Why does my shortcut to a nonexistent file end up with spaces changed to underscores?

devblogs.microsoft.com/oldnewthing/20180509-00

May 9, 2018

Raymond Chen

A customer wanted to create shortcuts to a network drive that hadn't yet been mapped. The idea is that they would create these shortcuts pointing at a network drive `N:` and deploy them. When the user logs in, a script will map the `N:` drive to an appropriate network server where the files will exist. The customer cannot have the shortcuts point directly at the server via a UNC because the UNC connection requires special credentials that users won't have.

The customer found that if they tried to create the shortcut to a nonexistent network drive, the path was being corrupted. Specifically, spaces were being changed to underscores. Here's a sample program. (Error checking elided for expository purposes.)

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

int __cdecl wmain(int, wchar_t**)
{
  CCoInitialize init;
  CComPtr<IShellLink> link;
  CoCreateInstance(CLSID_ShellLink, nullptr, CLSCTX_INPROC_SERVER,
                   IID_PPV_ARGS(&link));
  link->SetPath(L"N:\\dir\\some file that doesn't exist.txt");
  CComQIPtr<IPersistFile>(link)->Save(L"C:\\test\\test.lnk", TRUE);
  return 0;
}
```

Why is the shortcut being saved incorrectly?

The short answer is that the shortcut is being saved incorrectly because you set an invalid path. When you call `SetPath` and pass a path to something that doesn't exist, the shortcut code panics and says, "Oh no, what is this thing?" It then goes into a series of compatibility

heuristics to try to make sense of the invalid parameter.

- Did the caller erroneously put quotation marks around the path? If so, then remove them.
- Maybe the caller intended to link to a program but forgot the `.exe` extension. See if adding `.exe` helps.
- Maybe the caller intended to link to a program but didn't pass a fully-qualified path. Search the path for a matching program and use the first one you find, if any.
- Maybe the caller used slashes instead of backslashes. Fix any wayward slashes.
- Maybe, maybe, maybe…

These are all heuristics and should not be relied upon.

The heuristic that is triggering in this case is one that says, "Well, I see a drive letter, and the drive won't tell me if it supports long file names, so I'll assume it doesn't, and I'm going to replace characters that aren't legal in short file names with underscores. Maybe that'll help." It doesn't help, but the damage is done. The spaces became underscores.

The critical step is the fact that there is no `N:` drive, which means that when the code checks whether the volume in drive `N:` supports long file names, the answer is "Volume? What volume?" That's why this occurs only when you try to create a shortcut to a nonexisting volume. If you try to create a shortcut to a nonexistent file on an existing volume, then this heuristic won't kick in, because the existing volume supports long file names.

The heuristics are constantly being tweaked. In the Windows 10 Creators Update, the heuristic about short file names was removed, presumably on the theory that everybody worth supporting supports long file names now.

Okay, so what is the correct thing to do if you want to create a shortcut to a file that doesn't exist yet?

Instead of using `IShellLink::SetPath`, use `IShellLink::SetIDList` with a simple pidl.

```
int __cdecl wmain(int, wchar_t**)
{
  CCoInitialize init;
  CComPtr<IShellLink> link;
  CoCreateInstance(CLSID_ShellLink, nullptr, CLSCTX_INPROC_SERVER,
                   IID_PPV_ARGS(&link));

  WIN32_FIND_DATAW fd = {};

  // it's a file (not a directory)
  fd.dwFileAttributes = FILE_ATTRIBUTE_NORMAL;

  CComHeapPtr<ITEMIDLIST_ABSOLUTE> simplePidl;
  CreateSimplePidl(&fd,
    L"N:\\dir\\some file that doesn't exist.txt",
    &simplePidl);

  link->SetIDList(simplePidl);

  CComQIPtr<IPersistFile>(link)->Save(L"C:\\test\\test.lnk", TRUE);
  return 0;
}
```

A simple pidl lets you talk about something that might not exist. We use it here to create a shortcut to a file that might not exist.

Raymond Chen

Follow