Instead of trying to figure out what shortcut class to use, just ask the shell to do it for you

devblogs.microsoft.com/oldnewthing/20100702-00

July 2, 2010

Raymond Chen

If a shell namespace item has the `SFGAO_LINK` attribute, then it is a shortcut to another location. The most common type of shortcut is the `.lnk` file, which you can load by creating the `CLSID_ShellLink` object and using `IPersistFile::Load`, but what if you have some other type of shortcut? How do you know what CLSID to use?

Since anybody can create their own shortcut file types, a hard-coded list mapping file extensions to CLSIDs is not going to work for long. But fortunately, you don't have to know how to look up the CLSID for a particular shortcut; you can just ask the namespace to do it for you by asking for the `IShellLink` UI object.

```
#include <windows.h>
#include <shlobj.h>
#include <ole2.h>
#include <stdio.h>
#include <tchar.h>
#include <shellapi.h>
// GetUIObjectOfFile function incorporated by reference
int __cdecl _tmain()
{
  int argc;
  LPWSTR *argv = CommandLineToArgvW(GetCommandLineW(), &argc);
  if (argv == NULL || argc != 2) return 0;
  if (SUCCEEDED(CoInitialize(NULL))) {
    IShellLink *psl;
    if (SUCCEEDED(GetUIObjectOfFile(NULL, argv[1], IID_PPV_ARGS(&psl)))) {
      TCHAR sz[MAX_PATH];
      if (SUCCEEDED(psl->GetPath(sz, MAX_PATH, NULL, 0))) {
        _tprintf(TEXT("-> %ls\n"), sz);
      }
      else _tprintf(TEXT("GetPath failed\n"));
      psl->Release();
     }
     else _tprintf(TEXT("GetUIObjectOf failed\n"));
    CoUninitialize();
  }
  LocalFree(argv);
  return 0;
}
```

I've limited myself to files here for simplicity of exposition, and I assume that you've passed a fully-qualified path on the command line. Of course, you can have shortcuts to non-file objects as well, and for those shortcuts, `IShellLink::GetPath` is unlikely to return an actual file path. (In fact, for things like shortcuts to the Control Panel, they're unlikely to return anything at all.) I've also used the `CommandLineToArgvW` function instead of the built-in `argc` and `argv` because the `GetUIObjectOfFile` function wants a Unicode file name, but the C runtime's `argv` is a `TCHAR *` string, which might not be Unicode.

Let's take this program for a spin.

Warning: I am using hard-coded paths. In real life, you would use appropriate functions to obtain the paths to the files you care about. (Actually, in real life, you probably will have a pidl to the item rather than a path, so the issue of paths disappears.)

```
>set STARTMENU=%APPDATA%\Microsoft\Windows\Start Menu\Programs
>scratch "C:\ProgramData\Microsoft\Windows\Start
Menu\Programs\Accessories\Calculator.lnk"
-> C:\Windows\System32\calc.exe
>scratch "%STARTMENU%\Internet Explorer.lnk"
-> C:\Program Files\Internet Explorer\iexplore.exe
```

Okay, these are your regular `.lnk` files, so there's nothing special going on here. Let's try something fancier, like a symbolic link.

```
>echo > blah.txt
>mklink other blah.txt
symbolic link created for other <<===>> blah.txt
>scratch "%CD%\other"
-> C:\test\blah.txt
```

Via the Add Network Location wizard, I created a network location (which is internally represented as a Folder Shortcut). Let's see what happens with that:

```
> scratch "%APPDATA%\Microsoft\Windows\Network Shortcuts\Tools"
-> \\live.sysinternals.com\tools
```

How about Internet shortcuts?

```
> scratch "%USERPROFILE%\Favorites\MSN Websites\MSN.url"
-> http://go.microsoft.com/fwlink/?LinkId=54729
```

OneClick shortcuts? (MS Space is an internal application which lets you view floor plans of every Microsoft building, book conference rooms, reserve touchdown space, that sort of thing.)

```
> scratch "%STARTMENU%\MS Space.appref-ms"
GetUIObjectOf failed
```

Huh? What happened?

It so happens that the people who wrote the shortcut handler for OneClick applications only bothered to implement the Unicode version of the `IShellLink` interface. We built our application as ANSI, so our attempt to get the `IShellLinkA` interface failed. But that's easily worked around:

```
#define _UNICODE
#define UNICODE
#include <windows.h>
#include <shlobj.h>
#include <ole2.h>
...
```

(In real life, your program would probably first ask for the Unicode interface, and if the call fails, then ask for the ANSI interface.)

With the Unicode version of the program, the shortcut resolves:

```
> scratch "%STARTMENU%\MS Space.appref-ms"
-> C:\Users\...\MSSpaceDeploy.exe
```

(I elided some of the ugly path because, well, it’s ugly. The full unabbreviated path is 139 characters, most of which is just hex digits.)

Anyway, the point for today wasn’t the minutiae of obtaining shortcut targets from shell namespace items. It was the principle that if you want something from the shell namespace, the `IShellFolder::GetUIObjectOf` method will often get it for you.

Raymond Chen

Follow