策略王COM元件使用說明

-下單-證券智慧單-

## 函式範例

### SKOrderLib(國內證券)

*函式*

1. 智慧單

|  |  |  |
| --- | --- | --- |
| **功能** | **函式名稱** | **備註** |
| 1. **證券智慧單當沖條件委託** | [SendStockStrategyDayTrade](#_SendStockStrategyDayTrade) | 簽署證券智慧單風險預告書 |
| 1. **證券智慧單出清條件委託** | [SendStockStrategyClear](#_SendStockStrategyClear_1) |  |
| 1. **證券智慧單MIT條件委託** | [SendStockStrategyMIT](#_SendStockStrategyMIT_1) |  |
| 1. **證券智慧單OCO條件委託** | [SendStockStrategyOCO](#_SendStockStrategyOCO_1) |  |
| 1. **證券智慧單MIOC條件委託** | [SendStockStrategyMIOC](#_SendStockStrategyMIOC_1) |  |
| 1. **證券智慧單MST條件委託** | [SendStockStrategyMST](#_SendStockStrategyMST_1) |  |
| 1. **證券AB單委託** | [SendStockStrategyAB](#_SendStockStrategyAB_1) |  |
| 1. **證券自組單委託** | [SendStockStrategyCB](#_SendStockStrategyCB_1) |  |
| 1. **取消證券智慧單委託** | |  | | --- | | **新版**[CancelTSStrategyOrderV1](#_CancelTSStrategyOrderV1_1) | | [CancelTSStrategyOrder](#_CancelTSStrategyOrder_1) | |  |
| 1. **取消多筆智慧單委託** | [CancelStrategyList](#_CancelStrategyList) |  |
| 1. **證券智慧單被動查詢** | [GetTSSmartStrategyReport](#_GetTSSmartStrategyReport) |  |

*事件*

|  |  |  |
| --- | --- | --- |
| **功能** | **事件名稱** | **備註** |
| 1. **證券智慧單資料** | [OnTSSmartStrategyReport](#_OnTSSmartStrategyReport) |  |
| 1. **非同步委託結果** | [OnAsyncOrder](#_OnAsyncOrder_1) |  |

#### 智慧單

##### SendStockStrategyDayTrade

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單當沖條件委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyDayTrade ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDER**\* 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**[STOCKSTRATEGYORDER](#_證券當沖)**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，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。**  **其他錯誤由「智慧單主機」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_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 | |

|  |
| --- |
| C#範例程式碼\_TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出13個TextBox，命名(Name)為「textBoxbstrStockNoDT」、「textBoxnQtyDT」、「textBoxbstrOrderPriceDT」、「textBoxbstrMITTriggerPriceDT」、「textBoxbstrMITDealPriceDT」、「textBoxbstrSLTrigger」、「textBoxbstrSLPercentDT」、「textBoxbstrSLOrderPriceDT」、「textBoxbstrTPTrigger」、「textBoxbstrTPPercent」、「textBoxbstrTPOrderPriceDT」、「textBoxbstrClearCancelTimeDT」、「textBoxbstrClearAllOrderPriceDT」。 拉出17個ComboBox，命名(Name)為「comboBoxnBuySellDT」、「comboBoxnOrderPriceCondDT」、「comboBoxnOrderPriceTypeDT」、「comboBoxnInnerOrderIsMITDT」、「comboBoxnMITDirDT」、「comboBoxnStopLossFlagDT」、「comboBoxnRDOSLPercent」、「comboBoxnRDSLMarketPriceType」、「comboBoxnStopLossOrderCond」、「comboBoxnTakeProfitFlagDT」、「comboBoxnRDOTPPercent」、「comboBoxnRDTPMarketPriceType」、「comboBoxnTakeProfitOrderCond」、「comboBoxnClearAllFlagDT」、「comboBoxnClearAllPriceTypeDT」、「comboBoxnClearOrderCondDT」、「comboBoxnFinalClearFlagDT」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyDayTrade」。 | |
| 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 |
| // DayTrade  {  // comboBoxnBuySellDT  {  comboBoxnBuySellDT.Items.Add("0:現股買");  comboBoxnBuySellDT.Items.Add("1:無券賣出");  }  // comboBoxnOrderPriceCondDT  {  comboBoxnOrderPriceCondDT.Items.Add("0:ROD");  comboBoxnOrderPriceCondDT.Items.Add("3:IOC");  comboBoxnOrderPriceCondDT.Items.Add("4:FOK");  }  // comboBoxnOrderPriceTypeDT  {  comboBoxnOrderPriceTypeDT.Items.Add("1:市價");  comboBoxnOrderPriceTypeDT.Items.Add("2:限價");  }  // comboBoxnInnerOrderIsMITDT  {  comboBoxnInnerOrderIsMITDT.Items.Add("0:N");  comboBoxnInnerOrderIsMITDT.Items.Add("1:Y");  }  // comboBoxnMITDirDT  {  comboBoxnMITDirDT.Items.Add("0:未啟用MIT");  comboBoxnMITDirDT.Items.Add("1:向上觸發(大於等於)");  comboBoxnMITDirDT.Items.Add("2:向下觸發(小於等於)");  }  // comboBoxnClearAllFlagDT  {  comboBoxnClearAllFlagDT.Items.Add("0:否");  comboBoxnClearAllFlagDT.Items.Add("1:是");  }  // comboBoxnClearAllPriceTypeDT  {  comboBoxnClearAllPriceTypeDT.Items.Add("1:市價");  comboBoxnClearAllPriceTypeDT.Items.Add("2:限價");  }  // comboBoxnFinalClearFlagDT  {  comboBoxnFinalClearFlagDT.Items.Add("0:否");  comboBoxnFinalClearFlagDT.Items.Add("1:是");  }  // comboBoxnTakeProfitFlagDT  {  comboBoxnTakeProfitFlagDT.Items.Add("0:否");  comboBoxnTakeProfitFlagDT.Items.Add("1:是");  }  // comboBoxnStopLossFlagDT  {  comboBoxnStopLossFlagDT.Items.Add("0:否");  comboBoxnStopLossFlagDT.Items.Add("1:是");  }  // comboBoxnClearOrderCondDT  {  comboBoxnClearOrderCondDT.Items.Add("0:ROD");  comboBoxnClearOrderCondDT.Items.Add("3:IOC");  comboBoxnClearOrderCondDT.Items.Add("4:FOK");  }  // comboBoxnRDOTPPercent  {  comboBoxnRDOTPPercent.Items.Add("0:觸發價");  comboBoxnRDOTPPercent.Items.Add("1:漲幅");  }  // comboBoxnRDTPMarketPriceType  {  comboBoxnRDTPMarketPriceType.Items.Add("1:市價");  comboBoxnRDTPMarketPriceType.Items.Add("2:限價");  }  // comboBoxnRDOSLPercent  {  comboBoxnRDOSLPercent.Items.Add("0:觸發價");  comboBoxnRDOSLPercent.Items.Add("1:漲跌幅");  }  // comboBoxnRDSLMarketPriceType  {  comboBoxnRDSLMarketPriceType.Items.Add("1:市價");  comboBoxnRDSLMarketPriceType.Items.Add("2:限價");  }  // comboBoxnTakeProfitOrderCond  {  comboBoxnTakeProfitOrderCond.Items.Add("0:ROD");  comboBoxnTakeProfitOrderCond.Items.Add("3:IOC");  comboBoxnTakeProfitOrderCond.Items.Add("4:FOK");  }  // comboBoxnStopLossOrderCond  {  comboBoxnStopLossOrderCond.Items.Add("0:ROD");  comboBoxnStopLossOrderCond.Items.Add("3:IOC");  comboBoxnStopLossOrderCond.Items.Add("4:FOK");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyDayTrade\_Click的Click事件裡) |
| STOCKSTRATEGYORDER pOrder = new STOCKSTRATEGYORDER();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoDT.Text;  pOrder.nQty = int.Parse(textBoxnQtyDT.Text);  pOrder.bstrOrderPrice = textBoxbstrOrderPriceDT.Text;  if (comboBoxnBuySellDT.Text == "0:現股買") pOrder.nBuySell = 0;  else if (comboBoxnBuySellDT.Text == "1:無券賣出") pOrder.nBuySell = 1;  if (comboBoxnOrderPriceCondDT.Text == "0:ROD") pOrder.nOrderPriceCond = 0;  else if (comboBoxnOrderPriceCondDT.Text == "3:IOC") pOrder.nOrderPriceCond = 3;  else if (comboBoxnOrderPriceCondDT.Text == "4:FOK") pOrder.nOrderPriceCond = 4;  if (comboBoxnOrderPriceTypeDT.Text == "1:市價") pOrder.nOrderPriceType = 1;  else if (comboBoxnOrderPriceTypeDT.Text == "2:限價") pOrder.nOrderPriceType = 2;  if (comboBoxnInnerOrderIsMITDT.Text == "0:N") pOrder.nInnerOrderIsMIT = 0;  else if (comboBoxnInnerOrderIsMITDT.Text == "1:Y") pOrder.nInnerOrderIsMIT = 1;  if (comboBoxnMITDirDT.Text == "0:未啟用MIT") pOrder.nMITDir = 0;  else if (comboBoxnMITDirDT.Text == "1:向上觸發(大於等於)") pOrder.nMITDir = 1;  else if (comboBoxnMITDirDT.Text == "2:向下觸發(小於等於)") pOrder.nMITDir = 2;  pOrder.bstrMITTriggerPrice = textBoxbstrMITTriggerPriceDT.Text;  pOrder.bstrMITDealPrice = textBoxbstrMITDealPriceDT.Text;  if (comboBoxnClearAllFlagDT.Text == "0:否") pOrder.nClearAllFlag = 0;  else if (comboBoxnClearAllFlagDT.Text == "1:是") pOrder.nClearAllFlag = 1;  pOrder.bstrClearCancelTime = textBoxbstrClearCancelTimeDT.Text;  if (comboBoxnClearAllPriceTypeDT.Text == "1:市價") pOrder.nClearAllPriceType = 1;  else if (comboBoxnClearAllPriceTypeDT.Text == "2:限價") pOrder.nClearAllPriceType = 2;  pOrder.bstrClearAllOrderPrice = textBoxbstrClearAllOrderPriceDT.Text;  if (comboBoxnFinalClearFlagDT.Text == "0:否") pOrder.nFinalClearFlag = 0;  else if (comboBoxnFinalClearFlagDT.Text == "1:是") pOrder.nFinalClearFlag = 1;  if (comboBoxnTakeProfitFlagDT.Text == "0:否") pOrder.nTakeProfitFlag = 0;  else if (comboBoxnTakeProfitFlagDT.Text == "1:是") pOrder.nTakeProfitFlag = 1;  if (comboBoxnRDOTPPercent.Text == "0:觸發價") pOrder.nRDOTPPercent = 0;  else if (comboBoxnRDOTPPercent.Text == "1:漲幅") pOrder.nRDOTPPercent = 1;  pOrder.bstrTPPercent = textBoxbstrTPPercent.Text;  pOrder.bstrTPTrigger = textBoxbstrTPTrigger.Text;  if (comboBoxnRDTPMarketPriceType.Text == "1:市價") pOrder.nRDTPMarketPriceType = 1;  else if (comboBoxnRDTPMarketPriceType.Text == "2:限價") pOrder.nRDTPMarketPriceType = 2;  pOrder.bstrTPOrderPrice = textBoxbstrTPOrderPriceDT.Text;  if (comboBoxnStopLossFlagDT.Text == "0:否") pOrder.nStopLossFlag = 0;  else if (comboBoxnStopLossFlagDT.Text == "1:是") pOrder.nStopLossFlag = 1;  if (comboBoxnRDOSLPercent.Text == "0:觸發價") pOrder.nRDOSLPercent = 0;  else if (comboBoxnRDOSLPercent.Text == "1:漲跌幅") pOrder.nRDOSLPercent = 1;  pOrder.bstrSLPercent = textBoxbstrSLPercentDT.Text;  pOrder.bstrSLTrigger = textBoxbstrSLTrigger.Text;  if (comboBoxnRDSLMarketPriceType.Text == "1:市價") pOrder.nRDSLMarketPriceType = 1;  else if (comboBoxnRDSLMarketPriceType.Text == "2:限價") pOrder.nRDSLMarketPriceType = 2;  pOrder.bstrSLOrderPrice = textBoxbstrSLOrderPriceDT.Text;  if (comboBoxnTakeProfitOrderCond.Text == "0:ROD") pOrder.nTakeProfitOrderCond = 0;  else if (comboBoxnTakeProfitOrderCond.Text == "3:IOC") pOrder.nTakeProfitOrderCond = 3;  else if (comboBoxnTakeProfitOrderCond.Text == "4:FOK") pOrder.nTakeProfitOrderCond = 4;  if (comboBoxnStopLossOrderCond.Text == "0:ROD") pOrder.nStopLossOrderCond = 0;  else if (comboBoxnStopLossOrderCond.Text == "3:IOC") pOrder.nStopLossOrderCond = 3;  else if (comboBoxnStopLossOrderCond.Text == "4:FOK") pOrder.nStopLossOrderCond = 4;  if (comboBoxnClearOrderCondDT.Text == "0:ROD") pOrder.nClearOrderCond = 0;  else if (comboBoxnClearOrderCondDT.Text == "3:IOC") pOrder.nClearOrderCond = 3;  else if (comboBoxnClearOrderCondDT.Text == "4:FOK") pOrder.nClearOrderCond = 4;  string bstrMessage;  // 送出證券智慧單當沖條件委託。  int nCode = m\_pSKOrder.SendStockStrategyDayTrade(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyDayTrade】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockStrategyClear

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單出清條件委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyDayTrade ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDEROUT**\* 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**[STOCKSTRATEGYORDEROUT](#_STOCKSTRATEGYORDEROUT(證券智慧單物件-出清)_1)**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，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。錯誤代碼可參考對照表。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 | |

|  |
| --- |
| C#範例程式碼\_ TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出8個TextBox，命名(Name)為「textBoxbstrStockNoC」、「textBoxnQtyC」、「textBoxbstrGTETriggerPrice」、「textBoxbstrGTEOrderPrice」、「textBoxbstrLTETriggerPrice」、「textBoxbstrLTEOrderPrice」、「textBoxbstrClearCancelTimeC」、「textBoxbstrClearAllOrderPriceC」。 拉出12個ComboBox，命名(Name)為「comboBoxnBuySellC」、「comboBoxnOrderTypeC」、「comboBoxnGTEFlag」、「comboBoxnGTEMarketPrice」、「comboBoxnGTEOrderCond」、「comboBoxnLTEFlag」、「comboBoxnLTEMarketPrice」、「comboBoxnLTEOrderCond」、「comboBoxnClearAllFlagC」、「comboBoxnClearAllPriceTypeC」、「comboBoxnClearOrderCondC」、「comboBoxnFinalClearFlagC」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyClear」。 | |
| 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 |
| // ClearOut  {  // comboBoxnBuySellC  {  comboBoxnBuySellC.Items.Add("0:買");  comboBoxnBuySellC.Items.Add("1:賣");  }  // comboBoxnClearAllFlagC  {  comboBoxnClearAllFlagC.Items.Add("0:否");  comboBoxnClearAllFlagC.Items.Add("1:是");  }  // comboBoxnClearAllPriceTypeC  {  comboBoxnClearAllPriceTypeC.Items.Add("1:市價");  comboBoxnClearAllPriceTypeC.Items.Add("2:限價");  }  // comboBoxnFinalClearFlagC  {  comboBoxnFinalClearFlagC.Items.Add("0:否");  comboBoxnFinalClearFlagC.Items.Add("1:是");  }  // comboBoxnOrderTypeC  {  comboBoxnOrderTypeC.Items.Add("0:現股");  comboBoxnOrderTypeC.Items.Add("3:融資");  comboBoxnOrderTypeC.Items.Add("4:融券");  }  // comboBoxnClearOrderCondC  {  comboBoxnClearOrderCondC.Items.Add("0:ROD");  comboBoxnClearOrderCondC.Items.Add("3:IOC");  comboBoxnClearOrderCondC.Items.Add("4:FOK");  }  // comboBoxnLTEFlag  {  comboBoxnLTEFlag.Items.Add("0:否");  comboBoxnLTEFlag.Items.Add("1:是");  }  // comboBoxnLTEMarketPrice  {  comboBoxnLTEMarketPrice.Items.Add("1:市價");  comboBoxnLTEMarketPrice.Items.Add("2:限價");  }  // comboBoxnGTEFlag  {  comboBoxnGTEFlag.Items.Add("0:否");  comboBoxnGTEFlag.Items.Add("1:是");  }  // comboBoxnGTEMarketPrice  {  comboBoxnGTEMarketPrice.Items.Add("1:市價");  comboBoxnGTEMarketPrice.Items.Add("2:限價");  }  // comboBoxnGTEOrderCond  {  comboBoxnGTEOrderCond.Items.Add("0:ROD");  comboBoxnGTEOrderCond.Items.Add("3:IOC");  comboBoxnGTEOrderCond.Items.Add("4:FOK");  }  // comboBoxnLTEOrderCond  {  comboBoxnLTEOrderCond.Items.Add("0:ROD");  comboBoxnLTEOrderCond.Items.Add("3:IOC");  comboBoxnLTEOrderCond.Items.Add("4:FOK");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyClear\_Click的Click事件裡) |
| STOCKSTRATEGYORDEROUT pOrder = new STOCKSTRATEGYORDEROUT();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoC.Text;  pOrder.bstrClearCancelTime = textBoxbstrClearCancelTimeC.Text;  pOrder.bstrClearAllOrderPrice = textBoxbstrClearAllOrderPriceC.Text;  pOrder.bstrLTETriggerPrice = textBoxbstrLTETriggerPrice.Text;  pOrder.bstrLTEOrderPrice = textBoxbstrLTEOrderPrice.Text;  pOrder.bstrGTETriggerPrice = textBoxbstrGTETriggerPrice.Text;  pOrder.bstrGTEOrderPrice = textBoxbstrGTEOrderPrice.Text;  pOrder.nQty = int.Parse(textBoxnQtyC.Text);  if (comboBoxnBuySellC.Text == "0:買") pOrder.nBuySell = 0;  else if (comboBoxnBuySellC.Text == "1:賣") pOrder.nBuySell = 1;  if (comboBoxnClearAllFlagC.Text == "0:否") pOrder.nClearAllFlag = 0;  else if (comboBoxnClearAllFlagC.Text == "1:是") pOrder.nClearAllFlag = 1;  if (comboBoxnClearAllPriceTypeC.Text == "1:市價") pOrder.nClearAllPriceType = 1;  else if (comboBoxnClearAllPriceTypeC.Text == "2:限價") pOrder.nClearAllPriceType = 2;  if (comboBoxnFinalClearFlagC.Text == "0:否") pOrder.nFinalClearFlag = 0;  else if (comboBoxnFinalClearFlagC.Text == "1:是") pOrder.nFinalClearFlag = 1;  if (comboBoxnOrderTypeC.Text == "0:現股") pOrder.nOrderType = 0;  else if (comboBoxnOrderTypeC.Text == "3:融資") pOrder.nOrderType = 3;  else if (comboBoxnOrderTypeC.Text == "4:融券") pOrder.nOrderType = 4;  if (comboBoxnLTEFlag.Text == "0:否") pOrder.nLTEFlag = 0;  else if (comboBoxnLTEFlag.Text == "1:是") pOrder.nLTEFlag = 1;  if (comboBoxnLTEMarketPrice.Text == "1:市價") pOrder.nLTEMarketPrice = 1;  else if (comboBoxnLTEMarketPrice.Text == "2:限價") pOrder.nLTEMarketPrice = 2;  if (comboBoxnGTEFlag.Text == "0:否") pOrder.nGTEFlag = 0;  else if (comboBoxnGTEFlag.Text == "1:是") pOrder.nGTEFlag = 1;  if (comboBoxnGTEMarketPrice.Text == "1:市價") pOrder.nGTEMarketPrice = 1;  else if (comboBoxnGTEMarketPrice.Text == "2:限價") pOrder.nGTEMarketPrice = 2;  if (comboBoxnGTEOrderCond.Text == "0:ROD") pOrder.nGTEOrderCond = 0;  else if (comboBoxnGTEOrderCond.Text == "3:IOC") pOrder.nGTEOrderCond = 3;  else if (comboBoxnGTEOrderCond.Text == "4:FOK") pOrder.nGTEOrderCond = 4;  if (comboBoxnLTEOrderCond.Text == "0:ROD") pOrder.nLTEOrderCond = 0;  else if (comboBoxnLTEOrderCond.Text == "3:IOC") pOrder.nLTEOrderCond = 3;  else if (comboBoxnLTEOrderCond.Text == "4:FOK") pOrder.nLTEOrderCond = 4;  if (comboBoxnClearOrderCondC.Text == "0:ROD") pOrder.nClearOrderCond = 0;  else if (comboBoxnClearOrderCondC.Text == "3:IOC") pOrder.nClearOrderCond = 3;  else if (comboBoxnClearOrderCondC.Text == "4:FOK") pOrder.nClearOrderCond = 4;  string bstrMessage;  // 送出證券智慧單出清條件委託。  int nCode = m\_pSKOrder.SendStockStrategyClear(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyClear】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockStrategyMIT

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單MIT條件委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyMIT ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDERMIT**\* 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**[STOCKSTRATEGYORDERMIT](#_MIT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，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。錯誤代碼可參考對照表。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。   * 自V2.13.30(含)版本起，客戶可自設觸價方向(物件**[STOCKSTRATEGYORDERMIT](#_MIT單)**)。 * 相關回報部分：被動回報[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)，主動回報請參考回報文件。 * ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 * 預風控說明：此功能將於主機收到智慧單委託後立刻進行風控計算，可以大幅縮短觸價至送出委託的所耗時間。 * 預風控注意事項：此功能會在觸發前預先扣除條件中委託所使用之額度(包括能夠進行操作的庫存)， 使用時請務必留意是否有進行中的MIT智慧單與委託，在未達成條件前取消智慧單委託可釋放占用額度。 | |

|  |
| --- |
| C#範例程式碼\_ TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出6個TextBox，命名(Name)為「textBoxbstrStockNoMIT」、「textBoxbstrDealPriceMIT」、「textBoxbstrTriggerPriceMIT」、「textBoxnQtyMIT」、「textBoxbstrOrderPriceMIT」、「textBoxbstrLongEndDateMIT」。 拉出8個ComboBox，命名(Name)為「comboBoxnOrderTypeMIT」、「comboBoxnOrderCondMIT」、「comboBoxnBuySellMIT」、「comboBoxnTriggerDirMIT」、「comboBoxnOrderPriceTypeMIT」、「comboBoxnLongActionFlagMIT」、「comboBoxnLATypeMIT」、「comboBoxnPreRiskFlag」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyMIT」。 | |
| 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 |
| // MIT  {  // comboBoxnOrderTypeMIT  {  comboBoxnOrderTypeMIT.Items.Add("0:現股");  comboBoxnOrderTypeMIT.Items.Add("3:融資");  comboBoxnOrderTypeMIT.Items.Add("4:融券");  comboBoxnOrderTypeMIT.Items.Add("8:無券普賣");  }  // comboBoxnBuySellMIT  {  comboBoxnBuySellMIT.Items.Add("0:買");  comboBoxnBuySellMIT.Items.Add("1:賣");  }  // comboBoxnOrderPriceTypeMIT  {  comboBoxnOrderPriceTypeMIT.Items.Add("1:市價");  comboBoxnOrderPriceTypeMIT.Items.Add("2:限價");  }  // comboBoxnOrderCondMIT  {  comboBoxnOrderCondMIT.Items.Add("0:ROD");  comboBoxnOrderCondMIT.Items.Add("3:IOC");  comboBoxnOrderCondMIT.Items.Add("4:FOK");  }  // comboBoxnTriggerDirMIT  {  comboBoxnTriggerDirMIT.Items.Add("1:GTE大於等於");  comboBoxnTriggerDirMIT.Items.Add("2:LTE小於等於");  }  // comboBoxnLongActionFlagMIT  {  comboBoxnLongActionFlagMIT.Items.Add("0:否");  comboBoxnLongActionFlagMIT.Items.Add("1:是");  }  // comboBoxnLATypeMIT  {  comboBoxnLATypeMIT.Items.Add("0:非長效單");  comboBoxnLATypeMIT.Items.Add("1:效期內觸發即失效");  comboBoxnLATypeMIT.Items.Add("3:效期內完全成交即失效");  }  // comboBoxnPreRiskFlag  {  comboBoxnPreRiskFlag.Items.Add("0:關閉預風控");  comboBoxnPreRiskFlag.Items.Add("1:開啟預風控");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyMIT\_Click的Click事件裡) |
| STOCKSTRATEGYORDERMIT pOrder = new STOCKSTRATEGYORDERMIT();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoMIT.Text;  pOrder.bstrOrderPrice = textBoxbstrOrderPriceMIT.Text;  pOrder.bstrTriggerPrice = textBoxbstrTriggerPriceMIT.Text;  pOrder.bstrDealPrice = textBoxbstrDealPriceMIT.Text;  pOrder.bstrLongEndDate = textBoxbstrLongEndDateMIT.Text;  pOrder.nQty = int.Parse(textBoxnQtyMIT.Text);  if (comboBoxnOrderTypeMIT.Text == "0:現股") pOrder.nOrderType = 0;  else if (comboBoxnOrderTypeMIT.Text == "3:融資") pOrder.nOrderType = 3;  else if (comboBoxnOrderTypeMIT.Text == "4:融券") pOrder.nOrderType = 4;  else if (comboBoxnOrderTypeMIT.Text == "8:無券普賣") pOrder.nOrderType = 8;  if (comboBoxnBuySellMIT.Text == "0:買") pOrder.nBuySell = 0;  else if (comboBoxnBuySellMIT.Text == "1:賣") pOrder.nBuySell = 1;  if (comboBoxnOrderPriceTypeMIT.Text == "1:市價") pOrder.nOrderPriceType = 1;  else if (comboBoxnOrderPriceTypeMIT.Text == "2:限價") pOrder.nOrderPriceType = 2;  if (comboBoxnOrderCondMIT.Text == "0:ROD") pOrder.nOrderCond = 0;  else if (comboBoxnOrderCondMIT.Text == "3:IOC") pOrder.nOrderCond = 3;  else if (comboBoxnOrderCondMIT.Text == "4:FOK") pOrder.nOrderCond = 4;  if (comboBoxnTriggerDirMIT.Text == "1:GTE大於等於") pOrder.nTriggerDir = 1;  else if (comboBoxnTriggerDirMIT.Text == "2:LTE小於等於") pOrder.nTriggerDir = 2;  if (comboBoxnPreRiskFlag.Text == "0:關閉預風控") pOrder.nPreRiskFlag = 0;  else if (comboBoxnPreRiskFlag.Text == "1:開啟預風控") pOrder.nPreRiskFlag = 1;  if (comboBoxnLongActionFlagMIT.Text == "0:否") pOrder.nLongActionFlag = 0;  else if (comboBoxnLongActionFlagMIT.Text == "1:是") pOrder.nLongActionFlag = 1;  if (comboBoxnLATypeMIT.Text == "1:效期內觸發即失效") pOrder.nLAType = 1;  else if (comboBoxnLATypeMIT.Text == "3:效期內完全成交即失效") pOrder.nLAType = 3;  string bstrMessage;  // 送出證券智慧單MIT條件委託。  int nCode = m\_pSKOrder.SendStockStrategyMIT(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyMIT】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockStrategyOCO

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單二擇一OCO條件委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyOCO ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDEROCO**\* 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**[STOCKSTRATEGYORDEROCO](#_STOCKSTRATEGYORDEROCO(證券智慧單物件-OCO))**物件，將下單條件填入該物件後，再帶入此欄位中。 |
| bstrMessage | **同步委託：**如果回傳值為 0表示委託成功，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。錯誤代碼可參考對照表。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 | |

|  |
| --- |
| C#範例程式碼\_ TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出6個TextBox，命名(Name)為「textBoxbstrStockNoOCO」、「textBoxbstrTriggerUp」、「textBoxbstrOrderPriceOCO」、「textBoxbstrTriggerDown」、「textBoxbstrOrderPrice2」、「textBoxnQtyOCO」。 拉出8個ComboBox，命名(Name)為「comboBoxnBuySellUp」、「comboBoxnOrderCondUp」、「comboBoxnOrderTypeUp」、「comboBoxnOrderPriceTypeUp」、「comboBoxnBuySellDown」、「comboBoxnOrderCondDown」、「comboBoxnOrderTypeDown」、「comboBoxnOrderPriceTypeDown」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyOCO」。 | |
| 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 |
| // OCO  {  // comboBoxnBuySellUp  {  comboBoxnBuySellUp.Items.Add("0:買");  comboBoxnBuySellUp.Items.Add("1:賣");  }  // comboBoxnBuySellDown  {  comboBoxnBuySellDown.Items.Add("0:買");  comboBoxnBuySellDown.Items.Add("1:賣");  }  // comboBoxnOrderCondUp  {  comboBoxnOrderCondUp.Items.Add("0:ROD");  comboBoxnOrderCondUp.Items.Add("3:IOC");  comboBoxnOrderCondUp.Items.Add("4:FOK");  }  // comboBoxnOrderCondDown  {  comboBoxnOrderCondDown.Items.Add("0:ROD");  comboBoxnOrderCondDown.Items.Add("3:IOC");  comboBoxnOrderCondDown.Items.Add("4:FOK");  }  // comboBoxnOrderTypeUp  {  comboBoxnOrderTypeUp.Items.Add("0:現股");  comboBoxnOrderTypeUp.Items.Add("3:融資");  comboBoxnOrderTypeUp.Items.Add("4:融券");  comboBoxnOrderTypeUp.Items.Add("8:無券普賣");  }  // comboBoxnOrderTypeDown  {  comboBoxnOrderTypeDown.Items.Add("0:現股");  comboBoxnOrderTypeDown.Items.Add("3:融資");  comboBoxnOrderTypeDown.Items.Add("4:融券");  comboBoxnOrderTypeDown.Items.Add("8:無券普賣");  }  // comboBoxnOrderPriceTypeUp  {  comboBoxnOrderPriceTypeUp.Items.Add("1:市價");  comboBoxnOrderPriceTypeUp.Items.Add("2:限價");  }  // comboBoxnOrderPriceTypeDown  {  comboBoxnOrderPriceTypeDown.Items.Add("1:市價");  comboBoxnOrderPriceTypeDown.Items.Add("2:限價");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyOCO\_Click的Click事件裡) |
| STOCKSTRATEGYORDEROCO pOrder = new STOCKSTRATEGYORDEROCO();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoOCO.Text;  pOrder.nQty = int.Parse(textBoxnQtyOCO.Text);  pOrder.bstrOrderPrice = textBoxbstrOrderPriceOCO.Text;  pOrder.bstrTriggerUp = textBoxbstrTriggerUp.Text;  pOrder.bstrOrderPrice2 = textBoxbstrOrderPrice2.Text;  pOrder.bstrTriggerDown = textBoxbstrTriggerDown.Text;  if (comboBoxnBuySellUp.Text == "0:買") pOrder.nBuySellUp = 0;  else if (comboBoxnBuySellUp.Text == "1:賣") pOrder.nBuySellUp = 1;  if (comboBoxnBuySellDown.Text == "0:買") pOrder.nBuySellDown = 0;  else if (comboBoxnBuySellDown.Text == "1:賣") pOrder.nBuySellDown = 1;  if (comboBoxnOrderCondUp.Text == "0:ROD") pOrder.nOrderCondUp = 0;  else if (comboBoxnOrderCondUp.Text == "3:IOC") pOrder.nOrderCondUp = 3;  else if (comboBoxnOrderCondUp.Text == "4:FOK") pOrder.nOrderCondUp = 4;  if (comboBoxnOrderCondDown.Text == "0:ROD") pOrder.nOrderCondDown = 0;  else if (comboBoxnOrderCondDown.Text == "3:IOC") pOrder.nOrderCondDown = 3;  else if (comboBoxnOrderCondDown.Text == "4:FOK") pOrder.nOrderCondDown = 4;  if (comboBoxnOrderTypeUp.Text == "0:現股") pOrder.nOrderTypeUp = 0;  else if (comboBoxnOrderTypeUp.Text == "3:融資") pOrder.nOrderTypeUp = 3;  else if (comboBoxnOrderTypeUp.Text == "4:融券") pOrder.nOrderTypeUp = 4;  else if (comboBoxnOrderTypeUp.Text == "8:無券普賣") pOrder.nOrderTypeUp = 8;  if (comboBoxnOrderTypeDown.Text == "0:現股") pOrder.nOrderTypeDown = 0;  else if (comboBoxnOrderTypeDown.Text == "3:融資") pOrder.nOrderTypeDown = 3;  else if (comboBoxnOrderTypeDown.Text == "4:融券") pOrder.nOrderTypeDown = 4;  else if (comboBoxnOrderTypeDown.Text == "8:無券普賣") pOrder.nOrderTypeDown = 8;  if (comboBoxnOrderPriceTypeUp.Text == "1:市價") pOrder.nOrderPriceTypeUp = 1;  else if (comboBoxnOrderPriceTypeUp.Text == "2:限價") pOrder.nOrderPriceTypeUp = 2;  if (comboBoxnOrderPriceTypeDown.Text == "1:市價") pOrder.nOrderPriceTypeDown = 1;  else if (comboBoxnOrderPriceTypeDown.Text == "2:限價") pOrder.nOrderPriceTypeDown = 2;  string bstrMessage;  // 送出證券智慧單二擇一OCO條件委託。  int nCode = m\_pSKOrder.SendStockStrategyOCO(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyOCO】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockStrategyMIOC

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單多次IOC條件委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyMIOC ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDERMIOC**\* 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**[STOCKSTRATEGYORDERMIOC](#_STOCKSTRATEGYORDERMIOC(證券智慧單物件-多次IO_1)**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，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。錯誤代碼可參考對照表。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 | |

|  |
| --- |
| C#範例程式碼\_TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出5個TextBox，命名(Name)為「textBoxbstrStockNoMIOC」、「textBoxnOneceQtyLimit」、「textBoxbstrOrderPriceUp」、「textBoxbstrOrderPriceDown」、「textBoxnTotalQty」。 拉出4個ComboBox，命名(Name)為「comboBoxnPrimeMIOC」、「comboBoxnOrderTypeMIOC」、「comboBoxnBuySellMIOC1」、「comboBoxnOrderPriceTypeMIOC」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyMIOC」。 | |
| 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 |
| // MIOC  {  // comboBoxnPrimeMIOC  {  comboBoxnPrimeMIOC.Items.Add("0:上市");  comboBoxnPrimeMIOC.Items.Add("1:上櫃");  }  // comboBoxnOrderTypeMIOC  {  comboBoxnOrderTypeMIOC.Items.Add("0:現股");  comboBoxnOrderTypeMIOC.Items.Add("3:融資");  comboBoxnOrderTypeMIOC.Items.Add("4:融券");  comboBoxnOrderTypeMIOC.Items.Add("8:無券普賣");  }  // comboBoxnBuySellMIOC1  {  comboBoxnBuySellMIOC1.Items.Add("1:買");  comboBoxnBuySellMIOC1.Items.Add("2:賣");  }    // comboBoxnOrderPriceTypeMIOC  {  comboBoxnOrderPriceTypeMIOC.Items.Add("0:市價");  comboBoxnOrderPriceTypeMIOC.Items.Add("1:(買單)委賣價或(賣單)委買價");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyMIOC\_Click的Click事件裡) |
| STOCKSTRATEGYORDERMIOC pOrder = new STOCKSTRATEGYORDERMIOC();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoMIOC.Text;  pOrder.bstrOrderPriceUp = textBoxbstrOrderPriceUp.Text;  pOrder.bstrOrderPriceDown = textBoxbstrOrderPriceDown.Text;  pOrder.nOneceQtyLimit = int.Parse(textBoxnOneceQtyLimit.Text);  pOrder.nTotalQty = int.Parse(textBoxnTotalQty.Text);  if (comboBoxnPrimeMIOC.Text == "0:上市") pOrder.nPrime = 0;  else if (comboBoxnPrimeMIOC.Text == "1:上櫃") pOrder.nPrime = 1;  if (comboBoxnBuySellMIOC1.Text == "1:買") pOrder.nBuySell = 1;  else if (comboBoxnBuySellMIOC1.Text == "2:賣") pOrder.nBuySell = 2;  if (comboBoxnOrderTypeMIOC.Text == "0:現股") pOrder.nOrderType = 0;  else if (comboBoxnOrderTypeMIOC.Text == "3:融資") pOrder.nOrderType = 3;  else if (comboBoxnOrderTypeMIOC.Text == "4:融券") pOrder.nOrderType = 4;  else if (comboBoxnOrderTypeMIOC.Text == "8:無券普賣") pOrder.nOrderType = 8;  if (comboBoxnOrderPriceTypeMIOC.Text == "0:市價") pOrder.nOrderPriceType = 1;  else if (comboBoxnOrderPriceTypeMIOC.Text == "1:(買單)委賣價或(賣單)委買價") pOrder.nOrderPriceType = 2;  string bstrMessage;  // 送出證券智慧單多次IOC條件委託。  int nCode = m\_pSKOrder.SendStockStrategyMIOC(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyMIOC】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockStrategyMST

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單移動停損條件委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyMST ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDERMIT**\* 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**[STOCKSTRATEGYORDERMIT](#_MST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，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。錯誤代碼可參考對照表。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。   * 自V2.13.30(含)版本起，客戶可選擇是否自訂啟動價格、自訂啟動觸價方向等條件(詳參考物件**[STOCKSTRATEGYORDERMIT](#_MST單)**)。 * 相關回報部分：被動回報[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)，主動回報請參考回報文件。 * ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 | |

|  |
| --- |
| C#範例程式碼\_TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出7個TextBox，命名(Name)為「textBoxbstrStockNoMST」、「textBoxbstrStartPriceMST」、「textBoxbstrDealPriceMST」、「textBoxbstrMovePoint」、「textBoxnQtyMST」、「textBoxbstrOrderPriceMST」、「textBoxbstrLongEndDateMST」。 拉出9個ComboBox，命名(Name)為「comboBoxnPrimeMST」、「comboBoxnOrderTypeMST」、「comboBoxnOrderCondMST」、「comboBoxnBuySellMST」、「comboBoxnTriggerMethod」、「comboBoxnTriggerDirMST」、「comboBoxnOrderPriceTypeMST」、「comboBoxnLongActionFlagMST」、「comboBoxnLATypeMST」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyMST」。 | |
| 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 |
| // MST  {  // comboBoxnOrderTypeMST  {  comboBoxnOrderTypeMST.Items.Add("0:現股");  comboBoxnOrderTypeMST.Items.Add("3:融資");  comboBoxnOrderTypeMST.Items.Add("4:融券");  comboBoxnOrderTypeMST.Items.Add("8:無券普賣");  }  // comboBoxnBuySellMST  {  comboBoxnBuySellMST.Items.Add("0:買");  comboBoxnBuySellMST.Items.Add("1:賣");  }  // comboBoxnOrderPriceTypeMST  {  comboBoxnOrderPriceTypeMST.Items.Add("1:市價");  comboBoxnOrderPriceTypeMST.Items.Add("2:限價");  }  // comboBoxnOrderCondMST  {  comboBoxnOrderCondMST.Items.Add("0:ROD");  comboBoxnOrderCondMST.Items.Add("3:IOC");  comboBoxnOrderCondMST.Items.Add("4:FOK");  }  // comboBoxnTriggerDirMST  {  comboBoxnTriggerDirMST.Items.Add("1:GTE大於等於");  comboBoxnTriggerDirMST.Items.Add("2:LTE小於等於");  }  // comboBoxnPrimeMST  {  comboBoxnPrimeMST.Items.Add("0:上市");  comboBoxnPrimeMST.Items.Add("1:上櫃");  }  // comboBoxnLongActionFlagMST  {  comboBoxnLongActionFlagMST.Items.Add("0:否");  comboBoxnLongActionFlagMST.Items.Add("1:是");  }  // comboBoxnLATypeMST  {  comboBoxnLATypeMST.Items.Add("0:非長效單");  comboBoxnLATypeMST.Items.Add("1:效期內觸發即失效");  }  // comboBoxnTriggerMethod  {  comboBoxnTriggerMethod.Items.Add("0:否,由市價啟動");  comboBoxnTriggerMethod.Items.Add("1:是,由自訂價格啟動");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyMST\_Click的Click事件裡) |
| STOCKSTRATEGYORDERMIT pOrder = new STOCKSTRATEGYORDERMIT();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoMST.Text;  pOrder.bstrOrderPrice = textBoxbstrOrderPriceMST.Text;  pOrder.bstrMovePoint = textBoxbstrMovePoint.Text;  pOrder.bstrDealPrice = textBoxbstrDealPriceMST.Text;  pOrder.bstrStartPrice = textBoxbstrStartPriceMST.Text;  pOrder.nQty = int.Parse(textBoxnQtyMST.Text);  pOrder.bstrLongEndDate = textBoxbstrLongEndDateMST.Text;  if (comboBoxnOrderTypeMST.Text == "0:現股") pOrder.nOrderType = 0;  else if (comboBoxnOrderTypeMST.Text == "3融資") pOrder.nOrderType = 3;  else if (comboBoxnOrderTypeMST.Text == "4:融券") pOrder.nOrderType = 4;  else if (comboBoxnOrderTypeMST.Text == "8:無券普賣") pOrder.nOrderType = 8;  if (comboBoxnBuySellMST.Text == "0:買") pOrder.nBuySell = 0;  else if (comboBoxnBuySellMST.Text == "1:賣") pOrder.nBuySell = 1;  if (comboBoxnOrderPriceTypeMST.Text == "1:市價") pOrder.nOrderPriceType = 1;  else if (comboBoxnOrderPriceTypeMST.Text == "2:限價") pOrder.nOrderPriceType = 2;  if (comboBoxnOrderCondMST.Text == "0:ROD") pOrder.nOrderCond = 0;  else if (comboBoxnOrderCondMST.Text == "3:IOC") pOrder.nOrderCond = 3;  else if (comboBoxnOrderCondMST.Text == "4:FOK") pOrder.nOrderCond = 4;  if (comboBoxnTriggerDirMST.Text == "1:GTE大於等於") pOrder.nTriggerDir = 1;  else if (comboBoxnTriggerDirMST.Text == "2:LTE小於等於") pOrder.nTriggerDir = 2;  if (comboBoxnTriggerMethod.Text == "0:否,由市價啟動") pOrder.nTriggerMethod = 0;  else if (comboBoxnTriggerMethod.Text == "1:是,由自訂價格啟動") pOrder.nTriggerMethod = 1;  if (comboBoxnPrimeMST.Text == "0:上市") pOrder.nPrime = 0;  else if (comboBoxnPrimeMST.Text == "1:上櫃") pOrder.nPrime = 1;  if (comboBoxnLongActionFlagMST.Text == "0:否") pOrder.nLongActionFlag = 0;  else if (comboBoxnLongActionFlagMST.Text == "1:是") pOrder.nLongActionFlag = 1;  if (comboBoxnLATypeMST.Text == "0:非長效單") pOrder.nLAType = 0;  else if (comboBoxnLATypeMST.Text == "1:效期內觸發即失效") pOrder.nLAType = 1;  string bstrMessage;  // 送出證券智慧單移動停損條件委託。  int nCode = m\_pSKOrder.SendStockStrategyMST(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyMST】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockStrategyAB

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單看A下B單委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyAB ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDERMIT**\* 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**[STOCKSTRATEGYORDERMIT](#_AB單_1)**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，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。錯誤代碼可參考對照表。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。   * 相關回報部分：被動回報[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)，主動回報請參考回報文件。 * ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 | |

|  |
| --- |
| C#範例程式碼\_TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出7個TextBox，命名(Name)為「textBoxbstrStockNo2」、「textBoxbstrExchangeNo」、「textBoxbstrStartPriceAB」、「textBoxbstrTriggerPriceAB」、「textBoxbstrStockNoAB」、「textBoxnQtyAB」、「textBoxbstrOrderPriceAB」。 拉出8個ComboBox，命名(Name)為「comboBoxnMarketNo」、「comboBoxnTriggerDirAB」、「comboBoxnBuySellAB」、「comboBoxnOrderTypeAB」、「comboBoxnOrderCondAB」、「comboBoxnPrimeAB」、「comboBoxnOrderPriceTypeAB」、「comboBoxnReserved」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyAB」。 | |
| 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 |
| // AB  {  // comboBoxnOrderTypeAB  {  comboBoxnOrderTypeAB.Items.Add("0:現股");  comboBoxnOrderTypeAB.Items.Add("3:融資");  comboBoxnOrderTypeAB.Items.Add("4:融券");  comboBoxnOrderTypeAB.Items.Add("8:無券普賣");  }  // comboBoxnBuySellAB  {  comboBoxnBuySellAB.Items.Add("0:買");  comboBoxnBuySellAB.Items.Add("1:賣");  }  // comboBoxnOrderPriceTypeAB  {  comboBoxnOrderPriceTypeAB.Items.Add("1:市價");  comboBoxnOrderPriceTypeAB.Items.Add("2:限價");  }  // comboBoxnOrderCondAB  {  comboBoxnOrderCondAB.Items.Add("0:ROD");  comboBoxnOrderCondAB.Items.Add("3:IOC");  comboBoxnOrderCondAB.Items.Add("4:FOK");  }  // comboBoxnTriggerDirAB  {  comboBoxnTriggerDirAB.Items.Add("1:GTE大於等於");  comboBoxnTriggerDirAB.Items.Add("2:LTE小於等於");  }  // comboBoxnPrimeAB  {  comboBoxnPrimeAB.Items.Add("0:上市");  comboBoxnPrimeAB.Items.Add("1:上櫃");  }  // comboBoxnMarketNo  {  comboBoxnMarketNo.Items.Add("1:國內證");  comboBoxnMarketNo.Items.Add("2:國內期");  comboBoxnMarketNo.Items.Add("3:國外證");  comboBoxnMarketNo.Items.Add("4:國外期");  }  // comboBoxnReserved  {  comboBoxnReserved.Items.Add("0:否");  comboBoxnReserved.Items.Add("1:是");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyAB\_Click的Click事件裡) |
| STOCKSTRATEGYORDERMIT pOrder = new STOCKSTRATEGYORDERMIT();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoAB.Text;  pOrder.bstrOrderPrice = textBoxbstrOrderPriceAB.Text;  pOrder.bstrTriggerPrice = textBoxbstrTriggerPriceAB.Text;  pOrder.bstrExchangeNo = textBoxbstrExchangeNo.Text;  pOrder.bstrStockNo2 = textBoxbstrStockNo2.Text;  pOrder.bstrStartPrice = textBoxbstrStartPriceAB.Text;  pOrder.nQty = int.Parse(textBoxnQtyAB.Text);  if (comboBoxnOrderTypeAB.Text == "0:現股") pOrder.nOrderType = 0;  else if (comboBoxnOrderTypeAB.Text == "3融資") pOrder.nOrderType = 3;  else if (comboBoxnOrderTypeAB.Text == "4:融券") pOrder.nOrderType = 4;  else if (comboBoxnOrderTypeAB.Text == "8:無券普賣") pOrder.nOrderType = 8;  if (comboBoxnBuySellAB.Text == "0:買") pOrder.nBuySell = 0;  else if (comboBoxnBuySellAB.Text == "1:賣") pOrder.nBuySell = 1;  if (comboBoxnOrderPriceTypeAB.Text == "1:市價") pOrder.nOrderPriceType = 1;  else if (comboBoxnOrderPriceTypeAB.Text == "2:限價") pOrder.nOrderPriceType = 2;  if (comboBoxnOrderCondAB.Text == "0:ROD") pOrder.nOrderCond = 0;  else if (comboBoxnOrderCondAB.Text == "3:IOC") pOrder.nOrderCond = 3;  else if (comboBoxnOrderCondAB.Text == "4:FOK") pOrder.nOrderCond = 4;  if (comboBoxnTriggerDirAB.Text == "1:GTE大於等於") pOrder.nTriggerDir = 1;  else if (comboBoxnTriggerDirAB.Text == "2:LTE小於等於") pOrder.nTriggerDir = 2;  if (comboBoxnMarketNo.Text == "1:國內證") pOrder.nMarketNo = 1;  else if (comboBoxnMarketNo.Text == "2:國內期") pOrder.nMarketNo = 2;  else if (comboBoxnMarketNo.Text == "3:國外證") pOrder.nMarketNo = 3;  else if (comboBoxnMarketNo.Text == "4:國外期") pOrder.nMarketNo = 4;  if (comboBoxnReserved.Text == "0:否") pOrder.nReserved = 0;  else if (comboBoxnReserved.Text == "1:是") pOrder.nReserved = 1;  if (comboBoxnPrimeAB.Text == "0:上市") pOrder.nPrime = 1;  else if (comboBoxnPrimeAB.Text == "1:上櫃") pOrder.nPrime = 2;  string bstrMessage;  // 送出證券智慧單看A下B單委託。  int nCode = m\_pSKOrder.SendStockStrategyAB(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyAB】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### SendStockStrategyCB

|  |  |  |
| --- | --- | --- |
| 送出證券智慧單自組單委託。  智慧單用戶請注意備註。 | | |
| **宣告** | Long SendStockStrategyCB ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] struct **STOCKSTRATEGYORDER\*** 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**[STOCKSTRATEGYORDER](#_證券自組單(CB)_1)**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，訊息內容則為委託日期、條件單號 (智慧單序號)等委託資料。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。錯誤代碼可參考對照表。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。   * 相關回報部分：被動回報[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)，主動回報請參考回報文件。 * ［智慧單用戶請注意］：智慧單為獨立運作機制，若盤中已自行出清或回補，請務必於盤中自行取消已設定的智慧單，避免條件觸發後重覆成交。 | |

|  |
| --- |
| C#範例程式碼\_TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出11個TextBox，命名(Name)為「textBoxbstrStockNoCB」、「textBoxbstrMITTriggerPriceCB」、「textBoxbstrTPOrderPriceCB」、「textBoxbstrSLPercentCB」、「textBoxbstrPreQtyCB」、「textBoxbstrTickCB」、「textBoxbstrSLOrderPriceCB」、「textBoxbstrSumQtyCB」、「textBoxbstrClearCancelTimeCB」、「textBoxnQtyCB」、「textBoxbstrOrderPriceCB」。 拉出20個ComboBox，命名(Name)為「comboBoxnOrderTypeCB」、「comboBoxnOrderPriceCondCB」、「comboBoxnBuySellCB」、「comboBoxnInnerOrderIsMITCB」、「comboBoxnMITDirCB」、「comboBoxnTakeProfitFlagCB」、「comboBoxnTPDir」、「comboBoxnUpDownFlag」、「comboBoxnUpDownDir」、「comboBoxnPreQtyFlagCB」、「comboBoxnPreQtyDirCB」、「comboBoxnTickFlagCB」、「comboBoxnTickDir」、「comboBoxnStopLossFlagCB」、「comboBoxnSLDirCB」、「comboBoxnSumQtyFlagCB」、「comboBoxnSumQtyDir」、「comboBoxnClearAllFlagCB」、「comboBoxnFinalClearFlagCB」、「comboBoxnOrderPriceTypeCB」。  拉出 1個Button，命名(Name)為「buttonSendStockStrategyCB」。 | |
| 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 |
| // CB  {  // comboBoxnOrderTypeCB  {  comboBoxnOrderTypeCB.Items.Add("0:現");  comboBoxnOrderTypeCB.Items.Add("3:(自)融資");  comboBoxnOrderTypeCB.Items.Add("4:(自)融券");  comboBoxnOrderTypeCB.Items.Add("8:無券賣出");  }  // comboBoxnOrderPriceCondCB  {  comboBoxnOrderPriceCondCB.Items.Add("0:ROD");  comboBoxnOrderPriceCondCB.Items.Add("3:IOC");  comboBoxnOrderPriceCondCB.Items.Add("4:FOK");  }  // comboBoxnBuySellCB  {  comboBoxnBuySellCB.Items.Add("0:買");  comboBoxnBuySellCB.Items.Add("1:賣");  }  // comboBoxnOrderPriceTypeCB  {  comboBoxnOrderPriceTypeCB.Items.Add("1:市價");  comboBoxnOrderPriceTypeCB.Items.Add("2:限價");  }  // comboBoxnInnerOrderIsMITCB  {  comboBoxnInnerOrderIsMITCB.Items.Add("0:N");  comboBoxnInnerOrderIsMITCB.Items.Add("1:Y");  }  // comboBoxnStopLossFlagCB  {  comboBoxnStopLossFlagCB.Items.Add("0:否");  comboBoxnStopLossFlagCB.Items.Add("1:是");  }  // comboBoxnSLDirCB  {  comboBoxnSLDirCB.Items.Add("0:None");  comboBoxnSLDirCB.Items.Add("1:GTE大於等於");  comboBoxnSLDirCB.Items.Add("2:LTE小於等於");  }  // comboBoxnTickFlagCB  {  comboBoxnTickFlagCB.Items.Add("0:否");  comboBoxnTickFlagCB.Items.Add("1:是");  }  // comboBoxnPreQtyFlagCB  {  comboBoxnPreQtyFlagCB.Items.Add("0:否");  comboBoxnPreQtyFlagCB.Items.Add("1:是");  }  // comboBoxnPreQtyDirCB  {  comboBoxnPreQtyDirCB.Items.Add("0:None");  comboBoxnPreQtyDirCB.Items.Add("1:GTE大於等於");  comboBoxnPreQtyDirCB.Items.Add("2:LTE小於等於");  }  // comboBoxnSumQtyFlagCB  {  comboBoxnSumQtyFlagCB.Items.Add("0:否");  comboBoxnSumQtyFlagCB.Items.Add("1:是");  }  // comboBoxnClearAllFlagCB  {  comboBoxnClearAllFlagCB.Items.Add("0:否");  comboBoxnClearAllFlagCB.Items.Add("1:是");  }  // comboBoxnFinalClearFlagCB  {  comboBoxnFinalClearFlagCB.Items.Add("0:否");  comboBoxnFinalClearFlagCB.Items.Add("1:是");  }  // comboBoxnMITDirCB  {  comboBoxnMITDirCB.Items.Add("0:None");  comboBoxnMITDirCB.Items.Add("1:GTE大於等於");  comboBoxnMITDirCB.Items.Add("2:LTE小於等於");  }  // comboBoxnTakeProfitFlagCB  {  comboBoxnTakeProfitFlagCB.Items.Add("0:否");  comboBoxnTakeProfitFlagCB.Items.Add("1:是");  }  // comboBoxnTPDir  {  comboBoxnTPDir.Items.Add("0:None");  comboBoxnTPDir.Items.Add("1:GTE大於等於");  comboBoxnTPDir.Items.Add("2:LTE小於等於");  }  // comboBoxnTickDir  {  comboBoxnTickDir.Items.Add("0:None");  comboBoxnTickDir.Items.Add("1:GTE大於等於");  comboBoxnTickDir.Items.Add("2:LTE小於等於");  }  // comboBoxnUpDownFlag  {  comboBoxnUpDownFlag.Items.Add("0:否");  comboBoxnUpDownFlag.Items.Add("1:是");  }  // comboBoxnUpDownDir  {  comboBoxnUpDownDir.Items.Add("0:None");  comboBoxnUpDownDir.Items.Add("1:GTE大於等於");  comboBoxnUpDownDir.Items.Add("2:LTE小於等於");  }  // comboBoxnSumQtyDir  {  comboBoxnSumQtyDir.Items.Add("0:None");  comboBoxnSumQtyDir.Items.Add("1:GTE大於等於");  comboBoxnSumQtyDir.Items.Add("2:LTE小於等於");  }  } |

|  |
| --- |
| Current Block (可放在buttonSendStockStrategyCB\_Click的Click事件裡) |
| STOCKSTRATEGYORDER pOrder = new STOCKSTRATEGYORDER();  pOrder.bstrFullAccount = comboBoxAccount.Text;  pOrder.bstrStockNo = textBoxbstrStockNoCB.Text;  pOrder.nQty = int.Parse(textBoxnQtyCB.Text);  pOrder.bstrOrderPrice = textBoxbstrOrderPriceCB.Text;  pOrder.bstrMITTriggerPrice = textBoxbstrMITTriggerPriceCB.Text;  pOrder.bstrTPOrderPrice = textBoxbstrTPOrderPriceCB.Text;  pOrder.bstrSLOrderPrice = textBoxbstrSLOrderPriceCB.Text;  pOrder.bstrTick = textBoxbstrTickCB.Text;  pOrder.bstrSLPercent = textBoxbstrSLPercentCB.Text;  pOrder.bstrPreQty = textBoxbstrPreQtyCB.Text;  pOrder.bstrSumQty = textBoxbstrSumQtyCB.Text;  pOrder.bstrClearCancelTime = textBoxbstrClearCancelTimeCB.Text;  if (comboBoxnOrderTypeCB.Text == "0:現股") pOrder.nOrderType = 0;  else if (comboBoxnOrderTypeCB.Text == "3:融資") pOrder.nOrderType = 3;  else if (comboBoxnOrderTypeCB.Text == "4:融券") pOrder.nOrderType = 4;  else if (comboBoxnOrderTypeCB.Text == "8:無券普賣") pOrder.nOrderType = 8;  if (comboBoxnOrderPriceCondCB.Text == "0:ROD") pOrder.nOrderPriceCond = 0;  else if (comboBoxnOrderPriceCondCB.Text == "3:IOC") pOrder.nOrderPriceCond = 3;  else if (comboBoxnOrderPriceCondCB.Text == "4:FOK") pOrder.nOrderPriceCond = 4;  if (comboBoxnBuySellCB.Text == "0:買") pOrder.nBuySell = 0;  else if (comboBoxnBuySellCB.Text == "1:賣") pOrder.nBuySell = 1;  if (comboBoxnOrderPriceTypeCB.Text == "1:市價") pOrder.nOrderPriceType = 1;  else if (comboBoxnOrderPriceTypeCB.Text == "2:限價") pOrder.nOrderPriceType = 2;  if (comboBoxnInnerOrderIsMITCB.Text == "0:否") pOrder.nInnerOrderIsMIT = 0;  else if (comboBoxnInnerOrderIsMITCB.Text == "1:是") pOrder.nInnerOrderIsMIT = 1;  if (comboBoxnTPDir.Text == "0:None") pOrder.nTPDir = 0;  else if (comboBoxnTPDir.Text == "1:GTE大於等於") pOrder.nTPDir = 1;  else if (comboBoxnTPDir.Text == "2:LTE小於等於") pOrder.nTPDir = 2;  if (comboBoxnStopLossFlagCB.Text == "0:否") pOrder.nStopLossFlag = 0;  else if (comboBoxnStopLossFlagCB.Text == "1:是") pOrder.nStopLossFlag = 1;  if (comboBoxnSLDirCB.Text == "0:None") pOrder.nSLDir = 0;  else if (comboBoxnSLDirCB.Text == "1:GTE大於等於") pOrder.nSLDir = 1;  else if (comboBoxnSLDirCB.Text == "2:LTE小於等於") pOrder.nSLDir = 2;  if (comboBoxnTickFlagCB.Text == "0:否") pOrder.nTickFlag = 0;  else if (comboBoxnTickFlagCB.Text == "1:是") pOrder.nTickFlag = 1;  if (comboBoxnTickDir.Text == "0:None") pOrder.nTickDir = 0;  else if (comboBoxnTickDir.Text == "1:GTE大於等於") pOrder.nTickDir = 1;  else if (comboBoxnTickDir.Text == "2:LTE小於等於") pOrder.nTickDir = 2;  if (comboBoxnUpDownFlag.Text == "0:否") pOrder.nUpDownFlag = 0;  else if (comboBoxnUpDownFlag.Text == "1:是") pOrder.nUpDownFlag = 1;  if (comboBoxnUpDownDir.Text == "0:None") pOrder.nUpDownDir = 0;  else if (comboBoxnUpDownDir.Text == "1:GTE大於等於") pOrder.nUpDownDir = 1;  else if (comboBoxnUpDownDir.Text == "2:LTE小於等於") pOrder.nUpDownDir = 2;  if (comboBoxnPreQtyFlagCB.Text == "0:否") pOrder.nPreQtyFlag = 0;  else if (comboBoxnPreQtyFlagCB.Text == "1:是") pOrder.nPreQtyFlag = 1;  if (comboBoxnPreQtyDirCB.Text == "0:None") pOrder.nPreQtyDir = 0;  else if (comboBoxnPreQtyDirCB.Text == "1:GTE大於等於") pOrder.nPreQtyDir = 1;  else if (comboBoxnPreQtyDirCB.Text == "2:LTE小於等於") pOrder.nPreQtyDir = 2;  if (comboBoxnSumQtyFlagCB.Text == "0:否") pOrder.nSumQtyFlag = 0;  else if (comboBoxnSumQtyFlagCB.Text == "1:是") pOrder.nSumQtyFlag = 1;  if (comboBoxnSumQtyDir.Text == "0:None") pOrder.nSumQtyDir = 0;  else if (comboBoxnSumQtyDir.Text == "1:GTE大於等於") pOrder.nSumQtyDir = 1;  else if (comboBoxnSumQtyDir.Text == "2:LTE小於等於") pOrder.nSumQtyDir = 2;  if (comboBoxnClearAllFlagCB.Text == "0:否") pOrder.nClearAllFlag = 0;  else if (comboBoxnClearAllFlagCB.Text == "1:是") pOrder.nClearAllFlag = 1;  if (comboBoxnFinalClearFlagCB.Text == "0:否") pOrder.nFinalClearFlag = 0;  else if (comboBoxnFinalClearFlagCB.Text == "1:是") pOrder.nFinalClearFlag = 1;  if (comboBoxnMITDirCB.Text == "0:None") pOrder.nMITDir = 0;  else if (comboBoxnMITDirCB.Text == "1:GTE大於等於") pOrder.nMITDir = 1;  else if (comboBoxnMITDirCB.Text == "2:LTE小於等於") pOrder.nMITDir = 2;  if (comboBoxnTakeProfitFlagCB.Text == "0:否") pOrder.nTakeProfitFlag = 0;  else if (comboBoxnTakeProfitFlagCB.Text == "1:是") pOrder.nTakeProfitFlag = 1;  string bstrMessage;  // 送出證券智慧單自組單委託。  int nCode = m\_pSKOrder.SendStockStrategyCB(comboBoxUserID.Text, bAsyncOrder, ref pOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【SendStockStrategyCB】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### 取消證券智慧單委託

###### CancelTSStrategyOrderV1

|  |  |  |
| --- | --- | --- |
| 取消證券智慧單委託新版本。刪單欄位請參考GetTSSmartStrategyReport 回傳的內容。 | | |
| **宣告** | Long CancelTSStrategyOrderV1 ([in] BSTR bstrLogInID, [in] struct CANCELSTRATEGYORDER\* pCancelOrder, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pCancelOrder | 智慧單刪單物件([CANCELSTRATEGYORDER](#_CANCELSTRATEGYORDER(智慧單刪單物件))) |
| bstrMessage | **非同步刪單：**  參照OnAsyncOrder。  如果回傳值為 0表示刪單成功，訊息內容則為刪單之Thread ID。  回傳值非0表示刪單失敗，訊息內容為失敗原因。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  刪單後，請透過智慧單被動回報確認狀態。 | |

###### CancelTSStrategyOrder

|  |  |  |
| --- | --- | --- |
| 取消證券智慧單委託。欄位請參考GetTSSmartStrategyReport 回傳的內容。  注意，當已經觸發的智慧單，將無法取消委託。 | | |
| **宣告** | Long CancelTSStrategyOrder ([in] BSTR bstrLogInID, [in] VARIANT\_BOOL bAsyncOrder, [in] BSTR bstrAccount, [in] BSTR bstrSmartKey, [in] Long nTradeKind, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| bAsyncOrder | 是否為非同步委託。 |
| bstrAccount | 委託帳號 ( BrokerID＋帳號) 。 |
| bstrSmartKey | 智慧單序號或進場單號。 |
| nTradeKind | 智慧單類型  6:MIOC;7:MST; **8:MIT**。 |
| bstrMessage | **同步刪單：**如果回傳值為 0表示刪單成功，訊息內容則為智慧單號刪單訊息。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** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  刪單後，請透過智慧單被動回報確認狀態。  \*當沖單/出清單/OCO二擇一: 請用[CancelTSStrategyOrderV1](#_CancelTSStrategyOrderV1_1) | |

|  |
| --- |
| C#範例程式碼\_ TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出6個TextBox，命名(Name)為「textBoxbstrSmartKey」、「textBoxbstrSeqNo」、「textBoxbstrOrderNo」、「textBoxbstrLongActionKey」、「textBoxbstrParentSmartKey」、「textBoxbstrSmartKeyOut」。 拉出4個ComboBox，命名(Name)為「comboBoxnTradeKindVersion」、「comboBoxnTradeKind」、「comboBoxnTradeKindV1」、「comboBoxnDeletType」。  拉出 2個Button，命名(Name)為「buttonCancelTSStrategyOrderV1」、「buttonCancelTSStrategyOrder」。 | |
| 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 |
| // 刪單  {  // comboBoxnTradeKindVersion  {  comboBoxnTradeKindVersion.Items.Add("舊版");  comboBoxnTradeKindVersion.Items.Add("V1版");  }  // comboBoxnTradeKind  {  comboBoxnTradeKind.Items.Add("6:MIOC");  comboBoxnTradeKind.Items.Add("7:MST");  comboBoxnTradeKind.Items.Add("8:MIT");  }  // comboBoxnTradeKindV1  {  comboBoxnTradeKindV1.Items.Add("3:OCO");  comboBoxnTradeKindV1.Items.Add("8:MIT");  comboBoxnTradeKindV1.Items.Add("9:MST");  comboBoxnTradeKindV1.Items.Add("10:AB");  comboBoxnTradeKindV1.Items.Add("11:當沖");  comboBoxnTradeKindV1.Items.Add("17:出清");  comboBoxnTradeKindV1.Items.Add("27：CB");  }  // comboBoxnDeletType  {  comboBoxnDeletType.Items.Add("1:全部");  comboBoxnDeletType.Items.Add("2:進場單");  comboBoxnDeletType.Items.Add("3:出場單");  }  // comboBoxnMarket  {  comboBoxnMarket.Items.Add("1：國內證");  comboBoxnMarket.Items.Add("2：國內期");  comboBoxnMarket.Items.Add("3：國外證");  comboBoxnMarket.Items.Add("4：國外期");  }  } |

|  |
| --- |
| Current Block (可放在buttonCancelTSStrategyOrderV1\_Click的Click事件裡) |
| CANCELSTRATEGYORDER pCancelOrder = new CANCELSTRATEGYORDER(); // 智慧單刪單物件  pCancelOrder.bstrFullAccount = comboBoxAccount.Text;  //市場別 1：國內證、2：國內期、3：國外證、4：國外期  if (comboBoxnMarket.Text == "1：國內證") pCancelOrder.nMarket = 1;  else if (comboBoxnMarket.Text == "2：國內期") pCancelOrder.nMarket = 2;  else if (comboBoxnMarket.Text == "3：國外證") pCancelOrder.nMarket = 3;  else if (comboBoxnMarket.Text == "4：國外期") pCancelOrder.nMarket = 4;  pCancelOrder.bstrParentSmartKey = textBoxbstrParentSmartKey.Text; // 智慧母單號;  pCancelOrder.bstrSmartKey = textBoxbstrSmartKey.Text; // 智慧單號; 出場單則給出場單號  if (comboBoxnTradeKindV1.Text == "3:OCO") pCancelOrder.nTradeKind = 3;  else if (comboBoxnTradeKindV1.Text == "8:MIT") pCancelOrder.nTradeKind = 8;  else if (comboBoxnTradeKindV1.Text == "9:MST") pCancelOrder.nTradeKind = 9;  else if (comboBoxnTradeKindV1.Text == "10:AB") pCancelOrder.nTradeKind = 10;  else if (comboBoxnTradeKindV1.Text == "11:當沖") pCancelOrder.nTradeKind = 11;  else if (comboBoxnTradeKindV1.Text == "17:出清") pCancelOrder.nTradeKind = 17;  else if (comboBoxnTradeKindV1.Text == "27：CB") pCancelOrder.nTradeKind = 27;  else pCancelOrder.nTradeKind = 0;  if (comboBoxnDeletType.Text == "1:全部") pCancelOrder.nDeleteType = 1;  else if (comboBoxnDeletType.Text == "2:進場單") pCancelOrder.nDeleteType = 2;  else if (comboBoxnDeletType.Text == "3:出場單") pCancelOrder.nDeleteType = 3;  pCancelOrder.bstrSeqNo = textBoxbstrSeqNo.Text;//委託序號(1~3三種類型均需提供, 預約單可忽略)  pCancelOrder.bstrOrderNo = textBoxbstrOrderNo.Text; //((刪進場/出場單提供))委託書號，（若觸發，需給書號）  pCancelOrder.bstrSmartKeyOut = textBoxbstrSmartKeyOut.Text; //出場單號(刪全部時提供，刪進場單及出場單  string bstrMessage;  // 取消證券智慧單委託新版本。刪單欄位請參考GetTSStrategyOrder 回傳的內容。  int nCode = m\_pSKOrder.CancelTSStrategyOrderV1(comboBoxUserID.Text, pCancelOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【CancelTSStrategyOrderV1】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |

|  |
| --- |
| Current Block (可放在buttonCancelTSStrategyOrder\_Click的Click事件裡) |
| int nTradeKind = 0; // 智慧單類型6:MIOC;7:MST; 8:MIT。  string bstrMessage;  if (comboBoxnTradeKind.Text == "6:MIOC") nTradeKind = 6;  else if (comboBoxnTradeKind.Text == "7:MST") nTradeKind = 7;  else if (comboBoxnTradeKind.Text == "8:MIT") nTradeKind = 8;  // 取消證券智慧單委託。欄位請參考GetTSStrategyOrder 回傳的內容。注意，當已經觸發的智慧單，將無法取消委託。  int nCode = m\_pSKOrder.CancelTSStrategyOrder(comboBoxUserID.Text, bAsyncOrder, comboBoxAccount.Text, textBoxbstrSmartKey.Text, nTradeKind, out bstrMessage);  // 取得回傳訊息  string msg = "【CancelTSStrategyOrder】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

##### CancelStrategyList

|  |  |  |
| --- | --- | --- |
| 取消多筆智慧單委託。刪單欄位請參考智慧單被動回報回傳的內容。 | | |
| **宣告** | Long CancelStrategyList ([in] BSTR bstrLogInID, [in] struct CANCELSTRATEGYORDER\* pCancelOrder, [**out**] BSTR\* bstrMessage); | |
| **參數** | bstrLogInID | 登入ID。 |
| pCancelOrder | 智慧單刪單物件(參考[CANCELSTRATEGYORDER](#_智慧單多筆刪單)) |
| bstrMessage | **非同步刪單：**  參照OnAsyncOrder。  如果回傳值為 0表示刪單成功，訊息內容則為刪單之Thread ID。  回傳值非0表示刪單失敗，訊息內容為失敗原因。 |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 使用非同步委託，委託結果請由[OnAsyncOrder](#_OnAsyncOrder_1)取得。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。  刪單後，請透過智慧單被動回報確認狀態。 | |

|  |
| --- |
| C#範例程式碼\_ TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 虛線做法可參考，取消智慧單委託  拉出 1個Button，命名(Name)為「buttonCancelStrategyList」。 | |
| 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 buttonCancelStrategyList\_Click的Click事件裡) |
| CANCELSTRATEGYORDER pCancelOrder = new CANCELSTRATEGYORDER(); // 智慧單刪單物件  pCancelOrder.bstrLogInID = comboBoxUserID.Text;  pCancelOrder.bstrFullAccount = comboBoxAccount.Text;  //市場別 1：國內證、2：國內期、3：國外證、4：國外期  if (comboBoxnMarket.Text == "1：國內證") pCancelOrder.nMarket = 1;  else if (comboBoxnMarket.Text == "2：國內期") pCancelOrder.nMarket = 2;  else if (comboBoxnMarket.Text == "3：國外證") pCancelOrder.nMarket = 3;  else if (comboBoxnMarket.Text == "4：國外期") pCancelOrder.nMarket = 4;  pCancelOrder.bstrSmartKey = textBoxbstrSmartKey.Text; // 智慧單號  string bstrMessage;  // 取消多筆智慧單委託。刪單欄位請參考智慧單被動回報回傳的內容。  int nCode = m\_pSKOrder.CancelStrategyList(comboBoxUserID.Text, pCancelOrder, out bstrMessage);  // 取得回傳訊息  string msg = "【CancelStrategyList】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |

|  |
| --- |
| 執行成功畫面 |
|  |

##### GetTSSmartStrategyReport

|  |  |  |
| --- | --- | --- |
| 查詢證券智慧單。 | | |
| **宣告** | Long GetTSSmartStrategyReport ([in] BSTR bstrLogInID, [in] BSTR bstrAccount, [in] BSTR bstrMarketType, [in] LONG nReportStatus, [in] BSTR bstrKind, [in] BSTR bstrDate); | |
| **參數** | bstrLogInID | 登入ID。 |
| bstrAccount | 委託帳號 (帳號) 。 |
| bstrMarketType | TS :證券市場 |
| nReportStatus | 0:全部的委託單。 |
| bstrKind | 智慧單類型  DayTrade:當沖  ClearOut:出清  **MIT：觸價單、MIT長效單**  **OCO：二擇一**  **MIOC：多次IOC**  **MST：移動停損、MST長效單**  **AB：看A下B單**  **CB：自組單** |
| bstrDate | 查詢日期(ex:20201001) |
| **回傳值** | 0表示成功，其餘非0數值都表示失敗。 | |
| **備註** | 結果由 [OnTSSmartStrategyReport](#_OnRealBalanceReport) 事件回傳。  需簽署證券智慧單風險預告書及未簽署者可至金融網－同意書簽署專區申請。 | |

|  |
| --- |
| C#範例程式碼\_TSStrategyOrderForm.cs |
| UI示意圖 |
| |  | | --- | |  | | 拉出1個TextBox，命名(Name)為「textBoxbstrDate」。 拉出1個ComboBox，命名(Name)為「comboBoxbstrKind」。  拉出 1個Button，命名(Name)為「buttonGetTSSmartStrategyReport」。 | |
| 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 |
| // 查詢  {  // comboBoxbstrKind  {  comboBoxbstrKind.Items.Add("DayTrade:當沖");  comboBoxbstrKind.Items.Add("ClearOut:出清");  comboBoxbstrKind.Items.Add("MIT：觸價單、MIT長效單");  comboBoxbstrKind.Items.Add("OCO：二擇一");  comboBoxbstrKind.Items.Add("MIOC：多次IOC");  comboBoxbstrKind.Items.Add("MST：移動停損、MST長效單");  comboBoxbstrKind.Items.Add("AB：看A下B單");  comboBoxbstrKind.Items.Add("CB：自組單");  }  } |

|  |
| --- |
| Current Block (可放在buttonGetTSSmartStrategyReport\_Click的Click事件裡) |
| string bstrKind = "DayTrade";  string bstrDate = textBoxbstrDate.Text; // 查詢日期(ex:20220601)  if (comboBoxbstrKind.Text == "DayTrade:當沖") bstrKind = "DayTrade";  else if (comboBoxbstrKind.Text == "ClearOut:出清") bstrKind = "ClearOut";  else if (comboBoxbstrKind.Text == "MIT：觸價單、MIT長效單") bstrKind = "MIT";  else if (comboBoxbstrKind.Text == "OCO：二擇一") bstrKind = "OCO";  else if (comboBoxbstrKind.Text == "MIOC：多次IOC") bstrKind = "MIOC";  else if (comboBoxbstrKind.Text == "MST：移動停損、MST長效單") bstrKind = "MST";  else if (comboBoxbstrKind.Text == "AB：看A下B單") bstrKind = "AB";  else if (comboBoxbstrKind.Text == "CB：自組單") bstrKind = "CB";  // 查詢證券智慧單  int nCode = m\_pSKOrder.GetTSSmartStrategyReport(comboBoxUserID.Text, comboBoxAccount.Text, "TS", 0, bstrKind, bstrDate);  // 取得回傳訊息  string msg = "【GetTSSmartStrategyReport】" + m\_pSKCenter.SKCenterLib\_GetReturnCodeMessage(nCode);  richTextBoxMethodMessage.AppendText(msg + "\n"); |
| 執行成功畫面 |
|  |

## 事件

### OnTSSmartStrategyReport

<新版證券智慧單被動回報>依各種策略單別提供格式內容

#### MIOC

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 29:MIOC | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 1：代資  2：代券  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 3：立即成交IOC | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 單次委託量 |  | | 18 | 委託總數量 |  | | 19 | 委託價格上限 |  | | 20 | 委託價格下限 |  | | 21 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 22 | 目前成交數量 |  | | 23 | 訊息更新時間 | 2020-07-07 13:30:41 | | 24 | 訊息 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | |  |  |  | |  |  |  | |
| **備註** | * 回傳【M003,無資料】表示目前查無被動回報 * 資料排序為：先回傳最後一筆委託, 後回傳最早之委託。 * 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。 | |

#### MST

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 9:MST | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 **3：融資**  **4：融券**  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 移動點數 |  | | 19 | 委託數量 |  | | 20 | 觸發價 |  | | 21 | 觸發時間 |  | | 22 | 市價 |  | | 23 | 觸價基準 |  | | 24 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  35：洗價中-觸發價更新(移動停損單)  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 25 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 26 | 訊息更新時間 | 2020-07-07 13:30:41 | | 27 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 28 | 前一個觸發價格 |  | | 29 | 是否自訂啟動價格 | 0:否,由市價啟動價格  1:是,由自訂啟動價格 | | 30 | [自訂]啟動價格 |  | | 31 | [自訂]啟動價格觸發方向 |  | |
| **備註** | * 回傳【M003,無資料】表示目前查無被動回報 * 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。 | |

#### 新版MIT

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 | 無子帳以0000000 | | 8 | 策略別 | 8:MIT | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 **3：融資**  **4：融券**  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 委託數量 |  | | 19 | 觸發價 |  | | 20 | 觸發時間 |  | | 21 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 22 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 23 | 訊息更新時間 | 2020-07-07 13:30:41 | | 24 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 25 | 當下市價價格 |  | | 26 | 觸價方向 | 1:大於等於 ; 2: 小於等於 | | 27 | 成交價觸發價格 |  | | 28 | 萬用訊息 | 適用於智慧單狀態代碼為999情況 | | 29 | 預風控註記 | 0:無開啟 ; 1: 開啟預風控 | | 30 | 拆單註記 |  | | 31 | 是否觸發即停止 | 0：否  1：是 | | 32 | 是否為內外盤委託量 |  | | 33 | 內外盤委託量 | 1：大於等於  2：小於等於 | | 34 | 內外盤委託量觸發方向 | 1:條件觸發後，該筆長效單功能即失效 0:不管有無觸發，每日都送一次條件單，直到長效單結束日後才失效 | | 35 | 長效單註記 | 0：非長效單  1：長效單 | | 36 | 長效單序號 |  | | 37 | 長效單結束日期 | 20230829 | | 38 | 長效單類別 | 0：None  1：效期內觸發即失效  2：[停用]長效單結束日失效  3：效期內完全成交即失效 | | 39 | 長效單委託總量 |  | | 40 | 長效單成交總量 |  | |
| **備註** | * 回傳【M003,無資料】表示目前查無被動回報 * 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。 * 自SKCOM V2.13.30(含)版本起，提供MIT新版被動回報 | |

#### 當沖單- 母單

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 11:DayTrade當沖 | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 **3：融資**  **4：融券**  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 委託數量 |  | | 19 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用)  999:萬用狀態，請注意「萬用訊息」欄說明 | | 20 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 21 | 訊息更新時間 | 2020-07-07 13:30:41 | | 22 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 23 | 委託書號 |  | | 24 | 流水號(13碼) |  | | 25 | 原始流水號(13碼) |  | | 26 | 是否啟用MIT進場 | 0:否; 1:是 | | 27 | MIT進場觸發價 |  | | 28 | MIT進場觸發時間 |  | | 29 | 出清方式 | 17=只選時間出清條件  18=GTE(條件一)  19=LTE(條件二)  20=GTE\_LTE(條件一+條件二) | | 30 | 停利洗價模式 | 1=指定價格  2=指定漲跌幅(%) | | 31 | 停利指定漲跌幅 |  | | 32 | 停利觸發價 |  | | 33 | 停利委託方式 | 1=市價  2=限價 | | 34 | 停利委託價 |  | | 35 | 停利委託時效 | 0=ROD  3=IOC  4=FOK | | 36 | 停損洗價模式 | 1=指定價格  2=指定漲跌幅(%) | | 37 | 停損指定漲跌幅 |  | | 38 | 停損觸發價 |  | | 39 | 停損委託方式 | 1=市價  2=限價 | | 40 | 停損委託價 |  | | 41 | 停損委託時效 | 0=ROD  3=IOC  4=FOK | | 42 | 是否執行時間出清 |  | | 43 | 指定出清時間 |  | | 44 | 時間出清委託方式 | 1=市價  2=限價 | | 45 | 時間出清委託價 |  | | 46 | 時間出清委託時效 | 0=ROD  3=IOC  4=FOK | | 47 | 是否執行盤後定價 |  | | 48 | 盤後定價執行時間 |  | | 49 | 萬用訊息 | 適用於智慧單狀態代碼為999情況 | | 49 | 進場單資訊 | (預約單無此欄), 若設進場MIT才有此欄  細項參閱當沖-進場單格式  **若無此欄將以[in]代表無進場單資訊** | | 50 | 出場單資訊 | (預約單無此欄)  細項參閱當沖-出場單格式  **若無此欄將以[out]代表無出場單資訊** | |

#### 當沖單- 進場單

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 11:DayTrade當沖 | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 **3：融資**  **4：融券**  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 委託數量 |  | | 19 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 20 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 21 | 訊息更新時間 | 2020-07-07 13:30:41 | | 22 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 23 | 委託書號 |  | | 24 | 流水號(13碼) |  | | 25 | 原始流水號(13碼) |  | | 26 | 是否啟用MIT進場 | 0:否; 1:是 | | 27 | MIT進場觸發價 |  | | 28 | MIT進場觸發時間 |  | | 29 | 出清方式 | 17=只選時間出清條件  18=GTE(條件一)  19=LTE(條件二)  20=GTE\_LTE(條件一+條件二) | | 30 | 停利洗價模式 | 1=指定價格  2=指定漲跌幅(%) | | 31 | 停利指定漲跌幅 |  | | 32 | 停利觸發價 |  | | 33 | 停利委託方式 | 1=市價  2=限價 | | 34 | 停利委託價 |  | | 35 | 停利委託時效 | 0=ROD  3=IOC  4=FOK | | 36 | 停損洗價模式 | 1=指定價格  2=指定漲跌幅(%) | | 37 | 停損指定漲跌幅 |  | | 38 | 停損觸發價 |  | | 39 | 停損委託方式 | 1=市價  2=限價 | | 40 | 停損委託價 |  | | 41 | 停損委託時效 | 0=ROD  3=IOC  4=FOK | | 42 | 是否執行時間出清 |  | | 43 | 指定出清時間 |  | | 44 | 時間出清委託方式 | 1=市價  2=限價 | | 45 | 時間出清委託價 |  | | 46 | 時間出清委託時效 | 0=ROD  3=IOC  4=FOK | | 47 | 是否執行盤後定價 |  | | 48 | 盤後定價執行時間 |  | | 49 | 進場單資訊 | (預約單無此欄), 若設進場MIT才有此欄  細項參閱當沖-進場單格式  **若無此欄將以[in]代表無進場單資訊** | | 50 | 出場單資訊 | (預約單無此欄)  細項參閱當沖-出場單格式  **若無此欄將以[out]代表無出場單資訊** |   當沖-進場單格式, 若設進場MIT才有此欄   |  |  |  | | --- | --- | --- | | 49-1 | 流水號(13碼) |  | | 49-2 | 原始流水號(13碼) |  | | 49-3 | 委託書號 |  | | 49-4 | 委託狀態 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單 39：下單失敗  40：使用者刪單 41：全部成交  42：進場單委託中(當沖)  43：進場單成交，執行出場條件(當沖) | | 49-5 | 委託數量 |  | | 49-6 | 觸價方向 | 1:大於等於  2:小於等於 | | 49-7 | 觸發價格 | MIT進場單觸發價 | | 49-8 | 觸發時間 | MIT進場單觸發時間 | | 49-9 | 訊息 | 智慧單回報：  [智慧單號][商品代碼]訊息  市場回報：參考市場委託狀態 | | 49-10 | 更新時間 | 智慧單異動時間 | | 49-11 | 市場回報狀態 | 空白：無委託單 0：預約 2：全部成交 3：全部取消 4：部分成交，剩餘已取消 5：部分成交，剩餘可取消 6：委託失敗 7:委託成功 8:取消失敗  【交易所動態退單為F開頭編碼】 | |

當沖單- 出場單

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 11:DayTrade當沖 | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 **3：融資**  **4：融券**  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 委託數量 |  | | 19 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交  \*可刪單狀態: 32, 34, 37, 38 | | 20 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 21 | 訊息更新時間 | 2020-07-07 13:30:41 | | 22 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 23 | 委託書號 |  | | 24 | 流水號(13碼) |  | | 25 | 原始流水號(13碼) |  | | 26 | 是否啟用MIT進場 | 0:否; 1:是 | | 27 | MIT進場觸發價 |  | | 28 | MIT進場觸發時間 |  | | 29 | 出清方式 | 17=只選時間出清條件  18=GTE(條件一)  19=LTE(條件二)  20=GTE\_LTE(條件一+條件二) | | 30 | 停利洗價模式 | 1=指定價格  2=指定漲跌幅(%) | | 31 | 停利指定漲跌幅 |  | | 32 | 停利觸發價 |  | | 33 | 停利委託方式 | 1=市價  2=限價 | | 34 | 停利委託價 |  | | 35 | 停利委託時效 | 0=ROD  3=IOC  4=FOK | | 36 | 停損洗價模式 | 1=指定價格  2=指定漲跌幅(%) | | 37 | 停損指定漲跌幅 |  | | 38 | 停損觸發價 |  | | 39 | 停損委託方式 | 1=市價  2=限價 | | 40 | 停損委託價 |  | | 41 | 停損委託時效 | 0=ROD  3=IOC  4=FOK | | 42 | 是否執行時間出清 | 0:否;1:是 | | 43 | 指定出清時間 |  | | 44 | 時間出清委託方式 | 1=市價  2=限價 | | 45 | 時間出清委託價 |  | | 46 | 時間出清委託時效 | 0=ROD  3=IOC  4=FOK | | 47 | 是否執行盤後定價 | 0:否;1:是 | | 48 | 盤後定價執行時間 |  | | 49 | 進場單資訊 | (預約單無此欄), 若設進場MIT才有此欄  細項參閱當沖-進場單格式  **若無此欄將以[in]代表無進場單資訊** | | 50 | 出場單資訊 | (預約單無此欄)  細項參閱當沖-出場單格式  **若無此欄將以[out]代表無出場單資訊** | | 51 |  |  |   當沖-進場單格式, 若設進場MIT才有此欄   |  |  |  | | --- | --- | --- | | 49-1 | 流水號(13碼) |  | | 49-2 | 原始流水號(13碼) |  | | 49-3 | 委託書號 |  | | 49-4 | 委託狀態 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單 39：下單失敗  40：使用者刪單 41：全部成交  42：進場單委託中(當沖)  43：進場單成交，執行出場條件(當沖)  \*可刪單狀態: 32, 34, 37, 38 | | 49-5 | 委託數量 |  | | 49-6 | 觸價方向 | 1:大於等於  2:小於等於 | | 49-7 | 觸發價格 | MIT進場單觸發價 | | 49-8 | 觸發時間 | MIT進場單觸發時間 | | 49-9 | 訊息 | 智慧單回報：  [智慧單號][商品代碼]訊息  市場回報：參考市場委託狀態 | | 49-10 | 更新時間 | 智慧單異動時間 | | 49-11 | 市場回報狀態 | 空白：無委託單 0：預約 2：全部成交 3：全部取消 4：部分成交，剩餘已取消 5：部分成交，剩餘可取消 6：委託失敗 7:委託成功 8:取消失敗  【交易所動態退單為F開頭編碼】 |   當沖-出場單格式   |  |  |  | | --- | --- | --- | | 50-1 | 出場單號 |  | | 50-2 | 流水號(13碼) |  | | 50-3 | 原始流水號(13碼) |  | | 50-4 | 委託書號 |  | | 50-5 | 委託狀態 | 32：中台收單成功 ;33：中台收單失敗 34：洗價中 ;36：洗價失敗  37：洗價觸發 ;38：觸發下單  39：下單失敗 ; 40：使用者刪單 41：全部成交 ; 42：進場單委託中(當沖)  43：進場單成交，執行出場條件(當沖)  \*可刪單狀態: 32, 34, 37, 38 | | 50-6 | 進場成交價 |  | | 50-7 | 停利觸發價 |  | | 50-8 | 停損觸發價 |  | | 50-9 | 觸發時間 | MIT進場單觸發時間 | | 50-10 | 觸發條件 | 條件一觸發，價格:xxx 條件二觸發，價格:xxx 時間出清  盤後定價 | | 50-11 | 訊息 | 智慧單回報：  [**智慧單出場單號**][商品代碼]訊息  市場回報：參考市場委託狀態 | | 50-12 | 市場回報狀態 | 空白：無委託單 0：預約 2：全部成交 3：全部取消 4：部分成交，剩餘已取消 5：部分成交，剩餘可取消 6：委託失敗 7:委託成功 8:取消失敗  【交易所動態退單為F開頭編碼】 | | 50-13 | 委託數量 | 32：中台收單成功 33：中台收單失敗 34：洗價中  35：洗價中-觸發價更新(移動停損單) 36：洗價失敗  37：洗價觸發  38：觸發下單 39：下單失敗  40：使用者刪單 41：全部成交  42：進場單委託中(當沖)  43：進場單成交，執行出場條件(當沖) | | 50-14 | 已成交量 |  | |

#### 出清單

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫 GetTSSmartStrategyReport 後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 9:MST | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 **3：融資**  **4：融券**  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 委託數量 |  | | 19 | 觸發價 |  | | 20 | 觸發時間 |  | | 21 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 22 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 23 | 訊息更新時間 | 2020-07-07 13:30:41 | | 24 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 25 | 委託書號 |  | | 26 | 流水號(13碼) |  | | 27 | 原始流水號(13碼) |  | | 28 | 出清條件類別 | 17=只勾選時間出清條件  18=GTE(條件一)大於  19=LTE(條件二)小於  20=GTE\_LTE(條件一+條件二) | | 29 | 條件一觸發價 |  | | 30 | 條件一委託方式 | 1=市價  2=限價 | | 31 | 條件一委託價 |  | | 32 | 條件一委託時效 | 0=ROD  3=IOC  4=FOK | | 33 | 條件二觸發價 |  | | 34 | 條件二委託方式 | 1=市價  2=限價 | | 35 | 條件二委託價 |  | | 36 | 條件二委託時效 | 0=ROD  3=IOC  4=FOK | | 37 | 是否執行時間出清 |  | | 38 | 指定出清時間 |  | | 39 | 時間出清委託方式 | 1=市價  2=限價 | | 40 | 時間出清委託價 |  | | 41 | 時間出清委託時效 | 0=ROD  3=IOC  4=FOK | | 42 | 是否執行盤後定價 |  | | 43 | 盤後定價執行時間 |  | | 44 | 條件一是否有觸發 | 0=否  1=是 | | 45 | 條件一觸發價格 |  | | 46 | 條件二是否有觸發 | 0=否  1=是 | | 47 | 條件二觸發價格 |  | | 48 | 時間出清是否有觸發 | 0=否  1=是 | | 49 | 盤後定價是否有觸發 | 0=否  1=是 | | 50 | 總委託量 |  | | 51 | 已成交量 |  | | 52 | 進場成交價 |  | | 53 | 子單 | (預約單無此欄)  細項參閱出清-子單格式  **若無此欄將以[out]代表無子單資訊** |   出清子單—**觸**發歷程：   |  |  |  | | --- | --- | --- | | 53-1 | 流水號(13碼) |  | | 53-2 | 委託書號 |  | | 53-3 | 觸發時間 |  | | 53-4 | 觸發條件 | 條件一觸發，價格:xxx 條件二觸發，價格:xxx 時間出清  盤後定價 | | 53-5 | 委託訊息 | [智慧單號][商品代碼]訊息 | | 53-6 | 市場委託狀態 | 空白：無委託單 0：預約 2：全部成交 3：全部取消 4：部分成交，剩餘已取消 5：部分成交，剩餘可取消 6：委託失敗 7:委託成功 8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消 F2：動態退單-部分成交，剩餘已取消 F3：動態退單-部分委託成功 | | 53-7 | 成交張數 |  | |  |  |  | |
| **備註** | * 回傳【M003,無資料】表示目前查無被動回報 * 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。 | |

|  |  |  |
| --- | --- | --- |
| [OCO 二擇一單]證券智慧單被動查詢結果。透過呼叫 GetTSSmartStrategyReport後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 9:MST | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 **3：融資**  **4：融券**  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 委託數量 |  | | 19 | 觸發價 |  | | 20 | 觸發時間 |  | | 21 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 22 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 23 | 訊息更新時間 | 2020-07-07 13:30:41 | | 24 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 25 | 委託書號 |  | | 26 | 流水號(13碼) |  | | 27 | 原始流水號(13碼) |  | | 28 | 第一隻腳觸發價 |  | | 29 | 第二隻腳觸發價 |  | | 30 | 第二隻腳買賣別 | B：買 S：賣 | | 31 | 第二隻腳委託價格別 | 0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 32 | 第二隻腳委託種類別 | 0：現股 3：自資  4：自券  8：無券普賣 | | 33 | 第二隻腳委託價格類別 | 1：市價  2：限價 | | 34 | 第二隻腳委託價 |  | | 35 | 第二隻腳委託時效 | 0：當日有效ROD  3：立即成交IOC  4：立即全部成交FOK | |
| **備註** | * 回傳【M003,無資料】表示目前查無被動回報 * 當全部資料已經全部回傳完畢，將回傳一筆以「##」開頭的內容，表示查詢結束。 | |

#### AB單

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 | 無子帳以0000000 | | 8 | 委託書號 |  | | 9 | 流水號13碼 |  | | 10 | 原始流水號 |  | | 11 | 策略別 | 10:AB | | 12 | 商品代碼 |  | | 13 | 下單市場別 |  | | 14 | 下單交易所別 |  | | 15 | 下單商品代碼 |  | | 16 | 商品契約月份 |  | | 17 | 履約價格 |  | | 18 | 下單商品代碼2 |  | | 19 | 商品契約月份2 |  | | 20 | 履約價格2 |  | | 21 | 買賣別 | B：買 S：賣 | | 22 | 委託種類別 | 0：現股 3：融資  4：融券  8：無券普賣 | | 23 | 委託別 | 0：一般 | | 24 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 25 | 委託價格別 | 1：市價  2：限價 | | 26 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 27 | 委託價格 |  | | 28 | 委託數量 |  | | 29 | 觸發價 |  | | 30 | 觸發時間 |  | | 31 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 32 | 下單時間 | 2020-07-07 12:32:10.000 | | 33 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價  訊息中逗號皆為全形”，” | | 34 | 訊息更新時間 | 2020-07-07 13:30:41 | | 35 | 市場委託狀熊 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 36 | 當下市價價格 |  | | 37 | 觸價方向 | 1：大於等於  2：小於等於 | | 38 | 是否為選擇權商品 | 0：否  1：是 | | 39 | 預約單註記 | 0：否  1：是 | | 40 | 預約單序號 |  | | 41 | 預約單交易日 |  | | 42 | 是否觸發即停止 | 0：否  1：是 | | 43 | 是否為內外盤委託量 | 0：否  1：是 | | 44 | 內外盤委託量 |  | | 45 | 內外盤委託量觸發方向 | 1：大於等於  2：小於等於 | | 46 | 成交價觸發價格 |  | | 47 | 成交量 |  | | 48 | 新平倉 | 0：新單  1：平倉  2：自動 | | 49 | 是否為當沖 | 0：否  1：是 | | 50 | 買賣權 | N：None  C：Call  P：Put | | 51 | 委託價分子 |  | | 52 | 委託價分母 |  | | 53 | 成交價分子 |  | | 54 | 成交價分母 |  | | 55 | 觸發價分子 |  | | 56 | 觸發價分母 |  | | 57 | 萬用訊息 |  | |
| **備註** | * 回傳【M003,無資料】表示目前查無被動回報 * 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。 * 自SKCOM V2.13.30(含)版本起，提供MIT新版被動回報 | |

#### CB單(自組單)

|  |  |  |
| --- | --- | --- |
| 證券智慧單被動查詢結果。透過呼叫[GetTSSmartStrategyReport](#_GetTSSmartStrategyReport)後，資訊由該事件回傳。 | | |
| **宣告** | void OnTSSmartStrategyReport ([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 | | 1 | 智慧單號 |  | | 2 | 市場別 | TS：國內證券 | | 3 | 交易日 | 20200707 | | 4 | 交易所代碼 | TSE上市  OTC上櫃 | | 5 | 分公司代碼 |  | | 6 | 帳號 |  | | 7 | 子帳 |  | | 8 | 策略別 | 27:CB | | 9 | 商品代碼 |  | | 10 | 買賣別 | B：買 S：賣 | | 11 | 委託種類別 | 0：現股 3：融資  4：融券  8：無券普賣 | | 12 | 委託別 | 0：一般 | | 13 | 委託價格類別 | 第一委託價格別  0:前日收盤價 (平盤價)  1:漲停價  2:跌停價  7:使用者輸入價 | | 14 | 委託價格別 | 1：市價  2：限價 | | 15 | 委託時效 | 0:ROD;3：IOC;4:FOK | | 16 | 下單時間 | 2020-07-07 12:32:10.000 | | 17 | 委託價格 |  | | 18 | 委託數量 |  | | 19 | 觸發價 |  | | 20 | 觸發時間 |  | | 21 | 智慧單委託狀態-  代碼 | 32：中台收單成功  33：中台收單失敗  34：洗價中  35：洗價中-觸發價更新(移動停損單)  36：洗價失敗  37：洗價觸發  38：觸發下單  39：下單失敗  40：使用者刪單  41：全部成交(多次IOC用) | | 22 | 市場委託狀態 | 空白：無委託單  0：預約  2：全部成交  3：全部取消  4：部分成交，剩餘已取消  5：部分成交，剩餘可取消  6：委託失敗  7:委託成功  8:取消失敗  【交易所動態退單為F開頭編碼】  F1：動態退單-全部取消  F2：動態退單-部分成交，剩餘已取消  F3：動態退單-部分委託成功 | | 23 | 訊息更新時間 | 2020-07-07 13:30:41 | | 24 | 訊息含狀態 | [智慧單號]  [商品代碼]  訊息:參考市場委託狀態  Ex: [537500][2801]已加入洗價 | | 25 | 是否全部條件成立 | 0：任一條件成立  1：全部條件成立 | | 26 | 是否自訂啟動時間 | 0：否  1：是 | | 27 | 自訂啟動時間 | 08:00:00.000 | | 28 | 是否執行成交價 | 0：否  1：是 | | 29 | 成交價 |  | | 30 | 成交價觸價方向 | 0：None  1：GTE  2：LTE | | 31 | 成交價觸發紀錄 | 0：未觸發  1：觸發 | | 32 | 成交價觸發價格 |  | | 33 | 是否執行委買價 | 0：否  1：是 | | 34 | 委買價 |  | | 35 | 委買價觸價方向 | 0：None  1：GTE  2：LTE | | 36 | 委買價觸發紀錄 | 0：未觸發  1：觸發 | | 37 | 委買價觸發價格 |  | | 38 | 是否執行委賣價 | 0：否  1：是 | | 39 | 委賣價 |  | | 40 | 委賣價觸價方向 | 0：None  1：GTE  2：LTE | | 41 | 委賣價觸發記號 | 0：未觸發  1：觸發 | | 42 | 委賣價觸發價格 |  | | 43 | 是否執行漲跌tick | 0：否  1：是 | | 44 | Tick數 |  | | 45 | 漲跌tick觸價方向 | 0：None  1：GTE  2：LTE | | 46 | 漲跌tick觸發記號 | 0：未觸發  1：觸發 | | 47 | 漲跌tick觸發tick |  | | 48 | 是否執行漲跌幅(%) | 0：否  1：是 | | 49 | 漲跌幅(%) |  | | 50 | 漲跌幅的觸價方向 | 0：None  1：GTE  2：LTE | | 51 | 漲跌幅觸發記號 | 0：未觸發  1：觸發 | | 52 | 漲跌幅觸發幅度 |  | | 53 | 是否執行總量 | 0：否  1：是 | | 54 | 總量 |  | | 55 | 總量觸價方向 | 0：None  1：GTE  2：LTE | | 56 | 總量觸發記號 | 0：未觸發  1：觸發 | | 57 | 總量觸發數量 |  | | 58 | 是否執行單量 | 0：否  1：是 | | 59 | 單量 |  | | 60 | 單量觸價方向 | 0：None  1：GTE  2：LTE | | 61 | 單量觸發記號 | 0：未觸發  1：觸發 | | 62 | 單量觸發數量 |  | | 63 | 萬用訊息 | 適用於智慧單狀態代碼為999情況 | |
| 備註 | * 回傳【M003,無資料】表示目前查無被動回報 * 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\_TSStrategyOrderForm.cs |
| UI示意圖 |
| 拉出1個 RichTextBox，命名(Name)為「richTextBoxOnTSSmartStrategyReport」。 |
| 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  //<新版證券智慧單被動回報>依各種策略單別提供格式內容  m\_pSKOrder.OnTSSmartStrategyReport += new \_ISKOrderLibEvents\_OnTSSmartStrategyReportEventHandler(OnTSSmartStrategyReport);  void OnTSSmartStrategyReport(string bstrData)  {  richTextBoxOnTSSmartStrategyReport.AppendText(bstrData + '\n');  } |

### 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\_TSStrategyOrderForm.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");  } |

## Struct 結構物件

### STOCKSTRATEGYORDER(證券智慧單物件-當沖/CB)

#### 證券當沖

struct STOCKSTRATEGYORDER

{

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

//商品//

|  |
| --- |
| BSTR bstrStockNo; //委託股票代號  LONG nBuySell; //當沖進場 0:現股買;1:無券賣出  LONG nQty; //委託張數  LONG nOrderPriceCond; //委託時效(0:ROD,3:IOC,4:FOK)  BSTR bstrOrderPrice; //當沖單進場委託價  LONG nOrderPriceType; //委託價類別(1:市價; 2:限價) |

//當沖進場條件//

|  |
| --- |
| LONG nInnerOrderIsMIT; //進場是否MIT( 0: N; 1:Y)  LONG nMITDir; //進場MIT觸價方向(0:未啟用MIT,1: 向上觸發(大於等於) , 2:向下觸發(小於等於))  BSTR bstrMITTriggerPrice; //進場MIT觸發價(若未啟用MIT,請填0)  BSTR bstrMITDealPrice; //進場MIT當下市價(若未啟用MIT,請填0) |

//停損出場條件//

|  |
| --- |
| LONG nStopLossFlag; //出場停損條件0:否；1:是  LONG nRDOSLPercent; //停損類型0:觸發價；1:漲跌幅  BSTR bstrSLTrigger; //停損觸發價  BSTR bstrSLPercent; //停損百分比  BSTR bstrSLOrderPrice; //停損委託價  LONG nRDSLMarketPriceType; ///停損委託價方式1:市價; 2:限價  LONG nStopLossOrderCond; //停損出場時效 0:ROD;3:IOC;4:FOK |

//停利出場條件//

|  |
| --- |
| LONG nTakeProfitFlag; //出場停利條件 0:否；1:是  LONG nRDOTPPercent; //停利類型 0:觸發價；1:漲幅  BSTR bstrTPTrigger; //停利觸發價  BSTR bstrTPPercent; //停利百分比  BSTR bstrTPOrderPrice; //停利委託價  LONG nRDTPMarketPriceType; //停利委託價方式1:市價; 2:限價  LONG nTakeProfitOrderCond; //停利出場時效 0:ROD;3:IOC;4:FOK |

//時間出清條件//

|  |
| --- |
| LONG nClearAllFlag; //執行時間出清0:否；1:是  BSTR bstrClearCancelTime; //出清時間-時＋分（hhmm）{每日13:20截止}  BSTR bstrClearAllOrderPrice; //出清委託價  LONG 　　nClearAllPriceType; //出清委託價方式 1:市價; 2:限價  LONG nClearOrderCond; //出清出場時效 0:ROD;3:IOC;4:FOK |

//盤後定價交易出場條件//

|  |
| --- |
| LONG nFinalClearFlag; //盤後定盤交易 0:否；1:是 |

}

#### 證券自組單(CB)

struct STOCKSTRATEGYORDER

{

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

//商品//

|  |
| --- |
| BSTR bstrStockNo; //委託股票代號  LONG nOrderType;  //委託交易別(0:現股, 3:融資, 4:融券, 8:無券普賣)  LONG nOrderPriceCond; //委託時效(0:ROD,3:IOC,4:FOK)  LONG nBuySell; //0買;1賣 |

//組合//

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | LONG nInnerOrderIsMIT; //觸發條件是否為成交價(0:否, 1:是)  LONG nMITDir; //成交價觸價方向(0:None, 1:GTE大於等於, 2:LTE小於等於)  BSTR bstrMITTriggerPrice;//成交價 | | LONG nTakeProfitFlag;             //觸發條件是否為委買價(0:否, 1:是)  LONG  nTPDir; //委買價觸價方向 (0:None, 1:GTE大於等於, 2:LTE小於等於)  BSTR bstrTPOrderPrice;          //委買價 | | LONG nUpDownFlag; //是否為漲跌幅 (0:否, 1:是)  LONG nUpDownDir; //漲跌幅觸價方向(0:None, 1:GTE大於等於, 2:LTE小於等於)  BSTR bstrSLPercent; //漲跌幅數 | | LONG nPreQtyFlag; //是否執行單量 (0:否, 1:是)  LONG nPreQtyDir; //單量觸價方向(0:None, 1:GTE大於等於, 2:LTE小於等於)  BSTR bstrPreQty; //單量 | | LONG nStopLossFlag;  //是否為委賣價 (0:否, 1:是)  LONG nSLDir; //委賣價觸價方向(0:None, 1:GTE大於等於, 2:LTE小於等於)  BSTR bstrSLOrderPrice;           //委賣價 | | LONG nTickFlag; //是否為tick觸發 (0:否, 1:是)  LONG nTickDir; //tick觸發方向 (0:None, 1:GTE大於等於, 2:LTE小於等於)  BSTR bstrTick; //tick數 | | LONG nSumQtyFlag; //觸發條件是否為總量 (0:否, 1:是)  LONG nSumQtyDir; //總量觸價方向(0:None, 1:GTE大於等於, 2:LTE小於等於)  BSTR bstrSumQty; //總量 | |

//觸發時間//

|  |
| --- |
| LONG nClearAllFlag; //是否為自訂啟動時間 (0:立即, 1:自訂)  BSTR bstrClearCancelTime; //自訂啟動時間(14天內) 格式：hhmmss (時時分分秒秒) |

//組合條件關係//

|  |
| --- |
| LONG nFinalClearFlag; //條件是否為全部成立 (0:任一成立, 1:全部成立) |

//委託張數//

|  |
| --- |
| LONG nQty; //委託張數 |

//委託張數//

|  |
| --- |
| LONG nOrderPriceType; //委託價類別(1:市價; 2:限價)  BSTR bstrOrderPrice; //委託價 |

)

### STOCKSTRATEGYORDEROUT(證券智慧單物件-出清)

struct STOCKSTRATEGYORDEROUT

{

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

//商品//

|  |
| --- |
| BSTR bstrStockNo; //委託股票代號  LONG nBuySell; //0買;1賣  LONG nOrderType; //出清進場委託交易別(0:現股, 3:融資, 4:融券)  LONG nQty; //委託張數 |

//大於出場條件//

|  |
| --- |
| LONG nGTEFlag; //出場成交價大於指定價後觸發0:否；1:是  BSTR bstrGTETriggerPrice; //大於指定觸發價  BSTR bstrGTEOrderPrice; //大於出場指定委託價  LONG nGTEMarketPrice; //(大於條件)以1:市價; 2:限價 做為出場價格  LONG nGTEOrderCond; //大於條件出場時效0:ROD;3:IOC;4:FOK |

//小於出場條件//

|  |
| --- |
| **LONG** nLTEFlag; //出場成交價小於指定價後觸發0:否；1:是  BSTR bstrLTETriggerPrice; //小於指定觸發價  BSTR bstrLTEOrderPrice; //小於出場指定委託價  LONG nLTEMarketPrice; //(小於條件)以1:市價; 2:限價 做為出場價格  LONG nLTEOrderCond; //小於條件出場時效0:ROD;3:IOC;4:FOK |

//時間出清條件//

|  |
| --- |
| LONG nClearAllFlag; //執行時間出清0:否；1:是  BSTR bstrClearCancelTime; //出清時間-時＋分（hhmm）{每日13:20截止}  BSTR bstrClearAllOrderPrice; //出清委託價  LONG nClearAllPriceType; //出清委託價方式=1:市價; 2:限價  LONG nClearOrderCond; //出清出場時效0:ROD;3:IOC;4:FOK |

//盤後定價交易條件//

|  |
| --- |
| LONG nFinalClearFlag; //盤後定盤交易 0:否；1:是 |

}

### STOCKSTRATEGYORDERMIT(證券智慧單物件-MIT/MST/AB)

#### MIT單

**(新版本MIT， V2.13.30 含以上版本適用)**，並自V2.13.30 (含)以上版本起，不再支援舊版下單)

struct STOCKSTRATEGYORDERMIT

{

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

BSTR bstrDealPrice; //成交價 (當下市價)，洗價機留存用

//商品//

|  |
| --- |
| BSTR bstrStockNo; //委託股票代號  LONG nOrderType; //委託交易別(0:現股, 3:融資, 4:融券, 8:無券普賣)  LONG nOrderCond; //委託時效(0:ROD,3:IOC,4:FOK)  LONG nBuySell; //0買;1賣 |

//組合//

|  |
| --- |
| LONG nTriggerDir; //觸價方向(1:GTE大於等於,2:LTE小於等於)  BSTR bstrTriggerPrice; //觸發價  LONG nQty; //委託張數  BSTR bstrOrderPrice; //委託價 (若為市價單, 委託價請填0)  LONG nOrderPriceType; //委託價類別(1:市價; 2:限價)  LONG nLongActionFlag; //是否為長效單(0:否, 1:是)  BSTR bstrLongEndDate //長效單結束日期(YYYYMMDD共8碼, EX: 20220630)  LONG nLAType //觸發結束條件(1:效期內觸發即失效, 3:效期內完全成交即失效) |

//預風控//

|  |
| --- |
| LONG nPreRiskFlag; //預風控功能 0:關閉預風控 ;1:開啟預風控(不支援信用交易) |

};

比較表

|  |
| --- |
| 異動後V2.13.45 |
| **MIT下單新增長效單功能：**是否為長效單(0:否, 1:是)、長效單結束日期(YYYYMMDD共8碼)、觸發結束條件  **MIT下單新增預風控欄位** |

#### MST單

struct STOCKSTRATEGYORDERMIT

{

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

//商品//

|  |
| --- |
| BSTR bstrStockNo; //委託股票代號  LONG nPrime; //0:上市,1:上櫃  LONG nOrderType;  //委託交易別(0:現股, 3:融資, 4:融券, 8:無券普賣)  LONG nOrderCond; //委託時效(0:ROD,3:IOC,4:FOK)  LONG nBuySell; // 0:買; 1:賣 |

//組合//

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | LONG nTriggerMethod; //啟動條件(0：以市價立即啟動, 1：以自訂價格啟動)  BSTR bstrDealPrice; //[市價啟動適用]請填市價  BSTR bstrStartPrice; //[自訂價格啟動適用]觸發價  LONG nTriggerDir //觸價方向(1:GTE大於等於,2:LTE小於等於) | | //移動點數  BSTR bstrMovePoint; //移動點數 | | //委託張數  LONG nQty; //委託張數 | | //委託價格  BSTR bstrOrderPrice; //委託價  LONG nOrderPriceType; //委託價類別(1:市價; 2:限價) | | //長效單  LONG nLongActionFlag; //是否為長效單(0:否, 1:是)  BSTR bstrLongEndDate //長效單結束日期(YYYYMMDD共8碼, EX: 20220630) | | //觸發結束條件  LONG nLAType //觸發結束條件(0:非長效單, 1:效期內觸發即失效) | |

};

#### AB單

struct STOCKSTRATEGYORDERMIT

{

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

//看A//

|  |
| --- |
| BSTR bstrStockNo2; //A商品 委託股票代號  LONG nMarketNo; //A商品 市場編號(1:國內證, 2:國內期, 3:國外證, 4:國外期)  BSTR bstrExchangeNo; // A商品 交易所代碼(EX: TSE、TAIFEX、CME)  BSTR bstrStartPrice; // A商品 成交價  LONG nTriggerDir; // A商品 觸價方向(1:GTE大於等於,  2:LTE小於等於)  BSTR bstrTriggerPrice; // A商品 觸發價 |

//下B//

|  |  |  |
| --- | --- | --- |
| |  | | --- | | //委託條件  LONG nBuySell; // B商品 買賣別(0:買, 1:賣)  LONG nOrderType;  // B商品 委託交易別(0:現股, 3:融資, 4:融券, 8:無券普賣)  LONG nOrderCond; // B商品 委託時效(0:ROD,3:IOC,4:FOK) | | //委託商品  BSTR bstrStockNo; // B商品 委託股票代號  LONG nPrime; // B商品 0:上市,1:上櫃  LONG nQty; // B商品 委託張數  LONG nOrderPriceType; // B商品委託價類別(1:市價; 2:限價)  BSTR bstrOrderPrice; // B商品 委託價 | |

LONG nReserved; //是否為預約單(0:否, 1:是)，A商品為國內期選商品時可選預約單，其餘市場為非預約單

}

### STOCKSTRATEGYORDEROCO(證券智慧單物件-OCO)

struct STOCKSTRATEGYORDEROCO

{

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

//商品//

|  |
| --- |
| BSTR bstrStockNo; //委託股票代號 |

//組合//

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | //當市價>=  BSTR bstrTriggerUp; //第一腳 觸發價  LONG nBuySellUp; //第一腳 0:買;1:賣  LONG nOrderCondUp; //第一腳 委託時效(0:ROD,3:IOC,4:FOK)  LONG nOrderTypeUp; //第一腳 委託交易別(0:現股, 3:融資, 4:融券;8:無券普賣)  BSTR bstrOrderPrice; //第一腳 委託價  LONG nOrderPriceTypeUp; //第一腳 委託價類別(1:市價; 2:限價) | | //當市價<=  BSTR bstrTriggerDown; //第二腳 觸發價  LONG nBuySellDown; //第二腳 0:買;1:賣  LONG nOrderCondDown; //第二腳 委託時效(0:ROD,3:IOC,4:FOK)  LONG nOrderTypeDown; //第二腳 委託交易別(0:現股, 3:融資, 4:融券;8:無券普賣)  BSTR bstrOrderPrice2; //第二腳 委託價  LONG nOrderPriceTypeDown; //第二腳 委託價類別(1:市價; 2:限價) | | //委託張數  LONG nQty; //委託張數 | |

};

\*v2.13.48新增無券普賣委託別

### STOCKSTRATEGYORDERMIOC(證券智慧單物件-多次IOC)

struct STOCKSTRATEGYORDERMIOC

{

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

//商品//

|  |
| --- |
| BSTR bstrStockNo; //委託商品代號  LONG nPrime; //0:上市,1:上櫃  LONG nOrderType; //委託交易別(0:現;3: (自)融資; 4: (自)融券;8無券賣)  LONG nBuySell; // 0:買進 1:賣出 |

//組合//

|  |
| --- |
| LONG nOrderPriceType; //0:市價; 1: (買單)委賣價或(賣單)委買價;  //委買、賣價實際價格由中台決定  LONG nOneceQtyLimit; //單次交易張數上限  BSTR bstrOrderPriceUp; //委託價上限  BSTR bstrOrderPriceDown;//委託價下限  LONG nTotalQty; //總委張數 |

};

### CANCELSTRATEGYORDER(智慧單刪單物件)

#### 證券當沖

struct CANCELSTRATEGYORDER

{

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

LONG nMarket; //市場別1:國內證券

BSTR bstrParentSmartKey; //智慧母單號;

BSTR bstrSmartKey; //智慧單號; 出場單則給出場單號

LONG nTradeKind; //11:當沖;

LONG nDeleteType; //刪單類型 1: 全部; 2: 進場單 ; 3:出場單

BSTR bstrSeqNo; //委託序號(1~3三種類型均需提供, 預約單可忽略)

BSTR bstrOrderNo; //((刪進場/出場單提供))委託書號，（若觸發，需給書號）

BSTR bstrSmartKeyOut; //出場單號(**刪全部時提供**，刪進場單及出場單時不需提供出場單號) //委託量大於1時，會不止一筆出場單，出場單號以逗號「, 」區隔

}

＊若被動查詢回報結果，當無書號時，則不需提供；若無尚無出場單號，則可不填。

＊刪全部及刪進場單時，智慧單母單號與智慧單號相同。

#### 證券出清

struct CANCELSTRATEGYORDER

{

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

LONG nMarket; //市場別1:國內證券

BSTR bstrSmartKey; //智慧單號; 出場單則給出場單號

LONG nTradeKind; //17:出清

BSTR bstrSeqNo; //委託序號 (預約單可忽略)

BSTR bstrOrderNo; //委託書號（若觸發，需給書號）

}

#### 證券二擇一OCO、MIT、MST、AB、CB

struct CANCELSTRATEGYORDER

{

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

LONG nMarket; //市場別1:國內證券 (AB單為1:國內證、2:國內期、3:國外證、4:國外期)

BSTR bstrSmartKey; //智慧單號

LONG nTradeKind; //3:OCO、8:MIT、9:MST、10:AB、27：CB

BSTR bstrSeqNo; //委託序號 (預約單可忽略)

BSTR bstrOrderNo; //委託書號（若觸發，需給書號）

BSTR bstrLongActionKey; //長效單號(非長效單請輸入0)

}

比較表

|  |
| --- |
| 異動後V2.13.45 |
| **智慧單刪單物件新增長效單號、新增刪單別：**非長效單之刪單，長效單號可忽略 |
| 異動後V2.13.48 |
| **移除MBA、LLS刪單。**  **非長效單請於長效單號輸入0。** |

#### 智慧單多筆刪單

struct CANCELSTRATEGYORDER

{

BSTR bstrFullAccount; //帳號

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

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

}

#### 證券智慧單—可刪單狀態參考

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 代碼 | 是否可刪單 | 當沖(進場單) | 當沖(出場單) | MIOC單 | 當沖(刪全) | 出清 |
| 32 | Y | Y | Y | Y | Y | Y |
| 34 | Y | Y | Y | Y | Y | Y |
| 35\* | Y |  |  |  |  |  |
| 37 | Y | Y | Y | Y | Y | Y |
| 38 | Y | Y | Y | Y | Y | Y |
| 42 |  | Y |  |  | Y |  |
| 43 |  | Y |  |  | Y |  |

##### 代碼參考表

|  |  |  |
| --- | --- | --- |
| 代碼 | 說明+前台顯示訊息 | Message |
| 32 | 中台收單成功 | [智慧單號][股票代號]收單 ex. [375][1101]收單 |
| 34 | 洗價中 | [智慧單號][股票代號] 已加入洗價 ex. [375][1101]已加入洗價 |
| 35 | 洗價中-觸發價更新(移動停損單) | [智慧單號][股票代號] 觸發價更新 |
| 37 | 洗價觸發 | [智慧單號][股票代號]觸發條件觸發 ex.[375][1101]成交價觸發 |
| 38 | 觸發下單 | [智慧單號][股票代號]Tandem回覆訊息  ex. [353][1101]委託資料傳送交易所中! |
| 42 | 進場單委託中(當沖) | MIT進場->[智慧單號][股票代號]進場單-MIT觸發，送出委託 |
| 43 | 進場單成交，執行出場條件(當沖) | [智慧單號][股票代號]進場單成交，執行出場條件 |