Why does the common file save dialog create a temporary file and then delete it?

devblogs.microsoft.com/oldnewthing/20140429-00

April 29, 2014

Raymond Chen

When you call `GetSaveFileName`, the common file save dialog will ask the user to choose a file name, and just before it returns it does a little create/delete dance where it creates the file the user entered, and then deletes it. What's up with that? This is a leftover from the ancient days of 16-bit Windows 3.1, back when file systems were real file systems and didn't have this namby-pamby "long file name" or "security" nonsense. (Insert sound effect of muscle flexing and gutteral grunting.) Back in those days, the file system interface was MS-DOS, and MS-DOS didn't have a way to query security attributes because, well, the file systems of the day didn't have security attributes to query in the first place. But network servers did. If you mapped a network drive from a server running one of those fancy new file systems, then you were in this case where your computer didn't know anything about file system security, *but the server did*. The only way to find out whether you had permission to create a file in a directory was to try it and see whether it worked or whether it failed with the error `ERROR_ACCESS_DENIED` (or, as it was called back in the MS-DOS days, "5"), Another reason why a server might reject a file name was that it contained a character that, while legal in Windows, was not legal on the server. At the time, the most common reason for this was that you used a so-called "extended character" (in other words, a character outside the ASCII range like an accented lowercase e) which was part of your local code page but not on the server's. Yet another possibility was that the file name you chose would exceed the server's path name limit. For example, suppose the server is running Windows for Workgroups (which has a 64-character maximum path name limit), and it shared out C:\some\deep\directory\on\the\server as \\server\share. If you mapped M: to \\server\share, then the maximum path name on M: was only about 30 characters because C:\some\deep\directory\on\the\server used up half of your 64-character limit. The only way to tell whether the file could be created, then, was to try to create it and see what happens. After creating the test file (to see if it could), the common file save dialog immediately deleted it in order to cover its tracks. (This could lead to some weird behavior if users picked a directory where they had permission to create files but no permission to delete files that they created!)

This “test to see if I can create the file by creating it” behavior has been carried forward ever since, but you can suppress it by passing the `OFN_NOTESTFILECREATE` flag.

Raymond Chen

Follow