DLite 1.5 API Manual

For WinCon-8000

Contents

CHAPTER 1 DLITE API FOR WINCON-8000 INTRODUCTION 3

1-1 REAL-TIME MESSAGING USING DLITE API 3

1-2 XDLITE OCX METHOD AND EVENTS 4

Property: 4

Method: 4

Event: 5

1-3 USING ACTIVEX COMPONENT IN EVC++ AND WINCON-8000 12

CHAPTER 2 WINCON-8000 DLITE API SAMPLES 15

2-1 CREATE DEMO PROJECT 15

2-2 DEPLOY CONTROL INTO THE PROJECT 18

2-3 EDIT CODE 19

On Start 19

On Timer 20

On Disconnect 22

Complier options 23

ABOUT ESOFTSYSTEM CORP. 24

Chapter 1 DLite API for WinCon-8000 introduction

WinCon-8000 is a programmable controller based on Windows CE system. DLite provided a Windows CE version of xDLite.ocx that could be used on WinCon-8000. WinCon-8000 could use this OCX to send real-time data and message to EKAN LED display.

1-1 Real-time messaging using DLite API

DLite 1.5 API for WinCon-8000 is based on Window CE and ActiveX technology. DLite provides an OCX control to access real-time function of DLite LED display.

User could use OCX to design more flexible and powerful application with DLite API.

We illustrate an eVC++ sample here to show you how to use these API functions.

1-2 xDLite OCX Method and Events

xDLite OCX provides events callback for developer to control and detect EKAN LED display, these are:

Property:

□ **EkanIP:** Connection IP address

□ **EkanPort:** Ekan connection port is 7000

□ **Password:** The password need to connect to EKAN

Method:

□ **Connect:** User provides IP address to connect remote LED display.

□ **Disconnect:** for disconnect the connection with LED display.

□ **SendMessage:** Used for send text message for real-time display.

□ **SendMessageS:** Send small font message

□ **DisplayBMP:** Trigger BMP on LED display

□ **DisplatDate:** Display Date on LED display

□ **DisplayTime:** Display Time on LED system

□ **FillRect:** Fill rectangular area for specific color

□ **ClearRect:** Clear rectangular area

□ **DrawLine:** Draw line on display

□ **Pixel:** Draw pixel on LED

□ **Upload:** Upload bmp or script files to LED.

□ **Download:** Download files from LED display.

□ **Reset:** Clear all data on LED.

□ **Clear:** Clear all text in the display screen.

Event:

□ **Dissconnected:** triggered after disconnect method.

Following table listed API method arguments:

Property & Method	Argument	Syntax
EkanIP	String like “192.168.0.100”	xDlite1.EkanIP=”192.168.0.100”
EkanPort	Number like 7000	xDlite1.Port=7000
EkanPassword	String like “Admin”	xDLite1.Password=”admin”
Connect	IP address as string	xDLite1.Connect
Disconnect	None	XDLite1.Disconnect
SendMessage	Need 5 arguments to send the message. PosX and PosY are the coordinate of the Text message, start from LED display upper left corner. Both are Integer type variable. fgClr and bgClr are the foreground color and background color of the displayed text. Both are Integer type variable. Color code index is: **0: Black** **1: Green** **2: Red** **3: Orange** szMsg is the text string needed to be displayed.	xDLite1.SendMessage 0,0,1,0,”Hello World!” Return > 0 success
SendMessageS	Need 5 arguments to send the small font message. PosX and PosY are the coordinate of the Text message, start from LED display upper	xDLite1.SendMessageS 0,0,1,0,”Hello World!” Return > 0 success

	left corner. Both are Integer type variable. fgClr and bgClr are the foreground color and background color of the displayed text. Both are Integer type variable. Color code index is: **0: Black** **1: Green** **2: Red** **3: Orange** szMsg is the text string needed to be displayed.	
DisplayBMP	Need 5 arguments to send the small font message. PosX and PosY are the coordinate of the Text message, start from LED display upper left corner. Both are Integer type variable. fgClr and bgClr are the foreground color and background color of the displayed text. Both are Integer type variable. Color code index is: **0: Black** **1: Green** **2: Red** **3: Orange** szMsg is the text string needed	xDLite1.DisplayBMP 0,0,1,0,”1.BMP” Return > 0 success

	to be displayed.	
DisplayDate	Need 4 arguments to send the small font message. PosX and PosY are the coordinate of the Text message, start from LED display upper left corner. Both are Integer type variable. fgClr and bgClr are the foreground color and background color of the displayed text. Both are Integer type variable. Color code index is: **0: Black** **1: Green** **2: Red** **3: Orange**	xDLite1.DisplayDate 0,0,1,0
DisplayTime	Need 4 arguments to send the small font message. PosX and PosY are the coordinate of the Text message, start from LED display upper left corner. Both are Integer type variable. fgClr and bgClr are the foreground color and background color of the displayed text. Both are Integer type variable. Color code index is: **0: Black**	xDLite1.DisplayTime 0,0,1,0

	1: Green **2: Red** **3: Orange**	
DrawLine	Needed 5 arguments to send the small font message. PosX1 and PosY1 and PosX2 and PosY2, start from LED display upper left corner. Both are Integer type variable. Color **0: Black** **1: Green** **2: Red** **3: Orange**	xDLite1.Line 0,0,10,10, 3
Pixel	Needed 3 arguments to send the small font message. PosX1 and PosY1 , start from LED display upper left corner. Both are Integer type variable. Color **0: Black** **1: Green** **2: Red** **3: Orange**	xDLite1.Pixel 0,0, 3
FillRect	Fill rectangular area with specify color Need 5 arguments to send the small font message. PosX1 and PosY1 and PosX2 and PosY2, start from LED	xDLite1.FillRect 0,0,10,10,3

	display upper left corner. Both are Integer type variable. Color **0: Black** **1: Green** **2: Red** **3: Orange**	
ClearRect	Clear specify rectangular area Need 4 arguments to send the small font message. PosX1 and PosY1 and PosX2 and PosY2, start from LED display upper left corner	xDLite1.ClearRect 0,0,10,10
Upload	Full upload string with file name	xDlite1.uplaod (“c:\ekan.ils”)
Download	Full download path for files download. All the file on DLite LED will be downloaded.	xDLite1.download (“c:\test”)
Reset	Delete all files on EKAN memory	xDLite1.Reset
Clear	Used for clear the DLite message on LED display.	xDLite1.Clear

Table 1-1 OCX supported method

Following table listed these events that API method might trigger:

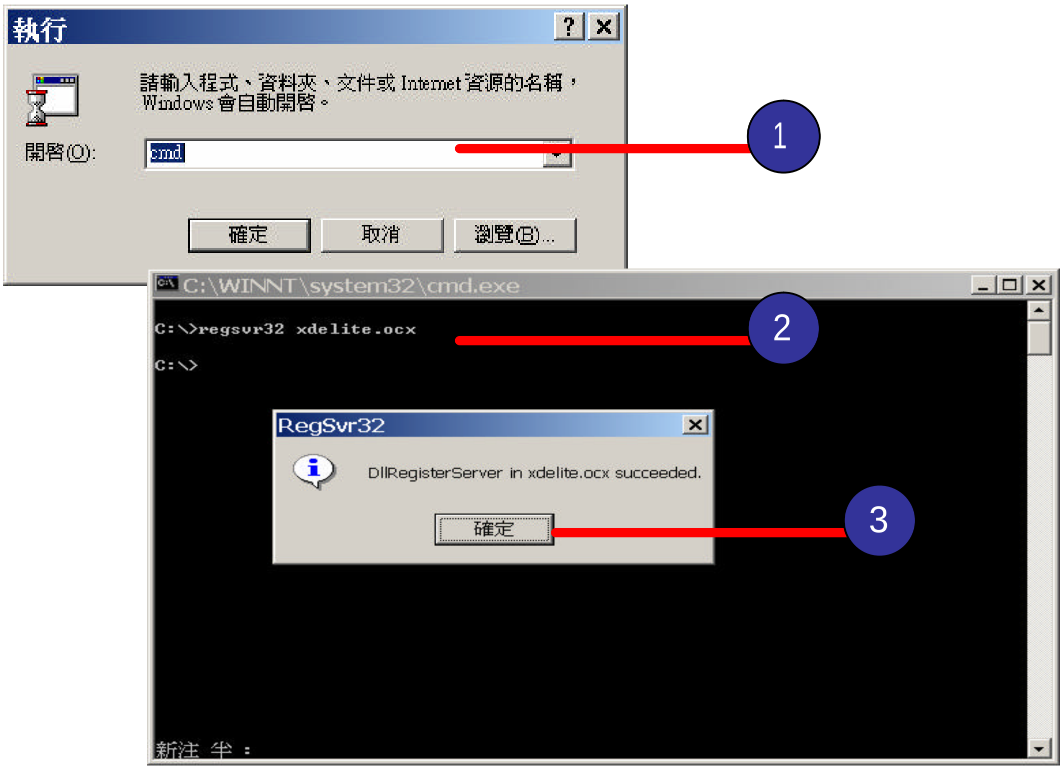
Event	When it Happened?	Pass in parameter
Disconnected	After broken the connection with LED device	None

Table 1-2 xDLite.ocx supported events

1-3 Using ActiveX component in eVC++ and WinCon-8000

Before you start using xDLite OCX on eMbedded Visual C++, you must register this OCX to your windows registry first. Registration on PC side and WinCon-8000 side are both required.

The PC side OCX version is same as desktop version of xDLite.ocx. Please copy the ocx to the program directory, and enter windows command mode, type: regsvr32 xDLite.ocx



執行

請輸入程式、資料夾、文件或 Internet 資源的名稱，Windows 會自動開啟。

開啟(O): cmd

```
C:\WINNT\system32\cmd.exe
C:\>regsvr32 xdelite.ocx

C:\>
```

RegSvr32

DllRegisterServer in xdelite.ocx succeeded.

Step 1: Enter the command mode, move to the directory that xDLite.ocx exist.

Step 2: type **regsvr32 xDLite.ocx**

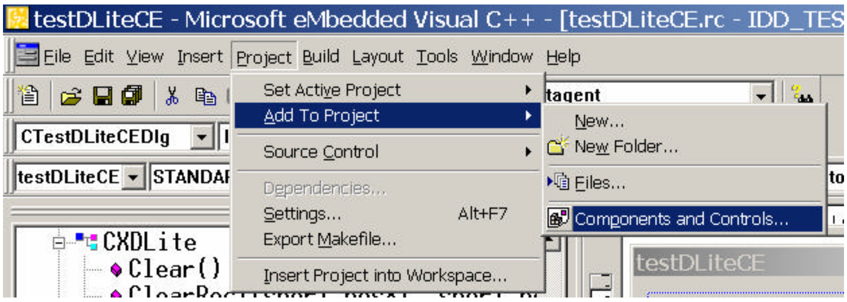
Step 3: Windows message dialog box shows succeed

In ***WinCon-8000 side***, you must register this ocx by **REGSVRCE** or **REGSVR32** program provided with WinCon-8000 to finish to registration process. The register process is Same, but the OCX version is ***different***.

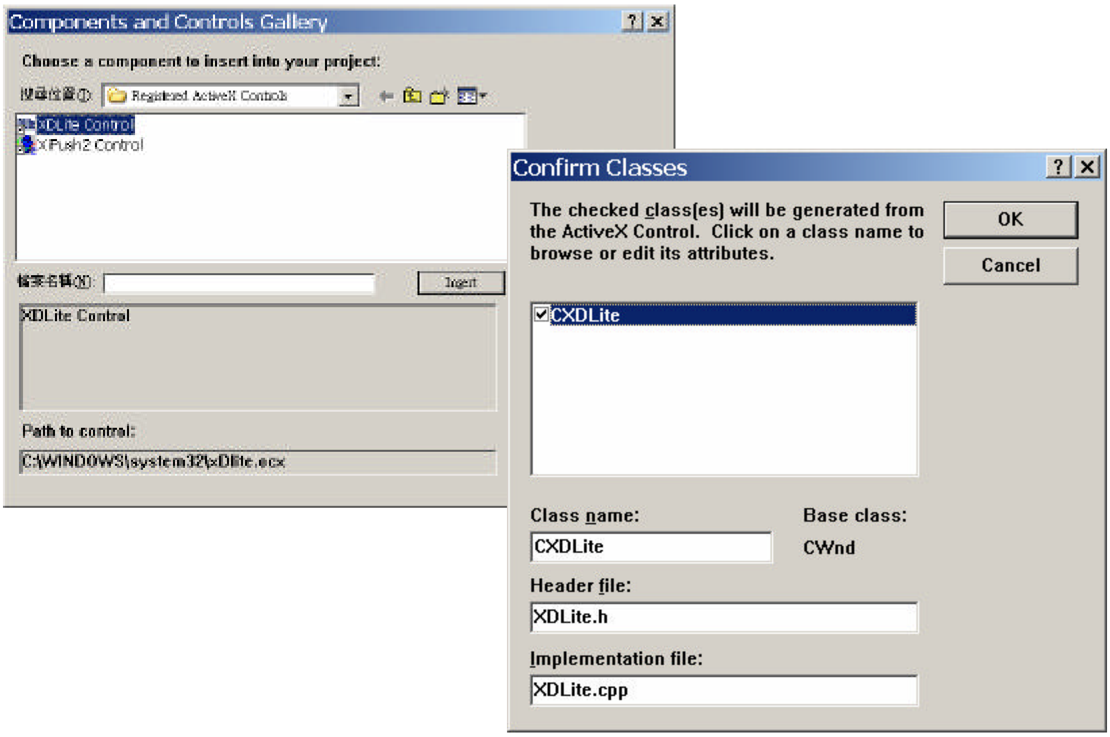
WinCon-8000 is using Windows CE version of xDLite.ocx**.**

After OCX resisted to Windows and Windows CE, you could start your development process. Please make sure your ocx version is up to date. Please visit eSoftsystem web site at www.esoftsystem.com for further information.

After you register the **PC side OCX** into system, you could refer the OCX into eVC++ project.


testDLiteCE - Microsoft eMbedded Visual C++ - [testDLiteCE.rc - IDD_TES
File Edit View Insert Project Build Layout Tools Window Help
Set Active Project
Add To Project
Source Control
Dependencies...
Settings... Alt+F7
Export Makefile...
Insert Project into Workspace...
New...
New Folder...
Files...
Components and Controls...
CTestDLiteCEDlg
testDLiteCE
STANDAR
CXDLite
Clear()
testDLiteCE

Then press **Insert** button to add xDLite.ocx into eVC++ project. If you could not find xDLite.ocx, please press **Add** to browse and add Control.


Components and Controls Gallery

Choose a component to insert into your project:

搜尋位置(I): Registered ActiveX Controls

XDLite Control
XPush2 Control

檔案名稱(N):

XDLite Control

Path to control:

C:\WINDOWS\system32\xDlite.ocx

Confirm Classes

The checked class(es) will be generated from the ActiveX Control. Click on a class name to browse or edit its attributes.

☑ CXDLite

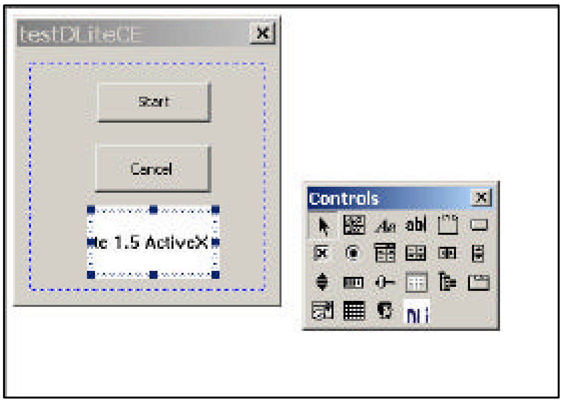
Class name: CXDLite

Base class: CWnd

Header file: XDLite.h

Implementation file: XDLite.cpp

After you add the control to the toolbar, you could drag and drop the xDLite.ocx into the project.



Chapter 2 WinCon-8000 DLite API Samples

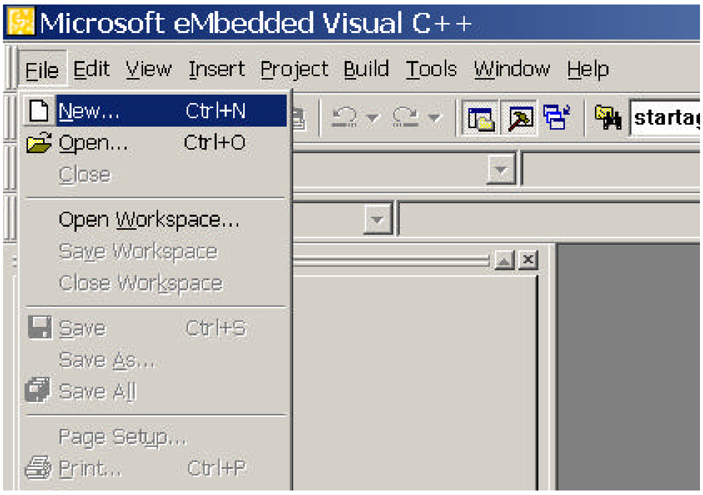
We provide a sample here, which using the eVC++ 4.0 as development environment.

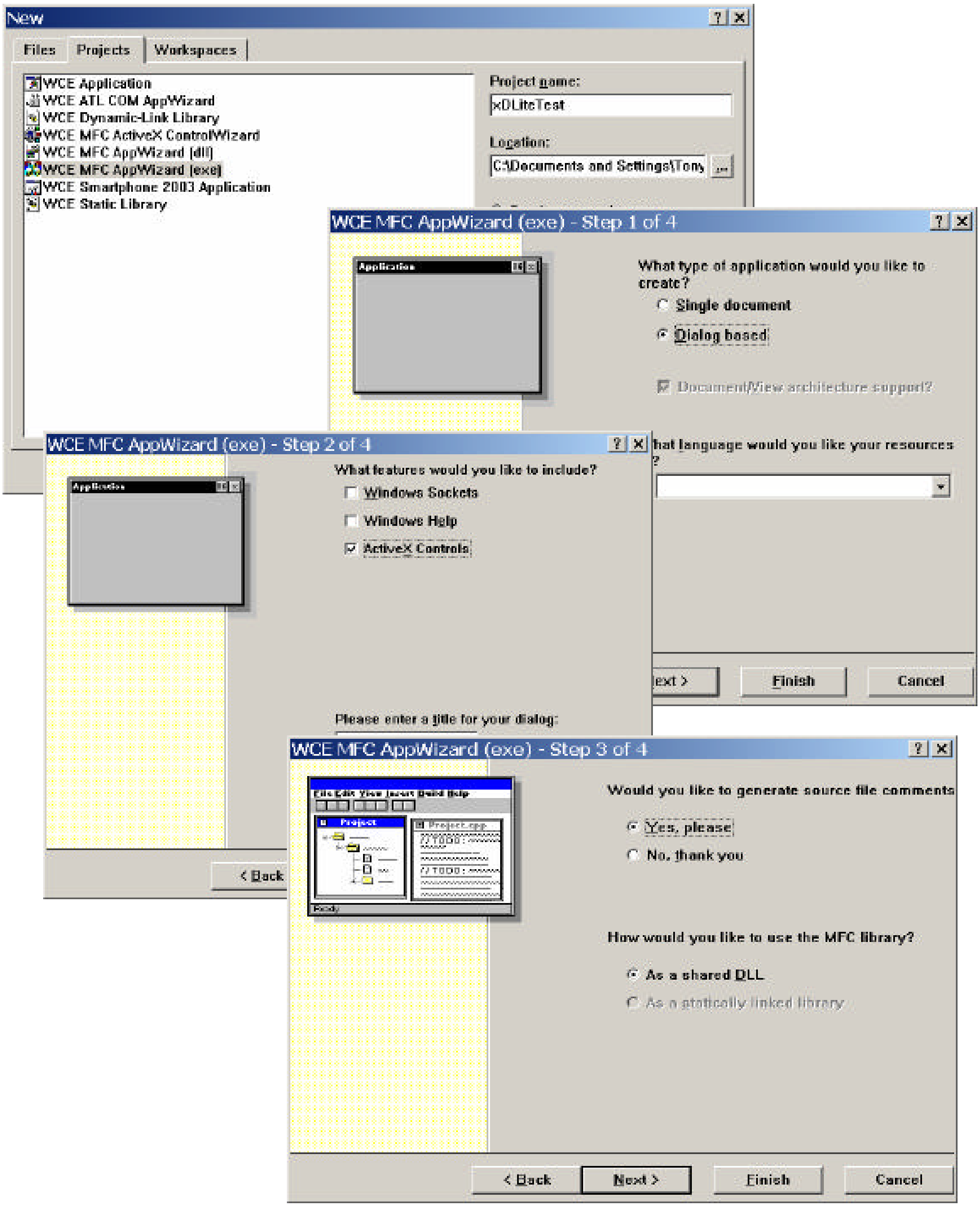
2-1 Create Demo Project

In the Embedded Visual C++ sample, we will demonstration the basic function of the OCX, and make a simple program to send message to EKAN LED display. First you must register PC side of xDLite.ocx, you could refer previous chapter for details.

Here we use a dialog based program with 2 buttons for start and stop Demo play. And the most important part is to refer xDLite.ocx into eVC++ project and generate the CXDLite wrapper class. Here are detail steps to create Demo project.

We use xDLiteTest as project name, to create the demo project. Select the **File / New** to create project.





New

Files | Projects | Workspaces

- WCE Application
- WCE ATL COM AppWizard
- WCE Dynamic-Link Library
- WCE MFC ActiveX ControlWizard
- WCE MFC AppWizard (dll)
- WCE MFC AppWizard (exe)
- WCE Smartphone 2003 Application
- WCE Static Library

Project name: xDLiteTest

Location: C:\Documents and Settings\Tony

WCE MFC AppWizard (exe) - Step 1 of 4

What type of application would you like to create?

○ Single document
● Dialog based

☑ Document/View architecture support?

hat language would you like your resources ?

WCE MFC AppWizard (exe) - Step 2 of 4

What features would you like to include?

□ Windows Sockets
□ Windows Help
☑ ActiveX Controls

Please enter a title for your dialog:

WCE MFC AppWizard (exe) - Step 3 of 4

Would you like to generate source file comments

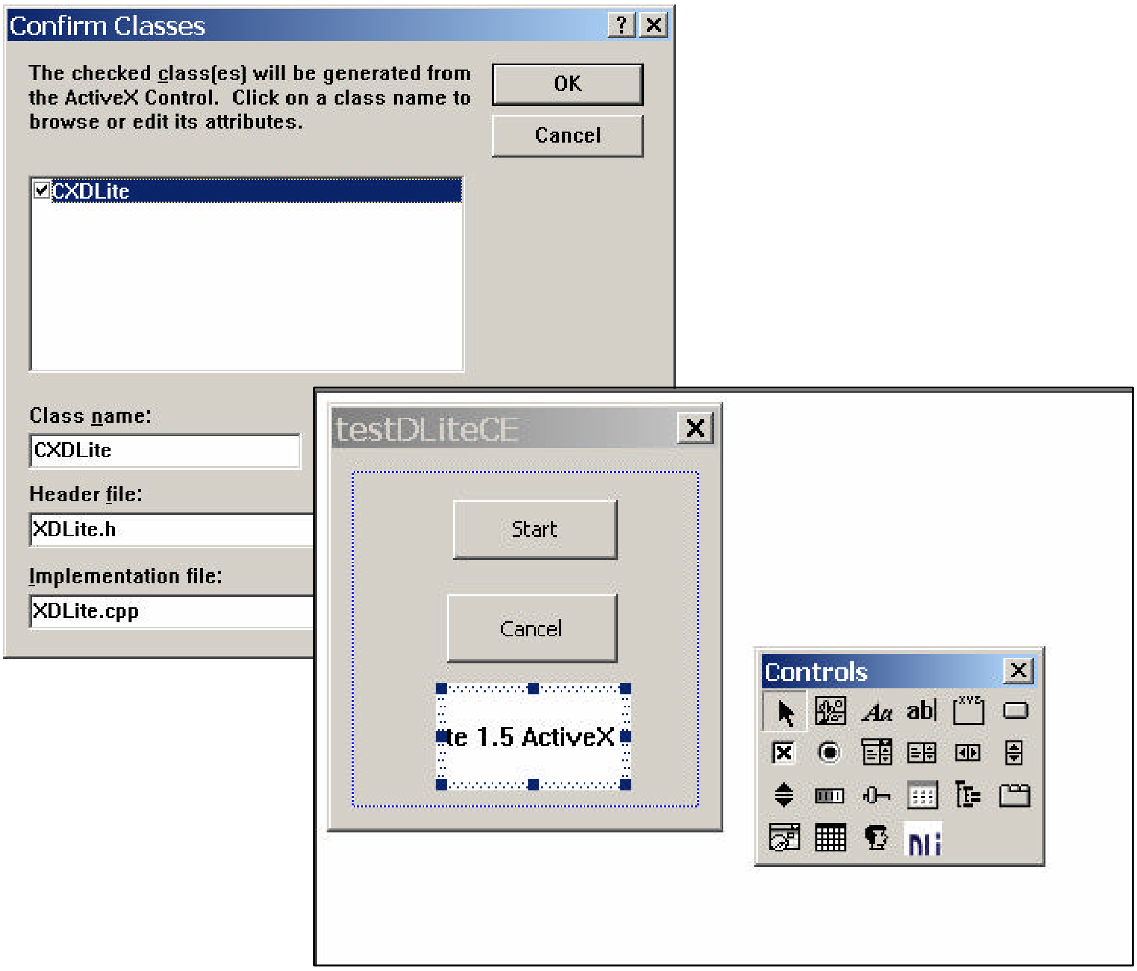
● Yes, please
○ No, thank you

How would you like to use the MFC library?

● As a shared DLL
○ As a statically linked library

After you create the project, you could add 2 buttons into dialog box, first one for the start function, second for the stop function. Then add the xDLite.ocx into the project, please refer to the previous chapter for details. eVC will automatically generate **CXDLite** class declaration for the xDLite.ocx.

When finished, you will see following interface in the project resource files.



Confirm Classes

The checked class(es) will be generated from the ActiveX Control. Click on a class name to browse or edit its attributes.

☑ CXDLite

Class name: CXDLite

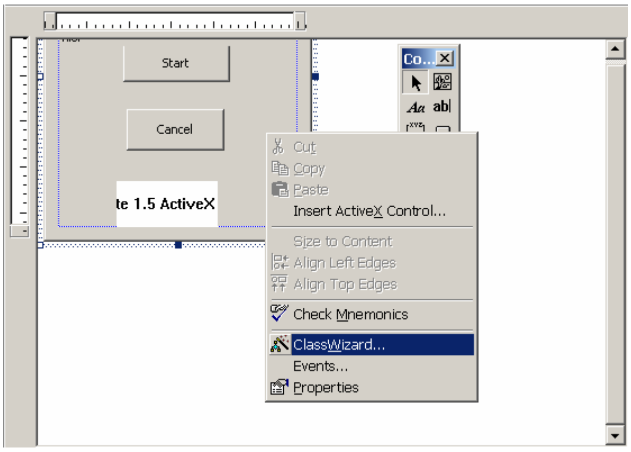
Header file: XDLite.h

Implementation file: XDLite.cpp

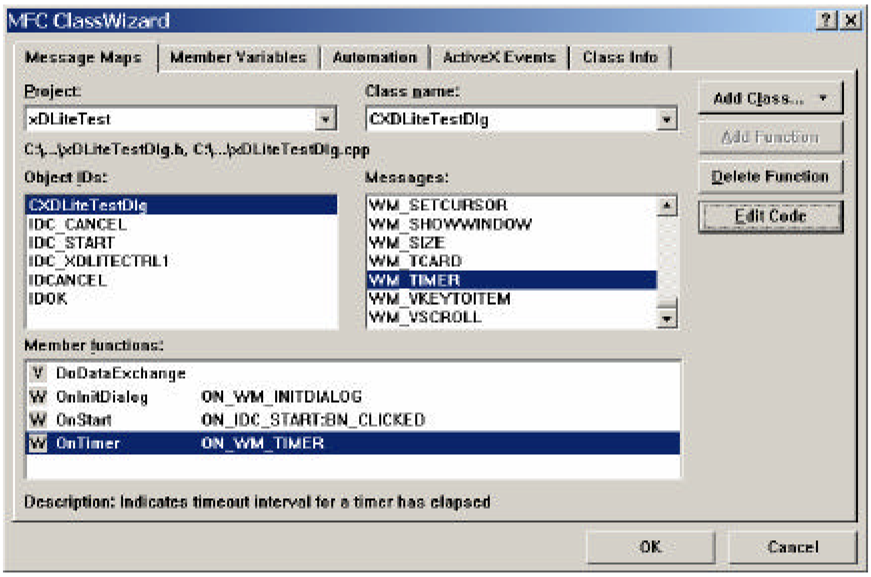
2-2 Deploy control into the project

Then you must add a member function of CXDLite into the CXDLiteTestDlg. In the demo, we use m_EKANAgent as CXDLite member variable. You could use class wizard to help you create the member variable into the xDLiteTestDlg.

Please click right mouse button on the dialog box the add member function and Timer event into the dialog box.



Add timer event into the dialog box. Later We will use the Timer event to automatically send the message.



Project: xDLiteTest

Class name: CXDLiteTestDlg

C:\...\xDLiteTestDlg.h, C:\...\xDLiteTestDlg.cpp

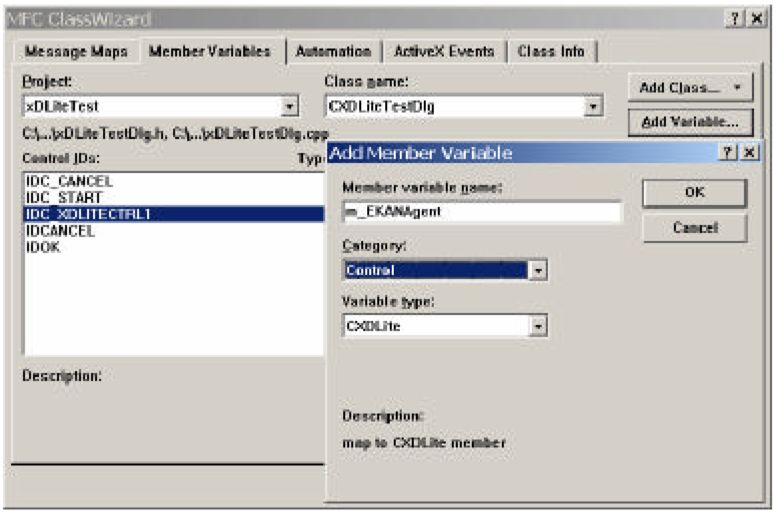
Object IDs:
CXDLiteTestDlg
IDC_CANCEL
IDC_START
IDC_XDLITECTRL1
IDCANCEL
IDOK

Messages:
WM_SETCURSOR
WM_SHOWWINDOW
WM_SIZE
WM_TCARD
WM_TIMER
WM_VKEYTOITEM
WM_VSCROLL

Member functions:
V DoDataExchange
W OnInitDialog ON_WM_INITDIALOG
W OnStart ON_IDC_START:BN_CLICKED
W OnTimer ON_WM_TIMER

Description: Indicates timeout interval for a timer has elapsed

And then, add CXDLite member function into the dialog box file.



Project: xDLiteTest

Class name: CXDLiteTestDlg

C:\...\xDLiteTestDlg.h, C:\...\xDLiteTestDlg.cpp

Control IDs:

IDC_CANCEL
IDC_START
IDC_XDLITECTRL1
IDCANCEL
IDOK

Description:

Member variable name: m_EKANAgent

Category: Control

Variable type: CXDLite

Description:
map to CXDLite member

2-3 Edit Code

Now we add code into the button Start, Exit, and Timer event.

On Start

On the Start, we make connection, setup timer parameter, then start timer play.

```
void CXDLiteTestDlg::OnStart()
{
    // TODO: Add your control notification handler code here
    m_nCount = 0;
    m_nColor = 0;
    m_EKANAgent.SetEkanPassword(L"admin");
    m_EKANAgent.SetEkanPort(7000);
    m_EKANAgent.SetEkanIP(L"192.168.100.96");
    if(m_EKANAgent.Connect() > 0)
```

```
     {
          GetDlgItem(IDC_START)->EnableWindow(FALSE);
          SetTimer(1,1000,NULL);
          AfxMessageBox(L"Connect OK!");
     }
     else
          AfxMessageBox(L"connect fail!");

}
```

On Timer

After the Timer start play, the program demo all the real-time drawing function to the LED display. We use m_nCount as counter to decide which function will be played.

```
void CXDLiteTestDlg::OnTimer(UINT nIDEvent)
{
     // TODO: Add your message handler code here and/or call default
     if(m_nCount < 3)
     {
            m_nColor = (m_nColor + 1) % 4;
          if(m_nColor == 0)
               m_nColor++;
            m_EKANAgent.SendMessage(0,0,m_nColor,0,L"Hello,world");
     }
     else if(m_nCount < 6)
     {
            m_nColor = (m_nColor + 1) % 4;
          if(m_nColor == 0)
               m_nColor++;
            m_EKANAgent.SendMessage(0,0,0,m_nColor,L"Hello,world");
     }
     else if(m_nCount == 6)
            m_EKANAgent.Clear();
     else if(m_nCount < 10)
     {
          m_nColor = (m_nColor + 1) % 4;
        if(m_nColor == 0)
            m_nColor++;
          m_EKANAgent.SendMessageS(0,5,m_nColor,0,L"Hello,small
font");
    }
    else if(m_nCount < 13)
    {
          m_nColor = (m_nColor + 1) % 4;
        if(m_nColor == 0)
            m_nColor++;
          m_EKANAgent.SendMessageS(0,5,0,m_nColor,L"Hello,small
font");
    }
     else if(m_nCount == 13)
         m_EKANAgent.Clear();
     else if(m_nCount < 17)
    {
          m_nColor = (m_nColor + 1) % 4;
        if(m_nColor == 0)
            m_nColor++;
          m_EKANAgent.DisplayDate(0,0,m_nColor,0);
    }
    else if(m_nCount < 20)
    {
          m_nColor = (m_nColor + 1) % 4;
        if(m_nColor == 0)
            m_nColor++;
          m_EKANAgent.DisplayDate(0,0,0,m_nColor);
    }
     else if(m_nCount == 20)
          m_EKANAgent.Clear();
    else if(m_nCount < 24)
    {
          m_nColor = (m_nColor + 1) % 4;
        if(m_nColor == 0)
            m_nColor++;
          m_EKANAgent.DisplayTime(0,0,m_nColor,0);
    }
    else if(m_nCount < 28)
    {
          m_nColor = (m_nColor + 1) % 4;
        if(m_nColor == 0)
            m_nColor++;
          m_EKANAgent.DisplayTime(0,0,0,m_nColor);
    }
    else if(m_nCount == 28)
```

```
        m_EKANAgent.Clear();
     else if(m_nCount < 32)
    {
          m_nColor = (m_nColor + 1) % 4;
        if(m_nColor == 0)
             m_nColor++;
        m_EKANAgent.FillRect(0,0,95,15,m_nColor);
    }
    else if(m_nCount == 32)
        m_EKANAgent.ClearRect(0,0,95,15);
    else if(m_nCount   == 33)
        m_EKANAgent.DrawLine(0,7,95,7,1);
    else if(m_nCount == 34)
        m_EKANAgent.Pixel(47,8,2);
    else if(m_nCount == 35)
        m_EKANAgent.Clear();
    else if(m_nCount == 36)
         m_EKANAgent.UploadFile(L"JPN.bmp");
    else if(m_nCount == 37)
         m_EKANAgent.UploadFile(L"GRAPH.bmp");
    else if(m_nCount == 38)
         m_EKANAgent.UploadFile(L"GB.bmp");
    else if(m_nCount == 39)
         m_EKANAgent.DisplayBMP(0,0,1,0,L"JPN.bmp");
    else if(m_nCount == 40)
        m_EKANAgent.DownloadFile(L"*.*",L"\\Download");
    else
    {
          KillTimer(1);
        m_EKANAgent.Disconnect();
        GetDlgItem(IDC_START)->EnableWindow(TRUE);
        return ;
    }

     m_nCount++;
     //CDialog::OnTimer(nIDEvent);
}
```

On Disconnect

Then finally we process disconnect event on the following code segment.

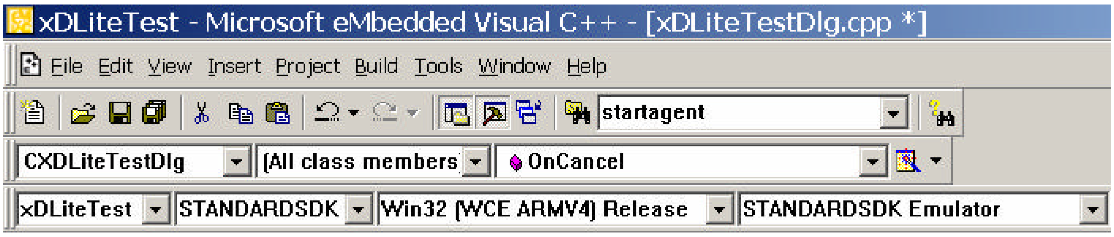
```
BEGIN_EVENTSINK_MAP(CXDLiteTestDlg, CDialog)
```

```
      //{{AFX_EVENTSINK_MAP(CXDLiteTestDlg)
    ON_EVENT(CXDLiteTestDlg, IDC_XDLITECTRL1, 1 /* Disconnected */,
OnDisconnectedXdlitectrl1, VTS_NONE)
     //}}AFX_EVENTSINK_MAP
END_EVENTSINK_MAP()

void CXDLiteTestDlg::OnDisconnectedXdlitectrl1()
{
     // TODO: Add your control notification handler code here
     KillTimer(1);
     GetDlgItem(IDC_START)->EnableWindow(TRUE);
     AfxMessageBox(L"Disconnected!");
}
```

Complier options

WinCon-8000 is using Strong-Arm as main processor, so you must select Win32 (WCE ARMV4) as compiler options. Then you can copy the .exe file into the WinCon-8000 that already registers xDLite.ocx for Windows CE version in chapter 1. To see the final result of this demo project.


xDLiteTest - Microsoft eMbedded Visual C++ - [xDLiteTestDlg.cpp *]

xDLiteTest | STANDARDSDK | Win32 (WCE ARMV4) Release | STANDARDSDK Emulator

About eSoftsystem Corp.

eSoftsystem Technology Corp. is the most innovative embedded solution provider. It has built up a team of world experts in embedded software and hardware systems to provide customer high-performance and high-quality embedded solution product and services.