

Advantech Web Studio Version 5.0

中文操作手冊 – 2001/9/6



E-mail taiwan@advantech.com.tw

I. Advantech Studio 軟體介紹

研華的 Studio™ 是一套完整的全方位 SCADA(Supervisory Control And Data Acquisition)或 HMI (Human-Man Interface)自動化設計軟體,方便使用者在 Microsoft® Windows® NT/2000/CE 的平台上開發。

此套軟體包括動態的操作介面,多種 PLC 的驅動程式,I/O 點資料庫,警報點處理,趨勢圖,配方管理,時程管理以及安全管理. 同時與其他軟體可利用工業標準相容的介面互相結合如:ODBC, DDE, NetDDE, OPC 或 TCP/IP 通訊協定.


此套軟體共分二個版本:

- 發展版:可讓使用者利用一般桌上型電腦,筆記型電腦或工業級電腦在 Windows® NT/2000 平台上設計開發人機介面監督控制應用程式。
- 執行版:僅允許使用者在 Windows® NT/2000 or Windows® CE 平台上在操作而無法作發展的動作. 若是使用 Windows® CE 的平台(或稱 CEView)Studio 的執行版通常已事先建立在 HMI 中. 一旦須要修改時可使用裝有發展的主機, 透過網路直接線上下載新的應用程式到 WinCE 中.

此 Advantech Studio 中文教材適用於所有 Advantech Studio 的使用者. 各章節的指引如下:

I. Advantech Studio 軟體介紹.....	2
I.1. 相關參考手冊.....	3
I.2. 字型提示方式.....	4
I.3. Mouse and Selection Conventions.....	4
I.4. Windows Conventions.....	4
I.5. 系統配備.....	5
I.6. 主要特色.....	5
I.7. Advantech Studio 安裝方法.....	7
I.8. Advantech Studio 解除安裝方法.....	9
I.9. 啓動 Advantech Studio.....	11
II. 開發環境.....	12
II.1. 標提欄.....	13
II.2. 訊息列.....	13
II.3. Menu Bar.....	14
II.3.1. File Menu.....	14
II.3.2. Edit Menu.....	16
II.3.3. View Menu.....	17
II.3.4. Insert Menu.....	22
II.3.5. Project Menu.....	25
II.3.6. 工具列(Tools Menu).....	28
II.3.7. 視窗選單(Window Menu).....	31
II.3.8. 說明(Help Menu).....	32
II.4. Toolbars.....	33
II.4.1. Standard Toolbar.....	33
II.4.2. Tag Properties 工作列.....	35
II.4.3. Execution Control 工作列.....	36
II.4.5. Align 與 Distribute 工作列.....	38
Changing object Layers.....	41
II.4.6. Mode Toolbar.....	42
II.4.7. Bitmap 工作列.....	43
II.4.8. Static objects 工作列.....	44

II.4.9. <i>Dynamic Properties</i> 工作列.....	47
II.4.10. 動態式物件工具箱(<i>Active Objects Toolbar</i>).....	52
II.5. 工作區 (Workspace).....	67
II.5.1. <i>Database</i> 標籤.....	68
II.5.2. <i>Graphics</i> 標籤.....	75
II.5.3. <i>Tasks Tab</i>	81
工作表單主體的選項敘述如下：.....	85
II.5.4. <i>Comm Tab</i>	95
II.6. 翻譯功能.....	105
II.7. Tag Reference.....	107
II.7.1. Tag Syntax.....	107
II.7.2. Tag Field Syntax.....	108
II.7.3. Tag Types.....	109
II.7.4. Tag Values.....	109
II.7.5. Array Tags.....	110
II.7.6. Indirect Tags.....	110
II.7.7. Tag 性質 – 參數.....	111
II.7.8. Tag Properties – Alarms.....	112
II.7.9. Tag Properties – History.....	115
II.8. Functions List.....	116
II.8.1. Messaging Function for LogWin.....	116
II.8.2. Arithmetical Functions.....	116
II.8.3. Statistical Functions.....	119
II.8.4. Logarithmic Functions.....	120
II.8.5. Logical Functions.....	121
II.8.6. Functions for Manipulating Strings.....	122
II.8.7. Functions for Manipulating the Date and Time.....	126
II.8.8. Trigonometric Functions.....	128
II.8.9. Functions for Opening and Closing Windows.....	130
II.8.10. Security System Functions.....	131
II.8.11. Functions for Activating Modules.....	131
II.8.12. Functions for Manipulating Files.....	140
II.8.13. Functions for Printing Graphical Screens.....	144
II.8.14. Functions for Text Translations.....	144
II.8.15. Multimedia Functions.....	145
II.8.16. System Information.....	145
II.8.17. Database Access Functions.....	149
II.8.18. Loops.....	150
II.8.19. ODBC Functions.....	150
II.8.20. Mail Functions.....	158
II.8.21. Dial-up connections.....	161


 **備註:** 此操作手冊適用於已熟悉視窗環境的使用者。若您還未熟悉視窗操作建議您先學習視窗操作再使用此手冊。

I.1. 相關參考手冊

除了此手冊外相關參考手冊如下:

- **Quick Start Manual:** 提供 Advantech Studio™ 最基本的操作程序, 使用者可依照手冊的步驟建立一個很簡單的應用程式並可借由此操作程序加速對此軟體的熟悉度.
- **Tutorial Manual:** 詳細指導您利用產品特性建立一個較複雜的應用程式, 您可將此手冊當成自我教育訓練的教材.




- *Drivers User Guides*: 詳細描述各個 Advantech Studio 所支援的驅動程式定義方式以及欄位的規劃。

 **備註:** 產品操作手冊擺放在 Advantech Studio 光碟片 *Documentation* 目錄中。若您已安裝完 Advantech Studio 在 **DRV** 的目錄下您也可發現 *Drivers User Guides*。您也可以發展的環境下利用 **Help** 選項獲得更多的技術資料。

1.2. 字型提示方式

在手冊當中，不同的主題會用不同的字型來提示。有些訊息也會另外用備註的方式提醒您以方便您很快的閱讀手冊。

- **Titles, labels, new terms, 及 messages** (如 *Object Properties*) 使用斜體字來提示。
- 當可讓您由 **File names** 及 **text** 進入時(如 **d:\Setup.exe**) 使用粗斜體來提示。
- **Buttons, menu options, 及 keyboard keys** (如 **Enter**) 使用窄粗體來提示。
- 若是強烈要求的項目會使用粗體字以引起使用者注意。

有些說明是用提示盒來表示如:  **提示**,  **備註** 及  **警示**。

- **提示(Tips)** 包含了有用的資訊方便使用者節省發展的時間或使應用程式的執行效率更好。
- **備註(Notes)** 包含了額外的訊息以方便使用者更容易了解在備註前所述敘的含意。
- **警示(Cautions)** 包含了必須的訊息以避免執行應用程式時發生錯誤而引起不必要的問題。

1.3. Mouse and Selection Conventions

因為大部分的電腦應用程式都是用滑鼠操作，此操作手冊假設您也使用滑鼠來寫。通常，電腦滑鼠都規劃為左鍵是主要的按鍵右鍵是次要的按鍵。此操作手冊依照使用滑鼠的慣例來提示：

- *Double-click* 是指快速按兩次滑鼠的左鍵。
- *Right-click* 是指按一次滑鼠的右鍵。
- *Click and select* 是指先將滑鼠的游標移到要選擇的項目按左鍵使項目在螢幕上被選定。若是用觸控螢幕時操作方式除了是用手指操作外與使用滑鼠是一樣的。若是用鍵盤操作的話通常需要使用 **Tab** 鍵來移動選項，利用 **Enter** 鍵打開目錄。可用 **Alt** 鍵加上有底線標示的英文字來選擇選單上的項目。
- *Dragging* 是指選定物件後按著滑鼠左鍵然後拖曳到目標位置。

1.4. Windows Conventions


此操作手冊依照視窗的使用慣例：

- *Dialogs, or dialog boxes*, 是一個傳達資訊的對話窗。
- *Text boxes* 會提供一個視窗的空間在當中您可以打入文字。
- *Radio buttons* 在按鈕的中央有個黑點則表示項目被選定當您再按一次則黑點可消失。

- **Check boxes** 用來致能或取消一個選項命令, 當按下滑鼠左鍵會有個“X”符號出現在盒中.
- **Buttons** 是具有圖示的按鈕, 當按下時會有被按下的效果.
- **Lists** 會列出一排下拉式的視窗列出可供選擇的選單.
- **Drop-down lists** 為下拉式選單, 按下向下的箭頭即會列出所有選項.
- 在此操作手冊中 **interface** 是指整個 Advantech Studio 視窗.
- **Toolbars** 為視窗的入口僅包含按鈕及文字盒.

1.5. 系統配備

- IBM PC 相容電腦具有 Intel® Pentium II 相容微處理器.
- 發展版適用於 Windows NT/2000 作業環境
- 執行版適用於 Windows NT/2000 或 Windows CE v3.00
- 至少 32MB 的隨機快取記憶體; 建議最好有 64MB 或較多的記憶體
- IE 4.0 瀏覽器或更高的版本
- 需要 150MB 硬碟剩餘空間(完全預留給 A-Studio 用,其他程式不能使用); 建議最少預留 300MB.
- 3.5" 軟碟機
- CD-ROM 光碟機
- 附有 F1 - F12 功能鍵標準鍵盤
- 平行列印埠(可選用)
- 100% IBM 相容顯示卡,具有 2MB Video RAM (VRAM)
- Microsoft 相容指標配備(如滑鼠,軌跡球或觸控螢幕)
- 一個或二個通訊埠以及下載程式的轉接器(可選用)
- 乙太網路通訊埠(可選用)

 **備註:** Advantech Studio 是用 UNICODE 製作而成因此不適用於不支援 UNICODE 的平台上(Windows 9x/ME). 不過若您僅是使用瀏覽器來監控 A-Studio 畫面時(Web Thin Clients) 並不受此限定.

1.6. 主要特色

以下是 Advantech Studio 主要特色:

- 整合了視窗發展環境如工具列,對話框及選單:
 - 發展環境中在任何區域都可直接按右鍵而出現選項. 顯示出選項這些選項將根據內容來作變化.
 - 您可以自組您所要的工具列.
 - 程序(Tasks), 物件(objects)以及控制項均以樹狀結構展現.

- 多功能物件及動態的使用方式支援使用者設計畫面:
 - 提供按鍵,矩形,橢圓,多邊形,直線及文字等繪圖工具.
 - 提供動態的物件屬性設定例如:動態長條圖(bar graphs), 顏色(color), 改變大小(resizing), 位置(position), 隱藏/顯示(hide/unhide), 旋轉(rotation), 執行命令(command), 網站鍵結(hyperlink)及文字輸出/輸入(text Input/Output).
 - 即時及歷史警報顯示
 - 即時及歷史曲線圖
 - 物件對齊及分散工具
 - 背景圖的產生及編輯
 - 圖形的載入
 - Active-X 介面
- 提供線上遠端即時修改軟體的功能
- 提供 OPC 介面及 XML 格式
- 具有 Web 的功能, 可以讓任何位於 Internet/Intranet 的節點透過 TCP/IP 以瀏覽器線上監控現場畫面並可看到目前即時資料.
- 圖庫中提供 100 多種符號及動態物件圖, 例如: 按鍵(pushbuttons), 計量器(meters), 滑動器(sliders), 開關(switches), 文字(text)及數字(numeric)顯示, LED 指示燈(LED-style indicators), 導管(pipes), 幫浦(pumps), 圖標 (icons), 車輛(vehicles), 閥門(valves), 框架(frames), 馬達(motors), 精密量測器(gauges), 控制元件(common controls).
- 除錯工具:
 - 資料庫追蹤視窗可監控資料點及強迫其輸出入值同時也允許呼叫功能函數.
 - 視窗日誌(LogWin)可記錄 OPC,DDE 及 TCP/IP 的傳輸狀況或訊息並可追蹤某個點運作狀態.
 - 在 project 的任何地方均可將資料點(Tag)作交互的參考.
 - 線上系統及網路偵錯.
- 具有強大及彈性化的資料庫(使用者可將資料點設計成陣列, 間接定址點, 分類點, 布林點, 整數點, 實數及字串點).
- 提供 API 函數可讓外部程式與 A-Studio 作資料交換.
- 提供多國語言的轉換編輯器, 可讓使用者在程式執行時作線上轉換.
- 提供 TCP/IP 主從模組, 可使不同的電腦透過 TCP/IP 模組相互交換現場值也可經由此功能建立備份(redundancy)系統.
- 提供 200 多種不同設備(如 PLC)廠牌的驅動程式例如:Allen-Bradley, Siemens, GE-Fanuc, 以及 MODBUS RTU/ASCII,DeviceNet,Profibus,Interbus 等標準的通訊協定.
- 支援 OPC, 可當 OPC client.
- 提供 256 個層級密碼鎖, 可在執行期時保護畫面及任何物件的操作.
- 提供 200 多種函數方便使用者撰寫程式.
- 配方及報表功能(ASCII and RTF format)已結合於產品當中.
- 時程管理功能可依固定時間,間隔時間或事件發生時執行所指定之程序(最快執行頻率可到 100ms).
- 應用程式的視窗可互相重疊, 也就是說在發展環境中您可以將工作表單及繪圖草圖作模組式的重疊.
- 可與規劃 PC-based 控制器所使用的套裝軟體整合(可將其資料庫直接匯入)如 ISaGRAF, SteepleChase, Think&Do, ASAP 等.

- Real-time project documentation
- Screen resolution converter

 **備註:** Advantech Studio 不同的版本(例如 CEView)有部份功能是不支援的.詳細的文件請參考光碟片中的 *TargetVersions.pdf* .

I.7. Advantech Studio 安裝方法


Advantech Studio 可執行於 Microsoft Windows® NT/2000. 安裝程式會新增一個目錄同時將所需檔案複製到硬碟中,同時也會新增捷徑在您的桌面上.

Advantech Studio 包裝在一片光碟片中, 您可以直接從光碟中安裝或是將安裝程序作成 3.5" 磁碟片再安裝.

Advantech Studio 可以使用同樣的開發環境設計不同平台的應用程式. 您可經由直接串列埠或 TCP/IP 與 Windows CE 平台的人機介面連結,利用這樣的介面將作好的 Windows CE 應用程式下載到人機介面上.

 **備註:** 若您是使用 Windows NT 的作業系統, 在安裝 Advantech Studio 時請確認您必需是 Administrator 的權限.當您有新的 Advantech Studio 版本要更新時, 建議先將舊的版本解除安裝.

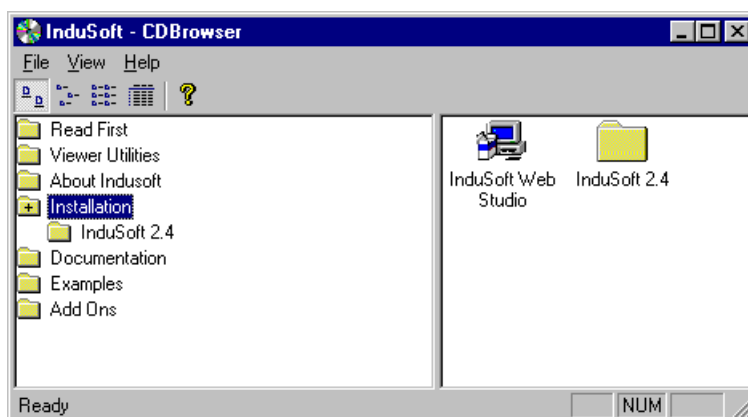
請使用以下的步驟安裝 Advantech Studio:

 **備註:** 您可以直接從光碟中安裝或是將安裝程序作成 3.5"磁碟片再安裝. 若您要作成 3.5"磁碟片, 請將 `d:\Installation\Disk1\` 的資料夾(`d` 是光碟機的磁碟路徑)複製到磁碟片的#1. `d:\Installation\Disk2\` 複製到磁碟片的#2, 其他以此類推. 當要安裝時將編號#1 磁碟片插入磁碟機中, 然後執行 **Setup.exe**. 請依照安裝精靈完成安裝.

1. 啓動您的電腦進入 Windows NT/2000 操作系統, 同時確認並無其他應用程式在執行.

2. 放入準備安裝的光碟片或 3.5"磁碟片.

螢幕會自動出現光碟片瀏覽畫面. 假如並未自行出現瀏覽畫面, 請用檔案總管在 **d:\Installation**(d 指光碟機的位置)目錄下執行 **Setup.exe**.



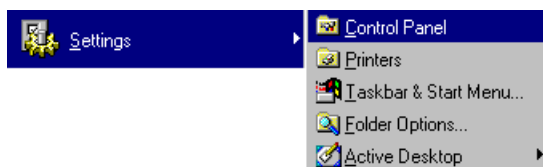
3. 選取 **Installation** 目錄並執行 **Advantech Studio** 圖示開始進入安裝精靈. 安裝進度的對話盒將會載入安裝程序.
4. 依照安裝指示進行安裝.
5. 當安裝程序完成後螢幕會出現需要重新啓動電腦的對話框, 請選擇 **Yes**, 然後按 **I want to restart my computer now** 的選擇鈕按 **OK**.
6. 重新開機後就可以執行 **Advantech Studio** 開始規劃(p.11).

I.8. Advantech Studio 解除安裝方法

假如您需要解除安裝, 請依照以下的操作說明:

⚠ **警告:** 在進行解除安裝之前, 請先確認您已備份...**Advantech Studio** 目錄下將來還會使用的檔案. 為了安全起見也請您確認已保存了安裝的光碟片(或磁碟片)以便可以再重新安裝(新的版本或同樣版本).

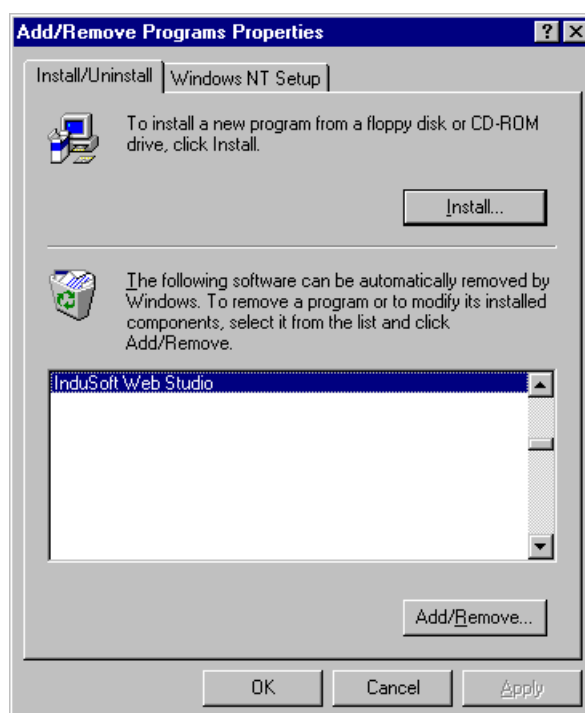
1. 在開始欄中選**設定->控制台**.



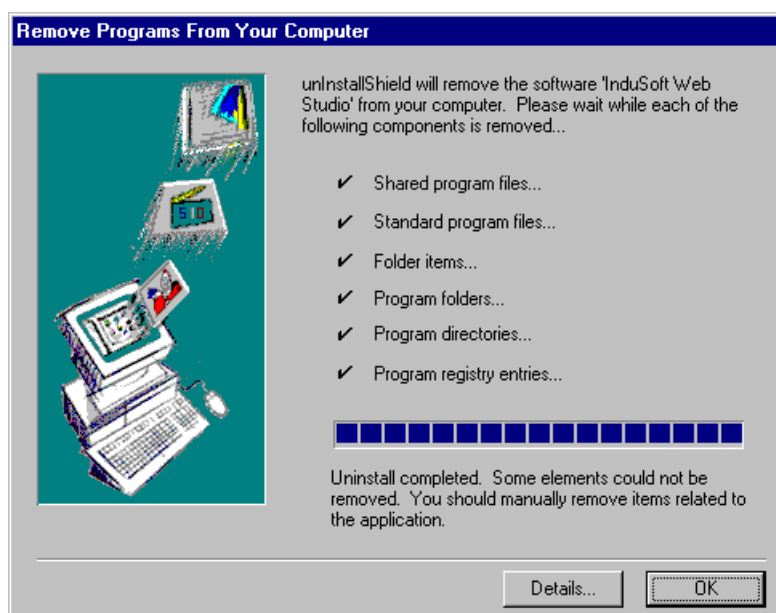
2. 在**控制台**中點取新增移除程式圖式按鈕(icon).



3. 出現新增移除畫面在視窗清單中選擇 **Advantech Studio**, 然後按新增/移除按鈕.



- 當出現移除軟體的對話框時請按確認按鈕, 移除程序將自動完成.



- 當出現解除安裝程序完成的訊息時請按確認, **Advantech Studio** 將不再顯示在清單中.
- 按取消鍵或直接關掉視窗(**X**), 結束控制台視窗.
- 選開始->程式集->檔案總管.
- 確認 **Advantech Studio** 目錄已被移除了, 手動刪除任何遺留的檔案.

 **備註:** 在 ...**Advantech Studio**\Projects\ 目錄下新增或修改的檔案將不會因為解除安裝而被刪除.

I.9. 啓動 Advantech Studio

依照以下步驟開始執行 *Advantech Studio*:

1. 執行桌面上的 *Advantech Studio* 或是執行開始->程式集->*Advantech Studio Tool*.



即可進入 *Advantech Studio* 開發環境.

提示: 你可以用任何像素來發展 *Advantech Studio*. 但是, 強烈建議您使用 800x600 (或更高)的解析度及超過 256 色的開發環境來設計. 應用程式的解析度(螢幕大小) 是依據操作平台的設定.


II. 開發環境

Advantech Studio 依循視窗的樣式並且採用標準的工具及介面使系統更具親和力(user-friendly). 在開發環境中可以快速且容易的取得工具或資訊.



開發環境是由以下基本區塊組合而成:

- 標題欄(*Title Bar*)
可顯示在視窗中的應用程式或工作區
- 訊息列(*Status Bar*)
此欄位於視窗的最底部, 用來快速的顯示目前狀態或訊息
- 主選單(*Menu Bar*)
包含了應用程式選單的名稱及控制項可以方便使用者用滑鼠或鍵盤選擇所要執行的功能.
- 輔助工具列(*Auxiliary Tool Bars*)
在發展環境中提供主要功能的捷徑.
- 繪圖工具列(*Displays Building Tool Bars*)
提供物件屬性設計及工具方便使用者可以新增及編修物件或是作動態顯示
- 工作區(*Workspace*)
提供樹狀的操作方式讓使用者可以直接進入物件工作表單進行設定工作
- 資料庫偵測視窗(*Database Spy Window*)
提供除錯工具讓使用者可以監視某特定點或強迫輸出值甚至直接執行函數
- 輸出視窗(*Output Window*)
顯示除錯時的訊息
- 顯示/工作區(*Displays / Workspace*)
提供一個區域在此區域可以繪圖及設計工作表單


 **備註:** 先前的畫面是以預設畫面來顯示. 您可以依據您的需求自行調整發展環境中各區域的大小及位置.

II.1. 標提欄



標提欄包括了以下項目(從左至右):

- **Advantech Studio** 圖示及名稱.
- 正在執行, 開啓的畫面或工作表單名稱.
- 最小化盒(☐) – 按此鍵可以將 **Advantech Studio** 畫面縮小.
- 改變大小/最大化盒(☐/☐) – 此兩個按鍵是互相變換的. 改變大小盒可以使 **Advantech Studio** 與其他視窗如磚塊般堆疊而最大化盒可以使其回復到最大.
- 離開盒(☒) – 按此鍵可使 **Advantech Studio** 自動儲存資料庫後關閉. 若您有修改過的畫面或工作表單則會出現提示畫面提醒您先儲存. 此按鍵與檔案中的離開類似.

 **備註:** 關閉發展環境並不會連同執行系統一起關閉.

II.2. 訊息列



狀態列可顯示目前按下了那個工具列的按鈕及提供正在運作中的畫面資訊. 顯示的區域如下(從左到右):

- **Hint field**
顯示簡短的敘述所按下的工具鈕或是目前滑鼠所選定的物件.
- **Caps Lock field**
顯示鍵盤的大寫鍵(**Caps Lock**) 是否已鎖定.
- **Num Lock field**
顯示鍵盤的數字鍵(**Num Lock**)是否已鎖定.
- **Scroll Lock field**
顯示鍵盤的螢幕鎖定鍵(**Scroll Lock**)是否已被鎖定.
- **ID field**
顯示畫面上所選定物件的辨識碼(ID).
- **Screen Coordinate field**
顯示目前運作中畫面上的游標座標, **X** 是橫座標 **Y** 是縱座標.
- **Object Size field**
顯示所選物件的大小 **W** 是寬度 **H** 是高度.
- **No DRAG field**
顯示運作中的畫面的拖曳功能是否是致能.

II.3. Menu Bar

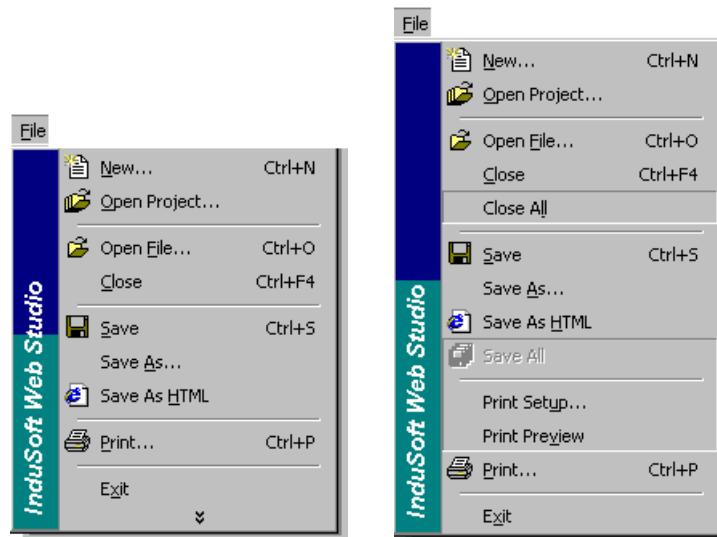
File Edit View Insert Project Tools Window Help

選單欄中包括了檔案(**F**ile), 編輯(**E**dit), 檢視(**V**iew), 插入(**I**nsert), 專案(**P**roject), 工具(**T**ools), 視窗(**W**indow)及說明(**H**elp).

備註: 主選單是所有選項的入門. 在選單上按左鍵彈跳出選項. 一旦選項被選後即會反白.

II.3.1. File Menu

File 選單包含了可以讓您管理檔案的命令及工具列.

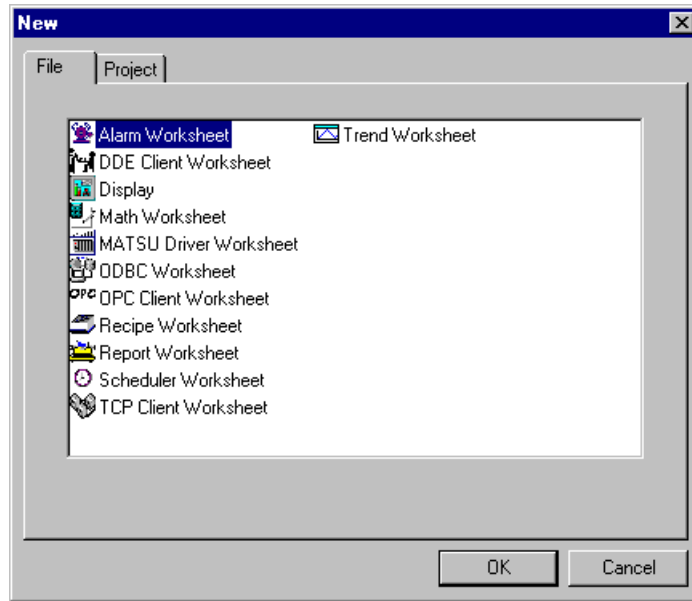


New...

打開新的視窗其中包含了 **File** 及 **Project** 標籤, 在當中您可以新增一個 application (project)或應用程式中的某個附屬程式(Display, Math worksheet, 等). 您也可以在標準的工具列中選擇 **New** 或是從 **Insert** 選單中選 **Document...**

File 標籤中允許您選用新的 *Alarm*, *DDE Client*, *Math*, *ODBC*, *OPC Client*, *Recipe*, *Report*, *Scheduler*, *TCP Client*, 及 *Trend* 工作表單或新的畫面. 當您在應用程式中新增一個 I/O 驅動程式時, 也可以利用其中的選項增加新的工作表單. **Project** 標籤中允許您新增一個專案(project).

備註: DDE Client 及 ODBC 的工作表單並不會顯示在 Windows CE 的應用程式中 (此兩種功能並不支援 WinCE).



Open Project...


顯示 *Open* 視窗, 在當中您可以操縱及打開另外一個 *Advantech Studio* 應用程式. 您也可以利用檔案總管到專案(project)所在目錄直接執行或在標準工具列中按 **Open Project**.



Open File

開啓 *Advantech Studio* 應用程式. 從所開啓的視窗中, 您可以從下拉選項中選擇所需要的檔案型態.

Close

關閉執行中的螢幕或工作表單. 程式將提醒您在離開前儲存檔案. 此按鍵()與檔案中的關閉類似.

Close All

關閉所有執行中的畫面或工作表單. 程式將提醒您在離開前儲存檔案.



Save

儲存任何執行中及打開的工作表單或畫面. 您也可以利用標準工具列中 **Save** 的按鍵來儲存. 此 **Save** 鍵只有在你曾更改檔案時才會被顯示.

Save As

儲存執行中的工作表單或畫面並且在儲存時可以選擇檔案儲存位置及檔名.



Save As HTML

將執行的畫面儲存成 HTML 的格式.



Save All

儲存所有開啓的工作表單或畫面. 您也可以利用標準工具列中 **Save All** 的按鍵來儲存. 此 **Save All** 鍵只有在你曾更改檔案時才會被顯示.

Print Setup...

允許您設定列表機. 設定預設印表機時, 選擇開始->設定->印表機. 在您相要的印表機上按右鍵, 然後在彈跳式的選單中選 **Set As Default**. 假如在選完後出現檢查窗, 被選用的印表機將會變為預設的印表機.

Print Preview

此命令與標準的視窗 *Print Preview* 命令相同. *Print Preview* 視窗會被顯示在工作表單上以利您可以事先看到列印出來的樣子. 您可以利用 **Zoom In** 來檢察細項且可以用 **Zoom Out** 來變回預設的大小尺寸. 您可以看下一頁 (**Next Page**), 前一頁 (**Prev Page**), 或同時看

二頁 (Two Page). 在任何時候, 您可以用 **Print...** 來列印, 此方式就如同在視窗中按 **Print**, 您可以按 **Close** 來關閉 **Print Preview** 視窗.



Print

開啓 **Print** 視窗. 您可以列印正在使用的畫面或工作表單. 除此之外, 您可以指定印表機的名稱, 屬性, 及複印的張數. 您可以在標準工具列中選擇 **Print** 鍵來列印檔案.

Previous File List

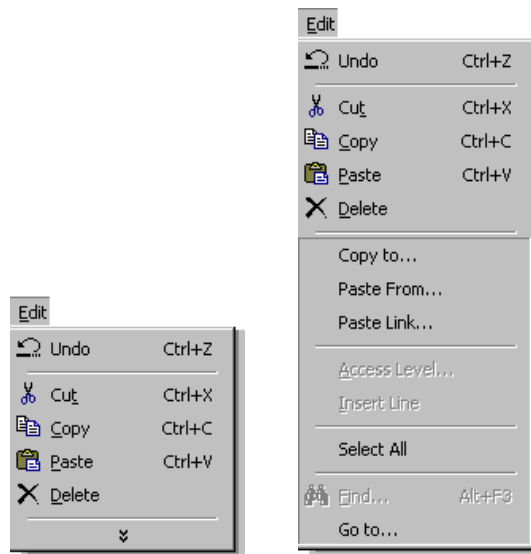
列出最近開啓使用的檔案. 選擇檔案並開啓.

Exit

關閉 **Advantech Studio** 並自動儲存資料庫. 若您有做了修改在關閉前將會提示您作儲存的工作. 此項目就如同在提示列中的 **Exit** 鍵 (X).

II.3.2. Edit Menu

此選單包含了命令及工具使您可以編輯畫面及工作表單.



Undo

復原最後的指令或刪除您最後輸入的項目. 最多可復原 20 個動作. 若要同時復原超過一個以上的動作, 按一下 旁邊的箭號, 然後按一下您要復原的動作. 如果您不能復原上一個動作, 指令的名稱改變至 [無法復原].



Cut

移除作用文件中的選定範圍, 並將其黏貼到剪貼簿上. 您可以使用 **Cut** 將所選擇的物件移到螢幕另外的地方或是另一個螢幕上. 您也可以標準工具列上使用 **Cut** 鍵.



Copy

將選定範圍複製到剪貼簿上或將所選擇的物件移到螢幕另外的地方或是另一個螢幕上或作多個物件的複製, 您也可以標準工具列上使用 **Copy** 鍵.



Paste

在插入點插入剪貼簿的內容, 並取代任何選定範圍. 當您剪下或複製物件、文字或儲存格內容時, 才可使用此指令, 您也可以標準工具列上使用 **Paste** 鍵.



Delete

刪除選取物件或文字, 而不會將它置於剪貼簿上. 只有選取物件或文字時, 才可使用此指令. 您也可以標準工具列上使用 **Delete** 鍵.

Copy to...

開啓 *Save As* 的視窗同時複製已被選用的元件 (一個物件或由物件所組合而成的群組的屬性) 使用 *Advantech Studio* 特定格式存入檔案. 這些物件會有靜態或動態的屬性, 點陣圖物件也是如此.

 **提示:** 您可以使用 **Copy to...** 命令來存入自訂的物件, 這些物件可在工作區 (*Workspace*) 中 **Graphics** 標籤的 **Symbol** 目錄中快速的取得. 選擇您想要儲存或複製的物件 (或群組化的物件) 到應用程式的 **\Symbol** 路徑下. 您必須將檔案存為 **.sym** 的附加檔名.

Paste From...

將一個圖型符號, 圖形, 或將剪下的檔案載入畫面中. **.sym** 檔中的物件是靜態且多樣的屬相. **.bmp** 檔即是點陣圖 (此文件可以讓您轉換成點陣圖的物件). **.cut** 檔即是點陣圖 (此文件可以讓您轉換成點陣圖的物件).

Access Level

在所使用的表單中設定保密權限.

Insert Line

在工作表單中插入一行.

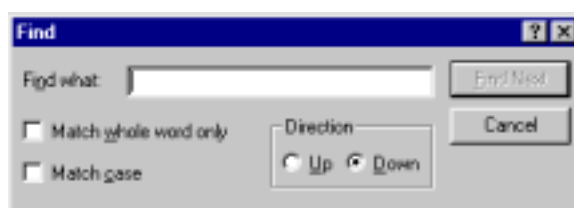
Select All

選擇所有在畫面上的物件.




Find...

開啓 *Find* 視窗使您可以在使用中的工作表單中尋找字.



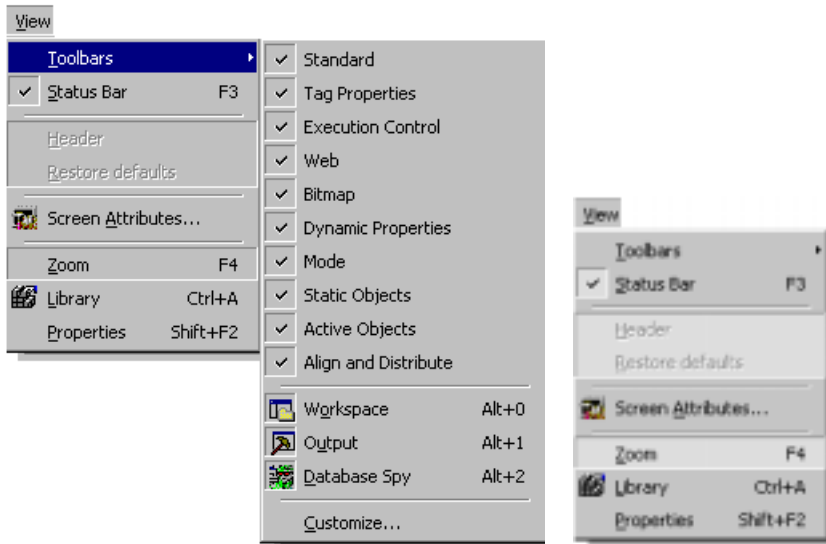
Go to...

跳到工作表單中的某一行或在畫面中物件的編號. *Advantech Studio* 會將物件依序編號, 會由 0 開始編起. 當您選到物件時物件編號會顯示在螢幕下方的狀態列中




 **提示:** 假如您有很多重疊在一起的物件時, 您將很難去用指標指到所要的物件, 此時您就可以用 **Go to...** 來直接跳到所要的物件上.

II.3.3. View Menu

此選單包含了可以讓您管理工具列的隱藏或顯示. 這些工具列通常都以捷徑的方式擺放在工作列

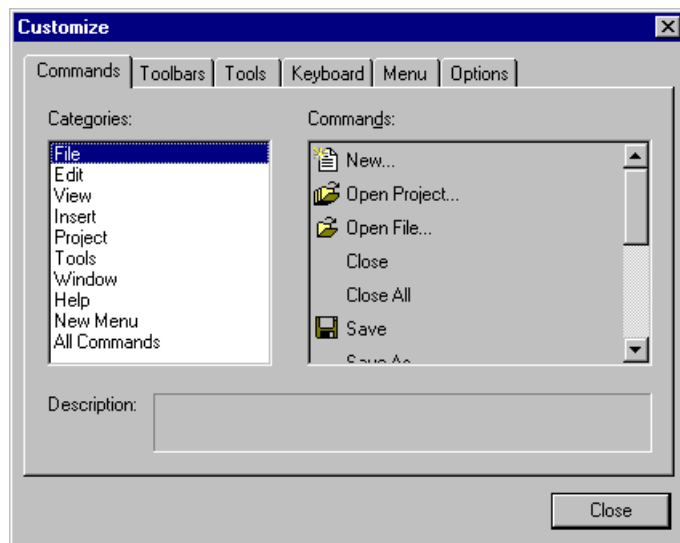


Toolbars

可以讓您隱藏或顯示工具列中的工具, 例如  *Workspace*,  *Output* 及  *Database* 追蹤視窗. 使用 **C**ustomize... 選項可自訂在發展環境中顯示的工具項目, 自訂的項目有:

- **Commands**

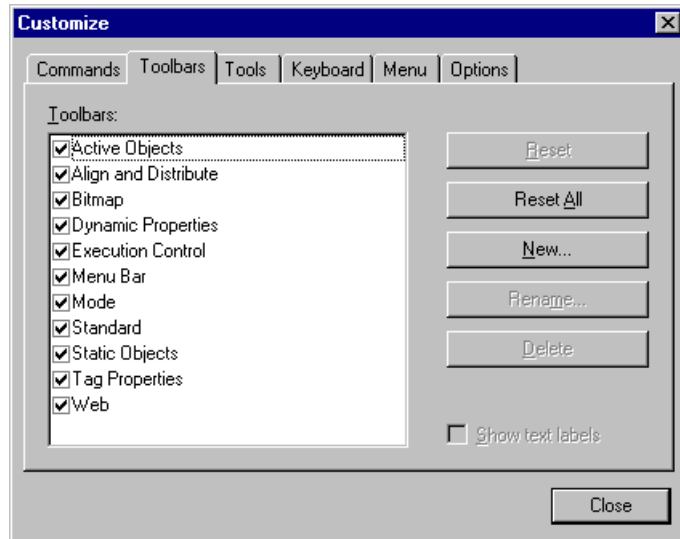
自訂選單的項目. 您可以在命令清單中選擇任何命令然後拖拉到發展環境中任何的選單列或工具列.



- **Toolbars**

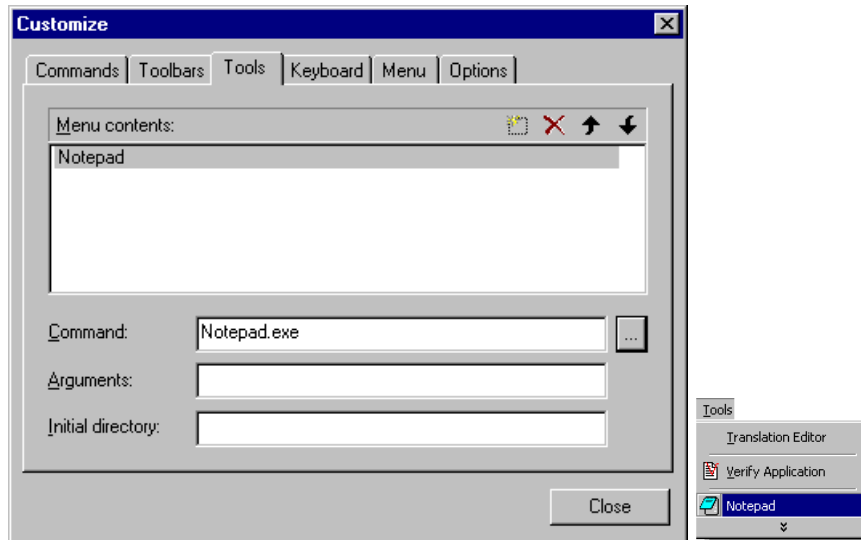
自訂工具列. 您可以設定在 *Toolbars* 清單中任何工具列的顯示或隱藏. 使用 **R**eset 鍵可恢復所選工具列的預設值. 使用 **R**eset **A**ll 鍵可恢復所有工具列的預設值. 使 **N**ew... 鍵去新增一個工具列.

新增之後您可以從 **C**ommands 中拖拉圖示到新增的工具列. 使用 **R**ename 鍵可以將您新增的工具列重新命名或用 **D**elete 鍵除去. **S**how text labels 的複選框可以讓您選擇要不要顯示所選用工具列圖示的標籤.



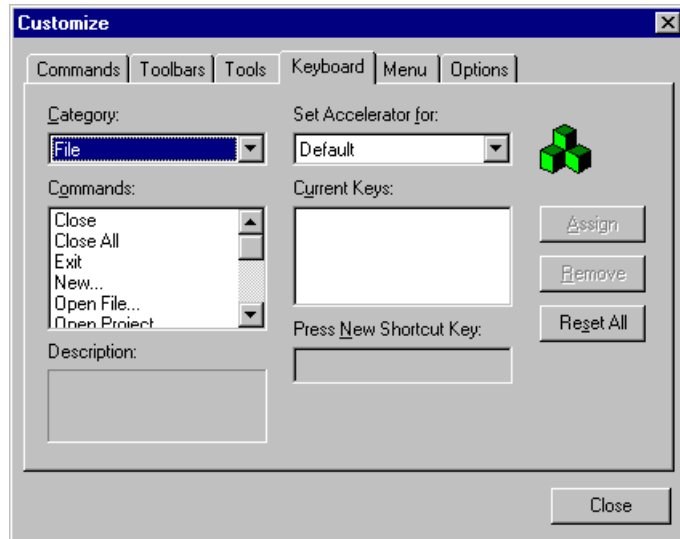
- **Tools**

自訂 **Tools** 選單的選項. 你可以新增任何外部程式的捷徑使得外部程式可以在 **Tools** 的選單中被選用. 當要新增捷徑時, 請按  **New (Insert)** 然後設定 **Command**, **Arguments**, 及 **Initial directory** 選項. 使用  **Delete** 鍵除去所選的捷徑, 您也可以使用   **Move Item** 去轉換在 **Tools** 選單中捷徑的位置.



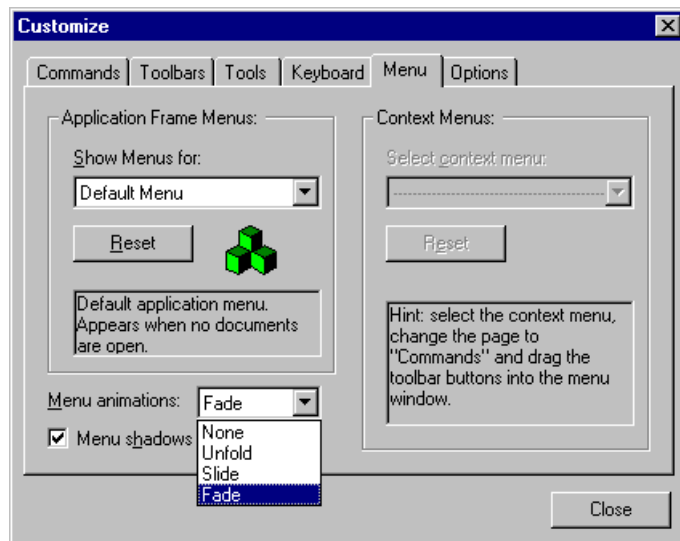
- **Keyboard**

在命令選單中自訂快速鍵. 利用種類 (**Categories**) 及命令 (**Commands**) 清單去選用所要的選項. **Current Keys** 欄位顯示了被選用的命令所指定的捷徑. 您可以在 **Press New Shortcut Key** 欄位中鍵入捷徑以規劃一個新命令的捷徑鍵, 完成後按 **Assign**. 您可使用 **Remove** 鍵移掉已選的捷徑鍵同時也可使用 **Reset All** 鍵將預設值重新回復.



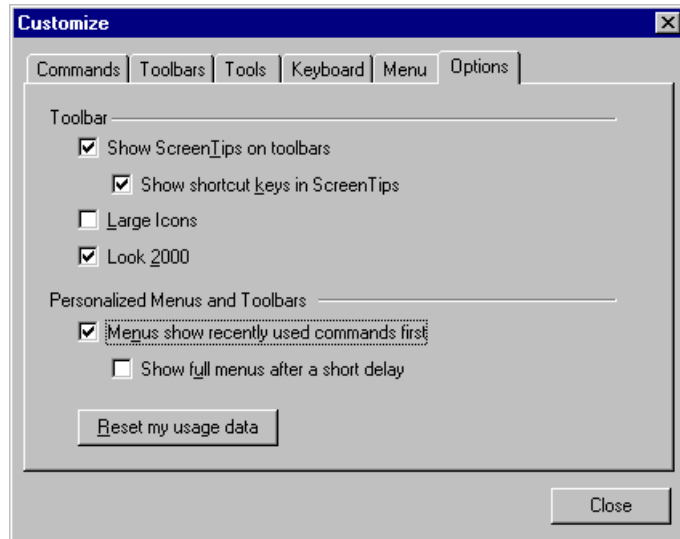
- **Menu**

您可使用 *Menu animations* 下拉式選項來設定彈跳式(pop-up)選單的效果(*None, Unfold, Slide or Fade*). 勾选 *Menu shadows* 檢查盒使得彈跳式(pop-up)選單有陰影的效果.



- **Options**

使用者可自定工具列及選單一般的顯示樣子. 使用 *Reset my usage data* 鍵可將預設設定值回復.



Status Bar

可在螢幕下方顯示狀態列. 勾选了检查框, 则会显示状态列.

Header

当使用工作表单时即会显示. 当您选用此项设定则工作表单将会具有表头. 如果不选用此项, 就不会显示表头.

Restore defaults

在开发环境中回存预设设定值.




Screen Attributes...

开启 *Screen Attributes* 对话框您可以在其中设定画面的内部相关设定(在开发环境中).

Zoom


提供一個分割畫面, 放大您所指定的局部畫面.

 **提示:** 按二下按鍵可以減少放大的效果回到 100%. 雙擊(Double-click)則會增大到 3200%.



Library

開啓事先已規劃好的圖庫. 您也可以按下在標準工具列中的圖示.

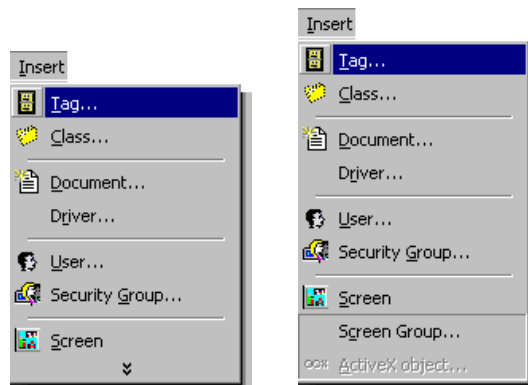
 **提示:** 此圖庫提供了許多種動態的設定. 您可以使用此圖庫直接將物件放到畫面中使用來節省應用程式發展的時間. 您也可以在所繪製的圖面上按右鍵選 *Send to library* 將畫面加到圖庫中. 應用程式會將所有畫面上的物件一起加到圖庫中.

Properties

開啓 *Object Properties* 對話框, 在此您可以規劃物件的參數及各屬性的設定.

II.3.4. Insert Menu

此選單包含了可以讓您新增及設定 tags 的命令.



Tag...

打開 *New Tag* 對話框, 在當中您可以新增一個 tag 及設定 tag 的主要屬性. 您可以在 *Database* 的工作區域 (*Workspace*) 中選 *Application Tags* 目錄中的 tag list 按右鍵 *Insert Tag* 即會彈跳出以下畫面.

New Tag

Name:

Array Size:

Type:

Description:

Web Data:

OK Cancel



Class...

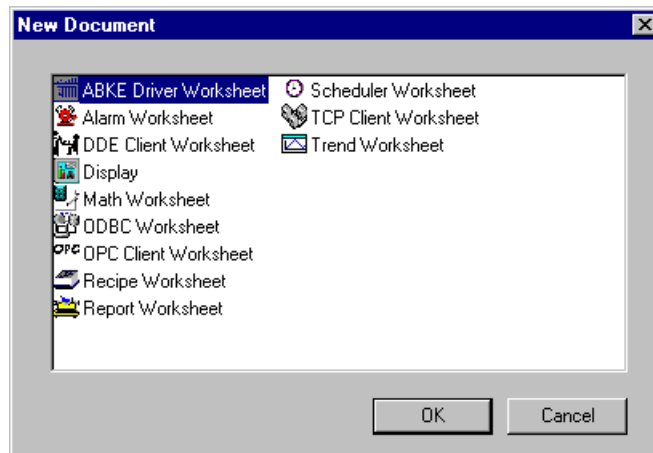
打開 *Insert Class* 對話框, 在當中您可以新增一個 tag 的 class 及設定 tag 的主要屬性.

您也可以在 Database 的工作區域(Workspace)中選 **Classes** 目錄中的 class list 按右鍵 **Insert Class** 即會彈跳出以下畫面.



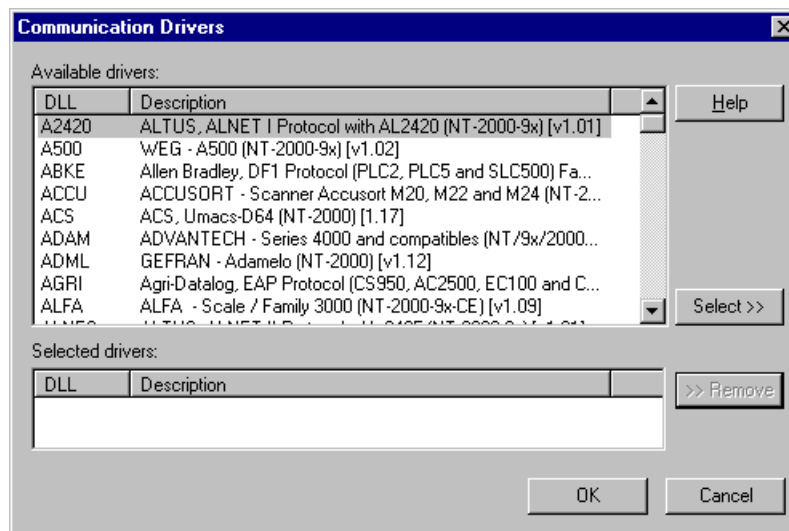
Document...

打開 **New Document** 對話框, 在當中您可以新增一個畫面或工作表單. 您也可以用 **File->New** 來新增.



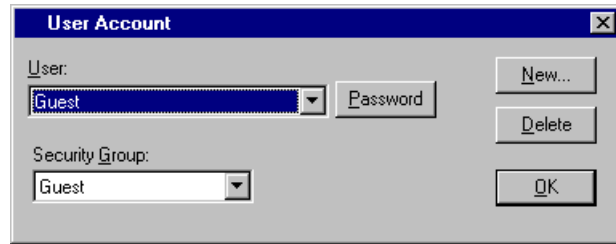
Driver...

打開 **Drivers** 對話框, 在當中您可以新增一個與設備(如 PLC)通訊的驅動程式. 當要新增時, 可以從清單中選擇所需要的驅動程式然後按 **Select >>**. 您也可以在 **Comm** 的工作區域(Workspace)中選 **Drivers** 目錄按右鍵 **Add/Remove drivers** 即會彈跳出以下畫面.



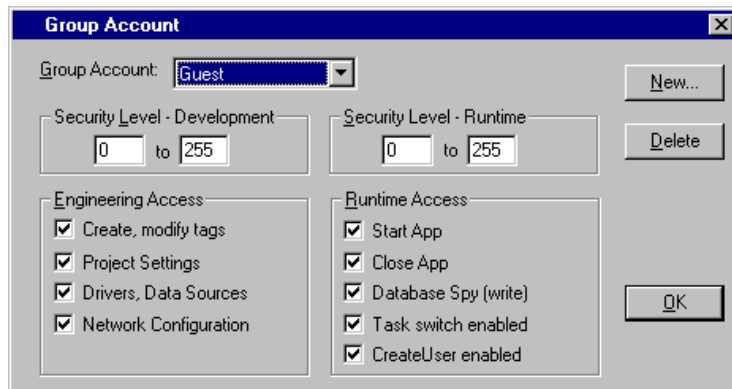
User...

打開 **User Account** 對話框, 在當中您可以在保密系統中新增一個使用者. 您也可以在 Database 的工作區域(Workspace)中選 **Users** 目錄按右鍵 **Insert user** 即會彈跳出以下畫面.



Security Group...

打開 *Group Account* 對話框, 在當中您可以在保密系統中新增一個群組. 您也可以可以在 *Database* 的工作區域(*Workspace*)中選 **Groups** 目錄按右鍵 *Insert a group* 即會彈跳出以下畫面.

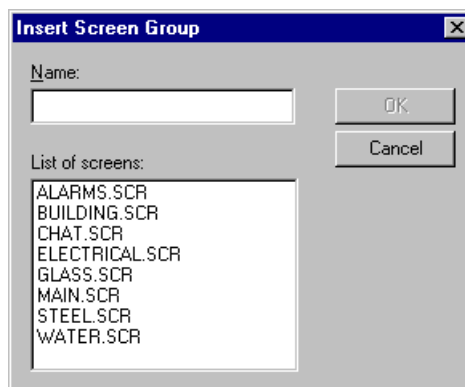


Screen

新增一個畫面. 您也可以可以在 *Graphics* 的工作區域(*Workspace*)中選 **Screens** 目錄按右鍵 *Insert* 即會彈跳出以下畫面.

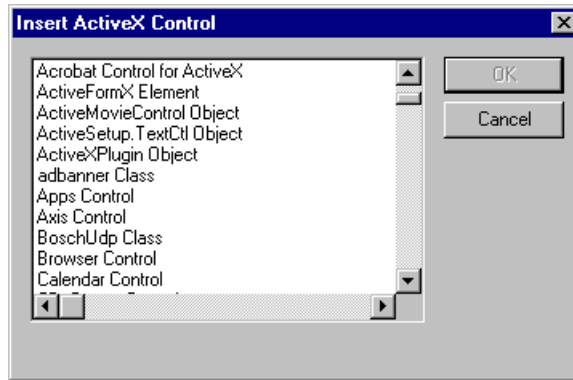
Screen Group...

打開 *Insert Screen Group* 對話框, 在當中您可以新增一個群組畫面. 您也可以可以在 *Graphics* 的工作區域(*Workspace*)中選 **Groups Screen** 目錄按右鍵 *Insert screen group* 即會彈跳出以下畫面.



ActiveX object...

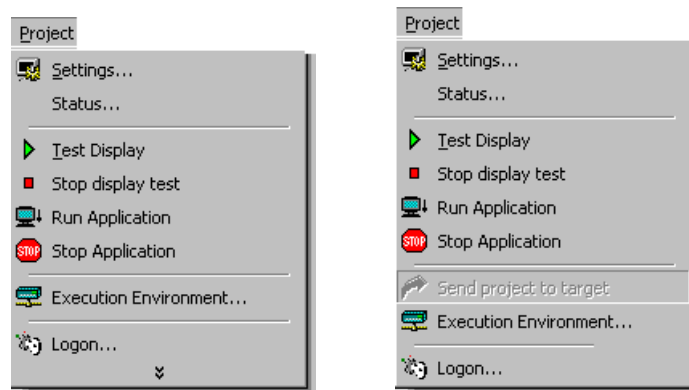
打開 *Insert ActiveX Control* 對話框, 在當中您可以在螢幕中新增一個 ActiveX 元件. 您也可以直接工作列中的 **Active** 來將元件加入.



備註: 當您按下 *Insert ActiveX Control* 時會列出所有目前在電腦中已有註冊登記的 Active 元件. 在您新增一個 ActiveX 元件時您必須用視窗的命令 `regsvr32 <ControlName>` 進行註冊. 例: `regsvr32 e:\winnt\system32\ISSymbol.ocx`.

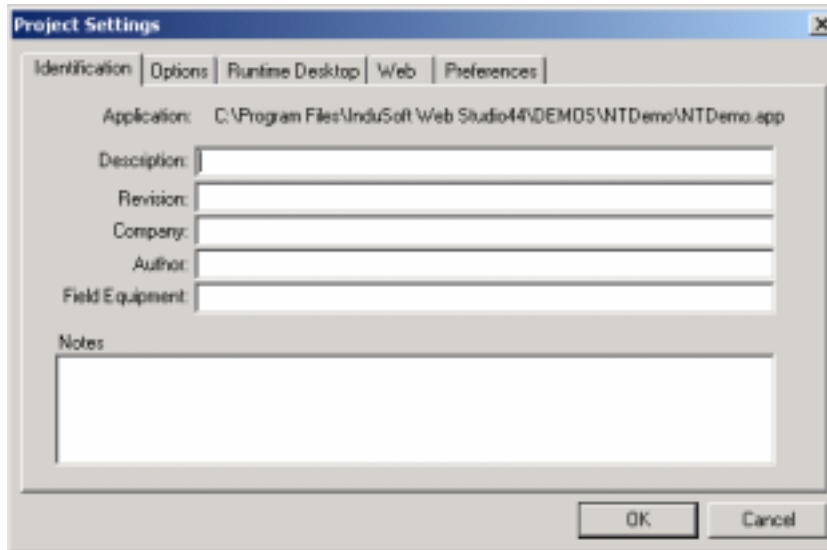
II.3.5. Project Menu

此選單包含了管理執行本地端及遠端應用程式的命令及工具. 同時也提供連結, 連到應用程式的一般設定.



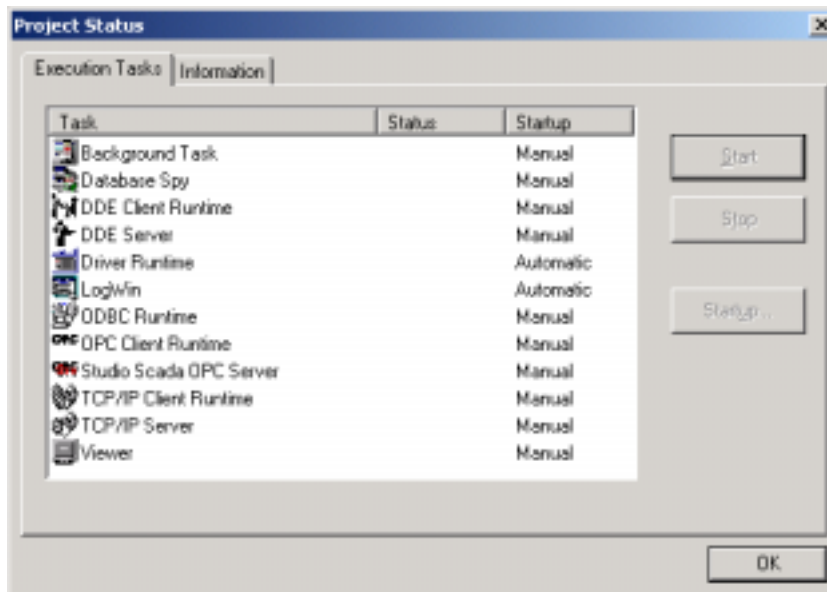
Settings...

可開 *Project Settings* 視窗會有五個 tab: Identification, Options, Runtime Desktop, Web, 及 Preferences. 所出現的控制設定的對話框將影響到應用程式.



Status...

打開 *Project Status* 視窗會有兩個 tabs: **Execution Tasks** and **Information**. **Execution Tasks** tab 包含了程序清單, 以及各程序的狀態及啟動方式(自動或手動). 您可以設定在應用程式開始執行時連同執行所指定的程序. 您也可以手動停止或啟動任何正在執行的程序. **Information** tab 提供了一些在發展環境中及應用程式一般的, 能看而不能改的資訊.



Test Display

進行畫面顯示測試. 在 *Execution Control* toolbar 中也可以按 **Test Display** 執行測試. 畫面測試允許您可以在發展環境中線上修改應用程式的動態繪圖設定. 此功能並不會執行命令或 輸出入資料或是執行工作表單的項目.



Stop display test


停止畫面顯示測試. 在 *Execution Control* toolbar 中也可以按 **Stop Test Display** 停止執行測試.



Run Application

執行在 **Execution Tasks** tab 中被設定為自動執行的程序. 在 *Execution Control* toolbar 中

也可以按 **Run Application** 來執行. 當您執行 *Viewer* 時, 會將目前正在編輯的圖全部打開. 假如並沒有編輯的圖面, 程式將會打開 *Project Settings-> Runtime Desktop* 中的 *Startup screen* 所設定的圖面檔名.

 **備註:** 假如您並未設定任何執行期會自動執行的背景程式, 在執行 **Run Application** 時 *Viewer* 及 *BGTask* 會自動被啟動.

 **警告:** 在 *Execution Environment* 對話框中所設定的背景程啟動方式將會影響到真正要執行的 *Target Station*. 當您要執行 **Run Application** 之前請先確認 *Target Station* (本地或遠端)的需求.



Stop Application


停止執行中的程序. 您也可以按 *Execution Control toolbar* 按 **Stop Application** 停止執行.

 **警告:** 在 *Execution Environment* 對話框中所設定的背景程啟動方式將會影響到真正要執行的 *Target Station*. 當您要執行 **Run Application** 之前請先確認 *Target Station* (本地或遠端)的需求.



Send app to target

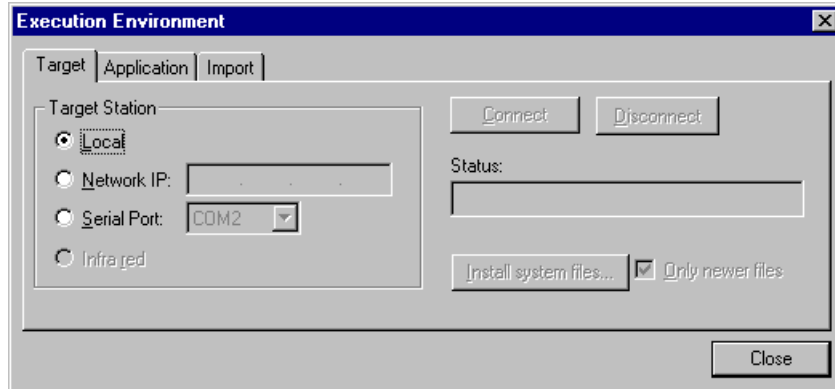
當執行時(在裝有發展版的平台連上裝有執行版的工作平台後), 您可以利用這個命令將程式傳送到遠端的 *Target Station*, 您也可以按 *Execution Control toolbar* 按 **Send app to target** 開始傳送程式.

 **警告:** 當您將程式傳送到遠端工作站時, 所更改的程序將會被即時修改. 換句話說, 一旦應用程式被下載到指定的工作站時, 即使程式還在執行先前的程序也會被自動更新, 您不用將程式停掉. 同樣的, 假如您在下載時在 *Execution Environment -> Application* 中並未勾选 *Only newer files*, 所有先前的檔案都將會被移除再進行下載.



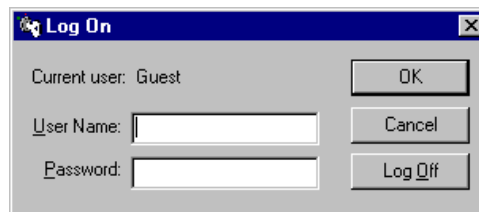
Execution Environment...

您可以直接執行也可以在 *Execution Control* toolbar 按 **Execution Environment** 來執行下載程序. 此對話框提供了管理遠端工作站的介面(下載及上傳檔案, 或是執行及停止遠端程式).



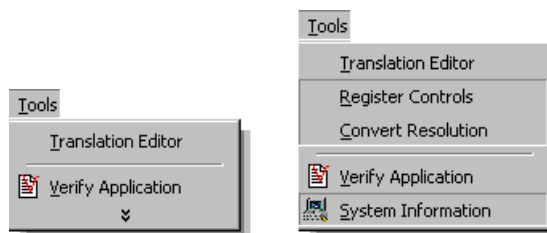
Logon...

打開 *Log On* 視窗, 其中包含了 *User Name* 及 *Password* 的文字框. 您可使用這個對話框 log on 或 log off .



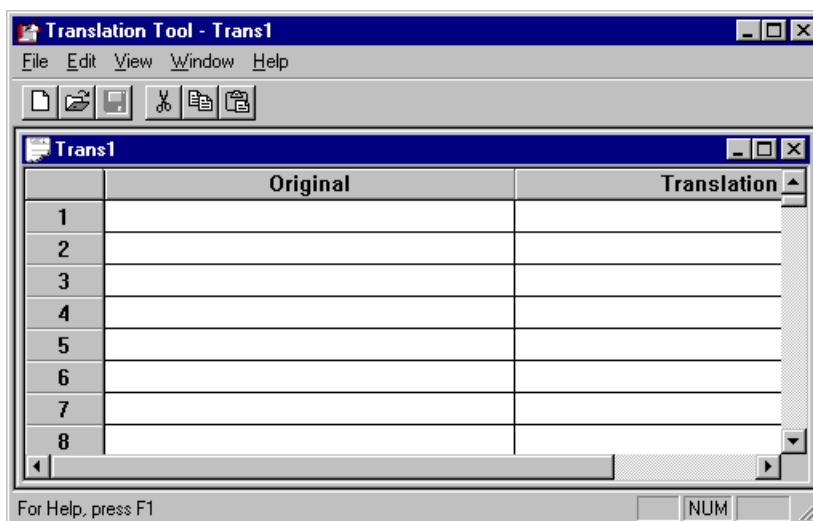
II.3.6. 工具列(Tools Menu)

此選項提供您連結到其他輔助工具.



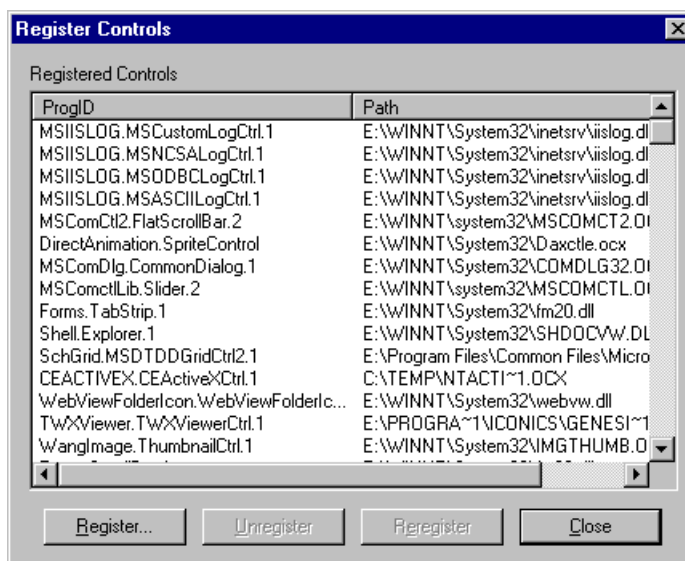
語言轉換(Translation Editor)

打開語言轉換工具的視窗可以讓您產生新的轉換工作表單。



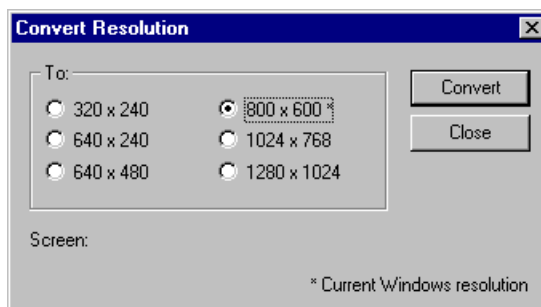
Register Controls

打開 *Register Controls* 視窗可以讓您註冊, 取消註冊, 或重新註冊 ActiveX 元件。



Convert Resolution


打開轉換視窗大小的對話框. 所有視窗大小的最後的設定值都會被存於 *application*\目錄下的 **SCREEN\BACKUP**.





Verify Application


重新編譯 *math worksheets* 及螢幕中的邏輯設定, 還有更新因在 *Project Settings* 中更改 **Web** 設定所用到的 HTML 檔.

 **提示:** 當您儲存螢幕或表單時同時也會將目前資料庫版本的代碼一起儲存. 當您執行程式時, 所執行的螢幕及表單會與資料庫作版本比對. 若版本不符則所有的敘述會重新編譯. 爲了避免此動作在執行期發生, 您最好在下載或完成程式時先執行 **Verify Application**. 若是舊版本的應用程式更新時也必須作一次 **Verify Application**.



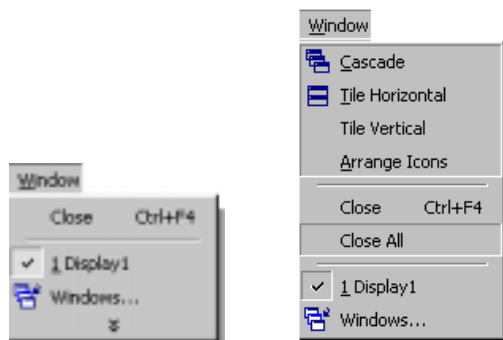
System Information


當您打開 *System Information* 視窗時此視窗包含了目前的操作系統, 目錄, 運作中的程序, 磁碟容量, 顯示器, 及記憶體等相關資訊. *Network Resources* 描述您目前的網路設定. *Applications* 可列出目前運作中的程式. *Processes* 顯示所有正在執行中的視窗程序. *Services* 列出使用中的 Windows NT/2000 服務(限 Windows NT/2000).

 **備註:** 雖然 *System Information* 是在 *Advantech Studio* 中但是所提供的資訊即是您目前所使用的工作平台或網路設定. 此功能無法提供此應用程式特定的資訊.

II.3.7. 視窗選單(Window Menu)

此選單提供命令或工具讓您在發展環境中可以管理畫面及表單。



 **備註:** 若螢幕已被設定成沒有標題列(title bars)則 Cascade, Tile Horizontal, Tile Vertical, 及 Arrange Icons 功能將無效。

Cascade

將已開啓的工作表單及顯示畫面以層疊的方式排列。

Tile Horizontal

將已開啓的工作表單及顯示畫面以垂直重疊的方式排列。

Tile Vertical

將已開啓的工作表單及顯示畫面以水平重疊的方式排列。

Arrange Icons

將工作表單及顯示畫面縮小在工作區的底部。

Close

將目前運作的螢幕或工作表單關閉。程式將提示你儲存更改過的畫面或設定。此命令與檔案(File)選單中的關閉(Close)及標提列的關閉(X)是一樣的。

Close All

將目前運作的所有螢幕或工作表單關閉。程式將一一提示你儲存更改過的畫面或設定。

Window Listing

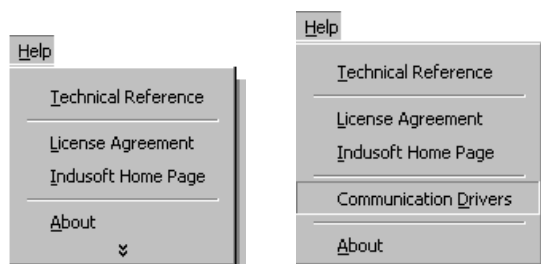
列出所有已開啓縮小在 Window 選單中。目前運作中的檔會以()來表示。您可以選擇任一個檔案來運作。

Windows...

打開可以列出所有已開啓的工作表單及顯示畫面的清單。

II.3.8. 說明(Help Menu)

提供產品及公司的相關資訊。



技術參考(Technical Reference)

開啓主要輔助說明視窗。

版權授權(License Agreement)

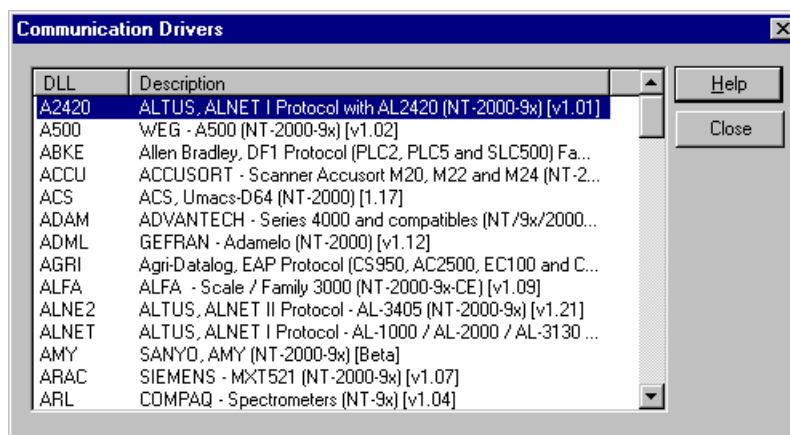
顯示 Advantech *Studio* 軟體授權說明(以 Acrobat 來閱讀)。

網站連結(Home Page)

直接連結到 Advantech 網站。

通訊驅動程式(Communication Drivers)

顯示所有通訊驅動程式的清單，您可以選擇所要使用的驅動程式然後按 **Help** 去查閱設定說明。




關於(About)

顯示版權期限，版本，產品系列名稱及版本保護碼。

II.4. Toolbars

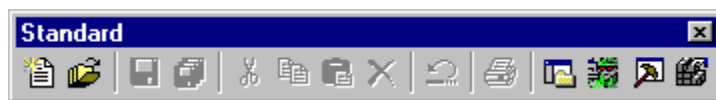
Standard, Tag Properties, Execution Control, Web, Align 與 *Distribute* 工作列在預設狀態是陳列在工作平台的上方, 就在目錄列的下方. 而 *Mode, Static objects, Active objects* 與 *Dynamic Properties* 等包括螢幕編輯工具的的工作列在預設狀態是陳列在右下方. 但 *Bitmap* 工作列在預設狀態為隱藏狀態.

所有工作列皆可變換其在螢幕上的位置.

 **提示:** 為了提醒你有關工作列的功能可查看在 *InduSoft Studio* 介面底下狀態列的左側. 有一簡短的說明將會因為滑鼠的點選而被展示出來.

II.4.1. Standard Toolbar

Standard 工作列提供許多圖示讓你可以執行一些基本的功能.

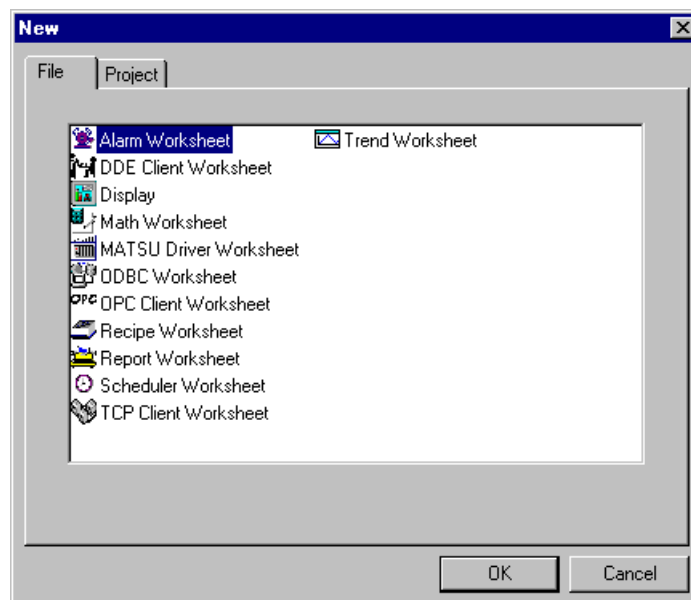


New

打開一視窗包括 **File** 與 **Project** 按鈕, 它可讓你創造出新的應用程式或專案. 你也可以藉由從 *Standard* 或在 *Insert* 目錄中選擇 **Document...** 來點選 **New** 按鈕 而開啓一新的視窗.

File 按鈕允許你創造新的 *Alarm, DDE Client, Math, ODBC, OPC Client, Recipe, Report, Scheduler, TCP Client*, 與 *Trend worksheets* 或者是一個新的 *Display screen*. 當你增加一新的 I/O 驅動程式到你的應用程式時. **Project** 按鈕允許你創造一新專案.

 **備註:** DDE Client 與 ODBC worksheets 將不會顯示在 Windows CE 的應用程式中.





Open Project...

打開 *Open* 視窗, 你可藉此使用或開啓另一 *InduSoft Studio* 應用程式. 你也可以藉由在 *File* 目錄中的 *Open Project* 裡雙擊專案名稱來打開一專案.



Save

儲存任何執行中或開啓的 *worksheets* 或 *screen*. 你可選擇在 *File* 目錄中的 *Save* .

Save 功能將只有在執行中檔案被修改時才被啓動.



Save All

儲存所有任何執行中或開啓的 *worksheets* 或 *screens*. 你也可以選擇 *File* 目錄中的 *Save All*. *Save All* 功能將只有在執行中檔案被修改時才被啓動.



Cut

移除所有選擇與儲存在剪貼簿中, 並且將取代先前在剪貼簿中已儲存的選擇. 你可以使用 *Cut* 來選擇一物體並移動它到螢幕上另一位置 或另一螢幕上. 你也可以從 *Edit* 目錄中來選擇. *Cut* 功能鍵 .



Copy

複製一選擇至剪貼簿中. *Copy* 允許你去貼上在另一螢幕上所選取的物件至所想要的位置 或是作多份複製. 你也可以在 *Edit* 目錄中來選擇 *Copy* 功能鍵.



Paste

複製剪貼簿上的內容至執行中的螢幕上. 如果這剪貼簿內包含一選擇, 它將被複製在螢幕的左上角. 你也可以在 *Edit* 目錄中來選擇 *Paste* 功能鍵.



Delete

刪除一選擇. 如果你不小心刪除了一物件, 你可以利用 *Undo* 功能鍵來恢復. 你也可以在 *Edit* 目錄中來選擇 *Delete* 功能鍵.



Undo

取消是在螢幕工作中最後一個被執行的動作. 它最多可取消 20 個先前被執行的動作. 你也可以在 *Edit* 目錄中來選擇 *Undo* 功能鍵.



Print

開啓一 *Print* 視窗. 你可以列印全部文件或特定的工作文件. 此外, 你也可以指定印表機名稱 特性欲列印的數量. 你可以藉由選擇 *File* 目錄中的 *Print* 來列印目前使用中的檔案.



Workspace

展示或移除一 *Workspace* 視窗. 當你點選這按鈕, *Workspace* 視窗開啓. *Workspace* 開關選項也可在 *View* 目錄中得到.



Database Spy

展示或移除一 *Database Spy* 視窗. 當你點選這按鈕, *Database Spy* 視窗開啓. *Database Spy* 開關選項也可在 *View* 目錄中得到.



Output

開啓 or 關閉一 *Output* 視窗. 當你點選這按鈕, *Output* 視窗開啓. *Output* 開關選項也可在 *View* 目錄中得到.



Library

開啓已設定好的物件的 *Library*. *Library* 按鈕 也可由 *View* 目錄中得到, 你也可以從 *Workspace* 視窗 中的 *Graphics* 裏開啓 *Library* 子目錄.

提示: 物件 library 提供幾個可以動態預先修改的物件, 你能夠增加到專案的 screen 上以節省應用程式發展時間. 你也能夠用新物件來將這 library 升級. 爲了達到此目的, 在 screen 圖示上(在 *Workspace* 中)按滑鼠右鍵. 之後選擇 *Send 至 library* 選項. 所有包括物件的 library 將被加入 screen 中.

II.4.2. Tag Properties 工作列

Tag Properties 工作列包括一些特殊按鈕用來尋找及進入設定點(tags), 功能(function), tag 的屬性.



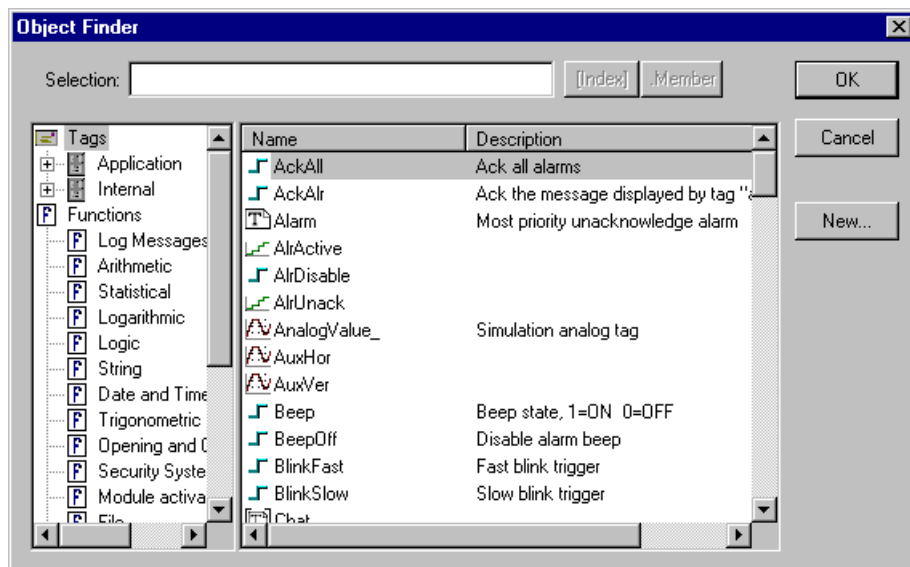
Tagname

提供你能夠鍵入一個點名稱(tag)的文字盒. 在 *Tag Properties* 工作列中的圖示 (**Cross Reference** 與 **Tag Properties**) 是用來查看此點名稱(Tag)的作用及在應用程式執行時的應用狀況.



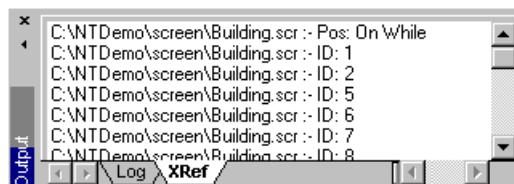
object Finder

開啓 *object Finder* 視窗, 它可列出所有功能及在應用程式中已被建立的點(Tag). 你也可以雙擊(double-click)某個點(Tag)讓點(Tag)顯示在 **Tagname** 文件盒中.



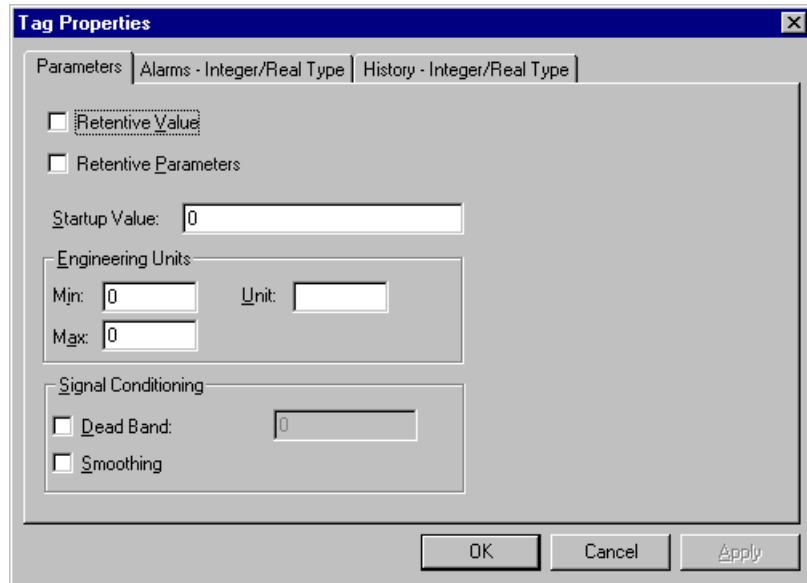
Cross Reference

從應用程式及工作表單(worksheets)中搜尋出在 **Tagname** 文件盒所設定的點(Tag)或功能. 這功能會將有用到此點(Tag)的相關檔案列出在 **XRef** 的選項中.



Tag Properties

開啓 *Tag Properties* 視窗, 你可以在此處設定點(Tag)的參數, 不同的點(Tag)型態 (Integer/Real, Boolean, 與 String)會有不同的視窗顯示.



II.4.3. Execution Control 工作列

此工作列提供你在本地或是遠端管理應用程式執行的工具。



Test Display


開始測試畫面顯示. 你也可以在 **Project** 子目錄中直接選擇 **Test Display** 執行測試. 這個功能允許你在發展環境中設定或修改應用程式時可以直接執行正在修改的圖, 此測試畫面的功能並不會執行命令鈕的設定及資料點的連結或是工作表單上的設定.


Stop display test

停止畫面測試模式. 你也可在 **Project** 目錄中直接執行 **Stop display test** .

Run Application

執行執行期程式, 此功能在 *Project Status* 視窗盒的 **Execution Tasks** 中已被設定為自動執行. 你也可以從 **Project** 目錄中直接執行 **Run Application**. 當你開始進入 *Viewer* 模式時, 它將開啓所有被編輯過的視窗畫面. 如果在發展環境中沒有任何編輯過的畫面, 此命令將開啓在 *Project Status* 視窗盒的 **Runtime Desktop** 標籤裡被設定成 *Startup screen* 的畫面檔.

 **備註:** 如果沒有任何程序被設定成在執行期自動執行時, 在你執行 **Run Application** 命令時將只會執行 *Viewer* 與 *BGTask* 二個背景程式.

 **警告:** 此命令將影響在主電腦(*Target Station*)上 *Execution Environment* 視窗中被設定的應用程式. 在執行 **Run Application** 命令之前, 請先確認在 *Execution Environment* 視窗中 *Target Station* 被設定為本地端或遠端電腦.

Stop Application

停止所有執行期程序. 你也可以從 **Project** 目錄中直接執行 **Stop Application** .

警告: 此命令將影響在主電腦(*Target Station*)上 *Execution Environment* 視窗中被設定的應用程式. 在執行 **Run Application** 命令之前, 請先確認在 *Execution Environment* 視窗中 *Target Station* 被設定為本地端或遠端電腦.



Send app to target

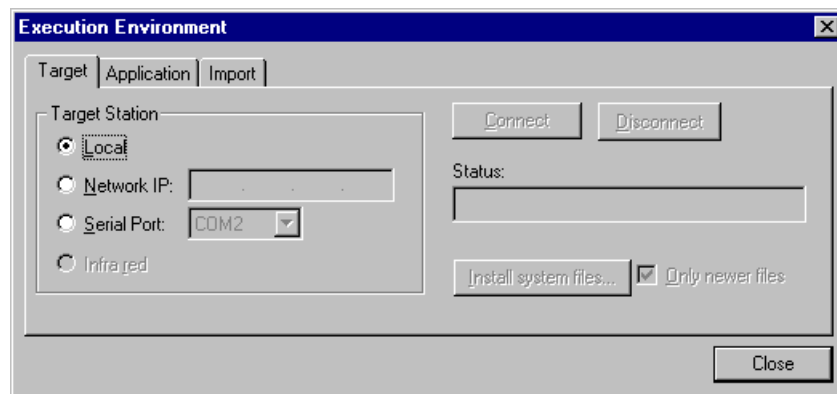
當啟動時(在發展環境中的電腦被連接至執行期的工作站), 此命令可以將應用程式下載至遠端工作站(*Target Station*), 在 *Execution Environment* 對話視窗盒中可設定遠端工作站, 你也可以從 **Project** 目錄中選擇 **Send project to target**.

警告: 當您將應用程式下載至遠端工作站時, 執行期的應用程式將立即改變. 換句話說, 一旦應用程式檔案被下載至遠端目標工作站時, 即使此應用程式正在使用當中, 它們將自動被新的應用程式所取代. 如果你在 *Execution Environment* 對話視窗盒裡 **Application** 按鈕中, 將 *Only newer files* 檢查盒取消, 所有遠端工作站應用程式目錄下的檔案將在下載新檔案之前全部被移除.



Execution Environment

打開一 *Execution Environment* 視窗會包含三個標籤選項 **Target**, **Application**, 與 **Import**. 你也可以在 **Project** 目錄中開啓選擇 **Execution Environment**. 這個對話框提供一個允許您管理遠端工作站的界面. (下載或上傳檔案, 執行或中止遠端應用程式)



II.4.4. Web Toolbar

此工作列提供打開 HTML 文件的工具。



備註: 在你使用此工作列上的工具之前必須先安裝瀏覽器 Internet Explorer v4.1 (或更新的版本)。



Back

在發展環境中開啓往前一個 URL 位址。



Forward

在發展環境中開啓往後一個 URL 位址。



Stop

取消所指定的 URL 位址檔案下載程序。



Refresh

重新從 *Address* 視窗中載入指定的 URL 位址。



Home

呼叫在瀏覽器中被設定的首頁。



Address

提供一個文字視窗, 在當中您可以輸入欲下載網頁的 URL 位址。



Go

開始下載在 *Address* 文字視窗中所指定的網頁。

II.4.5. *Align* 與 *Distribute* 工作列

此工作列提供輔助工具來編輯畫面中的物件。



Resizing 物件 s

當您用指標選擇一物件或將物件群組化時, 物件將會被幾個黑色小圈點包圍, 而這些小圈點被稱作操作點(handles)。這些操作點可用來改變物件的大小或形狀, 要改變物件大小時可將滑鼠的游標移到操作點上並持續按著滑鼠左鍵移動滑鼠到自己想要的物件大小。如果選擇中間的操作點原物件會被拉長或縮小。如果選擇四個角落中的其中一個操作點時原物件會作對稱性的放大或縮小。

備註: 所有被群組化的物件, 包括大多數 symbols 與圖庫(library)中的物件, 凡具有動態屬性的物件都有複合的物件屬性(*object Properties*)視窗。你可以從 *Selection* 物件屬性利用下拉選單來選取不同的物件屬性(*object Properties*)視窗與查看它們所有的屬性。

如果你重新定符號(symbol) 或群組化物件的尺寸, 所有在符號(symbol)或群組內的物件將因此而被重新定尺寸。

您可以將物件群組化後使用以下的工具列做物件尺寸大小的調整：



Resize width

設定所有選取物件的水平寬度依照第一個選取物件的寬度作調整。



Resize height

設定所有選取物件的垂直寬度依照第一個選取物件的寬度作調整。

提示: 你可以使用 **Resize width** 與 **Resize height** 工具將橢圓變成圓形或將長方形變成正方形. 在你使用這些工具前僅可選擇一種物件.

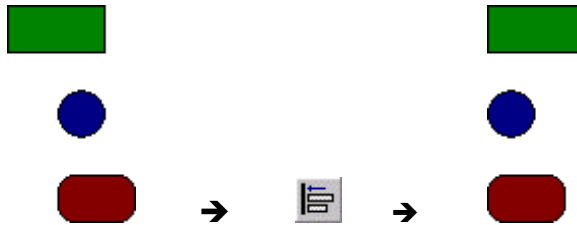
object Alignment

當你選擇一個或多個物件, 你可以使用對齊按鈕依照第一個所選取的物件位置來對齊. 第一個被選取物件的操作點(Handles)會以填滿的實心點表示. 使用工具列中的選項來作對齊.



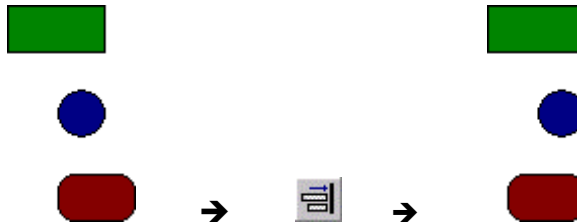
Align left

對齊所選物件的左緣至第一個選取物件的左緣.



Align right


對齊所選物件的右緣至第一個選取物件的右緣.



 **Align top**

對齊所選物件的上緣至第一個選取物件的上緣。



 **Align bottom**


對齊所選物件的下緣至第一個選取物件的下緣。



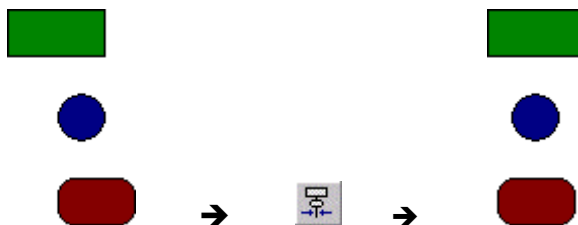
 **Center Vertically**

對齊所選物件的垂直中心至第一個選取物件的垂直中心。



 **Center Horizontally**

對齊所選物件的水平中心至第一個選取物件的水平中心。

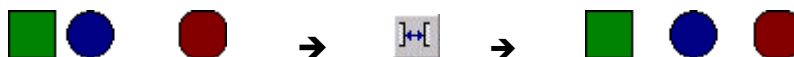


Spacing objects Equally

當你選擇一個或多個物件, 你可以使用空間按鈕來調物件之間相對的位置. 以下為相關指令的敘述:

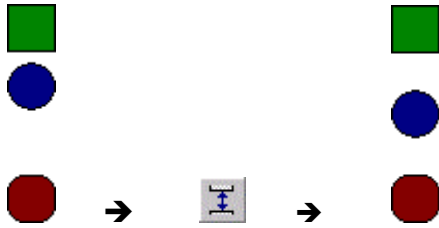
 **Evenly space horizontally**


將所選取物件間的水平距離設為相同。



 **Evenly space vertically**

將所選取物件間的垂直距離設為相同。



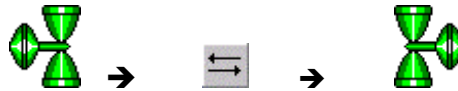
 **備註:** 此調整間距的功能可以藉由少數像素來使所有物件間距離相同的目的。

objects Orientation

物件方向按鈕將在您選擇單一物件時啟動。物件可以是已被群組化的物件，但是當你選擇多個群組化物件時此按鈕將不能使用。方向性按鈕解釋如下：

Flip Horizontally

將所選取的物件作水平方向的反轉。此展現的效果就如同物件依中心點作水平的翻轉。即是將物件作水平鏡射。



Flip Vertically

將所選取的物件作垂直方向的反轉。此展現的效果就如同物件依中心點作垂直的翻轉。即是將物件作垂直鏡射。



Rotate

將所選取物件順時鐘方向旋轉 90 度。



Changing object Layers

當你在 *Advantech Studio* 畫面中選取任一物件時此物件的辨識碼(ID)會被顯示在畫面底下的狀態列(在 *Advantech Studio* 畫面中的每一物件都會有一個辨識碼(ID))。此辨識碼(ID)決定了此物件顯示在另一物件的上層或下層。一個具有較低辨識碼(ID)的物件將會出現在較高物件的下層。辨識碼(ID)編號從零開始到螢幕上所有物件的數目為止。不會有兩個物件有相同的辨識碼(ID)。即使當你將一群物件放到後方或前方，所選取的物件群組仍然顯示著不同的層次。當您選取單一物件或物件群組時 **object Layers** 按鈕都將被啟動。

Move to back

當你壓下此紐，任何所選取的物件將被給予最低層次的辨識碼(ID)，並被顯示在其他物件的最下層。這功能也可在 popup menu 中使用。

Move to front

當你壓下此紐，任何所選取的物件將被給予最高層次的辨識碼(ID)，並被顯示在其他物件的最上層。這功能也可在 popup menu 中使用。

物件 Grouping 與 Ungrouping

無論是群組化物件或非群組化物件均可使用下列功能：



Group

將選取的項目群組化. 你可以在物件的 **popup menu** 中使用 **Group** 命令. 爲了物件的選取與處理方便群組化後的物件將被視爲單一物件. 但是你依然可以在 **object Properties** 視窗中選取群組的每一部份.



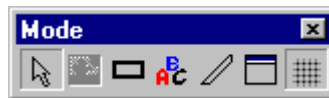
Ungroup

解除群組, 你選定群組後按滑鼠右鍵執行 **Ungroup** 命令. 群組化的物件包括許多獨立的物件群組, 所以在作解除群組時有時需要重複地選取群組解除群組來完整地解除一個複雜的物件群組.

備註: 所有被群組化的物件, 包括大多數 **symbols** 與圖庫(**library**)中的物件, 凡具有動態屬性的物件都有複合的物件屬性(**object Properties**)視窗. 你可以從 **Selection** 物件屬性利用下拉選單來選取不同的物件屬性(**object Properties**)視窗與查看它們所有的屬性.

II.4.6. Mode Toolbar

此工作列提供了螢幕編輯的一般使用工具:



Selection

顯示指標以便您選擇或移動物件.



Bitmap Editor

每一個畫面會有兩個基本的編輯層: 物件(**objects**)層與背景圖(**Background Picture**)層. **Bitmap Editor** 按鈕讓你在這兩層間互相轉換. 當按下編輯背景圖(**Background Picture**)時, **Bitmap** 工作列會自動顯示.

備註: 當 **Screen Attributes** 視窗內 **Enable Background** (僅使用 **BMP** 型態) 選項沒有被點選時, **Bitmap Editor** 按鈕會保持灰階(禁能).



Fill Color

設定可填滿物件的預設顏色, 你可以使用此命令在 **Closed Polygon**, **Ellipse**, **Rounded Rectangle** 與 **Rectangle** 物件.

提示: 你可以選擇一些物件一起改變其顏色以減少發展時間(任何以上曾提到的型態)



Fonts


設定所選取 **Text** 物件的字形與顏色, 你也可以使用這些設定作爲新物件的預設值.

提示: 你可以選擇一些 **Text** 物件利用 **Fonts** 按鈕一起改變其顏色及字形以減少發展時間. 如果你將 **Text** 物件群組化時, 這命令將不會正常運作.



Line Color

設定所選取物件線條的顏色, 你也可以使用這些設定作爲新物件的預設值. 你可以使用此命令在 **Open Polygon**, **Closed Polygon**, **Line**, **Ellipse**, **Rounded Rectangle**, 與 **Rectangle** 物件.

 **提示:** 你可以選擇 一些物件(任何以上曾提到的型態) 一起改變其線條顏色, 使用 **Line Color** 按鈕以減少發展時間.

Background color

設定螢幕背景顏色. 當你在 *Screen Attributes* 視窗內點選 **Enable Background** 選項時, 此命令將被自動取消.

Grid

在畫面編輯器中顯示或隱藏格點.

 **提示:** 你可以利用 *Grid* 視窗設定格點的預設值. 您可以在螢幕上按滑鼠右鍵在 popup menu 中選擇 **Grid Settings** 選項, 就可以打開 *Grid* 視窗.

II.4.7. *Bitmap* 工作列

此工作列提供在 *Bitmap* 編輯器中選取主要工具, 此工作列只有在 *Background Picture* 層被啟動時才會發生作用.



Select Area

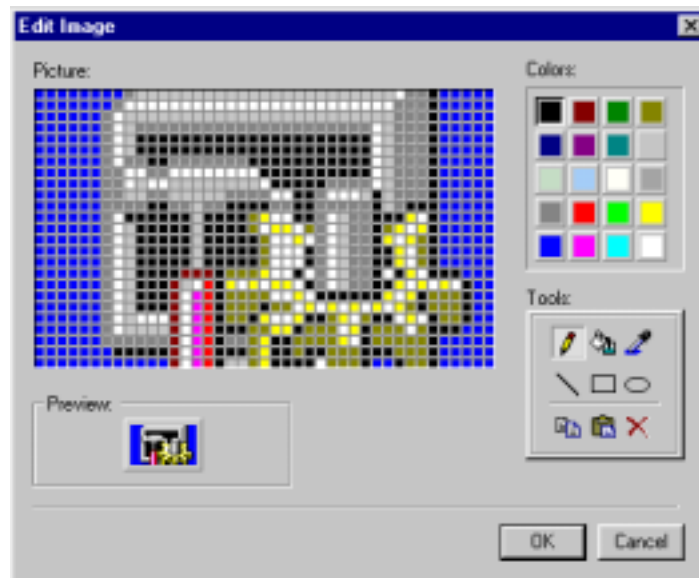
使你可以選擇 *bitmap* 編輯的區域.

Flood Fill

使你可以在畫面將在 **Fill Color** 按鈕所選擇的顏色填滿周圍區域.

Pixel Editing

打開此一放大的視窗可以讓您用像素詳細地描繪圖面.



Erase Area

以之前在 **Fill Color** 按鈕中所選擇的顏色填滿所選取的區域.

Change colors

在所選擇的區域內填滿顏色作為可通透的顏色. 在使用此命令之前, 你應該以 **Fill Color**

按鈕來選擇想要填滿的顏色. 利用 **Select Transparent Color** 按鈕選擇可穿透的顏色, 利用 **Select Area** 按鈕選擇要被填滿的區域.



Select Transparent Color

設定可穿透的顏色, 它是被用來作為 **Change Colors** 按鈕的參照顏色.



Toggle Transparent Color

使用 **Select Transparent Color** 按鈕所選取產生的顏色, 它將變成 **bitmap** 編輯器中 **bitmap** 的穿透顏色.

提示: 你可以使用 **Copy (Ctrl+C)** 與 **Paste (Ctrl+V)** 命令來交換 **Advantech Studio** **bitmap** 編輯器與任何其他 **bitmap** 編輯器(如 **Paint Brush**)之間的 **bitmap** 圖像.

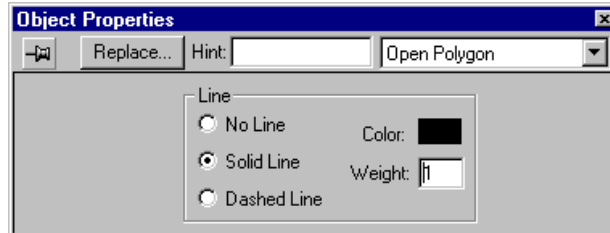
II.4.8. *Static objects* 工作列

Static objects 工作列使你可以畫出多邊形, 長方形, 直線與其他物件.



Open Polygon

以前景顏色來繪製一多邊形. 當您要在繪製區域繪製一多邊形時, 按下滑鼠左鍵來設定多邊形開始的點, 移動游標後再次點選按鈕來設定次一個頂點. 重複此程序直到你得到想要的多邊形. 最後雙擊(double-click)來完成此程序. 若要查看物件的屬性, 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



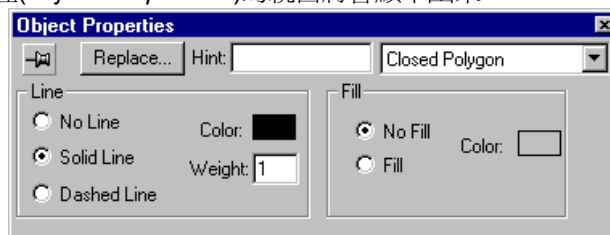
物件屬性(*object Properties*)視窗內的選項顯示如下:

- 您可點選 *No Line*, *Solid Line*, 或 *Dashed Line* 來選擇邊界的形式.
- 您可點選代表 *Color* 的長方形來打開設定顏色的視窗, 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕來選擇邊界的顏色.
- 您可以依您的需要在 *Weight* 欄位中輸入直線的寬度.



Closed Polygon

以前景顏色來繪製一封閉的多邊形, 當您要在繪製區域繪製一封閉多邊形時, 您可以點選左按鈕來設定多邊形開始的點, 然後繼續點選左鍵直到你得到想要的多邊形. 標示到最後一個點時, 雙擊(double-click)滑鼠左鍵或點選滑鼠右鍵. 雙擊(double-click)此物件, 設定物件屬性(*object Properties*)的視窗將會顯示出來.

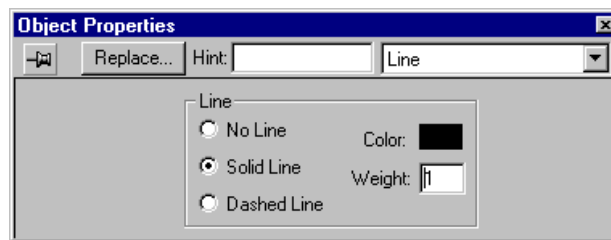


object Properties 視窗內的選項顯示如下：

- 您可點選 *No Line*, *Solid Line*, 或 *Dashed Line* 來選擇邊界的形式。
- 您可點選代表 *Color* 的長方形來打開設定顏色的視窗, 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕來選擇邊界的顏色。
- 您可以依您的需要在 *Weight* 欄位中輸入邊界的寬度。
- 您可以選擇 *No Fill* 或 *Fill* 來選擇要不要用顏色填滿或不填滿多邊形, 點選代表 *Color* 的長方形來打開設定顏色的視窗, 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕。

Line

繪製一直角直線。您可在想要的位置點選滑鼠左鍵, 將之拖曳至適當的尺寸, 再次點選一次滑鼠左鍵來定位此物件, 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來。

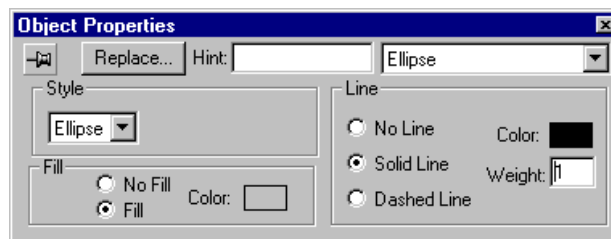


object Properties 視窗內的選項顯示如下：

- 您可點選 *No Line*, *Solid Line*, 或 *Dashed Line* 來選擇直線的形式。
- 您可點選代表 *Color* 的長方形來打開設定顏色的視窗, 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕來選擇直線的顏色。
- 您可以依您的需要在 *Weight* 欄位中輸入直線的寬度。

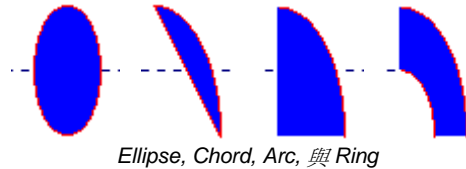
Ellipse

您可以繪製出中空或填滿顏色的橢圓形, 弦, 弧與環。在工作列中點選此橢圓的圖示, 在繪製區域內拖曳滑鼠來繪出橢圓的外形, 使用 *object Properties* 視窗來改變其外型成弦, 弧或環。雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來。



object Properties 視窗內的選項顯示如下：

在拉下選單內選擇物件的外形種類。您可選擇橢圓形, 弦, 弧或環狀。如果您繪製一個弦, 弧或環, 在 *Style* 區域內下拉選單會顯示出可所選圖形。您可以在 *Style* 區域內選擇 *Left-Bottom*, *Left-Top*, *Right-Bottom*, 或 *Right-Top* 來選擇所繪製的象限。例如, 如果你想要表現半圓形的管線, 你可以選繪製二個 *ring* 物件, 然後一個選 *Left-Bottom* 另一個選 *Right-Bottom*, 最後將之組合在一起, 成為半圓形的管線。



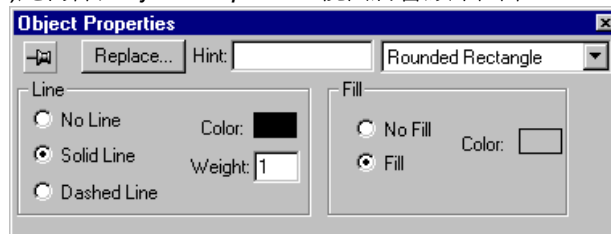
提示: *Ring* 型態很適合用於繪製配管時。

- 您可點選 *No Line*, *Solid Line*, 或 *Dashed Line* 來選擇邊界的形式。
- 您可點選代表 *Color* 的長方形來打開設定顏色的視窗, 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕來選擇邊界的顏色。
- 您可以依您的需要在 *Weight* 欄位中輸入直線的寬度。



Rounded Rectangle

您可以繪製出中空或填滿顏色的圓頂的長方形. 在工作列中點選此圓頂長方形的圖示, 在繪製區域內拖曳滑鼠來繪出長方形的外形(在 WinCE 應用程式下不支援此繪圖工具). 圓頂點長方形在左下角有一特別的追蹤點可使您修改弧的角度. 一但繪製好後, 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來。



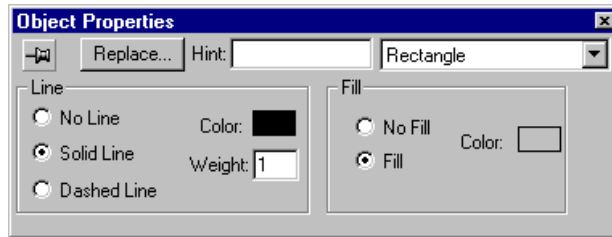
object Properties 視窗內的選項顯示如下;

- 您可點選 *No Line*, *Solid Line*, 或 *Dashed Line* 來選擇邊界的形式。
- 您可點選代表 *Color* 的長方形來打開設定顏色的視窗, 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕來選擇邊界的顏色。
- 您可以依您的需要在 *Weight* 欄位中輸入直線的寬度。
- 您可以選擇 *No Fill* 或 *Fill* 來選擇要不要用顏色填滿或不填滿多邊形, 點選代表 *Color* 的長方形來打開設定顏色的視窗, 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕。



Rectangle

您可以繪製出中空或填滿顏色的長方形. 在工作列中點選此長方形的圖示, 在繪製區域內拖曳滑鼠來繪出長方形的外形. 點選並拖曳長方形的邊緣可使您修長方形的外形. 一但繪製好後, 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來。



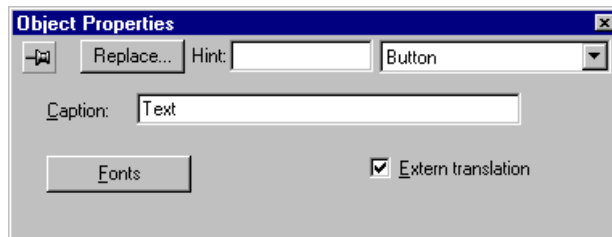
object Properties 視窗內的選項將被顯示如下：

- 您可點選 *No Line*, *Solid Line*, 或 *Dashed Line* 來選擇邊界的形式。
- 您可點選代表 *Color* 的長方形來打開設定顏色的視窗，雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕來選擇邊界的顏色。
- 您可以依您的需要在 *Weight* 欄位中輸入直線的寬度。
- 您可以選擇 *No Fill* 或 *Fill* 來選擇要不要用顏色填滿或不填滿長方形，點選代表 *Color* 的長方形來打開設定顏色的視窗，雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕。



button

您可以繪製出自訂尺寸的按鈕。在工作列中點選此按鈕圖示，在繪製區域內拖曳滑鼠來繪出按鈕的外形，拖曳滑鼠來來調整按鈕的形狀大小。雙擊(double-click)此物件，*object Properties* 視窗將會顯示出來。

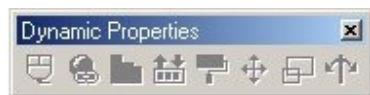


object Properties 視窗內的選項顯示如下：

- 您可以在 *Caption* 內鍵入按鈕的標題。
- 您可以點選 **Fonts** 按鈕並使用 *Font* 視窗來定義按鈕的字型。
- 您可選擇 *Extern translation* 選項，使按鈕上的標題可以利用外部轉換檔案進行語言轉換。

II.4.9. *Dynamic Properties* 工作列

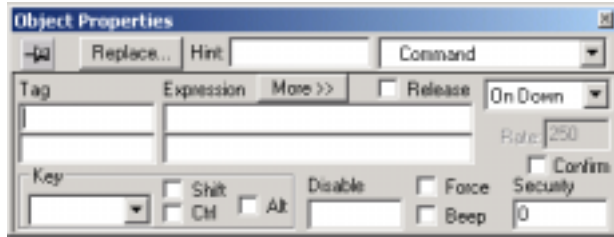
您可以將單一物件或一物件群組加上動態化，您可以藉此設定使物件在執行期可根據資料點(Tag)的數值作動態變化。部份動態化功能也可設定成可以執行命令或輸入數值(設定點)至資料點(Tag)。



Command

增加 *command* 特性到你選取的物件或物件群組。在執行期，如果你按下此物件或按下定

義鍵, 將會執行指令. 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



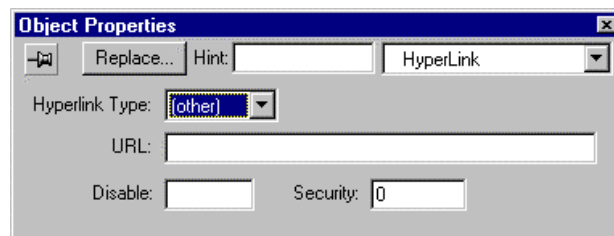
object Properties 視窗內的選項顯示如下:

- 每一事件最多可以設定十二行 (On Down, On While 與 On Up). 您可按下 **More>>** 查看額外的增加的資料點(*Tag*)與 指令(*Expression*). 運算式與函數功能可以在 **Expression** 欄中設定. 每一個運算式的結果(回傳值)會將值寫入資料點(*Tag*).
- **Release Check**-對話盒 - 當其被點選時, 不論您是否已將指標拖曳到物件外均會執行指令, 是否釋放滑鼠是無所謂的.
- **Confirm Check**-對話盒 -當其被點選時, 在執行期時要執行指令前程式會出現確認對話框.
- **Events Drop-List** - 你可以依下列事件來執行命令:
 - **On Down** - 當滑鼠被壓下執行指令敘述.
 - **On Up** -當滑鼠被壓下後釋放時才執行指令敘述.
 - **On While** -當滑鼠處在被壓下狀態時, 才執行指令敘述.
- **Rate Field** - 定義在毫秒內 **On While** 事件的速率.
- 爲了將鍵盤與物件或物件群組加以連結, 您可任選一個在 **Key drop-down list** 中的按鈕. 將它定爲當你壓下此按鈕時, 即執行指定的命令. 如果您要使用者選擇伴隨 **drop-down list** 所指定的 **Shift** 鍵. 選擇 **Shift** 對話盒. 或者選擇 **Ctrl** 或 **Alt** 對話盒, 讓使用者去點選所指定的按鍵.
- **Disable Field** -當在此欄中所鍵入的資料點(*Tag*)值大於時, 將會不執行此命令.
- 選擇 **Beep** 檢查盒時當命令開始執行時可聽到嗶聲.
- **Security** 欄 - 指定此物件的安全層次, 在 **Security** 功能中可定義所有的安全層次. 當登入的使用者沒有相當的安全層次時, 將無法執行此命令.




Hyperlink

當您增加 hyperlink 特性到你選取的物件或物件群組時, 在執行期, 如果你點選此物件或被定義的按鍵被壓下, 將執行與瀏覽器的鏈結, 指定的 url 將被載入. 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



object Properties 視窗內的選項顯示如下;

- 以下拉式選單選擇將被載入 *Hyperlink Type* (*HTTP*, *HTTPS*等)的 *URL*. 當 url 被載入時, 此協定將自動被使用. 在 *URL* 欄位中, 鍵入預載入 *URL* 的位址, 如 <http://indusoft.com.br>.

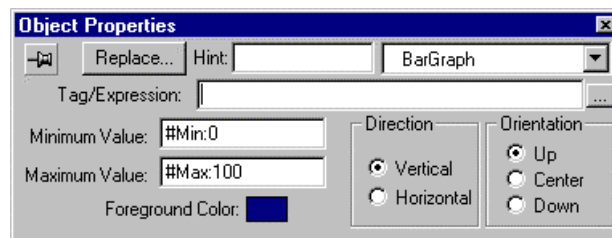
 **提示:** 此協定的型態(HTTP,HTTPS,等)並不需要被鍵入在 URL 欄位中. 一旦選定了 **Hyperlink Type**, 他的字首將自動被加入 url 位置裡.

- **Disable Field** –當在此欄中所鍵入的資料點(Tag)值大於 0 時, 將會不執行 hyperlink 命令.
- **Security Field** -指定此物件的安全層次, 在 **Security** 功能中可定義所有的安全層次.當登入的使用者沒有相當的安全層次時, 將無法執行此命令.




Bargraph

增加 **graph** 特性到你所選取的物件. 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



object Properties 視窗內的選項顯示如下:

- **Tag/Expression Field** – 資料點(Tag)或運算式用來使 bar graph 將值以圖形顯示.
- **Minimum Value Field** - 定義 bar 的高度(如果以垂直表現)或寬度(如果以水平表現)的最小值. 您可鍵入一數值或資料點(Tag)來定義最小值.
- **Maximum Value Field** – 定義 bar 的高度(如果以垂直表現)或寬度(如果以水平表現)的最大值. 您可鍵入一數值或資料點(Tag)來定義最大值. 如果鍵入的資料點(Tag)並不存在, 將會顯示詢問您是否要新增此資料點(Tag)的視窗.

 **提示:** 在您鍵入資料點(Tag)或數值的欄位上, 您也可以鍵入一個常數值. 此常數(用#字元來作提示符號)將等於一個數值, 此替代會出現在 **Tag Replace** 視窗中. 在文件化與建立常用物件時這種方式相當實用. 舉例來說: **#Name:100**. 在:之後的數值為常數, 名稱僅為幫助記憶用的並不會加入資料庫中.

- **Foreground Color** – 可讓您選擇將填滿物件的顏色, 點選 **Foreground Color** 旁長方形來顯示選擇顏色的視窗. 雙擊(double-click)想要的顏色或按顏色後按 **OK** 按鈕.
- **Direction Group** 對話盒 – 決定圖表的顯示方為水平或垂直. 您可點選想要的選項來決定填滿的方式.
- **Orientation Group** 對話盒 – 決定所繪製圖表以最大與最小值來決定方向. 您可點選想要的選項來繪製 **Up**, **Center** 或 **Down** 方式表示.

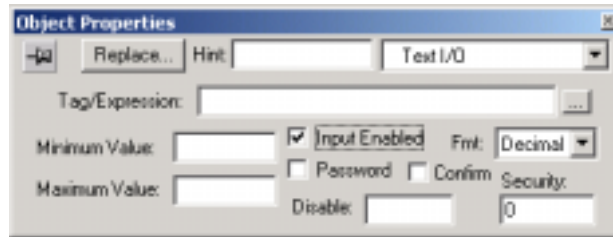


Text I/O

連結動態輸入或輸出點到目前選取的 **Text** 物件上. 當執行應用程式時, 您可使用鍵盤或觸控螢幕輸入資料點(Tag)的值以便即時顯示其值.

 **備註:** 此動態特性僅可以用在以#字元組合而成的 text 物件時, 每個“#”字號代表一位數

雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



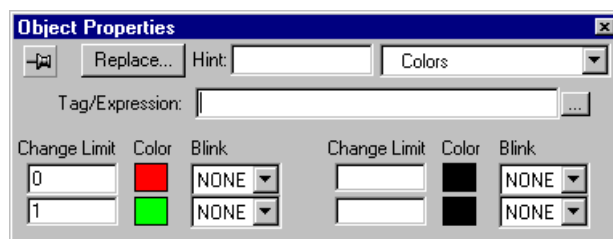
object Properties 視窗內的選項顯示如下:

- **Tag/Expression field** – 保持資料點(Tag)的值作輸入或輸出操作或僅用運算結果作為輸出.
- **Input Enable Check-對話盒** –允許使用者輸入值, 否則, 此動態特性將只資料的輸出.
- **Confirm Check-對話盒** –當其被點選時, 在執行期時要執行指令前程式會出現確認對話框.
- **Minimum Value Field** -定義資料點(Tag)的最小值與物件作結合. 使用者將不允許輸入比此設定值還要小的值.
- **Maximum Value Field** -定義資料點(Tag)的最大值與物件作結合. 使用者將不允許輸入比此設定值還要大的值.
- **Password Check-對話盒** – 若使用者無權限時使用者將只看 text 被 * 所取代.
- **Fmt Drop-list** – 從此清單你可以決定輸出入欄位的格式.
- **Disable Field** –當在這欄位中所鍵入的標籤值大於 0 時.所輸入的資料將無效.
- **Security Field** -指定此物件資料輸入的安全層次, 在 **Security** 功能中可定義此值.



Colors

增加改變物件顏色的特性到所選的物件. 此物件將依你將監控的資料點(Tag)的變化用顏色填滿所連結的物件, 此動態特性最多允許四個設定值來改變顏色. 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



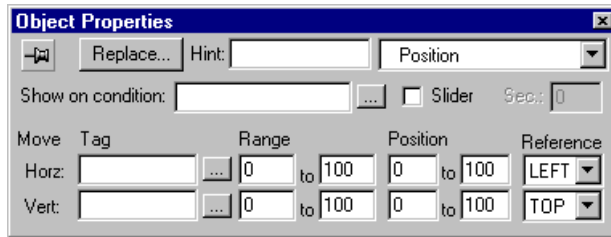
object Properties 視窗內的選項顯示如下:

- **Tag/Expression Field** –將資料點(Tag)的值或運算結果與物件作連結. 物件將依設定的值作顏色的改變.
- **Change Limit Field** –改變顏色的設定值, 他必須是一個數值或是資料點(Tag).
- **Color Rectangle** – 定義設定值的顏色. 點選長方形後選擇顏色的視窗將會出現. 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕.
- **Blink Drop-list** - 定義是否要閃爍, 如果要可設快或慢.



Position

您可以使物件根據你所指定的位置顯示在畫面上. 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



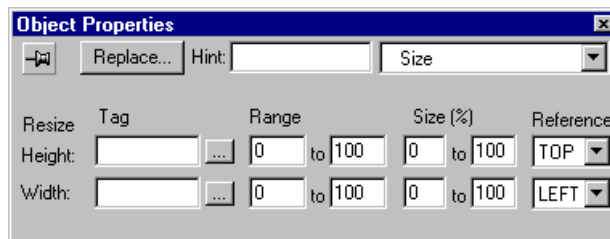
object Properties 視窗內的選項顯示如下：

- 如果你希望此物件為可見時，在 **Show on condition** 內輸入一大於零的數字或運算式。如果你輸入零或負數，此物件將被隱藏。如果你不指定數值，此物件將被預設為可見。
- 如果你希望此物件可以像滑動物一樣作動作時，選擇 **Slider** 對話盒。即表示您可以拖曳此物件將相關的值連結到資料點(Tag)。
- **Sec.:Field** –當使用 **slider** 選項時用此欄位設定物件的安全性，您可以定義操作者使用 **slider** 選項的安全層次。
- **Tag Field** – 資料點(Tag)會與此物作連結，使物件可以水平地或垂直地在螢幕上移動。
- **Range Field** – 定義資料點(Tag)值的最大與最小極限，使物件可根據建立的範圍，水平地或垂直地在螢幕上移動。
- **Position Field** -定義位置與像素的改變量，此物件可以根據建立好的條件範圍在螢幕上移動。此值在右邊的欄位可為負值。
- **Reference Drop-list** - 定義物件在螢幕上移動時的參考點，此選項只有在物件移動時，尺寸被改變了才需要。
 - Left** – 以物件的左緣為起點
 - Right** – 以物件的右緣為起點
 - Center** – 以物件的中心點為起點
 - Top** – 以物件的上緣為起點
 - Bottom** – 以物件物件的下緣為起點。



Resize

使你可以重新改變物件的大小或代表物的尺寸。雙擊(double-click)此物件，*object Properties* 視窗將會顯示出來。

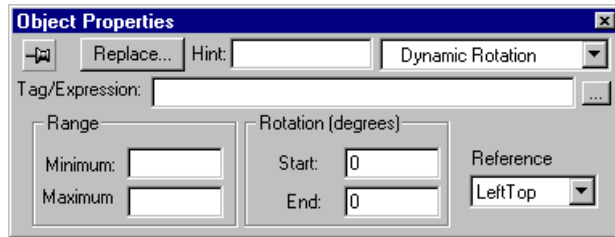


- **Tag** 選項 – 可依所設定的資料點值來增加或減少物件水平或垂直的尺寸。
- **Range** 選項 – 定義資料點(Tag)值的上下限做為增加或減少物件的尺寸
- **Size** 選項 –系統將依此設定的百分率來改變指定物件的尺寸。
- **Reference** 下拉選單 – 設定物件將如何增加它的垂直和水平的尺寸。
 - Left** – 自左邊開始
 - Righ** – 自右邊開始
 - Center** - 自中間開始向垂直及水平方向拓展
 - Top** - 自上方開始
 - Bottom** - 自底部開始



旋轉屬性(Rotation Property)

允許將線段作旋轉的設定，雙擊(double-click)此物件，*object Properties* 視窗將會顯示出來。



- **Tag/Expression** 選項- 資料點(Tag)會與此物件作連結依所讀取的值顯現在螢幕上.
- **Range Group Box** -定義資料點(Tag)的上下極限, 做為物件在螢幕內移動的根據.
- **Rotation (degrees) Group Box** -定義物件動態旋轉的開始和結束角度. 在 WinNT 線段能夠達到 360 度轉動. 但是, 在 WinCE 則無法超過 90 度.
- **Reference Drop-list**-定義物件在螢幕內轉動的參考點:
 - Left Top** -以物體的左上角.
 - Right Bottom** -以物體的右下角.
 - Center** - 以物體的中心點.

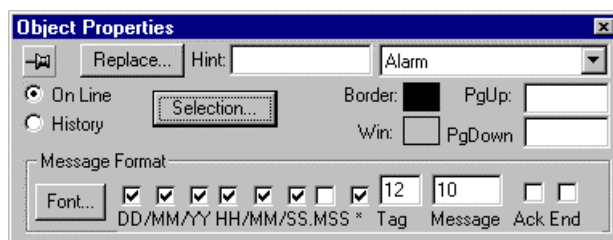
II.4.10. 動態式物件工具箱(Active Objects Toolbar)

相較於靜態的物件(Static Object), 這些物件包含了一些特定的動態顯示且需要更多的參數設定.



Alarm

這個選項可讓您選擇螢幕中的一塊區域做為顯示警報訊息. 點選物件編輯工具箱中的圖示, 然後, 將滑鼠移至繪圖範圍裡. 按滑鼠左鍵並拖拉滑鼠以調整形狀. 雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.

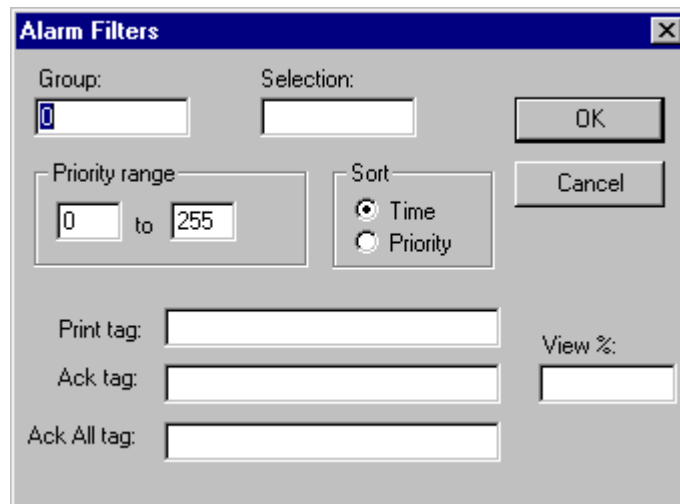


- **On Line** Radio Button - 設定此物件顯示即時警報訊息
- **History** Radio Button - 設定此物件從歷史檔案中顯示警報訊息


⚠注意：在程序(Tasks)標籤的警報工作單, 必須先設定 **Save to Disk** 的選項來儲存警報訊息. 才可以規劃成從歷史檔案顯示警報訊息.


- **Border** Rectangle -定義這個警報訊息邊框的顏色. 點選Color旁的矩形可顯示設定顏色的視窗. 雙擊(double-click)想要的顏色或點選想要的顏色之後按OK 按鈕.

- **Win Rectangle** -定義這個警報訊息視窗背景的颜色. 點選 **Color** 旁的矩形可顯示設定颜色的視窗. 雙擊(double-click)想要的颜色或點選想要的颜色之後按 **OK** 按钮.
- **PgUp Field** - 向上卷動警報的訊息列. 此設定會與一個資料點(Tag)名稱相連結.觸發一次將使表列上卷一頁.
- **PgDown Field** - 向下卷動警報的訊息列. 此設定會與一個資料點(Tag)名稱相連結.觸發一次將使表列下卷一頁.
- **Message Format Group Box** -定義警報訊息的格式. 設定警報是否包括日期, 時間, 名稱, 點名稱及訊息.
 - **Font Button** -開啓定義字型的視窗, 您可以設定字型的樣式, 尺寸, 颜色, 和字體型式.
 - **DD,MM,YY Check-boxes** - 若點選時則表示要在警報列中顯示日期
 - * 如果勾选時, 在警報日期/時間和警報訊息之間顯示星號.
 - **Tag Field** -輸入資料點(Tag)字元所佔的長度.
 - **Message Field** - 輸入一個訊息字元所佔的長度.
 - **Ack Check-box** -顯示警報確認的時間.
 - **End Check-box** -顯示警報回復正常的時間.
- **Selection ... Button** -打開過濾警報的視窗, 在其中您能夠在列表的警報訊息中過濾警報訊息:



- **Group Field** -允許輸入所要顯示的警報群組. 如果欄位中的值是 0, 所有警報都會被顯示. 反之, 若設定為其他的值則會顯示所設定的警報群組.
- **Selection Field** -設定過濾條件使警報區僅顯示符合條件的警報, 此條件(為一字串)會與在警報工作單上的過濾欄位中所設定的字串作比較, 符合的警報才會顯示.

 備註: 字串必須是在警報工作單中的過濾欄位中所設定的字串.

 提示: 你也可以在括弧{ }之間輸入一個字串型態的資料點(Tag), 使您可以在執行期修改過濾的字串內容.

- **Priority Range Group Box** -過濾器警報可根據在警報工作單所設定的優先權來顯示引起的警報. 它根據優先權將警報分組. 例如, 如果警報被指定的優先權為 1 到 5 但是在顯示警報的優先權範圍從 0 到 4, 則將只顯示優先權被設為 1 到 4 的警報, 優先權 5 的警報將不會顯示.
- **Sort Group Box** -可用選擇鈕來指定用時間或者優先權來進行排序. 警報將依據收到警報的時間或者警報的優先權顯示.
- **Print Tag Field** -當您在此欄位中輸入一個資料點(Tag)時, 當這個資料點(Tag)有變化時將會列印所過濾的所有警報.

- **Ack Tag Field** -當您在此欄位中輸入一個資料點(Tag)時, 當這個資料點(Tag)有變化時目前過濾的警報(在警報區域列在最上頭的警報)將會被確認。

 **提示:** 您也能夠用這個內部點 **AckAlr** 來確認應用程式中最後發生的警報。

- **Ack All Tag Field** -當您在此欄位中輸入一個資料點(Tag)時, 當這個資料點(Tag)有變化時, 所有過濾的有效警報將被確認。

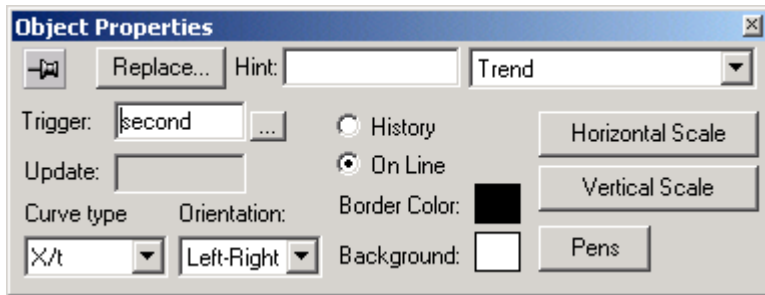
 **提示:** 您也能夠用這個內部點 **AckAll** 來確認應用程式中最後發生的警報。

- **View % Field** -當您在此欄位中輸入一個資料點(Tag)(字串或者整數), 在警報區域的警報訊息會依此值 0 和 100 之間(以百分比)與目前警報訊息位置作相應。



Trend

您可以在螢幕上選擇一個範圍作為趨勢圖的顯示。規劃欄位中可設定所顯示的區間, 顯示的值和圖表的格式。在趨勢物件中同時僅可展現達八個曲線。點入位於工具箱的按鈕, 然後, 把滑鼠放在區域範圍內。按下滑鼠左鍵並拖拉這滑鼠以調整形狀。雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來。

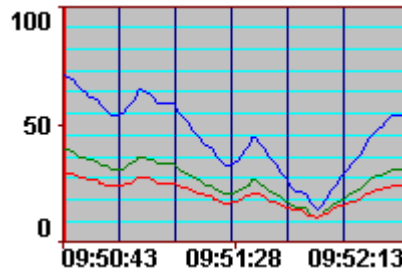


- **On Line Radio Button** -顯示即時趨勢曲線。

 **警告:** 使用歷史曲線圖形時, 您必須透過程序(**Tasks**)標籤中的趨勢圖(**Trend**)設定, 在歷史群組中新增資料點(Tag)並選定 **Save on Tag Change** 或 **Save on Trigger** 選項。這些資料點(Tag)將會在硬碟上儲存取樣點。

- **Trigger Field** -定義一個變數來重繪趨勢曲線。只要一有變化, 就會畫曲線。這個欄位在即趨勢是一定要輸入的, 但在歷史趨勢中是無效的。
- **Update Field** -當您在此欄位中輸入一個資料點(Tag), *Studio* 將參照此資料點(Tag)來產生一個趨勢圖。它僅能與 Crisp 趨勢圖一起使用。
- **Curve Type Drop-list** -定義趨勢圖表的曲線類型。
 - **X/t** -根據時間以資料點(Tag)的值來畫出圖形。
 - **X-Y** -根據資料點(Tag) 的值作出曲線圖。
 - **Crisp** -特定的格式用於 VAX 工作站的介面。
- **Orientation Drop-list** -定義(這些)趨勢圖繪圖的方向:
 - **Left-Right** -從右往左畫。
 - **Right-Left** -從左往右畫。
- **Border Color Rectangle** -定義趨勢圖表選擇的區域(範圍)外框的顏色。點選 **Color** 旁的矩形可顯示設定顏色的視窗。雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕。
- **Background Color Rectangle** -定義趨勢圖表選擇的區域(範圍)背景的颜色。點選 **Color** 旁的矩形可顯示設定顏色的視窗。雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕。

- **Horizontal Scale** Button –您可以開啓趨勢圖的水平設定視窗.
- **Vertical Scale** Button –您可以開啓趨勢圖的垂直設定視窗.
- **Pens** Button – 開啓一個視窗, 它可用於配置畫趨勢圖之筆.



- **HORIZONTAL SCALE WINDOW- Curve Type X/t (ON-LINE)**
在物件屬性趨勢視窗中已經選擇了 **On-line** 和 **Graph X/t** 選項

- **Scale Visible** Check-box – 設定趨勢圖表尺度為可見.
- **Number of Labels** Field – 設定用於趨勢圖表分割的標籤數.
 - **Divisions** Field – 設定趨勢圖區格線(行)的數目.
 - **Color** Rectangle - 定義趨勢圖區格線的顏色. 點選 **Color** 旁的矩形可顯示設定顏色的視窗. 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕. 如果您沒有在欄位中並沒有鍵入資料, 則不會顯示任何格線.
- **Scale String Format** Group Box - 規定用於趨勢圖的水平尺度標示字串的格式.
 - **Fonts** Button - 開啓定義字型的視窗, 您可以設定水平標籤字型的樣式, 尺寸, 顏色, 和字體型式.
 - **DD/MM/YY-HH:MM:SS** Check-boxes – 您可以設定標籤上的字串是否有小時, 分鐘和秒
- **Vertical Cursor** Group Box
 - **Enable** Check-box - 在趨勢視窗中可使用垂直的游標軸.
 - **Color** Rectangle – 可規畫垂直游標軸的顏色.
 - **Position (0-100)** Field - 當使用這個垂直游標時, 必須在此欄位中鍵入一個資料點(Tag)用來更新游標軸的位置.
 - **Date/Time Output** Field - 選擇一個資料點(Tag)用來接收垂直游標軸目前所在的時間點.
- **Horizontal Axis Duration** Field – 定義趨勢圖所顯示的橫座標軸的時間範圍. 這個欄位中可能是一個資料點(Tag)或者一個數字(值). 例如: 如果在欄位中鍵入時間= 0.03333 (即為 2 分鐘).

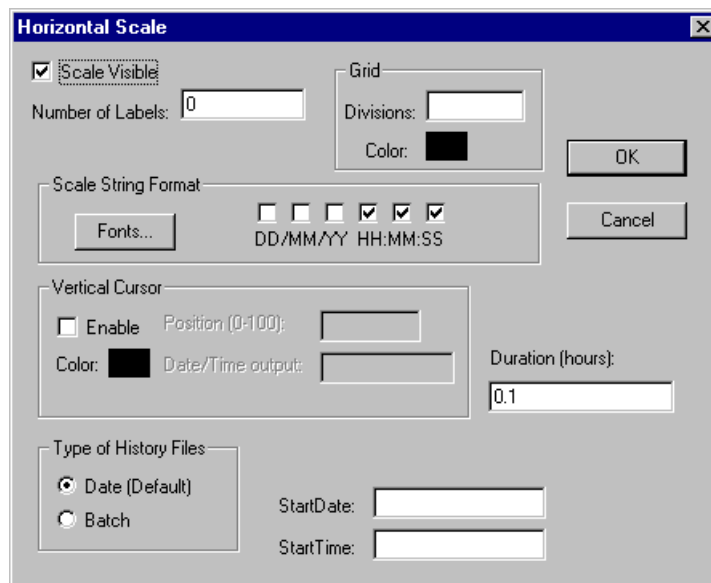
- **Hours Before Now Field** –可以利用此設定用捲軸顯示幾小時前的紀錄. 例如: 如果要顯示在此之前五小時的趨勢圖表, 您可以利用內部點 HOUR 來作動態的設定, 如果小時 = 5 即可顯示 5 小時前的紀錄.

警告: 在視窗中的趨勢資料可保持的樣品數為 16,000 .

備註: 當您使用 **Hours Before Now** 時, 因是處理歷史資料, 所以您必須先在趨勢群組中規劃群組裡畫趨勢圖的筆.

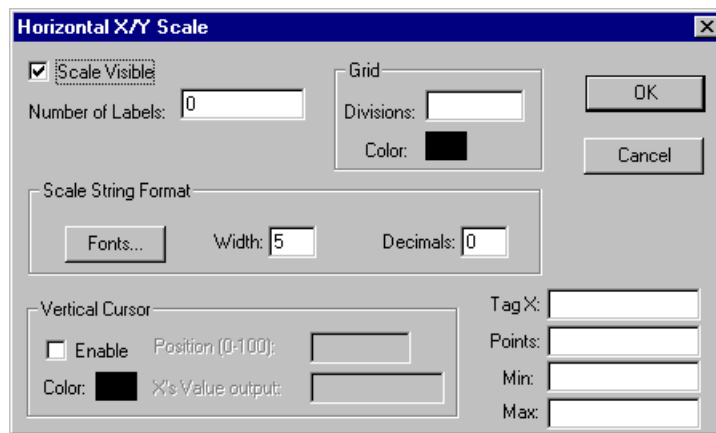
- **HORIZONTAL SCALE WINDOW- Curve Type X/t (HISTORY)**

若在 **Object Properties** 趨勢視窗中選擇 **History** 和 **Graph X/t** 選項時, 則不僅可以用來處理過去歷史資料, 同時還可以處理批次檔案.



- **Scale Visible** Check-box -設定趨勢圖表尺度為可見.
- **Number of Labels** Field -設定用於趨勢圖表分割的標籤數.
 - **Divisions** Field -設定趨勢圖區格線(行)的數目.
 - **Color** Rectangle -定義趨勢圖區格線的顏色. 點選 **Color** 旁的矩形可顯示設定顏色的視窗. 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕.如果您沒有在欄位中並沒有鍵入資料, 則不會顯示任何格線.
- **Scale String Format** Group Box -規定用於趨勢圖的水平尺度標示字串的格式.
 - **Fonts** Button -開啓定義字型的視窗, 您可以設定水平標籤字型的樣式, 尺寸, 顏色, 和字體型式.
 - **DD/MM/YY-HH:MM:SS** Check-boxes -您可以設定標籤上的字串是否有小時, 分鐘和秒.
- **Vertical Cursor** Group Box
 - **Enable** Check-box -在趨勢視窗中可使用垂直的游標軸.
 - **Color** Rectangle -可規畫垂直游標軸的顏色.
 - **Position (0-100)** Field -當使用這個垂直游標時, 必須在此欄位中鍵入一個資料點(Tag)用來更新游標軸的位置.
 - **Date/Time Output** Field -選擇一個資料點(Tag)用來接收垂直游標軸目前所在的時間點.

- **Duration (hours) Field** -定義趨勢圖所顯示的時間範圍. 這個欄位中可能是一個資料點(Tag)或者一個數字(值). 例如: 如果在欄位中鍵入時間= 0.03333 (即為 2 分鐘).
- **Types of History Files Group Box**
 - **Date** -所讀取的歷史資料檔在趨勢圖群組中是以日期格式作為檔名.
 - **Batch** -所讀取的歷史資料檔在趨勢圖群組中是以批次檔格式作為檔名.
- **Start Date Field** -歷史曲線開始的日期. 通常可使用一個字串類型的資料點(Tag). 它的日期格式必需是 DD/ MM/ YYYY .
- **Start Time Field** -設定歷史曲線開始的時間.
- **HORIZONTAL SCALE WINDOW- Curve Type X/Y**
若在 **Object Properties** 趨勢視窗中選擇 **X-Y** 選項時, 則可以用來處理從資料庫中讀到資料陣列中的大量資料, 將資料轉成趨勢圖, 您可定義 TagX 的資料點及垂直座標的最大最小值.

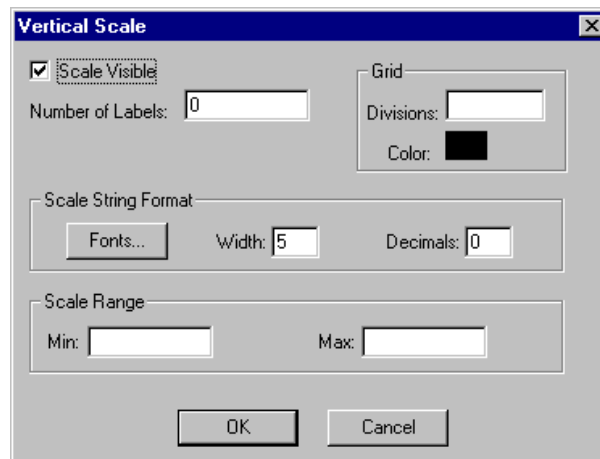


- **Scale Visible** Check-box -設定趨勢圖表尺度為可見.
- **Number of Labels** Field -設定用於趨勢圖表分割的標籤數.
- **Grid** Group Box
 - **Divisions** Field -設定趨勢圖區格線(行)的數目.
 - **Color** Rectangle -定義趨勢圖區格線的顏色. 點選 **Color** 旁的矩形可顯示設定顏色的視窗. 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕. 如果您沒有在欄位中並沒有鍵入資料, 則不會顯示任何格線.
- **Scale String Format** Group Box -規定用於趨勢圖的水平尺度標示字串的格式.
 - **Fonts** Button -開啓定義字型的視窗, 您可以設定水平標籤字型的樣式, 尺寸, 顏色, 和字體型式.
 - **DD/MM/YY-HH:MM:SS** Check-boxes -您可以設定標籤上的字串是否有小時, 分鐘和秒.
- **Vertical Cursor** Group Box
 - **Enable** Check-box -在趨勢視窗中可使用垂直的游標軸.
 - **Color** Rectangle -可規畫垂直游標軸的顏色.
 - **Position (0-100)** Field -當使用這個垂直游標時, 必須在此欄位中鍵入一個資料點(Tag)用來更新游標軸的位置.
 - **Date/Time Output** Field -選擇一個資料點(Tag)用來接收垂直游標軸目前所在的時間點.
- **Tag X** Field - 設定 X 軸的資料點(Tag), 此點必須是陣列型式, 要填此欄位時必須宣告陣列開始的位置. (e.g .MyTagX[1])
- **Points** Field - 圖表視窗取樣的樣品數.
- **Min** Field -資料點的最小值.

- **Max Field** -資料點的最大值.

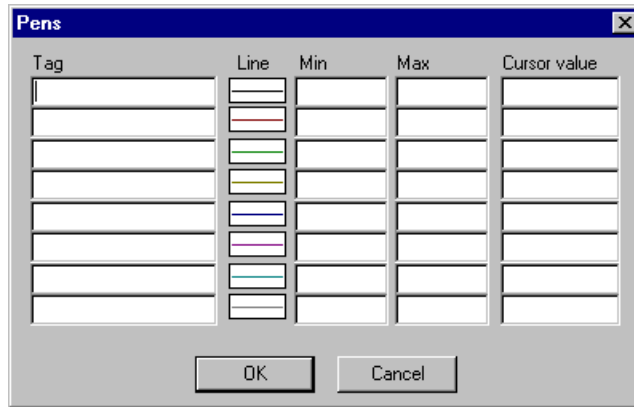
提示: 您可以使用 **Tasks** 標籤中的配方的特性來存檔使設定 X - Y 方式的歷史趨勢可以使用.

- **VERTICAL SCALE WINDOW** 垂直刻度窗.
在趨勢視窗的垂直尺度鈕上進入設定垂直尺度視窗的畫面.




- **Scale Visible** Check-box -設定趨勢圖表尺度為可見.
- **Number of Labels** Field -設定用於趨勢圖表分割的標籤數.
- **Grid** Group Box
 - **Divisions** Field -設定趨勢圖區格線(行)的數目.
 - **Color** Rectangle -定義趨勢圖區格線的顏色. 點選 **Color** 旁的矩形可顯示設定顏色的視窗. 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕.如果您沒有在欄位中並沒有鍵入資料, 則不會顯示任何格線.
- **Scale String Format** Group Box
 - **Fonts** Button -開啓定義字型的視窗, 您可以設定水平標籤字型的樣式, 尺寸, 顏色, 和字體型式.
 - **Width** Field -定義垂直座標的字串數目.
 - **Decimals** Field -定義垂直座標的字串小數的位數. 例如: 數值 " 長度= 3 " 和 " 小數= 2 " 表示有 3 個數字個小數點後有 2 位.
- **Scale Range** Group Box -比例值的範圍.
 - **Minimum** Field -趨勢圖座標的最小值.
 - **Maximum** Field -趨勢圖座標的最大值. 計算曲線位置的資料點或值在 **Pen** 視窗中定義.


- **PENS WINDOW**
按 **Pens** 按鈕進入設定趨勢圖鋼筆的視窗.



- **Tag Field** - 設定在趨勢圖中被監控的資料點(Tag)的名字. (e.g . MyTagY [1]), 你也可以用一個間接資料點來定義, 使應用更彈性化.
- **Line Box** - 定義每一個趨勢曲線的顏色. 點選 **Color** 旁的矩形可顯示設定顏色的視窗. 雙擊(double-click)想要的顏色或點選想要的顏色之後按 **OK** 按鈕. 最多可定義八條不同顏色的曲線.
- **Min Field** - 曲線的最小值; 可以是一個數值或者一個資料點(Tag).

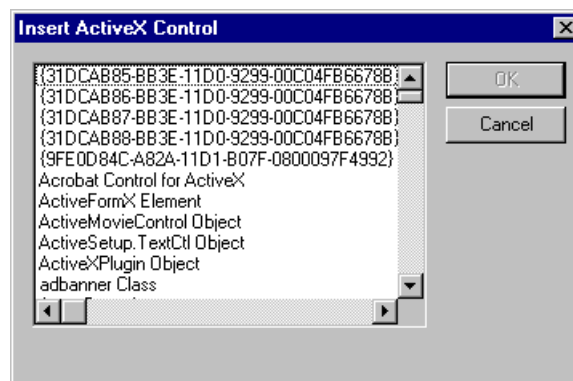
 **備註:** 每一個資料點(Tag)的最小值不一定要與趨勢圖座標的最小值相同.


- **Max Field** - 曲線的最大值; 可以是一個數值或者一個資料點(Tag).

 **備註:** 每一個資料點(Tag)的最大值不一定要與趨勢圖座標的最大值相同.

- **Cursor Value Field** - tag, 定義一個資料點用來接收垂直游標的值.

 **ActiveX Control:** 列出在您的電腦台中已註冊的 ActiveX 元件. 您可以選擇其中之一把物件載入螢幕中.



 **提示:** 您可以使用 XGet (), XSet () 和 XRun () 的功能在執行期來讀或寫屬性設定或執行 ActiveX 的方法.

備註: Windows CE 並不支援 ActiveX.

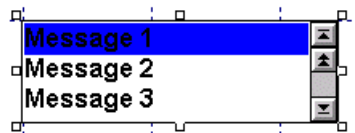


List Box ObjectIB: 是訊息的表單. 如果所列的表單列數比物件還長, 此物件會自動包含捲軸. 使用者能自由選擇表單中的訊息. 當訊息被選定時便會將值寫入一個資料點中.

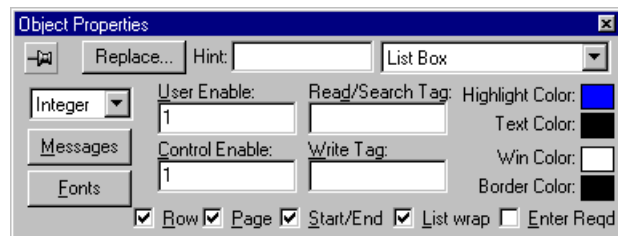
在執行期間, 表單將顯示多個訊息其中之一會以反白呈現. 如果只有表單盒而沒有任何訊息在其中, 表單盒物件還是會有效的運作. 如果有若干個表單盒和文字輸入的物件一起顯示, 您可用滑鼠或觸控式螢幕來操作或使用 Tab 鍵來選擇要操作那個物件. 若是停留在表單盒時, 當訊息被選定便會將值寫入一個資料點中.

您可以規劃兩種不同方式來選擇表單盒中的訊息, 差別在於是否勾選了物件屬性視窗中 **Enter Reqed** 欄位. 如果勾選了物件屬性視窗中 **Enter Reqed** 欄位, 則您能夠使用幾種方式來選擇訊息捲軸, 如: 鍵盤/輔助鍵盤, 圖庫中的列表控制物件, 指向裝置, 或是使用者利用 PostKeys () 自訂的功能鍵. 當您要選擇所要的訊息時, 使用 Enter 鍵來選定同時將值寫入一個資料點中. 在按下 Enter 鍵之前使用 Esc 及 Tab 可回到前一個畫面. 如果並無勾選物件屬性視窗中 **Enter Reqed** 欄位, 一旦訊息反白被選定便會將值寫入一個資料點中.

在螢幕中新增一個表單物件時, 在編輯工具中選擇 **List Box**, 然後把滑鼠放在區域範圍內, 按下滑鼠左鍵並拖拉這滑鼠以調整形狀. 矩形的高度決定一次可顯示的訊息數而廣度將決定能夠顯示的訊息長度. 通常物件產生後還是可以調整矩形的大小並且在給定的空間中也能夠改變字體的特性以便顯示更多和更長的訊息. 在一個畫面中可同時並存多個表單盒物件.

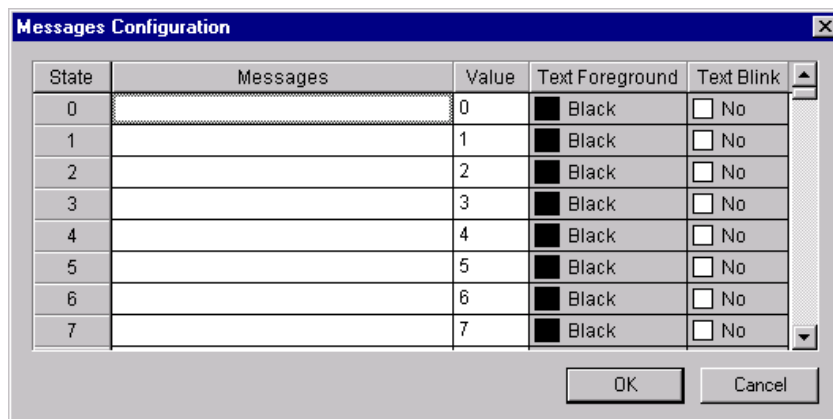


雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



- **Value drop**-此列表是在 **Replace...** 的下方, 用來作為訊息表單的索引. 選項有布林, 整數 (預設值) 和最低有效位元. 在 **Messages** 視窗中在 **State** 欄中可看見更多資訊.
- **User Enable** text box: 您可填入一個資料點(Tag), 運算式, 或者數字. 如果這個值是非零的值, 使用者在應用程式執行期能夠選擇一個訊息. 預設值為 1(true, enabled).
- **Control Enable** text box: 您可填入一個資料點(Tag), 運算式, 或者數字. 如果這個值是非零的值, 選擇的訊息將取決於目前讀取或搜尋的資料點(Tag) 的值. 此值會與在 message 視窗中所設的 value 欄位作鏈結. 預設值為 1(true, enabled). 若設定為 1 時將可用資料點(Tag) 值的改變來控制訊息列的選擇.
- **Read/Search Tag** text box: 您可填入一個布林或整數型態的資料點(Tag), 此點的值會與訊息列的選擇而隨之移動.
- **Write Tag** text box: 您可填入一個布林或整數型態的資料點(Tag), 可使用此點的值來控制訊息列的選擇.
- **Row** check box: 當選擇此項時, 表單盒物件的捲軸控制項會包括往下和往上箭頭.
- **Page** check box: 當選擇此項時, 表單盒物件的捲軸控制項會包括往上一頁和往下一頁.
- **Start/End** check box: 當選擇此項時, 表單盒物件的捲軸控制項會包括跳回頂端和末端的箭頭.

- **List wrap** check box: 當選擇此項時, 表單盒物件的捲軸若已到末端時, 會自動跳回頂端, 相反的若已到頂端則會跳到末端.
- **Enter Req'd** check box: 當選擇此項時, 只有 **Enter** 鍵可選擇訊息, **Tab** 鍵將無法選擇訊息.
- **Color Boxes**: 顏色盒的設定可控制表單盒物件的顯示顏色. 選擇這個顏色盒子來改變顏色. 顏色視窗中共有 16 - 顏色的顏色可選擇. 選擇顏色按 **OK** 或直接雙擊(double-click)顏色.
 - **Highlight Color** box: 訊息反白的顏色(預設值為藍色).
 - **Text Color** box: 訊息文字的顏色(預設值為黑色).
 - **Win Color** box: 表單盒背景的颜色(預設值為白色).
 - **Border Color** box: 表單盒邊框的顏色(預設值為黑色).
- **Fonts** button: 此視窗用來改變訊息文字字體的特性.
- **Messages** button: 開啓設定訊息架構的視窗.



- **State** field: 作為個別訊息的索引. 會與 read/search 資料點(Tag)作鍵連.
 - Boolean*: 有兩種有效狀態, 0 或 1.
 - Integer*: 有 255 種有效狀態, 由 1 到 255.
 - LSB*: 有 32 種有效狀態, 由一個整數值分析出 32 位元, 由 0 到 31.
- **Message** field: 在此區域中包含了在表單盒物件中所顯示的字串訊息. 您也可以使用 `{tag name}` 句法使訊息可隨時更動.
- **Value** field: 此欄位可設定與 read/search 資料點(Tag)互相搭配的訊息值. 此值也就是會被寫入 write tag 中所設定的資料點(Tag)的值. 如果 Value 物件的屬性是值選擇最低有效位元(LSB), 此欄位將不被使用而是用 State 來與 read/search 及 write 所設定的資料點(Tag)連結.
- **Text Foreground** color box field: 開啓選擇 16 - 色的視窗.
- **Text Blink** check box field: 選擇此項時, 訊息在顯示時會閃爍.



Smart Message Objects: 用來顯示訊息和圖形. 有三種顯示類型可選用, 將訊息值寫入資料點(Tag)和能夠顯示大量的訊息和圖形. 類型共有 **Message Display**, **Multistate Indicator** 及 **Multistate Puchbutton**.

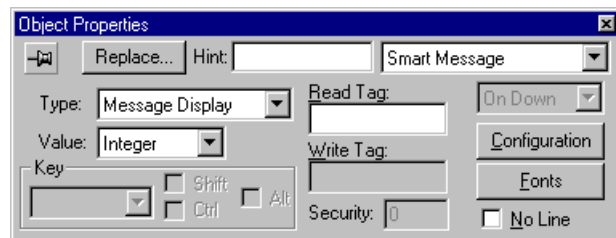
此訊息物件能夠接受操作員或程式在運行中的輸入而決定顯示哪個訊息. 在此單物件中可顯示許多訊息當中的一個訊息來顯示. 選擇 Multistate Indicator 也是一樣的作用, 只是它可用圖形來顯示訊息. Multistate Pushbutton 則訊息不僅可以以點陣圖方式顯示, 同時操作員可透過此方法作正反的切換, 就如同切換不同的開關.

在執行期, 訊息將顯示被選擇的訊息或圖形(取決於您的規畫).

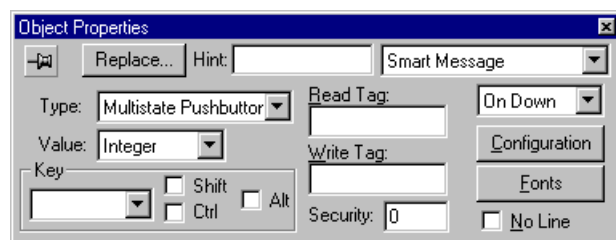
在螢幕中新增一個訊息物件時, 在編輯工具中選擇 **Smart Message**, 然後把滑鼠放在區域範圍內. 按下滑鼠左鍵並拖拉這滑鼠以調整形狀. 矩形的高度決定一次可顯示的訊息數而廣度將決定能夠顯示的訊息長度. 通常物件產生後還是可以調整矩形的大小並且在給定的空間中也能夠改變字體的特性以便顯示更多和更長的訊息. 在一個畫面中可同時並存多個訊息表單盒物件.



雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



- **Type** drop-down list: 此下拉式視窗可以讓您選擇訊息物件的型態. 選項有 *Message Display*(預設), *Multistate Indicator*, *Multistate Pushbutton*. 在 *Configuration* 視窗中可以看到更多的資訊. 此項設定及 *Value* 欄位可決定訊息的最大數目. 當選擇 *Multistate Pushbutton* 時, 原本反白的相關項目會自動變成可以使用.



- **Value** drop-down list: 用來作為訊息表單的索引. 選項有布林, 整數 (預設值)和最低有效位元. 在 **Configure** 視窗中在 **State** 欄中可看見更多資訊.
- **Read Tag** text box: 您可填入一個布林或整數型態的資料點(Tag), 此點的值會與訊息的選擇而跟著改變.
- **Write Tag** text box: 用來與 *Multistate Pushbutton* 互相搭配使用. 您可填入一個字串型態的資料點(Tag)來接收訊息列的最後選擇的值. 當畫面關掉後重開時此最後被寫入的值即決定畫面開啓時哪個訊息被顯示.
- **Security** text box: 此欄位可接受 0 到 255 的值以控管按 *Multistate Pushbutton* 使用者的權限, 在 **Security** 功能中可定義此值. 若您在此欄中設定 0 或空白即表示 *Multistate Pushbutton* 任何人員均可操作.
- **No line** check box: 假如選擇此項, 則訊息列將不顯示外框.

- **Key box:** 定義快速鍵或複合鍵用來快速切換到下一個訊息，此設定只有在 **Multistate Pushbutton** 才有用。
 - **Key drop-down list:** 此下拉式選單包含了鍵盤上的功能鍵。
 - **Shift, Ctrl, and Alt check boxes:** 作為複合式的功能鍵。

- **Fonts button:** 此視窗用來改變訊息文字字體的特性。

- **Configuration button:** 開啓 Configuration 視窗：

▪ **Smart Message Configuration Window**

Configuration 視窗的內容會隨著訊息型態而不同(Message Display, Multistate Indicator, and Multistate Pushbutton), 在 Multistate Pushbutton Configuration 視窗中包含了所有的欄位. Multistate Indicator Configuration 視窗中不包含 Value 的欄位. Message Display Configuration 視窗中不包含 Value, Graphic File, Transparent 的欄位. 欄位說明如下：

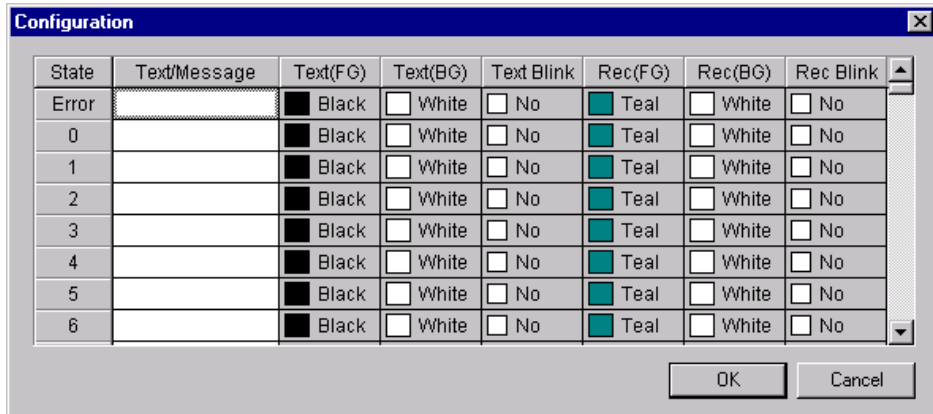
- **State field:** 作為個別訊息的索引. 會與 read/search 資料點(Tag)作鍵連. **Error state** 則可定義當 read/search 資料點(Tag)為不合法的值(Boolean or Integer or LSB)時將顯示的訊息. 在 Multistate Pushbutton 沒有 **Error State** 欄, **State** 欄位與 **Value** 和 **Type** 設定是互相關連的：

Integer: .

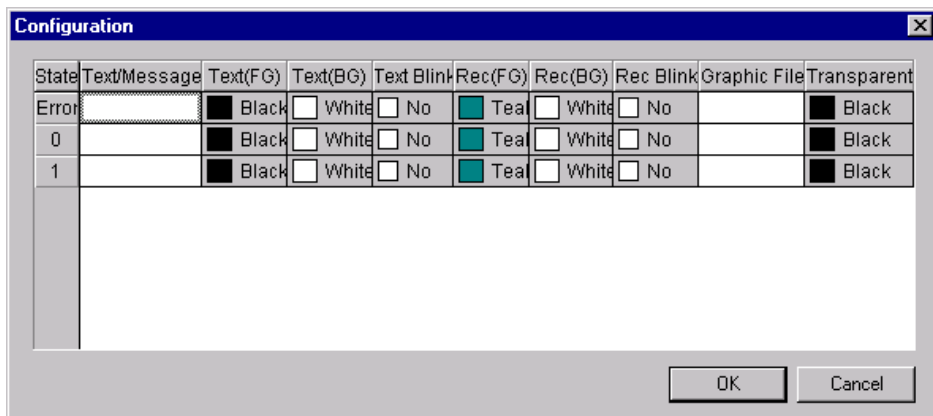
LSB: 有 32 種有效狀態, 由一個整數值分析出 32 位元, 由 0 到 31 .

- - *Boolean:*有兩種有效狀態, 0 或 1.
 - *Integer:* Message Display 及 Multistate Indicator 有 500 種有效狀態, 由 0 到 499, Multistate Pushbutton 有 15 種有效狀態, 由 0 到 15.
 - *LSB:* Message Display 及 Multistate Indicator 有 32 種有效狀態, 由一個整數值分析出 32 位元, 由 0 到 31 . Multistate Pushbutton 有 16 種有效狀態, 由 0 到 15.
- **Message field:** 在此區域中包含了在訊息物件中所顯示的字串. 您也可以使用 {tag name} 句法使訊息可隨時更動.
- **Value field:** 此欄位可設定與 read/search 資料點(Tag)互相搭配的訊息值. 此值也就是會被寫入 write tag 中所設定的資料點(Tag)的值. 如果 **Value** 物件的屬性是值選擇最低有效位元(LSB), 此欄位將不被使用而是用 **State** 來與 read/search 及 write 所設定的資料點(Tag)連結.
- **Text (FG) color box field:** 可設定訊息文字的前景顏色. 選擇這個顏色盒子來改變顏色. 顏色視窗中共有 16 - 顏色的顏色可選擇. 選擇顏色按 OK 或直接雙擊(double-click)顏色.
- **Text (BG) color box field:** 可設定訊息文字的背景顏色. 選擇這個顏色盒子來改變顏色. 顏色視窗中共有 16 - 顏色的顏色可選擇. 選擇顏色按 OK 或直接雙擊(double-click)顏色.
- **Text Blink** 選擇此項時, 訊息文字在顯示時會閃爍
- **Rec (FG) color box field:** 可設定訊息的外框顏色. 選擇這個顏色盒子來改變顏色. 顏色視窗中共有 16 - 顏色的顏色可選擇. 選擇顏色按 OK 或直接雙擊(double-click)顏色.
- **Rec (BG) color box field:** 可設定訊息的背景顏色. 選擇這個顏色盒子來改變顏色. 顏色視窗中共有 16 - 顏色的顏色可選擇. 選擇顏色按 OK 或直接雙擊(double-click)顏色.
- **Rec Blink check box field:** 訊息在顯示時會閃爍.

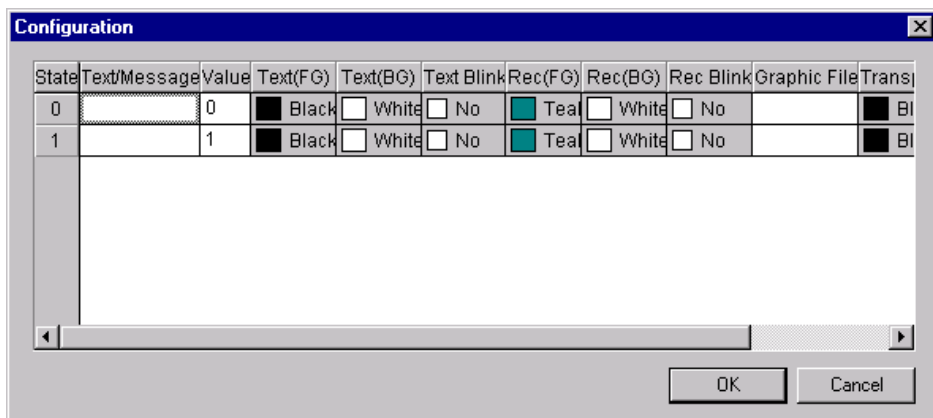
- **Graphic File field:** 您可輸入點陣圖的檔名(. bmp). 在 Windows CE 環境下此檔必需在 `\application\symbol` 目錄下. 若在 Windows NT 環境下, 則可加上檔案的路徑, 因此檔案位置可以在另外的目錄下. 但是預設的路徑依然是 `\application\symbol`.
 - **Transparent color box field:** 可使所顯示的點陣圖作通透性處理. 選擇這個顏色盒子來改變顏色. 顏色視窗中共有 16 - 顏色的顏色可選擇. 選擇顏色按 OK 或直接雙擊(double-click)顏色.
- **Message Display Smart Message Configuration Window**




- **Multistate Indicator Smart Message Configuration Window**



- **Multistate Pushbutton Smart Message Configuration Window**

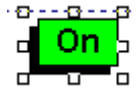


 **Pushbuttons:** Pushbuttons 此物件包含在 Command dynamic object 中. 它有三種型態: **Momentary**, **Maintained**, 及 **Latched**. 此三種型態是模擬標準面板而作名稱也相同.

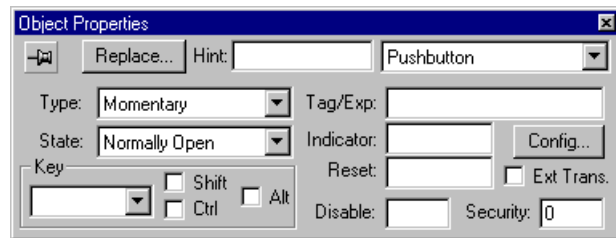
Pushbutton 有四種樣式: 均為矩形, 面板上可以有或沒有訊息顯示, 也可以是指示燈, 3-D 立體或是流動型態. 預設型態是 **Momentary** 樣式為沒有訊息顯示具燈號及流動指示.



在螢幕中新增一個 Pushbutton 物件時, 在編輯工具中選擇 Pushbutton, 然後把滑鼠放在區域範圍內. 按下滑鼠左鍵並拖拉這滑鼠以調整形狀. 矩形的高度決定一次可顯示的訊息數而廣度將決定能夠顯示的訊息長度. 通常物件產生後還是可以調整矩形的大小並且在給定的空間中也能夠改變字體的特性以便顯示更多和更長的訊息. 在一個畫面中可同時並存多個 Pushbutton 物件.



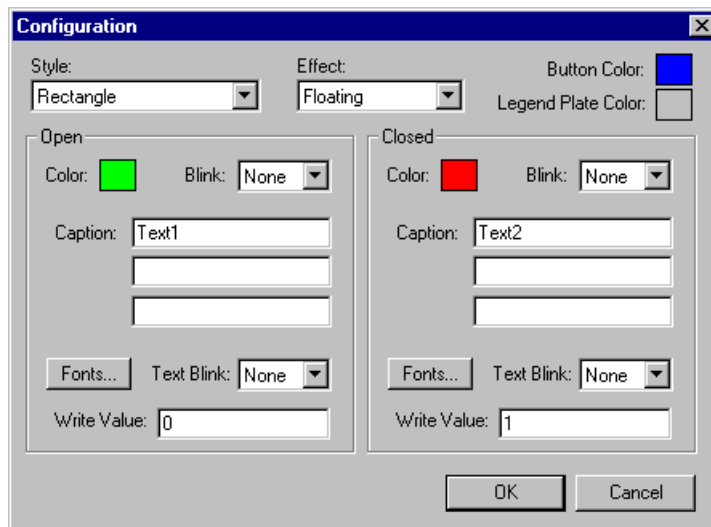
雙擊(double-click)此物件, *object Properties* 視窗將會顯示出來.



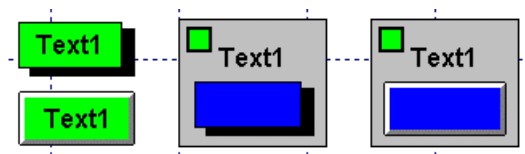
- **Type drop-down list:** 選擇訊息顯示的方式. 選項有 *Momentary* (預設值), *Maintained*, 及 *Latched*.
 - *Momentary:* 選擇此項, 可用 Pushbutton 來改變狀態(開啓或關閉), 按下時狀態即改變放開即變回.
 - *Maintained:* 選擇此項, 可用 Pushbutton 來改變狀態(開啓或關閉), 按下時狀態即改變放開時並不會變回. 必需要再按一下 Pushbutton 狀態才會變回.
 - *Latched:* 選擇此項, 可用 Pushbutton 來改變狀態(開啓或關閉), 按下時狀態即改變. 狀態會保持到 Reset tag 狀態改變才會改變, 此型態即為自保迴路.
- **State drop-down list:** 您可選擇 Pushbutton 的預設狀態值. 選項有 *Normally Open* (預設) 及 *Normally Closed*. 當按下按鍵時, 會改變為非預設值當變回時則會回到 Type 所設定的狀態. 在初始狀態時, 也會依據 Configuration 視窗中的設定來作顯示.
- **Tag/Exp text box:** 此欄的資料點是用來接收在 Configuration 視窗中群組盒 Write Value 所設定的值. 假如 Tag/Exp 欄位中是運算式的話, 則只要按鍵被按下即執行此運算式.
- **Indicator text box:** 假若在 Configuration 視窗中 Style 是選擇 *Rectangle with Indicator*, Pushbutton 將會多了類似指示燈的圖示在左上方, 燈號的顏色將隨著按鍵而改變. 當 Indicator 欄位中空白時, 則燈號顏色會隨著 Configuration 中 open 及 close 的顏色改變. 若是欄位中為非零的值則燈號會隨著所連結資料點而變.

假如您在此欄位中並未鍵入任何資料點而且 Pushbutton 型態包括燈號, 則當按下時僅僅是燈號會隨著改變. 若是 Pushbutton 並不包括燈號則按下時按鍵會隨著變色就如同燈號一般.
- **Reset text box:** 此欄位只有在設定為 *Latched* 樣式時才有用. 當所鏈結的資料點為 0 時, 按鍵會處於被鎖住狀態. 只有當資料點為非零的值時才能重新操作.
- **Key box:** 定義快速鍵或複合鍵用來快速切換到下一個訊息, 此設定可在無滑鼠或指示裝置時才有用.
 - **Key drop-down list:** 此下拉式選單包含了鍵盤上的功能鍵.

- **Shift, Ctrl, and Alt check boxes:** 作為複合式的功能鍵。
- **Disable text box:** 您可在欄位中鍵入一個資料點，當值為非零時則 **Pushbutton** 將會無效，按下時將不會產生任何反應。預設為空白，表示 **Pushbutton** 是有效的。
- **Ext Trans. check box:** 勾选此項將可用語言對照表來作語言轉換。您可參考 **Language Translation** 功能。
- **Security text box:** 此欄位可接受 0 到 255 的值以控管按 **Pushbutton** 使用者的權限，在 **Security** 功能中可定義此值。若您在此欄中設定 0 或空白即表示 **Multistate Pushbutton** 任何人員均可操作。
- **Config... button:** 開啓 **Configuration** 視窗在當中您可以設定型態及狀態的參數，打開視窗如下：



- **Style drop-down list:** 利用下拉式視窗來選擇 **Pushbutton** 型態。選項有 *Rectangle* (預設值) 及 *Rectangle with Indicator*。
- **Effect drop-down list:** 利用下拉式視窗來選擇 **Pushbutton** 為 3-D 立體效果。選項有 *Floating* (預設值) 及 *3D*。選擇 *Floating* 就像平面的物件作了陰影的效果；選擇 *3D* 在邊緣作了處理同時在按下時按鍵會下沉。
- **Style 及 Effect 選項** 可以讓您製作四種不同的按鍵，顯示如下。左上方的 **Pushbutton** 是 *Floating Rectangle* (預設值)，左下方是 *3D Rectangle*，中間則是 *Floating Rectangle with Indicator*，右邊是 *3D Rectangle with Indicator*。

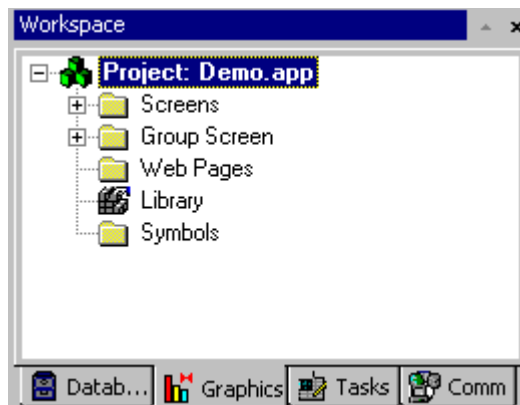


- **Button Color color box:** 可設定 **Pushbutton** 物件當設定為按鍵型態包括了指示燈時按鍵部份的顏色。選擇這個顏色盒子來改變顏色。選擇顏色按 **OK** 或直接雙擊(double-click)顏色。
- **Legend Plate Color color box:** 若 **Pushbutton** 的型態包括了指示燈時，除了按鍵和指示燈外還有文字顯示的部份。此項可讓您設定文字的背景顏色。選擇這個顏色盒子來改變顏色。選擇顏色按 **OK** 或直接雙擊(double-click)顏色。假如是選擇不含指示燈型態時，此項將沒作用。

- **Open and Closed group boxes:** 此欄所含的群組盒是用來規劃 Pushbutton 物件所顯的狀態(Open 及 Closed).
- **Color color box:** 當 Pushbutton 物件設定為按鈕型態包括了指示燈時, 此欄位可以讓您選擇指示燈在不同狀態下顯示不同的顏色. 當 Pushbutton 物件設定為按鈕型態不包括指示燈時, 此欄位可以讓您選擇按鈕在不同狀態下顯示不同的顏色. 選擇這個顏色盒子來改變顏色. 選擇顏色按 OK 或直接雙擊(double-click)顏色.
- **Blink drop down list:** 此欄位可設定所選定的顏色是否要閃爍以及要閃得多快. 每一個下拉式選單均可設定. 選項有 *None* (不閃爍—預設值), *Slow*, 及 *Fast*. 假如設定為顏色閃爍則閃爍的方式會依 **Color** 的設定作相互的交換, 即使是 **Legend Plate Color** (假若設為指示燈) 或是 **Button Color** (假若設為按鈕)也會有此功能.
- **Caption text boxes:** 此三個文字盒包含了最上層, 中間, 及最下層文字的設定(依位置), 按鈕將依設定來顯示(假如 **Pushbutton Style** 並不包括指示燈).
- **Fonts... button:** 此視窗用來改變訊息文字字體的特性.
- **Text Blink drop-down list:** 每個下拉式視窗可定義文字是否要閃爍或要閃多快. 選項有 *None* (不閃爍—預設值), *Slow*, 及 *Fast*. 與顏色閃爍不同的是, 閃爍的文字是以可見和不可見來顯示.
- **Write Value drop-down list:** 欄位中所設定的值會被寫入在 **Tag/Exp** 欄所設的資料點(Tag)內.

II.5. 工作區 (Workspace)

Studio 的工作區(*Workspace*)是一個方便的使用者介面, 可以讓使用者快速地找到可運用的發展元件 (tags, screens, worksheets, 等等). 這些可運用的元件由樹狀的架構來呈現, 並且各有其圖示和描述. 使用者可以移動, 重定大小或者隱藏工作區(*Workspace*)視窗.



工作區(*Workspace*)視窗分成四個標籤. 每一個標籤的檔案夾和元件的圖示將在下面描述. 使用者可以將游標移到檔案夾或元件的圖示上按右鍵以打開各相對的功能目錄.

- *Database* 標籤讓使用者可以使用設定應用程式中所使用的資料點(Tag)以及安全系統元件. 這個標籤包含了下列的檔案夾:
 - Application Tags
 - Classes
 - Shared Database
 - Internal Tags
 - Security

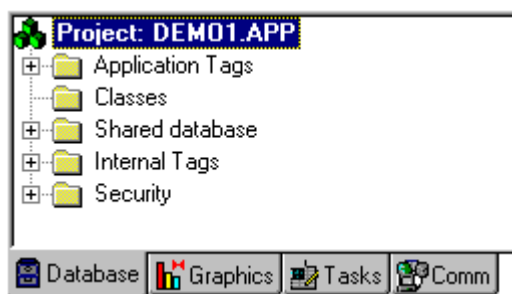
- *Graphics* 標籤讓使用者可以使用在這個應用程式中用到的 screens 以及 symbols. 這個標籤包含了下列的檔案夾：
 - Screens
 - Group Screen
 - Web Pages
 - Library
 - Symbols

- *Tasks* 標籤讓使用者可以使用在這個應用程式中所有的 task worksheets. 這個標籤包含了下列的檔案夾：
 - Alarms
 - Trend
 - Recipes
 - Report
 - ODBC
 - Math
 - Scheduler

- *Comm* 標籤讓使用者可以使用不同的通訊介面與另一裝置或軟體作溝通. 這個標籤包含了下列的檔案夾：
 - Drivers
 - OPC
 - TCP/IP
 - DDE


II.5.1. *Database* 標籤


Database 標籤讓使用者可以使用在這個應用程式中任何可用的資料點(Tag)以及安全系統元件. *Database* 標籤即如下列圖示：

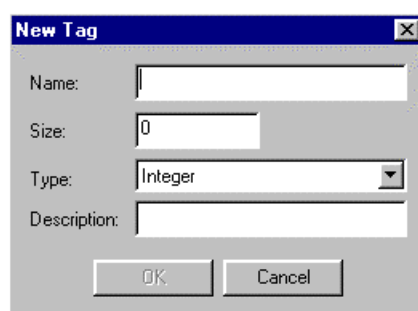


▪ *Application Tags* 檔案夾

Application Tags 檔案夾含有由使用者製作和定義的所有標籤. *Application tags* 是爲了顯示而製作的標籤, 它是對指定設備執行讀取與寫入的媒介, 也是用來執行控制或數學運算的工具. 在 *Application Tags* 檔案夾上按右鍵並選擇 *Refresh* 項目可以更新 *Application Tags* 的一覽表.


⚠ **警告：** 在刪除一個 tag 前, 強烈建議使用者使用在 *Tag Properties* 工作列上的 **Object Finder**  以確定這個 tag 不用於另一個應用程式 (screens, math sheets, 等等). 如果從應用資料庫刪除一個 tag, 但是, 此 tag 有另一個應用程式使用, 這將會導致編譯錯誤且應用程式將無法正常運作.

要創造一個新 tag, 可在 Application Tags 檔案夾上按右鍵, 執行 Tag List 次檔案夾, 或是利用 **Datasheet View** . 除此之外, 使用者可使用在主要選單的 Insert 下的 Tag 項目來達到此功能. New Tag 視窗如下:





New Tag 視窗功能如下:

- 在 Name 欄位可輸入達 32 個字. 第一個字元必須是字母.
 - Size 欄位可輸入 tag 的大小. 輸入範圍由 1 到 255 之間的任何數字. 如果輸入一個比 0 大的數字意味著 tag 是陣列的型式.
 - 從 Type 下拉式選單選擇一種 tag 類型. 除標準的 tag 類型以外(布林邏輯(Boolean), 整數(Integer), 實數(Real), 字串(String)), 使用者還能夠透過類別 (classes) 將資料點 (Tag)定義成結構化的型態.
 - 在 Description 欄位輸入對於 tag 的描述.
- **Web Data** – 在這個欄位僅僅有兩個選擇, Local 和 Server. 如果使用者選擇了 Server, 意味著這個 tag 中的資訊將能在網路上分享. 如果使用者選擇了 Local, 便不能在網路上夠分享這個資訊. 若不使用網路功能則兩個選擇都會不影響應用程式. 如果使用者的應用程式使用了網路功能, 資料點(Tag)在 Web Data 欄位中選擇 Local 時, 所有使用到此資料點(Tag)的物件, 將不正常地工作.

 **注意：兩個標籤不能有相同的名字(Name)**

要查看或者編輯一個 tag 的性質, 可以使用 Tag Property 視窗或者 Application Datasheet 視窗.

如要查看 Tag Property 視窗, 在 Tag Properties 工作列按下 **Tag Properties**  按鍵, 此時 tag 名字會在這個 Tag name 欄位出現. 或者透過在 Application Tags 檔案夾中 Tag List 次檔案夾中的 tag 名字上雙擊按鍵(double-click)來打開 Tag Property 視窗.

欲打開工作表單(Datasheet), 在 Application Tags 檔案夾中按下 **Datasheet View** . 工作表單(Datasheet)由四個欄位所組成:



	Name	Size	Type	Description
8	simu	8	Integer	simulation integer
9	RecipeValue	8	Integer	recipe values
10	RecipeNum	0	Integer	Recipe Number
11	pgup	0	Boolean	Alarm pg up
12	pgdown	0	Integer	Alarm pg down
13	AckGroup	0	Boolean	Ack alarm group
14	valv	8	Boolean	
15	cursor	0	Real	trend cursor
16	i	0	Integer	integer to use in interactions
17	cont100	0	Integer	
18				
19				
20				
21				
22				

此工作表單(datasheet) 可讓使用者新增, 修改或者刪除資料點(Tag). 您可以選擇資料點(tag)上的任一屬性並按右鍵, 則會出現視窗命令, 您可執行如剪下 cut (Ctrl+X), 拷貝 copy (Ctrl+C), 和貼上 paste (Ctrl+V)任何資料點(Tag)和它的屬性. 另外, 您還可以復原欄位中最後修正的資料.

▪ The Classes 檔案夾


Classes 檔案夾含有所有應用類別和他們的各自成員. Classes 是由使用者定義的資料類型架構(data type structures)或者資料的類型(data types) (整數, 實數, 布林邏輯和字串)組成的一個複合式資料點(Tag). Class 的設計是考慮到資料庫中的高層次應用. 一個被定義為 class 類型的資料點(Tag)所代表的不僅僅只是一個值, 而是由它所定義的成員組合而成的集合.

定義一個 class 意味著定義其成員和成員的類型. Class 的成員是擁有具有特性的物件的值的變量. 如此一來, 當使用者常常應用一些重複的變量時, class 的定義將很有用.

 **注意:** 當一個 class 檔案夾被創造時, 一個 Class  圖示也會出現在 *Application Tags* 檔案夾的 *Tag List* 這個次檔案夾中.

要使用一個 class 的成員, 使用句點(.)來作為與 tag 的分隔符號如:
 <TagName>.<MemberName>. 例如: tk.LEV 或是 tk.TMP. 其中如果資料點(Tag) tk 被定為陣列的話, 句法將為<ArrayTagName>[<ArrayIndex>].<MemberName>.
 例如: tk[1].LEV 或是 tk[n].TMP.

當新增一個 class 形式的資料點(Tag)時, 它並沒有一個單一的值, 而是擁有一組值與其 class 相關聯. 使用者能夠把簡單的資料點(Tag)歸類成 class 的成員類. 任何 Class 的成員的最大數目取決於產品規格. 如同之前的描述 class 的成員能夠保持標準的值 (整數, 實數, 布林邏輯, 字串).

要新一個新的 class, 可以在 *Classes* 檔案夾, *Members List* 次檔案夾上, 或者 *Classe* 檔案夾中的 *Datasheet View*  按滑鼠右鍵進入. 或者, 使用者能夠在主要選單中 *Insert* 下選擇這個 *Class* 項目功能. 或者直攘在 *Application Tags* 檔案夾中創新增一個 class tag. 利用這些方式新增 class 時都將顯示 *Insert Class* 視窗.



在這個名字欄位輸入一個 class 的名字。

注意: 不可以讓兩個 class 擁有相同的名字, 同時 shared tags 與 internal tags 也不能被放在 class 中。

一旦新增了 class 後, 將顯示這個 Class 的工作表單(datasheet). Class 工作表單(datasheet)允許使用者新增, 修改, 或者刪除任何 class 的成員和它的屬性. (在 Tag Property 視窗不能夠編輯 Class.) Class 檔案夾如在 Tag Application 檔案夾中的 Tag List 次檔案夾中出現, 便能夠當成一個 Application Tag 來用。

	Name	Type	Description
1	level	Integer	
2	temperature	Integer	
3	pressure	Real	
4	valve	Boolean	
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			

使用者能夠選擇 class 的成員, 或者任何成員的屬性, 並在選項上按滑鼠右鍵, 使用者將可使用視窗命令如剪下 cut (Ctrl+X), 拷貝 copy (Ctrl+C), 和貼上 paste (Ctrl+V). 另外還能夠取消欄位的最後修正動作。

刪除 class 時僅能透過在 class 檔案夾上按滑鼠有鍵來作刪除. 這選項將刪除 class 和所有它的成員. 如果任何執行時期的 task 正在執行時則這個選項將無效. 它將不會刪除一個與其他資料點(tag)產生關聯的 class. 以下為 class 成員的屬性:


- 在 Name 欄位可輸入達 32 個字. 第一個字元必須是字母.
- 從 Type 下拉式選單選擇一種資料點(Tag)類型(布林邏輯(Boolean), 整數(Integer), 實數(Real), 字串(String))
- 在 Description 欄位輸入資料點(Tag)的描述.

注意: class 的成員不能是其他的 class 形態. 如果一個 class 名字已經存在, 它不能再用來當成另一個 class 的名字. 不過, 相同名字的 class 成員存在不同 class 中是被允許的。

- **Shared Database 檔案夾**



Shared database 檔案夾包含了所有 *Studio* 分享的資料點(Tag)以及被選擇的 PC Based Control. 您必須在 PC Based Control 軟體中新增創造和修改資料點(Tag)並且能在下面的條件下自動地載入到 *Studio* 中:


- 使用者開始執行 *Studio*.
- 使用者在 *Shared Tags* 檔案夾上按滑鼠右鍵, 將可更新這個資料庫.

 **注意:** 每個 PC Based Control 軟體有其特定的介面與條件讓 *Studio* 將其 tags 匯入其中. 舉例來說, 如要讓 *Studio* 可以匯入 database tags 可能需要 PC Based Control 軟體來執行它的應用程式.

如果沒有 PC Based Control 軟體與應用程式聯繫, *Datasheet View* 及在 *Shared Database* 檔案夾中的 *Tag List* 次檔案夾將會空白.

使用者在 *Studio* 環境中不能編輯分享的資料點(Tag), 但是, 使用者可以在 PC Based control 軟體中修改它們. 不過, 他們能夠像任何其他 tag 那樣的在 *Studio* task 中被設置. Shared tags 在 *Tag Property* 視窗和 *Shared Tag datasheet* 中是唯讀的.

 **注意:** 在 *Shared Database* 檔案夾上按滑鼠右鍵或者 click *Datasheet View*  並選擇 *Refresh* 選擇項以更新使用者的 PC-based control 的 tags database 為最新的版本. 為了改變 PC Based control 資料庫時 (新增 tag, 刪除 tag, 改變 tag 屬性), 必須啟動這個命令以更新 *Studio* shared database.

要查看 *Tag Property* 視窗時, 在 *Tag Properties* 工作列上按下 *Tag Properties* , 此時 *Tag name* 欄位將顯示資料點(Tag)的名字. 或者, 透過在 *Application Tags* 檔案夾中 *Tag List* 次檔案夾中的 tag name 上雙擊(double-click)來存取 *Tag Property* 視窗.

Shared Tag datasheet 含有四欄 (名字, 尺寸, 類型, 和描述) 只能夠用來查看 shared tags 而不能修改.

- **Internal Tags 檔案夾**

Internal Tags 檔案夾含有預設的資料點(Tag), 這些點具有特定的功能 (如:時間, 日期, 告知收到警報, 用戶登入的存儲, 以此類推). 且是不能編輯或者刪除. 不過, 能夠從任何 *Studio* task 來存取他們的值, 並且他們能夠在其它地方被拷貝和使用.

 **注意:** 在 *Internal Tags* 檔案夾上按右鍵或選擇 *Datasheet View*  並選擇 *Refresh* 選項來更新與 *Internal Tags files* 在一起使用的 *Studio Shared Database*.

使用者能夠使用 *Internal Tags* datasheet 來查看 internal tag 的屬性, 它含有四欄(名字, 尺寸, 類型, 和描述).

⚠警告: 大部分的 internal tags 只能查看. 例如, 改變時間設定時, 用原有的數學函數會比直接寫入內部的時間點(Tag)來的實際.

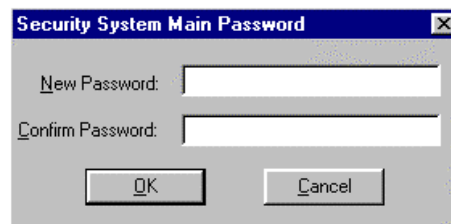
- **Security 檔案夾**

Security 檔案夾含有使用者群組和用戶的帳號, 它包括應用程式的安全系統. 此功能允許使用者定義群組和用戶, 以及他們的對 *Studio* 使用工具和對應用程式存取的權限. 透過 *Database tab*, 使用者能夠選擇或者創造新群組和用戶. 使用 *Security System* 視窗時, 在 *Security* 檔案夾上按右鍵即可.



如何用這視窗來設置安全系統, 說明如下:

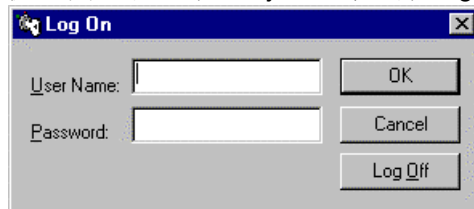
- 選擇 *Enable Security System* 查核框以啟動 *Studio* 的安全系統.
- 設定安全系統密碼時, 按 *Main Password* 鈕, *Security System Main Password* 視窗將出現如下:



在 *New Password* 欄位輸入密碼. 在 *Confirm Password* 欄位再輸入一次密碼確認密碼無誤. 然後按 *OK*.

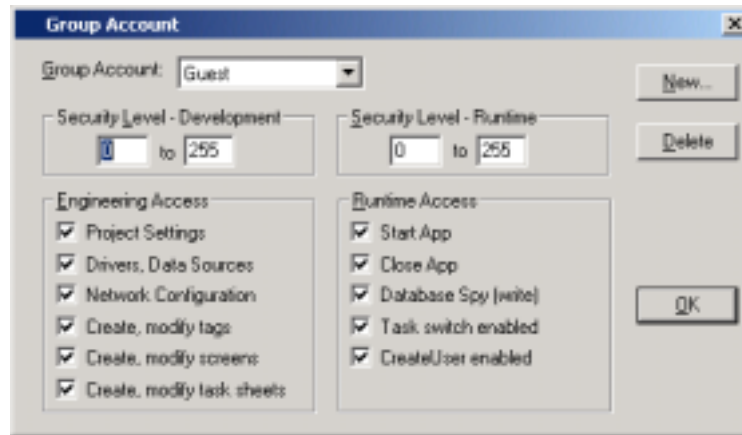
⚠警告: 在使用者定義密碼以後, 每次使用者存取安全系統將需要輸入此密碼, 所以使用者必需強記它.

- 使用者能夠透過從 *Project* 選單選擇 *Logon* 選項進行登入或登出.



輸入一個用戶名字和密碼在這些欄位上並按 *OK*. (或者按 *Log Off* 以退出系統. 當使用者退出時, **Guest** 用戶將自動地登入)如果沒有定義任何用戶, 內定的 **Guest** 將登入. 關於如何創造新用戶請參考下面的說明.

- 按 **Groups** 鈕以顯示 *Group Account* 視窗, 這允許使用者新增和維護用戶群組. 使用者能夠 **enable** 和 **disable** 這項操作以及設置安全範圍的權限. **Groups** 也能夠由開 **Security** 檔案夾之內的 **Groups** 檔案夾或者由主要選單在 **Insert** 下選擇這個 **Security Group** 選項來存取. 選擇要查看的群組.



規劃群組的方式如下：

- 從 *Group Account* 下拉式選單選擇一個群組來設置.
- 選擇群組在發展應用程式時的安全權限. *Security Level – Development* 可定義群組的安全權限範圍. 具體的指定 0 到 255 當中的一個數. 在 **Display screen** 中的的任何物件凡允許資料輸入(如輸入指令, **sliders** 或者 **screen**) 則會有一個安全權限的設定欄與此互相關聯. 如果物件所設定的權限不在當時登入的群組權限範圍中, 則您將無權操作此物件. 權限設 0 時, 意味著這個物件將不受權限所限制, 即任何人均可操作.

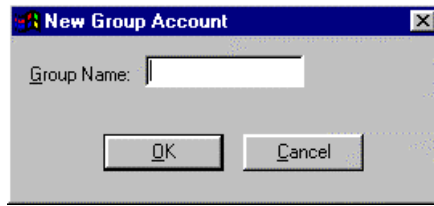
當開任何 *Advantech Studio* 中的工作表單(worksheet)時(Alarm, Math, Recipe, Report, Scheduler, TCP Client, Trend, OPC Client 和那些 CE 不支援的: DDE Client and ODBC), 可以單就工作表單(worksheet)設置存取範圍. 在工作表單(worksheet)上的任何一部分, 按主要選單 **Edit** 下的 **Access Level** 選項. 當選擇 **Access Level** 時, 使用者能夠規定一個 **access level**. 它意味著這個用戶如果想要再次編輯這個工作表單 **worksheet**, 則其 **access level** 必須與工作表單(worksheet)中的 **access level** 相匹配. 例如, GroupA 的 UserA 有 0 - 10 的權限範圍, GroupB 的 UserB 有 5 - 15 的權限範圍. 如果 Math worksheet 1 有一個 7 的 **access level**, 則兩個用戶都能夠存取它. 如果數學 worksheet 2 有 1 的 **access level**, 則只有 UserA 夠存取它.

- 同樣地, 透過規定 *Security Level - Runtime* 中的範圍, 設置用於執行時期的 **security level** 群組.
- 在 **Engineering Access** box 中, 選擇當這個小組裡的用戶登入時能夠存取的 **development tasks**.

⚠警告: 在發展環境中, 每一個 document(worksheets 和 displays)都可以設置 **Security level** 來做為保護. 使用者可以使用的項目可以參照 **Engineering Access** box.

- 在 **Runtime Access** box 中, 選擇當這個群組的用戶登入時能夠存取的執行時期模組.

要創造新群組, 按下在 *Group Account* 視窗上 **New** 按鈕, 將顯示下面的視窗:

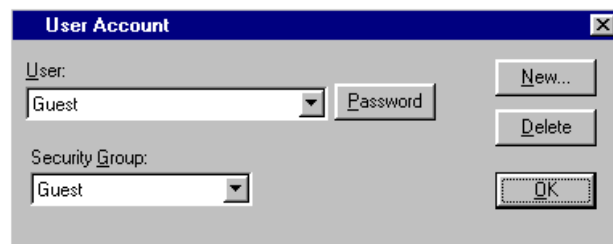


在這個欄位輸入新群組的名字並按 **OK**. 這個群組將用 *Group Account* 視窗中的設置選項來創造群組.

要刪除群組, 從 *Group Account* 下拉式選單選擇群組並按下 **Delete** 鈕.

注意: 使用者無法刪除 **Guest** 群組.

- 按下在 *Security System* 視窗上的 **Users** 鈕可以顯示 *User Account* 視窗, 它可以讓使用者新增及維護用戶帳號. 使用者可以為用戶者定一個群組. 要設置一個用戶, 使用者可以在 *Security* 檔案夾下打開 *Users* 檔案夾或是選擇在主選單上的 *Insert* 選單下的 *User* 選項. *User Account* 視窗如下:

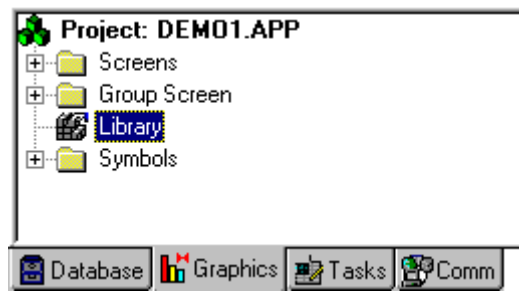


在啓動 *Studio* 之後, 預設的用戶(**Guest**) 便會登入. 如果沒有用戶登入或是目前的用戶已經登出, 則 **Guest** 用戶會自動的登入. 連同 **Guest** 用戶, 還有 **Guest** 群組, 其預設的權限可以使用所有的 **tasks**. 建議使用者編輯 **Guest** 群組並削減其權限至啓動程序應有的權限.

請依下列步驟設置用戶:

- 從 *User Account* 下拉式選單選擇要設置的用戶.
- 按下 **Password** 鈕來定義用戶的通行密碼. 在 *User Password* 視窗輸入想要的密碼並確認之.
- 從 *Security Group* 下拉式選單為用戶指定一個群組.
- 按下 **New** 鈕可以創造一個新的用戶. 將顯示 *New User Account* 視窗.
- 如要刪除一個用戶, 從 *User* 下拉式選單選擇一個用戶並按 **Delete**.

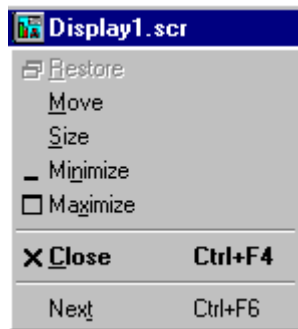
II.5.2. Graphics 標籤



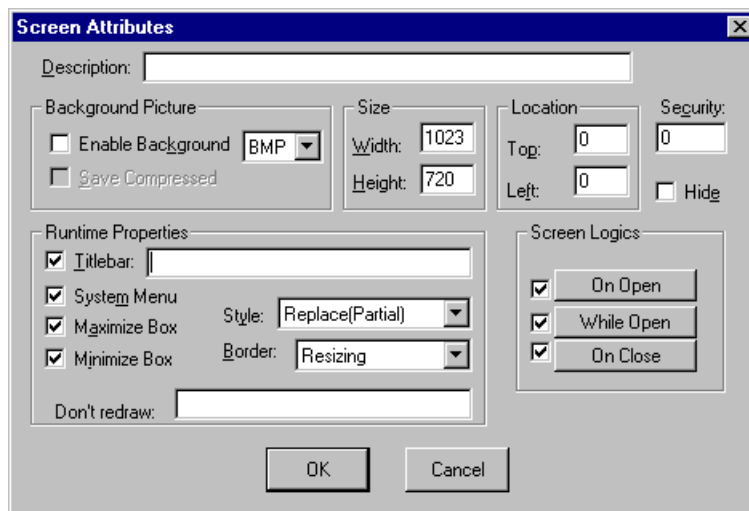
Graphics 標籤(Tab)擁有下列的檔案夾:

- **Screens 檔案夾**

這個檔案夾包含有畫面的細部設定以及畫面的工作草圖. 當開啓時, screen 會出現在 *Workspace* 視窗的右方. 要顯示已存在的 screens, 使用者可以展開 *Screens* 檔案夾並在想展示的 screen 上雙擊(double-click)滑鼠左鍵. 在 display screen 標題列上的 Screen 按鈕上按下左鍵則會出現一個下拉式視窗, 如下所示:



要創造一個新的視窗畫面, 在 *Screens* 檔案夾上按下滑鼠右鍵並選擇 insert a new display screen. 按下後出現會 *Screen Attributes* 視窗. 或者, 從 *File* 選單上選擇 *New*, 在 *Standard* 工具列按下 *New* 按鈕或是從 *Insert* 選單選擇 *Screen* 選項. 這些方法將開啓 *New Document* 視窗. 選擇 *Display* 並按下 *OK* 按鈕. *Screen Attributes* 視窗展示如下.



這個視窗選項功能以及欄位說明如下：

- *Description* - 這個欄位用來做為描述畫面的提示說明. 所輸入的文字將被顯示在 *Run Application* 模式下 *viewing screen* 右下角的狀態列上(當成預設值).
- 請依下列描述設置 *Background Picture* 群組對話盒:
 - *Enable background* 查核方框 – 允許使用點陣圖檔當作背景.
 - *Enable Background* 下拉式選單 – 讓使用者設定 *enable* 或 *disable* 在 WinCE 環境中將點陣圖檔當作背景圖. 預設值是 *disabled*. 除了 *BMP*, *background* 的選擇外還包括了 *TIF*, *DXF*, *EPS*, *WMF*, *IMG*, *JPG*, *WPG*, *PCD*, *PNG*, *FMF*, *FPX*, *FAX*, 和 *TGA*.
 - *Save Compressed* 查核方框 - 這個選項選擇是否用壓縮的形式儲存 *.BMP* 檔案.

⚠警告:如果 Windows 設定值被更改了或是被安裝在有不同色彩數的環境, 使用者將無法讀取以壓縮方式儲存的 *.BMP* 檔案. 在此建議使用者以非壓縮式儲存 *screens*, 以便萬一使用者想更改設定時用. 在 *CE* 中, 點陣圖檔必須是 *16-color*.

- *Size* 群組對話方框 – 在 *Width* 或 *Height* 對話方框中輸入一整數以定義視窗圖像素的大小.
- *Location* 群組對話方框 – 在 *Top* 或 *Left* 對話方框中輸入一整數以定義視窗圖像素單位下的位置.

⚠警告: 如果使用者直接用滑鼠變更視窗大小, 並在 *View* 選單上選擇 *Screen Attributes*, 使用者要立即更新目前的 *screen* 大小和位置.

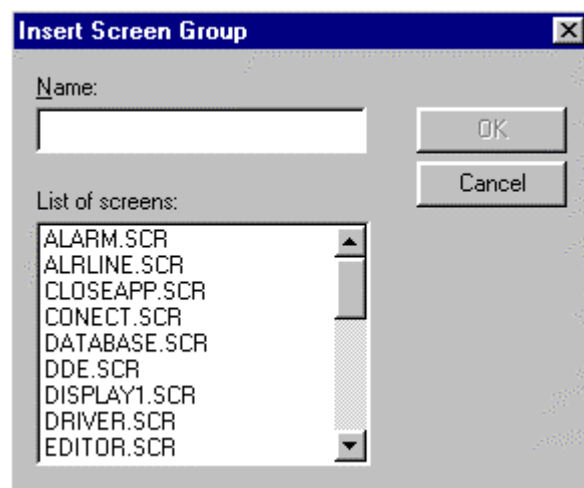
- *Security* 欄位 – 在 *Database* 標籤(Tab)裡的 *Security* 上設定視窗的 *security level*. *T* 預設值為 *0 (zero)*.
- *Hide* 查核方框 – 設定 *screen* 在第一次被呼叫後便常駐在記憶體中. 這個功能讓使用者開啓 *screen* 時可以快速的載入. *Screen Logics* 則會被正常的執行. 這個功能會耗費大量的 *GDI* 資源. 在開發過程中, 使用者應藉由呼叫 *InfoResources* 函數來監控這些資源. 這個欄位的預設值是 *disabled*.
- 您可以規劃在 *Runtime Properties* 群組對話方框的選項, 選項如下. 當正在執行 *Run Application* 時, 使用者可以使用這個群組對話方框來定義視窗的屬性.
 - *Titlebar* 欄位 – 輸入在 *Run Application mode* 時在 *viewing screen* 標題列的名字. 查核方框用來啓動或是不啓動標題列.
 - *System* 選單 – 啓動系統選單.
 - *Minimize* 查核方框 - 啓動或是不啓動 *Minimize* 按鈕.
 - *Maximize* 查核方框 - 啓動或是不啓動 *Maximize* 按鈕.
 - *Style* – 定義視窗樣式. 預設視窗是 *Replace* 樣式. 各樣式如下:
 - Overlapped* -視窗開啓時不關閉任何視窗, 新視窗以重疊的方式呈現.
 - Popup* -視窗開啓在其他視窗之上. 其他視窗還是 *enabled* 的.
 - Dialog* -視窗開啓在其他視窗之上. 在被開啓的視窗被關閉前, 其他視窗是 *disabled* 的.
 - Replace* - 視窗開啓時其他 *Replace* 和 *Popup* 樣式的視窗將被關閉.
 - *Border* –定義視窗的邊框. 要選擇邊框樣式在想要使用的選項上按滑鼠左鍵. 預設邊框樣式是 *Resizing*. 各邊框如下:
 - None* –無邊框. 此設定在視窗中便沒有標題列也不能調整大小.
 - Thin* –薄邊框視窗. 此設定在執行時期便不能調整視窗大小.
 - Resizing* – 一般的編框. 在執行時期能調整視窗大小.
 - *Don't Redraw* 欄位 – 接受使用資料點(Tag) 或值用來更新 *screen* 的動態設定. 當這個值比 *0* 大時, 所有 *screen* 的動態設定是被 *disabled* 的.

- **Screen Logics** 群組對話方框讓使用者可以靠下列的事件執行數學運算：On Open, While Open, On Close。使用者 選擇一個事件之後，按下 corresponding 按鈕。將開啓一個視窗，讓使用者輸入下列的訊息：
 - **Tag Name** – 鍵入資料點(Tag), 用來接收 **Expression** 欄位的回傳值。
 - **Expression** – 可以輸入所要運算的數學式或方程式。回傳值將傳遞給 **Tag Name** 欄位。
 - **Trigger** – 只有 **While Open** 視窗有此選項。這個選項可使資料點(Tag) 扮演 **trigger (any value change)**的角色來觸發工作表單(worksheet)中所要執行的工作。當這個 欄位空白時系統將以最快的間格速度執行工作表單 (worksheet)。

▪ **Group Screen 檔案夾**


這個檔案夾結合 **Screens** 檔案夾中所製作的個別畫面成爲更具管理功能的群組。然而，這個功能只能在 **NT** 上使用。不能在 **CE** 使用。要開啓特定的 **screen** 群組，在 **Group Screen** 檔案夾中的次檔案夾上按右鍵。要刪除特定的 **screen** 群組，在次檔案夾上按右鍵。按下 **delete** 刪除。




要創造一個新的 **screen** 群組，在 **Group Screen** 檔案夾上按右鍵來插入一個新的 **screen** 群組。按下該功能出現 **Insert Screen Group** 視窗或是在 **Insert** 選單下選擇 **Screen Group** 選項。



要創造 **screen** 群組，在 **Name** 欄位輸入檔案夾名稱，該檔案夾將包含被選的 **screens** 群組。從 **List of screens** 列表，選擇使用者想要加入的 **screens** 群組。按住 **Ctrl** 鍵，可以複選。這個 **screens** 列表包含儲存在 **Screens** 檔案夾中的 **screens**。

▪ **Web 檔案夾**

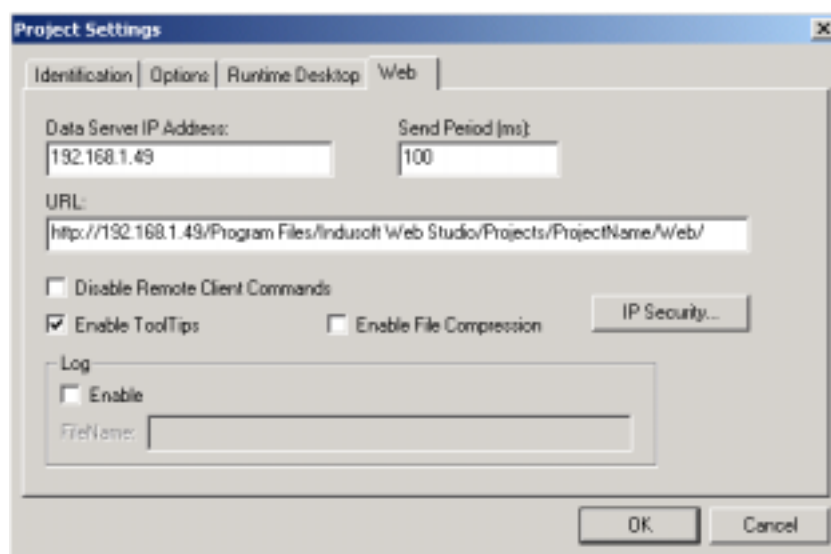
使用者也可以用 **HTML** 格式儲存 **screens**。在 **Web Pages** 檔案夾中的 **HTML pages** 並不是直接創造的，而是由事前存在的畫面而轉出來的。要創造一個 **HTML page**，使用者必須先創造一個畫面。先規劃 **screen**，創造畫面中的物件，增加屬性等等，但要朝著這個 **screen** 將成爲網頁而設計。一旦 **screen** 設計完成並存檔。最後，在這 **screen** 還開啓時，展開 **File** 選單 並選擇 **Save As HTML** .


 **警告:** 由  **Save As HTML** 所產生的網頁與產生它的 **screen file** 不會相互影響. 如此一來, 如果使用者對 **display screen** 作了一些改變, 這些改變並不會出現在網頁上, 除非使用者再次執行  **Save As HTML**.

要查看使用者的網頁, 首先使用者必須先設置這個網路設定. 您可以在 **Web** 標籤 (Tab) 中的 **Project Settings** 視窗中設定. 首先, 在 **Data Server IP Address** 欄位中輸入將來要執行此應用程式平台的 IP address. 接下來, 在 **URL** 欄位以下列格式輸入 URL :

`http://<the IP address of the unit where the web server is running>/<path from the server to the web page directory>/`

這兩個欄位填妥後, 按下 **OK** 按鈕. 從選單選擇 **Tools->Verify** 來確認此應用程式的執行狀況(執行前請先將在開發系統中被開啓任何視窗關閉).

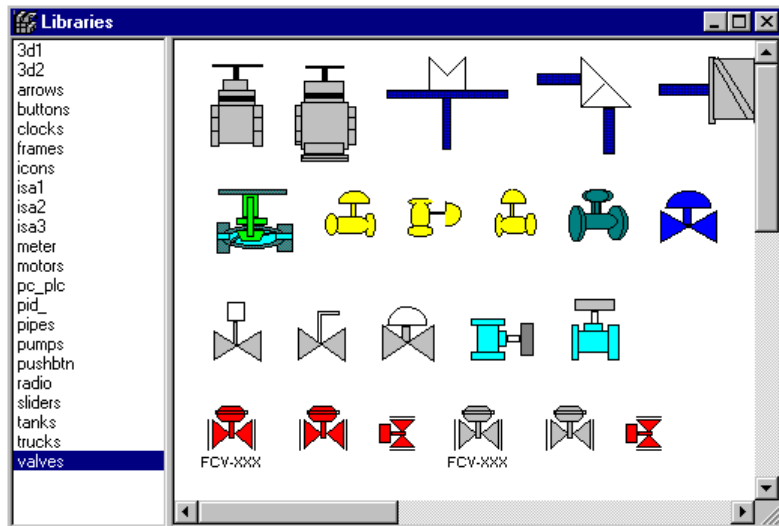


 **注意:** 如果使用者在 **Project Settings** 視窗改變了任何網頁, 使用者必須重新確認 (作一次 **verify**) 此應用程式新的設定才可以發揮作用. 因為網頁是從應用程式經過 **web** 伺服器來顯示訊息. 所以 **Runtime system**, **web server** 以及 **TCP/IP server** 必須執行才能看到網頁.

- **Library 檔案夾**

這是一個 **Studio** 提供的圖庫. 圖庫是由一組常用的圖示所組成, 由圖形所代表的意義來區別群組, 並儲存於特定的路徑中.

要開啓 **Libraries** 視窗, 雙擊(double-click)在 **Graphics tab** 上的 **Library** 或是在工具列或按下 **View** 選單中的 **Library** .




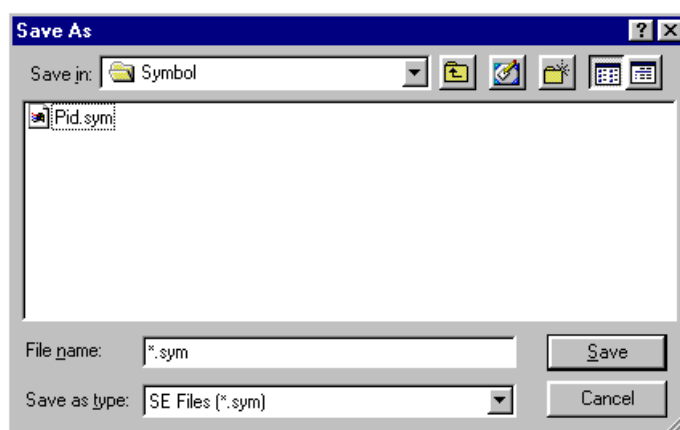
從 **Library** 左邊視窗選擇一個圖形群組來預覽可用的影像。要匯入一個影像到所繪製的圖形(display screen)中, 雙擊(double-click)使用者所選擇的影像。這樣使用者選擇的影像將保留下來並關閉 *Libraries* 視窗。點放在繪圖工作草圖上的任何地方來放置所選的影像。

注意: 大部分圖庫中的圖示有其事先定義的屬性。要改變這個屬性, 可以用 *Object Properties* 視窗上的 *Replace* 標籤(Tab)來改變。使用者也可以將用戶設計的圖形(screen)加入 *Symbol library* 中。製作一個 .scr 的圖形(screen)然後將其拷貝至 *Studio* 安裝目錄下 \LIB 的路徑。

▪ Symbols 檔案夾

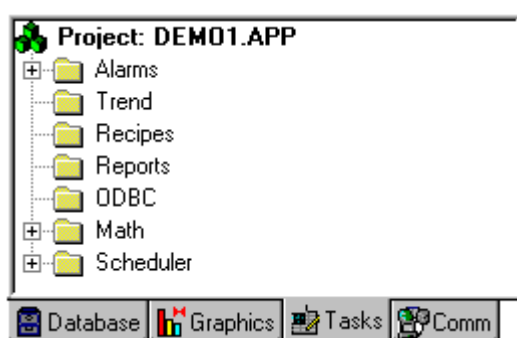
這是使用者定義圖像的收集目錄。這些符號是被群組化的影像或是文字。符號由在

Object Editing 工具列上的 **Group**  按鈕來創造。使用者可以在繪圖草圖上 (display screen)創造自訂的圖像並將它們儲存在這個檔案夾。選擇一個符號然後點選在 *Edit* 選單下的 *Copy to*。則會出現一個 *Save As* 視窗來存入到 *Symbol* 檔案夾。



要從 *Symbol* 檔案夾使用一個圖像時, 從 *Edit* 選單選擇 *Paste from*。則圖像將自動的被匯入到繪圖(screen)的草圖中。

II.5.3. Tasks Tab



Tasks tab 包含有下列的檔案夾:

- **Alarms 檔案夾**

這個檔案夾讓使用者可以規劃 alarm 群組以及與群組相關聯的資料點(Tags). Alarm task 定義 Studio 會產生的警報訊息. 警報的主要目的是告知操作員在生產過程中產生的任何的問題或是狀態的變化, 讓改善的措施可以被執行. 在檔案夾上按右鍵可以插入警報工作表單(Worksheet).

要讓警報訊息在螢幕上顯示, 使用者必須按下在 *Object Editing* 工具列上的 Alarm



按鈕創造一個警報物件.

注意:用來作識別用警報工作表單(Worksheet)的編號會隨著新增的工作表單(worksheet) 而循序增加.

要新增一個警報時, 在 *Alarm* 檔案夾上按右鍵. 按下相對的命令來開啓一個警報工作表單(worksheet). 或者, 從 *File* 選單選擇 *New* 或者在 *Standard* 工具列上的 *New*



按鈕按滑鼠左鍵. 這兩個方式會開啓 *New Document* 視窗. 選 *Alarm Worksheet* 並按 *OK* 按鈕. 一個工作表單(worksheet)將會產生如下:

	Tag Name	Type	Limit	Message
1				
2				
3				

Alarm worksheet 分成兩個部分：包含整個群組的資訊的主標題，以及定義每個群組的資料點(tag)的主體。每個工作表單(worksheet)上的選項描述如下：

- **Group Name** 欄位 -用來辨識 alarm 群組的名字。

⚠警告:改變 **Group Name** 欄位之前, 請先儲存 **alarm worksheet**, 因為在未儲存的 **worksheet** 上的 **alarm** 設定會有遺失的可能。

- **Description** 欄位 – 爲了文書化可在此輸入警報群組的註解。
- **Disable** 欄位 - **Disables** 在群組裡的所有警報。使用者必須在這個欄位填入一個資料點(Tag)來作控制。如果這個資料點(Tag)的值比 zero 大, 此群組是被 **disabled** 的, 且警報訊息不會產生。如欄位爲空白, 此群組將被 **enabled**。
- **Remote Ack** 欄位 – **Alarm** 已被確認的資料點(Tag)。當這個資料點(Tag)的值改變時即表示警報已被確認。
- **Total Active** 欄位 – 保持目前所有在群組中發生的警報數目。當其中的一個資料點(Tags)改變其狀況時, 系統會不斷的更新這個值。
- **Total Active or Unack** 欄位 –保持目前所有在群組中發生的警報或是還未被確認的警報數。當其中的一個資料點(Tags)改變其狀況時, 系統會不斷的更新這個值。
- 請依下列敘述設置群組 對話方框中選項如下:
 - **Summary** 查核方框 – 當此選項被選取, 即可在螢幕上接收警報訊息。

⚠警告: 如果使用者沒有選 **Summary** 選項, 這個群組的警報在執行時將不會出現在螢幕上並且也無法在執行期列印警報。

- **Ack** 查核方框 –警報可以被確認。只有在 **Summary** 欄位被 **enabled** 時才可以使用。
 - **Beep** 查核方框 – 產生嗶嗶聲音直到警報被確認。只有在 **Ack** 以及 **Summary** 欄位被 **enabled** 時可以使用。
 - **Printer** 查核方框 – 列印群組的每個警報訊息。這個 選項只能使用於點陣印表機 (或是其他以列爲單位列印的印表機)。
 - **Disk** 查核方框 – 存儲群組的每個警報訊息到硬碟中。使用者必須選擇這個選項如果使用者想要有歷史的 **alarm objects**。
 - **Generate Ack Messages** 查核方框 -當這個群組的警報被確時產生訊息。只有在 **Disk or Printer** 欄位是 **enabled** 才可以使用。
 - **Generate Norm Message** 查核方框 -當這個群組的警報回到其正常狀態時產生訊息。只有在 **Disk or Printer** 欄位是 **enabled** 才可以使用。
- 規劃 **Colors** 群組對話盒中的選項。這個群組對話方框將定義警報物件 (**alarm object**)中警報訊息顯示的顏色。每一個警報訊息將依群組定義的顏色顯示在警報物件(**alarm object**)中：
 - **Default Radio Button**
 - **Custom Radio Button**
 - **Start Color Rectangle**
 - **Ack Color Rectangle**
 - **Norm Color Rectangle**

按下顏色設定方框則顯示 **Color Selection** 視窗。在想要的顏色上雙擊(double-click)或者選取其顏色後按 **OK**。



- *Alarm worksheet* 的主體可定義警報群組中的資料點(Tag), 設定其警報狀態與訊息. 它有很多欄位 (在上面的 *Alarm worksheet* 例子中僅顯示出四欄).

Worksheet 主體的選項敘述如下：

- *Tag Name* 欄位 - Alarm 群組監控的資料點(Tag).
- *Type* 下拉式選單 - alarm 的型態敘述如下. 使用者可以在執行時期中改變任何的欄位. 如要更多的訊息, 詳見 *Application Tags*.
 - HiHi** – 警報上上限, 當資料點(Tag)的值大於或等於此值便產生警報訊息.
 - Hi** – 上限, 當資料點(Tag)的值大於或等於此值便產生警報訊息.
 - Lo** – 下限, 當資料點(Tag)的值小於或等於此值便產生警報訊息.
 - LoLo** – 下下限, 當資料點(Tag)的值小於或等於此值便產生警報訊息.
 - Rate** – 決定這個資料點(Tag)變化的速度. 如果變化的速度比這個欄位的速度值快則會產生警報. 這個速度可以以每秒, 分鐘或小時來計數.
 - Deviation +** - 較大值的偏移, 當資料點(Tag)的增加大於或等於此值時產生警報.
 - Deviation -** - 較小值的偏移, 當資料點(Tag)的減少大於或等於此值時產生警報.
- *Limit* 欄位 - 為 alarm 的產生限制的值.
- *Message* 欄位 – 顯示的警報訊息.

⚠警告：警報訊息可以用下列語法來表示 `tag message{tag_name}`, `tag_name` 可以是資料點及資料點的所有屬性.

- *Priority* 欄位 – 指示這群組中的優先權. 這個欄位可以填入整數(0 to 255). 有較大值則資料點(Tag)有較高的優先權.
- *Selection* 欄位 – 一個使用者定義的字串, 扮演 *alarm summary objects* 中作警報過濾的角色.

⚠警告: *Selection* 欄位最多只能輸入 7 個字元的字串 (更多的字元是無效的).

- *Alarm history* - alarm 歷史檔案. 當某個群組的這項功能被啟動時, 將會以以下列格式被儲存：

```
Alarm Summary
P1|P2|P3|P4|P5|P6|P7|P8|P9|P10|P11|P12|P13|P14|P15|P16|P17|
P1|P2|P3|P4|P5|P6|P7|P8|P9|P10|P11|P12|P13|P14|P15|P16|P17|
.
P1|P2|P3|P4|P5|P6|P7|P8|P9|P10|P11|P12|P13|P14|P15|P16|P17|
```

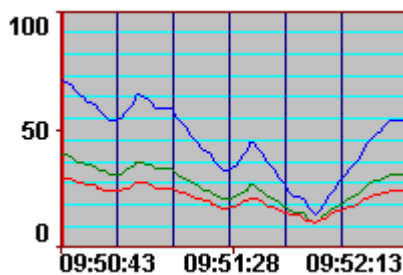
其中各值的意義如下：


- P1 = 檔案版本 (預設值: 000)
- P2 = 開始日期(DD/MM/YYYY)
- P3 = 開始時間(HH:MM:SS)
- P4 = 資料點(Tag)名稱
- P5 = 警報訊息
- P6 = Ack 其中 0 表示這個警報已被確認或不需要被確認, 其中 1 表示警報未被確認.
- P7 = Active, 其中 0 表示這個警報不是 active, 1 表示這警報是 active
- P8 = Limit Value
- P9 = 警報群組數
- P10 = 優先權數字
- P11 = 選擇
- P12 = 其中 1 代表 HiHi, 2 代表 Hi(On), 4 代表 Lo(Off), 8 代表 LoLo, 16 代表 Rate(Change), 32 代表 Deviation+, 64 代表 Deviation-
- P13 = AckReq, 其中 0 需要作確認, 1 不需要作確認
- P14 = 日期(DD/MM/YYYY)
- P15 = 時間(HH:MM:SS)
- P16 = 確認日期(DD/MM/YYYY)
- P17 = 確認時間(HH:MM:SS)

這個檔案儲存在應用程式目錄下的 **\ALARM** 路徑, 其名稱如下:
\app\ALARM\ALyymmdd.ALH, 其中 **yymmdd** 指的是 **alarm** 檔案被創造的年月日.


▪ **Trend 檔案夾**

這個檔案夾讓使用者可以設置 **history** 群組, 可以用來儲存歷史曲線. **Trend task** 讓使用者宣告哪一個 **tags** 要被作成 **trend**. 使用者可以在歷史檔案中儲存取樣點, 並且將即時以及歷史的取樣點展示出來. 要在螢幕上顯示 **trend graph**, 使用者可以在 **Object Editing** 工具列上的 **Trend**  按鈕產生 **trend object**. 在檔案夾上按右鍵可以插入 **trend graph** 工作表單.



 **注意:** 用來作為識別 **Trend worksheet** 的編號會隨著新增的工作表單而循序增加。使用 **4 bytes** 來儲存日期與時間的資訊；使用 **8 bytes** 來儲存各個取樣的變數。

其他的資訊, 請參考 **Converting Trend History Files**.

在 **Trend** 檔案夾上按右鍵來插入工作表單. 按下提示命令來開啓一個 **Trend** 的工作表單. 或者, 從 **File** 選單選擇 **New** 或者在 **Standard** 工具列上的 **New**  按鈕按滑

鼠左鍵. 這兩個方式會開啓 *New Document* 視窗. 選 *Trend Worksheet* 並按 **OK** 按鈕. 工作表單將會產生如下.

	Tag Name	Dead Band
1		
2		
3		
4		
5		

Trend 工作表單分成兩個部分：包含整個群組的資訊的主標提, 以及定義每個群組的資料點(tag)的主體. 每個工作表單(worksheet)上的選項描述如下：

- **Description** 欄位 -爲了文書化可在此輸入警報群組的註解.
- **Disable** 欄位 - 使用者可以在這個欄位填入一個資料點(Tag), 當這值大於 0 時則可暫時的 **disable** 警報的搜集.
- **File Life Time (days)** 欄位 – 決定這個歷史檔案要在磁碟中保存幾天. 決定了保存期間之後, 這個檔案將自動的移除. 這個選項只能用在以日期爲檔名的檔案.
- **Save on Trigger** 查核方框以及欄位 –當某個資料點(Tag)改變, 便會儲存 **trend** 取樣點. 這個資料點(Tag)的改變可以由 **Scheduler** 產生的事件控制.
- **Save on Tag Change** 欄位 – 當群組的任何資料點(Tag)改變其值, 則會儲存 **trend** 取樣點.
- **Name of History Files** 群組對話方框– 定義歷史檔案的名稱. **Trend** 歷史的檔案有兩種形式: 由日期或是由批次 (由事件). 對話方框的選項描述如下:
 - **Date (預設)** 查核方框 – 當選擇時, 會產生以日期爲檔名的歷史檔案. 如果使用者有連續性的生產流程則可以使用這個選項. 以這個例子, 產生的檔案將被儲存在 `\app\HST\gggyymmdd.HST` 其中:
 - `app` = 應用程式路徑
 - `gg` = 歷史群組編號(十六進位)
 - `yyyy` = 年
 - `mm` = 月
 - `dd` = 日
 - **Batch** 查核方框及欄位–當選擇, 產生的歷史檔案檔名, 是使用 **edition** 欄位中的名稱.這個欄位可以是資料點(Tag)值. 如果使用者是批次的生產流程可以使用這個選項. **Example: c:\history\file{TagBatchNumber}.hst**

注意: 要查看線上的曲線, 其 **files based on date** 要被 **enabled**. 對歷史曲線來說, 使用者 兩者都可以使用.

工作表單主體的選項敘述如下：

- **Tag Name** 欄位 – 要儲存在歷史檔案中的料資點(Tag).

警告: 每一個 Trend 群組最多可以擁有 100 資料點(Tags). 在這邊將建議使用者將其分為更多的群組.

- Dead Band -當選擇 Save on Tag Change 時在儲存值會過濾可接受的變化率. 例子: Dead Band =5, 如果資料值是由 50 變成 52, 因為變化量比 5 少, 系統將不會在資料庫中記錄這種變化. 如果變化相等或大於 5, 新的值將會在趨勢圖中被顯示。

▪ The Recipes 檔案夾

這個資料夾使你能夠編輯配方工作表單, 可以用在應用程式資料庫和磁碟檔之間以 ASCII 或 DBF 格式交換資料; 它在檔案和即時的記憶值之間作傳輸. 典型的使用是儲存程序控制的配方, 但是這些檔案能儲存任何類型的資料, 例如: 操作的紀錄, 密碼, 等等. 那配方的 task 可從檔案中讀寫資料點(Tag)的值, 這個模組可將資料點(Tag)的值從應用程式傳送到檔案或從檔案傳到應用程式.

備註: 用來作為識別 Recipe 工作表單的編號會隨著新增的工作表單而循序增加.

在配方資料夾上按滑鼠右鍵以插入一張新的工作表單. 按提示開啓一張配方工作表單, 或從檔案目錄中選擇 **New** 或在標準的工具列上按'New'的按鈕. 以這二個方法打開新的文件視窗. 挑選配方工作表單和按 **OK**. 一張新的工作表單將被顯示.

	Tag Name	Number
1		
2		
3		
4		
5		

配方工作表單被區分為二個部分: 包含整個群組的資訊的主標提, 以及定義每個群組的資料點(tag)的主體.


每個工作表單(worksheet)上的選項描述如下:

- Description - 輸入文字描述
- Save As XML - 如被選擇, 表示資料將會是以 XML 格式儲存。如未被選擇, 資料將會是以標準的 DAT 格式儲存


警告: 當資料以 .DAT 檔案格式儲存的時候, 能被另一個配方工作表單載入使用於不同的資料點(Tag); 資料在以 .XML 檔案格式儲存的時候, 只能被載入於相同名稱及格式的資料點(Tag).

如同 HTML 網頁, 必需執行 web server 以便能在網頁上看到 XML 資料, 在 runtime 環境時則與 HTML 網頁不同, 則不需要執行 web server 即能看到 XML 資料(僅有在 Internet Explorer versions 5.0 或更的版本才可看到 XML 資料).

- **File Name field** – 配方群組的檔案名稱. 檔案名能是靜態文字 (例如 File1) 或動態的資料點(Tag)的值(例如{FileNameTag}).
- **Register Number field** - 暫存器數, 定義在 DBF 檔案中被讀或寫之暫存器數的資料點(Tag).

 **備註:** 當你儲存工作表單的時候, 會提示您為表單命名(它沒有預定名稱). 組態檔內定的延申檔名為 .RCP(如果當選擇儲存為 XML 格式時則為 .XSL) 它包含配方組態, File Name 欄位包含將被讀寫資料的檔案名稱.


- **Tag Name field** – 此資料點(Tag)用來將檔案內容讀入或將資料點(Tag)的值寫到檔案中. 如果資料點(Tag)是陣列, 你一定要設定在被使用的陣列的第一個位置.
- **Number field** – 鍵入多少陣列的位置被使用.

 **警告:** 當一個陣列資料點(Tag)被定義的時候, 它的起始位置是 0, 它被系統所用, 以防有不合法的位置組態時. 無論如何避免使用 0(零) 的位置.


當讀或寫一個配方群組時, 可使用函數功能。

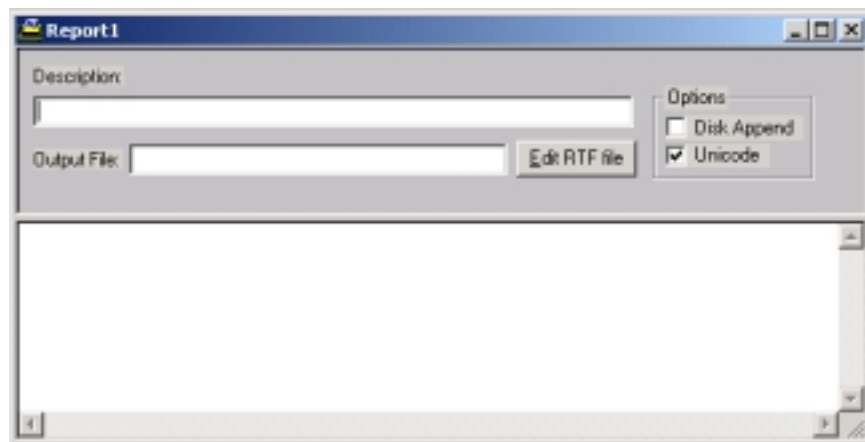
▪ The Reports Folder

這個資料夾包含能被送到印表機或磁碟的報表 (文字類型) 的定義. Report Task 允許你用來自系統的資料來規劃你的 report (文字類型). 此模組的主要目的要讓報表的製造更容易和更有效率.

 **備註:** report 的工作表單編號是隨著新工作表單的增加而循序遞增.

在任何情形下均可使用函數來印出報表.


在報表資料夾上按滑鼠右鍵來插入一張新的工作表單- 按提示列打開一張報表工作表單, 或從檔案目錄中選擇 **New**  或在標準的工具列上按 New 的按鈕. 以這二個方法打開新的文件視窗. 挑選報表工作表單按 **OK**. 一張新的工作表單將被顯示.



報表工作表單被區分為二個部分: 包含整個群組的資訊的主標提, 以及定義每個群組的資料點(tag)的主體.


每個工作表單(worksheet)上的選項描述如下:

- **Description field** -輸入文字描述.
- **Disk Append checkbox** - 當列印資料到檔案, 若有選擇此項, 則新的報告將為增加在現存報告之後. 如果不選擇, 新的報告將會在檔案中替換原先報告.
- **Unicode checkbox** -如果選擇, 報告是以 UNICODE 的格式儲存. 否則, 它是以 ANSI 的格式儲存.
- **Output File field** - 當列印到檔案的時候, 這裡所鍵入的是輸出檔案的名字. 若輸出的名字遵循造句法 {資料點(Tag)}, 資料點的值將是檔案名字的一部份. 在原先例子中: 'report{day}.out', 被產生的檔案可能 report1.out, report2.out... 檔名是依照 day 這個資料點(Tag)的值而產生.

 **備註:** 報表的規劃檔內定的延申檔名為 .REP. Output File 欄位中則是定義儲存資料檔案的名稱.

- **Edit RTF file button** -允許你存取及編輯 RTF 型態的報表,例如對背景的佈置或字體的修正等.

工作表單的主體則保留給格式化的報表. 你能在系統中用資料配置你的自己報告, 排列資料點(Tag)列印的位置. 每個資料點(Tag)的名字將會替換為:{tag_name}. 如果資料點(Tag)是實數的類型, 使用下列語法:{tag_name n} n 是你想要列印的位數.


 **備註:** 當使用標準的報告編輯程式時, 保留給資料點(Tag)值的字元位數相等於鍵入資料點(Tag)名字字元的位數(包括 2 個大括弧). 舉例來說, 配置 {TagA} 在報告主體上, 他們將會保留 6 個字元作為資料點(Tag)在報告檔案上的值. 但不適用於 RTF 格式的報告.

■ The ODBC Folder

ODBC 界面使用在網路環境中並且是使用視窗標準方式來規劃. ODBC Task 能在 Studio 與任何支援這個界面的資料庫之間交換資料. 在資料夾上按滑鼠右鍵來插入一張 ODBC 工作表單.

除了配置 ODBC 工作表單, 你也需要安裝視窗 ODBC 標準驅動程式. Studio 參照使用者的 DNS, 可在控制台中規劃. 如欲獲得更多的資訊,參照你的視窗文件.

 **備註:** ODBC 工作表單的編號是隨著新工作表單的增加而循序遞增.

在 ODBC 資料夾上按滑鼠右鍵來插入一張新的工作表單- 按提示列打開一張 ODBC 工作表單, 或從檔案目錄中選擇 **New**  或在標準的工具列上按 New 的按鈕. 以這二個方法打開新的文件視窗. 挑選 ODBC 工作表單和按 OK. 一張新的工作表單將被顯示.

	Tag Name	Column
1		
2		
3		
4		
5		

ODBC 工作表單被區分為二個部分: 包含整個群組的資訊的主標提, 以及定義每個群組的資料點(tag)的主體.

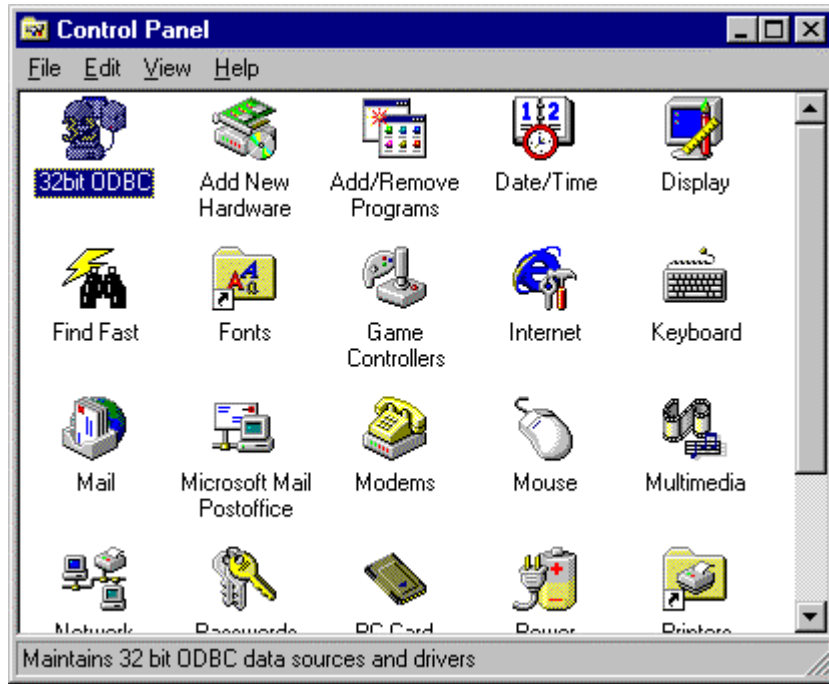
ODBC 工作表單的標頭檔讓你定義開始讀的資料點(Tag)以及寫入的事件控制點, 放置回傳值, 處理資料庫存取參數等等.

- **Description field** - 輸入文字描述.
- **Data Source Name field** - 資料庫來源名稱, 相關設定在控制台中的 ODBC 項目.
- **User field** - 鍵入具有資料庫存取權的使用者名字.
- **Password field** - 鍵入使用者的密碼.
- **Table field** - 在資料庫中的 Table 名字.
- **Condition field** - 搜尋條件或過濾條件.
- **Status field** - 回傳值(填入一個資料點(Tag)名字), 如果成功時資料點(Tag)標籤將為 0, 但如果失敗將回傳一個錯誤碼.
- **Transaction field** - 鍵入一個在執行時將會有值變化的資料點(Tag).
- **Select, Next, Insert, Delete, or Update Trigger fields** - 鍵入一個作為觸發用的資料點(Tag). 每次值變化即觸發系統執行指令, 至少需要鍵入其中一個 Trigger Field.

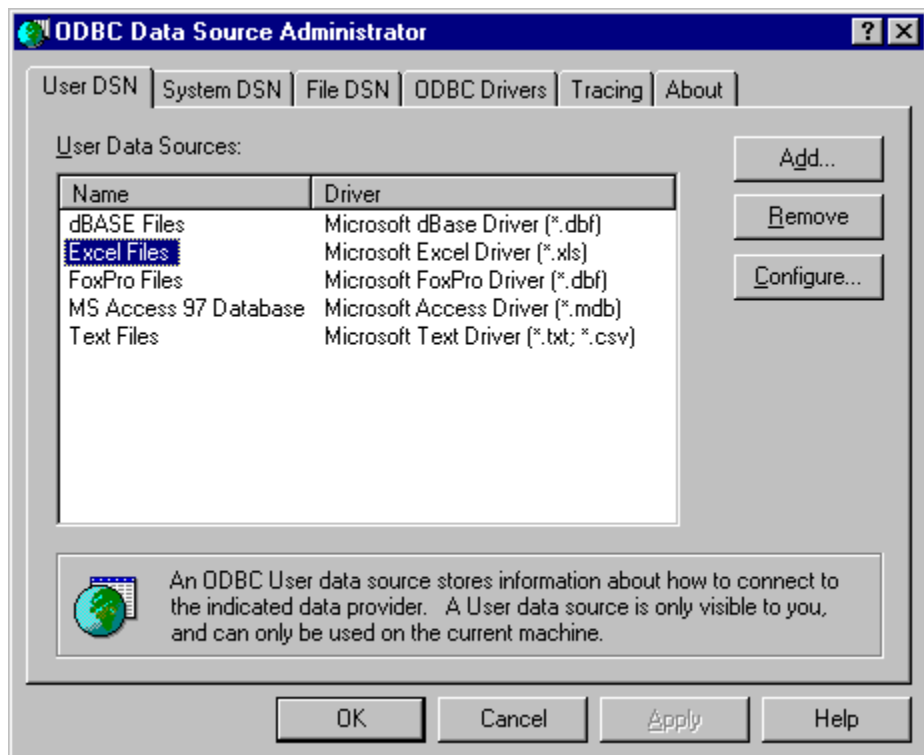
在 ODBC 工作表單主體中, 你可以在將現有的資料庫與資料點(Tag)作連結.

- **Tag Name field** - 被讀入或寫出的資料點(Tag).
- **Column** - 資料庫 Tab 中的欄位名稱. (例如 R3CH, Row 3, Column H 符合 Excel 的格式).

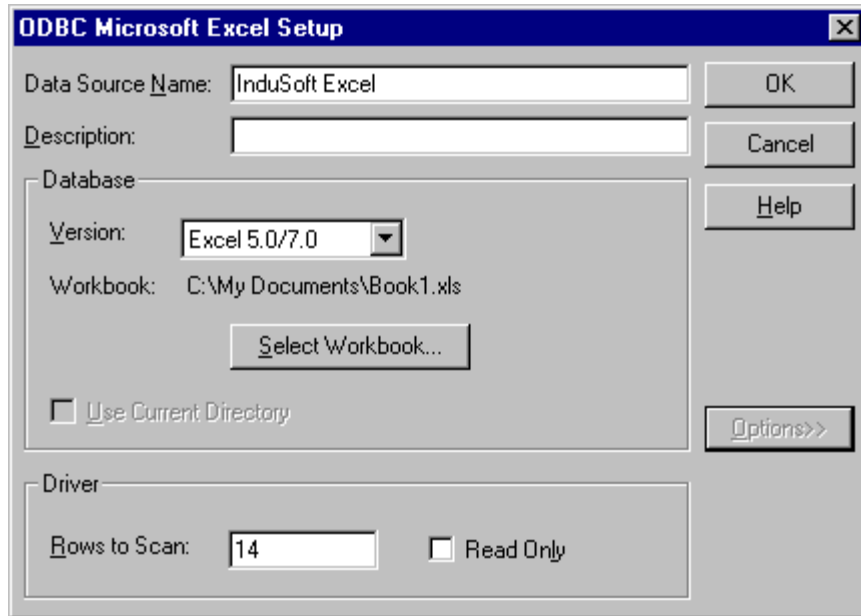
你必須在視窗控制台中建立 Excel 的 ODBC 界面.



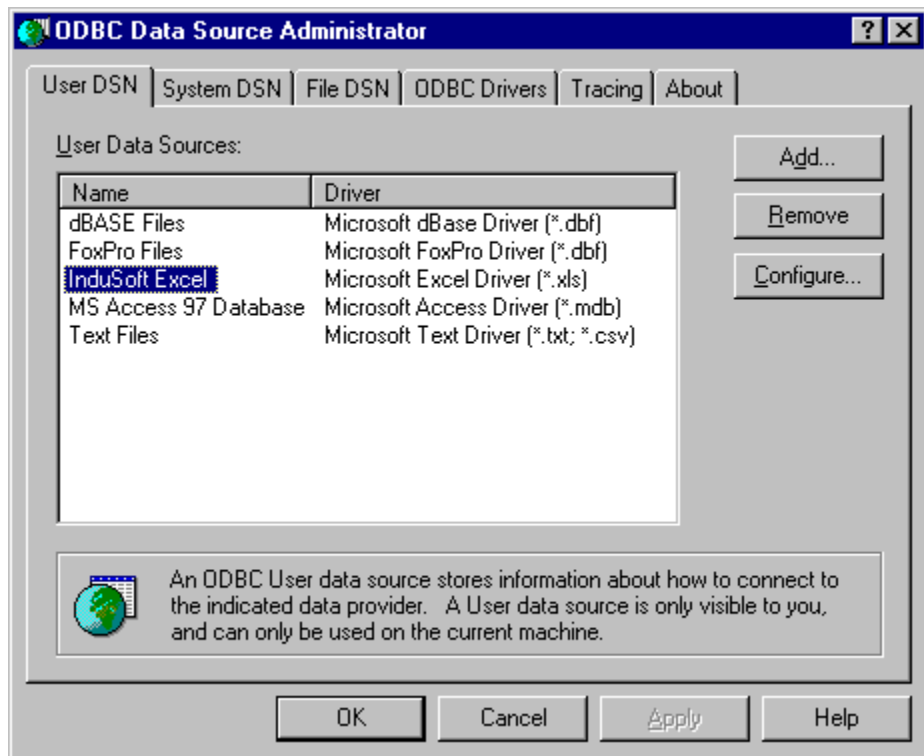
在控制台中的 ODBC 圖示上雙擊(double-click), 然後選擇 Excel 檔案。



點選 Configure 按鈕. 將顯示 ODBC 微軟 Excel 設定視窗。



在 Data Source Name 欄位中, 鍵入被用在 DSN 欄位中的 ODBC 工作表單的名字. 點選 **Select Workbook** 按鈕選擇你將會使用的 Excel 檔. 回到 'ODBC Data Source Administrator' 視窗; DSN 名稱將顯示在目錄中.



在你規劃 ODBC 視窗界面之後, 你必須先規劃 Studio ODBC 工作表單. 藉著 Task 標籤(Tab), 插入一張新的 ODBC 工作表單. 確認 ODBC Runtime 在 Project Menu 中的 Project Option 裡的 Runtime Tasks 已被設定. 你只需要執行這個 Project 來啟動這個設定. 應用程式將會處理所有資料交換的動作.

對於特定的錯誤碼的意義, 參照你的視窗文件. 下列各項是一連串的 Studio 錯誤碼:

Select command

- 1 - 錯誤在 ODBCPREPARE 功能中.
- 2 - 錯誤在 ODBCBINDCOL 功能中.
- 3 - 錯誤在 ODBCEXECUTE 功能中.
- 4 - 錯誤在 ODBCSETCH 功能中.

Next command

- 5 - 錯誤在 ODBCSETCH 功能中.

Insert command

- 6 - 錯誤在 ODBCPREPARE 功能中.
- 7 - 錯誤在 ODBCEXECUTE 功能中.
- 8 - 錯誤在 ODBCCommite 功能中.

Update command


- 9 - 錯誤在 ODBCPREPARE 功能中.
- 10 - 錯誤在 ODBCEXECUTE 功能中.
- 11 - 錯誤在 ODBCCommite 功能中.


Delete command

- 12 - 錯誤在 ODBCPREPARE 功能中.
- 13 - 錯誤在 ODBCEXECUTE 功能中.
- 14 - 錯誤在 ODBCCommite 功能中.

▪ The Math Folder

數學 Task 讓你能建立附加的運算式與 Studio Task 的基本功能一起工作. 數學工作表單是一群在 Runtime 期間的背景執行的程式. 數學工作表單提供自由的環境給一般邏輯常式和可能需要的數學運算. 要達成這些目的, Advantech Scripting 是非常簡單而容易使用的.

 **備註:** 數學工作表單的編號是隨著新工作表單的增加而循序遞增.

數學資料夾上按滑鼠右鍵來插入一張新的工作表單. 按提示列打開一張數學工作表單, 或從檔案目錄中選擇 New 或在標準的工具列上按 **New**  的按鈕. 以這二個方法打開新的文件視窗. 挑選數學工作表單和按 **OK**. 一張新的工作表單將被顯示.

The screenshot shows a window titled "Math002.mat". It contains two text input fields: "Description:" and "Execution:". Below these is a table with two columns: "Tag Name" and "Expression". The table has five rows, numbered 1 to 5 in the first column. The first four rows are empty, and the fifth row is partially visible. The table is currently empty of data.

數學工作表單被區分為二個部分: 包含整個群組的資訊的主標提, 以及定義每個群組的資料點(tag)的主體.

每個工作表單(worksheet)上的選項描述如下:

- *Description field* - 輸入文字描述.
- *Execution field* - 當工作表單執行的時候用來決定是否執行. 通常用單一資料點(Tag)值或常數值來表示.

警告: 只有當在實行欄位中的結果不是 0 的時候, 工作表單才被執行. 如果你想要工作表單總是被執行, 鍵入常數 1.

工作表單的主體用來定義程式列.

- *Tag Name field* - 用來接收計算結果的資料點(Tag).
- *Expression field* - 用來放置邏輯常式或數學計算式.

▪ The Scheduler Folder

Scheduler Task 可以根據時間來產生執行數學式的事件. 在資料夾上按滑鼠右鍵以插入一張 Scheduler 工作表單.

備註: Scheduler 工作表單的編號是隨著新工作表單的增加而循序遞增.

在 Scheduler 資料夾上按滑鼠右鍵來插入一張新的工作表單- 按提示列打開一張 Scheduler 工作表單, 或從檔案目錄中選擇 **New**  或在標準的工具列上按'New'的按鈕. 以這二個方法打開新的文件視窗. 挑選 Scheduler 工作表單和按 **OK**. 一張新的工作表單將被顯示.

	Event	Trigger	Time	Date	Tag	Expression	Disa
1							
2							
3							
4							
5							

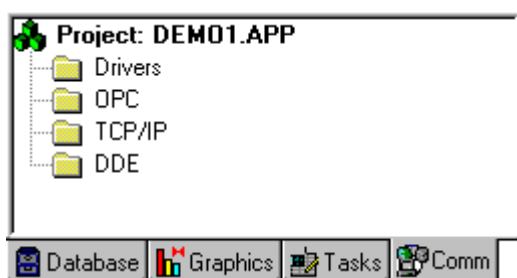
Scheduler 工作表單被區分為二個部分: 包含整個群組的資訊的主標提, 以及定義每個群組的資料點(tag)的主體. 每個工作表單(worksheet)上的選項描述如下:

- **Description field** - 輸入文字描述.
- **Event drop-down list** - 選擇事件的類型(行事曆, 時間, 變化).
 - **Calendar** - 能產生大於 24 小時的時間基礎. 例子: 你能定義一個每個星期五列印報告的事件.

備註: 當你想要事件在一個特定的日期實行的時候請填寫 Date 欄.

- **Clock** - 能產生小於 24 小時的時間基礎(間隔可能是分鐘或秒). 這個功能常應用於趨勢圖形. 例如: 定義一個資料點(Tag), 每個小時都加一次值.
 - **Change** - 在 Trigger 欄位中的資料點(Tag)變化時產生事件.
- **Trigger field** - 當資料點(Tag)的值變化的時候產生事件. 當資料點(Tag)的值變化發生的時候,其值被回傳到此資料點(Tag). 這個欄位只在變化事件中有用.
 - **Time field** - 設定發生事件的時間間隔: 小時, 分鐘, 和秒. 能被時間事件和日曆事件所使用.
 - **Date field** - 設定發生事件的特定日期: 天, 月, 和年. 如果是空白, 事件將每日發生. 這個欄位只被日曆事件所用.
 - **Tag field** - 在事件中收到新的回傳值.
 - **Expression Field** - 回傳值將寫入資料點(Tag). 這個欄位被所有的事件使用.
 - **Disable field** - 使此功能失去作用. 當它是空格或值等於零的時候, 功能將會被執行. 如果值是等於 1, 功能將不會執行(Disable = 1).

II.5.4. Comm Tab



Comm Tab 有下列各項資料夾:

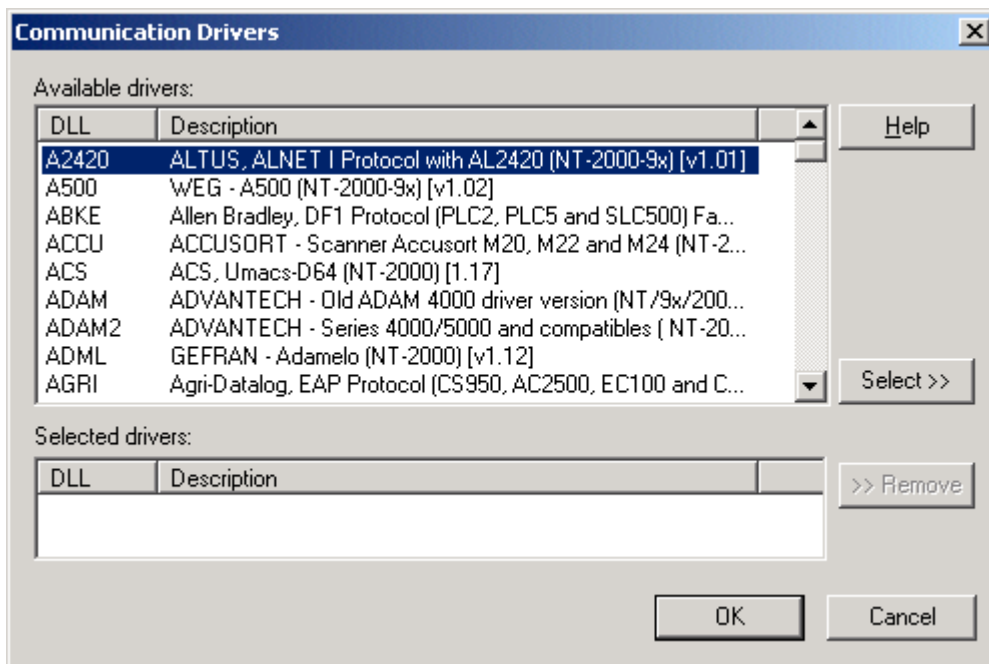
- **The Drivers Folder**

通訊的驅動程式是包含與特定的遠端儀器通訊且可以自訂通訊協定實現的 DLL 檔。若您要自行開發驅動程式, 要使用特定的發展工具, 請聯絡 Advantech 商討進一步的事宜。

在這個資料夾你可以定義 Project 中將使用的與遠端儀器的通訊界面, 例如 PLC , Single-loop, transmitter. 您可針對不同的驅動程式對設備定義不同的功能和屬性. 當發展一個應用程式的時候, 你應該參照每個驅動程式特定的文件, 這個文件通常位於 DRV 目錄之中。

規劃驅動程式時, 設定界面參數(舉例來說, 工作站位址和傳輸速率). 然後, 設定連接儀器的 I/O 位址到資料點(Tag).

在 driver 資料夾上按滑鼠右鍵增加或刪除一個被配置的 driver. 或在 Insert Manu 中選擇 Drivers. 兩者均可開啓一個 communication drivers 視窗及顯示一連串可選用的 driver.

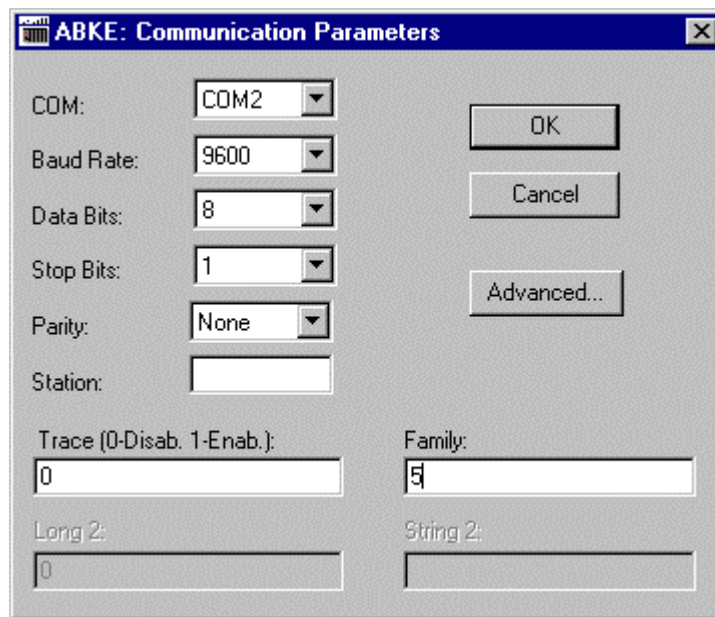


視窗中選項如下列各項:

- **Available Drivers field** –列出 driver 的名字和簡短的描述.

- **Select button** -若要選擇一個 driver, 直接在選單中選擇使之反白, 然後按這個按鈕.
- **Selected Drivers field** - 已選擇的 driver 連同描述一起在此被顯示.,
- **Remove button** -若要除去一個 driver, 從被選擇的 driver 領域反白它, 然後按這個按鈕.

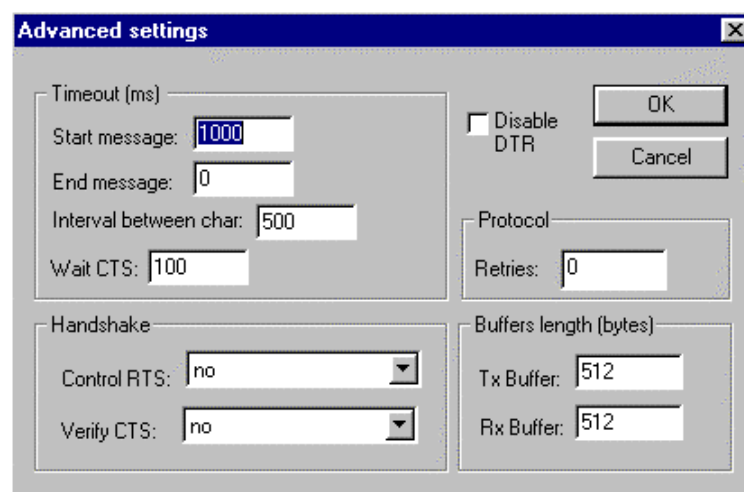
當你按 communication drivers 視窗上的 OK 時, 在 Comm Tab 上的 driver 資料夾中將新增被選擇的 driver 的一個子資料夾. 在被選擇的 driver 子資料夾上按滑鼠右鍵以即可顯示設定項目. 您將打開 communication Parameter 視窗.



視窗的選項如下:

- **COM field** - 序列通訊埠.
- **Baud Rate, Data Bits, Stop Bits, Parity fields** - 序列埠組態.
- **Station field** - 網路上工作站位址.
- **Long1, Long2, String1, and String2 fields** -這些欄位會自動地填上你所選擇的 communication driver 固定的資料格式.
- **Advanced button** - 打開 Advanced Setting 視窗, 在這裡你能改變內定的 driver 參數.

要規劃內定的 driver 參數, 顯示 Advanced Setting 視窗如下:



視窗選項如下:

- 群體盒子
 - *Start Message field* - 設定訊息開始逾時時間.
 - *End Message field* - 設定訊息結束逾時時間.
 - *Interval between char field*- 設定在每個字元之間的逾時時間.
 - *Wait CTS field*- 設定等候 CTS 逾時時間.
- The *Handshake group box*
 - *Control RTS drop-down list* - 挑選是否使用" RTS"控制.
 - *Verify CTS drop-down list* - 挑選是否使用"CTS"確認的類型.
- *Disable DTR checkbox* –您可 disable DTR 功能. 如果選擇, driver 在通之前將不會放置 DTR 信號.
- *Retries field in the Protocol group box* - 設定新的通訊重試的次數.
- The *Buffers length (bytes) group box*
 - *Tx Buffer field* - 設定傳輸緩衝區長度.
 - *Rx Buffer field* - 設定接收緩衝區長度.

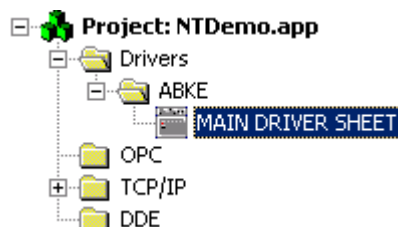
有二個能用來配置 driver 的界面:


- MAIN DRIVER SHEET;
- STANDARD DRIVER SHEETS.

MAIN driver sheet 是最容易配置 communication 的方法，而且資料點(Tag)被自動地聚集可在 runtime 期間提供最佳化的表現. 不過, 它不允許使用者像 standard driver sheet 一樣, 可個別地控制一群資料點(Tag)的掃描時間. 除此之外, 兩 sheet 能在相同的時間使用.

MAIN DRIVER SHEET

MAIN DRIVER SHEET 會自動地被插入在 driver 資料夾中.



 **備註:** 不是所有的 driver 都有 MAIN DRIVER SHEET。

要規劃 MAIN DRIVER SHEET, 按滑鼠右鍵, 選擇開啓選項:

	Tag Name	Station	I/O Address	Action	Scan	Div	Add
1	MyTagA	1	N7:W0	Read+Write	Screen		
2	MyTagB	1	B3:W0	Read	Always		
3							
4							
5							

MAIN DRIVER SHEET 選項如下:

- **Header:** 在 Header 中配置的參數會對到在這張工作表單主體中的所有資料點(Tag)生效:
 - *Description field* –可輸入文件的說明
 - *Disable field* –此欄位能輸入一個資料點(Tag)或運算式, 如果這個值是比 0 大, MAIN DRIVER SHEET 的資料點(Tag)是不作動的. 相反的若為 0 則它被是作動的. 這是一個能夠在程式執行中 enable or disable 每個 MAIN DRIVER SHEET 的 communication 的方法;
 - *Read Completed field* - 當閱讀指令被完成的時候 communication driver 改變資料點(Tag)的值.
 - *Read Status field* – 資料點(Tag)與最後的讀取指令的狀態一起更新.
 - *Write Completed field* - 當寫入指令被完成的時候 communication driver 改變資料點(Tag)的值.
 - *Read Status field* - 資料點(Tag)與最後的寫入指令狀態一起更新.
 - **Min and Max Checkbox** –此項選擇允許你設定來自現場儀器資料的最小值和最大值。
 - *Min and Max fields* –當 checkbox 被選擇的時候, 這些欄位才能夠使用. 這個欄位決定來自現場儀器的資料值的最小和最大的範圍. 應用程式會自動地做縮放比例. 工程範圍必須在資料點(Tag)屬性中被設定. 除一些已經在 driver sheet 的主體中制定 Min 和 Max 值的資料點(Tag)外, 這個設定會在工作表單中所有的資料點(Tag)中生效.

- **主體:** MAIN DRIVER SHEET 的主體可讓您規劃在應用程式中的資料點(Tag)和他們與現場儀器位址之間的關係:
 - *Tag Name field* –被 communication driver 所使用的資料點(Tag)名稱.
 - *Station field* - 儀器工作站的網路位址. 在這個欄位中語法改變決定於 communication driver. 關於進一步的資料請參照每個 driver 文件.
 - *I/O Address field* –對應到應用程式資料點(Tag)與現場的儀器位址. 在這個欄位中語法決定於 communication driver. 關於進一步的資料請參照每個 driver 文件.
 - *Action field* – 此欄位定義 communication 方向, 有三種選項: 讀; 寫, 讀+ 寫.
 - *Scan field* –此欄位定義何時更新讀取資料點(Tag). 有二選項: Always(永遠 enable), Screen(只有當畫面被開啓時, 才去更新被配置在畫面中的資料點(Tag)).

- **Div field** – 提供設定除法常數用來調整刻度. 在讀取運作時此值將會是除數, 但在寫入操作時此值將會是一乘數. 如果你已經在組態主體中使用 **Min** 或 **Max**, 切勿使用這個欄位.
- **Add field** - 用來設定附加常數調整刻度. 在讀取操作時此值將會是加數, 但在寫入操作時此值將會是一減數. 如果你已經在組態主體中使用 **Min** 或 **Max**, 切勿使用這個欄位.

提示: 使用 Scan 欄中的 Screen 選項, 將只有當螢幕打開時才去 scan 所顯示的資料點 (Tag). 它可以最佳化 communication, 因為系統將不會浪費時間閱讀不需要的資料, 而且將提昇整體的表現.

STANDARD DRIVER SHEETS

當每個 driver 只有單一 MAIN DRIVER SHEETS 的時候, 你能對每個 driver 創造一些 STANDARD DRIVER SHEETS. STANDARD DRIVER SHEETS 提供附加的欄位來控制 communication.

在被選擇的 driver 子資料夾上按滑鼠右鍵, 打開 driver 工作表單, 並且選擇插入.


	Tag Name	Address	Div	Add
1				
2				
3				
4				
5				

STANDARD DRIVER SHEETS 選項如下:


- **Header:** 在 Header 中配置的參數會對到在這張工作表單主體中的所有資料點(Tag) 生效:
 - **Description field** - 可輸入文件的說明.
 - **Increase Read Priority checkbox**–當 checkbox 被選擇的時候, 在此 sheet 中的讀取和寫入指令被觸發時將被放在 communication 序列的頂端優先處理.

警告: 當使用增加優先權選項的時候, 需特別的注意. 如果一工作表單使用增加優先權選項並持續觸發 communication 指令, 另一個 driver sheet 將無法被執行.


- **Read Trigger field** - 放入一個實行工作表單讀取的資料點(Tag), 當你改變資料點(Tag)的值時, 工作表單中讀取的工作將被執行.
- **Enable Read when Idle field** - 可輸入一個資料點(Tag)或常數值. 當資料點(Tag) (或常數) 的值比 0 大時, 即從儀器讀取資料.

 **警告:** 如果使用一個常數時(非 0), 確定您的應用程式是要不斷地讀取資料, 因為這將會在每個 communication 掃描中連續地發出讀取請求.

- **Read Completed field** - 當閱讀指令被完成的時候 communication driver 會改變資料點(Tag)的值.
- **Read Status field** - 資料點(Tag)與最後的讀取指令狀態一起更新.
- **Write Trigger field** - 當您改變資料點(Tag)的值時, 工作表單中寫出的工作將被執行.
- **Enable Write on Tag Change field** - 可以是一個資料點(Tag)或常數值. 每當值不是 0, communication driver 將不斷地查詢在工作表單中的資料點(Tag)值. 當值發生變化, 值將會被寫入在此位址的現場儀器.
- **Write Completed field** - 當寫入指令被完成的時候 communication driver 會改變資料點(Tag)值.
- **Write Status field** - 資料點(Tag)與最後的寫入指令狀態一起更新.
- **Station field** - 網路中儀器工作站的位址. 在這個欄位中語法改變決定於 communication driver. 關於進一步的資料請參照每個 driver 文件.
- **Header field** - 識別將在儀器上被讀或寫資料類型或起始位址. 在這個欄位中語法改變決定於 communication driver. 例如: station: {tagStation}, Header: MEMORY {tagAddress}.

 **備註:** Station 和 Header 欄位包含了資料點(Tag)值的文字, 使用語法: text{tag}.

- **Checkbox (not labeled)** - 此項選擇允許你設定來自現場儀器資料的最小值和最大值.
- **Min and Max fields** - 當 checkbox 被選擇的時候, 這些欄位方能夠使用. 這欄位決定來自現場儀器資料值的最小和最大的範圍. 應用程式會自動地做縮放比例. 工程範圍必須在資料點(Tag)屬性中被設定. 這個設定除一些已經在 driver sheet 的主體中設定 Min 和 Max 值的資料點(Tag)外, 會在工作表單中所有的資料點(Tag)生效.
- **Body: Standard DRIVER SHEET** 的主體允許你配置在應用程式中的資料點(Tag)和在他們現場儀器位址之間的關係:
 - **Tag Name field** - 被 communication driver 所用的資料點(Tag)名字.
 - **Address field** - 對應到應用程式資料點(Tag)的現場儀器位址. 在這個領域中語法改變決定於 communication driver. 關於進一步的資料請參照每個 driver 文件.
 - **Div field** - 提供設定除法常數用來調整刻度. 在讀取運作時此值將會是除數, 但在寫入操作時此值將會是一乘數. 如果你已經在組態主體中使用 Min 或 Max, 切勿使用這個欄位.
 - **Add field** - 用來設定附加常數調整刻度. 在讀取操作時此值將會是加數, 但在寫入操作時此值將會是一減數. 如果你已經在組態主體中使用 Min 或 Max, 切勿使用這個欄位.

 **備註:**
在 driver 的每個 communication 工作表單中資料點(Tag)的最大數值是 512, 對某一些 driver 這個數字可能會更少. (參考 driver 文件)

讀取操作: <資料點>=<在儀器中的值>/ Div+ Add. 寫入操作: <在儀器中的值>=<標籤> - Add)*Div. 如果你讓 Div 及 Add 為空白, 這個功能被忽略.

ERROR CODES (Status fields)

現場讀寫狀態將回傳錯誤碼給配置的資料點(Tag). 這些錯誤碼可以是標準或為每個 driver 客制化. 標準的錯誤碼通常是負值並且描述如下. 客制化錯誤碼被描述在每個 driver 的文件中.

- 0= OK
- 1= 不合法的序列埠
- 2= 不合法的鮑率
- 3= 不合法的位元數;
- 4= 不合法的停止位元數;
- 5= 不合法的同等位元
- 6= 不合法的 irq
- 7= 序列埠已經在使用中;
- 8= 不合法的緩衝區大小
- 9= 記憶體不足;
- 10= Tx 緩衝區空的
- 11= Tx 緩衝區滿了
- 12= Rx 緩衝區空的
- 13= Rx 緩衝區滿了
- 14= CTS 等候逾時
- 15= 開始訊息等候逾時
- 16= 等候一個訊息的完成逾時
- 17= 在 rx 的字元之間逾時
- 18= 在 tx 的字元之間逾時
- 19= 沒有 carrier
- 20= 沒有 DSR
- 21= 在位址中不能夠找到一 8250;
- 22= Tx 線是忙碌的
- 23= 使用者中斷
- 24= 功能不支援
- 25= overrun
- 26= Parity
- 27= overrun and parity
- 28= Framing
- 29= Framing 和 overrun
- 30= Framing 和 parity
- 31= Framing , overrun 和 parity
- 32= 等候 tx 訊息的完成逾時

 **備註:** 標準的錯誤碼的描述在\Advantech Studio\BIN 中的 UNICOMM.MSG 檔案. 描述每個 driver 客制化的錯誤碼被列出在\Advantech Studio\DRV 中的 <DriverName>. MSG.

▪ The OPC Folder

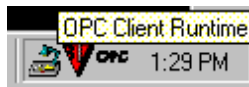
此資料夾讓你能藉著一個 OPC Server 配置一個 OPC 的界面到應用程式中. Advantech Studio OPC Client 使 Advantech Studio 能夠與任何裝置的 OPC Server 通訊. 此模組遵行 OPC 標準文件 " OLE for process control data access standard version 1.0 ", 此文件可在網站<http://www.opcfoundation.com> 獲得. 在使用 Advantech Studio OPC Client 之前, 你需要在執行應用程式的機器中安裝並且配置 OPC Server.

在 Client 機器上，你需要使用 OPC Client 組態計畫配置 Server 辨識器，communication 參數，和你想要連接的項目。若要存取 Client 組態，在 "COMM" table 插入新的 OPC Client 文件。

OPC 的組態表有下列各項輸入：

- **Description:** 作為說明使用
- **Server Identifier:** 這個領域應該包含你想要連接的 Server 的名字。如果 Server 在電腦中被安裝，它的名字能透過目錄盒子被選擇。
- **Disable:** 這個領域可以是一個標籤或一個常數。如果它的價值是不同的零，與 OPC Server 的 communication 將無效。
- **Update Rate:** 這個領域指示 Server 將會多麼的時常的更新這個群體(以毫秒計)。如果是零，指示 Server 應該使用最快速的實際費率。
- **Percent Deadband:** 表示超過多少百分比的項目值變化將會引起 Server 的一個通知。它只對類比項目有有效。
- **Tag Name:** 這領域應該包含連線到 Server 項目的標籤名稱。
- **Item:** 這些領域應該包含 Server 的項目名字。一旦你有選擇一個 OPC Server，你能使用 OPC 瀏覽器選擇 Server 的項目。在項目領域中按滑鼠右鍵- 點選和選擇 OPC 瀏覽器選項。

你能藉由選擇 Project - > Status 目錄選項建立 OPC Client runtime 組件自動地執行。在執行這個計畫之後，一個小的圖像在你的 system tray 中被顯示。要關閉 Advantech OPC Client runtime 組件，在它的圖像上按滑鼠右鍵，並挑選 Exit。



▪ The TCP/IP Folder

這個資料夾使你能夠配置到其他 Advantech Studio 工作站的 TCP/IP Client 界面。Advantech Studio TCP/IP Client 和 Server 組件使二個以上的應用程式能夠讓他們 databases 同步化。這些組件使用 TCP/ IP protocol 作為在應用程式之間的 communication。在使用 Advantech Studio TCP/IP Client 和 Server 組件之前，你需要在執行這些應用程式的機器中安裝並且配置 TCP/IP protocol。

你不需要為 Server 設定任何事。你只需要執行組件 Advantech Studio TCP/IP Server。你能選擇自動執行它，或藉由選擇 Project - > Status 選項手動執行它。在執行這個 Project 之後，一個小的圖像在你的 System tray 中被顯示。

要關閉 Advantech Studio TCP/IP Server runtime,在 System tray 中它的圖像上按滑鼠右鍵，並挑選 Exit。



你需要使用 TCP/IP Client Configuration 配置 Server IP 位址，以及你想要與 Client 系統上的 Server 分享的標籤配置。

TCP/ IP Client configuration program 位於 Comm table 之上，它有和 driver configuration program 相同的指令。以下是你需要填寫 TCP/IP Client configuration 的領域描述：

- **Description field** - 作為說明使用。
- **Connection Status field** - 這欄位包含一個資料點(Tag)名稱. TCP/ IP Client configuration 組件將會依照連接狀態而更新這個資料點(Tag), 如果資料點(Tag)的值是 0, 那麼連接是 OK. 否則, 它將由 windows socket library 回復錯誤碼.
- **Server IP Address field** - 這欄位包含 Server IP address. 它可能是字串, 或你可能使用一個加上大括號的資料點(Tag). 舉例來說, 如果你將這個欄位填上 {tag_name}, TCP/IP Client 將會試著藉由此資料點(Tag)tag_name 指示的 Server 作連接.
- **Tag Name field** -這欄位應該包含你想要對 Server 分享的資料點(Tag). 如果資料點(Tag)是陣列 或是 Class (或兩者都有), 每個元件和成員都將被分享. 你只需在這個欄位中放上資料點(Tag)名稱, 不需設定索引或 class 成員. 如果你設定一個索引或一個 class, TCP/ IP Client 將會不理睬它.
- **Remote Tag field** -這欄位應該包含遠端將與本地資料點(Tag)連結的資料點(Tag)名字. 這個欄位是可選擇的, 如果你將它留下空白, 在 Client 和 Server 將使用相同的資料點(Tag)名字.

⚠ **警告:** 如果你要分享陣列, 在 Server 中的資料點(Tag)應該和 Client 裡的資料點(Tag)包含相同數字的元件. 如果資料點(Tag)是一個 class, 在 Server 和 Client 應用程式中 class 定義應該是相同的. 如果你不遵從這些規則, 可能發生不可預知的結果.

你能藉由選擇 Project -> Status 目錄選項建立 Advantech Studio TCP/IP Client Runtime 組件自動地執行. 在執行這個 project 之後, 一個小的圖示在你的 system tray 中被顯示. 要關閉 Advantech Studio TCP/IP Client Runtime 組件, 在它的圖示上按滑鼠右鍵, 並選 Exit.



有三個參數你能在應用程式組態 (.app) 檔案中配置.

[TCP]

Port = TCP/ IP port number. 預設值 =1234

SendPeriod= Client/ Server 組件將會更新另一個機器的資料點(Tag)值的時間, 以毫秒計. 預設值 =250

ConnectRetryTimeout=在 Client 應該再試對 Server 連接的時間, 以秒計. 預設值 =30

在 Client 和 Server 中機器埠參數應該是相同的. 只有 Client 使用 ConnectionRetryTimeout .

▪ The DDE Folder

資料夾允許你配置一個 DDE Client 組態到一個 DDE Server 應用程式, 例如 Excel 和支援這個界面的任何其他的視窗程式.

動態的資料交換 (DDE) 是一個為在視窗應用程式間動態的資料交換的協定, 例如 Excel 和支援這個界面的任何其他的視窗程式. DDE 是一種 Server 和 Client 應用程式之間的對話. Studio 可以當做 Client 或為一個 Server. 你可以在主選單上的 Project Setting 選項之下的 Runtime Task 中看見 DDE Client Runtime 和 DDE Server.

要當一個 DDE Server, 只需在 Runtime Task 中啟動 DDE 或 NetDDE Server Task. 若要當一個 DDE Client, 則需配置 Comm Tab 上的 DDE 界面工作表單.

網路動態的資料交換 (NetDDE) 是 DDE 的延伸, 它透過網路在電腦間交換資料. 要 Studio 當一個 NetDDE Server, 需啟動 DDE Server Application. 若要 Studio 當一個 NetDDE Client, 則需配置 Comm Tab 上的 DDE 界面工作表單.

備註: 當執行 NetDDE 的時候, 只接受寫入指令. 要讀資料, 則需在 Server 電腦上配置一寫入指令.

在 DDE 資料夾上的按滑鼠右鍵插入一張新的工作表單的. 按提示打開一張 DDE 工作表單.

	Tag Name	Item
1		
2		
3		
4		
5		

DDE 工作表單被區分為二個部分: 有所有的整體資訊的標頭檔和關聯到 DDE Server 應用程式的資料點(Tag)和選項. 每個 DDE 界面以三層結構指向一個應用程式作為基礎, 即應用程式名字, Topic, 和 Item. 第一要務是找出在 DDE Server 應用程式中的 DDE 結構.

DDE Client 的標題允許你定義將會開始讀取和寫入的資料點(Tag), 和接收連接狀態的資料點(Tag).


- *Description* field - 作為說明使用.
- *Application Name* field - DDE Server 應用程式名字.
- *Topic* field - 在 Server 應用程式中的 Topic.
- *Connect* field - 控制 Studio DDE Client 和 DDE Server 應用程式連接的資料點(Tag). 當這個資料點(Tag)被設定成 1 的時候, 它對 Server 請求一個連接. 如果連接不被允許, 或是失敗, Studio 將資料點(Tag)設回 0. 如果連接是 OK, 這個資料點(Tag)值則維持為 1.
- *Read Trigger* field - 放入一個實行工作表單讀取的資料點(Tag), 當你改變資料點(Tag)的值時, 工作表單中讀取 DDE server 的工作將被執行. 此項目僅可用在本地端的 DDE, 而無法用在 NetDDE servers.
- *Enable Read when Idle* field - 可輸入一個資料點(Tag)或常數值. 當資料點(Tag) (或常數) 的值比 0 大時, 即從儀器讀取資料.
- *Read Status* field - 資料點(Tag)讀取的狀態.
- *Write Trigger* field - 改變資料點(Tag)的值, 開始下 poke 與 server 作整合工作.

- *Enable Write on Tag Change* field –當此欄位的資料點(Tag)值大於 0, communication driver 將不斷地查詢在工作表單中的資料點(Tag)值. 當值發生變化, 值將會被寫入在此位址的現場儀器.
- *Write Status* field – 寫入命令的狀態.

DDE client 主體的工作表單允許您規劃每一個資料點(Tag)與每一個 DDE server 位址上的 ITEM 作連結.

- *Tag Name* field – 設定接收 DDE server 應用程式中寫出或讀入的資料點(Tag).
- *Item* field – 是 DDE server 中所設定的位址即當為 DDE 的 ITEM. 參考在 server 上的軟體設定文件及 APP, TOPIC, 及 ITEM 相關的語法. 您可以用語法 text{tag}來設定 *Topic* 及 *Item* 欄. {tag}的值是個字串型態用於 address 欄中. 例如: Topic: topic_{tag_topic_name}_example; Item: {tag_item_name} or A{tag_number}.

規劃 NetDDE 的連結與規劃 DDE 是類似的, 除了要多定 Header Application name 及 topic. 在測試之前您必需先確認 DDE Server 在工作站上已執行.

 **備註:** 除了在 *Studio* 設定與 server 的連結外, 請參考在 server 上的軟體設定文件及 APP, TOPIC, 及 ITEM 相關的語法.

在 NetDDE Client 工作表單的前頭可以允許您設定資料點(Tag)來開始讀取或寫出, 也可以用來接收連結時的狀態.

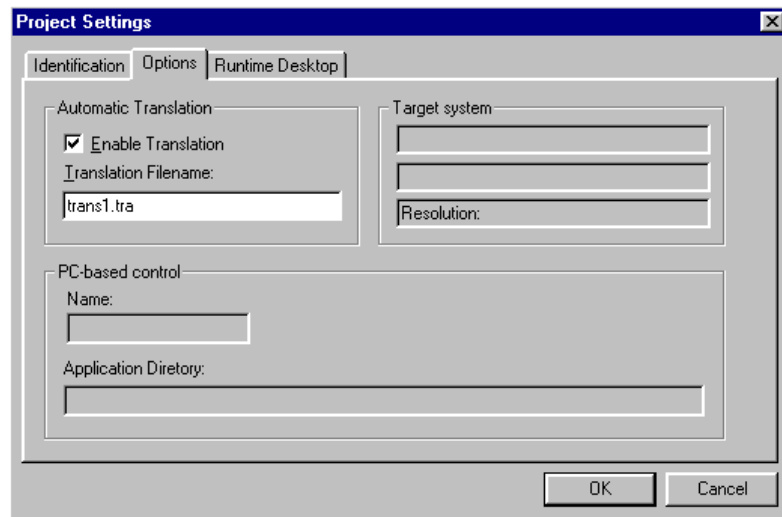
- *Application Name* field: 輸入 <computer name>\NDDE\$, <computer name> 必需是一個合法的網路上電腦名稱.
- *Topic* Field: 與其他 *Studio* 連結時, 使用 topic 名稱 UNISOFT\$.


DDE client 在工作表單中的主體中允許您規劃每個資料點(Tag)與每個 DDE server 中的 ITEM 作連結.

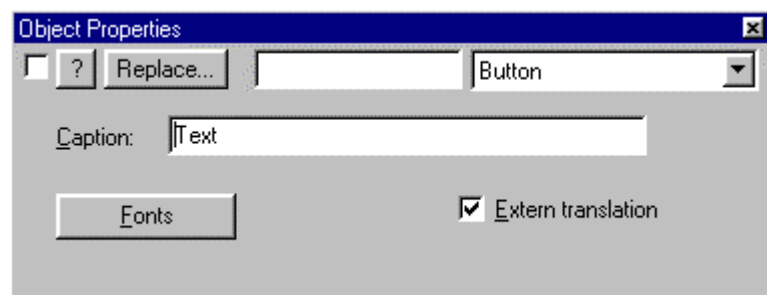
- *Tag Name* field – 指定本地端的 *Studio* 資料點(Tag)名稱, 與其他遠端的資料點 (Tag)作連結.
- *Item* field – 指定遠端的資料點(Tag), 與本地端資料點(Tag)作連結.


II.6. 翻譯功能

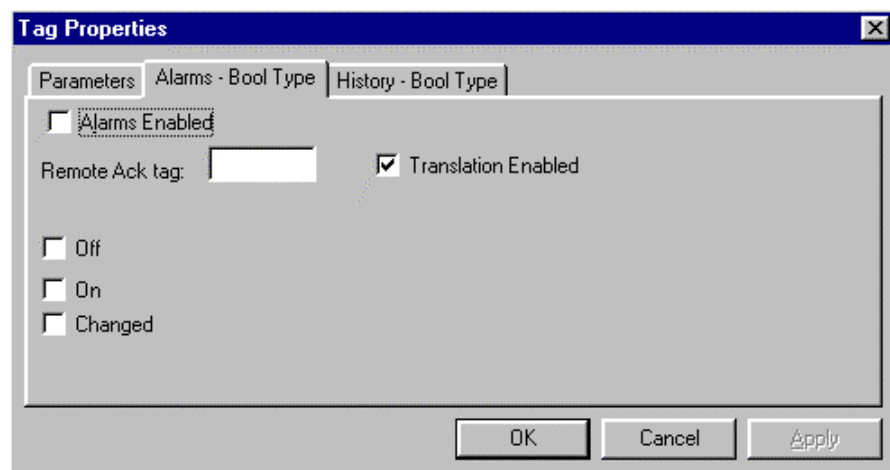
你可以翻譯成另外一種語言而不需要重新畫圖或建立警報訊息, 只要開一個檔案內建入對應字串, 當你執行應用程式時, 這些訊息會以所選擇語言來呈現, 建立翻譯檔案可以在 menu bar Project -> Setting -> Options 裡面啟動並鍵入翻譯檔案名稱.



首先用 **Text**  建一個字串, 叫出物件屬性, 勾選外部翻譯選項 (*Extern Translation checkbox*).

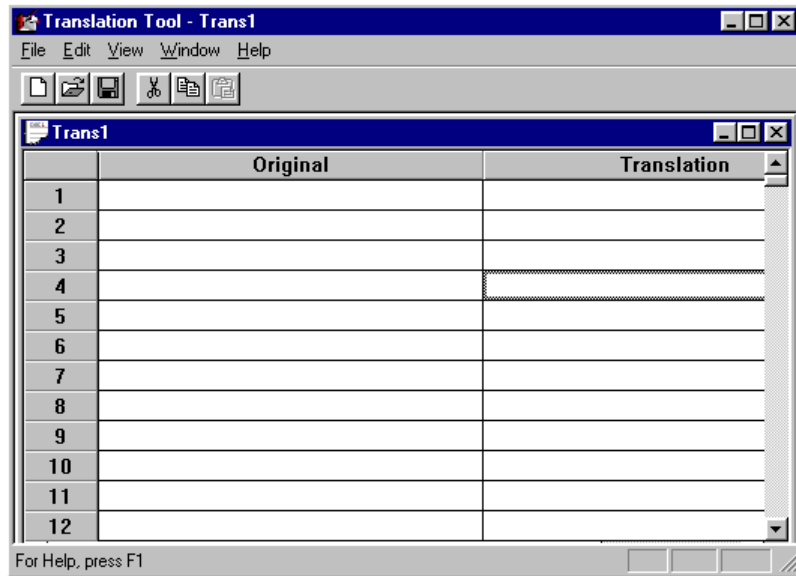


若要翻譯一個資料點(Tag) 選擇一個警報資料點(Tag) 按 **Tag Properties**  鍵, 顯示 *Tag Properties* 視窗, 勾選翻譯選項 *Translation Enabled checkbox*.



Translation Enabled Check-box on the Tag Properties Window

若要翻譯 *math expressions* 可使用 *Scripting Language's EXT()* 函數
 要開啓翻譯檔案, 選 *Tools-> Translation Tool* 假如你要翻譯第三種語言 選 *File->New* 開啓另外一張工作表.



備註: 用 *Scripting Language's* SetTranslationFile() 函數執行動態轉換字串.

II.7. Tag Reference

這一節提供建立和編輯資料點(Tag)的訊息

II.7.1. Tag Syntax

請遵循以下規則為資料點(Tag)取名字:


- 可以用字母數字和底線“_”
- 不能用下列字元: ~ ` ! @ # \$ % ^ & * () - = \ + \ [] { } < > ?
- 第一個必須為字元.
- 最大長度是 32 字元的資料點(Tag)或 16 字元的 class.
- 大小寫均視為相同但建議用 *TankLevel* 取代 *tanklevel*.
- 必須和 internal tag name 及 math functions 不同.

看 *Scripting Language*. 會有更多的訊息


備註: 字元 @ 放在字首表示當作是間接指向的資料點(Tag).

Tag 範例: temperature, pressure1, count, x

II.7.2. Tag Field Syntax

Field 是一組資料點(Tag)在數據庫裡的參數, Applications 在 runtime 時將它們視為 tag fields. 所有的參數都在 *Tag Properties* 視窗內定義, 按 toolbar 上的 **Tag Properties**  button. 用 TagName -> Field 的語法. 在 runtime 時 你可擷取到參數值:

- **Min** –資料點(Tag)之最小工程值.
- **Max** –資料點(Tag)之最大工程值.


 **備註:** 資料庫不會接受一個寫入超過範圍的值到資料點(Tag), 並會出現警告訊在 LogWin 和 OutPut 視窗, 如果你不想要使用這些性質, 填入 0 即可.

- **Unit** – 工程單位, 接受最多 9 個字元.
- **Description** –資料點(Tag)的描述.
- **Size** –陣列大小. 如果資料點(Tag)不是陣列, 則將展示" 0 " .
- **TimeStamp** –資料點(Tag)上次變化的日期和時間.
- **Quality** –資料點(Tag)的通訊品質, 例如 GOOD:192 ; Bad: 0. 這用於通信協定
- **B0-B31** – 整數的 Bit 0 到 Bit 31 的值.

你能夠用 Tagname->Field 句法來存取下面和警報相關的值.

- **Ack** –如果值比" 0 "大. 意味著它仍然是警報.
- **AlrDisable** – 1 = 失去能力, 0 = 賦予能力.
- **AlrStatus** – 值若為" 0 "表示沒有警報. 比" 0 " 大的值表示有相關的警報狀況.
- **Alarm Types** – 型態包括 HiHi, Hi, Lo, LoLo, Rate, Dev+, and Dev-.


所有的 field 都可以由具有下面的句法來閱讀: Tag->Field 。
例如 level->Max, Temp->Unit, 和 pv101->HiHiLimit.

 **警告:** 不能用 Tag 參數來構成警報或者趨勢圖的工作表

下面是你(們)在 runtime 期間能夠抓取的參數概要:

<u>Field Name</u>	<u>Boolean Tag</u>	<u>Integer Tag</u>	<u>Real Tag</u>	<u>String Tag</u>	<u>Allows change on the fly</u>
Min	no	yes	yes	no	no
Max	no	yes	yes	no	no
Unit	yes	yes	yes	yes	no
Description	yes	yes	yes	yes	no
Size	yes	yes	yes	yes	no
TimeStamp	yes	yes	yes	yes	no *
Quality	yes	yes	yes	yes	no *
B0-B31	no	yes	no	no	yes
Ack	yes	yes	yes	yes	no *
AlrDisable	yes	yes	yes	no	yes
AlrStatus	yes	yes	yes	no	no *
HiHiLimit	yes	yes	yes	no	yes
HiLimit	yes	yes	yes	no	yes
LoLimit	yes	yes	yes	no	yes

LoLoLimit	yes	yes	yes	no	yes
DevSetpoint	no	yes	yes	no	yes
Dev+Limit	no	yes	yes	no	yes
Dev-Limit	no	yes	yes	no	yes
RateLimit	no	yes	yes	no	yes
HiHi	yes	yes	yes	no	no
Hi	yes	yes	yes	no	no
Lo	yes	yes	yes	no	no
LoLo	yes	yes	yes	no	no
Dev+	no	yes	yes	no	no
Dev-	no	yes	yes	no	no
Rate	no	yes	yes	no	no

 警告: 儘管系統允許改變上述有星號 (*) 的參數 但不建議嘗試改變. 包括 AirStatus, TimeStamp, Quality, and Ack.






II.7.3. Tag Types

資料點(Tags)可以是設備的通信點, 計算的結果, 警報點等. 在 *Studio* 中, 所有的 Tags 都在 *Database* 工作表中定義.

- **Application Tags** – 使用者建立的資料點(tags)稱為 application tags. 用在圖面顯示, 讀寫現場值, 控制或是運算輔助點等.
- **Internal Tags** - *Studio* 本身已有的點稱為 internal tags. 具備已定義的函數 (時間, 日期, 警報確認, 登錄使用者) 並且不能被刪除或修改.
- **Shared Tags** - 資料點(Tags)在 PC-Based Control software 內載入到 *Studio* 的環境. Shared tags 並不能在 *Studio* 的環境內修改.
- **Classes** - Classes 是一種包起來的資料結構它不只包含單一數值 而是一組數值, 你可以建立 一組變數 members 為 class-type.

II.7.4. Tag Values

資料點(Tag)的數值可以是以下數種形式. 下列圖示會和其關聯的數值型態出現在 *Database* tab.

-  **Boolean** – 布林或數位變數(0 or 1).
-  **Integer** – 整數 (正 負 或零).
-  **Real** – 實數為 double word.
-  **String (ASCII text)** – 最多 255 字元的字串 可以是字元 數字或特殊字元 如: Recipe product X123, 01/01/90, *** On ***.
-  **Class** – 使用者自行定義的混合型態

II.7.5. Array Tags

Studio tags 能夠以矩陣的型態表示, 用索引來區分(0 – 255). 語法為 `<ArrayTagName>[ArrayIndex]`. *Example*: `tank[0]`, `tank[2]`, `tank[3]`, `tank[255]`.

⚠ **警告:** 最大的索引規劃在 *database* 的 *size* 欄位. 當 *size* “n” 輸入時, 表示矩陣範圍是 0 到 n, 例如: *TagA size* 是 3, *tags* 可以是: `TagA[1]`, `TagA[2]`, and `TagA[3]`. 不建議使用位置 [0] (zero) 因為它用在內部錯誤規劃.

矩陣的型態可以使記憶體作最有效的利用並且簡化規劃的工作, 例如: 作一張圖包含定義成變數 *index* 的矩陣 如;

`pressure[tk]`, `temperature[tk]`, `temperature[tk +1]`.

矩陣的索引可以是一個資料點(*tag*), 一個數值, 或是一個數學式含運算子(operator) “+”.

🔗 **備註:** 要參考具索引和有運算子“+”的陣列, 必須用以下的語法: `<ArrayTagName>[<NumValue1> + <NumValue2>]`, 其中 `<NumValue1>` 和 `<NumValue2>` 可以是整數變數或常數, 例如: `temperature[tk+2]`, `temperature[tk+6]`, `temperature[TagA + TagB]`.

用矩陣資料點(*tags*)可以節省很多發展時間, 假設你需要的資料點(*tag*)為四個槽的溫度傳統的做法是:

```
temperature1  high temperature on tank 1
temperature2  high temperature on tank 2
temperature3  high temperature on tank 3
temperature4  high temperature on tank 4
```

用矩陣資料點(*tags*)可以簡化為:

```
temperature[j]  high temperature on tank {j}
```

🔗 **備註:** 當你建立一個四個元件的矩陣, 系統自動產生 0 到 4. 例如: `tag_example[15]`, // 開始位置為 0// 因此 `the tag_example[15]` 有 16 個元件 .

II.7.6. Indirect Tags

Studio 支援間接擷取 *database* 的資料點(*tags*). 例如一個字串 *tag X*, 此字串的內容可能是另外一個 *tag* (也就是提供另外一個 *tag* 的 *pointer*). 語法是: `@<IndirectTagName>`. 例如: 假設一個 *tag X* 值為字串 “TEMP” 對 `@X` 作讀寫則連結到 *TEMP* 變數.

🔗 **備註:** 任意一個字串資料點(*tag*)均可成為 *indirect tag* (*pointer*).

參考到 *class-type tag* 亦可如下

```
Class - TANK 的 members Level
Tag - TK class:TANK 型態
Tag - XCLASS 字串型態
```

若要擷取 *TK.Level* 值, 必須將 “TK.Level” 存到 *XCLASS* 再抓 `@XCLASS`. 也可以應用 *class-type tag*, 如 `@XCLASS.Level`

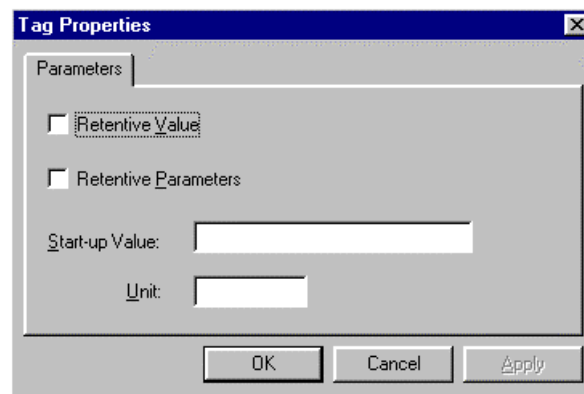
當要建立資料點 (tags) 給 indirect 使用時, 鍵入 @X 在 tag 欄位裡而不是鍵入字串 如: @Z Integer, @X Class:TANK.

II.7.7. Tag 性質 – 參數

每一個資料點 (tag) 都有以下的性質 (按 工具欄上的 Tags Properties):

- 參數
- 警報性質
- 歷史

PARAMETERS OF THE STRING TYPE TAGS



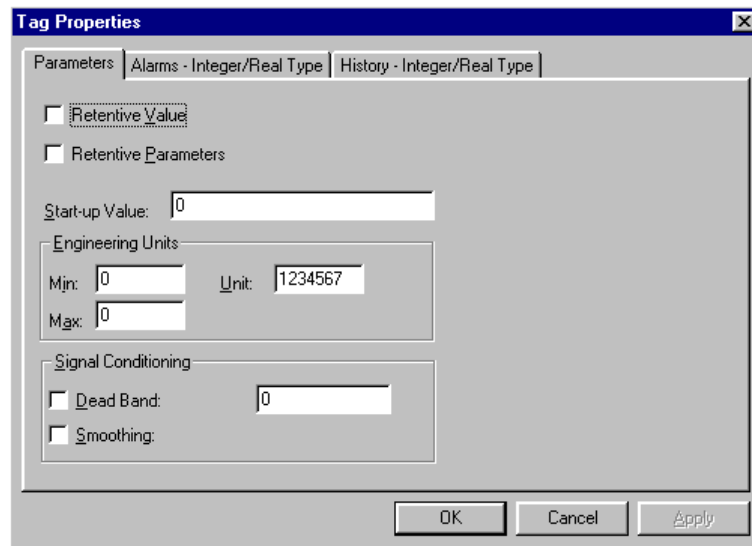
- **Retentive Value** checkbox – 防止系統當機, 連續存值, 當重開時會以上次值開始.

⚠警告: Avoid 此功能將造成頻繁的硬碟擷取, 會降低速度

- **Retentive Parameters** checkbox – 保留所有資料點(tag)欄位在 runtime 時的改變.
- **Start-up Value** – 系統載入時的資料點(Tag)起始值
- **Unit field** – 此位置接受任何字串 (最多 9 個字元) . 在 runtime 時亦可改變.

⚠警告: 在 Min 和 Max 範圍之外的值均不會接受, 同時寫入失敗的訊息會在 LOGWIN 產生.

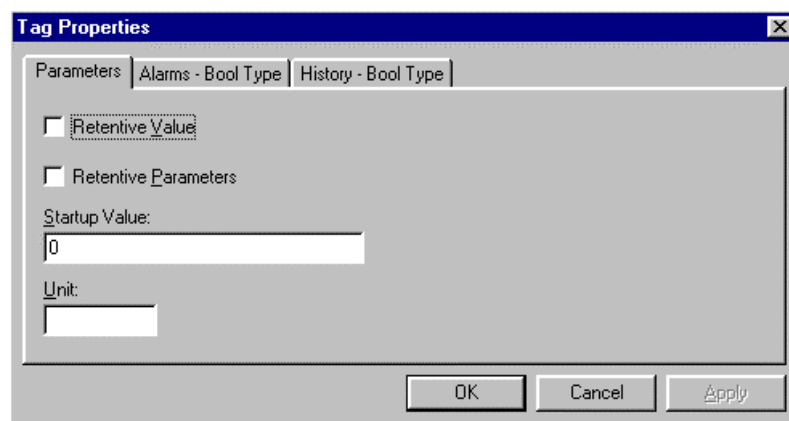
PARAMETERS OF THE INTEGER AND REAL TYPE TAGS



前面未提及的參數 在下面描述

- **Engineering Units** 群組
 - **Min** field - 工程單位最小值, 在 *during runtime* 時亦可輸入.
 - **Max** field - 工程單位最大值, 在 *during runtime* 時亦可輸入.
 - **Unit** field - 此位置接受任何字串 (最多 9 個字元) . 在 *runtime* 時亦可改變.
- **Signal Conditioning** 群組
 - **Dead Band** checkbox – 輸入資料點(Tag)的 “dead band” 值, 它是資料點(Tag)中心值的變化值, 當測量值小於它時警報不動作.
 - **Smoothing** checkbox – 減少資料點(Tag)值的變化, 只針對整數和實數. 例如: 前一次的值是 50 而下一次的值是 60, 系統將會得到平均值 55.

PARAMETERS OF THE BOOLEAN TYPE TAGS



這些 checkbox 和 field 的描述和前面所提是一樣的

II.7.8. Tag Properties – Alarms

在 *Tag Properties* 視窗裡, 可以看到警報組態, 假如有開啓中的警報工作表, 這個功能就會被解除, 在使用此功能前必須建立警報群組.

包括:

- **HiHi** – 非常高的警報出現.
- **Hi** – 高的警報出現.
- **Lo** – 低的警報出現.
- **LoLo** – 非常低的警報出現.
- **Rate** – 過於劇烈變化的警報出現.
- **Deviation** – 一個和設定值達某種差異的警報出現.

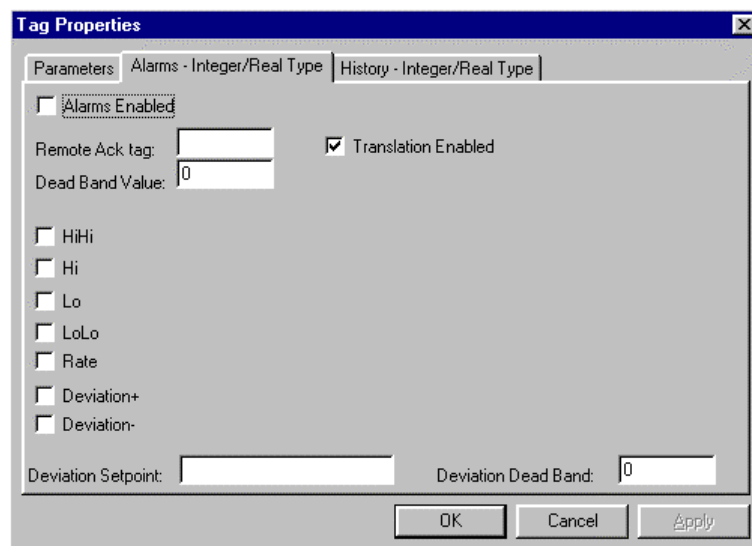
Deviation Alarm 的例子: 假如設定值 $SetPoint = 50$, 差異值 $Deviation + = 5$, 差異值 $Deviation - = 5$, 而差異值不反應區 $Deviation Dead Band = 0.5$, 一個警報在 $temp > 55.5$ 或 $temp < 44.5$ 會產生. 而在 $temp > 45$ 或 $temp < 55$ 時則會恢復正常.

ALARM LIMITS

警報限值如下:

- **HiHiLimit** – 這個欄位定義一個非常高的警報, 這個值在 runtime 時可以寫入
- **HiLimit** – 這個欄位定義一個高的警報, 這個值在 runtime 時可以寫入.
- **LoLimit** – 這個欄位定義一個低的警報, 這個值在 runtime 時可以寫入.
- **LoLoLimit** – 這個欄位定義一個非常低的警報, 這個值在 runtime 時可以寫入.
- **DevSetpoint** – 對一個資料點(tag)值的差異變化, 所設定的參考值能驅動一個警報, 這個值在 runtime 時可以寫入
- **Dev+Limit** – 當資料點(Tag)值超過 DevSetpoint 加上 Dev+Limit 時, 能驅動一個警報, 這個值在 runtime 時可以寫入
- **Dev-Limit** – 當資料點(Tag)值低過 DevSetpoint 減掉 Dev-Limit 時, 能驅動一個警報, 這個值在 runtime 時可以寫入
- **RateLimit** – 當資料點(Tag)值的每單位時間的變化量超過設定值時能驅動一個警報, 這個值在 runtime 時可以寫入

ALARMS FOR THE INTEGER AND REAL TYPE TAGS



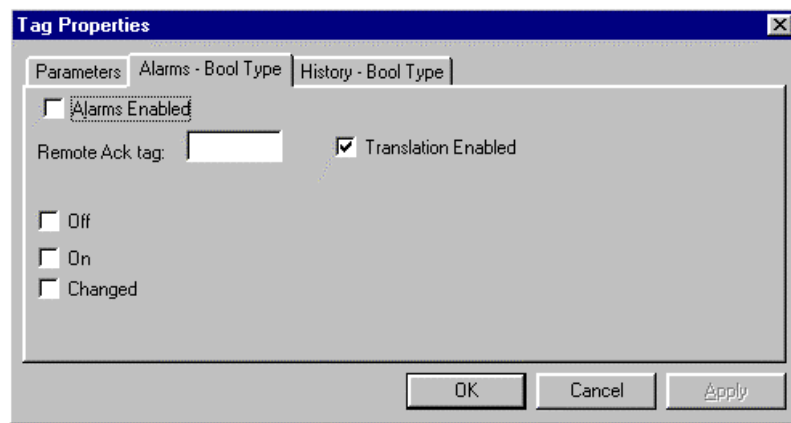
- **Alarms Enabled** checkbox – 啟動警報
- **Remote Ack tag** field – 這個資料點(tag)可確認警報
- **Dead Band Value** field - 警報產生區間, 例如:TEMP1, 規劃高警報= 90 和 區間值= 5, 當 $TEMP1 \geq 90$ 會產生警報, 當 $TEMP1 < 85$ 時恢復正常.
- **Translation Enabled** checkbox – 啟動翻譯警報訊息, 注意得先在 *Project Settings* -> *Options* 設定.

備註: 若勾選 *Translation Enabled* 則翻譯後的警報訊息會存在
“<MyProject>\DATABASE\ Alarm.TXT”

警報限值如下:

- **HiHiLimit** – 這個欄位定義一個非常高的警報, 這個值在 *runtime* 時可以寫入.
- **HiLimit** – 這個欄位定義一個高的警報, 這個值在 *runtime* 時可以寫入.
- **LoLimit** – 這個欄位定義一個低的警報, 這個值在 *runtime* 時可以寫入.
- **LoLoLimit** – 這個欄位定義一個非常低的警報, 這個值在 *runtime* 時可以寫入.
- **DevSetpoint** – 對一個 *tag* 值的差異變化 所設定的參考值, 能驅動一個警報 這個值在 *runtime* 時可以寫入.
- **Dev+Limit** – 當 *Tag* 值超過 *DevSetpoint* 加上 *Dev+Limit* 時能驅動一個警報, 這個值在 *runtime* 時可以寫入.
- **Dev-Limit** – 當 *Tag* 值低過 *DevSetpoint* 減掉 *Dev+Limit* 時能驅動一個警報, 這個值在 *runtime* 時可以寫入.
- **RateLimit** – 當 *Tag* 值的每單位時間的變化量超過設定值時能驅動一個警報, 這個值在 *runtime* 時可以寫入.
- **Deviation Dead Band** - 警報差異變化的產生區間

ALARM FOR THE BOOLEAN TYPE TAGS




- **Alarms Enabled** checkbox – 啟動警報
- **Translation Enabled** checkbox - 啟動翻譯警報訊息, 注意得先在 *Project Settings* -> *Options* 設定.

備註: 若勾選 *Translation Enabled* 則翻譯後的警報訊息會存在
“<MyProject>\DATABASE\ Alarm.TXT”

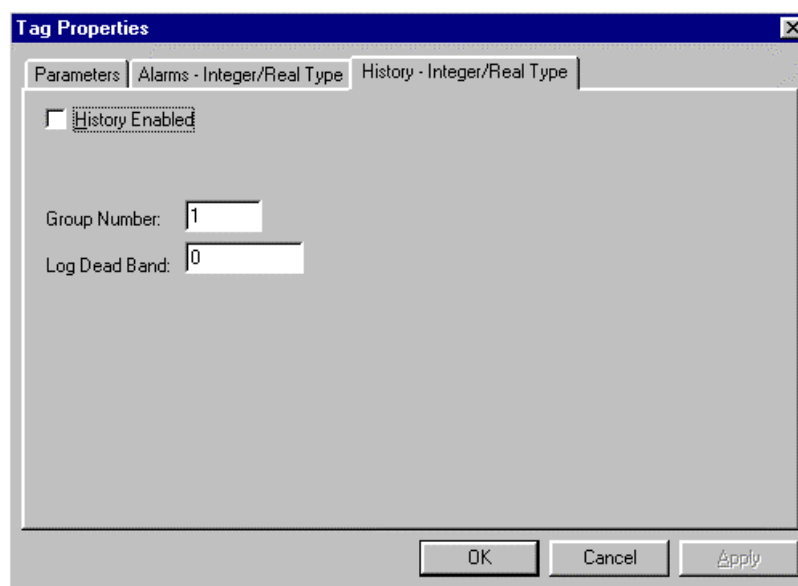
- **Off** checkbox – 當值為 0 時產生警報.
- **On** checkbox - 當值為 1 時產生警報.
- **Changed** checkbox - 當值 0 和 1 之間變化時產生警報.

II.7.9. Tag Properties – History

在這裡你可以設定啓動歷史與否, 如果有趨勢圖工作表單正在開啓, 則此功能會無效.

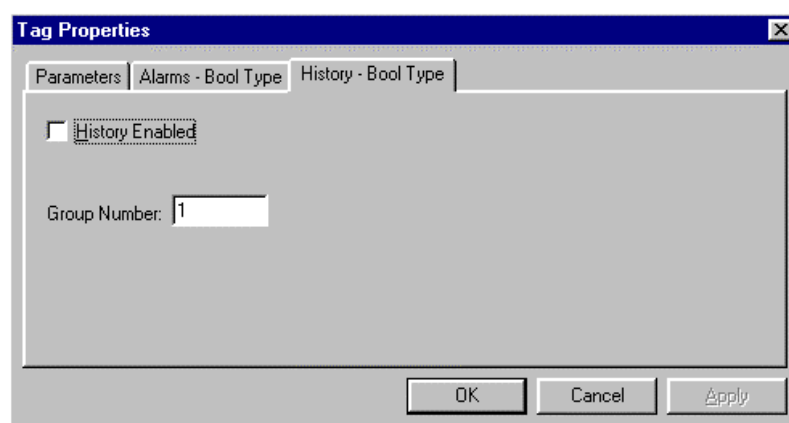
 **備註:** 字串無歷史功能. 僅能用 **Recipes** 來作儲存字串的工作.

INTEGER AND REAL TYPE HISTORY



- **History Enabled** checkbox – 啓動歷史儲存.
- **Group Number** field – 定義歷史群組.
- **Log Dead Band** field – 紀錄變化超過“不紀錄區間”的值.

BOOLEAN TYPE HISTORY



- **History Enabled** checkbox – 啓動歷史儲存.
- **Group Number** field – 定義歷史群組.

II.8. Functions List

Advantech Scripting Language has more than one hundred functions ready for use. The following is an overview of them:

II.8.1. Messaging Function for LogWin

▪ TRACE(arg)

Description: Shows the contents of *arg* in the LogWin screen. The *arg* variable can be a string constant or a string tag.

Examples:

```
TRACE("The value of the count has changed")
TRACE(DATE)
```

II.8.2. Arithmetical Functions

▪ ABS(arg)

Description: Returns the absolute value of argument.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Level	-20.153
Temp	abs(level) // temp=20.153

▪ DIV(arg1, arg2)

Description: Truncates and returns the division quotient of *arg1* by *arg2*.

Examples:

```
Div (temp, level)
Div (temp,4)
Div (4,level)
```

<u>Tag Name</u>	<u>Expression</u>
Level	5.648
Temp	2
Result	Div (level, temp) // result = 2

▪ FORMAT(arg1, arg2)

Description: Creates a formatted *string* from a number. *arg1* must be the mask ("%[0n] [flag]") and *arg2* is the number to be formatted.

Flags:

```
d, D decimal
x, X hexadecimal
o, O octal
b, B binary
f, F real
e, E scientific notation
g, G the same as F and E, but more compact
```

s, S string
 c, C ASCII character
 h, H hours
 n The number of digits to be shown.

Examples:

<u>Tag Name</u>	<u>Expression</u>	<u>Result</u>
Output[1]	format("%b", 8)	1000
Output[2]	format("%x", 255)	ff
Output[3]	format("%02X", 15)	0F
Output[4]	format("%o", 8)	10
Output[5]	format("%x", 17)	11
Output[6]	format("%f", 237.8)	237.800000
Output[7]	format("%d", level)	97
Output[8]	format(string_format, level)	97
Output[9]	format("%c", 38)	&
Output[10]	format("%c", 49)	1
Output[11]	format("%h", 37230)	10:20:30

⚠ **Note:** This function accepts the same flags after the symbol “%” that are used in the standard C function *printf()*, but only one value can be formatted in each cell.

▪ **GetBit (strTagName , strBitNumber)**

Description: Retrieves a bit from the tag.

Return Values: Error codes, as follows:

Bit value No error.
 -1 Invalid parameter.
 -2 Tag does not exist.

Example:

GetBit(“minute”,2)

▪ **MOD (arg1, arg2)**

Description: Returns the remainder of *arg1* divided by *arg2*.

Examples:

Mod (level, temp)
 Mod (level, 4)

<u>Tag Name</u>	<u>Expression</u>
Level	20
Temp	7
Result	Mod (level, temp) // result = 6

▪ **POW(arg1, arg2)**

Description: Returns the value of *arg1* (the base) raised to a power *arg2* (exponent).

Examples:

pow(base, exponent)
 pow(base,7)

pow (5,exponent)

<u>Tag Name</u>	<u>Expression</u>
Base	2
Exponent	3
Result	Pow (base, exponent) // result = 8

- **ResetBit (strTagName, strBitNumber)**

Description: Resets a bit from the tag.

Return Values: Error codes, as follows:

- 0 No error.
- 1 Invalid parameter.
- 2 Tag does not exist.

Example:

ResetBit("hour",1)

- **ROUND(arg)**

Description: Rounds the value of the *arg* argument to the next greater integer value.

Examples:

Round(level)

Round(-23.44)

<u>Tag Name</u>	<u>Expression</u>
Level	21.67
Result	Round (level) // result = 22

- **SetBit (strTagName , strBitNumber)**

Description: Sets a bit from the tag.

Return Values: Error codes, as follows:

- 0 No error.
- 1 Invalid parameter.
- 2 Tag does not exist.

Example:

SetBit("second",0)

- **SQRT(arg)**

Description: Returns the square root value of the *arg* argument.

Examples:

Sqrt(level)

Sqrt(level)

Sqrt(24)


<u>Tag Name</u>	<u>Expression</u>
Level	24

Result Sqrt (level) // result = 4.898979

- **Swap16(arg)**

Description: Swaps the two lower bytes from the tag. Returns an integer with the binary value corresponding to the swap of the two lower bytes from the tag.


Example:
Swap16(test16)

 **Note:** If the binary value of, for example, test16 is 1001111100000110, the Swap16 function returns the binary value 0000011010011111.

- **Swap32(arg)**

Description: Swaps the last two words from the tag. Returns an integer with the binary value correspondent to the swap of the two words from the tag.

Example:
Swap32(test32)

 **Note:** If the binary value of, for example, test32 is 1001111100000110000011111110000, the Swap32 function returns the binary value 00001111111100001001111100000110.

- **TRUNC(arg)**

Description: Returns the integer part of the *arg* argument.

Examples:
Trunc(level)
Trunc(-23.44)

<u>Tag Name</u>	<u>Expression</u>
Level	15.2345
Result	Trunc (level) // result = 15

II.8.3. Statistical Functions

- **AVG(arg1, arg2, ...avgN)**

Description: Returns the arithmetic average of the defined arguments.

Examples:
Avg(level, temp)
Avg(-23.44,level,temp)
Avg(12,24,32,88)

<u>Tag Name</u>	<u>Expression</u>
Level	20
Temp	40
Result	Avg (level, temp) // result = 30
Result	Avg(10,level,30,temp) // result = 25

- **MAX(arg1, arg2, ... argN)**

Description: Returns the highest value among the defined arguments.

Examples:

Max(level, temp)
 Max(-23.44, level, temp)
 Max(12,24,32,88)

<u>Tag Name</u>	<u>Expression</u>
Level	20
Temp	40
Result	Max (level, temp) // result = 40
Result	Max(10,level,30,temp,100) // result = 100

- **MIN(arg1, arg2, ... argN)**

Description: Returns the lowest value among the defined arguments.

Examples:

Min(level, temp)
 Min(-23.44,level,temp)
 Min(12,24,32,88)

<u>Tag Name</u>	<u>Expression</u>
Level	20
Temp	40
Result	Min (level, temp) // result = 20
Result	Min(10,level,30,temp) // result = 10

- **RAND()**

Description: Generates a random number in floating point between 0 and 1.

Example:

<u>Tag Name</u>	<u>Expression</u>
Result	Rand() // result = 0.104892

II.8.4. Logarithmic Functions

- **EXP(arg)**

Description: Exponentially calculates the *arg* argument ($e = 2.71828\dots$).

Examples:

Exp(level)
Exp(22)

<u>Tag Name</u>	<u>Expression</u>
Level	22
Result	Exp(level) // result = 3584912846.131592

▪ LOG(arg)

Description: Logarithmically calculates the *arg* argument ($e = 2.71828\dots$). Also known as natural logarithm.

Examples:

Log(level)
Log(22)

<u>Tag Name</u>	<u>Expression</u>
Level	22
Result	LOG(level) // result = 3.091042

▪ LOG10(arg)

Description: Logarithmically calculates the *arg* argument in the base 10.

Examples:

Log10(level)
Log10(22)

<u>Tag Name</u>	<u>Expression</u>
Level	22
Result	Log10(level) // result = 1.342423

II.8.5. Logical Functions

▪ IF (condition, true, false)

Description: Executes a conditional expression. *condition* is the expression to be tested. *true* is the result expression in case of true condition. *false* is the result expression in case of false condition (optional parameter).


Return Values: If the expression in the *condition* parameter is true (or more than zero), the *true* expression result returns, otherwise, the *false* expression. If the condition result is false (or equal to zero), or if the parameter was not declared, the tag of the Tag Name column remains unchanged.

Example:

if (tag > 20, tag/2, abs(count))

<u>Tag Name</u>	<u>Expression</u>
Account	if (account=10, 0, account+1)

Comment: If the value of the tag account = 10, account receives the value 0 (zero), otherwise, 1 will be added to its actual value.

 **Note:** The Database Spy utility cannot evaluate this function in a direct way.

- **TRUE(arg)**

Description: Verifies if the expression *arg* is true.

Return Values: Error codes, as follows:

- 1 If the expression is true.
- 0 If the expression is false.

Examples:

true (newtag)
true (a > b)

<u>Tag Name</u>	<u>Expression</u>
Cond	TRUE(cond=10)

If the value of tag cond = 10, the cond tag will receive the value 1; otherwise, it will receive 0.

- **FALSE(arg)**

Description: Verifies if the *arg* expression is false.

Return Values:

- 1 If the expression is false.
- 0 If the expression is true.

Examples:

false (newtag)
false (a > b)

<u>Tag Name</u>	<u>Expression</u>
Cond	FALSE(cond = 10)

If the value of tag cond = 10, cond will receive the value 0; otherwise, it will add 1 to its actual value.

II.8.6. Functions for Manipulating Strings

- **Asc2Str(arg1, arg2,....., argN)**

Description: Links characters in ASCII code to form a *string*.

Example:

49 is the ASCII code value for the character "1" and the number 50 is of the character "2".

<u>Tag Name</u>	<u>Expression</u>
Name	"test"

`new_string Asc2Str(test, 49, 50) // new_string = "test12"`

- **CharToValue("strTagName" , "numArray")**

Description: Converts a string to an integer array. Returns the number of characters. The trigger defines when the values must be updated.

Examples:

```
Tagstring="ABC"  
Tagnum ValueToChar("tagstring","vet[1]") => tagnum=3  
Vet[1]=65 // char'A'  
Vet[2]=66 // char'B'  
Vet[3]=67 // char'C'
```

- **CharToValueW("strTagName" , "numArray")**

Description: Same as CharToValue but using words instead of bytes.

- **NCOPY(str, n1, n2)**

Description: Returns a substring. *str* is a string or tag type string from which you want to extract a sub-string. *n1* is the initial position of the sub-string. *n2* is the number of the sub-string characters.

Return Value: String that starts in the *n1* of *str* characters and has the *n2* size.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Name	"System"
New_string	NCOPY(name, 3, 4) // new_string = "stem"

- **NUM(string)**

Description: Converts a *string* (or tag type *string*) to a numeric value.

Example:

num ("4")

<u>Tag Name</u>	<u>Expression</u>
new_tag	"4"
New_int	NUM (new_tag) // New_int = 4

- **STR(num)**

Description: Converts a numeric value (tag or value) to a string.

Example:

str (3)

<u>Tag Name</u>	<u>Expression</u>
New_tag	5
Str_n	STR (new_tag) // str_n = "5"

- **Str2Asc(arg1)**

Description: Returns the ASCII code of a character. *arg1* is a string.

Return value: Integer.

Example:

49 is the ASCII code value for the character "1".

<u>Tag Name</u>	<u>Expression</u>
Name	"1"
Num	Str2Asc(name) // num = 49

- **StrLeft (arg1, arg2)**

Description: Returns the bytes to the left of the string *arg1* (tag or value). *arg2* specifies how many bytes to return.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Strin	StrLeft("test",2) // strin = "te"
Strin	"TESTING"
New_str	StrLeft(strin,4) // new_str = "test"


- **StrLen(arg)**

Description: Returns the length in bytes of the string *arg*.

Return Value: Integer

Example:

<u>Tag Name</u>	<u>Expression</u>
Size	StrLen("test") // size = 4
Strin	"test"
Size2	StrLen(strin) // size2 = 6

 **Note:** Don't forget that for variables of the string type, the double quotation marks (") are considered characters.

- **StrLower (arg)**

Description: Converts a string specified by *arg* to lowercase letters.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Strin	StrLower("Test") // strin = "TEST"
Strin	"TESTING"
New_str	StrLower(strin) // new_str = "testing"

- **StrRChr("string, "char")**

Description: Finds a character specified by *char* in a string specified by *string*. It returns a string to the first occurrence of *char* in *string*, or "" (NULL string) if *char* is not found.

- **StrRight (arg1, arg2)**

Description: Returns the bytes to the right of the string *arg1*. *arg2* specifies how many bytes to return.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Strin	StrRight("test",2) // strin = "st"
Strin	"TESTING"
New_str	StrRight(strin,4) // new_str = "ting"

- **StrStr(arg1, arg2)**

Description: Searches the first occurrence of the string *arg2* in the string *arg1*.

Return Values: String *arg1* starting where the sequence *arg2* is found. String empty if it was not found.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Name	"test"
New_string	StrStr(name, "s") // new_string = "st"
New_string	StrStr("test", "s") // new_string = "st"


- **StrStrPos(arg1, arg2)**

Description: Searches the first occurrence of the string *arg2* in the string *arg1*.

Return Values: Integer number with the start position, or -1, if the string was not found.

Example:

<u>Tag Name</u>	<u>Expression</u>
Position	StrStrPos("test", "s") // position = 2

 **Note:** The first position of the string is considered to be the number zero position.

- **StrTrim (arg1, arg2)**

Description: Removes the white spaces of the string (or tag type string) *arg1*. *arg2* is an optional parameter. If 0 is specified for *arg2*, white space is removed from the right to the left (default). If 1 is specified, white space is removed from the left. If 2 is specified, white space is removed from the right.

Return Value: string

Examples:

<u>Tag Name</u>	<u>Expression</u>
Strin	" test "
Strin	StrTrim(strin) // strin = "test"

- **StrUpper(arg)**

Description: Converts a string specified by *arg* to uppercase letters.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Strin	StrUpper("test") // strin = "TEST"
Strin	"testing"
New_str	StrUpper(strin) // new_str = "TESTING"

- **ValueToChar("numArray", numChars)**

Description: Converts an integer array to a string. Returns the string. The *numArray* variable specifies the array to convert. The *numChars* variable defines which value to convert.

Examples:

```
Tagstring ValueToChar("vet[1]",3) => tagstring="ABC"
Vet[1]=65 // char'A'
Vet[2]=66 // char'B'
Vet[3]=67 // char'C'
```

- **ValueWToChar(("numArray", numChars)**

Description: Same as ValueToChar but using words instead of bytes.

II.8.7. Functions for Manipulating the Date and Time

- **ClockGetDate(arg)**

Description: Returns the related date with the number of elapsed seconds as a parameter. *arg* is a long integer that contains the date in seconds. The base date is 31/12/1969.

Return Value: String in the *DD/MM/AA* format.

Example:

<u>Tag Name</u>	<u>Expression</u>
Date	ClockGetDate(633000000) // data = "22/01/1990"

- **ClockGetDayOfWeek (arg)**

Description: Returns the day of the week according to the number of elapsed seconds as a parameter. *arg* is a long integer that contains the hour in seconds.

Return Value: Integers, as follows:

0	Sunday
1	Monday
2	Tuesday
3	Wednesday
4	Thursday

5 Friday
6 Saturday

Example:

<u>Tag Name</u>	<u>Expression</u>
Temp	ClockGetDayOfWeek (2999999) // temp = 3

▪ **ClockGetTime (arg)**

Description: Returns hours:minutes:seconds related with the number of elapsed seconds as a parameter. *arg* is a long integer that contains the data in seconds.

Return Value: String in the *HH:MM:SS* format.

Example:

<u>Tag Name</u>	<u>Expression</u>
Temp	ClockGetTime(633000000) // temp = "01:20:00"

▪ **DateTime2Clock(arg1, arg2)**

Description: Returns the number of seconds, beginning on 31/12/1969 until the *arg1* date and *arg2* hour.

Example:

<u>Tag Name</u>	<u>Expression</u>
num_of_sec	DateTime2Clock("10/10/1990","11:02:30") // num_of_sec = 655581750

▪ **GetClock(arg)**

Description: Returns the number of seconds counted, beginning on 31/12/1969 up to the current date and time.

Example:

<u>Tag Name</u>	<u>Expression</u>
num_of_sec	GetClock(0) // num_of_sec = 862252573

▪ **Hour2Clock(arg)**

Description: Converts a time specified in number of seconds, where *arg* is a string (or tag type string) with a time (*HH:MM:SS*) to be converted.

Return Value: Integer value of the time in seconds.

Examples:

<u>Tag Name</u>	<u>Expression</u>
s[1]	Hour2Clock("00:01:00") // s[1] = 60
S[2]	Hour2Clock("10:00:00") // s[2] = 36000
new_time	"10:20:30"
s[3]	Hour2Clock(new_time) // s[3] = 37230

- **SetSystemDate (arg)**

Description: Modifies the system date in your computer. *arg* is a string (or tag type string) that contains the desired date.

Example:

<u>Tag Name</u>	<u>Expression</u>
	SetSystemDate("22/09/1995")
new_date	"23/09/1996"
	SetSystemDate(new_date)

- **SetSystemTime (arg)**

Description: Modifies the system time in your computer. *arg* is a tag or constant of the type string with the desired hour.

Example:

<u>Tag Name</u>	<u>Expression</u>
	SetSystemTime("12:00:00")
new_time	"23:09:19"
	SetSystemDate(new_time)

II.8.8. Trigonometric Functions

- **ACOS(arg)**

Description: Returns the arc-cosine value of *arg*.

Examples:

<u>Tag Name</u>	<u>Expression</u>
value_acos	ACOS(0.997495) // value_acos = 0.070796
value	0.707107
value_acos	ACOS(value) // value_acos = 0.785398

- **ASIN(arg)**

Description: Returns the arc-sine value of *arg*.

Examples:

<u>Tag Name</u>	<u>Expression</u>
value_asin	ASIN(0.997495) // value_asin = 1.500000
Value	0.707107
value_asin	ASIN(value) // value_asin = 0.785398

- **ATAN(arg)**

Description: Returns the arc-tangent value of *arg*.

Examples:

<u>Tag Name</u>	<u>Expression</u>
value_atan	ATAN(14.101420) // value_atan = 1.500000
Value	2

value_atan ATAN(value) // value_atan = 1.107149

▪ COS(arg)

Description: Returns the cosine of *arg* (*arg* in radians).

Examples:

<u>Tag Name</u>	<u>Expression</u>
value_cos	COS(1.5) // value_cos = 0.070737
Angle	PI()/4
value_cos	COS(angle) // value_cos = 0.707107

▪ COT(arg)

Description: Returns the co-tangent of *arg* (*arg* in radians).

Examples:

<u>Tag Name</u>	<u>Expression</u>
value_cotan	ATAN(1.5) // value_cotan = 0.982794
Angle	PI()/4
value_cotan	ATAN(angle) // value_cotan = 0.665774

▪ PI()

Description: Returns the value of the PI numeric constant.

Return Value: (= 3.141593) with seven decimal places.

Example:

<u>Tag Name</u>	<u>Expression</u>
value_pi	PI() // value_pi = 3.141593

▪ SIN(arg)

Description: Returns the sine of *arg* (*arg* in radians).

Examples:

<u>Tag Name</u>	<u>Expression</u>
value_sin	SIN(1.5) // value_sin = 0.997495
Angle	PI()/4
value_sin	SIN(angle) // value_sin = 0.707107

▪ TAN(arg)

Description: Returns the tangent of *arg* (*arg* in radians).

Examples:

<u>Tag Name</u>	<u>Expression</u>
value_tan	TAN(1.5) // value_tan = 14.101420
Angle	PI()/4
value_tan	TAN(angle) // value_tan = 1.000000

II.8.9. Functions for Opening and Closing Windows

- **OPEN(arg, x1, y1, x2, y2)**

Description: Opens a screen or group of screens of an application during the execution. *arg* is the name of the screen (default extension is .SCR) or group of screens (extension .SG). It can be a tag or constant of the string type. *x1, y1, x2, y2* - Optional parameters that define the initial coordinates of the window to be opened

Return Values:

0 Function executed successfully.
1 Function cannot be executed.

Examples:

"screenlb.scr" is the name of a screen created in the Graphical Interface, so:

<u>Tag Name</u>	<u>Expression</u>
Status	OPEN("screenlb") // it is the same as OPEN("screenlb.scr")

- **CLOSE(arg)**

Description: Closes the window specified by *arg*.

Example:

<u>Tag Name</u>	<u>Expression</u>
	CLOSE("screenlb")

⚠ **Caution:** When you open a window of the Replace style, it automatically closes the windows with Replace and Popup attributes that intercept the new window. In this case, it is not necessary to call the CLOSE(arg) function.

II.8.10. Security System Functions

▪ **CreateUser(Username, Group, Password)**

Description: Adds a user in the Security System. *UserName* is the tag or value of the string type for the user to be inserted in a group of the Security System. *Group* is the tag or value of the string type of the group in the Security System. *Password* is the tag or value of the string type of the user's password.

Return values:

- 0 Success
- 1 Invalid number of parameters
- 2 Wrong parameter type
- 3 User already exists.
- 4 Group does not exist.
- 5 It is not possible to safely write the data.
- 6 It is not possible to use the CreateUser function.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Status	CreateUser ("John","Projects","8763")
UserName	"John"
Group	"Projects"
Password	"8763"
Status	CreateUser (UserName, Group, Password)

▪ **RemoveUser(Username)**

Description: Removes a user as specified by *UserName* from the Security System.

Return values:

- 0 Success
- 1 Invalid number of parameters
- 2 Wrong parameter type
- 3 User does not exist.
- 4 It is not possible to safely write the data.

Example:

<u>Tag Name</u>	<u>Expression</u>
	RemoveUser ("John")
UserName	"John"
	RemoveUser (UserName)


II.8.11. Functions for Activating Modules

▪ **ShutDown()**

Description: Closes all of the active runtime programs of *Studio*.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	ShutDown()

 **Caution:** This function does not close the configuration application, the Database, or LogWin.

▪ **AppActivate (arg1, arg2)**

Description: Activates an application. *arg1* is a string with the application title and *arg2* is an optional parameter where you can specify an integer of the command activation. See the Windows documentation about the following options:

- 0 SW_HIDE
- 1 SW_SHOWNORMAL
- 2 SW_SHOWMINIMIZED
- 3 SW_SHOWMAXIMIZED
- 4 SW_SHOWNOACTIVATE
- 5 SW_SHOW
- 6 SW_MINIMIZE
- 7 SW_SHOWMINNOACTIVE
- 8 SW_SHOWNA
- 9 SW_RESTORE (default) Must be 9.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	AppActivate("notepad - (untitled)")

▪ **ApplsRunning (arg)**

Description: Verifies if an application is being executed. *arg* specifies a tag or string type with the application title.

Return values:

- 0 Application is running.
- 1 Application was not found.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	ApplsRunning ("Microsoft Word - test.doc")

▪ **AppPostMessage (arg1, arg2, arg3)**

Description: Sends a message to an application. *arg1* is a tag or string type value with the application title. *arg2* is an integer with the Windows message wParam, or tag (or value) of the string type with the following values:

- "MINIMIZE"
- "MAXIMIZE"
- "RESTORE"
- "CLOSE"

arg3 is the lParam of the Windows message.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	AppPostMessage("Calculator", "CLOSE", 0)

- **AppSendKeys (arg1, arg2, ...argN)**

Description: Sends keyboard codes to the foreground Windows application. *arg1* is a tag or string type value with the commands to be sent or a tag or integer value with the keyboard codes to be sent. *arg2* is the same as *arg1*, but it has a delay of 200ms between the sending of *arg1* and *arg2*.

⚠ Caution: To send a code equal as the keyboard commands ALT, CTRL, or SHIFT, use <CTRL> or <SHIFT> in the text. To send the < character, send <<.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Status	WinExec("write.exe c:\windows\README.WRI")
	AppActivate("Write - README.WRI")
	AppSendKeys("<ALT>FP") //for Windows version in English.
	Wait(1000)
	AppSendKeys("<ENTER>")
	Wait(1000)
	AppSendKeys("<ESC>")
	AppPostMessage("Write - README.WRI", "CLOSE")

Special Keyboard Commands: You can use the following special keyboard commands. To use a command, put the text between the brackets (<>).

BACKSPACE
BREAK
CAPSLOCK
DELETE
DEL
DOWN
END
ENTER
ESCAPE
ESC
F1F12
HOME
INSERT
LEFT
NUMLOCK
PGDN
PGUP
PRTSC
RIGHT
TAB
UP

- **CleanReadQueue()**

Description: Removes all reading messages in the communication drivers.

Example:

<u>Tag Name</u>	<u>Expression</u>
	CleanReadQueue ()

▪ **CloseSplashWindow()**

Description: Closes *Studio* Splash window.

▪ **DisableMath()**

Description: Stops the execution of the mathematical task until the call of the EnableMath().

Example:

<u>Tag Name</u>	<u>Expression</u>
	DisableMath ()

▪ **EnableMath()**

Description: Enables the execution of the mathematical task after the use of the DisableMath().

Example:

<u>Tag Name</u>	<u>Expression</u>
	EnableMath ()

▪ **ExitWindows (arg)**

Description: Exits Windows, where *arg* is an integer from 0 to 2, as follows:

- 0 Reboot Windows.
 - 1 Log off.
 - 2 Shut down.
- Force (unsaved data is lost).

Example:

<u>Tag Name</u>	<u>Expression</u>
	ExitWindows(1)

▪ **IsScreenOpen (arg)**

Description: Verifies if a *Studio* screen is opened in the execution. *arg* is a tag or value of the string type with the name of the screen.

Return values:

- 0 If the screen is not open.
- 1 If the screen is open.

Examples:

<u>Tag Name</u>	<u>Expression</u>
-----------------	-------------------

actual_screen	IsScreenOpen("menu.scr")
Status	"menu"
	IsScreenOpen(actual_screen)

- **IsViewerInFocus()**

Description: Verifies if that the Viewer task is in focus.

Return Values: Error codes, as follows:

- 1 Viewer has the focus.
- 0 Viewer does not have the focus.

- **KeyPad("strTagName")**

Description: Calls the Virtual Keypad and allows the user to set a value to the tag specified as argument of the function.

Parameters:

"strTagName": Tag which will store the value typed by the Virtual Keypad.

Return Values:

- 0 - Success
- 1 - User pressed the key ESC from the Keypad (tag value didn't change)
- 2 - Invalid parameter

Examples:


KeyPad("TagSetPoint")

- **LogOff()**

Description: Disables the *Studio* Log On/Off utility. When a user of a determined group logs off, the **Guest** group is always activated.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	LOGOFF()

 Note: Status receives the return value.
--

- **LogOn(arg1, arg2)**

Description: Activates the *Studio* Log On/Off utility where *arg1* is the username and *arg2* is the user's password. Both arguments are optional.

Examples:

<u>Tag Name</u>	<u>Expression</u>
status[1]	LOGON("Smith")
	LOGON("Smith","senha")
LOGON()	

- **Math(arg)**

Description: Executes the math worksheet. (*Background Task* on the *Runtime Tasks* tab in the *Project Status* window must be running.) *arg* is an integer with math worksheet number.

Example:

<u>Tag Name</u>	<u>Expression</u>
	Math(5) // executes math 5

▪ **NoInputTime()**

Description: Returns the time from the last keyboard action.

Return value: Integer

Examples:

<u>Tag Name</u>	<u>Expression</u>
Number	NoInputTime()

▪ **RECIPE (arg)**

Description: Activates the recipe functions. *arg* is the tag or string type value with a specific format, depending on the operation to be accomplished. The string format is operation:configuration_file. The valid values of the operation are:

save	Save values operation.
load	Load values operation.
delete	Delete file operation.
init	Initialize file operation.

Return Values: Error codes, as follows:

0	No errors.
1	The tag is numeric.
2	Expression doesn't contain ":".
3	Previous command to the invalid ":".
4	Task not found by the system.
5	Disk error.

⚠ Caution:

The recipes work with two kinds of files: configuration files and data files. The configuration file contains the tag declarations that form a part of a recipe worksheet (or group) that has the .RCP extension. The name of the data file is defined in the report worksheet in the Output File camp; that will contain the report (with the .OUT extension).

Also, *Background Task* on the *Runtime Tasks* tab in the *Project Status* window needs to be running in order to execute the recipe functions.

▪ **REPORT(arg)**

Description: Activates the report functions. *arg* is the tag or string type value with a specific format that contains the command for a report task. The string format is operation:configuration_file. The valid values to the operation are:

disk	Disk load operation.
------	----------------------

prn Sends values to the printer.

Return Values: Error codes, as follows:

- 0 No error.
- 1 If the tag is numeric..
- 2 Expression doesn't contain ":".
- 3 Previous command to the invalid ":".
- 4 Task not found by the system.
- 5 Disk error.

⚠ Cautions:

The reports work with two kinds of files: configuration files and data files. The configuration file contains the tag declarations and strings that form a report model (with the .RPT extension). The name of the data file is defined in the report worksheet in the Output File camp; that will contain the report (with the .OUT extension).

Also, *Background Task* on the *Runtime Tasks* tab in the *Project Status* window needs to be running in order to execute the report functions.

▪ **SetAppPath (arg)**

Description: Points the subdirectories **HST** and **ALARM** for the current application. *arg* is a value of the string type with the name of the new application directory.

Return values:

- 0 Failure
- 1 Success

Examples:

<u>Tag Name</u>	<u>Expression</u>
	SetAppPath ("C:\INDUSOFT")
actual_dir	"C:\INDUSOFT"
	SetAppPath (actual_dir)

▪ **SetViewerInFocus()**

Description: Sets the focus to the Viewer task.

▪ **ViewerPostMessage (arg1, arg2, arg3)**

Description: Sends an internal message to Viewer. *arg1* is a tag or string type value with the screen title. *arg2* is an integer with the wParam of the Windows message. *arg3* is the lParam of the Windows message.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	ViewerPostMessage("demo.scr", "CLOSE",0)

▪ **Wait (arg)**

Description: Interrupts the execution for *arg* milliseconds.

⚠ **Caution:** The wait function may only be used in Math worksheets. However, **IT IS DANGEROUS** to use this function anywhere in your application. Wait() pauses the application, any information coming into the application during a wait is ignored.

Example:

<u>Tag Name</u>	<u>Expression</u>
	Wait (200) // interrupts the execution for 200 ms.

▪ **WinExec (arg1, arg2)**

Description: Activates an external program to *Studio*. *arg1* is a tag or constant of the string type that contains the path of the executable file. *arg2* is optional and specifies a numeric value with the initial state of the new application:

- 0 Hides the application and gives control to another one.
- 1 Activates and displays the application (default).
- 2 Activates the application and displays it as an icon.
- 3 Activates the application and maximizes it.
- 4 Shows the application with its recent size. The application is still active.
- 7 Shows the application as an icon. The application is still active.

Return Values:

- 0 The operation was not correctly executed.
- 1 The operation was successfully executed.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Status[1]	WinExec("write.exe mytext.wri") // edits the text file mytext.wri
Status[2]	WinExec("\\INDUSOFT\\BIN\\pserver.exe myprint.txt") // prints the in disk text file myprint.txt . WinExec("\\INDUSOFT\\BIN\\logon.exe /OFF) // deactivates the LOGON utility of Studio .

▪ **SendKeyObject (numEvent, strMainKey, numShift, numCtrl, numAlt, strTargetScreen)**

Description: Sends key codes to objects on the open screens. The "Command" dynamics from the objects can be triggered by this function.

Parameters:


numEvent specifies the code of the key event, as follows:

- 0 On Down
- 1 While Down
- 2 On Up

strMainKey specifies the tag or string with of the key to be sent to the object. The following are acceptable: "F1" .. "F20", "+", "-", "/", "*", "HOME", "END", "INSERT", "DELETE", "DOWN", "UP", "LEFT", "RIGHT", "PAGEUP", "PAGEDOWN", "SPACE", "RETURN", "BACKSPACE", "ESCAPE", "A" .. "Z". This parameter is case insensitive.

numShift indicates that the "Shift" key code will be sent too. It is an optional parameter.

numCtrl indicates that the "Ctrl" key code will be sent too. It is an optional parameter.
numAlt indicates that the "Alt" key code will be sent too. It is an optional parameter.
strTargetScreen specifies the screen, which will receive the keys code. This is an optional parameter.

 **Notes:**
 The *numEvent* parameter defines if the function will execute the expressions configured in the "On Down", "On While", or "On Up" of the objects "Command" dynamic. It requires special attention to the "On While" event. Each time that the `SendKeyObject()` function is executed, it executes the expressions configured in the "On While" sheet (from the objects "Command" dynamic) just at once.

The parameters "numShift" , "numCtrl" and "numAlt" and "strTargetScreen" are optional. However, if one of them will be used, the other will must to be configured too.

Return Values:

- 2 Memory allocation error
- 1 Viewer is not running
- 0 Invalid parameter(s)
- 1 Success

Examples:


Tag Name	Expression
Status Down"	<code>SendKeyObject(0 , "F1") // Sends the "F1" key code. Event = "On</code>
Status	<code>SendKeyObject(2 , "V" , 1 , 0 , 0) // Sends the "Shift+V" key code. Event = "On Up"</code>

▪ **SetViewerPos(numLeft, numTop, numWidth, numHeight)**

Description: Sets the Viewer window position and/or size. This function is especially useful when using dual monitor feature from operating system.

Parameters:

- numLeft* specifies the horizontal coordinate, in pixels, of the left border from the Viewer window.
- numTop* specifies the vertical coordinate, in pixels, of the top border from the Viewer window.
- numWidth* (optional parameter) specifies the screen window width, in pixels.
- numHeight* (optional parameter) specifies the screen window height, in pixels.

 **Note:** When the optional parameters (*numWidth* and *numHeight*) are omitted, the Viewer window will get the size from the application resolution.

Return values:

- 0 Error
- 1 Success

Examples:

Tag Name	Expression
ErrorCode	<code>SetViewerPos(TagLeft , TagRight, TagWidth, TagHeight)</code>
ErrorCode	<code>SetViewerPos(0 , 0 , 800 , 600)</code>
ErrorCode	<code>SetViewerPos(0 , 0)</code>

- **StartTask(strTaskName)**

Description: Starts an *Indusoft* Task. *strTaskName* is a string or string tag with the name of the task to be started. It may be one of the following tasks:

BGTASK – Background Tasks
VIEWER – Viewer
DBSPY – Database Spy
LOGWIN – LogWin
DRIVER – Driver
UNIDDECL – DDE client
UNINDDE – DDE server
UNIODBC – ODBC
TCPSEVER – TCP/IP Server
TCPCLIENT – TCP/IP Client
OPCCLIENT – OPC

Examples:

StartTask("BGTASK")
StartTask("VIEWER")

- **IsTaskRunning(strTaskName)**

Description: Verifies if an *Indusoft* task is running. *strTaskName* is a string or string tag with the name of the task to be started. It may be one of the tasks used in StartTask function.

Return Values:

0 Task is not running
1 Task is running

Examples:

IsTaskRunning("BGTASK")
IsTaskRunning ("VIEWER")

- **EndTask(strTaskName)**

Description: Stops an *Indusoft* task. *strTaskName* is a string or string tag with the name of the task to be stopped. It may be one of the tasks used in StartTask function.

Examples:

EndTask("BGTASK")
EndTask("VIEWER")
EndTask("DRIVER")

II.8.12. Functions for Manipulating Files

- **DeleteOlderFiles(strPath, strMask, strDate)**

Description: Delete the files older than a date that matches to the configured mask from the configured path.

Parameters:

strPath: Path of the files which will be deleted;

strMask: Mask of the files which will be deleted;

strDate: Limit date – only files older than this date will be deleted.

Return Values:

0 – None file deleted;

1 – One file deleted;

2 – Two files deleted;

.


.

<N> - <N> files deleted;

Examples:

DeleteOlderFiles("c:\MyApp\Hst", "*.hst", "04/05/2001")

DeleteOlderFiles(TagStrPath, TagStrMask, TagStrDate)

 **Note:** The third parameter (StrDate) must be configured according to the date format set to the application (MDY, DMY, etc) as such as the separator (/ , : , . , etc).

▪ **FileCopy(arg1, arg2)**

Description: Copies the file *arg1* to *arg2*.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	FileCopy("file.txt", "file.bak")

▪ **FileDelete(arg)**

Description: Deletes the file specified by *arg*.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	FileDelete("file.txt")

▪ **FileLength (arg)**

Description: Returns the file size for the file specified by *arg*.

Return values:

0 Fail

Otherwise, returns the size of the file (float).

Examples:

<u>Tag Name</u>	<u>Expression</u>
Length	FindLength("Text.txt")
Filename	"Text.txt"
Length	FindLength(filename)

- **FileRename(arg1, arg2)**

Description: Renames the file *arg1* with a new name specified by *arg2*.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	FileRename("file.txt","new_file.txt")

- **FindFile(arg)**

Description: Verifies that the indicated *arg* files exist.

Example:

<u>Tag Name</u>	<u>Expression</u>
n_files	FindFile("*.hst")

- **FindPath(strPath)**

Description: Verifies if a directory exist.

Parameters:

strPath: Path which will be searched.

Return Values:

0 – Path not found;

1 – Path found.

Examples:

GetPath("c:\Test")

GetPath(TagStrPath)

- **Print(arg)**

Description: Prints an ASCII file.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	PRINT("file.txt")

- **RDFilen("filename", "path", "mask", "ChangeDir")**

Description: Returns a user-selected filename. *filename* is a tag of the string type. This tag receives the filename the user chooses. *path* is the path (including subdirectories) for search and file selection. *mask* is the tag or string that contains the mask (options or filters) of the search. It can contain the characters "*" and "?" for generic searches. Finally, *changeDir* is the tag or value of the integer type. If 0 does not allow change directory, of other value does.

Return values:

- 0 Success
- 1 One of the parameters is not a string.
- 2 Parameter 1 contains an invalid tag name.
- 3 The user canceled the operation. Must be 3.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Status	RDFilen (filename, "C:\InduSoft", "*.txt")
Path	"C:\INDUSOFT"
Masc	"*.txt"
Status	RDFilen (filename, path, masc)

- **GetFileAttributes("strFileName")**

Description: Returns attributes for a specified file. The attributes can be one or more of the following values (in hexadecimal):

error = -1 (decimal)
 READONLY = 0x00000001 [bit 0]
 HIDDEN = 0x00000002 [bit 1]
 SYSTEM = 0x00000004 [bit 2]
 DIRECTORY = 0x00000010 [bit 4]
 ARCHIVE = 0x00000020 [bit 5]
 NORMAL = 0x00000080 [bit 7]
 TEMPORARY = 0x00000100 [bit 8]

- **GetFileTime("strFileName" , strNumber)**

Description: Returns a string with date and/or time of a file. *strNumber* identifies the return of the function as follows:

0 returns the date and time from the file
 1 returns only the file date
 2 returns only the file time

- **GetLine(strFileName , strSeqChar, "strStoreTag", numOptCase, "numOptOverflowTag")**

Description: Searches a sequence of characters (string) in an ASCII file and stores (in a string tag) the contents of the whole line where the sequence of characters has been found. The function searches just the first occurrence of the string in the ASCII file.

Parameters:

strFileName - String or tag string with the path and name of the ASCII file where the sequence of chars is going to be searched.

strSeqChar - String or tag string with the sequence of chars to be found in the ASCII file.

strStoreTag - Name of the tag (string type), which will receive the contents of the whole line, where the sequence of chars has been found. If the string is not found in the file, this tag will receive null value.

numOptCase (Optional parameter) - Numerical value or numerical tag with case sensitive settings (0 = No case sensitive [Default]; 1 = Case sensitive).

numOptOverflowTag (Optional parameter) - Name of the tag (integer type), which will receive the result of overflow verification. The line has more than 255 chars (0 = OK, 1 = Overflow). This flag tag checks just the first occurrence of the string in the ASCII file.

Return values:

-7 Invalid number of parameters (this function should have more than 2 parameters and less than 6).

- 6 Invalid *numOptOverflowTag* parameter.
- 5 Invalid *numOptCase* parameter.
- 4 Invalid *strStoreTag* parameter.
- 3 Invalid *strSeqChr* parameter.
- 2 Invalid *strFileName* parameter.
- 1 ASCII File has not been found.
- 0 String has not been found in the target ASCII file.
- N* Amount of lines where the "sequence of characters" has been found from the target ASCII file.

Examples:

```

Tag Name Expression
ErrorCode GetLine( "c:\Settings.txt" , "Studio" , "TagLine")
ErrorCode GetLine( TagPath , TagSeqChr , "TagLine2" )
ErrorCode GetLine( "c:\Settings.txt" , TagSeqChr , "TagLine2" , 0)
ErrorCode GetLine( TagPath , "Studio" , "T+C52agLine2" , 0 , "TagOverflow")

```


II.8.13. Functions for Printing Graphical Screens

- **PrintWindow(arg)**

Description: Prints any application screen (.SCR extension). This screen can be utilized and opened by the Viewer or not. If it is in a disk screen, it will be loaded to the memory, actualized with the tag's values and curves, and printed. This operation does not interfere with the screens in use in the Viewer. *arg* is the tag or value with the screen name to be printed (with or without extension).

Examples:

<u>Tag Name</u>	<u>Expression</u>
Status	PrintWindow("screen.scr")
Status	PrintWindow("screen")

 **Caution:** You can use the PrintWindow() function to print reports in graphical format.

II.8.14. Functions for Text Translations

- **EXT(arg)**


Description: Translates text in the application. *arg* is a tag or value of the string type that contains the text to be translated.

Example:

<u>Tag Name</u>	<u>Expression</u>
Output	EXT("Text") // output is a <i>string</i> tag that will receive the translation of "Text", according to the translation file.

- **SetTranslationFile(filename)**

Description: Translates files. The system uses the translation file and changes all objects with text outputs of the application for its translation. *filename* is the tag or value of the string type that contains the translation filename.

 **Caution:** You must have a translation file in the translation file utility.

Return values:

- 0 Success
- 1 Invalid number of parameters
- 2 Wrong parameter type
- 3 Translation file could not be opened or localized.

Examples:

<u>Tag Name</u>	<u>Expression</u>
Status	SetTranslationFile ("trans1.TRA")
Filename	"trans1.TRA"
Status	SetTranslationFile (filename)

II.8.15. Multimedia Functions

- **Play(arg)**

Description: Plays the .WAV file passed as a parameter.

Example:

<u>Tag Name</u>	<u>Expression</u>
Status	Play("songs.wav")

II.8.16. System Information

- **DbVersion()**

Description: Returns the database version number of the current application.

Example:

<u>Tag Name</u>	<u>Expression</u>
Version_db	DbVersion () // version_db = 173

- **GetAppHorizontalResolution()**

Description: Gets the value from the .app file, section [Info]. Returns the value on [Info], it does not test the Windows configuration.

Example:

[Info]
AppResolution=640 480

- **GetAppVerticalResolution()**

Description: Gets the value from the .app file, section [Info]. Returns the value on [Info], it does not test the Windows configuration.

Example:

[Info]
AppResolution=640 480

- **GetComputerName()**

Description: Returns the local computer name.

- **GetHardkeyModel()**

Description: Returns the name of your hardkey model.

Return value: Returns a string with the hardkey model name. If none is returned, a hardkey not installed or found. Otherwise, returns a string with the hardkey model name.

Example:

<u>Tag Name</u>	<u>Expression</u>
Hardkey_mod	GetHardkeyModel () // hardkey_mod =
InduSoft Full Version	

⚠ Caution: For the correct execution of this function, you must do the hardkey installation first.

- **GetHardkeySN()**

Description: Returns the serial number of the hardkey.

Return value: Returns a string with the serial number of the hardkey. If 0 is returned, the hardkey not installed or found. Otherwise, returns a string with the hardkey serial number.

Example:

<u>Tag Name</u>	<u>Expression</u>
Hardey_num	GetHardkeysn() // hardkey_num= 120.745

⚠ Caution: For the correct execution of this function, you must do the hardkey installation first.

- **GetProductPath()**

Description: Returns the path to the *Studio* directory.

- **GetOS()**

Description: Returns the operating system, as follows:

0	Windows 3.1x
1	Windows 95
2	Windows NT

Example:

<u>Tag Name</u>	<u>Expression</u>
Os_version	GetOS () // Os_version = 2

- **GetPrivateProfileString()**

Description: Reads .ini files.

- **GetTickCount()**

Description: Returns the current value of the clock ticks counter.

Return Value: Integer with the milliseconds counted by the clock for each initialization of the operational system.

Example:

<u>Tag Name</u>	<u>Expression</u>
num_of_ms	GetTickCount() // num_of_ms will receive the // counted milliseconds since the computer initialization.

- **InfoAppAlrDir()**

Description: Returns the alarm directory of the current application.

Example:

<u>Tag Name</u>	<u>Expression</u>
Alr_hst_path	InfoAppAlrDir() // alr_hst_path = "D:\INDUSOFT\TEST\alarm\"

- **InfoAppDir()**

Description: Returns the application's current directory.

Example:

<u>Tag Name</u>	<u>Expression</u>
Curr_appl	InfoAppDir () // curr_appl = "D:\INDUSOFT\TEST"

▪ **InfoAppHstDir()**

Description: Returns the data historic directory of the current application.

Example:

<u>Tag Name</u>	<u>Expression</u>
Hst_path	InfoAppHstDir () // hst_path = "D:\INDUSOFT\TEST\hst"

▪ **InfoDiskFree(arg)**

Description: Returns disposable free space in the disk.

Example:

<u>Tag Name</u>	<u>Expression</u>
free_bytes	InfoDiskFree ("C") // free_bytes = 60604416.000000

▪ **InfoResources(arg)**

Description: Returns the Window's disposable resources. On NT, only valid parameter with value "3" (available memory). Others are not used. *arg1* is an integer as follows:

- 0 System functions
- 1 GDI functions
- 2 USER functions
- 3 Memory. Must be 3.

Examples:

<u>Tag Name</u>	<u>Expression</u>
System	InfoResources (0) // system = 76.000000 %
GDI	InfoResources (1) // GDI = 76.000000 %
USER	InfoResources (2) // USER = 80.000000 %
Memory	InfoResources (3) // memory = 16150528.000000 bytes

▪ **IsActiveXReg(numType , strProgIDorFileName)**

Description: Returns if the ActiveX Control is registered

Parameters:

numType: 0 - Verified by ID; 1 - Verified by FileName

strProgIDorFileName: ID or FileName of ActiveX

Return Values:

- 0 – ActiveX is registered;
- 1 – ActiveX is not registered.

Examples:

IsActiveXReg(0, "ActiveXName")

IsActiveXReg(1, "e:\winnt\system\myactive.ocx")

- **NoInputTime()**

Description: Returns the time from the last keyboard action.

Return value: Integer.

Example:

<u>Tag Name</u>	<u>Expression</u>
Number	NoInputTime ()

- **ProductVersion()**

Description: Returns the *Studio* number version.

Example:

<u>Tag Name</u>	<u>Expression</u>
Version	ProductVersion() // version = 1.130000

- **SetAppAlarmPath()**

Description: Sets the current alarm application path.

- **SetAppHstPath()**

Description: Sets the current host application path.

- **SetDateFormat()**

Description: Sets the separator and date format (DMY, DYM, MDY, MYD, YDM, or YMD).

Return Value: Error codes, as follows:

0	No error.
1	Invalid parameter.

II.8.17. Database Access Functions

- **ForceTagChange(arg1, arg2)**

Description: In some cases, you may want to write a value in a tag, forcing the system to act as if it were a new value. This function forces the value *arg2* on the tag *arg1*.

Example:

<u>Tag Name</u>	<u>Expression</u>
	ForceTagChange("tagcount", 100)

II.8.18. Loops

- **FOR(first_value, last_value, step)**

Description: Implements execution steps. *first_value* is a tag, numerical value, or expression with the first step of the variable. *last_value* is a tag, numerical value, or expression with the last step of the variable. *step* is a tag, numerical value, or expression with the incremental step for the variable value.


Return value: Numerical value.

Examples:

<u>Tag Name</u>	<u>Expression</u>
J	FOR (1, tag_test, 1)
Temperat[j]	count / j Next

- **NEXT**

Description: Points to the next increment.

 **Note:** Any FOR function must have its corresponding NEXT function.

II.8.19. ODBC Functions

- **ODBCOpen(DSN, User, Password, TableName, Filter, Sort)**

Description: Opens a connection to a database. This function returns a handler that should be used in subsequent calls to the ODBC functions. After calling this function, no register has been read from the database yet. You need to bind the columns and call the ODBCQuery function to retrieve the first register.


Parameters: The ODBCOpen function accepts the following arguments:

DSN – Data Source Name (string)
User – User name (string)
Password - Password (string)
TableName - Database table name (string)
Filter – SQL WHERE clause (string)
Sort – SQL ORDER BY clause (string)

Returns

On success, this function returns a handler greater than zero that identifies the database. Each database opened by this function receives a different handler. On fail, this function returns:

- 1 Invalid parameter. Every parameter should be string.
- 2 *DSN* or *TableName* contain an empty string.

 **Note:** This function doesn't open the database itself. It just creates a handle to manipulate the database. To open the database, you need to bind the columns, and call the ODBCQuery function.

▪ **ODBCClose(Handler)**

Description: Closes a connection to a database. *Handler* is a handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns 1, indicating an invalid handler.

▪ **ODBCBindCol(Handler , ColName, ColType, TagName)**

Description: Binds a column to a tag. Every time you finish making the columns binding, you need to call the ODBCQuery function.

Parameters: The ODBCBindCol function accepts the following arguments:

Handler – Handler returned by the ODBCOpen function (integer)

ColName - Database column name (string)

ColType – SQL data type (string). It may be one of the following types:

- SQL_BIT
- SQL_TINYINT
- SQL_LONGVARCHAR
- SQL_CHAR
- SQL_VARCHAR
- SQL_DECIMAL
- SQL_NUMERIC
- SQL_DATE
- SQL_TIME
- SQL_TIMESTAMP
- SQL_DOUBLE
- SQL_REAL
- SQL_SMALLINT
- SQL_INTEGER

TagName – Name of the tag bound to the column (string).

Returns:

On success, this function returns 0.

On fail, this function returns:

1 – Invalid handler.

2 – Invalid parameter type.

3 – One of the parameters has an empty string

4 – ColType contains an invalid type

- **ODBCUnbindCol(Handler , ColName)**

Description: Unbinds a column from a tag. *Handler* is the handler returned by the ODBCOpen function (integer) and *ColName* is the database column name (string).

Returns

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Invalid parameter type.
- 3 – Column not bound

- **ODBCSetFilter(Handler , Filter)**

Description: Constrains the records *InduSoft* selects. This is useful for selecting a subset of records, such as "all salespersons based in California" ("state = 'CA'"). Remember to call ODBCQuery after calling this function. *Handler* is the handler returned by the ODBCOpen function (integer) and *Filter* is a SQL WHERE clause (string).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Invalid parameter type.

- **ODBCSetSort(Handler, Sort)**

Description: Sorts the records *InduSoft* selects. You can use this feature to sort the records on one or more columns. Remember to call ODBCQuery after calling this function. *Handler* is the handler returned by the ODBCOpen function (integer) and *Sort* is a SQL ORDER BY clause (string).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Invalid parameter type.

- **ODBCQuery(Handler)**

Description: Retrieves the first register after opening and binding the column. If you modify the column binding, or modify the filter and sort, you need to call this function again. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – No columns bound.
- 3 – Couldn't open database.
- 4 – Can't restart database.
- 5 – Query error.

▪ **ODBCInsert(Handler)**

Description: Inserts a new register to the database. *InduSoft* will use the values of the tags bound by the ODBCBindCol function to create the new register. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – Insert error.

▪ **ODBCDelete(Handler)**

Description: Deletes the current register. After a successful deletion, you need to explicitly call one of the Move functions in order to move off the deleted record. *Handler* is a handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – Delete error.

▪ **ODBCUpdate(Handler)**

Description: Updates the current register. *InduSoft* will use the values of the tags bound by the ODBCBindCol function to update the current register. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:
1 – Invalid handler.
2 – Database not open.
3 – Update error.

▪ **ODBCMove(Handler, Offset)**

Description: Moves the current record pointer within the record set, either forward or backward. If you pass a value of 0 for *Offset*, ODBCMove refreshes the current record. *Handler* is the handler returned by the ODBCOpen function (integer) and *Offset* is the number of rows to move forward or backward. Positive values move forward, toward the end of the record set. Negative values move backward, toward the beginning (integer)

Returns:

On success, this function returns 0.
On fail, this function returns:
1 – Invalid handler.
2 – Database not open.
3 – Move error.

▪ **ODBCMoveFirst(Handler)**

Description: Moves to the first record within the record set. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.
On fail, this function returns:
1 – Invalid handler.
2 – Database not open.
3 – Move error.

▪ **ODBCMoveLast(Handler)**

Description: Moves to the last record within the record set. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.
On fail, this function returns:
1 – Invalid handler.
2 – Database not open.
3 – Move error.

- **ODBCMoveNext(Handler)**

Description: Moves to the next record within the record set. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – End of record set reached.
- 4 – Move error.

- **ODBCMovePrev(Handler)**

Description: Moves to the next record within the record set. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – Begin of record set reached.
- 4 – Move error.

- **ODBCCanAppend(Handler)**

Description: Returns whether the database allows you to add new records. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

Non-zero if the database allows adding new records; otherwise 0.

- **ODBCCanTransact(Handler)**

Description: Returns whether the database allows transactions. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

Non-zero if the database allows transactions; otherwise 0.

- **ODBCCanUpdate(Handler)**

Description: Returns whether the database can be updated. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

Non-zero if the database can be updated; otherwise 0.

- **ODBCIsBOF(Handler)**

Description: Returns whether you have gone before the first record of the record set. Call this function before you scroll from record to record. You can also use ODBCIsBOF function along with ODBCIsEOF to determine whether the record set contains any records or is empty. Immediately after you call ODBCQuery, if the record set contains no records, ODBCIsBOF returns nonzero. When you open a record set that has at least one record, the first record is the current record and ODBCIsBOF returns 0. If the first record is the current record and you call ODBCMovePrev, ODBCIsBOF will subsequently return nonzero.

Handler is the handler returned by the ODBCOpen function (integer).

Returns:

Non-zero if the record set contains no records or if you have scrolled backward before the first record; otherwise 0.

- **ODBCIsEOF(Handler)**

Description: Returns whether you have gone beyond the last record of the record set. Call this function as you scroll from record to record. You can also use ODBCIsEOF to determine whether the record set contains any records or is empty. Immediately after you call ODBCQuery, if the record set contains no records, ODBCIsEOF returns non-zero. When you open a record set that has at least one record, the first record is the current record and ODBCIsEOF returns 0. If the last record is the current record when you call ODBCMoveNext, ODBCIsEOF will subsequently return nonzero.

Handler is the handler returned by the ODBCOpen function (integer).

Returns:

Non-zero if the record set contains no records or if you have scrolled beyond the last record; otherwise 0.

- **ODBCIsDeleted(Handler)**

Description: Returns whether the current record has been deleted. If you scroll to a record and ODBCIsDeleted returns nonzero, then you must scroll to another record before you can perform any other operations. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

Nonzero if the record set is positioned on a deleted record; otherwise 0.

- **ODBCBeginTrans(Handler)**

Description: Begins a transaction with the connected data source. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
 - Error beginning transaction.

- **ODBCCommitTrans(Handler)**

Description: Commits a transaction. Call this function upon completing transactions. *Handler* is a handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – Error committing transaction.

- **ODBCRollback(Handler)**

Description: Reverses the changes made during a transaction. *Handler* is the handler returned by the ODBCOpen function (integer).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – Error rolling back transaction.

- **ODBCExecuteSQL(Handler, SQLCommand)**

Description: Executes an SQL command directly. ODBCExecuteSQL does not return data records. *Handler* is the handler returned by the ODBCOpen function (integer) and *SQLCommand* specifies a valid SQL command (string).

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – Invalid parameter.
- 4 – Error executing SQL command.

- **ODBCIsFieldNull(Handler, ColName)**

Description: Returns whether the specified field of a record set has been flagged as Null. *Handler* is the handler returned by the ODBCOpen function (integer) and *ColName* specifies a column name (string).

Returns:

Nonzero if the specified field is flagged as Null; otherwise 0.

- **ODBCIsFieldNullable(Handler, ColName)**

Description: Returns whether the specified field is "null able" (can be set to a Null value). *Handler* is the handler returned by the ODBCOpen function (integer) and *ColName* specifies a column name (string).

Returns:

Nonzero if the specified field is flagged as Null; otherwise 0.

- **ODBCSetFieldNull(Handler, ColName, Value)**

Description: Flags a field data member of the record set as Null (specifically having no value) or as non-Null. *Handler* is the handler returned by the ODBCOpen function (integer), *ColName* specifies a column name (string), and *Value* specifies a nonzero if the field data member is to be flagged as having no value (Null). Specify 0 for *Value* if the field data member is to be flagged as non-Null.

Returns:

On success, this function returns 0.

On fail, this function returns:

- 1 – Invalid handler.
- 2 – Database not open.
- 3 – Invalid parameter.
- 4 – Invalid column name.

II.8.20. Mail Functions

- **CnfEmail (strSmtplib, strFrom, strPOP3, strUser, strPassword, numOptionalTimeOut)**

Description: Sets SMTP (Simple Mail Transfer Protocol) parameters. This function must be executed to configure these parameters before sending emails with the SendEmail() function.

Parameters:

strSmtplib - String or string tag of the SMTP server name or IP address. For CEView application, you can only specify the SMTP IP address.

strFrom - String or string tag with the sender address.

strPOP3 - POP3 name of the sender.

strUser - User account name of the sender.

strPassword - Password for the user account name.

numOptionalTimeOut - Timeout limit (in seconds) used when sending messages. This parameter is optional. When it's not configured, the default timeout from operating system is used (this is recommended).

Return Values:

0 - Success

1 - Invalid format for parameter 1 (*strSMTP*)

2 - Invalid format for parameter 2 (*strFrom*)

3 - Invalid format for parameter 3 (*strPOP3*)

4 - Invalid format for parameter 4 (*strUser*)

5 - Invalid format for parameter 5 (*strPassword*)

6 - Invalid format for parameter 6 (*numOptionalTimeOut*)

7 - Wrong amount of parameters

8 - Error getting host IP address (invalid POP3 server)

9 - Error Connecting POP3 server

10- Error sending username

11- Error sending password

Examples:

```
CNFEMail("smtp.test.com.br", "factoryaddress@machine.com.br", "pop3.mail.com", "MyUserName", "MyPassword")
```

```
CNFEMail(TagString1, TagString2, TagString3, TagString4, TagString5)
```

- **SendEmail(strSubject, strMessage, strTo)**

Description: Sends e-mail messages. Before executing this function, it's necessary to set some parameters with the CnfEmail() function.

Parameters:

strSubject - String or string tag with the e-mail subject.

strMessage - String or string tag with the e-mail message (up to 255 characters).

strTo - String or string tag with recipient address (target).

Return Values:

0 - Success

1 - Invalid format for parameter 1 (strSubject)

2 - Invalid format for parameter 1 (strMessage)

3 - Invalid format for parameter 3 (strTo)

4 - Wrong amount of parameters

5 - Start Socket error

6 - Error getting host IP Address (invalid SMTP server)

7 - Error Connecting SMTP server

- 8 - Error sending HELO command (initialization)
- 9 - Error sending MAIL command (sending FROM address)
- 10- Error sending RCPT command (sending TO address)
- 11- Error sending DATA (sending message)

Examples:

```
SendEmail("Factory 1","Error to start process",MyAddress@HostName.com")
SendEmail(TagSubject,TagMessage,TagMyAddress)
```

- **SendEmailExt(strSubject, strMessage, strTO, strCC, strBCC, strFile1, ..., strFileN)**

Description: Sends e-mail messages with attached files. Before executing this function, it's necessary to set some parameters with the CnfEmail() function.

Parameters:


strSubject - String or string tag with the e-mail subject (up to 255 characters).
strMessage - String or string tag with the e-mail message (up to 255 characters).
strTO - String or string tag with recipient address TO.
strCC - String or string tag with recipient address CC.
strBCC - String or string tag with recipient address BCC.
strFile1 - Path and name of the first file to the sent attached in the email message.
strFileN - Path and name of the Nth file to the sent attached in the email message.

Return Values:

- 0 - Success
- 1 - Cannot execute the function because the last email was not sent yet
- 2 - Internal error
- 1 - The library INDMail.DLL is corrupted
- 2 - The version of the library INDMail.DLL is incorrect
- 3 - Wrong amount of parameters (at least 3 parameters are required)
- 4 - Some of the attached files were not found

Examples:

```
SendEmailExt("Emergency","Motor is out of order","maint@mail.com")
SendEmailExt(TagSub,TagMsg,TagToAddr)
SendEmailExt(TagSub,TagMsg,"e1@mail.com;g@t.de","", "cust@mail.com")
SendEmailExt(TagSub,TagMsg,"emp1@mail.com;emp2@mail.com","", "c:\Rep1.txt", "c:\Rep2.txt")
```

 **Notes:**

This function is asynchronous. An internal thread is created to execute this function whenever it's called. Although the function is asynchronous, it cannot be executed while the last email has not been completely sent. Otherwise, the function will return the error code 1 and the email will not be sent.

It's mandatory to configure the parameters *strSubject*, *strMessage*, and *strTO*. The other parameters are optional.

It can be used the null string value ("") to the parameters *strTO* , *strCC*, or *strBCC* when some of them will not be used.

It's possible to assign more than one recipient in the *strTO*, *strCC*, or *strBCC* parameters, using the semicolon (;) char to share the addresses.

▪ **GetStatusSendEMailExt(numUpdate)**

Description: Returns the status of the last email sent using the function SendEmailExt(). *numUpdate* specifies when the tag configured in this parameter change of value, the function updates its return value. This parameter is optional but it must be used when this function is configured in any screen dynamic (Text I/O, Position, etc).

Return Values:

- 0 - Function SendEmailExt() is not being executed.
- 1 - The last email is still being sent. The function SendEmailExt() cannot be executed.
- 2 - The last email has been sent successfully. The function SendEmailExt() can be executed again.
- 3 - There was an error when sending the last email. The function SendEmailExt() can be executed again.
- 1 - The library INDMail.DLL is corrupted.
- 2 - The version of the library INDMail.DLL is incorrect.

Examples:

```
TagStatus GetStatusSendEMailExt(second)
TagStatus GetStatusSendEMailExt()
```

II.8.21. Dial-up connections

▪ **DialError(numType , strPhonebookEntryOrModem , strStatus , tagRefresh)**

Description: Returns the Error Codes regarding each connection;

Parameters:

numType: (0: using a PhoneBook name as reference; 1: using a Modem name as reference);

strPhonebookEntryOrModem: This parameter depends of the "numType" parameter;

If numType = 0, then strPhonebookEntryOrModem = Phonebook Name

If numType = 1, then strPhonebookEntryOrModem = Modem Name

strStatus: Returns the description of the error. This parameter must be filled with a tag between double-quotation (e.g.: "ErrorDescTag"). This tag will receive the description of the error. This parameter is optional;

tagRefresh: When this function is configured in a "Text I/O" dynamic, it's necessary to configure a tag in this parameter. Whenever this tag change of value, the function updates the error values. This parameter is optional.

Return Values:

- 0 - Ok;
- 1 - Error: INDRas.DLL not found;
- 2 - Error: INDRas.DLL damaged;
- 3 - Error: invalid amount of parameters (minimum=2);
- 4 - Invalid value for the "numType" parameter (0 or 1);
- 5 - PhoneBook or Modem does not exist;
- 600 - An operation is pending.
- 601 - The port handle is invalid.
- 602 - The port is already open.
- 603 - Caller's buffer is too small.

604 - Wrong information specified.
605 - Cannot set port information.
606 - The port is not connected
607 - The event is invalid.
608 - The device does not exist.
609 - The device type does not exist.
610 - The buffer is invalid.
611 - The route is not available.
612 - The route is not allocated.
613 - Invalid compression specified.
614 - Out of buffers.
615 - The port was not found.
616 - An asynchronous request is pending.
617 - The port or device is already disconnecting.
618 - The port is not open.
619 - The port is disconnected.
620 - There are no endpoints.
621 - Cannot open the phone book file.
622 - Cannot load the phone book file.
623 - Cannot find the phone book entry.
624 - Cannot write the phone book file.
625 - Invalid information found in the phone book file.
626 - Cannot load a string.
627 - Cannot find key.
628 - The port was disconnected.
629 - The data link was terminated by the remote machine.
630 - The port was disconnected due to hardware failure.
631 - The port was disconnected by the user.
632 - The structure size is incorrect.
633 - The port is already in use or is not configured for Remote Access dial out.
634 - Cannot register your computer on on the remote network.
635 - Unknown error.
636 - The wrong device is attached to the port.
637 - The string could not be converted.
638 - The request has timed out.
639 - No asynchronous net available.
640 - A NetBIOS error has occurred.
641 - The server cannot allocate NetBIOS resources needed to support the client.
642 - One of your NetBIOS names is already registered on the remote network.
643 - A network adapter at the server failed.
644 - You will not receive network message popups.
645 - Internal authentication error.
646 - The account is not permitted to logon at this time of day.
647 - The account is disabled.
648 - The password has expired.
649 - The account does not have Remote Access permission.
650 - The Remote Access server is not responding.
651 - Your modem (or other connecting device) has reported an error.
652 - Unrecognized response from the device.
653 - A macro required by the device was not found in the device .INF file section.
654 - A command or response in the device .INF file section refers to an undefined acro.
655 - The <message macro was not found in the device .INF file section.
656 - The <defaultoff macro in the device .INF file section contains an undefined macro.
657 - The device .INF file could not be opened.
658 - The device name in the device .INF or media .INI file is too long.
659 - The media .INI file refers to an unknown device name.
660 - The device .INF file contains no responses for the command.
661 - The device .INF file is missing a command.

662 - Attempted to set a macro not listed in device .INF file section.
663 - The media .INI file refers to an unknown device type.
664 - Cannot allocate memory.
665 - The port is not configured for Remote Access.
666 - Your modem (or other connecting device) is not functioning.
667 - Cannot read the media .INI file.
668 - The connection dropped.
669 - The usage parameter in the media .INI file is invalid.
670 - Cannot read the section name from the media .INI file.
671 - Cannot read the device type from the media .INI file.
672 - Cannot read the device name from the media .INI file.
673 - Cannot read the usage from the media .INI file.
674 - Cannot read the maximum connection BPS rate from the media .INI file.
675 - Cannot read the maximum carrier BPS rate from the media .INI file.
676 - The line is busy.
677 - A person answered instead of a modem.
678 - There is no answer.
679 - Cannot detect carrier.
680 - There is no dial tone.
681 - General error reported by device.
682 - ERROR_WRITING_SECTIONNAME
683 - ERROR_WRITING_DEVICETYPE
684 - ERROR_WRITING_DEVICENAME
685 - ERROR_WRITING_MAXCONNECTBPS
686 - ERROR_WRITING_MAXCARRIERBPS
687 - ERROR_WRITING_USAGE
688 - ERROR_WRITING_DEFAULTOFF
689 - ERROR_READING_DEFAULTOFF
690 - ERROR_EMPTY_INI_FILE
691 - Access denied because username and/or password is invalid on the domain.
692 - Hardware failure in port or attached device.
693 - ERROR_NOT_BINARY_MACRO
694 - ERROR_DCB_NOT_FOUND
695 - ERROR_STATE_MACHINES_NOT_STARTED
696 - ERROR_STATE_MACHINES_ALREADY_STARTED
697 - ERROR_PARTIAL_RESPONSE_LOOPING
698 - A response keyname in the device .INF file is not in the expected format.
699 - The device response caused buffer overflow.
700 - The expanded command in the device .INF file is too long.
701 - The device moved to a BPS rate not supported by the COM driver.
702 - Device response received when none expected.
703 - The Application does not allow user interactionThe connection requires
nteraction with the user to complete successfully
704 - ERROR_BAD_CALLBACK_NUMBER
705 - ERROR_INVALID_AUTH_STATE
706 - ERROR_WRITING_INITBPS
707 - X.25 diagnostic indication.
708 - The account has expired.
709 - Error changing password on domain The password may be too short or may
match a previously used password.
710 - Serial overrun errors were detected while communicating with your modem.
711 - RasMan initialization failure Check the event log.
712 - Bipler port initializing Wait a few seconds and redial.
713 - No active ISDN lines are available.
714 - No ISDN channels are available to make the call.
715 - Too many errors occurred because of poor phone line quality.
716 - The Remote Access IP configuration is unusable.
717 - No IP addresses are available in the static pool of Remote Access IP addresses.
718 - Timed out waiting for a valid response from the remote PPP peer.
719 - PPP terminated by remote machine.

720 - No PPP control protocols configured.
 721 - Remote PPP peer is not responding.
 722 - The PPP packet is invalid.
 723 - The phone number including prefix and suffix is too long.
 724 - The IPX protocol cannot dial-out on the port because the machine is an IPX router.
 725 - The IPX protocol cannot dial-in on the port because the IPX router is not installed.
 726 - The IPX protocol cannot be used for dial-out on more than one port at a time.
 727 - Cannot access TCPCFG.DLL.
 728 - Cannot find an IP adapter bound to Remote Access.
 729 - SLIP cannot be used unless the IP protocol is installed.
 730 - Computer registration is not complete.
 731 - The protocol is not configured.
 732 - The PPP negotiation is not converging.
 733 - The PPP control protocol for this network protocol is not available on the server.
 734 - The PPP link control protocol terminated.
 735 - The requested address was rejected by the server.
 736 - The remote computer terminated the control protocol.
 737 - Loopback detected.
 738 - The server did not assign an address.
 739 - The authentication protocol required by the remote server cannot use the Windows NT encrypted password Redial, entering the password explicitly.
 740 - Invalid TAPI configuration.
 741 - The local computer does not support the required encryption type.
 742 - The remote computer does not support the required encryption type.
 743 - The remote computer requires encryption.
 744 - Cannot use the IPX network number assigned by remote server Check the event log.
 745 - ERROR_INVALID_SMM
 746 - ERROR_SMM_UNINITIALIZED
 747 - ERROR_NO_MAC_FOR_PORT
 748 - ERROR_SMM_TIMEOUT
 749 - ERROR_BAD_PHONE_NUMBER
 750 - ERROR_WRONG_MODULE
 751 - Invalid callback number Only the characters 0 to 9, T, P, W, (,), -, @, and space are allowed in the number.
 752 - A syntax error was encountered while processing a script.
 753 - The connection could not be disconnected because it was created by the Multi-Protocol Router.

- **DialUp**(numType , strPhonebookEntryOrModem , strUserName , strPassword , strDomain , strPhoneNumber)

Description: Trigger a dial-up connection;

Parameters:

numType (0: using a PhoneBook name as reference; 1: using a Modem name as reference)

strPhonebookEntryOrModem: This parameter depends of the "numType" parameter;

If numType = 0, then strPhonebookEntryOrModem = Phonebook Name

If numType = 1, then strPhonebookEntryOrModem = Modem Name

strUserName: UserName registered in the Server;


strPassword: Password for the user registered in the Server;

strDomain: Domain Name in the server (It's an optional parameter and can be filled with null string "");

strPhoneNumber: Telephone number (used only when the parameter numType=1)

Return Values:

- 0 - OK: dialing started;
- 1 - Error: INDRas.DLL not found;
- 2 - Error: INDRas.DLL damaged;
- 3 - Error: invalid amount of parameters (minimum=5);
- 4 - Invalid value for the "numType" parameter (0 or 1);
- 5 - Invalid value for the parameter "strPhonebookEntryOrModem" (string);
- 6 - PhoneBook or Modem does not exist;
- 7 - PhoneBook or Modem is in use;
- 8 - Depends of the "numType" parameter:
 - if "numType" = 0: Could not read properties from PhoneBook;
 - if "numType" = 1: More than 1000 connections are enabled at same time;
- 9 - Unable to create a temporary PhoneBook.

 **Note:** The Dial-In for WinNT/2K stations is executed automatically by the operating system RAS Server.

- **DialUpToCE**(numModem , numDialPhone , numMyNumber , strUser , strPassword , strDomain , boolAutoDial , boolAutoClose)

Description: Executes the program DialUpToCE, which sends the information necessary to CERasSvr.exe calls back to the WinNT/2K station.

Parameters:


- numModem:** Number of the modem used to dial to the WinCE station;
- numDialPhone:** Telephone number of the WinCE remote station;
- numMyNumber:** Telephone number sent to WinCE remote station. CERasSvr.exe will call back to this phone number;
- strUser:** User name sent to WinCE remote station. CERasSvr.exe will use this user name to connect to the WinNT/2K station after calling back to it;
- strPassword:** Password sent to WinCE remote station. CERasSvr.exe will use this password to connect to the WinNT/2K station after calling back to it;
- strDomain:** Domain name – optional parameter;
- boolAutoDial:** Trigger the DialUp connection automatically when the function is executed – optional parameter;
- boolAutoClose:** Close the DialUpToCE dialog window automatically after dialing to the WinCE remote station – optional parameter.

Return Values:

- 0 - Fail (unable to call DialUpToCE)
- 1 - Success (executed DialUpToCE)

Examples:

DialUpToCE(0,"12344321","98765432","Administrator","MyPass")
DialUpToCE(0,"12344321","98765432","Administrator","MyPass","",1,1)

 **Note:** The program DialUpToCE was developed to dial to a WinCE remote station. Once Windows CE v3.00 does not provide a RAS Server, the program CERasSvr.exe must be running under the WinCE device to answer the call and call back to the WinNT/2K station using the parameters sent by the DialUpToCE() function. The RAS Server service must be configured in the WinNT/2K station to answer the call back from the WinCE device and set the TCP/IP connection.

- **DialStatus**(numType , strPhonebookEntryOrModem , "tagStatus" , tagRefresh)

Description: Returns the Status of each connection;

Parameters:

numType: (0: using a PhoneBook name as reference; 1: using a Modem name as reference);

strPhonebookEntryOrModem: This parameter depends of the "numType" parameter;

If numType = 0, then strPhonebookEntryOrModem = Phonebook Name

If numType = 1, then strPhonebookEntryOrModem = Modem Name

tagStatus: Returns the description of the status. This parameter must be filled with a tag between double-quotation (e.g.: "StatusDescTag"). This tag will receive the description of the error. This parameter is optional;

tagRefresh: When this function is configured in a "Text I/O" dynamic, it's necessary to configure a tag in this parameter. Whenever this tag change of value, the function updates the error values. This parameter is optional.

Return Values:

- 1 - Error: INDRas.DLL not found;
- 2 - Error: INDRas.DLL damaged;
- 3 - Error: invalid amount of parameters (minimum=2);
- 4 - Invalid value for the "numType" parameter (0 or 1);
- 5 - PhoneBook or Modem does not exist;
- 0 - Opening the port...
- 1 - Port has been opened successfully.
- 2 - Connecting to the device...
- 3 - The device has connected successfully.
- 4 - All devices in the device chain have successfully connected.
- 5 - Verifying the user name and password...
- 6 - An authentication event has occurred.
- 7 - Requested another validation attempt with a new user.
- 8 - Server has requested a callback number.
- 9 - The client has requested to change the password
- 10 - Registering your computer on the network...
- 11 - The link-speed calculation phase is starting...
- 12 - An authentication request is being acknowledged.
- 13 - Reauthentication (after callback) is starting.
- 14 - The client has successfully completed authentication.
- 15 - The line is about to disconnect for callback.
- 16 - Delaying to give the modem time to reset for callback.
- 17 - Waiting for an incoming call from server.
- 18 - Projection result information is available.
- 19 - User authentication is being initiated or retried.
- 20 - Client has been called back and is about to resume authentication.
- 21 - Logging on to the network...
- 22 - Subentry has been connected.
- 23 - Subentry has been disconnected
- 24 - Terminal state supported by RASPHONE.EXE.
- 25 - Retry authentication state supported by RASPHONE.EXE.
- 26 - Callback state supported by RASPHONE.EXE.
- 27 - Change password state supported by RASPHONE.EXE.
- 8192 - Connected to remote server successfully!
- 8193 - Disconnected.

- **FindModem("tagArray"):**

Description: Returns the list of all available modems in the local station;

Parameters:

tagArray: This field must be filled with the name of a string array tag between double quotation ("Modems"). Each index of the array will receive the name of each modem available in the station.

Return Value: Amount of available modems in the local station.

▪ **HangUp(numType , strPhonebookEntryOrModem)**

Description: hang-up a dial-up connection;

Parameters:

numType: (0: using a PhoneBook name as reference; 1: using a Modem name as reference);

strPhonebookEntryOrModem: This parameter depends of the "numType" parameter;

 If numType = 0, then strPhonebookEntryOrModem = Phonebook Name

 If numType = 1, then strPhonebookEntryOrModem = Modem Name

Return Values:

0 - Ok.

-1 - Error: INDRas.DLL not found;

-2 - Error: INDRas.DLL damaged;

-3 - Invalid value for the "numType" parameter (0 or 1);

-4 - PhoneBook or Modem does not exist;

-5 - Does not exist any modem configured;

1 - General Error