策略王COM元件使用說明

-下單-國內期選-

# 函式範例

## SKOrderLib(國內期選)

*函式*

1. **查詢**

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| **(新)期貨未平倉查詢-GW** | [GetOpenInterest](#_GetOpenInterestGW)GW |  |
| **查詢國內權益數** | [GetFutureRights](#_GetFutureRights) |  |

1. **期貨商品互抵**

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| **(大小台/電/金)指定期貨商品互抵**  **大小台互抵** | |  | | --- | | [SendTFOffset](#_SendTFOffSet) | | [SendTXOffset](#_SendTXOffset) | |  |
| **指定大、小、微台互抵** | [SendTFOffsetNew](#_SendTFOffsetNew) |  |

1. **(選擇權)組合/複式單拆解/雙邊了結**

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| **組合部位** | [AssembleOptions](#_AssembleOptions) | 非交易行為，無回報 |
| **複式單拆解** | [DisassembleOptions](#_DisassembleOptions) | 非交易行為，無回報 |
| **雙邊部位了結** | [CoverAllProduct](#_CoverAllProduct) | 非交易行為，無回報 |
| **全拆組單、全拆組平單** | [AllCoverDisOptions](#_AllCoverDisOptions) | 非交易行為，無回報 |

1. **一般下單/刪改單**

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| **期貨委託含倉位盤別** | [SendFutureOrderCLR](#_SendFutureOrderCLR) | (新增)可選盤別、倉別 |
| **選擇權委託** | [SendOptionOrder](#_SendOptionOrder) |  |
| **國內選擇權複式單** | [SendDuplexOrder](#_SendDuplexOrder) |  |
| 1. **證期權改價** | [CorrectPriceBySeqNo](#_CorrectPriceBySeqNo) | 依序號改價 |
| 1. **證期權改價** | [CorrectPriceByBookNo](#_CorrectPriceByBookNo_1) | 依書號改價 |
| 1. **委託減量** | [DecreaseOrderBySeqNo](#_DecreaseOrderBySeqNo) |  |
| 1. **國內刪單-序號** | [CancelOrderBySeqNo](#_CancelOrderBySeqNo_1) | 13碼序號 |
| 1. **國內刪單-書號** | [CancelOrderByBookNo](#_CancelOrderByBookNo_1) | 依5碼書號刪單 |
| 1. **國內刪單-商品代號** | [CancelOrderByStockNo](#_CancelOrderByStockNo_1) | 依帳號所屬**登入ID**+商品代號 |
| 1. **國內刪單-商品代號(By帳號所屬登入ID+商品代號+買賣別+委託價格)** | [CancelOrderByStockNoAdvance](#_CancelOrderByStockNoAdvance) |  |

1. **Proxy下單/刪改單**

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| **期貨委託含倉位盤別** | [SendFutureProxyOrderCLR](#_SendFutureProxyOrderCLR) | 須選倉別、盤別 |
| **選擇權委託** | [SendOptionProxyOrder](#_SendOptionProxyOrder) |  |
| **選擇權複式下單** | [SendDuplexProxyOrder](#_SendDuplexProxyOrder) |  |
| **期貨刪改單** | [SendFutureProxyAlter](#_SendFutureProxyAlter) |  |
| **選擇權刪改單** | [SendOptionProxyAlter](#_SendOptionProxyAlter) |  |

*事件*

|  |  |  |
| --- | --- | --- |
| **功能** | **事件名稱** | **備註** |
| **期貨未平倉** | [OnOpenInterest](#_4-2-b_OnAsyncOrder) |  |
| **國內權益數** | [OnFutureRights](#_OnFutureRights) |  |
| **非同步委託結果** | [OnAsyncOrder](#_OnAsyncOrder) |  |
| **Proxy委託結果** | [OnProxyOrder](#_OnProxyOrder) |  |

#### **查詢**

##### 期貨未平倉

###### **GetOpenInterestGW**

|  |  |  |
| --- | --- | --- |
| 查詢期貨未平倉－可指定回傳格式。 | | |
| **宣告** | Long GetOpenInterestGW([in] BSTR bstrLogInID, [in] BSTR bstrAccount,[in] LONG nFormat); | |
| **參數** | bstrLogInID | 登入ID。 |
| bstrAccount | 委託帳號 ( IB＋帳號) 。 |
| nFormat | 回傳格式：1。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 結果由[OnOpenInterest](#_(新)期貨未平倉GW。透過呼叫_GetOpenInterestGW_後)事件回傳。  新增參數：指定回傳格式（內容參考[OnOpenInterest](#_4-2-b_OnAsyncOrder)） | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別ReadForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # buttonGetOpenInterestGW          self.buttonGetOpenInterestGW = tk.Button(self)          self.buttonGetOpenInterestGW["text"] = "期貨未平倉"          self.buttonGetOpenInterestGW["command"] = self.buttonGetOpenInterestGW\_Click          self.buttonGetOpenInterestGW.grid(row=0, column=1) |

|  |
| --- |
| 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 buttonGetOpenInterestGW\_Click(self):          nCode = m\_pSKOrder.GetOpenInterestGW(comboBoxUserID.get(), comboBoxAccount.get(), 1)          msg = "【buttonGetOpenInterestGW】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode)          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

##### GetFutureRights

|  |  |  |
| --- | --- | --- |
| 查詢國內權益數。 | | |
| **宣告** | Long GetFutureRights([in] BSTR bstrLogInID, [in] BSTR bstrAccount,[in]SHORT sCoinType) | |
| **參數** | bstrLogInID | 登入ID。 |
| bstrAccount | 委託帳號 ( IB＋帳號) 。 |
| sCoinType | 0:全幣別，1:基幣(台幣TWD)，2:人民幣RMB。 |
|  |  |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | * 結果由[OnFutureRights](#_OnFutureRights)事件回傳。 * 全幣別選項含基幣。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別ReadForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # buttonGetFutureRights          self.buttonGetFutureRights = tk.Button(self)          self.buttonGetFutureRights["text"] = "國內權益數"          self.buttonGetFutureRights["command"] = self.buttonGetFutureRights\_Click          self.buttonGetFutureRights.grid(row=1, column=1)          # comboBoxCoinType          tk.Label(self, text = "幣別").grid(row = 1,column = 2)          self.comboBoxCoinType = ttk.Combobox(self, state='readonly')          self.comboBoxCoinType['values'] = Config.comboBoxCoinType          self.comboBoxCoinType.grid(row=1, column=3)          global comboBoxCoinType          comboBoxCoinType = self.comboBoxCoinType |

|  |
| --- |
| Config.py |
| # ReadForm  comboBoxCoinType =(      "0:全幣別", "1:基幣(台幣TWD)", "2:人民幣RMB"  ) |
| 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 buttonGetFutureRights\_Click(self):          if (comboBoxCoinType.get() == "0:全幣別"):              sCoinType = 0          elif (comboBoxCoinType.get() == "1:基幣(台幣TWD)"):              sCoinType = 1          elif (comboBoxCoinType.get() == "2:人民幣RMB"):              sCoinType = 2            nCode = m\_pSKOrder.GetFutureRights(comboBoxUserID.get(), comboBoxAccount.get(), sCoinType)          msg = "【GetFutureRights】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode)          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

#### **期貨商品互抵**

##### SendTFOffSet

|  |  |  |
| --- | --- | --- |
| 指定大小台/電/金互抵。 | | |
| **宣告** | Long SendTFOffset([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] LONG nCommodity, [in] BSTR bstrYearMonth, [in] LONG nBuySell, [in] LONG nQty, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 委託帳號 ( IB＋帳號) 。 |
| nCommodity | 指定商品0:大小台; 1:大小電; 2:大小金 |
| bstrYearMonth | 大小台(電、金)年月(YYYYMM)。 |
| nBuySell | 大台(電、金)買賣別。0:多方(買) 1:空方(賣) |
| nQty | 互抵口數，以大台(電、金)口數為基本單位。 |
| bstrMessage | **同步：**如果回傳值為 0表示委託成功，訊息內容則為互抵訊息。回傳值非0表示互抵失敗，訊息內容為失敗原因。  **非同步：**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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 回傳值0 表示委託伺服器接收成功，詳細委託內容請致電交易室查詢，  回傳值非0時表示失敗，請參考訊息確認錯誤原因。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  EX:互抵需要1口大台抵4口小台(金)或一口大電抵8口小電，且大小台(電、金)需互為反向部位、相同月份，才可以互抵成功。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別OffSetForm(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)          # comboBoxSendTFOffSetnCommodity          tk.Label(self, text = "類別").grid(row = 0,column = 1)          self.comboBoxSendTFOffSetnCommodity = ttk.Combobox(self, state='readonly')          self.comboBoxSendTFOffSetnCommodity['values'] = Config.comboBoxSendTFOffSetnCommodity          self.comboBoxSendTFOffSetnCommodity.grid(row=0, column=2)          global comboBoxSendTFOffSetnCommodity          comboBoxSendTFOffSetnCommodity = self.comboBoxSendTFOffSetnCommodity          # textBoxSendTFOffSetbstrYearMonth          tk.Label(self, text = "年月").grid(row=1, column=1)          #輸入框          self.textBoxSendTFOffSetbstrYearMonth = tk.Entry(self)          self.textBoxSendTFOffSetbstrYearMonth.grid(row=1, column=2)          global textBoxSendTFOffSetbstrYearMonth          textBoxSendTFOffSetbstrYearMonth = self.textBoxSendTFOffSetbstrYearMonth          # comboBoxSendTFOffSetnBuySell          tk.Label(self, text = "買賣別").grid(row = 1,column = 3)          self.comboBoxSendTFOffSetnBuySell = ttk.Combobox(self, state='readonly')          self.comboBoxSendTFOffSetnBuySell['values'] = Config.comboBoxSendTFOffSetnBuySell          self.comboBoxSendTFOffSetnBuySell.grid(row=1, column=4)          global comboBoxSendTFOffSetnBuySell          comboBoxSendTFOffSetnBuySell = self.comboBoxSendTFOffSetnBuySell          # textBoxSendTFOffSetnQty          tk.Label(self, text = "互抵口數").grid(row=2, column=1)          #輸入框          self.textBoxSendTFOffSetnQty = tk.Entry(self)          self.textBoxSendTFOffSetnQty.grid(row=2, column=2)          global textBoxSendTFOffSetnQty          textBoxSendTFOffSetnQty = self.textBoxSendTFOffSetnQty          # buttonSendTFOffSet          self.buttonSendTFOffSet = tk.Button(self)          self.buttonSendTFOffSet["text"] = "大小台電金互抵"          self.buttonSendTFOffSet["command"] = self.buttonSendTFOffSet\_Click          self.buttonSendTFOffSet.grid(row=3, column=1)          global buttonSendTFOffSet          buttonSendTFOffSet = self.buttonSendTFOffSet |

|  |
| --- |
| Config.py |
| # OffSetForm  comboBoxSendTFOffSetnCommodity =(      "大小台", "大小電", "大小金"  )  comboBoxSendTFOffSetnBuySell =(      "多方(買)", "空方(賣)"  ) |
| 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 buttonSendTFOffSet\_Click(self):          if (comboBoxSendTFOffSetnCommodity.get() == "大小台"):              nCommodity = 0          elif (comboBoxSendTFOffSetnCommodity.get() == "大小電"):              nCommodity = 1          elif (comboBoxSendTFOffSetnCommodity.get() == "大小金"):              nCommodity = 2          if (comboBoxSendTFOffSetnBuySell.get() == "多方(買)"):              nBuySell = 0          elif (comboBoxSendTFOffSetnBuySell.get() == "空方(賣)"):              nBuySell = 1          nQty = int(textBoxSendTFOffSetnQty.get())          bstrMessage, nCode = m\_pSKOrder.SendTFOffSet(comboBoxUserID.get(), bAsyncOrder, comboBoxAccount.get(), nCommodity, textBoxSendTFOffSetbstrYearMonth.get(), nBuySell, nQty)          msg = "【SendTFOffSet】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

###### SendTXOffset

|  |  |  |
| --- | --- | --- |
| 大小台互抵。 | | |
| **宣告** | Long SendTXOffset([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrYearMonth, [in] LONG nBuySell, [in] LONG nQty, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 委託帳號 ( IB＋帳號) 。 |
| bstrYearMonth | 大小台年月(YYYYMM)。 |
| nBuySell | 大台買賣別。0:多方(買) 1:空方(賣) |
| nQty | 互抵口數，以大台口數為基本單位。 |
| bstrMessage | **同步：**如果回傳值為 0表示委託成功，訊息內容則為互抵訊息。回傳值非0表示互抵失敗，訊息內容為失敗原因。  **非同步：**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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 回傳值0 表示委託伺服器接收成功，詳細委託內容請致電交易室查詢，  回傳值非0時表示失敗，請參考訊息確認錯誤原因。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  大小台互抵需要1口大台抵4口小台，且大小台需互為反向部位、相同月份，才可以互抵成功。 | |

##### SendTFOffsetNew

|  |  |  |
| --- | --- | --- |
| 指定大、小、微台互抵。 | | |
| **宣告** | Long SendTFOffsetNew([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] LONG nCommodity, [in] BSTR bstrYearMonth, [in] LONG nBuySell, [in] LONG nQty, [in] LONG nQty\_2, [in] LONG nQty\_3 [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 委託帳號 ( IB＋帳號) 。 |
| nCommodity | 指定商品0:大抵微; 1:小抵微; 2:大小抵微; 3:大抵小微; 4:小抵大微; 5:大抵小 |
| bstrYearMonth | 大小微台年月(YYYYMM)。 |
| nBuySell | 大台微買賣別。0:多方(買) 1:空方(賣) |
| nQty | 互抵口數，以大台口數為基本單位。 |
| nQty\_2 | 互抵口數，以小台口數為基本單位。 |
| nQty\_3 | 互抵口數，以微台口數為基本單位。 |
| bstrMessage | **同步：**如果回傳值為 0表示委託成功，訊息內容則為互抵訊息。回傳值非0表示互抵失敗，訊息內容為失敗原因。  **非同步：**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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 回傳值0 表示委託伺服器接收成功，詳細委託內容請致電交易室查詢，  回傳值非0時表示失敗，請參考訊息確認錯誤原因。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  EX:互抵需要1口大台抵4口小台，且大小、微台相同月份，才可以互抵成功。1大口等於20微台，1小口等於5微台。  大抵小微：以大台為買賣別基準。EX: 2大口(Sell)抵1口小台(Buy)及20口微台(Buy)。  大小抵微：以大台、小台為買賣別基準。EX: 1大口(Buy)+ 1口小台(Buy)抵25口微台(Sell)。  \*不須進行互抵的商品請於口數(Qty)帶入0 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別 OffsetNewForm(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)  # comboBoxSendTFOffSetNewnCommodity  tk.Label(self, text = "類別").grid(row = 0,column = 1)  self.comboBoxSendTFOffSetNewnCommodity = ttk.Combobox(self, state='readonly')  self.comboBoxSendTFOffSetNewnCommodity['values'] = Config.comboBoxSendTFOffSetNewnCommodity  self.comboBoxSendTFOffSetNewnCommodity.grid(row=0, column=2)  global comboBoxSendTFOffSetNewnCommodity  comboBoxSendTFOffSetNewnCommodity = self.comboBoxSendTFOffSetNewnCommodity  # textBoxSendTFOffSetNewbstrYearMonth  tk.Label(self, text = "年月").grid(row=1, column=1)  #輸入框  self.textBoxSendTFOffSetNewbstrYearMonth = tk.Entry(self)  self.textBoxSendTFOffSetNewbstrYearMonth.grid(row=1, column=2)  global textBoxSendTFOffSetNewbstrYearMonth  textBoxSendTFOffSetNewbstrYearMonth = self.textBoxSendTFOffSetNewbstrYearMonth  # comboBoxSendTFOffSetNewnBuySell  tk.Label(self, text = "買賣別").grid(row = 1,column = 3)  self.comboBoxSendTFOffSetNewnBuySell = ttk.Combobox(self, state='readonly')  self.comboBoxSendTFOffSetNewnBuySell['values'] = Config.comboBoxSendTFOffSetNewnBuySell  self.comboBoxSendTFOffSetNewnBuySell.grid(row=1, column=4)  global comboBoxSendTFOffSetNewnBuySell  comboBoxSendTFOffSetNewnBuySell = self.comboBoxSendTFOffSetNewnBuySell  # textBoxSendTFOffSetNewnQty  tk.Label(self, text = "互抵口數(大台)").grid(row=2, column=1)  #輸入框  self.textBoxSendTFOffSetNewnQty = tk.Entry(self)  self.textBoxSendTFOffSetNewnQty.grid(row=2, column=2)  global textBoxSendTFOffSetNewnQty  textBoxSendTFOffSetNewnQty = self.textBoxSendTFOffSetNewnQty  # textBoxSendTFOffSetNewnQty\_2  tk.Label(self, text = "互抵口數(小台)").grid(row=2, column=3)  #輸入框  self.textBoxSendTFOffSetNewnQty\_2 = tk.Entry(self)  self.textBoxSendTFOffSetNewnQty\_2.grid(row=2, column=4)  global textBoxSendTFOffSetNewnQty\_2  textBoxSendTFOffSetNewnQty\_2 = self.textBoxSendTFOffSetNewnQty\_2  # textBoxSendTFOffSetNewnQty\_3  tk.Label(self, text = "互抵口數(微台)").grid(row=2, column=5)  #輸入框  self.textBoxSendTFOffSetNewnQty\_3 = tk.Entry(self)  self.textBoxSendTFOffSetNewnQty\_3.grid(row=2, column=6)  global textBoxSendTFOffSetNewnQty\_3  textBoxSendTFOffSetNewnQty\_3 = self.textBoxSendTFOffSetNewnQty\_3  # buttonSendTFOffSetNew  self.buttonSendTFOffSetNew = tk.Button(self)  self.buttonSendTFOffSetNew["text"] = "大小微台互抵"  self.buttonSendTFOffSetNew["command"] = self.buttonSendTFOffSetNew\_Click  self.buttonSendTFOffSetNew.grid(row=3, column=1)  global buttonSendTFOffSetNew  buttonSendTFOffSetNew = self.buttonSendTFOffSetNew |

|  |
| --- |
| Config.py |
| # OffSetNewForm  comboBoxSendTFOffSetNewnCommodity =(  "0:大抵微", "1:小抵微", "2:大小抵微","3:大抵小微", "4:小抵大微", "5:大抵小"  )  comboBoxSendTFOffSetNewnBuySell =(  "多方(買)", "空方(賣)"  ) |
| 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) |
| def checkBoxAsyncOrder\_CheckedChanged(self):  global bAsyncOrder  if self.var1.get() == True:  bAsyncOrder = True  else:  bAsyncOrder = False  def buttonSendTFOffSetNew\_Click(self):  if (comboBoxSendTFOffSetNewnCommodity.get() == "0:大抵微"):  nCommodity = 0  elif (comboBoxSendTFOffSetNewnCommodity.get() == "1:小抵微"):  nCommodity = 1  elif (comboBoxSendTFOffSetNewnCommodity.get() == "2:大小抵微"):  nCommodity = 2  elif (comboBoxSendTFOffSetNewnCommodity.get() == "3:大抵小微"):  nCommodity = 3  elif (comboBoxSendTFOffSetNewnCommodity.get() == "4:小抵大微"):  nCommodity = 4  elif (comboBoxSendTFOffSetNewnCommodity.get() == "5:大抵小"):  nCommodity = 5  if (comboBoxSendTFOffSetNewnBuySell.get() == "多方(買)"):  nBuySell = 0  elif (comboBoxSendTFOffSetNewnBuySell.get() == "空方(賣)"):  nBuySell = 1  nQty = int(textBoxSendTFOffSetNewnQty.get())  nQty\_2 = int(textBoxSendTFOffSetNewnQty\_2.get())  nQty\_3 = int(textBoxSendTFOffSetNewnQty\_3.get())  bstrMessage, nCode = m\_pSKOrder.SendTFOffsetNew(comboBoxUserID.get(), bAsyncOrder, comboBoxAccount.get(), nCommodity, textBoxSendTFOffSetNewbstrYearMonth.get(), nBuySell, nQty, nQty\_2, nQty\_3)  msg = "【SendTFOffsetNew】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage  richTextBoxMethodMessage.insert('end', msg + "\n")  richTextBoxMethodMessage.see('end') |

#### **(選擇權)組合/複式單拆解/雙邊了結**

##### AssembleOptions

|  |  |  |
| --- | --- | --- |
| 國內選擇權組合部位。 | | |
| **宣告** | Long AssebleOptions([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **FUTUREORDER**\* pOrder,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 可忽略，限同步委託。 |
| pOrder | SKCOM元件中的 [**FUTUREORDER**](#_組合部位、複式單拆解、雙邊部位了結▼)物件，將組合條件填入該物件後，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，表示委託已送出。  回傳值非0表示委託失敗，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。  **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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | ◆委託欄位，請參考期權下單物件說明，請留意需有可組合選擇權部位。  ★非交易的行為，無回報，可由未平倉查詢確認部位。  因組合/拆解/了結功能是直送結算所，需等處理結果回傳後，方可再進行下一筆。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別AssembleForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # comboBoxAssembleOptions          tk.Label(self, text = "類別").grid(row = 0,column = 1)          self.comboBoxAssembleOptions = ttk.Combobox(self, state='readonly')          self.comboBoxAssembleOptions['values'] = Config.comboBoxAssembleOptions          self.comboBoxAssembleOptions.grid(row=0, column=2)          global comboBoxAssembleOptions          comboBoxAssembleOptions = self.comboBoxAssembleOptions          # textBoxbstrStockNo          tk.Label(self, text = "選擇權代號1").grid(row=1, column=1)          #輸入框          self.textBoxbstrStockNo = tk.Entry(self)          self.textBoxbstrStockNo.grid(row=1, column=2)          global textBoxbstrStockNo          textBoxbstrStockNo = self.textBoxbstrStockNo          # textBoxbstrStockNo2          tk.Label(self, text = "選擇權代號2").grid(row=1, column=3)          #輸入框          self.textBoxbstrStockNo2 = tk.Entry(self)          self.textBoxbstrStockNo2.grid(row=1, column=4)          global textBoxbstrStockNo2          textBoxbstrStockNo2 = self.textBoxbstrStockNo2          # comboBoxsBuySell          tk.Label(self, text = "0:買進 1:賣出").grid(row = 2,column = 1)          self.comboBoxsBuySell = ttk.Combobox(self, state='readonly')          self.comboBoxsBuySell['values'] = Config.comboBoxsBuySell          self.comboBoxsBuySell.grid(row=2, column=2)          global comboBoxsBuySell          comboBoxsBuySell = self.comboBoxsBuySell          # comboBoxsBuySell2          tk.Label(self, text = "0:買進 1:賣出").grid(row = 2,column = 3)          self.comboBoxsBuySell2 = ttk.Combobox(self, state='readonly')          self.comboBoxsBuySell2['values'] = Config.comboBoxsBuySell2          self.comboBoxsBuySell2.grid(row=2, column=4)          global comboBoxsBuySell2          comboBoxsBuySell2 = self.comboBoxsBuySell2          # textBoxnQty          tk.Label(self, text = "交易口數").grid(row=3, column=1)          #輸入框          self.textBoxnQty = tk.Entry(self)          self.textBoxnQty.grid(row=3, column=2)          global textBoxnQty          textBoxnQty = self.textBoxnQty          # buttonAssembleOptions          self.buttonAssembleOptions = tk.Button(self)          self.buttonAssembleOptions["text"] = "選擇權部位送出"          self.buttonAssembleOptions["command"] = self.buttonAssembleOptions\_Click          self.buttonAssembleOptions.grid(row=4, column=1)          global buttonAssembleOptions          buttonAssembleOptions = self.buttonAssembleOptions |

|  |
| --- |
| Config.py |
| # AssembleForm  comboBoxAssembleOptions =(      "組合", "複式單拆解", "雙邊了結"  )  comboBoxsBuySell =(      "買進", "賣出"  )  comboBoxsBuySell2 =(      "買進", "賣出"  ) |
| 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 buttonAssembleOptions\_Click(self):          pOrder = sk.FUTUREORDER()          pOrder.bstrFullAccount = comboBoxAccount.get()          pOrder.bstrStockNo = textBoxbstrStockNo.get()          pOrder.bstrStockNo2 = textBoxbstrStockNo2.get()          if (comboBoxsBuySell.get() == "買進"):              pOrder.sBuySell = 0          elif (comboBoxsBuySell.get() == "賣出"):              pOrder.sBuySell = 1          if (comboBoxsBuySell2.get() == "買進"):              pOrder.sBuySell2 = 0          elif (comboBoxsBuySell2.get() == "賣出"):              pOrder.sBuySell2 = 1          pOrder.nQty = int(textBoxnQty.get())          if (comboBoxAssembleOptions.get() == "組合"):              # 國內選擇權組合部位              bstrMessage, nCode = m\_pSKOrder.AssembleOptions(comboBoxUserID.get(), bAsyncOrder, pOrder)              msg = "【AssembleOptions】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage              richTextBoxMethodMessage.insert('end',  msg + "\n")              richTextBoxMethodMessage.see('end')          elif (comboBoxAssembleOptions.get() == "複式單拆解"):              # 國內選擇權複式單拆解              bstrMessage, nCode = m\_pSKOrder.DisassembleOptions(comboBoxUserID.get(), bAsyncOrder, pOrder)              msg = "【DisassembleOptions】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage              richTextBoxMethodMessage.insert('end',  msg + "\n")              richTextBoxMethodMessage.see('end')          elif (comboBoxAssembleOptions.get() == "雙邊了結"):              # 國內選擇權雙邊部位了結              bstrMessage, nCode = m\_pSKOrder.CoverAllProduct(comboBoxUserID.get(), bAsyncOrder, pOrder)              msg = "【CoverAllProduct】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage              richTextBoxMethodMessage.insert('end',  msg + "\n")              richTextBoxMethodMessage.see('end') |

##### DisassembleOptions

|  |  |  |
| --- | --- | --- |
| 國內選擇權複式單拆解。 | | |
| **宣告** | Long DisassebleOptions([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **FUTUREORDER**\* pOrder,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 可忽略，限同步委託。 |
| pOrder | SKCOM元件中的 [**FUTUREORDER**](#_組合部位、複式單拆解、雙邊部位了結▼)物件，將拆解條件填入該物件後，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，表示委託已送出。  回傳值非0表示委託失敗，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。  **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。**  **其他錯誤由「交易主機」回傳錯誤代碼及錯誤原因，若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | ◆委託欄位，請參考期權下單物件說明，請留意需有複式單部位。  ★非交易的行為，無回報，可由未平倉查詢確認部位。  因組合/拆解/了結功能是直送結算所，需等處理結果回傳後，方可再進行下一筆。 | |

##### CoverAllProduct

|  |  |  |
| --- | --- | --- |
| 國內選擇權雙邊部位了結。 | | |
| **宣告** | Long CoverAllProduct([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **FUTUREORDER**\* pOrder,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 可忽略，限同步委託。 |
| pOrder | SKCOM元件中的 [**FUTUREORDER**](#_組合部位、複式單拆解、雙邊部位了結▼)物件，將了結條件填入該物件後，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，表示委託已送出。  回傳值非0表示委託失敗，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。  **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。**  **其他錯誤由「交易主機」回傳錯誤代碼及錯誤原因，若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | ◆委託欄位，請參考期權下單物件說明，請留意需具有雙邊部位。  ★非交易的行為，無回報，可由未平倉查詢確認部位。  因組合/拆解/了結功能是直送結算所，需等處理結果回傳後，方可再進行下一筆。 | |

##### AllCoverDisOptions

|  |  |  |
| --- | --- | --- |
| 全拆組單、全拆組平單下單。  依照試算結果進行委託。 | | |
| **宣告** | Long  AllCoverDisOptions ([in] BSTR bstrLogInID, [in] struct **FUTUREORDER**\* pOrder,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pOrder | SKCOM元件中的 [**FUTUREORDER**](#_全拆組單；全拆組平單委託▼)物件，將組合條件填入該物件後，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，表示委託已送出。  回傳值非0表示委託失敗，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。  **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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | ★非交易的行為，無回報，可由未平倉查詢確認部位。  因組合/拆解/了結功能是直送結算所，需等處理結果回傳後，方可再進行下一筆。 | |

#### **一般下單/刪改單**

##### SendFutureOrderCLR

|  |  |  |
| --- | --- | --- |
| 送出期貨委託，需設倉別與盤別。 | | |
| **宣告** | Long SendFutureOrderCLR([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **FUTUREORDER**\* pAsyncOrder,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| pOrder | SKCOM元件中的 [**FUTUREORDER**](#_一般期選委託▼)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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別SendForm(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)          # textBoxFutureID          tk.Label(self, text = "期貨代號").grid(row=1, column=1)              #輸入框          self.textBoxFutureID = tk.Entry(self)          self.textBoxFutureID.grid(row=1, column=2)          global textBoxFutureID          textBoxFutureID = self.textBoxFutureID          # comboBoxFutureDayTrade          tk.Label(self, text = "當沖").grid(row=2, column=1)              #輸入框          self.comboBoxFutureDayTrade = ttk.Combobox(self, state='readonly')          self.comboBoxFutureDayTrade['values'] = Config.comboBoxFutureDayTrade          self.comboBoxFutureDayTrade.grid(row=2, column=2)          global comboBoxFutureDayTrade          comboBoxFutureDayTrade = self.comboBoxFutureDayTrade          # comboBoxFutureReserved          tk.Label(self, text = "盤別").grid(row=3, column=1)              #輸入框          self.comboBoxFutureReserved = ttk.Combobox(self, state='readonly')          self.comboBoxFutureReserved['values'] = Config.comboBoxFutureReserved          self.comboBoxFutureReserved.grid(row=3, column=2)          global comboBoxFutureReserved          comboBoxFutureReserved = self.comboBoxFutureReserved          # comboBoxFutureNewClose          tk.Label(self, text = "新平倉").grid(row=4, column=1)              #輸入框          self.comboBoxFutureNewClose = ttk.Combobox(self, state='readonly')          self.comboBoxFutureNewClose['values'] = Config.comboBoxFutureNewClose          self.comboBoxFutureNewClose.grid(row=4, column=2)          global comboBoxFutureNewClose          comboBoxFutureNewClose = self.comboBoxFutureNewClose          # textBoxFutureQty          tk.Label(self, text = "口數").grid(row=5, column=1)              #輸入框          self.textBoxFutureQty = tk.Entry(self)          self.textBoxFutureQty.grid(row=5, column=2)          global textBoxFutureQty          textBoxFutureQty = self.textBoxFutureQty          # comboBoxFutureTradeType          tk.Label(self, text = "ROD/IOC/FOK").grid(row=6, column=1)              #輸入框          self.comboBoxFutureTradeType = ttk.Combobox(self, state='readonly')          self.comboBoxFutureTradeType['values'] = Config.comboBoxFutureTradeType          self.comboBoxFutureTradeType.grid(row=6, column=2)          global comboBoxFutureTradeType          comboBoxFutureTradeType = self.comboBoxFutureTradeType          # textBoxFuturePrice          tk.Label(self, text = "委託價").grid(row=7, column=1)              #輸入框          self.textBoxFuturePrice = tk.Entry(self)          self.textBoxFuturePrice.grid(row=7, column=2)          global textBoxFuturePrice          textBoxFuturePrice = self.textBoxFuturePrice          # comboBoxFutureBuySell          tk.Label(self, text = "買進/賣出").grid(row=8, column=1)              #輸入框          self.comboBoxFutureBuySell = ttk.Combobox(self, state='readonly')          self.comboBoxFutureBuySell['values'] = Config.comboBoxFutureBuySell          self.comboBoxFutureBuySell.grid(row=8, column=2)          global comboBoxFutureBuySell          comboBoxFutureBuySell = self.comboBoxFutureBuySell          # buttonSendFutureOrderCLR          self.buttonSendFutureOrderCLR = tk.Button(self)          self.buttonSendFutureOrderCLR["text"] = "期貨送出"          self.buttonSendFutureOrderCLR["command"] = self.buttonSendFutureOrderCLR\_Click          self.buttonSendFutureOrderCLR.grid(row=9, column=1) |

|  |
| --- |
| Config.py |
| # SendForm  comboBoxFutureDayTrade =(      "否", "是"  )  comboBoxFutureReserved =(      "盤中(T盤及T+1盤)", "T盤預約"  )  comboBoxFutureNewClose =(      "新倉", "平倉", "自動"  )  comboBoxFutureTradeType =(      "ROD", "IOC", "FOK"  )  comboBoxFutureBuySell =(      "買進", "賣出"  ) |
| 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 buttonSendFutureOrderCLR\_Click(self):          pOrder = sk.FUTUREORDER()          pOrder.bstrFullAccount = comboBoxAccount.get()          pOrder.bstrStockNo = textBoxFutureID.get()          if (comboBoxFutureTradeType.get() == "ROD"):              pOrder.sTradeType = 0          elif (comboBoxFutureTradeType.get() == "IOC"):              pOrder.sTradeType = 1          elif (comboBoxFutureTradeType.get() == "FOK"):              pOrder.sTradeType = 2          if (comboBoxFutureBuySell.get() == "買進"):              pOrder.sBuySell = 0          elif (comboBoxFutureBuySell.get() == "賣出"):              pOrder.sBuySell = 1          if (comboBoxFutureDayTrade.get() == "否"):              pOrder.sDayTrade = 0          elif (comboBoxFutureDayTrade.get() == "是"):              pOrder.sDayTrade = 1          if (comboBoxFutureNewClose.get() == "新倉"):              pOrder.sNewClose = 0          elif (comboBoxFutureNewClose.get() == "平倉"):              pOrder.sNewClose = 1          elif (comboBoxFutureNewClose.get() == "自動"):              pOrder.sNewClose = 2          pOrder.bstrPrice = textBoxFuturePrice.get()          pOrder.nQty = int(textBoxFutureQty.get())          if (comboBoxFutureReserved.get() == "盤中(T盤及T+1盤)"):              pOrder.sReserved = 0          elif (comboBoxFutureReserved.get() == "T盤預約"):              pOrder.sReserved = 1          bstrMessage,nCode= m\_pSKOrder.SendFutureOrderCLR(comboBoxUserID.get(), bAsyncOrder, pOrder)          msg = "【SendFutureOrderCLR】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

##### SendOptionOrder

|  |  |  |
| --- | --- | --- |
| 送出選擇權委託，需設盤別。 | | |
| **宣告** | Long SendOptionOrder([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **FUTUREORDER**\* 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 [**FUTUREORDER**](#_一般期選委託▼)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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別SendForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # checkBoxSendDuplexOrder          # 是否為複式單          self.checkBoxSendDuplexOrder = tk.Checkbutton(self)          self.var1 = tk.IntVar()          self.checkBoxSendDuplexOrder["variable"] = self.var1          self.checkBoxSendDuplexOrder["onvalue"] = True          self.checkBoxSendDuplexOrder["offvalue"] = False          self.checkBoxSendDuplexOrder["text"] = "是否為複式單"          self.checkBoxSendDuplexOrder["command"] = self.checkBoxSendDuplexOrder\_CheckedChanged          self.checkBoxSendDuplexOrder.grid( row = 0,column = 1)          # textBoxOptionID          tk.Label(self, text = "選擇權代號").grid(row=10, column=1)              #輸入框          self.textBoxOptionID = tk.Entry(self)          self.textBoxOptionID.grid(row=10, column=2)          global textBoxOptionID          textBoxOptionID = self.textBoxOptionID          # textBoxOptionID2          tk.Label(self, text = "選擇權代號2(複式單)").grid(row=11, column=1)              #輸入框          self.textBoxOptionID2 = tk.Entry(self)          self.textBoxOptionID2.grid(row=11, column=2)          global textBoxOptionID2          textBoxOptionID2 = self.textBoxOptionID2          # comboBoxOptionDayTrade          tk.Label(self, text = "當沖").grid(row=12, column=1)              #輸入框          self.comboBoxOptionDayTrade = ttk.Combobox(self, state='readonly')          self.comboBoxOptionDayTrade['values'] = Config.comboBoxOptionDayTrade          self.comboBoxOptionDayTrade.grid(row=12, column=2)          global comboBoxOptionDayTrade          comboBoxOptionDayTrade = self.comboBoxOptionDayTrade          # comboBoxOptionReserved          tk.Label(self, text = "盤別").grid(row=13, column=1)              #輸入框          self.comboBoxOptionReserved = ttk.Combobox(self, state='readonly')          self.comboBoxOptionReserved['values'] = Config.comboBoxOptionReserved          self.comboBoxOptionReserved.grid(row=13, column=2)          global comboBoxOptionReserved          comboBoxOptionReserved = self.comboBoxOptionReserved          # comboBoxOptionNewClose          tk.Label(self, text = "新平倉").grid(row=14, column=1)              #輸入框          self.comboBoxOptionNewClose = ttk.Combobox(self, state='readonly')          self.comboBoxOptionNewClose['values'] = Config.comboBoxOptionNewClose          self.comboBoxOptionNewClose.grid(row=14, column=2)          global comboBoxOptionNewClose          comboBoxOptionNewClose = self.comboBoxOptionNewClose          # textBoxOptionQty          tk.Label(self, text = "口數").grid(row=15, column=1)              #輸入框          self.textBoxOptionQty = tk.Entry(self)          self.textBoxOptionQty.grid(row=15, column=2)          global textBoxOptionQty          textBoxOptionQty = self.textBoxOptionQty          # comboBoxOptionTradeType          tk.Label(self, text = "ROD/IOC/FOK").grid(row=16, column=1)              #輸入框          self.comboBoxOptionTradeType = ttk.Combobox(self, state='readonly')          self.comboBoxOptionTradeType['values'] = Config.comboBoxOptionTradeType          self.comboBoxOptionTradeType.grid(row=16, column=2)          global comboBoxOptionTradeType          comboBoxOptionTradeType = self.comboBoxOptionTradeType          # textBoxOptionPrice          tk.Label(self, text = "委託價").grid(row=17, column=1)              #輸入框          self.textBoxOptionPrice = tk.Entry(self)          self.textBoxOptionPrice.grid(row=17, column=2)          global textBoxOptionPrice          textBoxOptionPrice = self.textBoxOptionPrice          # comboBoxOptionBuySell          tk.Label(self, text = "買進/賣出").grid(row=18, column=1)              #輸入框          self.comboBoxOptionBuySell = ttk.Combobox(self, state='readonly')          self.comboBoxOptionBuySell['values'] = Config.comboBoxOptionBuySell          self.comboBoxOptionBuySell.grid(row=18, column=2)          global comboBoxOptionBuySell          comboBoxOptionBuySell = self.comboBoxOptionBuySell          # comboBoxOptionBuySell2          tk.Label(self, text = "買進/賣出2(複式單)").grid(row=19, column=1)              #輸入框          self.comboBoxOptionBuySell2 = ttk.Combobox(self, state='readonly')          self.comboBoxOptionBuySell2['values'] = Config.comboBoxOptionBuySell2          self.comboBoxOptionBuySell2.grid(row=19, column=2)          global comboBoxOptionBuySell2          comboBoxOptionBuySell2 = self.comboBoxOptionBuySell2          # buttonSendOptionOrder          self.buttonSendOptionOrder = tk.Button(self)          self.buttonSendOptionOrder["text"] = "選擇權/複式單送出"          self.buttonSendOptionOrder["command"] = self.buttonSendOptionOrder\_Click          self.buttonSendOptionOrder.grid(row=20, column=1) |

|  |
| --- |
| Config.py |
| comboBoxOptionDayTrade =(      "否", "是"  )  comboBoxOptionReserved =(      "盤中(T盤及T+1盤)", "T盤預約"  )  comboBoxOptionNewClose =(      "新倉", "平倉", "自動"  )  comboBoxOptionTradeType =(      "ROD", "IOC", "FOK"  )  comboBoxOptionBuySell =(      "買進", "賣出"  )  comboBoxOptionBuySell2 =(      "買進", "賣出"  ) |
| 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)  # 是否為複式單  isDuplexOrder = False |

|  |
| --- |
| Current Block(放置於def createWidgets(self): 的同一層) |
| # checkBoxSendDuplexOrder      def checkBoxSendDuplexOrder\_CheckedChanged(self):          global isDuplexOrder          if self.var1.get() == True:              isDuplexOrder = True          else:              isDuplexOrder = False      def buttonSendOptionOrder\_Click(self):          pOrder = sk.FUTUREORDER()          pOrder.bstrFullAccount = comboBoxAccount.get()          pOrder.bstrStockNo = textBoxOptionID.get()          if (comboBoxOptionTradeType.get() == "ROD"):              pOrder.sTradeType = 0          elif (comboBoxOptionTradeType.get() == "IOC"):              pOrder.sTradeType = 1          elif (comboBoxOptionTradeType.get() == "FOK"):              pOrder.sTradeType = 2          if (comboBoxOptionBuySell.get() == "買進"):              pOrder.sBuySell = 0          elif (comboBoxOptionBuySell.get() == "賣出"):              pOrder.sBuySell = 1          if (comboBoxOptionDayTrade.get() == "否"):              pOrder.sDayTrade = 0          elif (comboBoxOptionDayTrade.get() == "是"):              pOrder.sDayTrade = 1          if (comboBoxOptionNewClose.get() == "新倉"):              pOrder.sNewClose = 0          elif (comboBoxOptionNewClose.get() == "平倉"):              pOrder.sNewClose = 1          elif (comboBoxOptionNewClose.get() == "自動"):              pOrder.sNewClose = 2          pOrder.bstrPrice = textBoxOptionPrice.get()          pOrder.nQty = int(textBoxOptionQty.get())          if (comboBoxOptionReserved.get() == "盤中(T盤及T+1盤)"):              pOrder.sReserved = 0          elif (comboBoxOptionReserved.get() == "T盤預約"):              pOrder.sReserved = 1          if (isDuplexOrder != True):              # 送出選擇權委託              bstrMessage,nCode= m\_pSKOrder.SendOptionOrder(comboBoxUserID.get(), bAsyncOrder, pOrder)              msg = "【SendOptionOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage              richTextBoxMethodMessage.insert('end',  msg + "\n")              richTextBoxMethodMessage.see('end')          else:              pOrder.bstrStockNo2 = textBoxOptionID2.get()              if (comboBoxOptionBuySell2.get() == "買進"):                  pOrder.sBuySell2 = 0              elif (comboBoxOptionBuySell2.get() == "賣出"):                  pOrder.sBuySell2 = 1              # 送出國內選擇權複式單委託              bstrMessage,nCode= m\_pSKOrder.SendDuplexOrder(comboBoxUserID.get(), bAsyncOrder, pOrder)              msg = "【SendDuplexOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage              richTextBoxMethodMessage.insert('end',  msg + "\n")              richTextBoxMethodMessage.see('end') |

###### SendDuplexOrder

|  |  |  |
| --- | --- | --- |
| 送出國內選擇權複式單委託。 | | |
| **宣告** | Long SendDuplexOrder([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **FUTUREORDER**\* 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 [**FUTUREORDER**](#_選擇權複式單▼)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。**  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。    **複式單下法** (FUTUREORDER 物件填法，以上圖為例):  Call多頭價差(策略王上溫漲)，買低履約價+賣高履約價  第一隻腳: bstrStockNo 填:高履約價(ex:11000)，sBuySell填:賣  第二隻腳: bstrStockNo2 填:低履約價(ex:10900)，sBuySell2填:買  bstrPrice填法基本上為:bstrStockNo2賣價(99)–bstrStockNo買價(46.5)= 52.5  如果bstrPrice價格填的越大，越能立即成交，假設目前價差52,5，超過52.5那就能幾乎能立即成交(快速直覺:Call作多價格越高買就相當於市價買進，越容易成交)。  Call空頭價差(策略王上溫跌作莊)  第一隻腳: bstrStockNo 填:高履約價(ex:11000)，sBuySell填:買  第二隻腳: bstrStockNo2 填:低履約價(ex:10900)，sBuySell2填:賣  bstrPrice填法基本上為:bstrStockNo2買價(97)–bstrStockNo賣價(47)= 50  如果bstrPrice價格填的越小，越能立即成交，假設目前價差50，小於50那就能幾乎能立即成交(快速直覺:Call作空價格越低賣就相當於賤價賣，越容易成交)。  Put多頭價差(策略王上溫漲作莊)  第一隻腳: bstrStockNo 填:低履約價(ex:10900)，sBuySell填:買  第二隻腳: bstrStockNo2 填:高履約價(ex:11000)，sBuySell2填:賣  bstrPrice填法基本上為:bstrStockNo2買價(109)–bstrStockNo賣價(64)= 45  如果bstrPrice價格填的越小，越能立即成交，假設目前價差45，小於45那就能幾乎能立即成交(快速直覺:Put作多價格越低作莊就相當於賤價賣，越容易成交)。  Put空頭價差(策略王上溫跌)  第一隻腳: bstrStockNo 填:低履約價(ex:10900)，sBuySell填:賣  第二隻腳: bstrStockNo2 填:高履約價(ex:11000)，sBuySell2填:買  bstrPrice填法基本上為:bstrStockNo2賣價(113)–bstrStockNo買價(62)= 51  如果bstrPrice價格填的越大，越能立即成交，假設目前價差51，大於51那就能幾乎能立即成交(快速直覺:Put作多價格越高買就相當於市價買，越容易成交)。  突破  第一隻腳: bstrStockNo填:call/put階可(不同履約價也可)，sBuySell填:買  第二隻腳: bstrStockNo2填: call/put階可(不同履約價也可)，sBuySell2填:買  bstrPrice填法基本上為:bstrStockNo2賣價(99)+bstrStockNo賣價(113)= 212  如果bstrPrice價格填的越大，越能立即成交，假設目前價差212，大於212那就能幾乎能立即成交。  盤整  第一隻腳: bstrStockNo填: call/put階可(不同履約價也可)，sBuySell填:賣  第二隻腳: bstrStockNo2填: call/put階可(不同履約價也可)，sBuySell2填:賣  bstrPrice填法基本上為:bstrStockNo2買價(97)+bstrStockNo買價(109)= 206  如果bstrPrice價格填的越小，越能立即成交，假設目前價差206，小於206那就能幾乎能立即成交。 | |

##### 刪改單

###### **CorrectPriceBySeqNo**

|  |  |  |
| --- | --- | --- |
| 證期權依序號改價。 | | |
| **宣告** | Long CorrectPriceBySeqNo([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrSeqNo, [in] BSTR bstrPrice, [in] LONG nTradeType,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 證券：委託帳號 (BrokerID＋帳號)。  期貨：委託帳號 ( IB＋帳號) 。 |
| bstrSeqNo | 欲改量的委託序號。 |
| bstrPrice | 修改價格。 |
| nTradeType | **證券0:ROD**  期選0:ROD 1:IOC 2:FOK |
| bstrMessage | **同步委託：**如果回傳值為 0表示委託成功，訊息內容則為修改訊息。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](#_OnAsyncOrder)。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 回傳值0 表示委託伺服器接收成功，詳細委託狀態仍須以委託**改價**回報內容為主。  其他非 0 數值表示異常，詳細原因請參考 Log 內容說明。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  ＊證券逐筆nTradeType:請設0(ROD).  ＊不適用盤中零股 | |

程式作法同：

文件「**5.下單-國內證券** > 一般下單/刪改單 > 3. CorrectPriceBySeqNo 到 8. CancelOrderByStockNo」

###### **CorrectPriceByBookNo**

|  |  |  |
| --- | --- | --- |
| 證期權依書號改價。 | | |
| **宣告** | Long CorrectPriceByBookNo([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrMarketSymbol, [in] BSTR bstrBookNo, [in] BSTR bstrPrice, [in] LONG nTradeType,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 證券：委託帳號 (BrokerID＋帳號)。  期貨：委託帳號 ( IB＋帳號) 。 |
| bstrMarketSymbol | 市場類別：TS: 證券TF:期貨TO:選擇權 |
| bstrBookNo | 欲改量的委託書號。 |
| bstrPrice | 修改價格。 |
| nTradeType | **證券0:ROD**  期選0:ROD 1:IOC 2:FOK |
| bstrMessage | **同步委託：**如果回傳值為 0表示委託成功，訊息內容則為修改訊息。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](#_OnAsyncOrder)。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 回傳值0 表示委託伺服器接收成功，詳細委託狀態仍須以委託**改價**回報內容為主。  其他非 0 數值表示異常，詳細原因請參考 Log 內容說明。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  ＊證券逐筆nTradeType:請設0(ROD).  ＊不適用盤中零股 | |

###### **DecreaseOrderBySeqNo**

|  |  |  |
| --- | --- | --- |
| 委託減量。 | | |
| **宣告** | Long DecreaseOrderBySeqNo([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrSeqNo, [in] LONG nDecreaseQty,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 證券：委託帳號 (BrokerID＋帳號)。  期貨：委託帳號 ( IB＋帳號) 。 |
| bstrSeqNo | 欲改量的委託序號。 |
| nDecreaseQty | 欲減少的數量。 |
| bstrMessage | **同步委託：**如果回傳值為 0表示委託成功，訊息內容則為修改訊息。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](#_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。  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 回傳值0 表示委託伺服器**接收成功**，詳細委託狀態仍須以委託**減量**回報內容為主。  其他非 0 數值表示異常，詳細原因請參考 Log 內容說明。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  ＊亦適用證券逐筆及擬真平台 | |

###### **CancelOrderBySeqNo**

|  |  |  |
| --- | --- | --- |
| 國內委託删單(By委託序號)。 | | |
| **宣告** | Long CancelOrderBySeqNo([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrSeqNo,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 證券：委託帳號 (BrokerID＋帳號)。  期貨：委託帳號 ( IB＋帳號) 。 |
| bstrSeqNo | 欲刪除的委託序號。 |
| bstrMessage | **同步刪單：**如果回傳值為 0表示刪單成功，訊息內容則為刪單訊息。回傳值非0表示刪單失敗，訊息內容為失敗原因。  **非同步刪單：**參照[OnAsyncOrder](#_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。  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 回傳值0 表示委託伺服器**接收成功**，詳細委託狀態仍須以委託**刪單**回報內容為主。  其他非 0 數值表示異常，詳細原因請參考 Log 內容說明。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  ＊亦適用證券逐筆及擬真平台 | |

###### **CancelOrderByBookNo**

|  |  |  |
| --- | --- | --- |
| 國內委託删單(By書號)。 | | |
| **宣告** | Long CancelOrderByBookNo([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrBookNo,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 證券：委託帳號 (BrokerID＋帳號)。  期貨：委託帳號 ( IB＋帳號) 。 |
| bstrBookNo | 欲刪除的委託書號。 |
| bstrMessage | **同步刪單：**如果回傳值為 0表示刪單成功，訊息內容則為刪單訊息。回傳值非0表示刪單失敗，訊息內容為失敗原因。  **非同步刪單：**參照OnAsyncOrder。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 回傳值0 表示委託伺服器接收成功，詳細委託狀態仍須以委託回報內容為主。  其他非 0 數值表示異常，詳細原因請參考 Log 內容說明。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。 | |

###### **CancelOrderByStockNo**

|  |  |  |
| --- | --- | --- |
| 國內委託删單(By帳號所屬**登入ID**+商品代號)。 | | |
| **宣告** | Long CancelOrderByStockNo([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrStockNo,[**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 證券：委託帳號 (BrokerID＋帳號)。  期貨：委託帳號 ( IB＋帳號) 。 |
| bstrStockNo | 欲刪除的委託商品代號，請輸入有效商品代碼。 |
| bstrMessage | **同步刪單：**如果回傳值為 0表示刪單成功，訊息內容則為刪單訊息。回傳值非0表示刪單失敗，訊息內容為失敗原因。  **非同步刪單：**參照[OnAsyncOrder](#_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。  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 回傳值0 表示委託伺服器**接收成功**，詳細委託狀態仍須以委託**刪單**回報內容為主。  其他非 0 數值表示異常，詳細原因請參考 Log 內容說明。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  ＊請留意證券代碼，包含刪除盤中零股委託 | |

|  |  |  |
| --- | --- | --- |
| 異動前v2.13.45 | 欄位 | 異動後V2.13.52 |
| 欲刪除的委託商品代號，請輸入有效商品代碼。 | bstrStockNo  欄位說明：  帶空字串，可刪除帳號下所有委託。 | 商品代碼為空，則為刪除全部委託。 |

###### **CancelOrderByStockNoAdvance**

|  |  |  |
| --- | --- | --- |
| 國內委託删單(By帳號所屬**登入ID**+商品代號+買賣別+委託價格)。 | | |
| **宣告** | Long CancelOrderByStockNoAdvance([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrStockNo, [in] LONG nBuySell, [in] BSTR bstrPrice, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 證券：委託帳號 (BrokerID＋帳號)。  期貨：委託帳號 ( IB＋帳號) 。 |
| bstrStockNo | 欲刪除的委託商品代號，請輸入有效商品代碼。 |
| nBuySell | 0:買進 1:賣出 |
| bstrPrice | 委託價格(不使用請帶空字串) |
| bstrMessage | **同步刪單：**如果回傳值為 0表示刪單成功，訊息內容則為刪單訊息。回傳值非0表示刪單失敗，訊息內容為失敗原因。  **非同步刪單：**參照[OnAsyncOrder](#_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。  **其他錯誤則由「交易主機」回傳錯誤代碼及錯誤原因。若仍有疑問請洽詢您所屬營業員。** | |
| **備註** | 回傳值0 表示委託伺服器**接收成功**，詳細委託狀態仍須以委託**刪單**回報內容為主。  其他非 0 數值表示異常，詳細原因請參考 Log 內容說明。  使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder)取得。  ＊請留意證券代碼，包含刪除盤中零股委託 | |

|  |
| --- |
| Python範例程式碼\_TSOrder.py類別UpdateForm(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, 軟體, 數字, 字型 的圖片  自動產生的描述  其它部分可參考[3. CorrectPriceBySeqNo](#_CorrectPriceBySeqNo_1)做法 |
| # buttonCancelOrderByStockNo          self.buttonCancelOrderByStockNo = tk.Button(self)          self.buttonCancelOrderByStockNo["text"] = "刪單(代號)"          self.buttonCancelOrderByStockNo["command"] = self.buttonCancelOrderByStockNo\_Click          self.buttonCancelOrderByStockNo.grid(row=3, column=3)  # comboBoxCancelOrderByStockNoAdvanceBuySell  tk.Label(self, text = "買/賣/無").grid(row=2, column=3)  #輸入框  self.comboBoxCancelOrderByStockNoAdvanceBuySell = ttk.Combobox(self, state='readonly')  self.comboBoxCancelOrderByStockNoAdvanceBuySell['values'] = Config.comboBoxCancelOrderByStockNoAdvanceBuySell  self.comboBoxCancelOrderByStockNoAdvanceBuySell.grid(row=2, column=4)  # textBoxCancelOrderByStockNoAdvancePrice  tk.Label(self, text = "委託價格").grid(row=2, column=5)  #輸入框  self.textBoxCancelOrderByStockNoAdvancePrice = tk.Entry(self)  self.textBoxCancelOrderByStockNoAdvancePrice.grid(row=2, column=6)  global textBoxCancelOrderByStockNoAdvancePrice  textBoxCancelOrderByStockNoAdvancePrice = self.textBoxCancelOrderByStockNoAdvancePrice |

|  |
| --- |
| Config.py |
| comboBoxCancelOrderByStockNoAdvanceBuySell =(     "買", "賣", "無"  ) |
| 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) |
| def buttonCancelOrderByStockNo\_Click(self):  if comboBoxCancelOrderByStockNoAdvanceBuySell.get() == "無":  bstrMessage,nCode= m\_pSKOrder.CancelOrderByStockNo(comboBoxUserID.get(), bAsyncOrder, comboBoxAccount.get(), textBoxCancelOrderByStockNo.get())  msg = "【CancelOrderByStockNo】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage  elif comboBoxCancelOrderByStockNoAdvanceBuySell.get() == "買":  nBuySell = 0  bstrMessage,nCode= m\_pSKOrder.CancelOrderByStockNoAdvance(comboBoxUserID.get(), bAsyncOrder, comboBoxAccount.get(), textBoxCancelOrderByStockNo.get(), nBuySell, textBoxCancelOrderByStockNoAdvancePrice.get())  msg = "【CancelOrderByStockNoAdvance】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage  else:  nBuySell = 1  bstrMessage,nCode= m\_pSKOrder.CancelOrderByStockNoAdvance(comboBoxUserID.get(), bAsyncOrder, comboBoxAccount.get(), textBoxCancelOrderByStockNo.get(), nBuySell, textBoxCancelOrderByStockNoAdvancePrice.get())  msg = "【CancelOrderByStockNoAdvance】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage  richTextBoxMethodMessage.insert('end', msg + "\n")  richTextBoxMethodMessage.see('end') |

#### **Proxy下單/刪改單**

##### SendFutureProxyOrderCLR

|  |  |  |
| --- | --- | --- |
| 經由proxy server送出期貨委託，需設倉別與盤別。 | | |
| **宣告** | Long SendFutureProxyOrderCLR ([in]BSTR bstrLogInID, [in]struct FUTUREPROXYORDER\* pFUTUREPROXYORDER, [out] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pFUTUREPROXYORDER | SKCOM元件中的[FUTUREPROXYORDER](#_Proxy_Server期貨委託)物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **委託送出成功：ORKEY**  **委託送出失敗：錯誤訊息** |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。錯誤代碼可參考對照表。 | |
| **備註** | 透過Proxy Server委託，委託結果請由[OnProxyOrder](#_OnProxyOrder)取得。  連線且成功登入，通知OnProxyStatus為5001時，送至proxy server進行下單。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別SendProxyForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # textBoxFutureID3          tk.Label(self, text = "期貨代號").grid(row=1, column=1)              #輸入框          self.textBoxFutureID3 = tk.Entry(self)          self.textBoxFutureID3.grid(row=1, column=2)          global textBoxFutureID3          textBoxFutureID3 = self.textBoxFutureID3          # textBoxTFbstrSettleYM3          tk.Label(self, text = "契約年月").grid(row=2, column=1)              #輸入框          self.textBoxTFbstrSettleYM3 = tk.Entry(self)          self.textBoxTFbstrSettleYM3.grid(row=2, column=2)          global textBoxTFbstrSettleYM3          textBoxTFbstrSettleYM3 = self.textBoxTFbstrSettleYM3          # comboBoxTFnPriceFlag3          tk.Label(self, text = "市價/限價/範圍市價").grid(row=3, column=1)              #輸入框          self.comboBoxTFnPriceFlag3 = ttk.Combobox(self, state='readonly')          self.comboBoxTFnPriceFlag3['values'] = Config.comboBoxTFnPriceFlag3          self.comboBoxTFnPriceFlag3.grid(row=3, column=2)          global comboBoxTFnPriceFlag3          comboBoxTFnPriceFlag3 = self.comboBoxTFnPriceFlag3          # comboBoxFutureDayTrade3          tk.Label(self, text = "當沖").grid(row=4, column=1)              #輸入框          self.comboBoxFutureDayTrade3 = ttk.Combobox(self, state='readonly')          self.comboBoxFutureDayTrade3['values'] = Config.comboBoxFutureDayTrade3          self.comboBoxFutureDayTrade3.grid(row=4, column=2)          global comboBoxFutureDayTrade3          comboBoxFutureDayTrade3 = self.comboBoxFutureDayTrade3          # comboBoxFutureReserved3          tk.Label(self, text = "盤別").grid(row=5, column=1)              #輸入框          self.comboBoxFutureReserved3 = ttk.Combobox(self, state='readonly')          self.comboBoxFutureReserved3['values'] = Config.comboBoxFutureReserved3          self.comboBoxFutureReserved3.grid(row=5, column=2)          global comboBoxFutureReserved3          comboBoxFutureReserved3 = self.comboBoxFutureReserved3            # comboBoxFutureNewClose3          tk.Label(self, text = "新平倉").grid(row=6, column=1)              #輸入框          self.comboBoxFutureNewClose3 = ttk.Combobox(self, state='readonly')          self.comboBoxFutureNewClose3['values'] = Config.comboBoxFutureNewClose3          self.comboBoxFutureNewClose3.grid(row=6, column=2)          global comboBoxFutureNewClose3          comboBoxFutureNewClose3 = self.comboBoxFutureNewClose3          # textBoxFutureQty3          tk.Label(self, text = "口數").grid(row=7, column=1)              #輸入框          self.textBoxFutureQty3 = tk.Entry(self)          self.textBoxFutureQty3.grid(row=7, column=2)          global textBoxFutureQty3          textBoxFutureQty3 = self.textBoxFutureQty3          # comboBoxFutureTradeType3          tk.Label(self, text = "ROD/IOC/FOK").grid(row=8, column=1)              #輸入框          self.comboBoxFutureTradeType3 = ttk.Combobox(self, state='readonly')          self.comboBoxFutureTradeType3['values'] = Config.comboBoxFutureTradeType3          self.comboBoxFutureTradeType3.grid(row=8, column=2)          global comboBoxFutureTradeType3          comboBoxFutureTradeType3 = self.comboBoxFutureTradeType3          # textBoxFuturePrice3          tk.Label(self, text = "委託價").grid(row=9, column=1)              #輸入框          self.textBoxFuturePrice3 = tk.Entry(self)          self.textBoxFuturePrice3.grid(row=9, column=2)          global textBoxFuturePrice3          textBoxFuturePrice3 = self.textBoxFuturePrice3          # comboBoxFutureBuySell3          tk.Label(self, text = "買進/賣出").grid(row=10, column=1)              #輸入框          self.comboBoxFutureBuySell3 = ttk.Combobox(self, state='readonly')          self.comboBoxFutureBuySell3['values'] = Config.comboBoxFutureBuySell3          self.comboBoxFutureBuySell3.grid(row=10, column=2)          global comboBoxFutureBuySell3          comboBoxFutureBuySell3 = self.comboBoxFutureBuySell3          # buttonSendFutureProxyOrderCLR          self.buttonSendFutureProxyOrderCLR = tk.Button(self)          self.buttonSendFutureProxyOrderCLR["text"] = "期貨送出"          self.buttonSendFutureProxyOrderCLR["command"] = self.buttonSendFutureProxyOrderCLR\_Click          self.buttonSendFutureProxyOrderCLR.grid(row=11, column=1) |

|  |
| --- |
| Config.py |
| # SendProxyForm  comboBoxTFnPriceFlag3 =(      "市價", "限價", "範圍市價"  )  comboBoxFutureDayTrade3 =(      "否", "是"  )  comboBoxFutureReserved3 =(      "盤中單", "預約單"  )  comboBoxFutureNewClose3 =(      "0:新倉", "1:平倉", "2:自動"  )  comboBoxFutureTradeType3 =(      "ROD", "IOC", "FOK"  )  comboBoxFutureBuySell3 =(      "買進", "賣出"  ) |
| 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 buttonSendFutureProxyOrderCLR\_Click(self):          pFUTUREPROXYORDER = sk.FUTUREPROXYORDER()          pFUTUREPROXYORDER.bstrFullAccount = comboBoxAccount.get()          pFUTUREPROXYORDER.bstrStockNo = textBoxFutureID3.get()          pFUTUREPROXYORDER.bstrSettleYM = textBoxTFbstrSettleYM3.get()          if (comboBoxTFnPriceFlag3.get() == "市價"):              pFUTUREPROXYORDER.nPriceFlag = 0          elif (comboBoxTFnPriceFlag3.get() == "限價"):              pFUTUREPROXYORDER.nPriceFlag = 1          elif (comboBoxTFnPriceFlag3.get() == "範圍市價"):              pFUTUREPROXYORDER.nPriceFlag = 2          if (comboBoxFutureTradeType3.get() == "ROD"):              pFUTUREPROXYORDER.nTradeType = 0          elif (comboBoxFutureTradeType3.get() == "IOC"):              pFUTUREPROXYORDER.nTradeType = 1          elif (comboBoxFutureTradeType3.get() == "FOK"):              pFUTUREPROXYORDER.nTradeType = 2          if (comboBoxFutureBuySell3.get() == "買進"):              pFUTUREPROXYORDER.nBuySell = 0          elif (comboBoxFutureBuySell3.get() == "賣出"):              pFUTUREPROXYORDER.nBuySell = 1          if (comboBoxFutureDayTrade3.get() == "否"):              pFUTUREPROXYORDER.nDayTrade = 0          elif (comboBoxFutureDayTrade3.get() == "是"):              pFUTUREPROXYORDER.nDayTrade = 1          if (comboBoxFutureNewClose3.get() == "0:新倉"):              pFUTUREPROXYORDER.bstrOrderType = "0"          elif (comboBoxFutureNewClose3.get() == "1:平倉"):              pFUTUREPROXYORDER.bstrOrderType = "1"          elif (comboBoxFutureNewClose3.get() == "2:自動"):              pFUTUREPROXYORDER.bstrOrderType = "2"          pFUTUREPROXYORDER.bstrPrice = textBoxFuturePrice3.get()          pFUTUREPROXYORDER.nQty = int(textBoxFutureQty3.get())          if (comboBoxFutureReserved3.get() == "盤中單"):              pFUTUREPROXYORDER.nReserved = 0          elif (comboBoxFutureReserved3.get() == "預約單"):              pFUTUREPROXYORDER.nReserved = 1          # 經由proxy server送出期貨委託，需設倉別與盤別          bstrMessage,nCode= m\_pSKOrder.SendFutureProxyOrderCLR(comboBoxUserID.get(), pFUTUREPROXYORDER)          if bstrMessage is not None:               msg = "【SendFutureProxyOrderCLR】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          else:               msg = "【SendFutureProxyOrderCLR】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + " No message"            richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

##### SendOptionProxyOrder

|  |  |  |
| --- | --- | --- |
| 經由proxy server送出選擇權委託 | | |
| **宣告** | Long SendOptionProxyOrder([in]BSTR bstrLogInID, [in]struct FUTUREPROXYORDER\* pFUTUREPROXYORDER, [out] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pFUTUREPROXYORDER | SKCOM元件中的[FUTUREPROXYORDER](#_Proxy_Server選擇權下單_1)物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **委託送出成功：ORKEY**  **委託送出失敗：錯誤訊息** |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。錯誤代碼可參考對照表。 | |
| **備註** | 透過Proxy Server委託，委託結果請由[OnProxyOrder](#_OnProxyOrder)取得。  連線且成功登入，通知OnProxyStatus為5001時，送至proxy server進行下單。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別SendProxyForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 |
| # checkBoxSendDuplexOrder          # 是否為複式單          self.checkBoxSendDuplexOrder = tk.Checkbutton(self)          self.var1 = tk.IntVar()          self.checkBoxSendDuplexOrder["variable"] = self.var1          self.checkBoxSendDuplexOrder["onvalue"] = True          self.checkBoxSendDuplexOrder["offvalue"] = False          self.checkBoxSendDuplexOrder["text"] = "是否為複式單"          self.checkBoxSendDuplexOrder["command"] = self.checkBoxSendDuplexOrder\_CheckedChanged          self.checkBoxSendDuplexOrder.grid( row = 0,column = 1)          # textBoxOptionID3          tk.Label(self, text = "選擇權代號").grid(row=12, column=1)              #輸入框          self.textBoxOptionID3 = tk.Entry(self)          self.textBoxOptionID3.grid(row=12, column=2)          global textBoxOptionID3          textBoxOptionID3 = self.textBoxOptionID3          # textBoxOptionID23          tk.Label(self, text = "選擇權代號2(複式單)").grid(row=13, column=1)              #輸入框          self.textBoxOptionID23 = tk.Entry(self)          self.textBoxOptionID23.grid(row=13, column=2)          global textBoxOptionID23          textBoxOptionID23 = self.textBoxOptionID23          # textBoxTObstrSettleYM3          tk.Label(self, text = "契約年月").grid(row=14, column=1)              #輸入框          self.textBoxTObstrSettleYM3 = tk.Entry(self)          self.textBoxTObstrSettleYM3.grid(row=14, column=2)          global textBoxTObstrSettleYM3          textBoxTObstrSettleYM3 = self.textBoxTObstrSettleYM3          # textBoxTObstrSettleYM23          tk.Label(self, text = "契約年月2").grid(row=15, column=1)              #輸入框          self.textBoxTObstrSettleYM23 = tk.Entry(self)          self.textBoxTObstrSettleYM23.grid(row=15, column=2)          global textBoxTObstrSettleYM23          textBoxTObstrSettleYM23 = self.textBoxTObstrSettleYM23          # textBoxbstrStrike3          tk.Label(self, text = "履約價1").grid(row=16, column=1)              #輸入框          self.textBoxbstrStrike3 = tk.Entry(self)          self.textBoxbstrStrike3.grid(row=16, column=2)          global textBoxbstrStrike3          textBoxbstrStrike3 = self.textBoxbstrStrike3          # textBoxbstrStrike23          tk.Label(self, text = "履約價2").grid(row=17, column=1)              #輸入框          self.textBoxbstrStrike23 = tk.Entry(self)          self.textBoxbstrStrike23.grid(row=17, column=2)          global textBoxbstrStrike23          textBoxbstrStrike23 = self.textBoxbstrStrike23          # comboBoxOptionDayTrade3          tk.Label(self, text = "當沖").grid(row=18, column=1)              #輸入框          self.comboBoxOptionDayTrade3 = ttk.Combobox(self, state='readonly')          self.comboBoxOptionDayTrade3['values'] = Config.comboBoxOptionDayTrade3          self.comboBoxOptionDayTrade3.grid(row=18, column=2)          global comboBoxOptionDayTrade3          comboBoxOptionDayTrade3 = self.comboBoxOptionDayTrade3          # comboBoxOptionReserved3          tk.Label(self, text = "盤別").grid(row=19, column=1)              #輸入框          self.comboBoxOptionReserved3 = ttk.Combobox(self, state='readonly')          self.comboBoxOptionReserved3['values'] = Config.comboBoxOptionReserved3          self.comboBoxOptionReserved3.grid(row=19, column=2)          global comboBoxOptionReserved3          comboBoxOptionReserved3 = self.comboBoxOptionReserved3          # comboBoxOptionbstrOrderType3          tk.Label(self, text = "新平倉").grid(row=20, column=1)              #輸入框          self.comboBoxOptionbstrOrderType3 = ttk.Combobox(self, state='readonly')          self.comboBoxOptionbstrOrderType3['values'] = Config.comboBoxOptionbstrOrderType3          self.comboBoxOptionbstrOrderType3.grid(row=20, column=2)          global comboBoxOptionbstrOrderType3          comboBoxOptionbstrOrderType3 = self.comboBoxOptionbstrOrderType3          # textBoxOptionQty3          tk.Label(self, text = "口數").grid(row=21, column=1)              #輸入框          self.textBoxOptionQty3 = tk.Entry(self)          self.textBoxOptionQty3.grid(row=21, column=2)          global textBoxOptionQty3          textBoxOptionQty3 = self.textBoxOptionQty3          # comboBoxOptionTradeType3          tk.Label(self, text = "ROD/IOC/FOK").grid(row=22, column=1)              #輸入框          self.comboBoxOptionTradeType3 = ttk.Combobox(self, state='readonly')          self.comboBoxOptionTradeType3['values'] = Config.comboBoxOptionTradeType3          self.comboBoxOptionTradeType3.grid(row=22, column=2)          global comboBoxOptionTradeType3          comboBoxOptionTradeType3 = self.comboBoxOptionTradeType3          # comboBoxnCP3          tk.Label(self, text = "CALL/PUT").grid(row=23, column=1)              #輸入框          self.comboBoxnCP3 = ttk.Combobox(self, state='readonly')          self.comboBoxnCP3['values'] = Config.comboBoxnCP3          self.comboBoxnCP3.grid(row=23, column=2)          global comboBoxnCP3          comboBoxnCP3 = self.comboBoxnCP3          # comboBoxnCP23          tk.Label(self, text = "CALL/PUT2").grid(row=24, column=1)              #輸入框          self.comboBoxnCP23 = ttk.Combobox(self, state='readonly')          self.comboBoxnCP23['values'] = Config.comboBoxnCP23          self.comboBoxnCP23.grid(row=24, column=2)          global comboBoxnCP23          comboBoxnCP23 = self.comboBoxnCP23          # comboBoxTOnPriceFlag3          tk.Label(self, text = "市價/限價/範圍市價").grid(row=25, column=1)              #輸入框          self.comboBoxTOnPriceFlag3 = ttk.Combobox(self, state='readonly')          self.comboBoxTOnPriceFlag3['values'] = Config.comboBoxTOnPriceFlag3          self.comboBoxTOnPriceFlag3.grid(row=25, column=2)          global comboBoxTOnPriceFlag3          comboBoxTOnPriceFlag3 = self.comboBoxTOnPriceFlag3          # textBoxOptionPrice3          tk.Label(self, text = "委託價").grid(row=26, column=1)              #輸入框          self.textBoxOptionPrice3 = tk.Entry(self)          self.textBoxOptionPrice3.grid(row=26, column=2)          global textBoxOptionPrice3          textBoxOptionPrice3 = self.textBoxOptionPrice3          # comboBoxOptionBuySell3          tk.Label(self, text = "買進/賣出").grid(row=27, column=1)              #輸入框          self.comboBoxOptionBuySell3 = ttk.Combobox(self, state='readonly')          self.comboBoxOptionBuySell3['values'] = Config.comboBoxOptionBuySell3          self.comboBoxOptionBuySell3.grid(row=27, column=2)            global comboBoxOptionBuySell3          comboBoxOptionBuySell3 = self.comboBoxOptionBuySell3          # comboBoxOptionBuySell23          tk.Label(self, text = "買進/賣出2").grid(row=28, column=1)              #輸入框          self.comboBoxOptionBuySell23 = ttk.Combobox(self, state='readonly')          self.comboBoxOptionBuySell23['values'] = Config.comboBoxOptionBuySell23          self.comboBoxOptionBuySell23.grid(row=28, column=2)          global comboBoxOptionBuySell23          comboBoxOptionBuySell23 = self.comboBoxOptionBuySell23          # buttonSendOptionProxyOrder          self.buttonSendOptionProxyOrder = tk.Button(self)          self.buttonSendOptionProxyOrder["text"] = "選擇權/複式單送出"          self.buttonSendOptionProxyOrder["command"] = self.buttonSendOptionProxyOrder\_Click          self.buttonSendOptionProxyOrder.grid(row=29, column=1) |

|  |
| --- |
| Config.py |
| comboBoxOptionDayTrade3 =(      "否", "是"  )  comboBoxOptionReserved3 =(      "盤中單", "預約單"  )  comboBoxOptionbstrOrderType3 =(      "0:新倉", "1:平倉", "2:自動"  )  comboBoxOptionTradeType3 =(      "ROD", "IOC", "FOK"  )  comboBoxnCP3 =(      "CALL", "PUT"  )  comboBoxnCP23 =(      "CALL", "PUT"  )  comboBoxTOnPriceFlag3 =(      "市價", "限價", "範圍市價"  )  comboBoxOptionBuySell3 =(      "買進", "賣出"  )  comboBoxOptionBuySell23 =(      "買進", "賣出"  ) |
| 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)  # 是否為複式單  isDuplexOrder = False |

|  |
| --- |
| Current Block(放置於def createWidgets(self): 的同一層) |
| # checkBoxSendDuplexOrder      def checkBoxSendDuplexOrder\_CheckedChanged(self):          global isDuplexOrder          if self.var1.get() == True:              isDuplexOrder = True          else:              isDuplexOrder = False      def buttonSendOptionProxyOrder\_Click(self):          pFUTUREPROXYORDER = sk.FUTUREPROXYORDER()          pFUTUREPROXYORDER.bstrFullAccount = comboBoxAccount.get()          pFUTUREPROXYORDER.bstrStockNo = textBoxOptionID3.get()          pFUTUREPROXYORDER.bstrSettleYM = textBoxTObstrSettleYM3.get()          pFUTUREPROXYORDER.bstrStrike = textBoxbstrStrike3.get()          if (comboBoxOptionTradeType3.get() == "ROD"):              pFUTUREPROXYORDER.nTradeType = 0          elif (comboBoxOptionTradeType3.get() == "IOC"):              pFUTUREPROXYORDER.nTradeType = 1          elif (comboBoxOptionTradeType3.get() == "FOK"):              pFUTUREPROXYORDER.nTradeType = 2          if (comboBoxOptionBuySell3.get() == "買進"):              pFUTUREPROXYORDER.nBuySell = 0          elif (comboBoxOptionBuySell3.get() == "賣出"):              pFUTUREPROXYORDER.nBuySell = 1          if (comboBoxOptionDayTrade3.get() == "否"):              pFUTUREPROXYORDER.nDayTrade = 0          elif (comboBoxOptionDayTrade3.get() == "是"):              pFUTUREPROXYORDER.nDayTrade = 1          if (comboBoxOptionbstrOrderType3.get() == "0:新倉"):              pFUTUREPROXYORDER.bstrOrderType = "0"          elif (comboBoxOptionbstrOrderType3.get() == "1:平倉"):              pFUTUREPROXYORDER.bstrOrderType = "1"          elif (comboBoxOptionbstrOrderType3.get() == "2:自動"):              pFUTUREPROXYORDER.bstrOrderType = "2"          pFUTUREPROXYORDER.bstrPrice = textBoxOptionPrice3.get()          pFUTUREPROXYORDER.nQty = int(textBoxOptionQty3.get())          if (comboBoxOptionReserved3.get() == "盤中單"):              pFUTUREPROXYORDER.nReserved = 0          elif (comboBoxOptionReserved3.get() == "預約單"):              pFUTUREPROXYORDER.nReserved = 1          if (comboBoxnCP3.get() == "CALL"):              pFUTUREPROXYORDER.nCP = 0          elif (comboBoxnCP3.get() == "PUT"):              pFUTUREPROXYORDER.nCP = 1          if (comboBoxTOnPriceFlag3.get() == "市價"):              pFUTUREPROXYORDER.nPriceFlag = 0          elif (comboBoxTOnPriceFlag3.get() == "限價"):              pFUTUREPROXYORDER.nPriceFlag = 1          elif (comboBoxTOnPriceFlag3.get() == "範圍市價"):              pFUTUREPROXYORDER.nPriceFlag = 2          if (isDuplexOrder != True):              # 送出選擇權委託              bstrMessage,nCode= m\_pSKOrder.SendOptionProxyOrder(comboBoxUserID.get(), pFUTUREPROXYORDER)              if bstrMessage is not None:                  msg = "【SendOptionProxyOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage              else:                  msg = "【SendOptionProxyOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + " No message"              richTextBoxMethodMessage.insert('end',  msg + "\n")              richTextBoxMethodMessage.see('end')          else:              pFUTUREPROXYORDER.bstrStockNo2 = textBoxOptionID23.get()              pFUTUREPROXYORDER.bstrSettleYM2 = textBoxTObstrSettleYM23.get()              pFUTUREPROXYORDER.bstrStrike2 = textBoxbstrStrike23.get()              if (comboBoxnCP23.get() == "CALL"):                  pFUTUREPROXYORDER.nCP2 = 0              elif (comboBoxnCP23.get() == "PUT"):                  pFUTUREPROXYORDER.nCP2 = 1              if (comboBoxOptionBuySell23.get() == "買進"):                  pFUTUREPROXYORDER.nBuySell2 = 0              elif (comboBoxOptionBuySell23.get() == "賣出"):                  pFUTUREPROXYORDER.nBuySell2 = 1                # 經由proxy server送出選擇權複式下單              bstrMessage,nCode= m\_pSKOrder.SendDuplexProxyOrder(comboBoxUserID.get(), pFUTUREPROXYORDER)              if bstrMessage is not None:                  msg = "【SendDuplexProxyOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage              else:                  msg = "【SendDuplexProxyOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + " No message"                richTextBoxMethodMessage.insert('end',  msg + "\n")              richTextBoxMethodMessage.see('end') |

###### SendDuplexProxyOrder

|  |  |  |
| --- | --- | --- |
| 經由proxy server送出選擇權複式下單。 | | |
| **宣告** | Long SendDuplexProxyOrder ([in]BSTR bstrLogInID, [in]struct FUTUREPROXYORDER\* pFUTUREPROXYORDER, [out] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pFUTUREPROXYORDER | SKCOM元件中的[FUTUREPROXYORDER](#_Proxy_Server選擇權複式單_1)物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **委託送出成功：ORKEY**  **委託送出失敗：錯誤訊息** |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。錯誤代碼可參考對照表。 | |
| **備註** | 透過Proxy Server委託，委託結果請由[OnProxyOrder](#_OnProxyOrder)取得。  連線且成功登入，通知OnProxyStatus為5001時，送至proxy server進行下單。 | |

##### 刪改單

###### SendFutureProxyAlter

|  |  |  |
| --- | --- | --- |
| 經由proxy server送出期貨刪改單。 | | |
| **宣告** | Long SendFutureProxyAlter ([in]BSTR bstrLogInID, [in]struct FUTUREPROXYORDER\* pFUTUREPROXYORDER, [out] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pFUTUREPROXYORDER | SKCOM元件中的[FUTUREPROXYORDER](#_Proxy_Server期選刪改單_1)物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **委託送出成功：ORKEY**  **委託送出失敗：錯誤訊息** |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。錯誤代碼可參考對照表。 | |
| **備註** | 透過Proxy Server委託，委託結果請由[OnProxyOrder](#_OnProxyOrder)取得。  連線且成功登入，通知OnProxyStatus為5001時，送至proxy server進行下單。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.py類別UpdateProxyForm(tk.Frame)裡面 |
| UI程式碼(放置於def createWidgets(self): 的下一層)/UI示意圖 (**包含選擇權作法**:[SendOptionProxyAlter](#_SendOptionProxyAlter)) |
| # comboBoxUpdateTFOrder4          tk.Label(self, text = "下單類別").grid(row=0, column=1)              #輸入框          self.comboBoxUpdateTFOrder4 = ttk.Combobox(self, state='readonly')          self.comboBoxUpdateTFOrder4['values'] = Config.comboBoxUpdateTFOrder4          self.comboBoxUpdateTFOrder4.grid(row=0, column=2)          global comboBoxUpdateTFOrder4          comboBoxUpdateTFOrder4 = self.comboBoxUpdateTFOrder4          # textBoxSeqNo4          tk.Label(self, text = "請輸入委託序號").grid(row=1, column=1)              #輸入框          self.textBoxSeqNo4 = tk.Entry(self)          self.textBoxSeqNo4.grid(row=1, column=2)          global textBoxSeqNo4          textBoxSeqNo4 = self.textBoxSeqNo4          # textBoxBookNo4          tk.Label(self, text = "請輸入委託書號").grid(row=2, column=1)              #輸入框          self.textBoxBookNo4 = tk.Entry(self)          self.textBoxBookNo4.grid(row=2, column=2)          global textBoxBookNo4          textBoxBookNo4 = self.textBoxBookNo4          # textBoxStockDecreaseQty4          tk.Label(self, text = "請輸入減少數量").grid(row=3, column=1)              #輸入框          self.textBoxStockDecreaseQty4 = tk.Entry(self)          self.textBoxStockDecreaseQty4.grid(row=3, column=2)          global textBoxStockDecreaseQty4          textBoxStockDecreaseQty4 = self.textBoxStockDecreaseQty4          # textBoxPrice4          tk.Label(self, text = "請輸入修改價格").grid(row=4, column=1)              #輸入框          self.textBoxPrice4 = tk.Entry(self)          self.textBoxPrice4.grid(row=4, column=2)          global textBoxPrice4          textBoxPrice4 = self.textBoxPrice4          # comboBoxFutureReserved4          tk.Label(self, text = "盤別").grid(row=5, column=1)              #輸入框          self.comboBoxFutureReserved4 = ttk.Combobox(self, state='readonly')          self.comboBoxFutureReserved4['values'] = Config.comboBoxFutureReserved4          self.comboBoxFutureReserved4.grid(row=5, column=2)          global comboBoxFutureReserved4          comboBoxFutureReserved4 = self.comboBoxFutureReserved4          # comboBoxTradeType4          tk.Label(self, text = "ROD/IOC/FOK").grid(row=6, column=1)              #輸入框          self.comboBoxTradeType4 = ttk.Combobox(self, state='readonly')          self.comboBoxTradeType4['values'] = Config.comboBoxTradeType4          self.comboBoxTradeType4.grid(row=6, column=2)          global comboBoxTradeType4          comboBoxTradeType4 = self.comboBoxTradeType4          # buttonSendFutureProxyAlter          self.buttonSendFutureProxyAlter = tk.Button(self)          self.buttonSendFutureProxyAlter["text"] = "期貨刪改單送出"          self.buttonSendFutureProxyAlter["command"] = self.buttonSendFutureProxyAlter\_Click          self.buttonSendFutureProxyAlter.grid(row=7, column=1)          # buttonSendOptionProxyAlter          self.buttonSendOptionProxyAlter = tk.Button(self)          self.buttonSendOptionProxyAlter["text"] = "選擇權刪改單送出"          self.buttonSendOptionProxyAlter["command"] = self.buttonSendOptionProxyAlter\_Click          self.buttonSendOptionProxyAlter.grid(row=7, column=2) |

|  |
| --- |
| Config.py |
| # UpdateProxyForm  comboBoxUpdateTFOrder4 =(      "刪單", "減量", "改價"  )  comboBoxFutureReserved4 =(      "盤中單", "預約單"  )  comboBoxTradeType4 =(      "ROD", "IOC", "FOK"  ) |
| 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 buttonSendFutureProxyAlter\_Click(self):          pFUTUREPROXYORDER = sk.FUTUREPROXYORDER()          pFUTUREPROXYORDER.bstrFullAccount = comboBoxAccount.get()          if (comboBoxUpdateTFOrder4.get() == "刪單"):              pFUTUREPROXYORDER.bstrOrderType = "0"          elif (comboBoxUpdateTFOrder4.get() == "減量"):              pFUTUREPROXYORDER.bstrOrderType = "1"          elif (comboBoxUpdateTFOrder4.get() == "改價"):              pFUTUREPROXYORDER.bstrOrderType = "2"          pFUTUREPROXYORDER.bstrPrice = textBoxPrice4.get()          if (comboBoxFutureReserved4.get() == "盤中單"):              pFUTUREPROXYORDER.nReserved = 0          elif (comboBoxFutureReserved4.get() == "預約單"):              pFUTUREPROXYORDER.nReserved = 1          pFUTUREPROXYORDER.nQty = int(textBoxStockDecreaseQty4.get())          if (comboBoxTradeType4.get() == "ROD"):              pFUTUREPROXYORDER.nTradeType = 0          elif (comboBoxTradeType4.get() == "IOC"):              pFUTUREPROXYORDER.nTradeType = 1          elif (comboBoxTradeType4.get() == "FOK"):              pFUTUREPROXYORDER.nTradeType = 2          pFUTUREPROXYORDER.bstrBookNo = textBoxBookNo4.get()          pFUTUREPROXYORDER.bstrSeqNo = textBoxSeqNo4.get()          # 經由proxy server送出期貨刪改單          bstrMessage,nCode= m\_pSKOrder.SendFutureProxyAlter(comboBoxUserID.get(), pFUTUREPROXYORDER)          if bstrMessage is not None:               msg = "【SendFutureProxyAlter】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          else:               msg = "【SendFutureProxyAlter】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + " No message"          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end')      def buttonSendOptionProxyAlter\_Click(self):          pFUTUREPROXYORDER = sk.FUTUREPROXYORDER()          pFUTUREPROXYORDER.bstrFullAccount = comboBoxAccount.get()          if (comboBoxUpdateTFOrder4.get() == "刪單"):              pFUTUREPROXYORDER.bstrOrderType = "0"          elif (comboBoxUpdateTFOrder4.get() == "減量"):              pFUTUREPROXYORDER.bstrOrderType = "1"          elif (comboBoxUpdateTFOrder4.get() == "改價"):              pFUTUREPROXYORDER.bstrOrderType = "2"          pFUTUREPROXYORDER.bstrPrice = textBoxPrice4.get()          if (comboBoxFutureReserved4.get() == "盤中單"):              pFUTUREPROXYORDER.nReserved = 0          elif (comboBoxFutureReserved4.get() == "預約單"):              pFUTUREPROXYORDER.nReserved = 1          pFUTUREPROXYORDER.nQty = int(textBoxStockDecreaseQty4.get())          if (comboBoxTradeType4.get() == "ROD"):              pFUTUREPROXYORDER.nTradeType = 0          elif (comboBoxTradeType4.get() == "IOC"):              pFUTUREPROXYORDER.nTradeType = 1          elif (comboBoxTradeType4.get() == "FOK"):              pFUTUREPROXYORDER.nTradeType = 2          pFUTUREPROXYORDER.bstrBookNo = textBoxBookNo4.get()          pFUTUREPROXYORDER.bstrSeqNo = textBoxSeqNo4.get()          # 經由proxy server送出選擇權刪改單          bstrMessage,nCode= m\_pSKOrder.SendOptionProxyAlter(comboBoxUserID.get(), pFUTUREPROXYORDER)          if bstrMessage is not None:               msg = "【SendOptionProxyAlter】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage          else:               msg = "【SendOptionProxyAlter】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + " No message"          richTextBoxMethodMessage.insert('end',  msg + "\n")          richTextBoxMethodMessage.see('end') |

###### SendOptionProxyAlter

|  |  |  |
| --- | --- | --- |
| 經由proxy server送出選擇權刪改單。 | | |
| **宣告** | Long SendOptionProxyAlter([in]BSTR bstrLogInID, [in]struct FUTUREPROXYORDER\* pFUTUREPROXYORDER, [out] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pFUTUREPROXYORDER | SKCOM元件中的[FUTUREPROXYORDER](#_Proxy_Server期選刪改單_1)物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **委託送出成功：ORKEY**  **委託送出失敗：錯誤訊息** |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。錯誤代碼可參考對照表。 | |
| **備註** | 透過Proxy Server委託，委託結果請由[OnProxyOrder](#_OnProxyOrder)取得。  連線且成功登入，通知OnProxyStatus為5001時，送至proxy server進行下單。 | |

# 事件

## OnOpenInterest

|  |  |  |
| --- | --- | --- |
| (新)期貨未平倉GW。透過呼叫 [GetOpenInterest](#_GetOpenInterestGW)GW 後，資訊由該事件回傳。 | | |
| **宣告** | void OnOpenInterest([in] BSTR bstrData); | |
| **參數** | bstrData | 每一筆資料以「,」分隔每一個欄位，欄位依序為：  格式1:   |  |  | | --- | --- | | 1 | 市場別 | | 2 | 帳號 | | 3 | 商品 | | 4 | 買賣別 | | 5 | 未平倉部位 | | 6 | 當沖未平倉部位 | | 7 | 平均成本(小數部分已處理) | | 8 | 單口手續費 | | 9 | 交易稅(萬分之X) | | 10 | LOGIN\_ID | |
| **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。  若查無資料，則回傳M003 NO DATA#   * 含複式單，市場別：TM | |

|  |
| --- |
| Python範例程式碼\_TFOrder.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) |
| # (新)期貨未平倉GW。透過呼叫 GetOpenInterestGW 後，資訊由該事件回傳      def OnOpenInterest(self, bstrData):          msg = "【OnOpenInterest】" + bstrData          richTextBoxMessage.insert('end', msg + "\n")          richTextBoxMessage.see('end') |

## OnFutureRights

|  |  |  |
| --- | --- | --- |
| 國內期貨權益數。透過呼叫 [[GetFuture](#_4-2-41_GetFutureRights)Rights](#_GetFutureRights) 後，資訊由該事件回傳。 | | |
| **宣告** | void OnFutureRights([in] BSTR bstrData); | |
| **參數** | bstrData | 回傳的字串內容，以「,」分隔每一個欄位，欄位依序為：  0 帳戶餘額  1 浮動損益  2 已實現費用  3 交易稅  4 預扣權利金  5 權利金收付  6 權益數  7 超額保證金  8 存提款  9 買方市值  10 賣方市值  11 期貨平倉損益  12 盤中未實現  13 原始保證金  14 維持保證金  15 部位原始保證金  16 部位維持保證金  17 委託保證金  18 超額最佳保證金  19 權利總值  20 預扣費用  21 原始保證金  22 昨日餘額  23 選擇權組合單加不加收保證金  24 維持率  25 幣別  26 足額原始保證金  27 足額維持保證金  28 足額可用  29 抵繳金額  30 有價可用  31 可用餘額  32 足額現金可用  33 有價價值  34 風險指標  35 選擇權到期差異  36 選擇權到期差損  37 期貨到期損益  38 加收保證金  **39 LOGIN\_ID**  **40 ACCOUNT\_NO** |
| **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。 * 全幣別（含基幣），回傳第一筆資料為基幣。 | |

|  |
| --- |
| Python範例程式碼\_TFOrder.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) |
| # 國內期貨權益數。透過呼叫 GetFutureRights 後，資訊由該事件回傳      def OnFutureRights(self, bstrData):          msg = "【OnFutureRights】" + bstrData;          richTextBoxMessage.insert('end', msg + "\n")          richTextBoxMessage.see('end') |

## 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\_TFOrder.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') |

## OnProxyOrder

|  |  |  |
| --- | --- | --- |
| Proxy委託結果。 | | |
| **宣告** | void OnProxyOrder([in] LONG nStampID [in] LONG nCode, [in] BSTR bstrMessage); | |
| **參數** | nStampID | StampID |
| nCode | Proxy收單結果代碼。 |
| bstrMessage | Proxy收單回傳訊息。 |
| **備註** | 送出委託後會回傳二筆通知   |  |  | | --- | --- | | 第一筆 | | | nCode為0代表成功;其餘非0 代表失敗. | bstrMessage  Pending Order送出委託  ORKEY:為Time Stamp供比對第二筆通知使用 | | 第二筆 | | | nCode 為0代表成功;其餘非0 代表失敗. | bstrMessage  **Proxy回傳訊息欄位說明**  SeqNo:代表原始委託13碼  OrderNo:代表委託書號  ORKEY:為Time Stamp供比對第一筆通知使用 | | |

\*v2.13.47異動：補回傳參數nStampID文字說明

|  |
| --- |
| Python範例程式碼\_TFOrder.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) |
| # Proxy委託結果。      def OnProxyOrder(self, nStampID, nCode, bstrMessage):          msg = "【OnProxyOrder】" + str(nStampID) + str(nCode) + bstrMessage;          richTextBoxMessage.insert('end', msg + "\n")          richTextBoxMessage.see('end') |

# Struct結構物件

## FUTUREORDER (期權下單物件)

### 一般期選委託▼

struct FUTUREORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //委託期權代號

SHORT sTradeType; //0:ROD 1:IOC 2:FOK

SHORT sBuySell; //0:買進 1:賣出

SHORT sDayTrade; //當沖0:否 1:是，可當沖商品請參考交易所規定。

SHORT sNewClose; //新平倉，0:新倉 1:平倉 2:自動{新期貨、選擇權使用}

BSTR bstrPrice; //委託價格(IOC and FOK，可用「M」表示市價，「P」表示範圍市價)

LONG nQty; //交易口數

SHORT sReserved; //{期貨委託SendFutureOrderCLR適用}盤別，0:盤中(T盤及T+1盤)；1:T盤預約

};

### 選擇權複式單▼

struct FUTUREORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //委託選擇權代號1

BSTR bstrStockNo2; //委託選擇權代號2{複式單}

SHORT sTradeType; // 1:IOC 2:FOK

SHORT sBuySell; //0:買進 1:賣出

SHORT sBuySell2; //0:買進 1:賣出{複式單}

SHORT sNewClose; //新平倉，0:新倉 1:平倉

BSTR bstrPrice; //委託價格

LONG nQty; //交易口數

};

### 組合部位、複式單拆解、雙邊部位了結▼

組合部位▼

struct FUTUREORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //委託選擇權代號1{組合部位}

BSTR bstrStockNo2; //委託選擇權代號2{組合部位}

SHORT sBuySell; //0:買進 1:賣出{組合部位}

SHORT sBuySell2; //0:買進 1:賣出{組合部位}

LONG nQty; //交易口數{組合部位}

};

複式單拆解(單邊或雙邊)▼

struct FUTUREORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //**複式單商品代碼**{複式單拆解}

SHORT sBuySell; //0:買進 1:賣出{複式單拆解}

LONG nQty; //交易口數{複式單拆解}

};

複式單拆解(單邊或雙邊)

|  |  |  |
| --- | --- | --- |
| 異動前 | 欄位 | 異動後 |
| 複式單拆解(單邊或雙邊)  struct FUTUREORDER  {  BSTR bstrFullAccount;  BSTR bstrStockNo;  BSTR **bstrStockNo2;**  SHORT sBuySell;  LONG nQty;  }; | 刪除  雙邊  **bstrStockNo2;**  **//委託選擇權代號2** | 複式單拆解(單邊)  struct FUTUREORDER  {  BSTR bstrFullAccount;  BSTR bstrStockNo;  SHORT sBuySell;  LONG nQty;  }; |

雙邊部位了結▼

struct FUTUREORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //委託選擇權代號1{雙邊部位了結}

LONG nQty; //交易口數{雙邊部位了結}

};

### 全拆組單；全拆組平單委託▼

struct FUTUREORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

LONG nFunType; //功能 1：全拆組 2：全拆組平

};

## Proxy Server期選

### Proxy Server期貨下單

struct FUTUREPROXYORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //委託商品代號 ex:FITX

BSTR bstrSettleYM; // 指定月份商品契約年月，共6碼EX:202212

LONG nBuySell; //0:買進 1:賣出

LONG nPriceFlag; //0:市價 1:限價 2:範圍市價

LONG nDayTrade; //當沖0:否 1:是，可當沖商品請參考交易所規定。

BSTR bstrOrderType; //0:新倉 1:平倉 2:自動

LONG nReserved; //0:盤中單 1:預約單

LONG nQty; //交易口數{組合部位}

BSTR bstrPrice; //委託價格

LONG nTradeType; // 0:ROD 1:IOC 2:FOK

};

|  |
| --- |
| 備註 |
| 欲下單近月商品，EX: TX00，請於bstrStockNo輸入FITX、bstrSettleYM輸入(近月月份)202402 |

### Proxy Server選擇權下單

struct FUTUREPROXYORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //委託商品代號 ex:TXO

BSTR bstrPrice; //委託價格

BSTR bstrSettleYM; // 指定月份商品契約年月，共6碼EX:202212

BSTR bstrStrike; //履約價1

BSTR bstrOrderType; //0:新倉 1:平倉 2:自動

LONG nReserved; //0:盤中單 1:預約單

LONG nQty; //交易口數{組合部位}

LONG nCP; //0:CALL 1:PUT

LONG nBuySell; //0:買進 1:賣出

LONG nPriceFlag; //0:市價 1:限價 2:範圍市價

LONG nTradeType; // 0:ROD 1:IOC 2:FOK

LONG nDayTrade; //當沖0:否 1:是，可當沖商品請參考交易所規定。

};

\*下單範例：2024年3月第4週選擇權商品，請於bstrStockNo輸入TX4、bstrSettleYM輸入202403。2024年4月第1週選擇權商品，請於bstrStockNo輸入TX1、bstrSettleYM輸入202404。

### Proxy Server選擇權複式單

struct FUTUREPROXYORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrStockNo; //委託商品代號 ex:TXO

BSTR bstrPrice; //委託價格

BSTR bstrSettleYM; //指定月份商品契約年月，共6碼EX:202212

BSTR bstrStrike; //履約價1

BSTR bstrSettleYM2; //指定月份商品契約年月2，共6碼EX:202301

BSTR bstrStrike2; //履約價2

BSTR bstrOrderType; //0:新倉 1:平倉 2:自動

LONG nReserved; //0:盤中單 1:預約單

LONG nQty; //交易口數{組合部位}

LONG nCP; //買賣權1 ( 0:CALL 1:PUT )

LONG nBuySell; //買賣別1 ( 0:買進 1:賣出)

BSTR bstrStockNo2; //委託商品代號2

LONG nCP2; //買賣權2 ( 0:CALL 1:PUT )

LONG nBuySell2; //買賣別2 ( 0:買進 1:賣出)

LONG nPriceFlag; // 0:市價 1:限價 2:範圍市價

LONG nTradeType // 1:IOC 2:FOK

LONG nDayTrade; //當沖0:否 1:是，可當沖商品請參考交易所規定。

};

### Proxy Server期選刪改單

struct FUTUREPROXYORDER

{

BSTR bstrFullAccount; //期貨帳號，Broker id (例如: F020000)＋帳號7碼

BSTR bstrOrderType; //0:刪單 1:減量 2:改價

BSTR bstrPrice; //委託價格

(改量:帶0，也可以帶""。改價:帶需要改的價。刪單:帶原始價，也可帶"")

LONG nReserved; //0:盤中單 1:預約單

LONG nQty; //交易口數

(改量 : 帶要減的量。改價 : 帶0也可帶""。刪單 : 帶原始量)

LONG nTradeType; // 0:ROD 1:IOC 2:FOK

BSTR bstrBookNo; //委託書號

BSTR bstrSeqNo; //委託序號

};