策略王COM元件使用說明

-下單-國內證券-

## 函式範例

### SKOrderLib(國內證券)

*函式*

1. 查詢

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| 1. **證券即時庫存** | [GetRealBalanceReport](#_GetRealBalanceReport) |  |
| 1. **查詢集保庫存** | [GetBalanceQuery](#_GetBalanceQuery) |  |
| 1. **查詢證券資券配額** | [GetMarginPurchaseAmountLimit](#_GetMarginPurchaseAmountLimit_1) |  |
| 1. **新損益試算查詢** | [GetProfitLossGWReport](#_GetProfitLossGWReport) |  |

1. 一般下單/刪改單

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| 1. **證券委託** | [SendStockOrder](#_SendStockOrder) | ＊適用逐筆交易 |
| 1. **證券盤中零股委託** | [SendStockOddLotOrder](#_SendStockOddLotOrder) |  |
| 1. **證期權改價** | [CorrectPriceBySeqNo](#_CorrectPriceBySeqNo_1) | 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. **Proxy下單**/**刪改單**

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| 1. **證券下單** | [SendStockProxyOrder](#_SendStockProxyOrder) |  |
| 1. **證券刪改單** | [SendStockProxyAlter](#_SendStockProxyAlter_1) |  |

*事件*

|  |  |  |
| --- | --- | --- |
| **功能** | **事件名稱** | **備註** |
| 1. **證券即時庫存** | [OnRealBalanceReport](#_OnRealBalanceReport) |  |
| 1. **查詢集保庫存** | [OnBalanceQuery](#_OnBalanceQuery_1) |  |
| 1. **查詢證券資券配額** | [OnMarginPurchaseAmountLimit](#_OnMarginPurchaseAmountLimit_1) |  |
| 1. **新損益試算資料** | [OnProfitLossGWReport](#_OnProfitLossGWReport_1) |  |
| 1. **非同步委託結果** | [OnAsyncOrder](#_OnAsyncOrder) |  |
| 1. **Proxy委託結果** | [OnProxyOrder](#_OnProxyOrder) |  |

#### 查詢

##### GetRealBalanceReport

|  |  |  |
| --- | --- | --- |
| 查詢證券即時庫存內容。 | | |
| **宣告** | Long GetRealBalanceReport([in] BSTR bstrLogInID, [in] BSTR bstrAccount); | |
| **參數** | bstrLogInID | 登入ID。 |
| bstrAccount | 證券帳號，分公司四碼＋帳號7碼。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 結果由[OnRealBalanceReport](#_OnRealBalanceReport)事件回傳。 | |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 拉出 1個Button，命名(Name)為「buttonGetRealBalanceReport」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |
| Current Block (可放在buttonGetRealBalanceReport\_Click的Click事件裡) |
| //Method  dataGridViewGetRealBalanceReport.Rows.Clear(); // 每次查詢前，清除上一次查詢結果  // 查詢證券即時庫存內容  string bstrAccount = comboBoxAccount.Text;  int nCode = m\_pSKOrder.GetRealBalanceReport(comboBoxUserID.Text, bstrAccount);  // 取得回傳訊息  string msg = "【GetRealBalanceReport】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### GetBalanceQuery

|  |  |  |
| --- | --- | --- |
| 集保庫存查詢。 | | |
| **宣告** | Long GetBalanceQuery ([in] BSTR bstrLogInID, [in] BSTR bstrAccount, [in] BSTR bstrStockNo) | |
| **參數** | bstrLogInID | 登入ID。 |
| bstrAccount | 證券帳號，分公司四碼＋帳號7碼，帶空為所有證券帳號查詢。 |
| bstrStockNo | 商品代碼，代空為全部商品回傳 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 結果由 [OnBalanceQuery](#_OnBalanceQuery) 事件回傳。 | |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 拉出 1個Button，命名(Name)為「buttonGetBalanceQuery」。  拉出 1 個 TextBox，命名(Name)為「textBoxbstrStockNo1」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |
| Current Block (可放在buttonGetBalanceQuery\_Click的Click事件裡) |
| //Method  dataGridViewGetBalanceQuery.Rows.Clear(); // 每次查詢前，清除上一次查詢結果  // 集保庫存查詢  string bstrAccount = comboBoxAccount.Text;  string bstrStockNo = textBoxbstrStockNo1.Text; // 代空為全部商品回傳  int nCode = m\_pSKOrder.GetBalanceQuery(comboBoxUserID.Text, bstrAccount, bstrStockNo);  // 取得回傳訊息  string msg = "【GetBalanceQuery】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### GetMarginPurchaseAmountLimit

|  |  |  |
| --- | --- | --- |
| 資券配額查詢。 | | |
| **宣告** | Long GetMarginPurchaseAmountLimit ([in] BSTR bstrLogInID, [in] BSTR bstrAccount[in] BSTR bstrStockNo) | |
| **參數** | bstrLogInID | 登入ID。 |
| bstrAccount | 證券帳號，分公司四碼＋帳號7碼。 |
| bstrStockNo | 商品代碼，代空為全部商品回傳 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 結果由 [OnMarginPurchaseAmountLimit](#_OnMarginPurchaseAmountLimit) 事件回傳。 | |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 拉出 1個Button，命名(Name)為「buttonGetMarginPurchaseAmountLimit」。  拉出 1 個TextBox，命名(Name)為「textBoxbstrStockNo2」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |
| Current Block (可放在buttonGetMarginPurchaseAmountLimit\_Click的Click事件裡) |
| //Method  string bstrLogInID = comboBoxUserID.Text; // 登入ID  string bstrAccount = comboBoxAccount.Text; // 證券帳號，分公司四碼＋帳號7碼  string bstrStockNo = textBoxbstrStockNo2.Text; // 商品代碼，代空為全部商品回傳  dataGridViewOnMarginPurchaseAmountLimit.Rows.Clear(); //清空  // 資券配額查詢  int nCode = m\_pSKOrder.GetMarginPurchaseAmountLimit(bstrLogInID, bstrAccount, bstrStockNo);  // 取得回傳訊息  string msg = "【GetMarginPurchaseAmountLimit】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### GetProfitLossGWReport

|  |  |  |
| --- | --- | --- |
| 未實現損益試算。 | | |
| **宣告** | Long GetProfitLossGWReport ([in] BSTR bstrLogInID, [in] struct **TSPROFITLOSSGWQUERY**\* pPLGWQuery); | |
| **參數** | bstrLogInID | 登入ID。 |
| pPLGWQuery | SKCOM元件中的**[TSPROFITLOSSGWQUERY](#_未實現)**物件，將未實現損益查詢條件填入該物件後，再帶入此欄位中。 |
| **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** | 結果由[OnProfitLossGWReport](#_OnProfitLossGWReport)事件回傳。 | |

|  |  |  |
| --- | --- | --- |
| 已實現損益查詢。 | | |
| **宣告** | Long GetProfitLossGWReport ([in] BSTR bstrLogInID, [in] struct **TSPROFITLOSSGWQUERY**\* pPLGWQuery); | |
| **參數** | bstrLogInID | 登入ID。 |
| pPLGWQuery | SKCOM元件中的**[TSPROFITLOSSGWQUERY](#_已實現)**物件，將未實現損益查詢條件填入該物件後，再帶入此欄位中。 |
| **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** | 結果由[OnProfitLossGWReport](#_OnProfitLossGWReport)事件回傳。   * 彙總：起始日西元年月日及結束日西元年月日 * 明細：起始日**民國年月日**，不需填結束日 | |

|  |  |  |
| --- | --- | --- |
| 現股當沖損益試算。 | | |
| **宣告** | Long GetProfitLossGWReport ([in] BSTR bstrLogInID, [in] struct **TSPROFITLOSSGWQUERY**\* pPLGWQuery); | |
| **參數** | bstrLogInID | 登入ID。 |
| pPLGWQuery | SKCOM元件中的**[TSPROFITLOSSGWQUERY](#_現股當沖:(限查閱當日交易))**物件，將未實現損益查詢條件填入該物件後，再帶入此欄位中。 |
| **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** | 結果由[OnProfitLossGWReport](#_現股當沖:(限查閱當日交易))事件回傳。   * 限查當日交易 * 損益回傳格式 1: 彙總 2: 明細 | |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出 5個TextBox，命名(Name)為「textBoxbstrStockNo3」、「textBoxbstrStartDate」、「textBoxbstrEndDate」、「textBoxbstrBookNo」、「textBoxbstrSeqNo」。 拉出 3個ComboBox，命名(Name)為「comboBoxnTPQueryType」、「comboBoxnFunc」、「comboBoxbstrTradeType」  拉出 1個Button，命名(Name)為「buttonGetProfitLossGWReport」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |

|  |
| --- |
| Constructor |
| // comboBoxnTPQueryType  {  comboBoxnTPQueryType.Items.Add("未實現損益");  comboBoxnTPQueryType.Items.Add("已實現損益");  comboBoxnTPQueryType.Items.Add("現股當沖");  }  // comboBoxnFunc  {  comboBoxnFunc.Items.Add("彙總");  comboBoxnFunc.Items.Add("明細");  comboBoxnFunc.Items.Add("投資總額");  comboBoxnFunc.Items.Add("彙總(依股票代號)");  }  // comboBoxbstrTradeType  {  comboBoxbstrTradeType.Items.Add("現股");  comboBoxbstrTradeType.Items.Add("融資(自)");  comboBoxbstrTradeType.Items.Add("融券(自)");  comboBoxbstrTradeType.Items.Add("融資(代)");  comboBoxbstrTradeType.Items.Add("融券(代)");  comboBoxbstrTradeType.Items.Add("券差");  comboBoxbstrTradeType.Items.Add("無券賣出");  } |

|  |
| --- |
| Current Block (可放在buttonGetProfitLossGWReport\_Click的Click事件裡) |
| //Method  string bstrLogInID = comboBoxUserID.Text; // 登入ID  TSPROFITLOSSGWQUERY pPLGWQuery = new TSPROFITLOSSGWQUERY(); // SKCOM元件中的TSPROFITLOSSGWQUERY物件，將損益查詢條件填入該物件後，再帶入此欄位中  pPLGWQuery.bstrFullAccount = comboBoxAccount.Text;  if (comboBoxnTPQueryType.Text == "未實現損益")//損益類別 0: 未實現損益；1: 已實現損益 2:現股當沖  {  pPLGWQuery.nTPQueryType = 0;  //損益回傳格式 //(未實現) 0:彙總 ；1:明細  if (comboBoxnFunc.Text == "彙總")  {  pPLGWQuery.nFunc = 0;  dataGridViewOnProfitLossGWReport1.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage1; // 未實現(彙總)  }  else if (comboBoxnFunc.Text == "明細")  {  pPLGWQuery.nFunc = 1;  dataGridViewOnProfitLossGWReport2.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage2; // 未實現(明細)  }  if (pPLGWQuery.nFunc == 1)  {  pPLGWQuery.bstrStockNo = textBoxbstrStockNo3.Text; // 當nFunc 為 1:明細 時,指定商品代碼  // 當nFunc 為 1:明細 時,指定交易類別 0:現股 3:融資(自) 4:融券(自) 8:券差 9:無券賣出  if (comboBoxbstrTradeType.Text == "現股") pPLGWQuery.bstrTradeType = "0";  else if (comboBoxbstrTradeType.Text == "融資(自)") pPLGWQuery.bstrTradeType = "3";  else if (comboBoxbstrTradeType.Text == "融券(自)") pPLGWQuery.bstrTradeType = "4";  else if (comboBoxbstrTradeType.Text == "券差") pPLGWQuery.bstrTradeType = "8";  else if (comboBoxbstrTradeType.Text == "無券賣出") pPLGWQuery.bstrTradeType = "9";  }  }  else if (comboBoxnTPQueryType.Text == "已實現損益")  {  pPLGWQuery.nTPQueryType = 1;  //損益回傳格式 //(已實現) // 0:彙總；1:明細; 2:投資總額; 3: 彙總(依股票代號)  if (comboBoxnFunc.Text == "彙總")  {  pPLGWQuery.nFunc = 0;  dataGridViewOnProfitLossGWReport3.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage3; // 已實現(彙總)  }  else if (comboBoxnFunc.Text == "明細")  {  pPLGWQuery.nFunc = 1;  dataGridViewOnProfitLossGWReport4.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage4; // 已實現(明細)  }  else if (comboBoxnFunc.Text == "投資總額")  {  pPLGWQuery.nFunc = 2;  dataGridViewOnProfitLossGWReport5.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage5; // 已實現(投資總額)  }  else if (comboBoxnFunc.Text == "彙總(依股票代號)")  {  pPLGWQuery.nFunc = 3;  dataGridViewOnProfitLossGWReport6.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage6; // 已實現(彙總(依股票代號))  }  if (pPLGWQuery.nFunc == 1 || pPLGWQuery.nFunc == 3)  {  pPLGWQuery.bstrStockNo = textBoxbstrStockNo3.Text;//當nFunc 為 1:明細 及3: 彙總依股票代號 時，指定商品代碼  }  if (pPLGWQuery.nFunc == 1)  {  // 當nFunc 為 1:明細 時,指定交易類別 0:現股 1:融資(代) 2:融券(代) 3:融資(自) 4:融券(自)  if (comboBoxbstrTradeType.Text == "現股") pPLGWQuery.bstrTradeType = "0";  else if (comboBoxbstrTradeType.Text == "融資(代)") pPLGWQuery.bstrTradeType = "1";  else if (comboBoxbstrTradeType.Text == "融券(代)") pPLGWQuery.bstrTradeType = "2";  else if (comboBoxbstrTradeType.Text == "融資(自)") pPLGWQuery.bstrTradeType = "3";  else if (comboBoxbstrTradeType.Text == "融券(自) ") pPLGWQuery.bstrTradeType = "4";  }  //當nFunc 為 0:彙總 時，起始日西元年月日//YYYYMMDD ； 當nFunc 為 1:明細 及 3:彙總依股票代號 時，起始日民國年月日  if (pPLGWQuery.nFunc == 0 || pPLGWQuery.nFunc == 1 || pPLGWQuery.nFunc == 3) pPLGWQuery.bstrStartDate = textBoxbstrStartDate.Text;  //當nFunc 為 0:彙總 時，結束日西元年月日//YYYYMMDD ； 當nFunc 為 1:明細格式時,不需填結束日 ； 當nFunc 3:彙總依股票代號 時，結束日 民國年月日  if (pPLGWQuery.nFunc == 0 || pPLGWQuery.nFunc == 3) pPLGWQuery.bstrEndDate = textBoxbstrEndDate.Text;  //當nFunc 為 1:明細 時,指定委託書號  if (pPLGWQuery.nFunc == 1) pPLGWQuery.bstrBookNo = textBoxbstrBookNo.Text;  //當nFunc 為 1:明細 時,指定序號  if (pPLGWQuery.nFunc == 1) pPLGWQuery.bstrSeqNo = textBoxbstrSeqNo.Text;  }  else if (comboBoxnTPQueryType.Text == "現股當沖")  {  pPLGWQuery.nTPQueryType = 2;  //損益回傳格式 //(未實現) 0:彙總 ；1:明細  if (comboBoxnFunc.Text == "彙總")  {  pPLGWQuery.nFunc = 0;  dataGridViewOnProfitLossGWReport7.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage7; // 現股當沖(彙總)  }  else if (comboBoxnFunc.Text == "明細")  {  pPLGWQuery.nFunc = 1;  dataGridViewOnProfitLossGWReport8.Rows.Clear();  tabControlTSPROFITLOSSGWQUERY.SelectedTab = tabPage8; // 現股當沖(明細)  }  if (pPLGWQuery.nFunc == 1)  {  pPLGWQuery.bstrStockNo = textBoxbstrStockNo3.Text;//當nFunc 為 1:明細時，需指定商品代碼  }  }  // 依照TSPROFITLOSSGWQUERY物件裡的nTPQueryType參數決定損益類別(0: 未實現損益；1: 已實現損益 2:現股當沖)  int nCode = m\_pSKOrder.GetProfitLossGWReport(bstrLogInID, pPLGWQuery);  // 取得回傳訊息  string msg = "【GetProfitLossGWReport】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

#### **一般下單/刪改單**

##### SendStockOrder

|  |  |  |
| --- | --- | --- |
| 送出證券委託。 | | |
| **宣告** | Long SendStockOrder([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKORDER**\* 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 [**STOCKORDER**](#_STOCKORDER_(_證券下單物件) 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](#_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。  ＊亦適用證券逐筆 | |

|  |
| --- |
| C#範例程式碼\_ TSSendOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 拉出 1個CheckBox，命名(Name)為「checkBoxAsyncOrder」。  拉出3個TextBox，命名(Name)為「textBoxStockID」、「textBoxnQty」、「textBoxbstrPrice」。 拉出6個ComboBox，命名(Name)為「comboBoxPrime」、「comboBoxPeriod」、「comboBoxFlag」、「comboBoxnTradeType」、「comboBoxnSpecialTradeType」、「comboBoxBuySell」。  拉出 1個Button，命名(Name)為「buttonSendStockOrder」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Constructor |
| // comboBoxPrime  comboBoxPrime.Items.Add("上市上櫃");  comboBoxPrime.Items.Add("興櫃");  // comboBoxPeriod  comboBoxPeriod.Items.Add("盤中");  comboBoxPeriod.Items.Add("盤後");  comboBoxPeriod.Items.Add("零股");  // comboBoxFlag  comboBoxFlag.Items.Add("現股");  comboBoxFlag.Items.Add("融資");  comboBoxFlag.Items.Add("融券");  comboBoxFlag.Items.Add("無券");  // comboBoxBuySell  comboBoxBuySell.Items.Add("買進");  comboBoxBuySell.Items.Add("賣出");  // comboBoxnTradeType  comboBoxnTradeType.Items.Add("ROD");  comboBoxnTradeType.Items.Add("IOC");  comboBoxnTradeType.Items.Add("FOK");  // comboBoxnSpecialTradeType  comboBoxnSpecialTradeType.Items.Add("市價");  comboBoxnSpecialTradeType.Items.Add("限價"); |

|  |
| --- |
| Current Block (可放在buttonSendStockOrder\_Click的Click事件裡) |
| //Method  STOCKORDER pOrder = new STOCKORDER();  string bstrMessage;  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxStockID.Text;  string selectedValue = comboBoxPrime.Text;  if (selectedValue == "上市上櫃") pOrder.sPrime = 0;  else if (selectedValue == "興櫃") pOrder.sPrime = 1;  selectedValue = comboBoxPeriod.Text;  if (selectedValue == "盤中") pOrder.sPeriod = 0;  else if (selectedValue == "盤後") pOrder.sPeriod = 1;  else if (selectedValue == "零股") pOrder.sPeriod = 2;  selectedValue = comboBoxFlag.Text;  if (selectedValue == "現股") pOrder.sFlag = 0;  else if (selectedValue == "融資") pOrder.sFlag = 1;  else if (selectedValue == "融券") pOrder.sFlag = 2;  else if (selectedValue == "無券") pOrder.sFlag = 3;  selectedValue = comboBoxBuySell.Text;  if (selectedValue == "買進") pOrder.sBuySell = 0;  else if (selectedValue == "賣出") pOrder.sBuySell = 1;  pOrder.bstrPrice = textBoxbstrPrice.Text;  pOrder.nQty = int.Parse(textBoxnQty.Text);  selectedValue = comboBoxnTradeType.Text;  if (selectedValue == "ROD") pOrder.nTradeType = 0;  else if (selectedValue == "IOC") pOrder.nTradeType = 1;  else if (selectedValue == "FOK") pOrder.nTradeType = 2;  selectedValue = comboBoxnSpecialTradeType.Text;  if (selectedValue == "市價") pOrder.nSpecialTradeType = 1;  else if (selectedValue == "限價") pOrder.nSpecialTradeType = 2;  string msg;  // 送出證券委託  int nCode = m\_pSKOrder.SendStockOrder(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  msg = "【SendStockOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  // 取得回傳訊息(bstrMessage)  if (bAsyncOrder == false)  {  msg = "【同步委託結果】" + bstrMessage;  richTextBoxMessage.AppendText(msg + "\n");  } |
| 執行成功畫面 |
|  |

##### SendStockOddLotOrder

|  |  |  |
| --- | --- | --- |
| 送出證券盤中零股委託。 | | |
| **宣告** | Long SendStockOddLotOrder([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct STOCKORDER\* 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 [STOCKORDER](#_盤中零股) 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](#_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。 | |

|  |
| --- |
| C#範例程式碼\_ TSSendOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 【紅色虛線部分控件可參考 [SendStockOrder](#_SendStockOrder) 做法】  拉出 1個Button，命名(Name)為「buttonSendStockOddLotOrder」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Current Block (可放在buttonSendStockOddLotOrder\_Click的Click事件裡) |
| STOCKORDER pOrder = new STOCKORDER();  string bstrMessage;  pOrder.bstrFullAccount = comboBoxAccount.Text;//證券帳號，分公司代碼＋帳號7碼  pOrder.bstrStockNo = textBoxStockID.Text;//委託股票代號  pOrder.sPeriod = 4;// 4:盤中零股  pOrder.sFlag = 0; //0:現股  string selectedValue = comboBoxBuySell.Text;  if (selectedValue == "買進") pOrder.sBuySell = 0;  else if (selectedValue == "賣出") pOrder.sBuySell = 1;//0:買進 1:賣出  pOrder.bstrPrice = textBoxbstrPrice.Text;//委託價格  pOrder.nQty = int.Parse(textBoxnQty.Text);//零股數量為股數1~999股  string msg;  // 送出證券盤中零股委託  int nCode = m\_pSKOrder.SendStockOddLotOrder(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  msg = "【SendStockOddLotOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n");  // 取得回傳訊息(bstrMessage)  if (bAsyncOrder == false)  {  msg = "【同步委託結果】" + bstrMessage;  richTextBoxMessage.AppendText(msg + "\n");  } |
| 執行成功畫面 |
|  |

##### 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 | |

|  |
| --- |
| C#範例程式碼\_ TSTFUpdateOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 拉出2個TextBox，命名(Name)為「textBoxSeqNo」、「textBoxPrice」。 拉出3個ComboBox，命名(Name)為「comboBoxUpdateTFOrder」、「comboBoxTradeType」、「comboBoxMarketSymbol」。  拉出 1個Button，命名(Name)為「 buttonCorrectPriceBySeqNo」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Constructor |
| // comboBoxTradeType  {  comboBoxTradeType.Items.Add("0:ROD");  comboBoxTradeType.Items.Add("1:IOC");  comboBoxTradeType.Items.Add("2:FOK");  }  // comboBoxCorrectPriceByBookNoMarketSymbol  {  comboBoxMarketSymbol.Items.Add("TS:證券");  comboBoxMarketSymbol.Items.Add("TF:期貨");  comboBoxMarketSymbol.Items.Add("TO:選擇權");  }  // comboBoxUpdateTFOrder  {  comboBoxUpdateTFOrder.Items.Add("刪單");  comboBoxUpdateTFOrder.Items.Add("減量");  comboBoxUpdateTFOrder.Items.Add("改價");  } |

|  |
| --- |
| Current Block (可放在buttonCorrectPriceBySeqNo\_Click的Click事件裡) |
| string bstrAccount = comboBoxAccount.Text;  string bstrSeqNo = textBoxSeqNo.Text; // 欲改量的委託序號  string bstrPrice = textBoxPrice.Text; // 修改價格  int nTradeType = 0; // 證券0:ROD 期選0: ROD 1:IOC 2:FOK  string selectedValue2 = comboBoxTradeType.Text;  if (selectedValue2 == "0:ROD") nTradeType = 0;  else if (selectedValue2 == "1:IOC") nTradeType = 1;  else if (selectedValue2 == "2:FOK") nTradeType = 2;  string bstrMessage; //同步刪單：如果回傳值為 0表示刪單成功，訊息內容則為修改訊息。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4 - 2 - b OnAsyncOrder。  // 證期權依序號改價(By委託序號)  int nCode = m\_pSKOrder.CorrectPriceBySeqNo(comboBoxUserID.Text, bAsyncOrder, bstrAccount, bstrSeqNo, bstrPrice, nTradeType, out bstrMessage);  // 取得回傳訊息  string msg = "【CorrectPriceBySeqNo】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage;  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### 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 | |

|  |
| --- |
| C#範例程式碼\_ TSTFUpdateOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 【紅色虛線部分控件可參考 [CorrectPriceBySeqNo](#_CorrectPriceBySeqNo_1)做法】  拉出1個TextBox，命名(Name)為「textBoxBookNo」。  拉出 1個Button，命名(Name)為「buttonCorrectPriceByBookNo」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Current Block (可放在buttonCorrectPriceByBookNo\_Click的Click事件裡) |
| string bstrAccount = comboBoxAccount.Text;  string bstrMarketSymbol = ""; // 市場類別 TS: 證券TF:期貨 TO:選擇權  string selectedValue3 = comboBoxMarketSymbol.Text;  if (selectedValue3 == "TS:證券") bstrMarketSymbol = "TS";  else if (selectedValue3 == "TF:期貨") bstrMarketSymbol = "TF";  else if (selectedValue3 == "TO:選擇權") bstrMarketSymbol = "TO";  string bstrBookNo = textBoxBookNo.Text; // 欲改量的委託書號  string bstrPrice = textBoxPrice.Text; // 修改價格  int nTradeType = 0; // 證券0:ROD 期選0: ROD 1:IOC 2:FOK  string selectedValue2 = comboBoxTradeType.Text;  if (selectedValue2 == "0:ROD") nTradeType = 0;  else if (selectedValue2 == "1:IOC") nTradeType = 1;  else if (selectedValue2 == "2:FOK") nTradeType = 2;  string bstrMessage; //同步刪單：如果回傳值為 0表示刪單成功，訊息內容則為修改訊息。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4 - 2 - b OnAsyncOrder。  // 證期權依書號改價(By委託書號)  int nCode = m\_pSKOrder.CorrectPriceByBookNo(comboBoxUserID.Text, bAsyncOrder, bstrAccount, bstrMarketSymbol, bstrBookNo, bstrPrice, nTradeType, out bstrMessage);  // 取得回傳訊息  string msg = "【CorrectPriceByBookNo】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage;  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### 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 | |

|  |
| --- |
| C#範例程式碼\_ TSTFUpdateOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 【紅色虛線部分控件可參考 [CorrectPriceBySeqNo](#_CorrectPriceBySeqNo_1)做法】  拉出1個TextBox，命名(Name)為「textBoxStockDecreaseQty」。  拉出 1個Button，命名(Name)為「buttonDecreaseOrderBySeqNo」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Current Block (可放在buttonDecreaseOrderBySeqNo\_Click的Click事件裡) |
| string bstrSeqNo = textBoxSeqNo.Text;  string bstrAccount = comboBoxAccount.Text;  int nDecreaseQty = int.Parse(textBoxStockDecreaseQty.Text);  string bstrMessage; //同步刪單：如果回傳值為 0表示刪單成功，訊息內容則為修改訊息。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4 - 2 - b OnAsyncOrder。  // 國內委託減量(By委託序號)  int nCode = m\_pSKOrder.DecreaseOrderBySeqNo(comboBoxUserID.Text, bAsyncOrder, bstrAccount, bstrSeqNo, nDecreaseQty, out bstrMessage);  // 取得回傳訊息  string msg = "【DecreaseOrderBySeqNo】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage;  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### 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 | |

|  |
| --- |
| C#範例程式碼\_ TSTFUpdateOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 【紅色虛線部分控件可參考 [CorrectPriceBySeqNo](#_CorrectPriceBySeqNo_1)做法】  拉出 1個Button，命名(Name)為「buttonCancelOrderBySeqNo」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Current Block (可放在buttonCancelOrderBySeqNo\_Click的Click事件裡) |
| string bstrSeqNo = textBoxSeqNo.Text;  string bstrAccount = comboBoxAccount.Text;  string bstrMessage; //同步刪單：如果回傳值為 0表示刪單成功，訊息內容則為修改訊息。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4 - 2 - b OnAsyncOrder。  // 國內委託删單(By委託序號)  int nCode = m\_pSKOrder.CancelOrderBySeqNo(comboBoxUserID.Text, bAsyncOrder, bstrAccount, bstrSeqNo, out bstrMessage);  // 取得回傳訊息  string msg = "【CancelOrderBySeqNo】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage;  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### 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。 | |

|  |
| --- |
| C#範例程式碼\_ TSTFUpdateOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 【紅色虛線部分控件可參考 [CorrectPriceBySeqNo](#_CorrectPriceBySeqNo_1)做法】  拉出 1個Button，命名(Name)為「buttonCancelOrderByBookNo」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Current Block (可放在buttonCancelOrderByBookNo\_Click的Click事件裡) |
| string bstrBookNo = textBoxBookNo.Text;  string bstrAccount = comboBoxAccount.Text;  string bstrMessage; //同步刪單：如果回傳值為 0表示刪單成功，訊息內容則為修改訊息。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4 - 2 - b OnAsyncOrder。  // 國內委託删單(By委託書號)  int nCode = m\_pSKOrder.CancelOrderByBookNo(comboBoxUserID.Text, bAsyncOrder, bstrAccount, bstrBookNo, out bstrMessage);  // 取得回傳訊息  string msg = "【CancelOrderByBookNo】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage;  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### 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 | 欄位 | 異動後V2.13.45 |
| 欲刪除的委託商品代號，空白則刪除所該帳號所屬登入ID之所有委託，請注意。 | bstrStockNo  欄位說明：  移除商品代碼為”空白”情況。  功能部分：  V2.13.45（含）以上，商品代碼不可填空白。 | 欲刪除的委託商品代號，請輸入有效商品代碼。 |

|  |
| --- |
| C#範例程式碼\_ TSTFUpdateOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 【紅色虛線部分控件可參考 [CorrectPriceBySeqNo](#_CorrectPriceBySeqNo_1)做法】  拉出 1個Button，命名(Name)為「buttonCancelOrderByStockNo」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |

|  |
| --- |
| Current Block (可放在buttonCancelOrderByStockNo\_Click的Click事件裡) |
| string bstrStockNo = textBoxCancelOrderByStockNo.Text;  string bstrAccount = comboBoxAccount.Text;  string bstrMessage; //同步刪單：如果回傳值為 0表示刪單成功，訊息內容則為修改訊息。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4 - 2 - b OnAsyncOrder。  // 國內委託删單(By ID+商品代號)  int nCode = m\_pSKOrder.CancelOrderByStockNo(comboBoxUserID.Text, bAsyncOrder, bstrAccount, bstrStockNo, out bstrMessage);  // 取得回傳訊息  string msg = "【CancelOrderByStockNo】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage;  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

#### **Proxy下單/刪改單**

##### SendStockProxyOrder

|  |  |  |
| --- | --- | --- |
| 經由proxy server送出證券委託。 | | |
| **宣告** | long SendStockProxyOrder([in]BSTR bstrLogInID, [in]struct STOCKPROXYORDER\* pSTOCKPROXYORDER, [out] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pSTOCKPROXYORDER | SKCOM元件中的 [STOCKPROXYORDER](#_Proxy_Server_證券下單) 物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **Proxy委託通知：**參照[OnProxyOrder](#_OnProxyOrder)。 |
| **回傳值** | 0表示連線成功，其餘非0數值都表示失敗。 | |
| **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。 | |

|  |
| --- |
| C#範例程式碼\_ TSSKProxySendOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 拉出3個TextBox，命名(Name)為「textBoxStockID」、「textBoxnQty」、「textBoxbstrPrice」。 拉出5個ComboBox，命名(Name)為「comboBoxPeriod」、「comboBoxbstrOrderType」、「comboBoxnTradeType」、「comboBoxnSpecialTradeType」、「comboBoxnPriceMark」。  拉出 1個Button，命名(Name)為「buttonSendStockOrder」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |

|  |
| --- |
| Constructor |
| // sPeriod  comboBoxPeriod.Items.Add("盤中");  comboBoxPeriod.Items.Add("零股");  comboBoxPeriod.Items.Add("盤後交易");  comboBoxPeriod.Items.Add("盤中零股");  // comboBoxbstrOrderType  comboBoxbstrOrderType.Items.Add("現股買進");  comboBoxbstrOrderType.Items.Add("現股賣出");  comboBoxbstrOrderType.Items.Add("融資買進");  comboBoxbstrOrderType.Items.Add("融資賣出");  comboBoxbstrOrderType.Items.Add("融券買進");  comboBoxbstrOrderType.Items.Add("融券賣出");  comboBoxbstrOrderType.Items.Add("無券賣出");  // nTradeType  comboBoxnTradeType.Items.Add("ROD");  comboBoxnTradeType.Items.Add("IOC");  comboBoxnTradeType.Items.Add("FOK");  // nSpecialTradeType  comboBoxnSpecialTradeType.Items.Add("市價");  comboBoxnSpecialTradeType.Items.Add("限價");  // comboBoxnPriceMark  comboBoxnPriceMark.Items.Add("一般定價");  comboBoxnPriceMark.Items.Add("前日收盤價");  comboBoxnPriceMark.Items.Add("漲停");  comboBoxnPriceMark.Items.Add("跌停"); |

|  |
| --- |
| Current Block (可放在buttonSendStockOrder\_Click的Click事件裡) |
| STOCKPROXYORDER pSTOCKPROXYORDER = new STOCKPROXYORDER();  string bstrMessage;  pSTOCKPROXYORDER.bstrFullAccount = comboBoxAccount.Text;  pSTOCKPROXYORDER.bstrStockNo = textBoxStockID.Text;  if (comboBoxPeriod.Text == "盤中") pSTOCKPROXYORDER.nPeriod = 0;  else if (comboBoxPeriod.Text == "零股") pSTOCKPROXYORDER.nPeriod = 1;  else if (comboBoxPeriod.Text == "盤後交易") pSTOCKPROXYORDER.nPeriod = 2;  else if (comboBoxPeriod.Text == "盤中零股") pSTOCKPROXYORDER.nPeriod = 3;  if (comboBoxbstrOrderType.Text == "現股買進") pSTOCKPROXYORDER.bstrOrderType = "1";  else if (comboBoxbstrOrderType.Text == "現股賣出") pSTOCKPROXYORDER.bstrOrderType = "2";  else if (comboBoxbstrOrderType.Text == "融資買進") pSTOCKPROXYORDER.bstrOrderType = "3";  else if (comboBoxbstrOrderType.Text == "融資賣出") pSTOCKPROXYORDER.bstrOrderType = "4";  else if (comboBoxbstrOrderType.Text == "融券買進") pSTOCKPROXYORDER.bstrOrderType = "5";  else if (comboBoxbstrOrderType.Text == "融券賣出") pSTOCKPROXYORDER.bstrOrderType = "6";  else if (comboBoxbstrOrderType.Text == "無券賣出") pSTOCKPROXYORDER.bstrOrderType = "7";  pSTOCKPROXYORDER.bstrPrice = textBoxbstrPrice.Text;  pSTOCKPROXYORDER.nQty = int.Parse(textBoxnQty.Text);  if (comboBoxnTradeType.Text == "ROD") pSTOCKPROXYORDER.nTradeType = 0;  else if (comboBoxnTradeType.Text == "IOC") pSTOCKPROXYORDER.nTradeType = 1;  else if (comboBoxnTradeType.Text == "FOK") pSTOCKPROXYORDER.nTradeType = 2;  if (comboBoxnSpecialTradeType.Text == "市價") pSTOCKPROXYORDER.nSpecialTradeType = 1;  else if (comboBoxnSpecialTradeType.Text == "限價") pSTOCKPROXYORDER.nSpecialTradeType = 2;  if (comboBoxnPriceMark.Text == "一般定價") pSTOCKPROXYORDER.nPriceMark = 0;  else if (comboBoxnPriceMark.Text == "前日收盤價") pSTOCKPROXYORDER.nPriceMark = 1;  else if (comboBoxnPriceMark.Text == "漲停") pSTOCKPROXYORDER.nPriceMark = 2;  else if (comboBoxnPriceMark.Text == "跌停") pSTOCKPROXYORDER.nPriceMark = 3;  // 送出證券委託  int nCode = m\_pSKOrder.SendStockProxyOrder(comboBoxUserID.Text, ref pSTOCKPROXYORDER, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockProxyOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n");  // 取得回傳訊息(bstrMessage)  msg = "【同步委託結果】" + bstrMessage;  richTextBoxMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockProxyAlter

|  |  |  |
| --- | --- | --- |
| 經由proxy server送出證券刪改單。 | | |
| **宣告** | long SendStockProxyAlter([in]BSTR bstrLogInID, [in]struct STOCKPROXYORDER\* pSTOCKPROXYORDER, [out] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pSTOCKPROXYORDER | SKCOM元件中的[STOCKPROXYORDER](#_Proxy_Server_證券刪改單) 物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **Proxy委託通知：**參照[OnProxyOrder](#_OnProxyOrder)。 |
| **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。 | |

|  |
| --- |
| C#範例程式碼\_ TSSKProxyUpdateOrderForm.cs |
| UI示意圖 |
| |  |  | | --- | --- | |  | 拉出5個TextBox，命名(Name)為「textBoxSeqNo」、「textBoxBookNo」、「textBoxCancelOrderByStockNo」、「textBoxStockDecreaseQty」、「textBoxPrice」。 拉出5個ComboBox，命名(Name)為「comboBoxbstrOrderType」、「comboBoxnSpecialTradeType」、「comboBoxPeriod」、「comboBoxTradeType」、「comboBoxnPriceMark」。  拉出 1個Button，命名(Name)為「buttonSendStockProxyAlter」。 | |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |

|  |
| --- |
| Constructor |
| // comboBoxTradeType  {  comboBoxTradeType.Items.Add("0:ROD");  comboBoxTradeType.Items.Add("1:IOC");  comboBoxTradeType.Items.Add("2:FOK");  }  // comboBoxbstrOrderType  {  comboBoxbstrOrderType.Items.Add("刪單");  comboBoxbstrOrderType.Items.Add("改量");  comboBoxbstrOrderType.Items.Add("改價");  }  // nSpecialTradeType  {  comboBoxnSpecialTradeType.Items.Add("市價");  comboBoxnSpecialTradeType.Items.Add("限價");  }  // sPeriod  {  comboBoxPeriod.Items.Add("盤中");  comboBoxPeriod.Items.Add("零股");  comboBoxPeriod.Items.Add("盤後交易");  comboBoxPeriod.Items.Add("盤中零股");  }  // comboBoxnPriceMark  {  comboBoxnPriceMark.Items.Add("一般定價");  comboBoxnPriceMark.Items.Add("前日收盤價");  comboBoxnPriceMark.Items.Add("漲停");  comboBoxnPriceMark.Items.Add("跌停");  } |

|  |
| --- |
| Current Block (可放在buttonSendStockProxyAlter\_Click的Click事件裡) |
| STOCKPROXYORDER pSTOCKPROXYORDER = new STOCKPROXYORDER();  pSTOCKPROXYORDER.bstrStockNo = textBoxCancelOrderByStockNo.Text;  pSTOCKPROXYORDER.bstrBookNo = textBoxBookNo.Text;  pSTOCKPROXYORDER.bstrSeqNo = textBoxSeqNo.Text;  pSTOCKPROXYORDER.bstrFullAccount = comboBoxAccount.Text;  if (comboBoxbstrOrderType.Text == "刪單") pSTOCKPROXYORDER.bstrOrderType = "0";  else if (comboBoxbstrOrderType.Text == "改量") pSTOCKPROXYORDER.bstrOrderType = "1";  else if (comboBoxbstrOrderType.Text == "改價") pSTOCKPROXYORDER.bstrOrderType = "2";  if (comboBoxnSpecialTradeType.Text == "市價") pSTOCKPROXYORDER.nSpecialTradeType = 1;  else if (comboBoxnSpecialTradeType.Text == "限價") pSTOCKPROXYORDER.nSpecialTradeType = 2;  if (comboBoxPeriod.Text == "盤中") pSTOCKPROXYORDER.nPeriod = 0;  else if (comboBoxPeriod.Text == "零股") pSTOCKPROXYORDER.nPeriod = 1;  else if (comboBoxPeriod.Text == "盤後交易") pSTOCKPROXYORDER.nPeriod = 2;  else pSTOCKPROXYORDER.nPeriod = 3;  pSTOCKPROXYORDER.bstrPrice = textBoxPrice.Text; // 委託價格  pSTOCKPROXYORDER.nQty = int.Parse(textBoxStockDecreaseQty.Text); // 股數  if (comboBoxTradeType.Text == "ROD") pSTOCKPROXYORDER.nTradeType = 0;  else if (comboBoxTradeType.Text == "IOC") pSTOCKPROXYORDER.nTradeType = 1;  else if (comboBoxTradeType.Text == "FOK") pSTOCKPROXYORDER.nTradeType = 2;  if (comboBoxnPriceMark.Text == "一般定價") pSTOCKPROXYORDER.nPriceMark = 0;  else if (comboBoxnPriceMark.Text == "前日收盤價") pSTOCKPROXYORDER.nPriceMark = 1;  else if (comboBoxnPriceMark.Text == "漲停") pSTOCKPROXYORDER.nPriceMark = 2;  else if (comboBoxnPriceMark.Text == "跌停") pSTOCKPROXYORDER.nPriceMark = 3;  string bstrMessage; //同步刪單：如果回傳值為 0表示刪單成功，訊息內容則為修改訊息。回傳值非0表示刪單失敗，訊息內容為失敗原因。  // 經由proxy server送出證券刪改單  int nCode = m\_pSKOrder.SendStockProxyAlter(comboBoxUserID.Text, ref pSTOCKPROXYORDER, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockProxyAlter】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + bstrMessage;  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

## 事件

### OnRealBalanceReport

|  |  |  |
| --- | --- | --- |
| 證券即時庫存。透過呼叫[GetRealBalanceReport](#_GetRealBalanceReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnRealBalanceReport([in] BSTR bstrData); | |
| **參數** | bstrData | 每一筆資料以「,」分隔每一個欄位，欄位依序為：   1. 股票代號, 2. 庫存種類, (T:集保 C:融資 L:融券) 3. 資額度(原始), 4. 資額度(可用), 5. 券額度(原始), 6. 券額度(可用), 7. 股數:昨日庫存, 8. 今日委買, 9. 今日委賣, 10. 今日買進成交, 11. 今日賣出成交, 12. 今日資券可回補/集保庫存可賣出, 13. 可資沖股數, 14. 可券沖股數, 15. 即時庫存, 16. **X(此欄請忽略)** 17. **即時個股維持率,** 18. **LOGIN\_ID** 19. **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。 | |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| 拉出1個 DataGridView，命名(Name)為「dataGridViewGetRealBalanceReport」。 |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |

|  |
| --- |
| Constructor |
| //dataGridViewGetRealBalanceReport  {  dataGridViewGetRealBalanceReport.Columns.Add("Column1", "股票代號");  dataGridViewGetRealBalanceReport.Columns.Add("Column2", "庫存種類(T:集保 C:融資 L:融券)");  dataGridViewGetRealBalanceReport.Columns.Add("Column3", "資額度(原始)");  dataGridViewGetRealBalanceReport.Columns.Add("Column4", "資額度(可用)");  dataGridViewGetRealBalanceReport.Columns.Add("Column5", "券額度(原始)");  dataGridViewGetRealBalanceReport.Columns.Add("Column6", "券額度(可用)");  dataGridViewGetRealBalanceReport.Columns.Add("Column7", "股數:昨日庫存");  dataGridViewGetRealBalanceReport.Columns.Add("Column8", "今日委買");  dataGridViewGetRealBalanceReport.Columns.Add("Column9", "今日委賣");  dataGridViewGetRealBalanceReport.Columns.Add("Column10", "今日買進成交");  dataGridViewGetRealBalanceReport.Columns.Add("Column11", "今日賣出成交");  dataGridViewGetRealBalanceReport.Columns.Add("Column12", "今日資券可回補/集保庫存可賣出");  dataGridViewGetRealBalanceReport.Columns.Add("Column13", "可資沖股數");  dataGridViewGetRealBalanceReport.Columns.Add("Column14", "可券沖股數");  dataGridViewGetRealBalanceReport.Columns.Add("Column15", "即時庫存");  dataGridViewGetRealBalanceReport.Columns.Add("Column16", "X(此欄請忽略)");  dataGridViewGetRealBalanceReport.Columns.Add("Column17", "即時個股維持率");  dataGridViewGetRealBalanceReport.Columns.Add("Column18", "LOGIN\_ID");  dataGridViewGetRealBalanceReport.Columns.Add("Column19", "ACCOUNT\_NO");  } |

|  |
| --- |
| Current Block (可放在Form的Load事件裡) |
| //Event  //證券即時庫存。透過呼叫GetRealBalanceReport後，資訊由該事件回傳。  m\_pSKOrder.OnRealBalanceReport += new \_ISKOrderLibEvents\_OnRealBalanceReportEventHandler(OnRealBalanceReport);  void OnRealBalanceReport(string bstrData)  {  // 使用 Split 方法將字串拆分成陣列  string[] values = bstrData.Split(',');  if (values[0] == "970") // 970,Mtalk發生錯誤  {  dataGridViewGetRealBalanceReport.Rows.Add(bstrData);  }  else  {  dataGridViewGetRealBalanceReport.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14], values[15], values[16], values[17]);  }  } |

### OnBalanceQuery

|  |  |  |
| --- | --- | --- |
| 集保庫存查詢。透過呼叫 [GetBalanceQuery](#_GetBalanceQuery)後，資訊由該事件回傳。 | | |
| **宣告** | void OnBalanceQuery ([in] BSTR bstrData); | |
| **參數** | bstrData | 回傳的內容以「,」分隔每一個欄位，欄位依序為：   |  |  | | --- | --- | | 參數 | 說明 | | 證券帳號 | 分公司4碼+帳號7碼 | | 股票代號 |  | | 股票股數 |  | | 融資股數 |  | | 融資金額 |  | | 融券股數 |  | | 融券金額 |  | | 保證金 |  | | 擔保品 |  | |
| **備註** | 回傳【M003】表示沒有庫存 | |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| 拉出1個 DataGridView，命名(Name)為「dataGridViewGetBalanceQuery」。 一張含有 白色, 螢幕擷取畫面, 設計 的圖片  自動產生的描述 |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |

|  |
| --- |
| Constructor |
| //dataGridViewGetBalanceQuery  {  dataGridViewGetBalanceQuery.Columns.Add("Column1", "證券帳號");  dataGridViewGetBalanceQuery.Columns.Add("Column2", "股票代號");  dataGridViewGetBalanceQuery.Columns.Add("Column3", "股票股數");  dataGridViewGetBalanceQuery.Columns.Add("Column4", "融資股數");  dataGridViewGetBalanceQuery.Columns.Add("Column5", "融資金額");  dataGridViewGetBalanceQuery.Columns.Add("Column6", "融券股數");  dataGridViewGetBalanceQuery.Columns.Add("Column7", "融券金額");  dataGridViewGetBalanceQuery.Columns.Add("Column8", "保證金");  dataGridViewGetBalanceQuery.Columns.Add("Column9", "擔保品");  } |

|  |
| --- |
| Current Block (可放在Form的Load事件裡) |
| //Event  //集保庫存查詢。透過呼叫 GetBalanceQuery後，資訊由該事件回傳。  m\_pSKOrder.OnBalanceQuery += new \_ISKOrderLibEvents\_OnBalanceQueryEventHandler(OnBalanceQuery);  void OnBalanceQuery(string bstrData)  {  // 使用 Split 方法將字串拆分成陣列  string[] values = bstrData.Split(',');  if (values[0] == "M003") // 查無資料  {  dataGridViewGetBalanceQuery.Rows.Add(bstrData + "查無資料");  }  else  {  dataGridViewGetBalanceQuery.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8]);  }  } |

### OnMarginPurchaseAmountLimit

|  |  |  |
| --- | --- | --- |
| 資券配額查詢。透過呼叫 [GetMarginPurchaseAmountLimit](#_GetMarginPurchaseAmountLimit_1)後，資訊由該事件回傳。 | | |
| **宣告** | void OnMarginPurchaseAmountLimit ([in] BSTR bstrData); | |
| **參數** | bstrData | 回傳的字串內容【M000】代表成功，接著以「,」分隔每一個欄位，欄位依序為：   |  |  | | --- | --- | | **參數** | **說明** | | 商品代號 |  | | 融資標記 | 0:正常1:停資9:取消信用交易 | | 融資限量 | 0:無配額 9999:無限制 | | 融資比率 |  | | 融券標記 | 0:正常1:停資9:取消信用交易 | | 融券限量 | 0:無配額 9999:無限制 | | 融券比率 |  | | 當沖標記 | 0:不可當沖 1:可當沖 | | 被降成標記 | 0:正常 1:被降成 2:被升成 | | 平盤放空標記 | 0:不可平盤放空 1:可平盤放空 | | 全額交割標記 | 0:正常 1:全額交割 | | 警示標記 | 0:正常1:公司內部警示記號 | | 處置股票標記 | 0:正常1處置股票2:再處置股票  3:彈性處置股票 | | 注意股票標記 | 0:正常1:注意股票 | | 受限股票標記 | 0正常1:證券商委託申報受限股票 | | 異常推介標記 | 0:正常1:異常推介個股 | | 特殊異常標記 | 0:正常1:特殊異常股票 | | 單筆委託張數限制 |  | | 多筆委託張數限制 |  | | 款券預收成數 |  | | **LOGIN\_ID** |  | | **ACCOUNT\_NO** |  | |
| **備註** | 回傳【M003】表示此股票無信用交易資格 | |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| 拉出1個 DataGridView，命名(Name)為「dataGridViewOnMarginPurchaseAmountLimit」。 一張含有 白色, 螢幕擷取畫面, 設計 的圖片  自動產生的描述 |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |

|  |
| --- |
| Constructor |
| //dataGridViewOnMarginPurchaseAmountLimit  {  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column1", "商品代號");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column2", "融資標記(0:正常1:停資9:取消信用交易)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column3", "融資限量(0:無配額 9999:無限制)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column4", "融資比率");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column5", "融券標記(0:正常1:停資9:取消信用交易)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column6", "融券限量(0:無配額 9999:無限制)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column7", "融券比率");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column8", "當沖標記(0:不可當沖 1:可當沖)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column9", "被降成標記(0:正常 1:被降成 2:被升成)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column10", "平盤放空標記(0:不可平盤放空 1:可平盤放空)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column11", "全額交割標記(0:正常 1:全額交割)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column12", "警示標記(0:正常1:公司內部警示記號)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column13", "處置股票標記(0:正常1處置股票2:再處置股票3:彈性處置股票)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column14", "注意股票標記(0:正常1:注意股票)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column15", "受限股票標記(0正常1:證券商委託申報受限股票)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column16", "異常推介標記(0:正常1:異常推介個股)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column17", "特殊異常標記(0:正常1:特殊異常股票)");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column18", "單筆委託張數限制");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column19", "多筆委託張數限制");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column20", "款券預收成數");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column21", "LOGIN\_ID");  dataGridViewOnMarginPurchaseAmountLimit.Columns.Add("Column22", "ACCOUNT\_NO");  } |

|  |
| --- |
| Current Block (可放在Form的Load事件裡) |
| //Event  //資券配額查詢。透過呼叫 GetMarginPurchaseAmountLimit後，資訊由該事件回傳  m\_pSKOrder.OnMarginPurchaseAmountLimit += new \_ISKOrderLibEvents\_OnMarginPurchaseAmountLimitEventHandler(OnMarginPurchaseAmountLimit);  void OnMarginPurchaseAmountLimit(string bstrData)  {  // 使用 Split 方法將字串拆分成陣列  string[] values = bstrData.Split(',');  if (values[0] == "M000") // 成功  {  //dataGridViewOnOverSeaFutureRight.Rows.Add(bstrData);  }  else  {  dataGridViewOnMarginPurchaseAmountLimit.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14], values[15], values[16], values[17], values[18], values[19], values[20], values[21]);  }  } |

### OnProfitLossGWReport

|  |  |  |
| --- | --- | --- |
| 證券新損益查詢結果。透過呼叫[GetProfitLossGWReport](#_GetProfitLossGWReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnProfitLossGWReport ([in] BSTR bstrData); | |
| **參數** | bstrData | 回傳的字串內容，第一筆為查詢結果。  成功時，成功代碼為「000」與訊息，中間以「,」分隔  失敗時，為錯誤代碼與錯誤訊息，中間以「,」分隔。  成功時：  第二筆資料開始就是資料內容，各筆資料以「,」分隔每一個欄位，欄位依序如下表，請參照下表。 |

|  |
| --- |
| C#範例程式碼\_ TSReadOrderForm.cs |
| UI示意圖 |
| 拉出8個 DataGridView，命名(Name)為「dataGridViewOnProfitLossGWReport1」(未實現(彙總))、「dataGridViewOnProfitLossGWReport2」(未實現(明細))、  「dataGridViewOnProfitLossGWReport3」(已實現(彙總))、  「dataGridViewOnProfitLossGWReport4」(已實現(明細))、  「dataGridViewOnProfitLossGWReport5」(已實現(投資總額))、  「dataGridViewOnProfitLossGWReport6」(已實現(彙總(依股票代號)))、「dataGridViewOnProfitLossGWReport7」(現股當沖(彙總))、 「dataGridViewOnProfitLossGWReport8」(現股當沖(明細))。 一張含有 白色, 螢幕擷取畫面, 設計 的圖片  自動產生的描述 |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |

|  |
| --- |
| Constructor |
| //未實現損益  {  //dataGridViewOnProfitLossGWReport1 (彙總)  {  dataGridViewOnProfitLossGWReport1.Columns.Add("Column1", "股票名稱");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column2", "股票代號");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column3", "幣別");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column4", "交易種類");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column5", "庫存股數");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column6", "市價");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column7", "今日市價漲跌");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column8", "市值");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column9", "淨值");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column10", "損益");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column11", "平均買進(券賣)成本");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column12", "付出成本");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column13", "成交價金");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column14", "手續費");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column15", "預估手續費");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column16", "交易稅");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column17", "預估交易稅");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column18", "融資自備款/融券保證金");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column19", "融資金/擔保品");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column20", "預估利息");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column21", "股息(目前此欄無值)");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column22", "試算報酬率");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column23", "未知成本股數");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column24", "備註");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column25", "詳細資料(Y:有資料 N:無資料)");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column26", "排序序號");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column27", "交易種類代號(0:現股3:融資(自)4:融券(自)8:券差9:無券賣出)");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column28", "損益兩平點(此欄只提供交易種類為現股之交易)");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column29", "LOGIN\_ID");  dataGridViewOnProfitLossGWReport1.Columns.Add("Column30", "ACCOUNT\_NO");  }  //dataGridViewOnProfitLossGWReport2 (明細)  {  dataGridViewOnProfitLossGWReport2.Columns.Add("Column1", "股票名稱");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column2", "股票代號");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column3", "成交日期");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column4", "交易種類");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column5", "買進股數");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column6", "可沖股數");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column7", "單價");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column8", "成交價金");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column9", "損益");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column10", "試算報酬率");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column11", "手續費");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column12", "預估手續費");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column13", "交易稅");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column14", "預估交易稅");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column15", "融資自備款/融券保證金");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column16", "融資金額/擔保品金額");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column17", "融資/融券利息");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column18", "融券手續費");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column19", "收付金額");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column20", "股息(目前此欄無值)");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column21", "備註");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column22", "委託書號");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column23", "交易種類代號(配合第4欄交易種類0:現股 3:融資(自) 4:融券(自))");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column24", "LOGIN\_ID");  dataGridViewOnProfitLossGWReport2.Columns.Add("Column25", "ACCOUNT\_NO");  }  }  //已實現損益  {  //dataGridViewOnProfitLossGWReport3 (彙總)  {  dataGridViewOnProfitLossGWReport3.Columns.Add("Column1", "交易種類");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column2", "交易種類代號(0:現股1:融資(代) 2:融券(代) 3:融資(自) 4:融券(自) 5:無券 6:現沖 8:券差)");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column3", "日期(民國年月日)");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column4", "營業單位代碼");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column5", "證券帳號");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column6", "股票代號");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column7", "股數");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column8", "成交價格");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column9", "損益");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column10", "報酬率");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column11", "備註");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column12", "詳細資料(Y:有資料 N:無資料)");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column13", "委託書號(賣)(交易種類是股息，此欄為編號)");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column14", "序號(賣)");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column15", "股票名稱");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column16", "幣別");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column17", "投資額");  dataGridViewOnProfitLossGWReport3.Columns.Add("Column18", "LOGIN\_ID");  }  //dataGridViewOnProfitLossGWReport4 (明細)  {  dataGridViewOnProfitLossGWReport4.Columns.Add("Column1", "日期(民國)");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column2", "交易種類");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column3", "股票代號");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column4", "股票名稱");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column5", "委託書號");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column6", "買賣別");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column7", "股數");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column8", "單價");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column9", "價金");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column10", "手續費");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column11", "交易稅");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column12", "融資自備款/融券保證金");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column13", "融資金額/擔保品金額");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column14", "融資/融券利息");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column15", "融券手續費");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column16", "收付金額");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column17", "已沖銷股數");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column18", "備註");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column19", "交易種類代號(0:現股 1:融資(代) 2:融券(代) 3:融資(自) 4:融券(自))");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column20", "LOGIN\_ID");  dataGridViewOnProfitLossGWReport4.Columns.Add("Column21", "ACCOUNT\_NO");  }  //dataGridViewOnProfitLossGWReport5 (投資總額)  {  dataGridViewOnProfitLossGWReport5.Columns.Add("Column1", "營業單位代碼");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column2", "證券帳號");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column3", "投資總額(台幣)");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column4", "已實現投資損益(台幣)");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column5", "報酬率(台幣)");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column6", "投資總額(人民幣)");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column7", "已實現投資損益(人民幣)");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column8", "報酬率(人民幣)");  dataGridViewOnProfitLossGWReport5.Columns.Add("Column9", "LOGIN\_ID");  }  //dataGridViewOnProfitLossGWReport6 (彙總-依商品代碼資料格式)  {  dataGridViewOnProfitLossGWReport6.Columns.Add("Column1", "交易種類");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column2", "交易種類代號");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column3", "營業單位代碼");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column4", "證券帳號");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column5", "股票代號");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column6", "股數");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column7", "已實現投資損益(台幣)");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column8", "報酬率(台幣)");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column9", "幣別");  dataGridViewOnProfitLossGWReport6.Columns.Add("Column10", "LOGIN\_ID");  }  }  //現股當沖  {  //dataGridViewOnProfitLossGWReport7 (彙總)  {  dataGridViewOnProfitLossGWReport7.Columns.Add("Column1", "序號(2:彙總總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column2", "股票名稱");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column3", "股票代號");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column4", "幣別(中文)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column5", "已沖銷股數");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column6", "待沖銷股數");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column7", "委買中股數");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column8", "委賣中股數");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column9", "可沖銷股數");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column10", "市價");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column11", "今日市價漲跌");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column12", "市值(序號為2時，市值總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column13", "淨值(序號為2時，淨價總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column14", "未實現損益(序號為2時，未實現損益總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column15", "已實現損益(序號為2時，已實現損益總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column16", "損益總計(序號為2時，損益總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column17", "平均單價");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column18", "付出成本(序號為2時，付出成本總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column19", "成交價金(序號為2時，成交價金總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column20", "手續費(序號為2時，手續費總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column21", "預估手續費(序號為2時，預估手續費總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column22", "交易稅(序號為2時，交易稅總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column23", "預估交易稅(序號為2時，預估交易稅總計)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column24", "試算報酬率");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column25", "備註");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column26", "詳細資料(Y:有資料 N:無資料)");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column27", "幣別");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column28", "LOGIN\_ID");  dataGridViewOnProfitLossGWReport7.Columns.Add("Column29", "ACCOUNT\_NO");  }  //dataGridViewOnProfitLossGWReport8 (明細)  {  dataGridViewOnProfitLossGWReport8.Columns.Add("Column1", "股票名稱");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column2", "委託書號");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column3", "買賣股數");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column4", "已沖股數");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column5", "單價");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column6", "成交價金");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column7", "手續費");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column8", "預估手續費");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column9", "交易稅");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column10", "預估交易稅");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column11", "收付金額");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column12", "股票代號");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column13", "買賣別 (B:買; S:賣)");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column14", "LOGIN\_ID");  dataGridViewOnProfitLossGWReport8.Columns.Add("Column15", "ACCOUNT\_NO");  }  } |

|  |
| --- |
| Current Block (可放在Form的Load事件裡) |
| //Event  //證券新損益查詢結果。透過呼叫 GetProfitLossGWReport 後，資訊由該事件回傳  m\_pSKOrder.OnProfitLossGWReport += new \_ISKOrderLibEvents\_OnProfitLossGWReportEventHandler(OnProfitLossGWReport);  void OnProfitLossGWReport(string bstrData)  {  // 使用 Split 方法將字串拆分成陣列  string[] values = bstrData.Split(',');  if (values[0] == "000") // 回傳成功  {  // DO NOTHING  }  else  {  if (comboBoxnTPQueryType.Text == "未實現損益")  {  if (comboBoxnFunc.Text == "彙總") dataGridViewOnProfitLossGWReport1.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14], values[15], values[16], values[17], values[18], values[19], values[20], values[21], values[22], values[23], values[24], values[25], values[26], values[27], values[28], values[29]);  else if (comboBoxnFunc.Text == "明細") dataGridViewOnProfitLossGWReport2.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14], values[15], values[16], values[17], values[18], values[19], values[20], values[21], values[22], values[23], values[24]);  }  else if (comboBoxnTPQueryType.Text == "已實現損益")  {  if (comboBoxnFunc.Text == "彙總") dataGridViewOnProfitLossGWReport3.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14], values[15], values[16], values[17]);  else if (comboBoxnFunc.Text == "明細") dataGridViewOnProfitLossGWReport4.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14], values[15], values[16], values[17], values[18], values[19], values[20]);  else if (comboBoxnFunc.Text == "投資總額") dataGridViewOnProfitLossGWReport5.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8]);  else if (comboBoxnFunc.Text == "彙總(依股票代號)") dataGridViewOnProfitLossGWReport6.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9]);  }  else if (comboBoxnTPQueryType.Text == "現股當沖")  {  if (comboBoxnFunc.Text == "彙總") dataGridViewOnProfitLossGWReport7.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14], values[15], values[16], values[17], values[18], values[19], values[20], values[21], values[22], values[23], values[24], values[25], values[26], values[27], values[28]);  else if (comboBoxnFunc.Text == "明細") dataGridViewOnProfitLossGWReport8.Rows.Add(values[0], values[1], values[2], values[3], values[4], values[5], values[6], values[7], values[8], values[9], values[10], values[11], values[12], values[13], values[14]);  }  }  } |

#### 未實現

|  |  |  |
| --- | --- | --- |
| 證券新損益查詢結果。透過呼叫 [GetProfitLossGWReport](#_4-2-73_GetProfitLossGWReport) 後，資訊由該事件回傳。  (未實現) | | |
| **宣告** | void OnProfitLossGWReport ([in] BSTR bstrData); | |
| **參數** | bstrData | 回傳的字串內容，第一筆為查詢結果。  成功代碼為「000」與訊息。  成功時：  第二筆資料開始就是資料內容  回傳的內容以「,」分隔每一個欄位，欄位依序為：  (彙總資料格式)   |  |  |  | | --- | --- | --- | |  | 參數 | 說明 | | 1 | 股票名稱 |  | | 2 | 股票代號 |  | | 3 | 幣別 |  | | 4 | 交易種類 |  | | 5 | 庫存股數 |  | | 6 | 市價 |  | | 7 | 今日市價漲跌 |  | | 8 | 市值 |  | | 9 | 淨值 |  | | 10 | 損益 |  | | 11 | 平均買進(券賣)成本 |  | | 12 | 付出成本 |  | | 13 | 成交價金 |  | | 14 | 手續費 |  | | 15 | 預估手續費 |  | | 16 | 交易稅 |  | | 17 | 預估交易稅 |  | | 18 | 融資自備款/融券保證金 |  | | 19 | 融資金/擔保品 |  | | 20 | 預估利息 |  | | 21 | 股息 | 目前此欄無值 | | 22 | 試算報酬率 |  | | 23 | 未知成本股數 |  | | 24 | 備註 |  | | 25 | 詳細資料 | Y:有資料 N:無資料 | | 26 | 排序序號 |  | | 27 | 交易種類代號 | 0:現股 3:融資(自) 4:融券(自) 8:券差 9:無券賣出 | | 28 | 損益兩平點 | 此欄只提供**交易種類為現股**之交易 | | 29 | LOGIN\_ID |  | | 30 | ACCOUNT\_NO |  |   (明細資料格式)   |  |  |  | | --- | --- | --- | |  | 參數 | 說明 | | 1 | 股票名稱 |  | | 2 | 股票代號 |  | | 3 | 成交日期 |  | | 4 | 交易種類 |  | | 5 | 買進股數 |  | | 6 | 可沖股數 |  | | 7 | 單價 |  | | 8 | 成交價金 |  | | 9 | 損益 |  | | 10 | 試算報酬率 |  | | 11 | 手續費 |  | | 12 | 預估手續費 |  | | 13 | 交易稅 |  | | 14 | 預估交易稅 |  | | 15 | 融資自備款/融券保證金 |  | | 16 | 融資金額/擔保品金額 |  | | 17 | 融資/融券利息 |  | | 18 | 融券手續費 |  | | 19 | 收付金額 |  | | 20 | 股息 | 目前此欄無值 | | 21 | 備註 |  | | 22 | 委託書號 |  | | 23 | 交易種類代號 | 配合第4欄交易種類  0:現股 3:融資(自) 4:融券(自) | | 24 | LOGIN\_ID |  | | 25 | 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。 | |

#### 已實現

|  |  |  |
| --- | --- | --- |
| 證券新損益查詢結果。透過呼叫 [GetProfitLossGWReport](#_4-2-73_GetProfitLossGWReport) 後，資訊由該事件回傳。  (已實現) | | |
| **宣告** | void OnProfitLossGWReport ([in] BSTR bstrData); | |
| **參數** | bstrData | 回傳的字串內容，第一筆為查詢結果。  成功代碼為「000」與訊息。  成功時：  第二筆資料開始就是資料內容  回傳的內容以「,」分隔每一個欄位，欄位依序為：  (彙總資料格式)   |  |  |  | | --- | --- | --- | |  | 參數 | 說明 | | 1 | 交易種類 | 現股、 融資(代) 、融券(代)、 融資(自) 、融券(自) 、無券、 現沖 、券差、**股息** | | 2 | 交易種類代號 | 0:現股1:融資(代) 2:融券(代) 3:融資(自) 4:融券(自) 5:無券 6:現沖 8:券差 | | 3 | 日期 | 民國年月日 | | 4 | 營業單位代碼 |  | | 5 | 證券帳號 |  | | 6 | 股票代號 |  | | 7 | 股數 |  | | 8 | 成交價格 |  | | 9 | 損益 |  | | 10 | 報酬率 |  | | 11 | 備註 |  | | 12 | 詳細資料 | Y:有資料 N:無資料 | | 13 | 委託書號(賣) | 交易種類 是股息，此欄為編號 | | 14 | 序號(賣) |  | | 15 | 股票名稱 |  | | 16 | 幣別 |  | | 17 | 投資額 |  | | 18 | LOGIN\_ID |  |   (明細資料格式)   |  |  |  | | --- | --- | --- | |  | 參數 | 說明 | | 1 | 日期 | 民國 | | 2 | 交易種類 |  | | 3 | 股票代號 |  | | 4 | 股票名稱 |  | | 5 | 委託書號 |  | | 6 | 買賣別 |  | | 7 | 股數 |  | | 8 | 單價 |  | | 9 | 價金 |  | | 10 | 手續費 |  | | 11 | 交易稅 |  | | 12 | 融資自備款/融券保證金 |  | | 13 | 融資金額/擔保品金額 |  | | 14 | 融資/融券利息 |  | | 15 | 融券手續費 |  | | 16 | 收付金額 |  | | 17 | 已沖銷股數 |  | | 18 | 備註 |  | | 19 | 交易種類代號 | 0:現股 1:融資(代) 2:融券(代) 3:融資(自) 4:融券(自) | | 20 | LOGIN\_ID |  | | 21 | ACCOUNT\_NO |  |   (投資總額)   |  |  |  | | --- | --- | --- | |  | 參數 | 說明 | | 1 | 營業單位代碼 |  | | 2 | 證券帳號 |  | | 3 | 投資總額 | 台幣 | | 4 | 已實現投資損益 | 台幣 | | 5 | 報酬率 | 台幣 | | 6 | 投資總額 | 人民幣 | | 7 | 已實現投資損益 | 人民幣 | | 8 | 報酬率 | 人民幣 | | 9 | LOGIN\_ID |  |   (彙總-依商品代碼資料格式)   |  |  |  | | --- | --- | --- | |  | 參數 | 說明 | | 1 | 交易種類 |  | | 2 | 交易種類代號 | 配合第1欄交易種類 | | 3 | 營業單位代碼 |  | | 4 | 證券帳號 |  | | 5 | 股票代號 |  | | 6 | 股數 |  | | 7 | 已實現投資損益 | 台幣 | | 8 | 報酬率 | 台幣 | | 9 | 幣別 |  | | 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。 | |

#### 現股當沖

|  |  |  |
| --- | --- | --- |
| 證券新損益查詢結果。透過呼叫 [GetProfitLossGWReport](#_4-2-73_GetProfitLossGWReport) 後，資訊由該事件回傳。  (現股當沖) | | |
| **宣告** | void OnProfitLossGWReport ([in] BSTR bstrData); | |
| 參數 | bstrData | 回傳的字串內容，第一筆為查詢結果。  成功代碼為「000」與訊息。  成功時：  第二筆資料開始就是資料內容  回傳的內容以「,」分隔每一個欄位，欄位依序為：  (彙總資料格式)   |  |  |  | | --- | --- | --- | |  | 參數 | 說明 | | 1 | 序號 | 2 : 彙總總計 | | 2 | 股票名稱 |  | | 3 | 股票代號 |  | | 4 | 幣別(中文) |  | | 5 | 已沖銷股數 |  | | 6 | 待沖銷股數 |  | | 7 | 委買中股數 |  | | 8 | 委賣中股數 |  | | 9 | 可沖銷股數 |  | | 10 | 市價 |  | | 11 | 今日市價漲跌 |  | | 12 | 市值 | 序號為2時，市值總計 | | 13 | 淨值 | 序號為2時，淨價總計 | | 14 | 未實現損益 | 序號為2時，未實現損益總計 | | 15 | 已實現損益 | 序號為2時，已實現損益總計 | | 16 | 損益總計 | 序號為2時，損益總計 | | 17 | 平均單價 |  | | 18 | 付出成本 | 序號為2時，付出成本總計 | | 19 | 成交價金 | 序號為2時，成交價金總計 | | 20 | 手續費 | 序號為2時，手續費總計 | | 21 | 預估手續費 | 序號為2時，預估手續費總計 | | 22 | 交易稅 | 序號為2時，交易稅總計 | | 23 | 預估交易稅 | 序號為2時，預估交易稅總計 | | 24 | 試算報酬率 |  | | 25 | 備註 |  | | 26 | 詳細資料 | Y:有資料 N:無資料 | | 27 | 幣別 |  | | 28 | LOGIN\_ID |  | | 29 | ACCOUNT\_NO |  |   (明細資料格式)：   |  |  |  | | --- | --- | --- | |  | 參數 |  | | 1 | 股票名稱 |  | | 2 | 委託書號 |  | | 3 | 買賣股數 |  | | 4 | 已沖股數 |  | | 5 | 單價 |  | | 6 | 成交價金 |  | | 7 | 手續費 |  | | 8 | 預估手續費 |  | | 9 | 交易稅 |  | | 10 | 預估交易稅 |  | | 11 | 收付金額 |  | | 12 | 股票代號 |  | | **13** | **買賣別 (B:買; S:賣)** | **V2.13.37起新增** | | 14 | LOGIN\_ID |  | | 15 | ACCOUNT\_NO |  | |

### 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 | |

|  |
| --- |
| C#範例程式碼\_TSSendOrderForm.cs |
| UI示意圖 |
| 拉出1個 RichTextBox，命名(Name)為「richTextBoxMessage」。 |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件  // 是否為非同步委託  bool bAsyncOrder = false; |
| Current Block (可放在Form的Load事件裡) |
| //Event  // 非同步委託結果  m\_pSKOrder.OnAsyncOrder += new \_ISKOrderLibEvents\_OnAsyncOrderEventHandler(OnAsyncOrder);  void OnAsyncOrder(int nThreadID, int nCode, string bstrMessage)  {  // 取得回傳訊息  string msg = "TID:" + nThreadID + "收單訊息:" + bstrMessage;  msg = "【非同步委託結果】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + msg;  richTextBoxMessage.AppendText(msg + "\n");  } |

### 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  (下列兩欄僅證券下單才回)  Action: 處理動作，'S'：需圈股票、'A'：需圈錢，其餘空白  BlockNum: 需圈存的金額或股數  Action為'S'或'A'才有值，其餘空白 | | |

\*v2.13.47異動：補回傳參數nStampID文字說明

\*v2.13.48異動：新增證券下單回傳欄位

|  |
| --- |
| C#範例程式碼\_TSSKProxySendOrderForm.cs |
| UI示意圖 |
| 拉出1個 RichTextBox，命名(Name)為「richTextBoxMessage」。 |
| 全域變數 |
| // 宣告物件  SKCenterLib m\_pSKCenter = new SKCenterLib(); //登入&環境設定物件  SKOrderLib m\_pSKOrder = new SKOrderLib(); //下單物件 |
| Current Block (可放在Form的Load事件裡) |
| //Event  // Proxy委託結果  m\_pSKOrder.OnProxyOrder += new \_ISKOrderLibEvents\_OnProxyOrderEventHandler(OnProxyOrder);  void OnProxyOrder(int nStampID, int nCode, string bstrMessage)  {  // 取得回傳訊息  string msg = "Time Stamp:" + nStampID + "收單訊息:" + bstrMessage;  msg = "【Proxy委託結果】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode) + msg;  richTextBoxMessage.AppendText(msg + "\n");  } |

## Struct 結構物件

### TSPROFITLOSSGWQUERY(新損益試算查詢物件)

#### **未實現**

struct TSPROFITLOSSGWQUERY

{

BSTR bstrFullAccount; //證券帳號，分公司四碼＋帳號7碼

LONG nTPQueryType; //損益類別 0: 未實現損益；1: 已實現損益 2:現股當沖

LONG nFunc; //損益回傳格式 //(未實現) 0:彙總 ；1:明細

BSTR bstrStockNo; //當nFunc 為 1:明細 時,指定商品代碼

BSTR bstrTradeType; //當nFunc 為 1:明細 時,指定交易類別

//0:現股 3:融資(自) 4:融券(自) 8:券差 9:無券賣出

};

#### **已實現**

struct TSPROFITLOSSGWQUERY

{

BSTR bstrFullAccount; //證券帳號，分公司四碼＋帳號7碼

LONG nTPQueryType; //損益類別 1: 已實現損益

LONG nFunc; //損益回傳格式

// 0:彙總；1:明細; 2:投資總額; 3: 彙總(依股票代號)

BSTR bstrStockNo; 　//當nFunc 為 1:明細 及3: 彙總依股票代號 時，指定商品代碼

　　BSTR bstrTradeType; //當nFunc 為 1:明細，指定交易類別

//0:現股 1:融資(代) 2:融券(代) 3:融資(自) 4:融券(自)

BSTR bstrStartDate;

//當nFunc 為 0:彙總 時，起始日**西元年月日**//YYYYMMDD

//當nFunc 為 1:明細 及 3:彙總依股票代號 時，起始日**民國年月日**

BSTR bstrEndDate; //當nFunc 為 0:彙總 時，結束日**西元年月日**//YYYYMMDD

//當nFunc 為 1:明細格式時,**不**需填結束日

//當nFunc 3:彙總依股票代號 時，結束日 **民國年月日**

BSTR bstrBookNo; //當nFunc 為 1:明細 時,指定委託書號

BSTR bstrSeqNo; //當nFunc 為 1:明細 時,指定序號

};

#### **現股當沖:(限查閱當日交易)**

struct TSPROFITLOSSGWQUERY

{

BSTR bstrFullAccount; //證券帳號，分公司四碼＋帳號7碼

LONG nTPQueryType; //損益類別 2:現股當沖

LONG nFunc; //損益回傳格式 //(現股當沖) 1: 彙總 2: 明細

BSTR bstrStockNo; //當nFunc 為 2時:明細時,需指定商品代碼

};

### STOCKORDER (證券下單物件)

struct STOCKORDER

{

BSTR bstrFullAccount; //證券帳號，分公司代碼＋帳號7碼

BSTR bstrStockNo; //委託股票代號

SHORT sPrime; //0:上市上櫃1:興櫃

SHORT sPeriod; //0:盤中 1:盤後 2:零股

SHORT sFlag; //0:現股 1:融資 2:融券 3:無券

SHORT sBuySell; //0:買進 1:賣出

BSTR bstrPrice; //委託價格，「M」表示參考價（昨收價）

LONG nQty; //整股交易為張數，如果是零股則為股數

LONG nTradeType; //[證券逐筆交易]**0:ROD ; 1:IOC ; 2:FOK**

LONG nSpecialTradeType; //[證券逐筆交易]**1:市價; 2:限價**

(市價單之委託價格Price請給0; 限價單之委託價格Price 不可為0)

};

#### **盤中零股**

struct STOCKORDER

{

BSTR bstrFullAccount; //證券帳號，分公司代碼＋帳號7碼

BSTR bstrStockNo; //委託股票代號

SHORT sPeriod; // 4:盤中零股

SHORT sFlag; //0:現股

SHORT sBuySell; //0:買進 1:賣出

BSTR bstrPrice; //委託價格

LONG nQty; //零股數量為股數1~999股

};

### Proxy Server 證券

#### **Proxy Server 證券下單**

struct STOCKPROXYORDER

{

BSTR bstrFullAccount; //證券帳號，分公司代碼＋帳號7碼

BSTR bstrStockNo; //委託股票代號

BSTR bstrOrderType; //下單類別

(1: 現股買進。2: 現股賣出。3: 融資買進。4: 融資賣出。5: 融券買進。6: 融券賣出。7: 無券賣出)

LONG nSpecialTradeType; // 1:市價 2:限價

LONG nPeriod; //0:盤中 1:零股 2:盤後交易 3:盤中零股

BSTR bstrPrice; //委託價格

LONG nQty; //股數

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

LONG nPriceMark; //價格旗標 0:一般定價 1:前日收盤價 2:漲停 3:跌停

};

#### **Proxy Server 證券刪改單**

struct STOCKPROXYORDER

{

BSTR bstrFullAccount; //證券帳號，分公司代碼＋帳號7碼

BSTR bstrStockNo; //委託股票代號

BSTR bstrOrderType; //下單類別 (0:刪單。1:改量。2:改價。)

LONG nSpecialTradeType; // 1:市價 2:限價

LONG nPeriod; //0:盤中 1:零股 2:盤後交易 3:盤中零股

BSTR bstrPrice; //委託價格

(改價:必填，帶””表示市價。減量:帶0，刪單:帶””或是原始價)

LONG nQty; //股數

(改價:帶0。減量:帶要減的量。刪單:帶原始量)

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

LONG nPriceMark; //價格旗標 0:一般定價 1:前日收盤價 2:漲停 3:跌停

BSTR bstrBookNo; //委託書號

BSTR bstrSeqNo; //委託序號

};