策略王COM元件使用說明

-下單-海外期選智慧單-

# 函式範例

## SKOrderLib(海外期選)

*函式*

智慧單

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| **海期二擇一單委託** | [SendOverSeaFutureOCOOrder](#_SendOverSeaFutureOCOOrder) |  |
| **海期AB單委託** | [SendOverSeaFutureABOrder](#_SendOverSeaFutureABOrder) |  |
| **海期智慧單刪單** | [CancelOFStrategyOrder](#_CancelOFStrategyOrder) |  |
| **取消多筆智慧單委託** | [CancelStrategyList](#_CancelStrategyList) |  |
| **海期智慧單被動查詢** | [GetOFSmartStrategyReport](#_GetOFSmartStrategyReport) |  |

*事件*

|  |  |  |
| --- | --- | --- |
| **功能** | **事件名稱** | **備註** |
| **非同步委託結果** | [OnAsyncOrder](#_OnAsyncOrder) |  |
| **海期智慧單資料** | [OnOFSmartStrategyReport](#_OnOFSmartStrategyReport) |  |

#### **智慧單**

##### SendOverSeaFutureOCOOrder

|  |  |  |
| --- | --- | --- |
| <請先執行商品檔下載 SKOrderLib\_LoadOSCommodity()> (可至文件「**9.下單-海外期選**」查看)  送出海外期貨OCO(含長效)委託。 | | |
| **宣告** | Long SendOverSeaFutureOCOOrder([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **OVERSEAFUTUREORDER**\* pOrder,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| pOrder | SKCOM元件中的 [**OVERSEAFUTUREORDER**](#_海期智慧單OCO)物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **同步委託：**如果回傳值為 0表示委託成功，訊息內容則為13碼的委託序號。回傳值**非0**表示委託失敗，訊息內容為失敗原因。  **非同步委託：**參照OnAsyncOrder。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。  **當錯誤代碼為4碼時，可參考文件「4.下單準備介紹」裡面的【錯誤代碼定義表】，所屬委託物件亦有欄位值說明供參。** | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_4-2-b_OnAsyncOrder)取得。 | |

|  |
| --- |
| Python範例程式碼\_OFStrategyOrder.py類別OCOForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # checkBoxAsyncOrder          # 是否為非同步委託          self.checkBoxAsyncOrder = tk.Checkbutton(self)          self.var1 = tk.IntVar()          self.checkBoxAsyncOrder["variable"] = self.var1          self.checkBoxAsyncOrder["onvalue"] = True          self.checkBoxAsyncOrder["offvalue"] = False          self.checkBoxAsyncOrder["text"] = "非同步委託"          self.checkBoxAsyncOrder["command"] = self.checkBoxAsyncOrder\_CheckedChanged          self.checkBoxAsyncOrder.grid( row = 0,column = 0)          # textBoxExchangeNo          tk.Label(self, text = "交易所代碼 (EX: CME)").grid(row=0, column=1)          #輸入框          self.textBoxExchangeNo = tk.Entry(self)          self.textBoxExchangeNo.grid(row=0, column=2)          global textBoxExchangeNo          textBoxExchangeNo = self.textBoxExchangeNo            # textBoxStockNo          tk.Label(self, text = "商品代碼 (EX: ES)").grid(row=0, column=3)          #輸入框          self.textBoxStockNo = tk.Entry(self)          self.textBoxStockNo.grid(row=0, column=4)          global textBoxStockNo          textBoxStockNo = self.textBoxStockNo            # textBoxbstrYearMonth          tk.Label(self, text = "商品年月(YYYYMM, EX: 202206)").grid(row=1, column=1)          #輸入框          self.textBoxbstrYearMonth = tk.Entry(self)          self.textBoxbstrYearMonth.grid(row=1, column=2)          global textBoxbstrYearMonth          textBoxbstrYearMonth = self.textBoxbstrYearMonth            # textBoxbstrTrigger          tk.Label(self, text = "第一隻腳觸發價(大於)").grid(row=2, column=1)          #輸入框          self.textBoxbstrTrigger = tk.Entry(self)          self.textBoxbstrTrigger.grid(row=2, column=2)          global textBoxbstrTrigger          textBoxbstrTrigger = self.textBoxbstrTrigger            # textBoxTriggerNumerator          tk.Label(self, text = "分子").grid(row=2, column=3)          #輸入框          self.textBoxTriggerNumerator = tk.Entry(self)          self.textBoxTriggerNumerator.grid(row=2, column=4)          global textBoxTriggerNumerator          textBoxTriggerNumerator = self.textBoxTriggerNumerator            # textBoxbstrTriggerDenominator          tk.Label(self, text = "分母").grid(row=2, column=5)          #輸入框          self.textBoxbstrTriggerDenominator = tk.Entry(self)          self.textBoxbstrTriggerDenominator.grid(row=2, column=6)          global textBoxbstrTriggerDenominator          textBoxbstrTriggerDenominator = self.textBoxbstrTriggerDenominator            # comboBoxsBuySell          tk.Label(self, text = "買賣別(0:買進, 1:賣出)").grid(row=2, column=7)              #輸入框          self.comboBoxsBuySell = ttk.Combobox(self, state='readonly')          self.comboBoxsBuySell['values'] = Config.comboBoxsBuySell          self.comboBoxsBuySell.grid(row=2, column=8)          global comboBoxsBuySell          comboBoxsBuySell = self.comboBoxsBuySell            # comboBoxnOrderPriceType          tk.Label(self, text = "委託價(1:市價,  2:限價)").grid(row=2, column=9)              #輸入框          self.comboBoxnOrderPriceType = ttk.Combobox(self, state='readonly')          self.comboBoxnOrderPriceType['values'] = Config.comboBoxnOrderPriceType          self.comboBoxnOrderPriceType.grid(row=2, column=10)          global comboBoxnOrderPriceType          comboBoxnOrderPriceType = self.comboBoxnOrderPriceType            # textBoxbstrOrder          tk.Label(self, text = "委託價").grid(row=2, column=11)          #輸入框          self.textBoxbstrOrder = tk.Entry(self)          self.textBoxbstrOrder.grid(row=2, column=12)          global textBoxbstrOrder          textBoxbstrOrder = self.textBoxbstrOrder            # textBoxbstrOrderNumerator          tk.Label(self, text = "分子").grid(row=2, column=13)          #輸入框          self.textBoxbstrOrderNumerator = tk.Entry(self)          self.textBoxbstrOrderNumerator.grid(row=2, column=14)          global textBoxbstrOrderNumerator          textBoxbstrOrderNumerator = self.textBoxbstrOrderNumerator            # textBoxbstrOrderDenominator          tk.Label(self, text = "分母").grid(row=2, column=15)          #輸入框          self.textBoxbstrOrderDenominator = tk.Entry(self)          self.textBoxbstrOrderDenominator.grid(row=2, column=16)          global textBoxbstrOrderDenominator          textBoxbstrOrderDenominator = self.textBoxbstrOrderDenominator            # comboBoxsTradeType          tk.Label(self, text = "委託時效(0:ROD, 3:IOC, 4:FOK)").grid(row=2, column=17)              #輸入框          self.comboBoxsTradeType = ttk.Combobox(self, state='readonly')          self.comboBoxsTradeType['values'] = Config.comboBoxsTradeType          self.comboBoxsTradeType.grid(row=2, column=18)          global comboBoxsTradeType          comboBoxsTradeType = self.comboBoxsTradeType            # textBoxbstrTrigger2          tk.Label(self, text = "第一隻腳觸發價(小於)").grid(row=3, column=1)          #輸入框          self.textBoxbstrTrigger2 = tk.Entry(self)          self.textBoxbstrTrigger2.grid(row=3, column=2)          global textBoxbstrTrigger2          textBoxbstrTrigger2 = self.textBoxbstrTrigger2            # textBoxbstrTriggerNumerator2          tk.Label(self, text = "分子").grid(row=3, column=3)          #輸入框          self.textBoxbstrTriggerNumerator2 = tk.Entry(self)          self.textBoxbstrTriggerNumerator2.grid(row=3, column=4)          global textBoxbstrTriggerNumerator2          textBoxbstrTriggerNumerator2 = self.textBoxbstrTriggerNumerator2            # textBoxbstrTriggerDenominator2          tk.Label(self, text = "分母").grid(row=3, column=5)          #輸入框          self.textBoxbstrTriggerDenominator2 = tk.Entry(self)          self.textBoxbstrTriggerDenominator2.grid(row=3, column=6)          global textBoxbstrTriggerDenominator2          textBoxbstrTriggerDenominator2 = self.textBoxbstrTriggerDenominator2            # comboBoxnBuySell2          tk.Label(self, text = "買賣別(0:買進, 1:賣出)").grid(row=3, column=7)              #輸入框          self.comboBoxnBuySell2 = ttk.Combobox(self, state='readonly')          self.comboBoxnBuySell2['values'] = Config.comboBoxnBuySell2          self.comboBoxnBuySell2.grid(row=3, column=8)          global comboBoxnBuySell2          comboBoxnBuySell2 = self.comboBoxnBuySell2            # textBoxbstrOrder2          tk.Label(self, text = "委託價").grid(row=3, column=11)          #輸入框          self.textBoxbstrOrder2 = tk.Entry(self)          self.textBoxbstrOrder2.grid(row=3, column=12)          global textBoxbstrOrder2          textBoxbstrOrder2 = self.textBoxbstrOrder2            # textBoxbstrOrderNumerator2          tk.Label(self, text = "分子").grid(row=3, column=13)          #輸入框          self.textBoxbstrOrderNumerator2 = tk.Entry(self)          self.textBoxbstrOrderNumerator2.grid(row=3, column=14)          global textBoxbstrOrderNumerator2          textBoxbstrOrderNumerator2 = self.textBoxbstrOrderNumerator2            # textBoxbstrOrderDenominator2          tk.Label(self, text = "分母").grid(row=3, column=15)          #輸入框          self.textBoxbstrOrderDenominator2 = tk.Entry(self)          self.textBoxbstrOrderDenominator2.grid(row=3, column=16)          global textBoxbstrOrderDenominator2          textBoxbstrOrderDenominator2 = self.textBoxbstrOrderDenominator2            # textBoxnQty          tk.Label(self, text = "委託口數").grid(row=4, column=1)          #輸入框          self.textBoxnQty = tk.Entry(self)          self.textBoxnQty.grid(row=4, column=2)          global textBoxnQty          textBoxnQty = self.textBoxnQty            # comboBoxsDayTrade          tk.Label(self, text = "當沖(0:否, 1:是)").grid(row=4, column=3)              #輸入框          self.comboBoxsDayTrade = ttk.Combobox(self, state='readonly')          self.comboBoxsDayTrade['values'] = Config.comboBoxsDayTrade          self.comboBoxsDayTrade.grid(row=4, column=4)          global comboBoxsDayTrade          comboBoxsDayTrade = self.comboBoxsDayTrade            # comboBoxnReserved          tk.Label(self, text = "預約單(0:否, 1:是)").grid(row=5, column=1)              #輸入框          self.comboBoxnReserved = ttk.Combobox(self, state='readonly')          self.comboBoxnReserved['values'] = Config.comboBoxnReserved          self.comboBoxnReserved.grid(row=5, column=2)          global comboBoxnReserved          comboBoxnReserved = self.comboBoxnReserved            # comboBoxnTimeFlag          tk.Label(self, text = "預約盤別(1:T盤, 2:T+1盤)").grid(row=5, column=3)              #輸入框          self.comboBoxnTimeFlag = ttk.Combobox(self, state='readonly')          self.comboBoxnTimeFlag['values'] = Config.comboBoxnTimeFlag          self.comboBoxnTimeFlag.grid(row=5, column=4)          global comboBoxnTimeFlag          comboBoxnTimeFlag = self.comboBoxnTimeFlag            # comboBoxnLongActionFlag          tk.Label(self, text = "是否為長效單(0:否, 1:是)").grid(row=6, column=1)              #輸入框          self.comboBoxnLongActionFlag = ttk.Combobox(self, state='readonly')          self.comboBoxnLongActionFlag['values'] = Config.comboBoxnLongActionFlag          self.comboBoxnLongActionFlag.grid(row=6, column=2)          global comboBoxnLongActionFlag          comboBoxnLongActionFlag = self.comboBoxnLongActionFlag            # textBoxbstrLongEndDate          tk.Label(self, text = "長效單結束日期(YYYYMMDD共8碼, EX: 20220630)").grid(row=7, column=1)          #輸入框          self.textBoxbstrLongEndDate = tk.Entry(self)          self.textBoxbstrLongEndDate.grid(row=7, column=2)          global textBoxbstrLongEndDate          textBoxbstrLongEndDate = self.textBoxbstrLongEndDate            # comboBoxnLAType          tk.Label(self, text = "觸發結束條件(1:效期內觸發即失效, 3:效期內完全成交即失效)").grid(row=8, column=1)              #輸入框          self.comboBoxnLAType = ttk.Combobox(self, state='readonly')          self.comboBoxnLAType['values'] = Config.comboBoxnLAType          self.comboBoxnLAType.grid(row=8, column=2)          global comboBoxnLAType          comboBoxnLAType = self.comboBoxnLAType          # buttonSendOverSeaFutureOCOOrder          self.buttonSendOverSeaFutureOCOOrder = tk.Button(self)          self.buttonSendOverSeaFutureOCOOrder["text"] = "OCO送出"          self.buttonSendOverSeaFutureOCOOrder["command"] = self.buttonSendOverSeaFutureOCOOrder\_Click          self.buttonSendOverSeaFutureOCOOrder.grid(row=9, column=1) |

|  |
| --- |
| Config.py |
| # OCOForm  comboBoxsBuySell =(      "0:買進", "1:賣出"  )  comboBoxnOrderPriceType =(      "1:市價", "2:限價"  )  comboBoxsTradeType =(      "0:ROD", "3:IOC", "4:FOK"  )  comboBoxnBuySell2 =(      "0:買進", "1:賣出"  )  comboBoxsDayTrade =(      "0:否", "1:是"  )  comboBoxnReserved =(      "0:否", "1:是"  )  comboBoxnTimeFlag =(      "1:T盤", "2:T+1盤"  )  comboBoxnLongActionFlag =(      "0:否", "1:是"  )  comboBoxnLAType =(      "1:效期內觸發即失效", "3:效期內完全成交即失效"  ) |
| 全域變數 |
| # 群益API元件導入Python code內用的物件宣告  m\_pSKCenter = comtypes.client.CreateObject(sk.SKCenterLib,interface=sk.ISKCenterLib)  m\_pSKOrder = comtypes.client.CreateObject(sk.SKOrderLib,interface=sk.ISKOrderLib)  # 是否為非同步委託  bAsyncOrder = False |
| Current Block(放置於def createWidgets(self): 的同一層) |
| # checkBoxAsyncOrder      def checkBoxAsyncOrder\_CheckedChanged(self):          global bAsyncOrder          if self.var1.get() == True:              bAsyncOrder = True          else:              bAsyncOrder = False        def buttonSendOverSeaFutureOCOOrder\_Click(self):            pOrder = sk.OVERSEAFUTUREORDER()          pOrder.bstrFullAccount = comboBoxAccount.get()          pOrder.bstrExchangeNo = textBoxExchangeNo.get()          pOrder.bstrStockNo = textBoxStockNo.get()          pOrder.bstrYearMonth = textBoxbstrYearMonth.get()          pOrder.bstrTrigger = textBoxbstrTrigger.get()          pOrder.bstrTriggerNumerator = textBoxTriggerNumerator.get()          pOrder.bstrTriggerDenominator = textBoxbstrTriggerDenominator.get()          pOrder.bstrOrder = textBoxbstrOrder.get()          pOrder.bstrOrderNumerator = textBoxbstrOrderNumerator.get()          pOrder.bstrOrderDenominator = textBoxbstrOrderDenominator.get()          if (comboBoxsBuySell.get() == "0:買進"):              pOrder.sBuySell = 0          elif (comboBoxsBuySell.get() == "1:賣出"):              pOrder.sBuySell = 1          pOrder.bstrOrder2 = textBoxbstrOrder2.get()          pOrder.bstrOrderNumerator2 = textBoxbstrOrderNumerator2.get()          pOrder.bstrOrderDenominator2 = textBoxbstrOrderDenominator2.get()          pOrder.bstrTrigger2 = textBoxbstrTrigger2.get()          pOrder.bstrTriggerNumerator2 = textBoxbstrTriggerNumerator2.get()          pOrder.bstrTriggerDenominator2 = textBoxbstrTriggerDenominator2.get()          if (comboBoxnBuySell2.get() == "0:買進"):              pOrder.nBuySell2 = 0          elif (comboBoxnBuySell2.get() == "1:賣出"):              pOrder.nBuySell2 = 1          if (comboBoxsTradeType.get() == "0:ROD"):              pOrder.sTradeType = 0          elif (comboBoxsTradeType.get() == "3:IOC"):              pOrder.sTradeType = 3          else:              pOrder.sTradeType = 4          if (comboBoxnReserved.get() == "0:否"):              pOrder.nReserved = 0          else:              pOrder.nReserved = 1          if (comboBoxnTimeFlag.get() == "1:T盤"):              pOrder.nTimeFlag = 1          else:              pOrder.nTimeFlag = 2          if (comboBoxnLongActionFlag.get() == "0:否"):              pOrder.nLongActionFlag = 0          else:              pOrder.nLongActionFlag = 1          pOrder.bstrLongEndDate = textBoxbstrLongEndDate.get()          if (comboBoxnLAType.get() == "1:效期內觸發即失效"):              pOrder.nLAType = 1          else:              pOrder.nLAType = 3          if (comboBoxnOrderPriceType.get() == "1:市價"):              pOrder.nOrderPriceType = 1          else:              pOrder.nOrderPriceType = 2          if (comboBoxsDayTrade.get() == "0:否"):              pOrder.sDayTrade = 0          else:              pOrder.sDayTrade = 1          pOrder.nQty = int(textBoxnQty.get())          # 送出海外期貨OCO(含長效)委託          bstrMessage,nCode= m\_pSKOrder.SendOverSeaFutureOCOOrder(comboBoxUserID.get(), bAsyncOrder, pOrder)          msg = "【SendOverSeaFutureOCOOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

##### SendOverSeaFutureABOrder

|  |  |  |
| --- | --- | --- |
| <請先執行商品檔下載 SKOrderLib\_LoadOSCommodity()> (可至文件「**9.下單-海外期選**」查看)  送出海外期貨AB單委託。 | | |
| **宣告** | Long SendOverSeaFutureABOrder([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **OVERSEAFUTUREORDER**\* pOrder,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| pOrder | SKCOM元件中的 [**OVERSEAFUTUREORDER**](#_海期智慧單AB)物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **同步委託：**如果回傳值為 0表示委託成功，訊息內容則為13碼的委託序號。回傳值**非0**表示委託失敗，訊息內容為失敗原因。  **非同步委託：**參照OnAsyncOrder。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。  **當錯誤代碼為4碼時，可參考文件「4.下單準備介紹」裡面的【錯誤代碼定義表】，所屬委託物件亦有欄位值說明供參。** | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_4-2-b_OnAsyncOrder)取得。 | |

|  |
| --- |
| Python範例程式碼\_OFStrategyOrder.py類別ABForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # checkBoxAsyncOrder          # 是否為非同步委託          self.checkBoxAsyncOrder = tk.Checkbutton(self)          self.var1 = tk.IntVar()          self.checkBoxAsyncOrder["variable"] = self.var1          self.checkBoxAsyncOrder["onvalue"] = True          self.checkBoxAsyncOrder["offvalue"] = False          self.checkBoxAsyncOrder["text"] = "非同步委託"          self.checkBoxAsyncOrder["command"] = self.checkBoxAsyncOrder\_CheckedChanged          self.checkBoxAsyncOrder.grid( row = 0,column = 0)          # textBoxExchangeNoAB          tk.Label(self, text = "A商品交易所代碼 (EX: CME)").grid(row=0, column=1)          #輸入框          self.textBoxExchangeNoAB = tk.Entry(self)          self.textBoxExchangeNoAB.grid(row=0, column=2)          global textBoxExchangeNoAB          textBoxExchangeNoAB = self.textBoxExchangeNoAB            # textBoxbstrStockNo2          tk.Label(self, text = "A商品商品代號").grid(row=0, column=3)          #輸入框          self.textBoxbstrStockNo2 = tk.Entry(self)          self.textBoxbstrStockNo2.grid(row=0, column=4)          global textBoxbstrStockNo2          textBoxbstrStockNo2 = self.textBoxbstrStockNo2          # comboBoxnMarketNo          tk.Label(self, text = "市場編號(1:國內證, 2:國內期, 3:國外證, 4:國外期)").grid(row=1, column=1)              #輸入框          self.comboBoxnMarketNo = ttk.Combobox(self, state='readonly')          self.comboBoxnMarketNo['values'] = Config.comboBoxnMarketNo          self.comboBoxnMarketNo.grid(row=1, column=2)          global comboBoxnMarketNo          comboBoxnMarketNo = self.comboBoxnMarketNo            # comboBoxOFSpecialTradeType          tk.Label(self, text = "0:非證券,1:上市, 2:上櫃").grid(row=1, column=3)              #輸入框          self.comboBoxOFSpecialTradeType = ttk.Combobox(self, state='readonly')          self.comboBoxOFSpecialTradeType['values'] = Config.comboBoxOFSpecialTradeType          self.comboBoxOFSpecialTradeType.grid(row=1, column=4)          global comboBoxOFSpecialTradeType          comboBoxOFSpecialTradeType = self.comboBoxOFSpecialTradeType            # textBoxbstrOrder2AB          tk.Label(self, text = "A商品市價").grid(row=2, column=1)          #輸入框          self.textBoxbstrOrder2AB = tk.Entry(self)          self.textBoxbstrOrder2AB.grid(row=2, column=2)          global textBoxbstrOrder2AB          textBoxbstrOrder2AB = self.textBoxbstrOrder2AB            # comboBoxnTriggerDirection          tk.Label(self, text = "觸價方向(1:GTE大於等於,2:LTE小於等於)").grid(row=2, column=3)              #輸入框          self.comboBoxnTriggerDirection = ttk.Combobox(self, state='readonly')          self.comboBoxnTriggerDirection['values'] = Config.comboBoxnTriggerDirection          self.comboBoxnTriggerDirection.grid(row=2, column=4)          global comboBoxnTriggerDirection          comboBoxnTriggerDirection = self.comboBoxnTriggerDirection            # textBoxbstrTriggerAB          tk.Label(self, text = "觸發價").grid(row=2, column=5)          #輸入框          self.textBoxbstrTriggerAB = tk.Entry(self)          self.textBoxbstrTriggerAB.grid(row=2, column=6)          global textBoxbstrTriggerAB          textBoxbstrTriggerAB = self.textBoxbstrTriggerAB            # textBoxTriggerNumeratorAB          tk.Label(self, text = "分子").grid(row=2, column=7)          #輸入框          self.textBoxTriggerNumeratorAB = tk.Entry(self)          self.textBoxTriggerNumeratorAB.grid(row=2, column=8)          global textBoxTriggerNumeratorAB          textBoxTriggerNumeratorAB = self.textBoxTriggerNumeratorAB            # textBoxbstrTriggerDenominatorAB          tk.Label(self, text = "分母").grid(row=2, column=9)          #輸入框          self.textBoxbstrTriggerDenominatorAB = tk.Entry(self)          self.textBoxbstrTriggerDenominatorAB.grid(row=2, column=10)          global textBoxbstrTriggerDenominatorAB          textBoxbstrTriggerDenominatorAB = self.textBoxbstrTriggerDenominatorAB            # comboBoxsBuySellAB          tk.Label(self, text = "買賣別(0:買進, 1:賣出)").grid(row=3, column=1)              #輸入框          self.comboBoxsBuySellAB = ttk.Combobox(self, state='readonly')          self.comboBoxsBuySellAB['values'] = Config.comboBoxsBuySellAB          self.comboBoxsBuySellAB.grid(row=3, column=2)          global comboBoxsBuySellAB          comboBoxsBuySellAB = self.comboBoxsBuySellAB            # comboBoxnBuySell2AB          tk.Label(self, text = "買賣別2(0:買進, 1:賣出) ，非價差商品請填0").grid(row=3, column=3)              #輸入框          self.comboBoxnBuySell2AB = ttk.Combobox(self, state='readonly')          self.comboBoxnBuySell2AB['values'] = Config.comboBoxnBuySell2AB          self.comboBoxnBuySell2AB.grid(row=3, column=4)          global comboBoxnBuySell2AB          comboBoxnBuySell2AB = self.comboBoxnBuySell2AB            # comboBoxsDayTradeAB          tk.Label(self, text = "是否為當沖(0:否, 1:是)").grid(row=3, column=5)              #輸入框          self.comboBoxsDayTradeAB = ttk.Combobox(self, state='readonly')          self.comboBoxsDayTradeAB['values'] = Config.comboBoxsDayTradeAB          self.comboBoxsDayTradeAB.grid(row=3, column=6)          global comboBoxsDayTradeAB          comboBoxsDayTradeAB = self.comboBoxsDayTradeAB            # comboBoxsTradeTypeAB          tk.Label(self, text = "委託時效(0:ROC, 3:IOC, 4:FOK, 委託海期範圍市價、海選市限價僅有ROD)").grid(row=3, column=7)              #輸入框          self.comboBoxsTradeTypeAB = ttk.Combobox(self, state='readonly')          self.comboBoxsTradeTypeAB['values'] = Config.comboBoxsTradeTypeAB          self.comboBoxsTradeTypeAB.grid(row=3, column=8)          global comboBoxsTradeTypeAB          comboBoxsTradeTypeAB = self.comboBoxsTradeTypeAB            # textBoxbstrExchangeNo2          tk.Label(self, text = "B商品交易所代碼 (EX: CME)").grid(row=4, column=1)          #輸入框          self.textBoxbstrExchangeNo2 = tk.Entry(self)          self.textBoxbstrExchangeNo2.grid(row=4, column=2)          global textBoxbstrExchangeNo2          textBoxbstrExchangeNo2 = self.textBoxbstrExchangeNo2            # textBoxStockNoAB          tk.Label(self, text = "B商品商品代碼 (EX: ES)").grid(row=4, column=3)          #輸入框          self.textBoxStockNoAB = tk.Entry(self)          self.textBoxStockNoAB.grid(row=4, column=4)          global textBoxStockNoAB          textBoxStockNoAB = self.textBoxStockNoAB            # textBoxbstrYearMonthAB          tk.Label(self, text = "商品契約月份(YYYYMM共6碼, EX: 202206)").grid(row=4, column=5)          #輸入框          self.textBoxbstrYearMonthAB = tk.Entry(self)          self.textBoxbstrYearMonthAB.grid(row=4, column=6)          global textBoxbstrYearMonthAB          textBoxbstrYearMonthAB = self.textBoxbstrYearMonthAB            # textBoxbstrYearMonth2          tk.Label(self, text = "商品契約月份2(非價差商品請填0)").grid(row=4, column=7)          #輸入框          self.textBoxbstrYearMonth2 = tk.Entry(self)          self.textBoxbstrYearMonth2.grid(row=4, column=8)          global textBoxbstrYearMonth2          textBoxbstrYearMonth2 = self.textBoxbstrYearMonth2            # comboBoxnSpreadFlag          tk.Label(self, text = "是否委託價差商品(0:否, 1:是)").grid(row=5, column=1)              #輸入框          self.comboBoxnSpreadFlag = ttk.Combobox(self, state='readonly')          self.comboBoxnSpreadFlag['values'] = Config.comboBoxnSpreadFlag          self.comboBoxnSpreadFlag.grid(row=5, column=2)          global comboBoxnSpreadFlag          comboBoxnSpreadFlag = self.comboBoxnSpreadFlag            # comboBoxsCallPut          tk.Label(self, text = "是否為選擇權(0:否, 1:Call, 2:Put)").grid(row=6, column=1)              #輸入框          self.comboBoxsCallPut = ttk.Combobox(self, state='readonly')          self.comboBoxsCallPut['values'] = Config.comboBoxsCallPut          self.comboBoxsCallPut.grid(row=6, column=2)          global comboBoxsCallPut          comboBoxsCallPut = self.comboBoxsCallPut            # textBoxbstrStrikePrice          tk.Label(self, text = "履約價格(非選擇權商品請填0)").grid(row=6, column=3)          #輸入框          self.textBoxbstrStrikePrice = tk.Entry(self)          self.textBoxbstrStrikePrice.grid(row=6, column=4)          global textBoxbstrStrikePrice          textBoxbstrStrikePrice = self.textBoxbstrStrikePrice            # comboBoxsNewClose          tk.Label(self, text = "新平倉(0:新倉, 1:平倉) 選擇權僅新、平倉，期貨商品僅新倉").grid(row=6, column=5)              #輸入框          self.comboBoxsNewClose = ttk.Combobox(self, state='readonly')          self.comboBoxsNewClose['values'] = Config.comboBoxsNewClose          self.comboBoxsNewClose.grid(row=6, column=6)          global comboBoxsNewClose          comboBoxsNewClose = self.comboBoxsNewClose            # textBoxnQtyAB          tk.Label(self, text = "委託口數").grid(row=7, column=1)          #輸入框          self.textBoxnQtyAB = tk.Entry(self)          self.textBoxnQtyAB.grid(row=7, column=2)          global textBoxnQtyAB          textBoxnQtyAB = self.textBoxnQtyAB            # comboBoxnOrderPriceTypeAB          tk.Label(self, text = "委託價類別(1:市價,  2:限價)").grid(row=7, column=3)              #輸入框          self.comboBoxnOrderPriceTypeAB = ttk.Combobox(self, state='readonly')          self.comboBoxnOrderPriceTypeAB['values'] = Config.comboBoxnOrderPriceTypeAB          self.comboBoxnOrderPriceTypeAB.grid(row=7, column=4)          global comboBoxnOrderPriceTypeAB          comboBoxnOrderPriceTypeAB = self.comboBoxnOrderPriceTypeAB            # textBoxbstrOrderAB          tk.Label(self, text = "委託價").grid(row=7, column=5)          #輸入框          self.textBoxbstrOrderAB = tk.Entry(self)          self.textBoxbstrOrderAB.grid(row=7, column=6)          global textBoxbstrOrderAB          textBoxbstrOrderAB = self.textBoxbstrOrderAB            # textBoxbstrOrderNumeratorAB          tk.Label(self, text = "分子").grid(row=7, column=7)          #輸入框          self.textBoxbstrOrderNumeratorAB = tk.Entry(self)          self.textBoxbstrOrderNumeratorAB.grid(row=7, column=8)          global textBoxbstrOrderNumeratorAB          textBoxbstrOrderNumeratorAB = self.textBoxbstrOrderNumeratorAB            # textBoxbstrOrderDenominatorAB          tk.Label(self, text = "分母").grid(row=7, column=9)          #輸入框          self.textBoxbstrOrderDenominatorAB = tk.Entry(self)          self.textBoxbstrOrderDenominatorAB.grid(row=7, column=10)          global textBoxbstrOrderDenominatorAB          textBoxbstrOrderDenominatorAB = self.textBoxbstrOrderDenominatorAB            # comboBoxnTimeFlagAB          tk.Label(self, text = "是否為預約單(0:否, 1:是)A商品為國內期選市場時可選擇預約單").grid(row=8, column=1)              #輸入框          self.comboBoxnTimeFlagAB = ttk.Combobox(self, state='readonly')          self.comboBoxnTimeFlagAB['values'] = Config.comboBoxnTimeFlagAB          self.comboBoxnTimeFlagAB.grid(row=8, column=2)          global comboBoxnTimeFlagAB          comboBoxnTimeFlagAB = self.comboBoxnTimeFlagAB          # buttonendOverSeaFutureABOrder          self.buttonendOverSeaFutureABOrder = tk.Button(self)          self.buttonendOverSeaFutureABOrder["text"] = "AB送出"          self.buttonendOverSeaFutureABOrder["command"] = self.buttonendOverSeaFutureABOrder\_Click          self.buttonendOverSeaFutureABOrder.grid(row=9, column=1) |

|  |
| --- |
| Config.py |
| # ABForm  comboBoxnMarketNo =(      "1:國內證", "2:國內期", "3:國外證", "4:國外期"  )  comboBoxOFSpecialTradeType =(     "0:非證券", "1:上市", "2:上櫃"  )  comboBoxnTriggerDirection =(      "1:GTE大於等於", "2:LTE小於等於"  )  comboBoxsBuySellAB =(      "0:買進", "1:賣出"  )  comboBoxnBuySell2AB =(      "0:買進", "1:賣出"  )  comboBoxsDayTradeAB =(      "0:否", "1:是"  )  comboBoxsTradeTypeAB =(      "0:ROD", "3:IOC", "4:FOK"  )  comboBoxnSpreadFlag =(      "0:否", "1:是"  )  comboBoxsCallPut =(     "0:否", "1:Call", "2:Put"  )  comboBoxsNewClose =(      "0:新倉", "1:平倉"  )  comboBoxnOrderPriceTypeAB =(      "1:市價", "2:限價"  )  comboBoxnTimeFlagAB =(      "0:否", "1:是"  ) |
| 全域變數 |
| # 群益API元件導入Python code內用的物件宣告  m\_pSKCenter = comtypes.client.CreateObject(sk.SKCenterLib,interface=sk.ISKCenterLib)  m\_pSKOrder = comtypes.client.CreateObject(sk.SKOrderLib,interface=sk.ISKOrderLib)  # 是否為非同步委託  bAsyncOrder = False |
| Current Block(放置於def createWidgets(self): 的同一層) |
| # checkBoxAsyncOrder      def checkBoxAsyncOrder\_CheckedChanged(self):          global bAsyncOrder          if self.var1.get() == True:              bAsyncOrder = True          else:              bAsyncOrder = False        def buttonendOverSeaFutureABOrder\_Click(self):          pOrder = sk.OVERSEAFUTUREORDER()          pOrder.bstrFullAccount = comboBoxAccount.get()          pOrder.bstrStockNo = textBoxStockNoAB.get()          pOrder.bstrExchangeNo = textBoxExchangeNoAB.get()          pOrder.bstrYearMonth = textBoxbstrYearMonthAB.get()          pOrder.bstrYearMonth2 = textBoxbstrYearMonth2.get()          pOrder.bstrOrder = textBoxbstrOrderAB.get()          pOrder.bstrOrderNumerator = textBoxbstrOrderNumeratorAB.get()          pOrder.bstrOrderDenominator = textBoxbstrOrderDenominatorAB.get()          pOrder.bstrTrigger = textBoxbstrTriggerAB.get()          pOrder.bstrTriggerNumerator = textBoxTriggerNumeratorAB.get()          pOrder.bstrTriggerDenominator = textBoxbstrTriggerDenominatorAB.get()          if (comboBoxsBuySellAB.get() == "0:買進"):              pOrder.sBuySell = 0          elif (comboBoxsBuySellAB.get() == "1:賣出"):              pOrder.sBuySell = 1          pOrder.bstrOrder2 = textBoxbstrOrder2AB.get()            if (comboBoxnBuySell2AB.get() == "0:買進"):              pOrder.nBuySell2 = 0          elif (comboBoxnBuySell2AB.get() == "1:賣出"):              pOrder.nBuySell2 = 1          if (comboBoxsTradeTypeAB.get() == "0:ROD"):              pOrder.sTradeType = 0          elif (comboBoxsTradeTypeAB.get() == "3:IOC"):              pOrder.sTradeType = 3          else:              pOrder.sTradeType = 4          if (comboBoxnTriggerDirection.get() == "1:GTE大於等於"):              pOrder.nTriggerDirection = 1          else:              pOrder.nTriggerDirection = 2          if (comboBoxnTimeFlagAB.get() == "0:否"):              pOrder.nTimeFlag = 0          else :              pOrder.nTimeFlag = 1          if (comboBoxnOrderPriceTypeAB.get() == "1:市價"):              pOrder.nOrderPriceType = 1          else:              pOrder.nOrderPriceType = 2          if (comboBoxsDayTradeAB.get() == "0:否"):              pOrder.sDayTrade = 0          else:              pOrder.sDayTrade = 1          pOrder.nQty = int(textBoxnQtyAB.get())          pOrder.bstrStockNo2 = textBoxbstrStockNo2.get()          if (comboBoxnMarketNo.get() == "1:國內證"):              pOrder.nMarketNo = 1          elif (comboBoxnMarketNo.get() == "2:國內期"):              pOrder.nMarketNo = 2          elif (comboBoxnMarketNo.get() == "3:國外證"):              pOrder.nMarketNo = 3          else:              pOrder.nMarketNo = 4          if (comboBoxOFSpecialTradeType.get() == "0:非證券"):              pOrder.sSpecialTradeType = 0          elif (comboBoxOFSpecialTradeType.get() == "1:上市"):              pOrder.sSpecialTradeType = 1          else:              pOrder.sSpecialTradeType = 2          if (comboBoxsCallPut.get() == "0:否"):              pOrder.sCallPut = 0          elif (comboBoxsCallPut.get() == "1:Call"):              pOrder.sCallPut = 1          else:              pOrder.sCallPut = 2          pOrder.bstrStrikePrice = textBoxbstrStrikePrice.get()          if (comboBoxnSpreadFlag.get() == "0:否"):              pOrder.nSpreadFlag = 0          else:              pOrder.nSpreadFlag = 1          pOrder.bstrYearMonth2 = textBoxbstrYearMonth2.get()          if (comboBoxsNewClose.get() == "0:新倉"):              pOrder.sNewClose = 0          else:              pOrder.sNewClose = 1          # 送出海外期貨AB單委託          bstrMessage,nCode= m\_pSKOrder.SendOverSeaFutureABOrder(comboBoxUserID.get(), bAsyncOrder, pOrder)          msg = "【SendOverSeaFutureABOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

##### CancelOFStrategyOrder

|  |  |  |  |
| --- | --- | --- | --- |
| 取消海期智慧單委託。刪單欄位請參考[GetOFSmartStrategyReport](#_GetOFSmartStrategyReport)回傳的內容。 | | | |
| **宣告** | | Long CancelOFStrategyOrder ([in] BSTR bstrLogInID, [in] struct **CANCELSTRATEGYORDER**\* pCancelOrder, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | | 登入ID。 |
| pCancelOrder | | 智慧單刪單物件(參考[CANCELSTRATEGYORDER](#_海期智慧單刪單OCO、AB)) |
| bstrMessage | | **非同步刪單：**  參照OnAsyncOrder。  如果回傳值為 0表示刪單成功，訊息內容則為刪單之Thread ID。  回傳值非0表示刪單失敗，訊息內容為失敗原因。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_4-2-b_OnAsyncOrder)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  刪單後，請透過智慧單被動回報確認狀態。 | | |

|  |
| --- |
| Python範例程式碼\_OFStrategyOrder.py類別CancelForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # checkBoxAsyncOrder          # 是否為非同步委託          self.checkBoxAsyncOrder = tk.Checkbutton(self)          self.var1 = tk.IntVar()          self.checkBoxAsyncOrder["variable"] = self.var1          self.checkBoxAsyncOrder["onvalue"] = True          self.checkBoxAsyncOrder["offvalue"] = False          self.checkBoxAsyncOrder["text"] = "非同步委託"          self.checkBoxAsyncOrder["command"] = self.checkBoxAsyncOrder\_CheckedChanged          self.checkBoxAsyncOrder.grid( row = 0,column = 0)          # textBoxbstrSmartKey          tk.Label(self, text = "智慧單號").grid(row=0, column=1)          #輸入框          self.textBoxbstrSmartKey = tk.Entry(self)          self.textBoxbstrSmartKey.grid(row=0, column=2)          global textBoxbstrSmartKey          textBoxbstrSmartKey = self.textBoxbstrSmartKey          # comboBoxnTradeKind          tk.Label(self, text = "3：OCO、10：AB").grid(row=1, column=1)              #輸入框          self.comboBoxnTradeKind = ttk.Combobox(self, state='readonly')          self.comboBoxnTradeKind['values'] = Config.comboBoxnTradeKind          self.comboBoxnTradeKind.grid(row=1, column=2)          global comboBoxnTradeKind          comboBoxnTradeKind = self.comboBoxnTradeKind            # comboBoxnMarket          tk.Label(self, text = "市場別(AB單需選欲刪單之A商品市場)").grid(row=2, column=1)              #輸入框          self.comboBoxnMarket = ttk.Combobox(self, state='readonly')          self.comboBoxnMarket['values'] = Config.comboBoxnMarket          self.comboBoxnMarket.grid(row=2, column=2)          global comboBoxnMarket          comboBoxnMarket = self.comboBoxnMarket            # textBoxbstrSeqNo          tk.Label(self, text = "委託序號 (預約單可忽略)").grid(row=3, column=1)          #輸入框          self.textBoxbstrSeqNo = tk.Entry(self)          self.textBoxbstrSeqNo.grid(row=3, column=2)          global textBoxbstrSeqNo          textBoxbstrSeqNo = self.textBoxbstrSeqNo          # textBoxbstrOrderNo          tk.Label(self, text = "委託書號（若觸發，需給書號）").grid(row=4, column=1)          #輸入框          self.textBoxbstrOrderNo = tk.Entry(self)          self.textBoxbstrOrderNo.grid(row=4, column=2)          global textBoxbstrOrderNo          textBoxbstrOrderNo = self.textBoxbstrOrderNo            # textBoxbstrLongActionKey          tk.Label(self, text = "長效單號(非長效單可忽略)").grid(row=5, column=1)          #輸入框          self.textBoxbstrLongActionKey = tk.Entry(self)          self.textBoxbstrLongActionKey.grid(row=5, column=2)          global textBoxbstrLongActionKey          textBoxbstrLongActionKey = self.textBoxbstrLongActionKey          # buttonCancelOFStrategyOrder          self.buttonCancelOFStrategyOrder = tk.Button(self)          self.buttonCancelOFStrategyOrder["text"] = "刪單送出"          self.buttonCancelOFStrategyOrder["command"] = self.buttonCancelOFStrategyOrder\_Click          self.buttonCancelOFStrategyOrder.grid(row=6, column=2) |

|  |
| --- |
| Config.py |
| # CancelForm  comboBoxnTradeKind =(      "3：OCO", "10：AB單"  )  comboBoxnMarket =(      "1:國內證", "2:國內期", "3:國外證", "4:國外期"  ) |
| 全域變數 |
| # 群益API元件導入Python code內用的物件宣告  m\_pSKCenter = comtypes.client.CreateObject(sk.SKCenterLib,interface=sk.ISKCenterLib)  m\_pSKOrder = comtypes.client.CreateObject(sk.SKOrderLib,interface=sk.ISKOrderLib)  # 是否為非同步委託  bAsyncOrder = False |
| Current Block(放置於def createWidgets(self): 的同一層) |
| # checkBoxAsyncOrder      def checkBoxAsyncOrder\_CheckedChanged(self):          global bAsyncOrder          if self.var1.get() == True:              bAsyncOrder = True          else:              bAsyncOrder = False        def buttonCancelOFStrategyOrder\_Click(self):            pCancelOrder = sk.CANCELSTRATEGYORDER()          pCancelOrder.bstrFullAccount = comboBoxAccount.get()          pCancelOrder.bstrSmartKey = textBoxbstrSmartKey.get()          if (comboBoxnMarket.get() == "1：國內證"):              pCancelOrder.nMarket = 1          elif (comboBoxnMarket.get() == "2：國內期"):              pCancelOrder.nMarket = 2          elif (comboBoxnMarket.get() == "3：國外證"):              pCancelOrder.nMarket = 3          elif (comboBoxnMarket.get() == "4：國外期"):              pCancelOrder.nMarket = 4          if (comboBoxnTradeKind.get() == "3:OCO"):              pCancelOrder.nTradeKind = 3          else:              pCancelOrder.nTradeKind = 10          pCancelOrder.bstrSeqNo = textBoxbstrSeqNo.get()          pCancelOrder.bstrOrderNo = textBoxbstrOrderNo.get()          pCancelOrder.bstrLongActionKey = textBoxbstrLongActionKey.get()          # 取消海期智慧單委託。刪單欄位請參考GetOFStrategyOrder 回傳的內容          bstrMessage,nCode= m\_pSKOrder.CancelOFStrategyOrder(comboBoxUserID.get(), pCancelOrder)          msg = "【CancelOFStrategyOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

##### CancelStrategyList

|  |  |  |
| --- | --- | --- |
| 取消多筆智慧單委託。刪單欄位請參考智慧單被動回報回傳的內容。 | | |
| **宣告** | Long CancelStrategyList ([in] BSTR bstrLogInID, [in] struct CANCELSTRATEGYORDER\* pCancelOrder, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pCancelOrder | 智慧單刪單物件(參考[CANCELSTRATEGYORDER](#_智慧單多筆刪單)) |
| bstrMessage | **非同步刪單：**  參照OnAsyncOrder。  如果回傳值為 0表示刪單成功，訊息內容則為刪單之Thread ID。  回傳值非0表示刪單失敗，訊息內容為失敗原因。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_4-2-b_OnAsyncOrder)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  刪單後，請透過智慧單被動回報確認狀態。 | |

|  |
| --- |
| Python範例程式碼\_OFStrategyOrder.py類別CancelForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖  一張含有 文字, 字型, 行, 螢幕擷取畫面 的圖片  自動產生的描述  虛線處請參考取消智慧單委託 |
| # buttonCancelStrategyList          self.buttonCancelStrategyList = tk.Button(self)          self.buttonCancelStrategyList["text"] = "多筆刪單送出(以逗號分隔，EX：682020,682021)"          self.buttonCancelStrategyList["command"] = self.buttonCancelStrategyList\_Click          self.buttonCancelStrategyList.grid(row=1, column=3) |

|  |
| --- |
| 全域變數 |
| # 群益API元件導入Python code內用的物件宣告  m\_pSKCenter = comtypes.client.CreateObject(sk.SKCenterLib,interface=sk.ISKCenterLib)  m\_pSKOrder = comtypes.client.CreateObject(sk.SKOrderLib,interface=sk.ISKOrderLib) |
| Current Block(放置於def createWidgets(self): 的同一層) |
| def buttonCancelStrategyList\_Click(self):            pCancelOrder = sk.CANCELSTRATEGYORDER()          pCancelOrder.bstrLogInID = comboBoxUserID.get()          pCancelOrder.bstrFullAccount = comboBoxAccount.get()          if (comboBoxnMarket.get() == "1：國內證"):              pCancelOrder.nMarket = 1          elif (comboBoxnMarket.get() == "2：國內期"):              pCancelOrder.nMarket = 2          elif (comboBoxnMarket.get() == "3：國外證"):              pCancelOrder.nMarket = 3          elif (comboBoxnMarket.get() == "4：國外期"):              pCancelOrder.nMarket = 4          pCancelOrder.bstrSmartKey = textBoxbstrSmartKey.get()          # 取消多筆智慧單委託。刪單欄位請參考智慧單被動回報回傳的內容。          bstrMessage,nCode= m\_pSKOrder.CancelStrategyList(comboBoxUserID.get(), pCancelOrder)          msg = "【CancelStrategyList】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

##### GetOFSmartStrategyReport

|  |  |  |
| --- | --- | --- |
| 查詢海期智慧單。 | | |
| 宣告 | Long GetOFSmartStrategyReport ([in] BSTR bstrLogInID, [in] BSTR bstrAccount, [in] BSTR bstrMarketType, [in] LONG nReportStatus, [in] BSTR bstrKind, [in] BSTR bstrDate); | |
| 參數 | bstrLogInID | 登入ID。 |
| bstrAccount | 委託帳號 (帳號) 。 |
| bstrMarketType | OF :海期市場 |
| nReportStatus | 0:全部的委託單。 |
| bstrKind | 智慧單類型  **OCO：二擇一**  **AB：AB單** |
| bstrDate | 查詢日期(ex:20201001) |
| 回傳值 | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 結果由 [OnOFSmartStrategyReport](#_OnOFSmartStrategyReport) 事件回傳。 | |

|  |
| --- |
| Python範例程式碼\_OFStrategyOrder.py類別GetForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # comboBoxbstrKind          tk.Label(self, text = "智慧單類型(OCO：二擇一、AB：AB單)").grid(row=0, column=1)              #輸入框          self.comboBoxbstrKind = ttk.Combobox(self, state='readonly')          self.comboBoxbstrKind['values'] = Config.comboBoxbstrKind          self.comboBoxbstrKind.grid(row=0, column=2)          global comboBoxbstrKind          comboBoxbstrKind = self.comboBoxbstrKind          # textBoxbstrDate          tk.Label(self, text = "查詢日期(ex:20220601)").grid(row=1, column=1)          #輸入框          self.textBoxbstrDate = tk.Entry(self)          self.textBoxbstrDate.grid(row=1, column=2)          global textBoxbstrDate          textBoxbstrDate = self.textBoxbstrDate          # buttonGetOFSmartStrategyReport          self.buttonGetOFSmartStrategyReport = tk.Button(self)          self.buttonGetOFSmartStrategyReport["text"] = "查詢送出"          self.buttonGetOFSmartStrategyReport["command"] = self.buttonGetOFSmartStrategyReport\_Click          self.buttonGetOFSmartStrategyReport.grid(row=1, column=3) |

|  |
| --- |
| Config.py |
| # GetForm  comboBoxbstrKind =(      "OCO：二擇一", "AB：AB單"  ) |
| 全域變數 |
| # 群益API元件導入Python code內用的物件宣告  m\_pSKCenter = comtypes.client.CreateObject(sk.SKCenterLib,interface=sk.ISKCenterLib)  m\_pSKOrder = comtypes.client.CreateObject(sk.SKOrderLib,interface=sk.ISKOrderLib) |
| Current Block(放置於def createWidgets(self): 的同一層) |
| def buttonGetOFSmartStrategyReport\_Click(self):          if (comboBoxbstrKind.get() == "OCO:二擇一(長效單)"):              bstrKind = "OCO"          else:              bstrKind = "AB"          # 查詢海期智慧單          nCode= m\_pSKOrder.GetOFSmartStrategyReport(comboBoxUserID.get(), comboBoxAccount.get(),"OF",  0, bstrKind, textBoxbstrDate.get())          msg = "【GetOFSmartStrategyReport】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode)          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

# 事件

## OnAsyncOrder

|  |  |  |
| --- | --- | --- |
| 非同步委託結果。 | | |
| **宣告** | void OnAsyncOrder([in] LONG nThreadID, [in] LONG nCode, [in] BSTR bstrMessage ); | |
| **參數** | nThreadID | Thread ID |
| nCode | 收單結果代碼。 |
| bstrMessage | 收單回傳訊息。 |
| **備註** | 送單函式中使用非同步下單時會取得一個Thread ID，當此事件觸發時可藉由nThreadID對應下單來源。   * 委託成功：收單回傳訊息為 委託序號 * 委託失敗：收單回傳訊息為 委託失敗原因 | |
| 期貨智慧單: 如果回傳值為 0表示委託成功，訊息內容則為委託日期、智慧單序號、委託書號、委託內容(帳號等)，以實際回傳內容為主。  回傳成功訊息範例:  EX1: 000,20220419,智慧條件單已送出 條件單號：568157,無委託書號,568157,1689200000003  EX2: 000,20220420,智慧條件單已送出 條件單號：568353,y0004,568353,1688000000001 | |

|  |
| --- |
| Python範例程式碼\_OFStrategyOrder.py |
| UI程式碼(放置於 def createWidgets(self): 的下一層)/UI示意圖 |
| # richTextBoxMessage          self.richTextBoxMessage = tk.Listbox(self, height=5)          self.richTextBoxMessage.grid(column = 0, row = 10, columnspan=5, sticky = tk.E + tk.W)          global richTextBoxMessage          richTextBoxMessage = self.richTextBoxMessage |

|  |
| --- |
| 事件(可放在事件SKOrderLibEvent註冊物件底下一層) |
| # 非同步委託結果。      def OnAsyncOrder(self, nThreadID, nCode, bstrMessage):          msg = "【OnAsyncOrder】" + str(nThreadID) + str(nCode) + bstrMessage          richTextBoxMessage.insert('end', msg + "\n")          richTextBoxMessage.see('end') |

## OnOFSmartStrategyReport

### OCO單

|  |  |  |
| --- | --- | --- |
| 海期智慧單查詢。透過呼叫 [GetOFSmartStrategyReport](#_GetOFSmartStrategyReport) 後，資訊由該事件回傳。 | | |
| **宣告** | | void OnOFSmartStrategyReport([in] BSTR bstrData); |
| **參數** | | bstrData |
|  | | ＊注意：新版本回傳內容及欄位與舊版本相異。  如果回傳的第一筆資料是一位英文M接連著三碼數字及智慧單總數，其中 "M000" 開頭，表示為成功回傳，其後提供欲查詢之智慧單總數。  Ex:  (成功)M000,12  第二筆後面接的就是訊息內容。回傳的字串內容，以「,」分隔每一個欄位，欄位依序為： |
| 回傳的內容以「，」分隔每一個欄位，欄位依序為：  OCO：   |  |  | | --- | --- | | 名稱 | 備註 | | 智慧單號 |  | | 市場別 | OF：海外期選 | | 交易所代碼 | EX: CME、CBOT | | 分公司代碼 |  | | 帳號 |  | | 子帳 |  | | 委託書號 |  | | 序號 | (13碼)已觸發時，成交單含IOC、 FOK產生的取消單，可以以此欄比對市場上回報資料 | | 原始序號 | (13碼) | | 策略別 | 3：OCO單  10：AB單 | | 智慧單委託狀態 | 32：中台收單成功 33：中台收單失敗 34：洗價中  35：洗價中-觸發價更新(移動停損單) 36：洗價失敗  37：洗價觸發 38：觸發下單 39：下單失敗 40：使用者刪單 | | 交易日 |  | | 下單時間 | 建立智慧單時間 | | 商品代碼 | 第一腳商品代碼 | | 商品契約月份 | 第一腳商品年月 | | 履約價格 | 若無則為0 | | 買賣別 | B：買 ;S：賣 | | 委託價格別 | 第一委託價格別  7:使用者輸入價 | | 委託價格類別 | 2：限價 ;3：範圍市價 | | 委託時效 | 0： ROD當日有效 ;3： IOC立即成交; 4： FOK立即全部成交 | | 委託價格 |  | | 委託數量 |  | | 觸發價格 |  | | 觸發時間 |  | | 訊息 | 智慧單回報：[智慧單號][商品代碼]訊息  市場回報：參考市場委託狀態  訊息中逗號皆為全形”，” | | 市場委託狀態 | 空白：無委託單 0：預約 2：全部成交 3：全部取消 4：部分成交，剩餘已取消 5：部分成交，剩餘可取消 6：委託失敗 7:委託成功 8:取消失敗  F:【交易所動態退單相關代碼】 | | 智慧單異動時間 | 智慧單異動時間 | | 預約單註記 | 0:盤中單  1:預約單 | | 預約單序號 | 當預約單註記為時1 | | 交易日 |  | | 第一腳下單商品代碼 | (MST 、OCO 不提供)後台下單代碼 EX：FITX | | 第二腳下單商品代碼 | (MST 、OCO 不提供)後台下單代碼 | | 第二腳商品契約月份 |  | | 第一腳倉位 | 倉位 0：新倉; 1：平倉 ; 2：自動 | | 是否為當沖 | 空:非當沖； Y:當沖 | | 第一腳觸發價 |  | | 第二腳觸發價 |  | | 第二腳買賣別 | B:買；S:賣 | | 第二腳委託價格別 | 7:使用者輸入價 | | 第二腳委託價格類別 | 2：限價;3：範圍市價 | | 第二腳委託價 |  | | 第二腳委託時效 | 0： ROD當日有效 ;3： IOC立即成交; 4： FOK立即全部成交 | | 第二腳倉位 | 倉位 0：新倉; 1：平倉 ; 2：自動 | | 萬用訊息 | 適用於智慧單狀態代碼為999情況 | | 是否觸發即停止 | 1:條件觸發後，該筆長效單功能即失效 0:不管有無觸發，每日都送一次條件單，直到長效單結束日後才失效 | | 長效單註記 | 0：非長效單  1：長效單 | | 長效單序號 |  | | 長效單結束日期 | 20230829 | | 長效單結束條件 | 0：None  1：效期內觸發即失效  2：[停用]長效單結束日失效  3：效期內完全成交即失效 | | 長效單委託總量 |  | | 長效單成交總量 |  | | | |
| **備註** | 當全部資料已經全部回傳完畢，將回傳一筆以「##」開頭的內容，表示查詢結束。 | |

### AB單

|  |  |
| --- | --- |
| 海期智慧單查詢。透過呼叫 [GetOFSmartStrategyReport](#_GetOFSmartStrategyReport) 後，資訊由該事件回傳。 | |
| **宣告** | void OnOFSmartStrategyReport( [in] BSTR bstrData ); |
| **參數** | bstrData |
|  | ＊注意：新版本回傳內容及欄位與舊版本相異。  如果回傳的第一筆資料是一位英文M接連著三碼數字及智慧單總數，其中 "M000" 開頭，表示為成功回傳，其後提供欲查詢之智慧單總數。  Ex:  (成功)M000,12  第二筆後面接的就是訊息內容。回傳的字串內容，以「,」分隔每一個欄位，欄位依序為： |
| 回傳的內容以「,」分隔每一個欄位，欄位依序為：  AB：   |  |  | | --- | --- | | 名稱 | 備註 | | 智慧單號 |  | | 市場別 | OF：國外期選 | | 交易日 | 20230911 | | 交易所代碼 | Ex：CME、CBOT | | 分公司代碼 |  | | 帳號 |  | | 子帳 | 無子帳以0000000 | | 委託書號 |  | | 流水號13碼 | (13碼)已觸發時，成交單含IOC、 FOK產生的取消單，可以以此欄比對市場上回報資料 | | 原始流水號 | (13碼) | | 策略別 | 3：OCO單  10：AB單 | | 商品代碼 |  | | 下單市場別 |  | | 下單交易所別 |  | | 下單商品代碼 |  | | 商品契約月份 |  | | 履約價格 |  | | 下單商品代碼2 |  | | 商品契約月份2 |  | | 履約價格2 |  | | 買賣別 | B：買 S：賣 | | 委託價格別 | 第一委託價格別  7:使用者輸入價 | | 委託價格類別 | 1：市價；2：限價 | | 委託時效 | 0：ROD當日有效  3：IOC立即成交  4：FOK立即全部成交 | | 委託價格 |  | | 委託數量 |  | | 觸發價格 |  | | 觸發時間 |  | | 下單時間 | 2020-07-07 12:32:10.000 | | 智慧單委託狀態 | 32：中台收單成功 33：中台收單失敗 34：洗價中  35：洗價中-觸發價更新(移動停損單) 36：洗價失敗  37：洗價觸發 38：觸發下單 39：下單失敗 40：使用者刪單 | | 訊息 | 智慧單回報：[智慧單號][商品代碼]訊息  市場回報：參考市場委託狀態  訊息中逗號皆為全形”，” | | 智慧單異動時間 | 智慧單異動時間 | | 市場委託狀態 | 空白：無委託單 0：預約 2：全部成交 3：全部取消 4：部分成交，剩餘已取消 5：部分成交，剩餘可取消 6：委託失敗 7:委託成功 8:取消失敗  F:【交易所動態退單相關代碼】 | | 當下市價價格 |  | | 觸價方向 | 1：大於等於  2：小於等於 | | 是否為選擇權商品 | 0：否  1：是 | | 預約單註記 | 0：否  1：是 | | 預約單序號 |  | | 預約單交易日 |  | | 成交量 |  | | 新平倉 | 0：新單  1：平倉  2：自動 | | 是否為當沖 | 0：否  1：是 | | 買賣權 | N：None  C：Call  P：Put | | 委託價分子 |  | | 委託價分母 |  | | 成交價分子 |  | | 成交價分母 |  | | 觸發價分子 |  | | 觸發價分母 |  | | 萬用訊息 |  | | |
| 當全部資料已經全部回傳完畢，將回傳一筆以「##」開頭的內容，表示查詢結束。 | |

|  |
| --- |
| Python範例程式碼\_OFStrategyOrder.py |
| UI程式碼(放置於 def createWidgets(self): 的下一層)/UI示意圖 |
| # richTextBoxMessage          self.richTextBoxMessage = tk.Listbox(self, height=5)          self.richTextBoxMessage.grid(column = 0, row = 10, columnspan=5, sticky = tk.E + tk.W)          global richTextBoxMessage          richTextBoxMessage = self.richTextBoxMessage |

|  |
| --- |
| 事件(可放在事件SKOrderLibEvent註冊物件底下一層) |
| # 海期智慧單查詢。透過呼叫 GetOFSmartStrategyReport 後，資訊由該事件回傳。      def OnOFSmartStrategyReport(self, bstrData):          msg = "【OnOFSmartStrategyReport】" + bstrData          richTextBoxMessage.insert('end', msg + "\n")          richTextBoxMessage.see('end') |

# Struct 結構物件

## OVERSEAFUTUREORDER(海期智慧單下單物件)

### 海期智慧單OCO

struct OVERSEAFUTUREORDER

{

BSTR bstrFullAccount; //海期帳號，分公司四碼＋帳號7碼

|  |
| --- |
| //商品  BSTR bstrExchangeNo; //交易所代碼 (EX: CME)  BSTR bstrStockNo; //商品代碼 (EX: ES)  BSTR bstrYearMonth; //商品年月(YYYYMM, EX: 202206) |

|  |
| --- |
| //第一隻腳  BSTR bstrTrigger; //第一隻腳 觸發價(大於)  BSTR bstrTriggerNumerator; //第一隻腳 觸發價格分子  BSTR bstrTriggerDenominator; //第一隻腳 觸發價格分母  SHORT sBuySell; //第一隻腳 買賣別(0:買進, 1:賣出)  LONG nOrderPriceType; //第一隻腳 價格委託方式(1:市價, 2:限價)  BSTR bstrOrder; //第一隻腳 委託價格  BSTR bstrOrderNumerator; //第一隻腳 委託價格分子  BSTR bstrOrderDenominator; //第一隻腳 委託價格分母  LONG sTradeType; //第一隻腳 委託時效(0:ROD, 3:IOC, 4:FOK) |

|  |
| --- |
| //第二隻腳  BSTR bstrTrigger2; //第二隻腳 觸發價(小於)  BSTR bstrTriggerNumerator2; //第二隻腳 觸發價格分子  BSTR bstrTriggerDenominator2; //第二隻腳 觸發價格分母  LONG nBuySell2; //第二隻腳 買賣別(0:買進, 1:賣出)  BSTR bstrOrder2; //第二隻腳 委託價格  BSTR bstrOrderNumerator2; //第二隻腳 委託價格分子  BSTR bstrOrderDenominator2; //第二隻腳 委託價格分母 |

|  |
| --- |
| //條件  LONG nQty; //委託口數  SHORT sDayTrade; //當沖(0:否, 1:是)  LONG nReserved; //預約單(0:否, 1:是)  LONG nTimeFlag; //預約盤別(1:T盤, 2:T+1盤)  LONG nLongActionFlag; //是否為長效單(0:否, 1:是)  BSTR bstrLongEndDate //長效單結束日期(YYYYMMDD共8碼, EX: 20220630)  LONG nLAType //觸發結束條件(1:效期內觸發即失效, 3:短期內完全成交即失效) |

\*v2.13.47修正nTimeFlag參數1:T盤, 2:T+1盤

### 海期智慧單AB

struct OVERSEAFUTUREORDER

{

BSTR bstrFullAccount; //期貨帳號，分公司四碼＋帳號7碼

|  |
| --- |
| //看A  BSTR bstrExchangeNo; //A商品交易所代碼 (EX: CME)  BSTR bstrStockNo2; //A商品商品代號  LONG nMarketNo; //市場編號(1:國內證, 2:國內期, 3:國外證, 4:國外期)  SHORT sSpecialTradeType; //0:非證券,1:上市, 2:上櫃  BSTR bstrOrder2; //A商品市價  LONG nTriggerDirection; //觸價方向(1:GTE大於等於,2:LTE小於等於)  BSTR bstrTrigger; //觸發價  BSTR bstrTriggerNumerator; //觸發價分子(若無請填0)  BSTR bstrTriggerDenominator; //觸發價分母(若無請填0) |

|  |
| --- |
| //下B  SHORT sBuySell; //買賣別(0:買進, 1:賣出)  LONG nBuySell2; //買賣別2(0:買進, 1:賣出) ，非價差商品請填0  SHORT sDayTrade; //是否為當沖(0:否, 1:是)  LONG sTradeType; //委託時效(0:ROC, 3:IOC, 4:FOK, 委託海期範圍市價、海選市限價僅有ROD)  BSTR bstrExchangeNo2; //B商品交易所代碼 (EX: CME)  BSTR bstrStockNo; //B商品商品代碼 (EX: ES)  BSTR bstrYearMonth; //商品契約月份(YYYYMM共6碼, EX: 202206)  BSTR bstrYearMonth2; //商品契約月份2(YYYYMM共6碼, EX: 202206)，非價差商品請填0  LONG nSpreadFlag; //是否委託價差商品(0:否, 1:是)  SHORT sCallPut; //是否為選擇權(0:否, 1:Call, 2:Put)  BSTR bstrStrikePrice; //履約價格(非選擇權商品請填0)  SHORT sNewClose; //新平倉(0:新倉, 1:平倉) 選擇權僅新、平倉，期貨商品僅新倉  LONG nQty; //委託口數  LONG nOrderPriceType; //委託價類別(2:限價, 3:範圍市價)  BSTR bstrOrder; //委託價  BSTR bstrOrderNumerator; //委託價分子(若無請填0)  BSTR bstrOrderDenominator; //委託價分母(若無請填0)  LONG nTimeFlag; //是否為預約單(0:否, 1:是)A商品為國內期選市場時可選擇預約單 |

## CANCELSTRATEGYORDER(智慧單刪單物件)

### 海期智慧單刪單OCO、AB

struct CANCELSTRATEGYORDER

{

BSTR bstrFullAccount; //帳號，分公司四碼＋帳號7碼

LONG nMarket; //市場別4:國外期

BSTR bstrSmartKey; //智慧單號

LONG nTradeKind; //**3：OCO長效單、10：AB單**

BSTR bstrSeqNo; //委託序號 (預約單可忽略)

BSTR bstrOrderNo; //委託書號（若觸發，需給書號）

BSTR bstrLongActionKey; //長效單號(非長效單可忽略)

}

### 智慧單多筆刪單

struct CANCELSTRATEGYORDER

{

BSTR bstrFullAccount; //帳號

LONG nMarket; //刪單市場別，1:國內證, 2:國內期, 3:國外證, 4:國外期 (AB單刪單請選擇A商品之市場別)

BSTR bstrSmartKey; //多筆智慧單號 (以逗號分隔，EX：682020,682021)

}