[**第二部分：OLE 数据传送（Data Transfer）**](http://blog.csdn.net/leehong2005/article/details/8609495)

分类： [COM](http://blog.csdn.net/leehong2005/article/category/1349085) 2013-02-25 14:44 544人阅读 [评论](http://blog.csdn.net/leehong2005/article/details/8609495#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/leehong2005/article/details/8609495#report)

目录[(?)[+]](http://blog.csdn.net/leehong2005/article/details/8609495)

    这一部分讲一讲OLE数据传输相关知识，多数来自于MSDN和网上其同志们的观点，加以总结而成。  
    COM接口提供了一种用于在不同的应用程序中交换数据的机制，这就是我们要讲的数据对象，对应的COM接口就是IDataObject。     
    大多数平台，包括Windows，都定义了一个用于在应用程序之间传输数据的标准协议，基于一系列的剪切板的函数。应用程序使用这些函数可以共享数据，即使它们的数据格式相差甚远，一般说来，剪切板有两个缺点：  
    1，不能指定目标设备，不灵活。  
    2，如果传输的数据很大，就有可能用到虚拟内存，效率不高。

**1.OLE数据描述**

    一 般情况下，数据传输会用到两种方式，一种是剪切板，另一种是DragDrop。不管用哪种方式，都要用到IDataObject这个COM接口，在关注 IDataObject之前，我们必须要了解数据传输所要用到一些辅助结构，FORMATETC和STGMEDIUM，它们用来描述OLE数据的格式和存 储等信息。  
   
    FORMATETC结构体，用来表示IDataObject提供或接收的数据类型，是标准Windows粘贴格式（CF\_TEXT）的扩展，它包含了数据格式外，还有一些其他信息，看看它的定义：

**[cpp]** [view plaincopy](http://blog.csdn.net/leehong2005/article/details/8609495)

1. typedef struct tagFORMATETC
2. {
3. CLIPFORMAT cfFormat;     // 数据格式，如CF\_TEXT
4. DVTARGETDEVICE \*ptd;     // 一般为NULL，目标设备
5. DWORD dwAspect;          // DV\_CONTENT rendering详细信息
6. LONG lindex;             // 一般为-1
7. DWORD tymed;             // 用于数据传输的存储媒体（HGLOBAL，IStream)
8. } FORMATETC, \*LPFORMATETC;

    FORMATETC结构的成员描述如下：  
    cfFormat：数据格式，可以是系统定义的（CF\_TEXT、CF\_BITMAP等），也可以是用RegisterClipboardFormat注册的自定义格式。  
    ptd：一般为NULL，提供已经rendered数据的设备信息，正常的粘贴板操作和拖放都是NULL。  
    dwAspect：描述数据的细节信息，有DV\_CONTENT、DVASPECT\_THUMBNAIL等。  
    lindex：最常用的值是-1。  
    tymed：描述用于存储数据的存储媒体，TYMED\_XXX等值。  
    具体的描述，大家可以参考MSDN，它上面讲得比较详细。

**2.OLE数据存储**

    结构体STGMEDIUM(STORAGE MEDIUM的缩写)提供一个用来存储数据的容器，因此叫存储媒体：

**[cpp]** [view plaincopy](http://blog.csdn.net/leehong2005/article/details/8609495)

1. typedef struct
2. {
3. DWORD tymed;                               // TYEMD\_HGLOBAL、TYPED\_ISTREAM等。
4. union
5. {
6. HBITMAP        hBitmap;
7. HMETAFILEPICT  hMetaFilePict;
8. HENHMETAFILE   hEnhMetaFile;
9. HGLOBAL        hGlobal;
10. LPWSTR         lpszFileName;
11. IStream        \*pstm;
12. IStorage       \*pstg;
13. };
14. IUnknown \*pUnkForRelease;
15. } STGMEDIUM;

**3.IDataObject成员**

    下面看一看IDataObject接口的成员方法：  
    **GetData**：Render在FORMATETC结构体中描述的数据，并通过STGMEDIUM结构体来传递数据。  
    **GetDataHere**：同上一方法相似，只是STGMEDIUM结构的内在是由调用者分配的。  
    **QueryGetData**：决定数据对象是否能够render在FORMATETC结构中描述的数据。  
    **GetCanonicalFormatEtc**：提供一下潜在不同的但逻辑上相同的FORMATETC结构。  
    **SetData**：提供一个用FORMATETC结构和STGMEDIUM结构描述的数据源对象。  
    **EnumFormatEtc**：创建并返回一个IEnumFORMATETC接口的指针来枚举数据对象支持的FORMATETC。  
    **DAdvise**：创建一个在数据对象和通知接收器之间的连接，因此通知接收器收到数据对象中通知的改变。  
    **DUnadvise**：销毁一个前面使用DAdvise方法安装的通知。  
    **EnumDAdvise**：创建和返回一个指向枚举当前通知的接口指针。  
  
    上面的接口，不用每个方法都要实现，只实现几个重要的，如GetData、SetData等。

**4.用IDataObject来访问剪切板(Clipboard)**

    先看看简单的访问剪切板的代码，很了一下IDataObject如何使用。  
    代码如下：

**[cpp]** [view plaincopy](http://blog.csdn.net/leehong2005/article/details/8609495)

1. void TestGetDataFromClipboard()
2. {
3. OleInitialize(NULL);
4. IDataObject \*pDataObject = NULL;
5. HRESULT hr = OleGetClipboard(&pDataObject);
6. if (SUCCEEDED(hr))
7. {
8. FORMATETC fmtetc = { CF\_TEXT, 0, DVASPECT\_CONTENT, -1, TYMED\_HGLOBAL };
9. STGMEDIUM stgmed;
10. hr = pDataObject->GetData(&fmtetc, &stgmed);
11. if (SUCCEEDED(hr))
12. {
13. PVOID pData = GlobalLock(stgmed.hGlobal);
14. CHAR \*pText = (CHAR\*)pData;
15. MessageBoxA(NULL, pText, "GetDataFromClipboard", 0);

18. GlobalUnlock(stgmed.hGlobal);
19. ReleaseStgMedium(&stgmed);
20. }
21. SAFE\_RELEASE(pDataObject);
22. }
23. OleUninitialize();
24. }

**[cpp]** [view plaincopy](http://blog.csdn.net/leehong2005/article/details/8609495)



    上面的方法演示了如何从剪切板里面取得字符串数据，最本质的方法还是通过IDataObject::GetData方法来取得数据。取数据的类型在FORMATETC结构体里面指定。

    注意，对于CF\_TEXT，根据MSDN上面的解释，字符串必须是ANSI，所以要把通过GlobalLock得到的指针（PVOID）转换成CHAR\*，最后调用ReleaseStgMedium来释放程序分配的内存。  
    好了，到这里，应该明白数据传输格式及数据存储是怎么回了，这里只是一个大概，还有很多细节需要仔细阅读MSDN才行。

OleInitialize(NULL);

IDataObject\* pDataObject = NULL;

TCHAR szText[MAX\_PATH\*100] = {0};

long nSzie;

HRESULT hr = S\_OK;

hr = OleGetClipboard(&pDataObject);//獲取剪切板ole數據

if(hr == S\_OK)

{

//初始化數據模型

//初始化數據

FORMATETC format;

STGMEDIUM data;

format.cfFormat = CF\_TEXT;

format.dwAspect = DVASPECT\_CONTENT;

format.lindex = -1;

format.ptd = NULL;

format.tymed = TYMED\_HGLOBAL;

memset(&data,0x00,sizeof(data));

data.tymed = TYMED\_HGLOBAL;

hr = pDataObject->GetData(&format,&data);

if(hr == S\_OK)

{

/\*

//直接讀取數據

PVOID pdata = (PVOID)GlobalLock(data.hGlobal);

TCHAR\* pbuffer = (TCHAR\*)pdata;

GlobalUnlock(data.hGlobal);

\*/

IStream\* pstream =NULL;

hr = CreateStreamOnHGlobal(data.hGlobal, TRUE, &pstream);

if(hr== S\_OK)

{

LARGE\_INTEGER l;

ULARGE\_INTEGER l2;

l.HighPart = 0;

l.LowPart = 0;

pstream->Seek(l, STREAM\_SEEK\_SET, &l2);

ULONG nRead = 1;

TCHAR szBuffer[MAX\_PATH] = {0};

ULONG n = MAX\_PATH;

int i = 0;

while(nRead > 0)

{

pstream->Read(szBuffer, n, &nRead);

if(nRead > 0)

{

memcpy(szText+i,szBuffer,nRead);

i += nRead;

}

}

::SetDlgItemText(m\_hWnd, IDC\_DATA, szText);

pstream->Release();

}

ReleaseStgMedium(&data);

}

pDataObject->Release();

}

OleUninitialize();