Direct Data communication API for SerialMagic application

Keywords

SM — Serial Magic application.
CA — Client application

User's Messages:

```
#define WM_APP 0x8000
```

WM_SM_REGISTER (WM_APP + 0x100)

This message is used to register CA in application's list in SM. When SM gets incoming data it will send packet to all registered CAs.

Usage:

```
HRESULT hr = ::SendMessage(hSMWnd, WM_SM_REGISTER, NULL,
(LPARAM)m_hWnd);
```

hSMWnd — SM main window.
m_hWnd — CA window which will get data from SM.
hr — result of operation:
ERROR_SUCCESS — operation successful. Any other result — fail.

SM main window. The handle to this window clients application should get with using FindWindow function.

```
HWND hSMWnd = ::FindWindow(NULL, _T("SerialMagic Professional"));
```

WM_SM_UNREGISTER (WM_APP + 0x101)

This message is used to deregister CA in application's list in SM. SM will not post any data to this window anymore.

Usage:

```
HRESULT hr = ::SendMessage(hSMWnd, WM_SM_UNREGISTER, NULL,
               (LPARAM)m_hWnd);
```

hSMWnd — SM main window.
m_hWnd — CA window which will never get data from SM.

WM_SM_STARTSTOP (WM_APP + 0x102)

This message is used to start/stop SM. SM will return result of operation.

Usage:

```
HRESULT hr = ::SendMessage(hSMWnd, WM_SM_STARTSTOP, NULL, NULL);
```

hSMWnd — SM main window.
hr — result of operation:

```
	SM_RESULT_STARTED = 256
	SM_RESULT_STOPPED = 512;
```

Example:

```
HRESULT hr = ::SendMessage(hSMWnd, WM_SM_STARTSTOP, NULL, NULL);
switch(hr)
{
	case SM_RESULT_STARTED:
		//SM started successfully
		//TODO:
		break;
	case SM_RESULT_STOPPED:
		//SM stopped successfully
		//TODO:
		break;
	default:
		//operation failed
		//TODO:
		break;
}
```

WM_SM_GETSTATE (WM_APP + 0x105)

This message is used to get current state of SM.

Usage:

```
HRESULT hr = ::SendMessage(hSMWnd, WM_SM_GETSTATE, NULL, NULL);
```

hSMWnd — SM main window.
hr — result of operation:

```
	SM_RESULT_STARTED = 256
	SM_RESULT_STOPPED = 512;
```

Copy Data messages:

WM_COPYDATA

This is a system native message which is used for data exchange between CA and SM.

Get Data from SM

SM uses this message to pass incoming data from scanner to CA.CA should be able to catch this message and process incoming data. Incoming data could be passed into CA in 2 modes:

- 'raw' format
- 'barcode' format

C++ Example:

```
// add listener for WM_COPYDATA message in the client application
// for more details about message processing see MSDN

BOOL CDirectDataTestDlg::OnCopyData(CWnd* pWnd, COPYDATASTRUCT* pCopyDataStruct)
{
      char id1[255];
      CString tempStr;
      tempStr.Format("Have data from Scanner %d: %d", pCopyDataStruct->dwData+1,
            pCopyDataStruct->cbData);
      m_ListLog.AddString(tempStr);
      memset(id1, 0, 255);
      strncpy(id1, (char*)pCopyDataStruct->lpData, pCopyDataStruct->cbData);
      m_ListLog.AddString(id1);
      m_ListLog.SetCurSel(m_ListLog.GetCount()-1);
      return CDialog::OnCopyData(pWnd, pCopyDataStruct);
}
```

VB.NET Example:

```
    Public Sub CopyData(ByRef m As System.Windows.Forms.Message)
        Dim cds As COPYDATASTRUCT
        'MsgBox("CopyData")
        cds = CType(Marshal.PtrToStructure(m.LParam, GetType(COPYDATASTRUCT)),
            COPYDATASTRUCT)

        If (cds.cbData > 0) Then
            Dim tmpString As String = "Have incoming data:" +
                  cds.cbData.ToString()
            logListBox.Items.Add(tmpString)

            Dim datab() As Byte
            datab = New Byte() {}
            ReDim datab(cds.cbData)
            Marshal.Copy(cds.lpData, datab, 0, cds.cbData)
            tmpString = System.Text.Encoding.ASCII.GetString(datab, 0,
                  cds.cbData)

            logListBox.SelectedIndex = logListBox.Items.Add(tmpString)
        End If
    End Sub
```

Send Command to SM

CA uses WM_COPYDATA message to send command to scanner over SM. WPARAM of SendMessage function pass habdle of the SM main window and the LPARAM it is pointer to the COPYDATASTRUCT structure. The **lpData** data field in the COPYDATASTRUCT structure is a pointer to data to be passed to the SM. The **cbData** field of the COPYDATASTRUCT Specifies the size, in bytes, of the data pointed to by the **lpData** member.

C++ Example:

```
COPYDATASTRUCT cd;

cd.cbData = nLength;//length of  szBuffer (e.g.sizeof(szBuffer))
cd.dwData = 0;//should be 0
cd.lpData = (PVOID)&szBuffer[0];//command data buffer

BOOL result = ::SendMessage(hSMWnd, WM_COPYDATA, (WPARAM)m_hWnd, (LPARAM)&cd);
```

VB.NET Example:

```
Dim lParam As IntPtr = IntPtr.Zero
Dim bytesPtr As IntPtr = IntPtr.Zero
Dim bytes() As Byte = System.Text.Encoding.ASCII.GetBytes(dataTextBox.Text)

bytesPtr = LocalAlloc(LPTR, bytes.Length + 1)

Marshal.Copy(bytes, 0, bytesPtr, bytes.Length)

Dim cbs As COPYDATASTRUCT = New COPYDATASTRUCT()
cbs.dwData = 0
cbs.cbData = dataTextBox.Text.Length
cbs.lpData = bytesPtr

lParam = LocalAlloc(LPTR, Marshal.SizeOf(cbs))
Marshal.StructureToPtr(cbs, lParam, False)
Dim res As Integer = SendMessage(foundWindow, WM_COPYDATA, MyBase.Handle,
  lParam)
If (res = 1) Then
  '- success
Else
  '- fail
End If

LocalFree(bytesPtr)
LocalFree(lParam)
```

Result.

Success if returned value is not NULL, otherwise — fail.

Remarks.

This feature can work in 2 modes:

- Raw packet. It is packet from scanner without any changing
- Barcode packet. In this mode SM will get incoming data from scanner, parse it (remove 'Start Flag' and 'Action Byte') and if incoming data is correct then SM passes barcode to CA.

Direct data mode should be enabled for both cases.

'Raw packet' mode
SM should use device type 'Other' and 'Byte mode' should be enabled.

'Barcode packet' mode
Device type could be any. 'Packet mode' should be selected.