The only thing you can do with display names is display them

devblogs.microsoft.com/oldnewthing/20070313-00

March 13, 2007

Raymond Chen

There are many functions that return strings called “display names”. And the only thing you can do with display names is display them. Don’t assume that the string actually means anything, because it doesn’t. Theoretically, a function like `SHGetFileInfo` could be implemented as

```
...
 if (uFlags & SHGFI_DISPLAYNAME) {
  StringCchCopy(psfi->szDisplayName, MAX_PATH, TEXT("Booga!"));
 }
...
```

and your program should still work.

(Of course, this is merely a ground rule. Specific functions may have exceptions. For example, the `IShellFolder::GetDisplayNameOf` has a special flag `SHGDN_FORPARSING` flag which explicitly indicates that the string returned is designed to be parsed.)

The purpose of a “display name” is to be a string suitable for displaying to the user. The display name for a file, for example, might be the file name, or it might have the extension removed, or the name might even be translated into the user’s preferred language! For example, on an English system with the German multilanguage pack installed and active, asking for the display name of `C:\Documents and Settings\Raymond\My Documents` will return `Eigene Dateien` because that’s the name for My Documents in German.

If your program assumed that the display name of `C:\Documents and Settings\Raymond\My Documents` would be something like “My Documents”, your program is in for a big surprise when I run it.

One of my colleagues was investigating a bug reported against a program that wouldn’t run. It claimed that the CD-ROM was not in the drive, even though it was. Why can’t the program find its CD-ROM?

After a few days' investigation, my colleague found the reason. The program wanted to find its CD-ROM, so it walked through all 26 drive letters and called `SHGetFileInfo` passing the `SHGFI_DISPLAYNAME` flag. Something like this:

```
// The actual code was much, much more convoluted than this.
char LookForCD(LPCTSTR pszVolumeLabel)
{
 for (TCHAR chDrive = TEXT('A'); chDrive <= TEXT('Z'); chDrive++) {
  TCHAR szRoot[4];
  wsprintf(szRoot, TEXT("%c:\\"), chDrive);
  SHFILEINFO sfi;
  if (SHGetFileInfo(szRoot, 0, &sfi, sizeof(sfi), SHGFI_DISPLAYNAME)) {
   TCHAR szExpected[MAX_PATH];
   wsprintf(szExpected, TEXT("%s (%c:)"), pszVolumeLabel, chDrive);
   if (strcmp(szExpected, sfi.szDisplayName) == 0) {
    return chDrive; // Found it!
   }
  }
 }
 return 0; // not found
}
```

The program asked for the display name of each drive and looked for one whose display name was of the form `LABEL (D:)`, where `LABEL` is the volume label they're looking for and `D:` is the drive letter.

In othe words, they were trying to interpret the display name.

Don't do that. There is no guarantee that the display name for a CD-ROM will be of any particular form. The default in Windows XP happens to be `LABEL (D:)`, but there are a variety of system policies that can be used to change this, and if any of those policies is in effect, the program can't find its CD-ROM and refuses to run.

(For the record, the correct way of doing this would be to pass each drive letter to the `GetDriveType` function to see if it is a `DRIVE_CDROM`; for each such drive, call `GetVolumeInformation` to get the CD-ROM's volume label.)

So remember, display names are just for display purposes.†

Nitpicker's corner

†See parenthetical remark at the top of this entry for clarification. The sentence intentionally overlooked the exceptions in order to provide a punchier ending to the story. It's called *writing style* and is a valid literary technique.

Raymond Chen

Follow