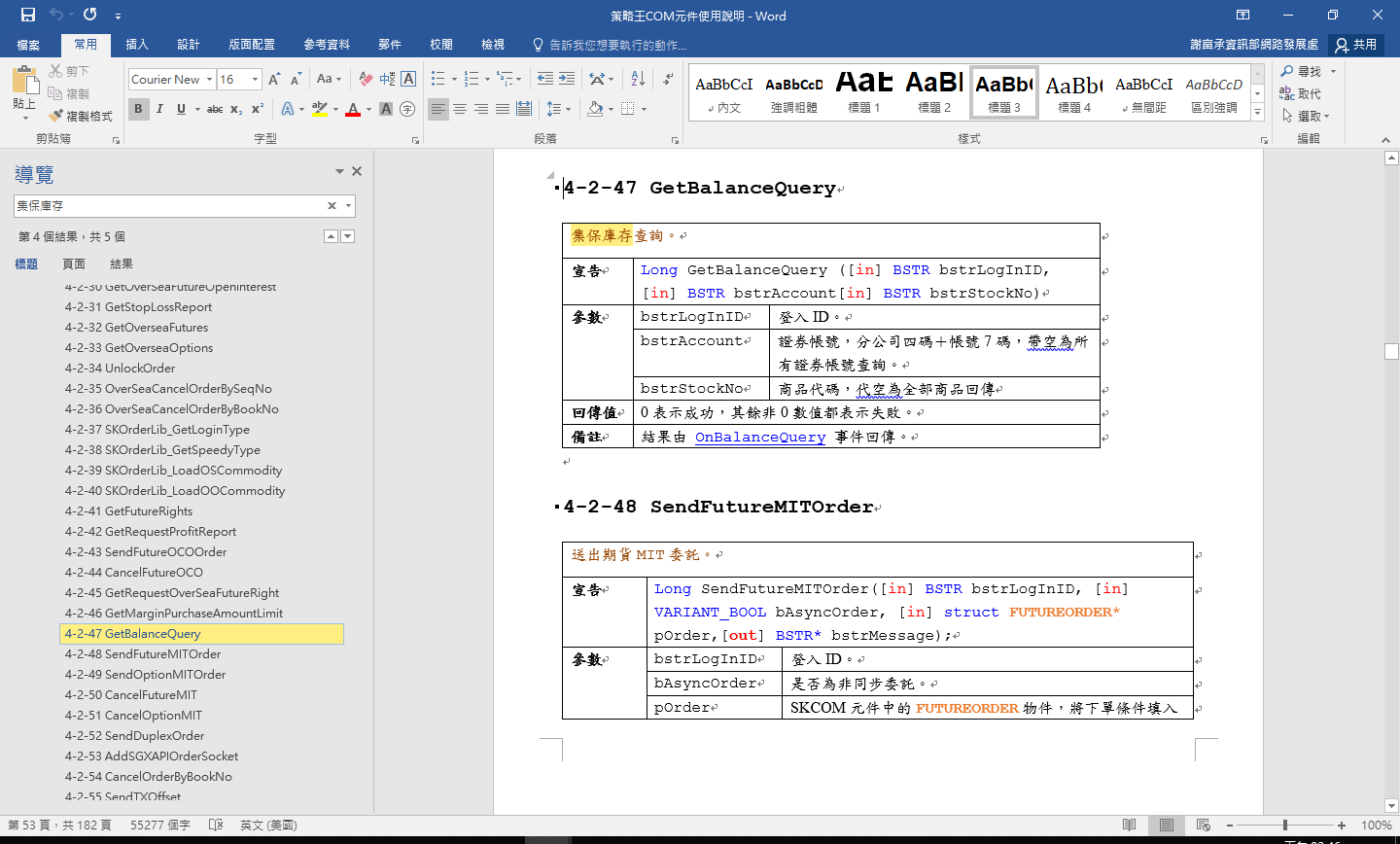
下方為部分程式碼



## OnNotifyEvent(callback) 範例

照以下步驟進行即可，使用 **查詢集保庫存做為例子介紹。**

1. 先參考《策略王COM元件使用說明》



此功能必要的參數為：bsrtLogInID, bstrAccount, bstrStockNo三個：

其中GetBalanceQuery則是我們需要使用到skO.GetBalanceQuery。

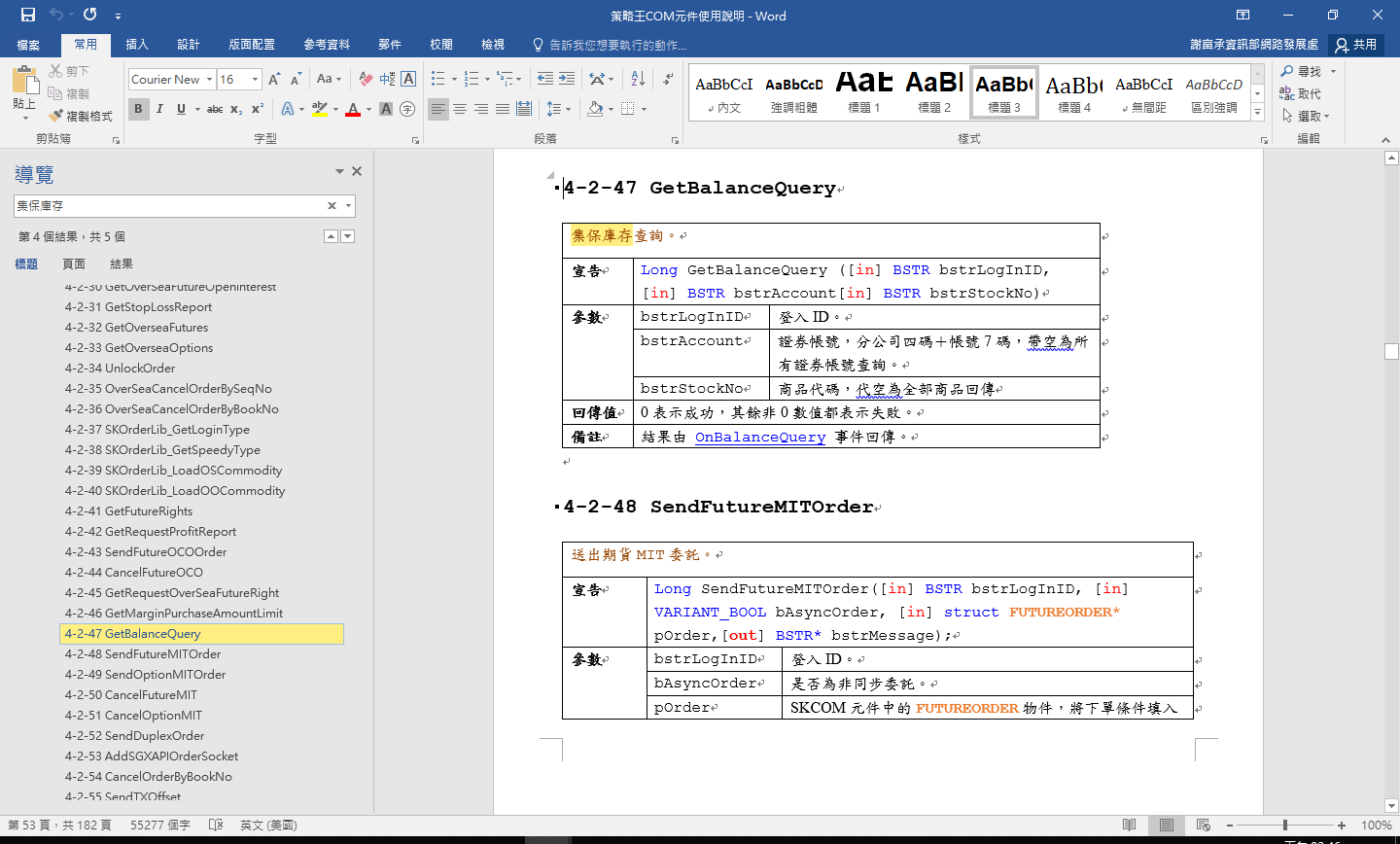
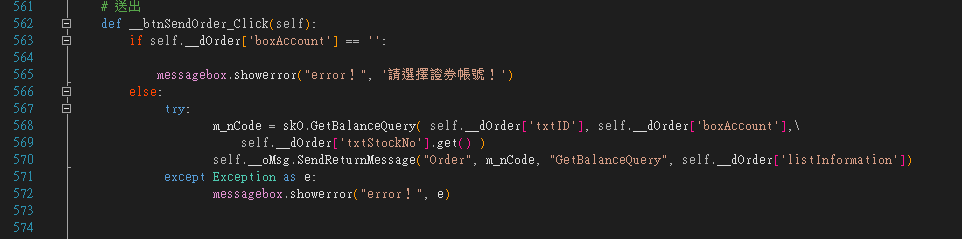
1. 照著上方，可以寫出以下python函式

m\_nCode = skO.GetBalanceQuery( self.\_\_dOrder['txtID'], self.\_\_dOrder['boxAccount'],\

self.\_\_dOrder['txtStockNo'].get() )

self.\_\_oMsg.SendReturnMessage("Order", m\_nCode, "GetBalanceQuery", self.\_\_dOrder['listInformation'])

下方紅框線，代表會執行函式sko.GetBalaceQuery，同時會觸發其對應callback事件：OnBalanceQuery

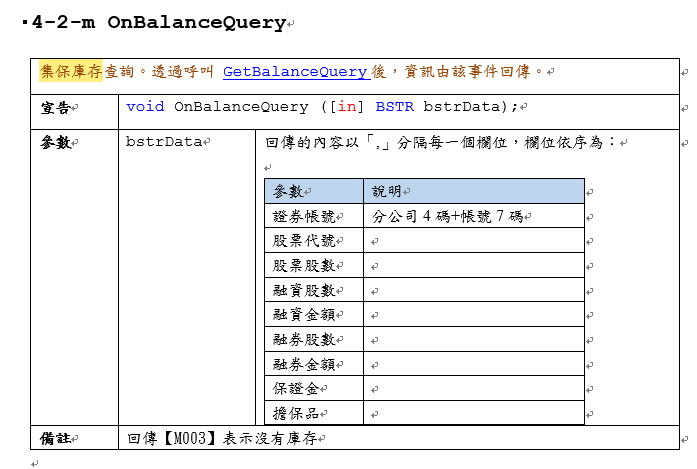


1. 由GetBalanceQuery函式觸發事件callbackOnBalanceQuery，將結果回傳

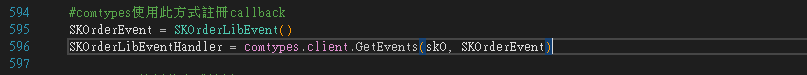
因此要定義一下callback事件，以解析回傳內容。

查詢OnBalanceQuery的回傳格式及內容：可在上圖（紅框）藍字OnBalanceQuery上 按住Ctrl+點擊滑鼠左鍵 或是直接查詢 OnBalanceQuery

回傳內容如下圖：



1. 將callback OnBalanceQuery新增至 class SKOrderLibEvent()，並參考下方comtypes 套件註冊SKOrderLibEvent。



## Function v.s OnNotifyEvent (callback)

1. function函式
2. OnNotifyEvent函式的通知事件
3. 註冊OnNotifyEvent(callback)

1.函式(螢光塗色部分)

m\_nCode = skO.GetBalanceQuery( self.\_\_dOrder['txtID'], self.\_\_dOrder['boxAccount'],\

self.\_\_dOrder['txtStockNo'].get() )

self.\_\_oMsg.SendReturnMessage("Order", m\_nCode, "GetBalanceQuery", self.\_\_dOrder['listInformation'])

2.事件

class SKOrderLibEvent():

\_\_account\_list = dict(

stock = [],

future = [],

sea\_future = [],

foreign\_stock = [],

)

def \_\_init\_\_(self):

self.\_\_obj = dict(

msg = MessageControl.MessageControl(),

)

def OnBalanceQuery(self, bstrData):

self.\_\_obj["msg"].WriteMessage(bstrData,GlobalListInformation)

3.

#comtypes使用此方式註冊callback

SKOrderEvent = SKOrderLibEvent()

SKOrderLibEventHandler = comtypes.client.GetEvents(skO, SKOrderEvent)

\*記得**先註冊COM元件**，詳細資訊可以看COM元件說明文件。

## 同步與非同步委託

委託時，需決定使用同步或非同步進行送單：

如下:  
message, m\_nCode = skO.SendFutureOrder(Global.Global\_IID, bAsyncOrder, oOrder)

bAsyncOrder 是決定同步或非同步的參數。

如果是使用非同步的委託，請由 PythonExample/order\_service/Order.py 中OnAsyncOrder 取得委託結果。

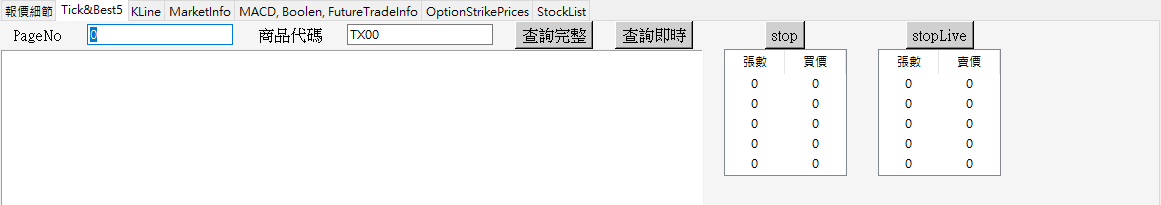
class SKOrderLibEvent()   
 def OnAsyncOrder(self,nThreadID,nCode,bstrMessage):

1. 國內報價

以下部分程式碼，可參考 Quote.py

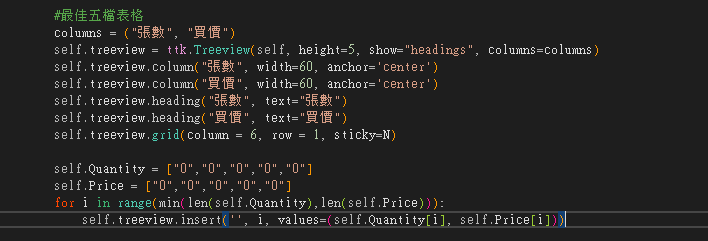
## Tick & Best5

## 最佳五檔

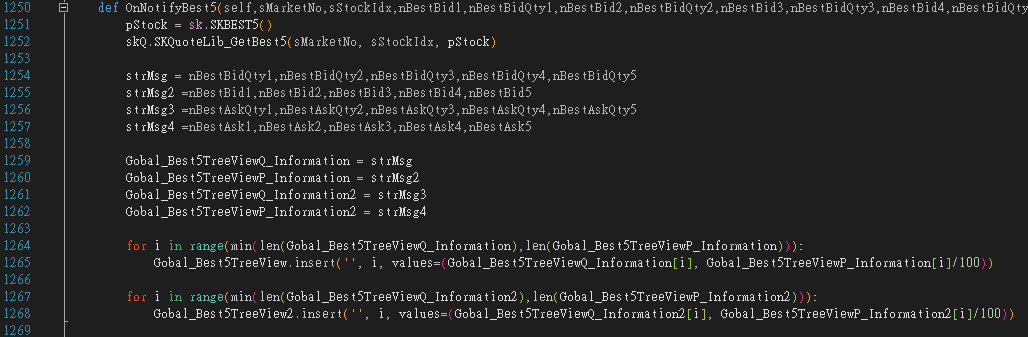


輸入商品代碼，點按查詢完整，即完成訂閱最佳五檔。

下方為部分程式碼，UI元件－表格部分採用treeview package



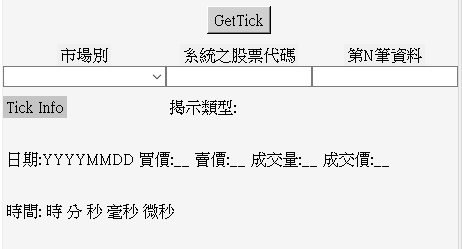
通知事件－OnNotifyBest5，回傳最佳五檔價格：



Best5 格式內容可以參照 《策略王COM元件使用說明》。

## GetTick

（已訂閱Tick，取指定一筆Tick資料）

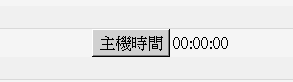


輸入選擇市場別、商品代碼以及指定第N筆Tick，即可獲得詳細指定Tick 資訊。

▼部分程式碼：

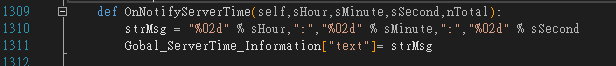


## 主機時間

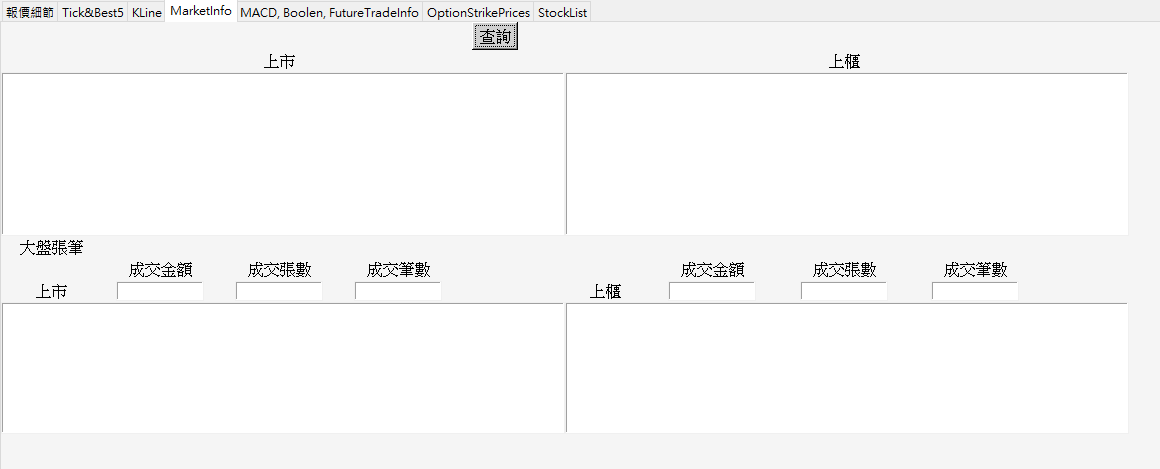


報價連線成功後，即會回傳主機時間，為 solace 行情主機時間

▼部分程式碼



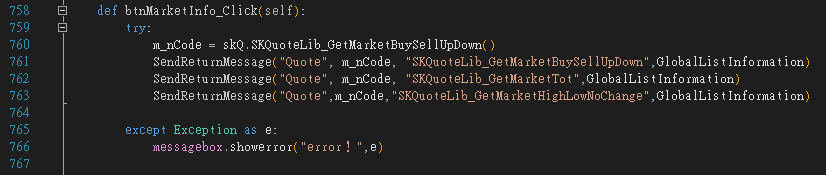
## 上市上櫃市場資訊Market Info



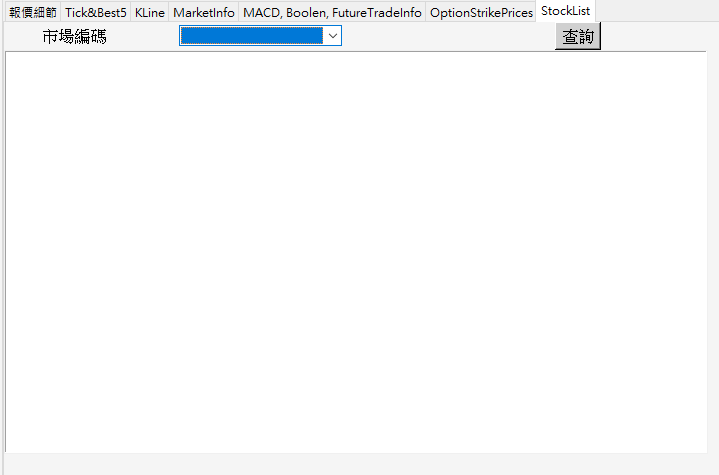
查詢後，回傳上市上櫃資訊：

成交(上漲/下跌)家數、大盤成交買進(張/筆)數、成交金額、成交(張/筆)數等資訊

▼部分程式碼

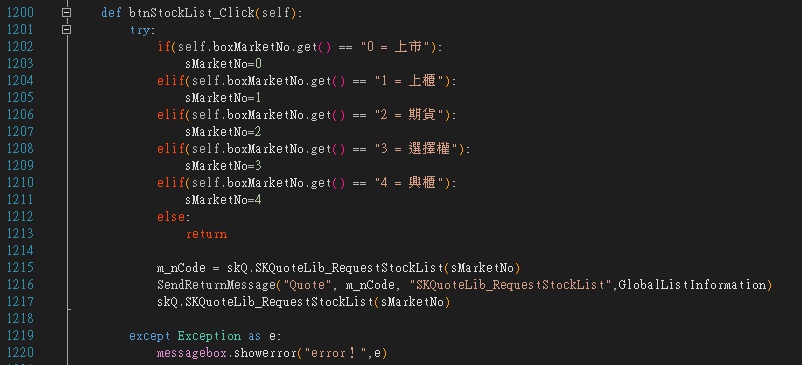


## Stock List 國內證期選商品檔



報價連線成功後，選取市場代碼再進行查詢，回傳該市場商品清單。

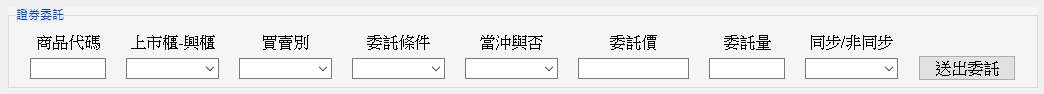
▼為部分程式碼



1. 下單
   1. 證券

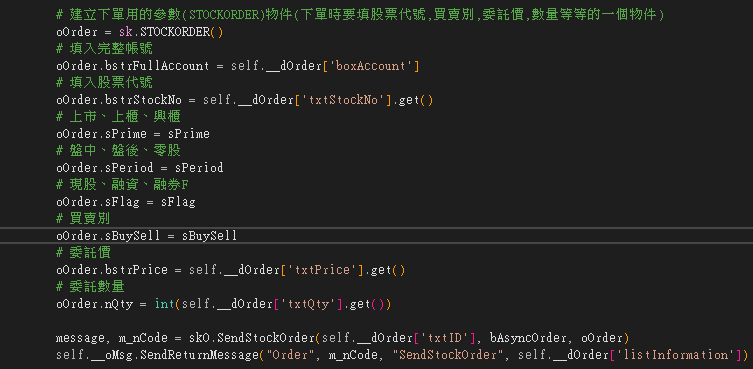
程式碼可參考 StockOrder.py

## 證券委託下單



依證券委託(struct STOCKORDER) ，將商品代碼(Ex. 2330)、上市櫃、買賣別、委託條件等相關資訊填入STOCKORDER後，送出委託，可同時收到委託結果。

▼範例部分程式碼

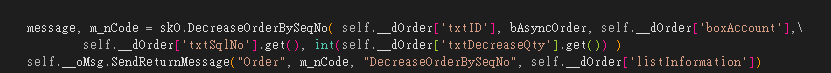


## 證券委託減量



填入委託序號(13碼數字)以及欲減少的數量，點選同步/非同步委託，送出後再確認委託結果。

▼範例部分程式碼



## 證券取消委託



取消委託有三種方式：

商品代碼：如：2330、2337等

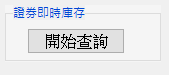
委託序號：13碼數字

委託書號：1英文+ 4數字

▼範例部分程式碼

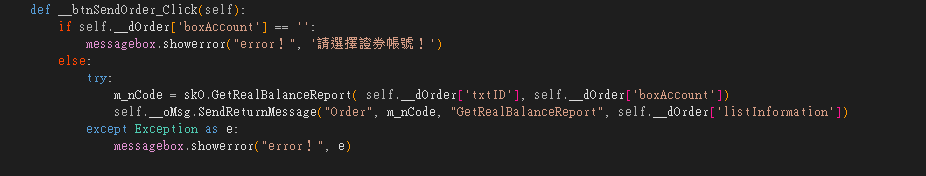


## 證券即時庫存

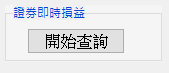


點開始查詢，即可回傳即時庫存。

▼範例部分程式碼

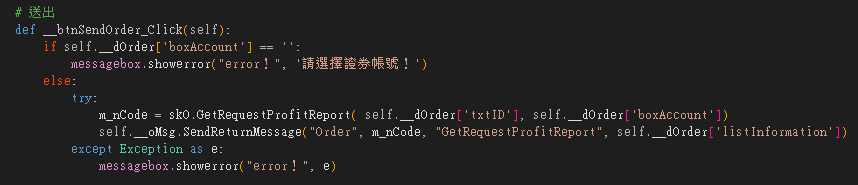


## 證券即時損益

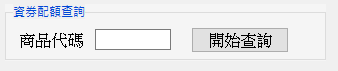


點開始查詢，即可回傳證券即時損益。

▼範例部分程式碼

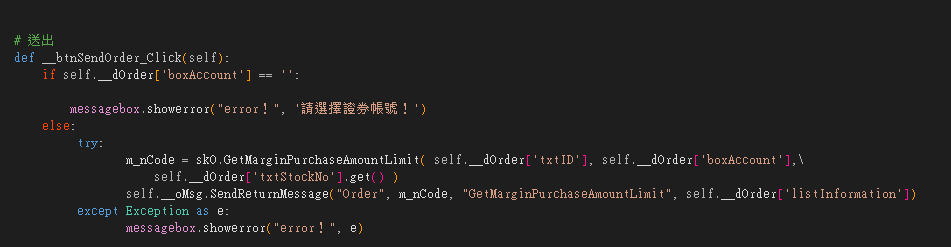


## 證券配額查詢

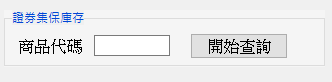


輸入商品代碼 點按開始查詢 即可查詢資券配額

▼範例部分程式碼

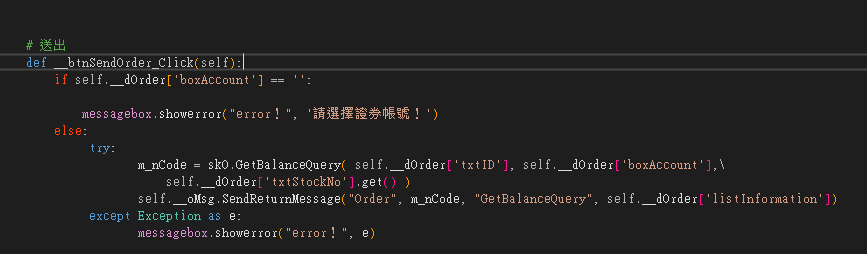


## 證券集保庫存



輸入商品代碼，即回傳證券集保庫存。

▼範例部分程式碼



註2：非同步部分由OnAsyncOrder通知事件回傳委託結果。

該事件程式碼位於Order.py中 class SKOrderLibEvent( ) 內

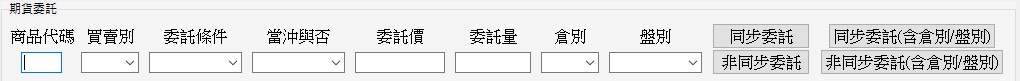
詳細操作說明可以參照 《策略王COM元件說明》



* 1. 期貨

程式碼可參考 PythonExample/order\_service/FutureOrder.py

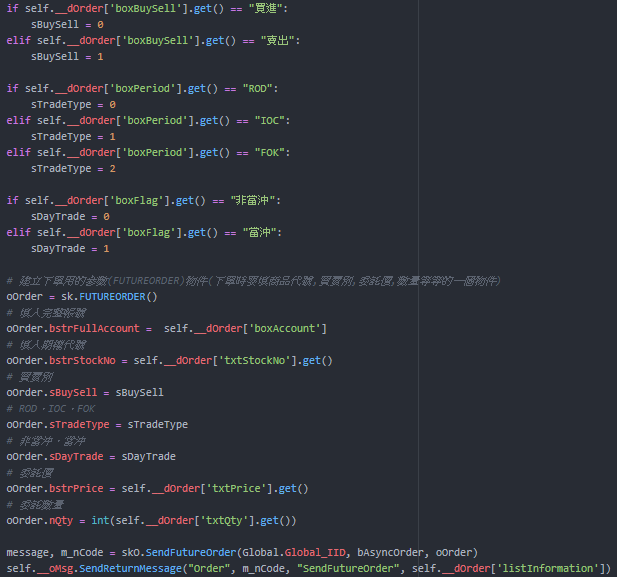
## 期貨委託下單



必要填入：商品代碼、買賣別、委託條件、當沖、委託價、委託量

如果是含倉別/盤別的委託，要全部填入。盤別分為 T盤及T+1盤。

▼下單範例的部分程式碼：



## 期貨委託減量

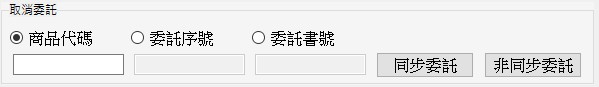


填入委託序號(13碼數字)及減少的口數量後，選擇同步或非同步委託即可。

▼委託減量範例的部分程式碼



## 期貨取消委託

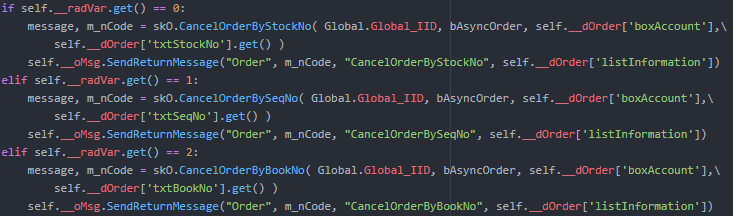


取消委託共有三種取消方法：  
1.依商品代碼取消

2.依委託序號取消(通常13碼數字)

3.依委託書號取消(通常是1個英文+4個數字)

▼取消委託範例的部分程式碼



## 國內期權未平倉查詢(含格式)

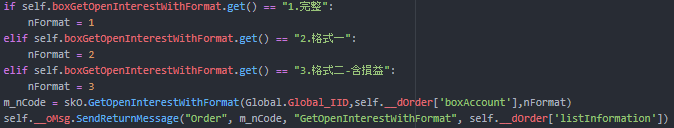


可以直接查詢，或者是選擇格式後，在進行查詢。

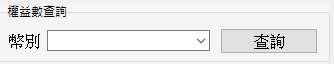
格式分三種 1.完整(含複式單) 2.格式一(含複式單) 3.格式二(含損益-不含複式單)

直接查詢與格式二的內容相同。若是要查詢複式單部位，需使用完整或是格式一。

▼國內期權未平倉查詢(含格式) 範例的部分程式碼

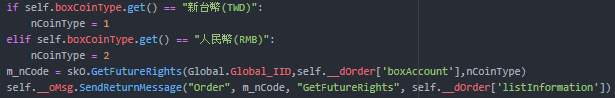


## 國內期貨權益數查詢



查詢指定期貨帳號的權益數，僅查詢國內期貨。幣別：新台幣跟人民幣。

▼權益數查詢範例的部分程式碼

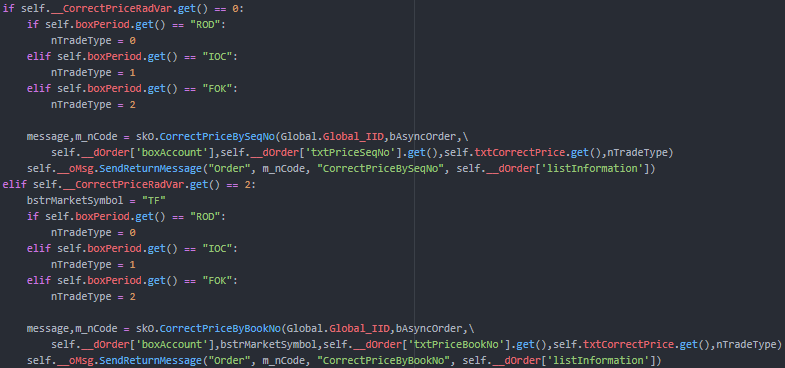


## 期貨改價



委託改價二種方式：依委託序號或委託書號

▼委託改價範例的部分程式碼。



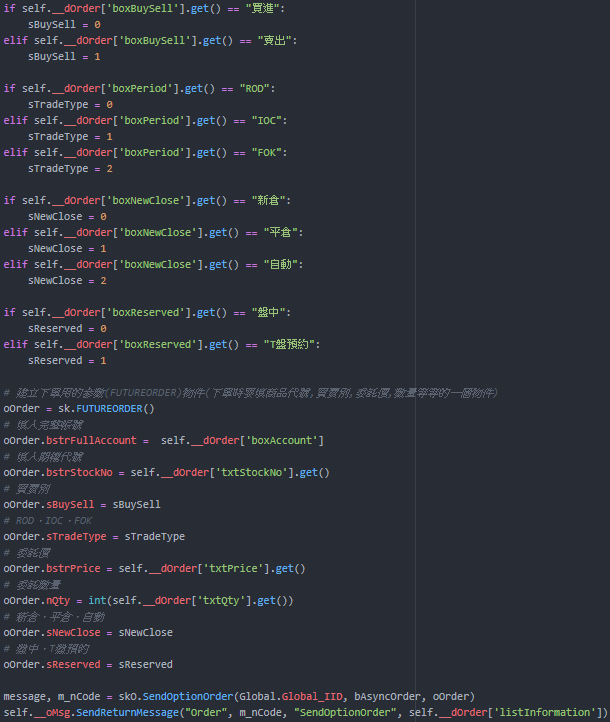
* 1. 選擇權

程式碼可參考 PythonExample/order\_service/OptionOrder.py

## 選擇權下單



選擇權的 委託下單FUTUREORDER結構物件都必須要填入。  
商品代碼 可經由策略王得知。  
▼委託下單範例的部分程式碼



## 複式單下單



複式單下單 需要填入兩組商品代碼。  
下單的策略請參照【策略王COM元件使用說明】 4-2-52 SendDuplexOrder

▼複式單下單範例 的部分程式碼



## 選擇權改價



選擇權改價兩種方式： 委託序號(13碼數字)或委託書號(1碼英文+4碼數字)。

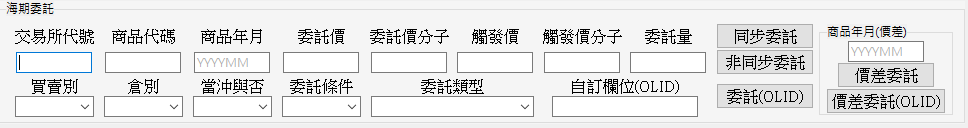
▼選擇權改價範例的部分程式碼



* 1. 海期

程式碼請參考 PythonExample/order\_service/SeaFutureOrder.py

## 海期委託下單(含OLID委託、價差委託、價差OLID委託



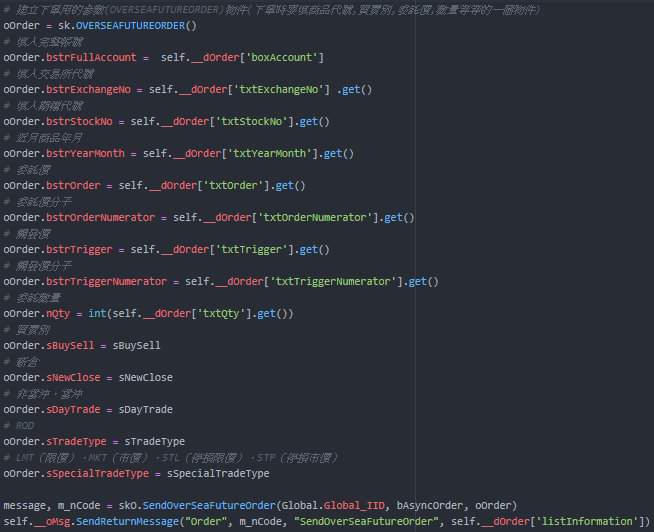
**請先下載海期、海選商品，再進行海期、海選下單!**

下單時，交易所代號、商品代碼、商品年月(西元，Ex:201912)、委託價、委託價分子、觸發價、觸發價分子、委託量、買賣別、倉別、當沖、委託條件、委託類型，都要必須填入，不然會無法下單。

交易所代號、商品代碼、商品年月(西元，Ex:201912)，可以透過 海外期貨商品清單查詢。

* 使用OLID進行委託時，則要在自訂欄位中填入。
* 使用 價差委託 時，在 商品年月(價差) 的地方需要填入。
* 使用 價差委託(OLID) 時，則要在 商品年月(價差)和自訂欄位 中填入。

▼委託下單 的部分程式碼



## **海期委託減量**

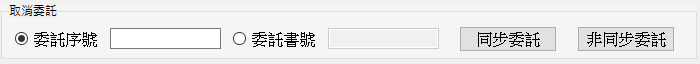


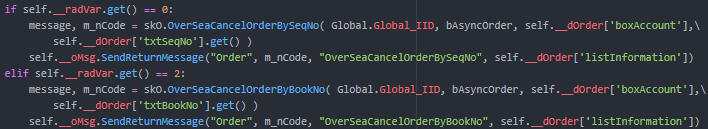
填入 委託序號(13碼數字)和減少的數量後，選擇同步或非同步委託即可。

▼委託減量 的部分程式碼



## 海期取消委託



填入 委託序號(13碼數字)或委託書號(1碼英文+4碼數字) 進行取消的委託。可以依序號或書號，取消海期或海選的委託單。  
▼取消委託 的部分程式碼  


## 海外期貨選擇權未平倉查詢



海外期貨選擇權－未平倉查詢  
▼海外期權未平倉查詢 的部分程式碼  


## 海外期貨選擇權－權益數查詢



海外期權權益數查詢  
▼海外期權權益數查詢 的部分程式碼



## 海外期貨商品清單查詢

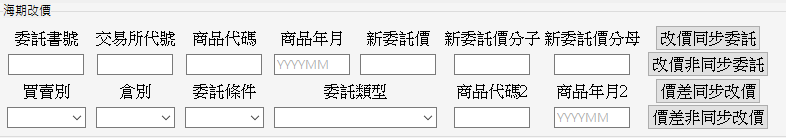


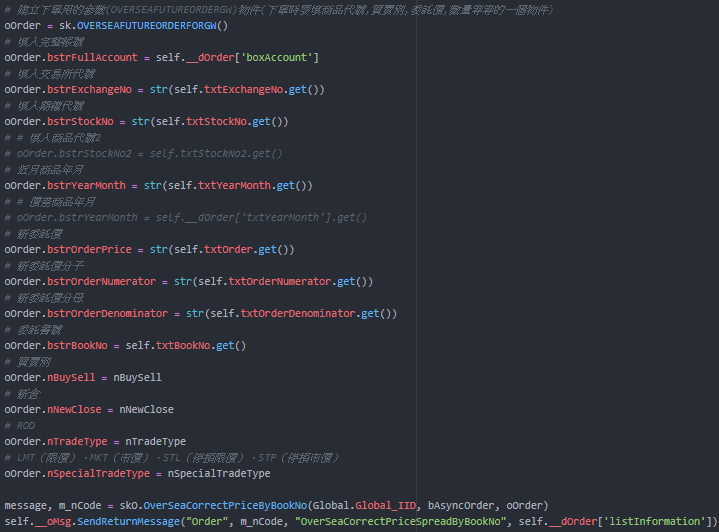
請先下載海期商品，再進行海期商品查詢！

使用此功能，可以得到目前海期可交易商品的清單。

▼海外期貨商品清單查詢 的部分程式碼  


## 海期委託改價

****

海期的委託改價，僅能用 委託書號 做委託，一般委託需要填入委託單的 交易所代號、商品代碼、商品年月、買賣別、倉別、委託條件、委託類型。需要另外填入的為 新委託價、新委託價分子、新委託價分母。  
價差改價 則是需要另外填入 商品代碼2和商品年月2。  
▼委託改價 的部分程式碼  
****

* 1. 海選

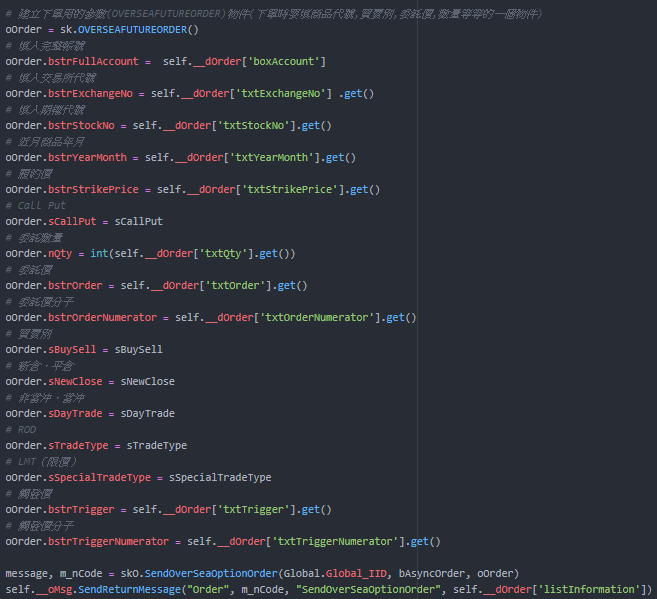
程式碼在 PythonExample/order\_service/SeaOptionOrder.py

## 海選委託下單

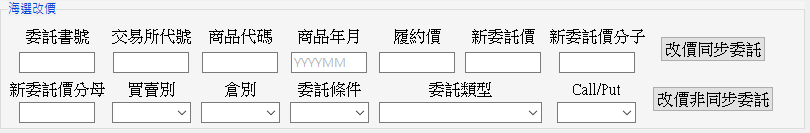


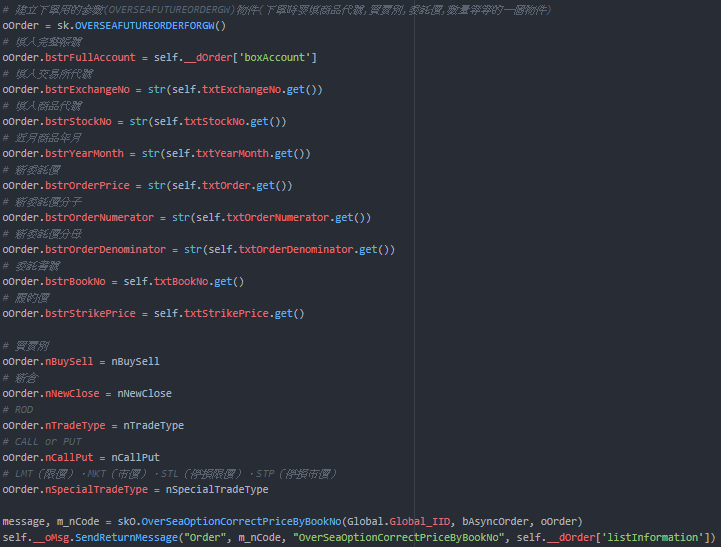
請先下載海期、海選商品，再進行海期、海選下單!

海選的委託下單要將上列的條件都填入，才能進行委託。  
交易所代號、商品代碼、商品年月 可經由 海外選擇權商品清單查詢 得知。

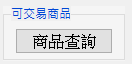
▼委託下單的部分程式碼  


## 海選委託改價



海選的委託改價，僅能用 委託書號 來進行，上列中除新委託價、新委託價分子、新委託價分母外，皆應與使用者填入的委託書號內容相同。  
下方是 委託改價 的部分程式碼  


## 海外選擇權商品查詢



請先下載海選商品，再進行可交易商品查詢!

使用此功能，可以得到目前海選可交易商品的清單。

下方是 海外選擇權商品清單查詢 的部分程式碼  
****

* 1. 期選智慧單

程式碼可參考 PythonExample/order\_service/StopLossOrderGui.py and MITOrder.py

## 期貨停損委託



務必簽屬期貨智慧單風險預告書，再進行下期選智慧單功能!

尚未簽屬用戶可至 [金融網－同意書簽署專區](https://wtrade.capital.com.tw/TSWEB/TradeAgree/agreeList.asp) 申請

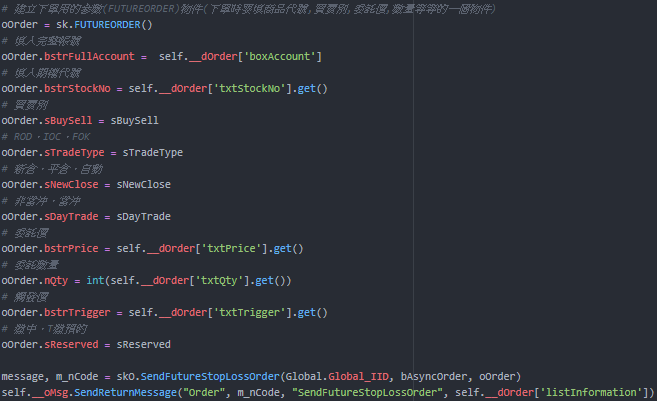
若未簽屬而使用期選智慧單，會出現以下訊息

【SK\_WARNING\_SIGN\_TF\_SMARTTRADE\_RICK\_NOITICE\_FIRST】

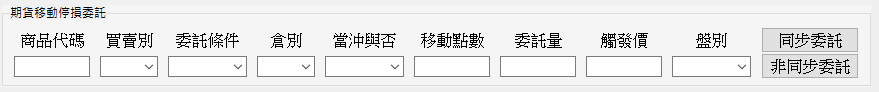
期貨智慧單委託，上方所列的結構物件(可參考元件說明-struct FUTUREORDER)需要全部填入。

~~與一般委託的差別，在於觸發價。~~

▼期貨停損委託 的部分程式碼



## 期貨移動停損委託



下智慧單前，務必簽屬期貨智慧單風險預告書!

尚未簽屬用戶可至 [金融網－同意書簽署專區](https://wtrade.capital.com.tw/TSWEB/TradeAgree/agreeList.asp) 申請。

若未簽屬而使用智慧單，會出現以下訊息  
【SK\_WARNING\_SIGN\_TF\_SMARTTRADE\_RICK\_NOITICE\_FIRST】

期貨移動停損，需要將上列的結構物件都填入，才能完成下單。

▼期貨移動停損委託 的部分程式碼  


## 選擇權停損委託



下智慧單前，務必簽屬期貨智慧單風險預告書!

尚未簽屬用戶可至 [金融網－同意書簽署專區](https://wtrade.capital.com.tw/TSWEB/TradeAgree/agreeList.asp) 申請。

若未簽屬而使用智慧單，會出現以下訊息  
【SK\_WARNING\_SIGN\_TF\_SMARTTRADE\_RICK\_NOITICE\_FIRST】

選擇權停損委託，必須將上列結構物件都填入後，才能進行委託。

▼選擇權停損委託 的部分程式碼  


## 期貨OCO委託

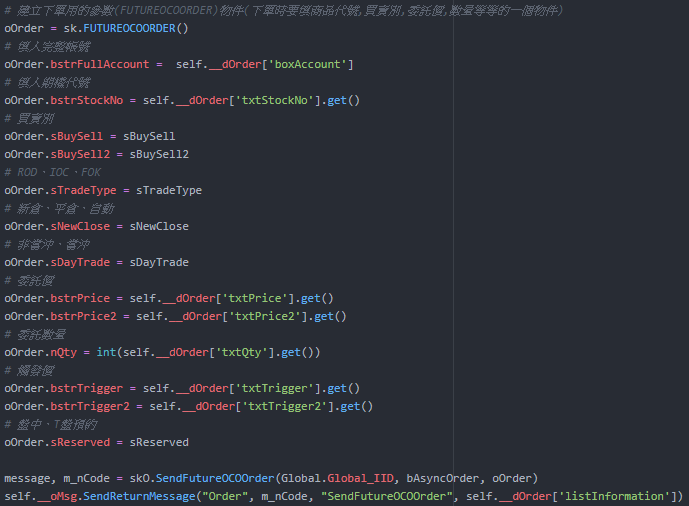


下智慧單前，務必簽屬期貨智慧單風險預告書!

尚未簽屬用戶可至 [金融網－同意書簽署專區](https://wtrade.capital.com.tw/TSWEB/TradeAgree/agreeList.asp) 申請。

若未簽屬而使用智慧單，會出現以下訊息  
【SK\_WARNING\_SIGN\_TF\_SMARTTRADE\_RICK\_NOITICE\_FIRST】

OCO委託 為 二擇一委託。

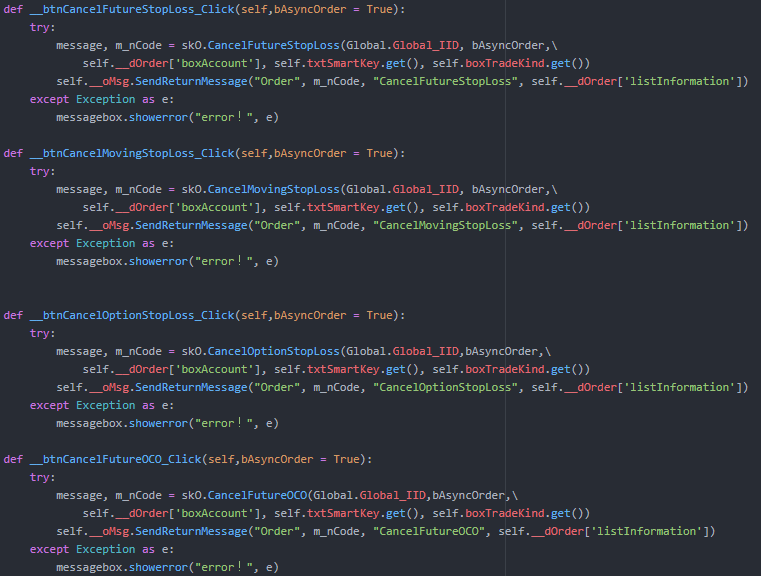
▼期貨OCO委託 的部分程式碼  


## 智慧單刪單 (STP/MST/OCO)



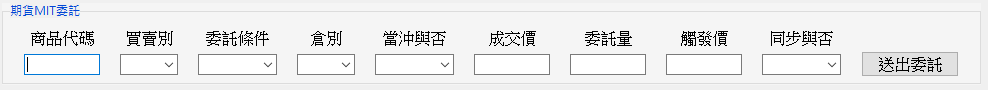
智慧單序號 通常為五碼數字。  
類型分別為：STP(期貨停損委託、選擇權停損委託)、MST(期貨移動停損委託)、OCO(期貨OCO委託)、 MIT(期貨MIT、選擇權MIT)

~~類型要配對正確，才能有效進行刪單動作!!!~~

▼智慧單刪單 的部分程式碼  


▼部分程式碼 皆可在 SendMITOrder.py中查詢完整代碼

## 期貨MIT委託



務必簽屬期貨智慧單風險預告書，再進行下期選智慧單功能!

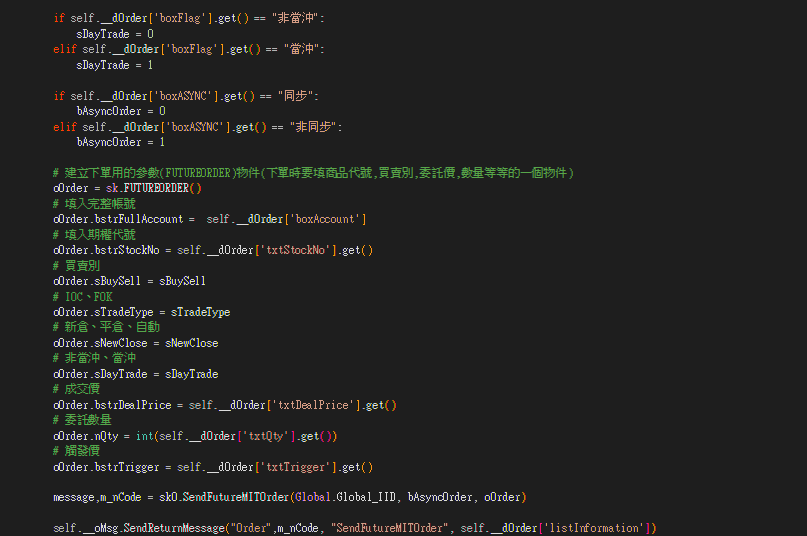
尚未簽屬用戶可至 [金融網－同意書簽署專區](https://wtrade.capital.com.tw/TSWEB/TradeAgree/agreeList.asp) 申請

若未簽屬而使用期選智慧單，會出現以下訊息

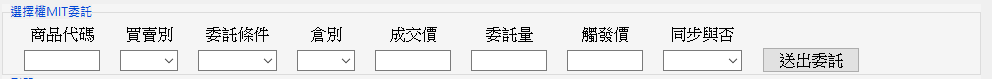
【SK\_WARNING\_SIGN\_TF\_SMARTTRADE\_RICK\_NOITICE\_FIRST】

期貨MIT智慧單委託，上方所列的結構物件(可參考元件說明-struct FUTUREORDER)需要全部填入。

▼期貨MIT部分程式碼



## 選擇權MIT委託



務必簽屬期貨智慧單風險預告書，再進行下期選智慧單功能!

尚未簽屬用戶可至 [金融網－同意書簽署專區](https://wtrade.capital.com.tw/TSWEB/TradeAgree/agreeList.asp) 申請

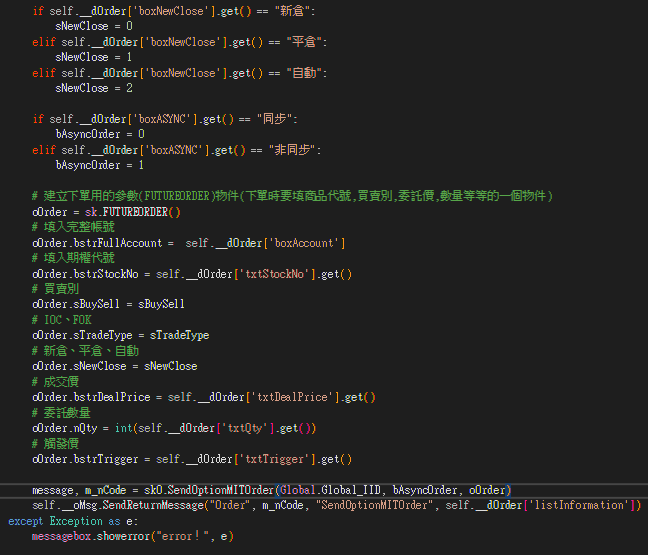
若未簽屬而使用期選智慧單，會出現以下訊息

【SK\_WARNING\_SIGN\_TF\_SMARTTRADE\_RICK\_NOITICE\_FIRST】

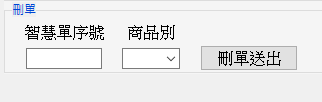
選擇權MIT智慧單委託，上方所列的結構物件(可參考元件說明-struct FUTUREORDER)需要全部填入。

\*關鍵的變動部分在於由sko.SendFutureMITOrder 變為 sko.SendOptionMITOrder

▼選擇權MIT部分程式碼



## MIT刪單

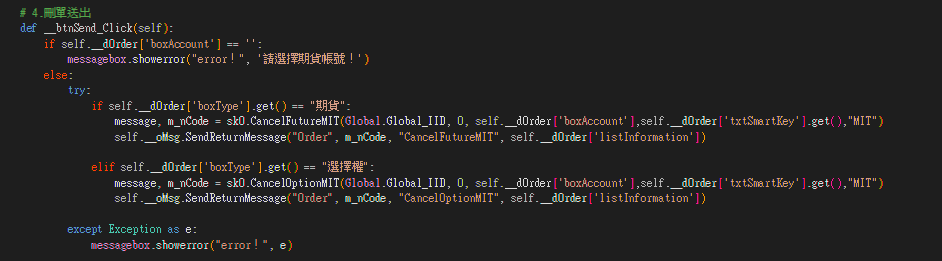


輸入智慧單序號(通常為五碼數字)，選擇商品別(期貨、選擇權) 進行刪單

智慧單序號 通常為五碼數字。  
類型分別為：STP(期貨停損委託、選擇權停損委託)、MST(期貨移動停損委託)、OCO(期貨OCO委託)、 MIT(期貨MIT、選擇權MIT)

**\*此功能預設送出刪單類別為 MIT**

▼MIT刪單部分程式碼

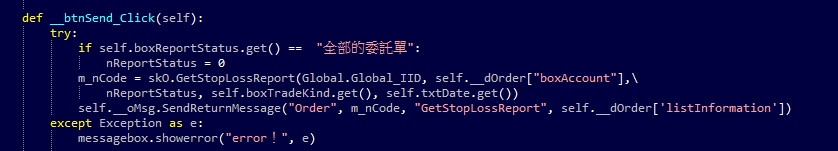


## 智慧單查詢

****

**類別只有一項可以選擇，就是全部的委託單。智慧單類型則依照自己想要查詢的類型選擇。日期則要輸入8碼 Ex: 20190101**

▼期貨及選擇權智慧單查詢 的部分程式碼



* 1. 證券智慧單

程式碼可參考 PythonExample/order\_service/StockSmartTrade.py

## 現股當沖組合



現股當沖組合的四個必填結構物件，商品代碼、買賣別、委託價、委託量。

停利、停損、出清、盤後定盤交易。

▼現股當沖組合 的部分程式碼

|  |  |
| --- | --- |
|  |  |
|  | |

## 出清策略



出清策略必填的四個結構物件分別為，商品代碼、買賣別、現資券、委託量。

買賣別對應使用者手中的庫存。現資券則是該股庫存的種類。

▼出清策略 的部分程式碼

|  |  |
| --- | --- |
|  |  |
|  | |

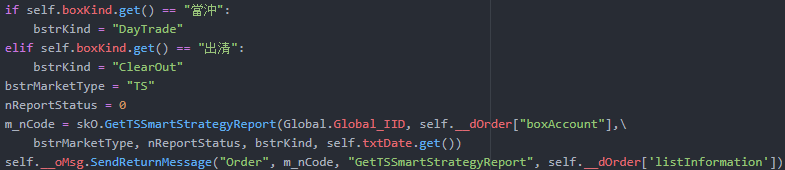
## 證券智慧單查詢



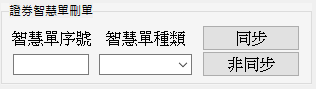
證券智慧單查詢，種類分為當沖單和出清單。

日期的部分，請輸入8碼數字。Ex: 20190101

▼證券智慧單查詢 的部分程式碼

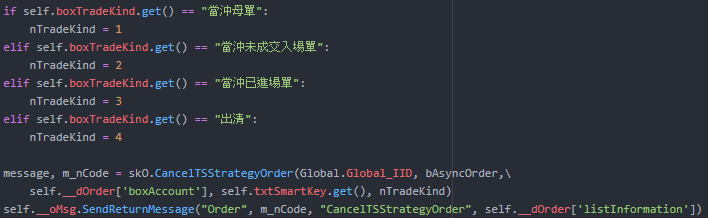
****

## 證券智慧單刪單



智慧單序號，通常為5碼數字。

智慧單種類分為:1:當沖母單、2:當沖未成交入場單、3:當沖已進場單、4:出清

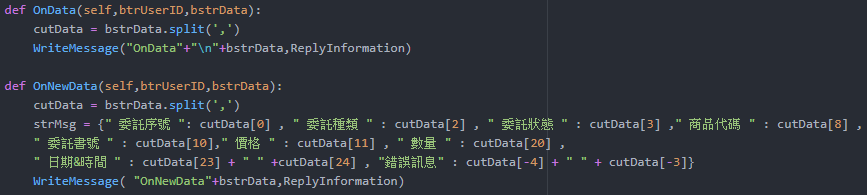
▼ 證券智慧單刪單 的部分程式碼  


1. 回報

程式碼可參考PythonExample/Reply\_Service/Reply.py

* 1. 一般回報

兩個回報的方式皆相同，每一筆資料以「,」分隔每一個欄位。每個欄位的詳細資料，請查看   
【策略王COM元件使用說明】4-3-d OnData & 4-3-g OnNewData。

▼ 一般回報 的部分程式碼  


* 1. 智慧單回報

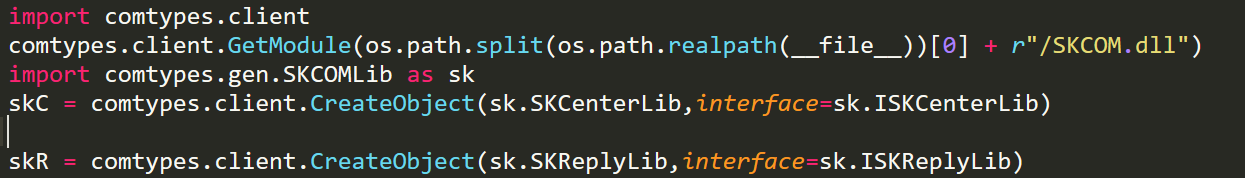
OnSmartData是智慧單回報，每一筆資料以「#」分隔每一個欄位。  
每個欄位的詳細資料，請查看  
【策略王COM元件使用說明】4-3-k OnSmartData

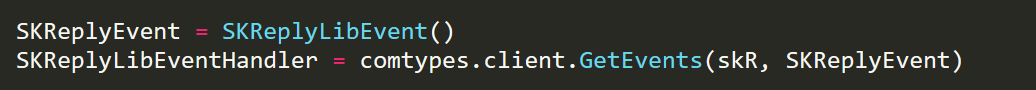
▼下方是 智慧單回報 的部分程式碼  


* 1. 註冊公告及接收公告方式

在登入SKCenterLib\_Login or SKCenterLib\_LoginSetQuote前，必須先註冊回報中的公告。

先建立SKReplyLib 物件及其事件SKReplyLib OnNotifyEvent





【策略王COM元件使用說明】4-3-e OnReplyMessage

▼下方是SKReplyLibEvent class中，處理公告通知OnReplyMessage函式 的部分程式碼並指定回傳-1

