Copying a file to the clipboard so you can paste it into Explorer or an email message or whatever

devblogs.microsoft.com/oldnewthing/20130520-00

May 20, 2013

Raymond Chen

Today's Little Program takes a fully-qualified file name from the command line and puts that file onto the clipboard. Once there, you can paste it into an Explorer window, or into an email message, or a word processing document, or anybody else who understands shell data objects.

```
#include <windows.h>
#include <shlobj.h>
#include <atlbase.h>
#include <shlobj.h>


class COleInitialize {
public:
 COleInitialize() : m_hr(OleInitialize(NULL)) { }
 ~COleInitialize() { if (SUCCEEDED(m_hr)) OleUninitialize(); }
 operator HRESULT() const { return m_hr; }
 HRESULT m_hr;
};


// GetUIObjectOfFile incorporated by reference


int __cdecl wmain(int argc, PWSTR argv[])
{
 COleInitialize init;
 CComPtr<IDataObject> spdto;


 if (SUCCEEDED(init) &&
     argc == 2 &&
     SUCCEEDED(GetUIObjectOfFile(nullptr, argv[1], IID_PPV_ARGS(&spdto))) &&
     SUCCEEDED(OleSetClipboard(spdto)) &&
     SUCCEEDED(OleFlushClipboard())) {
  // success
 }


 return 0;
}
```

The `COleInitialize` class is just the OLE counterpart to the `CCoInitialize` class we saw some time ago.

All the program does is take the file name on the command line, asks the shell for the corresponding data object, then puts that object onto the clipboard, erasing what was there before.

Once the data is on the clipboard, our job is done so we exit.

No, wait! If you exit while your application has data on the clipboard, that clipboard data may be lost. The documentation for `OleSetClipboard` notes:

> If you need to leave the data on the clipboard after your application is closed, you should call **OleFlushClipboard** rather than calling **OleSetClipboard** with a **NULL** parameter value.

Therefore, we stick in a call to `OleFlushClipboard` before exiting. This forces any delay-rendered content to be rendered immediately, because we ain't gonna be around to delay-render it no more.

Note that the file on the command line must be fully-qualified, because we pass it straight to `GetUIObjectOfFile`, which expects a fully-qualified path. Fixing the program to allow relative paths (and to actually print error messages and stuff) is left as an exercise, because Little Programs don't deal with annoying details like error checking and reporting.

Raymond Chen

Follow