Not all short filenames contain a tilde

14 Apr 2004 7:00 AM | 50

I'm sure everybody has seen the autogenerated short names for long file names. For the long name "Long name for file.txt", you might get "LONGNA~1.TXT" or possibly "LO18C9~1.TXT" if there are a lot of collisions.

What you may not know is that sometimes there is no tilde at all!

Each filesystem decides how it wants to implement short filenames. Windows 95 uses the "~n" method exclusively. Windows NT adds the hexadecimal hash overflow technique. But some filesystems (like Novell) just truncate the name. "Long name for file.txt" on a Novell server will come out to just "LONGNAME.TXT".

So don't assume that all short names contain tildes. They don't. This means no cheating on skipping a call to GetLongFileName if you don't see any tildes, since your optimization is invalid on Novell networks.

Blog - Comment List MSDN TechNet

Comments

Steve Sheppard
14 Apr 2004 7:04 AM
#
Novell still has networks? :)

byron
14 Apr 2004 7:09 AM
#
i've had to do plenty of workarounds due to some assumptions that were broken by novell.

eg. don't assume that every non-root directory will contain "." and "..". not so in novell.

CC
14 Apr 2004 7:49 AM
#
"i've had to do plenty of workarounds due to some assumptions that were broken by novell."

I have to ask, is this really a situation where novell has assumed the wrong thing? I assume the ~ thing is a byproduct of the long filename support, but do the docs say that you can expect a ~ or not, if not then I don't see why this is particularly wrong.

Relying on things like a tilde appearing in a filename or not, would generally make me itch

...

byron
14 Apr 2004 8:40 AM
#
"is this really a situation where novell has assumed the wrong thing"

no, i mean there were assumptions made by developers working on code that i've inherited. the developers had made assumptions which were not wise, given novell doesn't work the same as windows. novell isn't wrong, just different.

i guess i should of worded that sentance as:

"i've had to do plenty of workarounds due to some poor assumptions made by prior developers that were broken by novell's implementation."

".. would generally make me itch"

indeed. the specific case we had was window's directoryExists function always returned false when pointed at some particular version of samba's shares. of course a major client had that version of samba and couldn't upgrade.

so the code was changed to do a findFirst against anyFile, on the assumption that every directory has '.' and '..'

Andy
14 Apr 2004 9:58 AM
#
Long file names will contain a tilde for the first 9999 files with the same stem ie longfi~1 longfi~2. after that it seems to start randomizing the end bit. I know as I spent four days on a training course watching the machine slowly make files starting at longfilename1 through to longfilename10000000000 to see how far it would get. After all there would seem to be a (theoretical) limit to the number of files in a directory and i was curious to see what happened. The only thing that happened was that the process of making a file took longer and longer and longer as the number of files increased. Sometimes it would take 15 or so minutes to write one file.

Marc Wallace
14 Apr 2004 10:31 AM
#
I believe there was also a setting in Win95 where you could control how long filenames were converted into 8.3 -- whether Windows would use the ~n format, or use a kind of truncation.

The truncation did some sort of incrementing, so "filename1.txt" would become "filename.txt", and "filename2.txt" would then be either "filenamf.txt" or "filenam2.txt".

Ah, here we go. Create a DWORD at HKLM\System\CurrentControlSet\Control\FileSystem\NameNumericTail and set it to 0. (1 or missing means it uses tildes).

Dumb User
14 Apr 2004 11:00 AM
#
I have a quick question, I don't quite understand how you have a Novell file system on a windows machine, since isn't this call platform specific?

Anonymous Coward
14 Apr 2004 12:08 PM
#
A fun thing that happened to me. My directory had a whole bunch of *.htm files, generated by a help system. I placed in my own files index.html, others ending in .html etc. I wanted to delete the *.htm files, and did the obvious thing:

del *.htm

It deleted all the htm <b>AND</b> html files. This is on XP with NTFS.

The moral of the story is that only the first 3 characters of the extension matter if you only specify the first three.

Raymond Chen
14 Apr 2004 12:13 PM
#
Novell provides network file servers. You can access Novell file servers from a Windows client.

Wildcards match both the long and short name. The short name of "foo.html" is (say) "foo~1.htm" which matches *.htm.

Jonathan Payne
14 Apr 2004 12:46 PM
#
(Sorry this is off topic but it kind of relates to Anonymous Coward’s post)

One thing I am not so keen on Windows XP is the automagic deleting of HTML because the operating system is too broken to do the right thing (OK, maybe not so fair but this bug really annoyed me at the time).

I saved one of my web pages in IE using the "Web Page, Complete" option so I could edit it. I spent an hour or so editing the HTML file and then, just before uploading the file, I deleted the folder IE had created at the same time as making the file. Guess what happened...... Windows decided that the folder and the file were somehow linked and deleted both. As I had the recycle bin turned off, I lost all my work.

You can see this effect by saving any web page in IE and using the "Web page, complete" option and then deleting only the file or only the folder. Note that Explorer is kind enough to warn you if you rename the folder but happily deletes the folder without asking.

I am guessing that when this 'feature' was designated this way because someone thought that most users would only remember to delete the file or the folder but not both. I think the correct solution would be for IE not to link the files in this way or to put the HTML and associated files into a folder and associate the folder with the HTML or to store web pages in a single file.

Raymond Chen
14 Apr 2004 1:02 PM
#
You can save in "Web Archive, single file" format if you want everything bundled into a single file.

Connected files are a feature invented by Office. You can disable them via a system policy if you don't like them.

Unclear what you mean "put the HTML and associated files into a folder and associate the folder with the HTML" - that's what happened. "Foo_files" is assocated with "foo.htm".

John Topley
14 Apr 2004 1:35 PM
#
You can also disable them within Folder Options in Windows Explorer. There are three settings for "Managing pairs of Web pages and folders".

Norman Diamond
14 Apr 2004 5:12 PM
#
> For the long name "Long name for file.txt",
> you might get "LONGNA~1.TXT" or
> possibly "LO18C9~1.TXT"

Do you know why Windows 98 is unaware of that fact? The Scandisk command of Windows 98 diagnoses cases such as "LO18C9~1.TXT", saying that the short filename is broken and offering to repair it.

The Scandisk command of Windows 98 is still vitally important. After a blue screen, if you boot Windows 98 and let it run Scandisk, it writes a log file on the C: partition listing all of the files that it deletes. Then you can know which files you'd better restore from backup. But after a blue screen, if you boot Windows 2000 or XP, Chkdsk only displays a list of files on the screen, scrolling faster than could possibly be read, scrolling faster than can be copied down with pen and paper. Windows 2000 and XP know that you don't need the list of files it's deleting, they know you don't want to restore them from backups.

One time Windows 2000 deleted about 20,000 files that way during boot. The reason I had an estimate was that I had done an anti-virus scan a day or so before the blue screen and it reported the number of files scanned. During boot, although the scrolling was too fast to read, I noticed that a ton of them were in the Temporary Internet Files folder. (But does this mean 100% of them were? I doubt it, I think some important files probably got deleted too.) After it finished booting, I took a look at the Temporary Internet Files folder and it was empty, so that yielded an estimate that a few thousand files had gone away. Then I ran an anti-virus scan again and the number of files scanned was about 20,000 less than the previous day's scan. Since that time, on any dual-boot machine, I never let Windows 2000 or XP run Chkdsk during boot, I always reboot to Windows 98 and run Scandisk there. (As for non-dual-boot machines, well, sigh. But I'd better not have an external drive connected during boot time.)

Raymond Chen
14 Apr 2004 5:15 PM
#
Because for Windows 98, LO18C9~1.TXT is a corrupted short file name. Win98 would have use LONGNA~1.TXT.

Mike Dunn
14 Apr 2004 5:23 PM
#
Check out http://support.microsoft.com/?kbid=164351 for how the command prompt handles extensions of more than 3 characters. (The article says it applies to NT 4, so I'm not sure about 2000/XP, but it's worth reading anyway.)

Norman Diamond
14 Apr 2004 8:46 PM
#
4/14/2004 5:15 PM Raymond Chen:

> Because for Windows 98, LO18C9~1.TXT is a
> corrupted short file name.

I asked why Windows 98 thinks that.

> Win98 would have use LONGNA~1.TXT.

Even if, as you said in the base note,
> or possibly "LO18C9~1.TXT" if there are a
> lot of collisions.
? Surely collisions were a known situation while Windows 98 was being developed.

Raymond Chen
14 Apr 2004 9:09 PM
#
Windows 98 just uses larger and larger numbers (e.g., LO~18512.TXT). It never uses LOxxxx~1.TXT notation.

You don't need Novell to get this...
14 Apr 2004 9:46 PM
#

Actually each filesystem is free to choose (or not choose!) a short file name with whatever algorithm it likes. The only criteria is that it's a unique name.

All the Microsoft filesystems that ship in the box happen to use a common code library often called "fsrtl" which gives common behavior.

To some degree, the code reuse is problematic because people end up not seeing the potential variances frequently enough.

Also note that on FAT the way that filename comparisons are done is different than on NTFS (since the short filenames are actually in the OEM code page) so you might even get wackier unexpected matches.

Anonymous Coward
14 Apr 2004 10:38 PM
#

>Wildcards match both the long
>and short name. The short name
>of "foo.html" is (say)
>"foo~1.htm" which matches *.htm

Yes, I understood why it is happening (after having lost several hours work). I also happened to write an SMB/CIFS server about a decade ago. It took no less than 4 rewrites of the functions to list a directory. The Samba folks got so frustrated they actually wrote a tool that sends requests to multiple servers and then compares the results (the servers would be NT4, 95, 98, 2K etc).

It is just *VERY* annoying that a workaround for a filesystems from 20 years ago deletes my files now that are not stored on that filesystem.

I also can't see any wildcard that does match only the htm files. For example:

del "*.htm "

(Note space on end). That still lists htm and html.
Even the non-obvious (to people who haven't written an SMB server :-) pattern:

del ?????????????.htm

That doesn't work despite there being more wildcard characters than in a short name.

Since our server ran on Unix, we couldn't store a short name along with a long name. We used ~ followed by base36 [a-z0-9] encoded checksum of the full name, with administrator selectable number of digits.

I think Samba in the end resorted to a database. This is the only possibility if directories can grow to contain a large number of files.

Raymond Chen
14 Apr 2004 10:49 PM
#

The whole wildcard story is made triply complicated by all the backward compatibility requirements. For example "*.*" matches all files, even ones that don't have a dot in their names. Why? Because everybody expects "*.*" to match all files.

Jonathan Payne
15 Apr 2004 12:39 AM
#
> Unclear what you mean "put the HTML and associated files....

I was trying to suggest a system where IE put all the files in a folder and then changed the open action of the folder to display the HTML page in that folder rather than showing the folder contents. I realize that this would be a slight change to how Explorer works but it would be nowhere near as bad as implicitly linking files and folders.

I am still not clear why Windows goes to the effort of warning me if I try and rename one of these linked files but will delete data without asking.

If Microsoft really wants to break the whole selection metaphor (where context menus and keyboard UI work on the selected file), they could at least go to the effort of fixing the menu items and status bar text to reflect the change (so for the "Delete" menu item the status bar could read "Delete selected item and other implicitly linked items that you have no way of knowing about" instead of the misleading "Deletes the selected item").

CC
15 Apr 2004 12:42 AM
#

On this subject, is it likely that the short name hackery will be removed in future versions of Windows? Or are we stuck with it forever as a 'backward compatability' issue (I wonder what the oldest code is in Windows that is there solely for compat reasons?).

Petr Kadlec
15 Apr 2004 3:52 AM
#
As Marc Wallace has reminded, it is not true that "Windows 95 uses the "~n" method exclusively" (if you create the key in the registry, W9x do not use ~1 if the truncated filename is unique, although I'll make a small correction: when you create filename2.txt after filename1.txt is already there, the name does not get shortened to filenamf.txt, but to filena~1.txt). That setting is great and I think that it would be better if such behaviour would have been default. (I use that for a long time.)
But, I have to say that this option is probably just a hack that has not been much tested. If you create "a file with a long name.txt", you get afilewit.txt, which is great. But if you then try to copy a file named afilewit.txt into that directory, you will overwrite the file with the shortened name... WHY? (Because of that, I often have problems when unpacking a zip file that contains e.g. inttypes.m4 and inttypes-priv.m4, because the behaviour depends on the order in which the files are stored in the zip file.)

Raymond Chen
15 Apr 2004 7:16 AM
#
Why does it overwrite afilewit.txt? Because yo told it to! You said "take this file called

alfilewit.txt and copy it to that place, overwriting any existing file with the same name" and it so happens that there is one, namely afilewit.txt itself.

CC
15 Apr 2004 7:33 AM
#

Raymond, no offnse intended but does that not strike you as kind of lame?

Mat Hall
15 Apr 2004 7:44 AM
#
But the first file is *not* called afilewit.txt -- that's just a truncated version of its name.

Quite frankly, the whole 8.3 thing causes no end of problems; why they bothered with this shortening business I'll never know. If you insist on using apps that don't work with long filenames, don't give your files long names!

Petr Kadlec
15 Apr 2004 7:45 AM
#
Yes, you are almost right. But it is obviously not desirable. I haven't told it to "create file.txt, copy another file to file.txt". I told it to "create some_long_name.txt, copy another file to file.txt". The fact that meanwhile, some other file.txt (which is in fact some sort of internal representation of "some_long_name.txt") has appeared, should not be a thing I have to know and watch for.
(AFAIK the same behaviour happens with the ~n convention, the only difference in the far smaller probability of a normal file named hombre~1.txt)
Well, I presume this problem was one of the reasons the ~1 convention was chosen in favor of the more logical plain name shortening.

Petr Kadlec
15 Apr 2004 7:46 AM
#
My reply was obviously meant to Raymond, not Mat, who was just a little bit faster. :)

Raymond Chen
15 Apr 2004 8:03 AM
#
The problem is that every file has two names, a long and short (ignoring disabled short names), and both are the name of the file. If somebody says "Create a file called alongnam.txt, overwriting any existing one" then lo and behold there is one. It may not be the one you intended, but the OS isn't good at reading minds. Maybe you aren't LFN-enabled and all you see are short file names.

Yes the behavior is pretty lame, but the alternative is breaking compatibility with programs that do not support LFNs.

So why have short names at all? HPFS tried that: It didn't have short names. If a file had a long name, then SFN apps could not access it. This turned out to have been an obstacle to adoption, since almost no programs were LFN-enabled at the time. Your backup program couldn't back up LFN files, for example.

Even worse, a file management progam might try to delete all the files in a directory and then try to delete the directory, but it would fail because there were LFNs in the directory that it couldn't see. This drove me so batty that I simply stopped using LFNs on my OS/2 machine altogether.

CC
15 Apr 2004 8:11 AM
#

I guess Microsofts view on Backward Compat is another issue entirely, but I live in hope that one day we might loose some of the old cruft (sorry).

Raymond Chen
15 Apr 2004 8:20 AM
#
The problem with cruft is that NEW programs rely on old cruft. For example, this comment from Todor indicates that he's *relying on the cruft in a new program*.

http://blogs.msdn.com/oldnewthing/archive/2003/11/03/55532.aspx#72702

This makes it very hard to get rid of cruft.

CC
15 Apr 2004 8:28 AM
#

I guess it's made slightly worse by the un-guessability of some of the API. What I mean is that there is so much (many sp?) API and being fairly new to Windows I have quite a tough time looking for a call that suits my purpose (and isn't documented in petzold).

In fact most of the time I have to translate into English and search google for hints, although in some cases it IS possible to make a reasonable guess.

Peter Hartley
15 Apr 2004 2:31 PM
#
You get a second helping of all this woe if you try and read Unicode filenames from an ANSI application. A filename consisting of three Kanji characters comes back from the directory listing as "???". But if you open "???" then it successfully opens it. (It's not doing wildcard expansion -- "???" won't open a file called "abc".) If there are two Kanji filenames, directory enumeration says "???" twice, but both attempts to open "???" get the same (arbitrary) one of them. This is a Bad Thing for, say, a backup application.

While you've got to hope that no-one's still writing code that assumes filenames are representable in 8.3, plenty of people (not least the 95/98/ME crowd) are still writing code that assumes filenames are representable in 8-bit characters. People whinged about the Linux filesystem being retrospectively declared to be UTF-8, but that certainly makes things less weird for unaware programs than a heterogeneous mix of ANSI and Unicode filenames does.

Anonymous Coward
15 Apr 2004 3:41 PM
#

The wildcard situation is actually more complicated than first appears. When talking to an SMB server, *.* is converted into ????????.???. IIRC even * is converted to that pattern. I suspect that the internals of the various versions of Windows do all sorts of conversions like that before presenting the names to the local and/or remote filesystems.

That demonstrates a rule of programming. Be very careful about throwing away what the original user input you received was, especially if it is going to be flowed through several layers of libraries/abstractions.

Many times people write code that "normalises" incoming information, but in doing so throw away important parts of it. Later calls and further away abstractions can't guess what was there originally, and you end up with a fun backwards compatibility problem, combined with what many will consider arbitrary/broken behaviour.

Personally I do think UNIX got it right by leaving the applications and their libraries to deal with issues of case, encoding, wildcards etc.

Norman Diamond
15 Apr 2004 5:59 PM
#

4/15/2004 8:03 AM Raymond Chen:

> The problem is that every file has two
> names, a long and short (ignoring disabled
> short names),

The main problem under discussion here is that MOST files have two names, a long and a short (ignoring disabled short names). A different problem is that there are exceptions: some files don't have two names (even when short names are not disabled). It does not really help to misstate the main problem.

4/15/2004 2:31 PM Peter Hartley:

> You get a second helping of all this woe if
> you try and read Unicode filenames from an
> ANSI application.

But this happens all the time, in normal daily use (actually in normal minutely use if permitted to abuse English this way). So I will try to make some guesses about what you write next, but the underlying cause is not what you seem to think it is.

> A filename consisting of three Kanji
> characters comes back from the directory

> listing as "???".

Actually it comes back as the three Kanji characters. Exceptions are if you're using a non-Japanese version of Windows NT4 or ME or 98, in which case Windows Explorer displayed black rectangles but not "???". If you're using a non-Japanese version of Windows 2000 or XP or 2003 and didn't install Japanese fonts, hmm, I guess that might display "???" but I don't remember if I tried it.

> But if you open "???" then it successfully
> opens it.

OK then, you're not using Windows NT4 or 98 or ME, I guess you're using 2000 or XP or 2003 and you only neglected to install fonts.

> (It's not doing wildcard expansion -- "???"
> won't open a file called "abc".)

This is a lot harder to guess. If it were a directory name, typing "???" into Windows Explorer's address bar would fail to open the directory but would say that your site is undisplayable due to a DNS failure. But you didn't say directory, you said file. Anyway, I don't think Windows Explorer expands wildcards in its address bar.

So, are you using a command prompt instead? Did a command-line "DIR" command display question marks? If so, then we're probably back to the situation of not having installed or not having selected a suitable font for your command prompt window. But if you type question marks into a command prompt, then they sure should get expanded as wildcards. "???" sure should match a file called "abc".

So you're not talking about Windows Explorer and you're not talking about a command prompt, so what application are you talking about?

> If there are two Kanji filenames, directory
> enumeration says "???" twice, but both
> attempts to open "???" get the same
> (arbitrary) one of them.

I haven't seen this problem with Kanji filenames, nor with non-displayable filenames, but I've seen it with CD-ROMs. If a CD-ROM folder contains two perfectly valid files with two perfectly valid filenames, then an attempt to open the second one might get the first one instead. For example:
ABCDEFGH.txt
ABCDEFGHI.txt
Trying to open ABCDEFGHI.txt got ABCDEFGH.txt instead.

> This is a Bad Thing for, say, a backup
> application.

Yup.

Raymond Chen
15 Apr 2004 6:13 PM
#
So the conclusion is that basically nothing you can make non-LFN-enabled backup programs completely happy. The issue then is which makes them least unhappy.

josh
15 Apr 2004 9:04 PM
#
AC: Unix does the wrong thing with wildcards. Try "echo *" on both Windows/DOS and Unix. Windows gives you * back, Unix gives you a list of files. This is particularly annoying if hundreds of files match your pattern.

Of course it depends on your shell for Unix, but if it doesn't do this none of the regular utilities will work. :6

Anonymous Coward
15 Apr 2004 10:18 PM
#
Both Unix and Windows shells have rules about evaluation. On Unix shells, the rules are clear and consistent in that things outside any form of quoting are evaluated, inside double quotes environment variables are evaluated and inside single quotes nothing is evaluated. For your example, echo "*" or echo '*' will do the trick.

The Windows shell is a mess. Here is an example:

```
C:\>echo %a%
%a%

C:\>set a=foo

C:\>echo %a%
foo

C:\>echo %%a%%
%foo%

C:\>echo "%a%"
"foo"

C:\>echo "%%a%%"
"%foo%"

C:\>echo '%a%'
'foo'
```

I have no idea how to make it output the literal string %a%

Norman Diamond
15 Apr 2004 11:34 PM
#
4/15/2004 6:13 PM Raymond Chen:

> So the conclusion is that basically nothing
> you can make non-LFN-enabled backup programs
> completely happy.

Well, if Windows Explorer is an example of a non-LFN-enabled backup program then you're right.

(If the meaning is unclear, then delete the word "backup". If Windows Explorer is an example of a non-LFN-enabled program then you're still right. It isn't necessary to argue whether Windows Explorer is reasonable to use for making backups or not, even though that's what I use for copying files to external hard drives when it will let me.)

Anyway Mr. Hartley's problem remains unclear to me, and I stated why. Mr. Chen, you don't have to worry about that. I notice you didn't comment on the correction to your statement that "every" file has two names. By the way when I got hit by that, the application wasn't a backup program.

josh
16 Apr 2004 2:09 PM
#
echo ^%a^%

But that's just a matter of taste. How do you get it to echo 'foo' on unix? echo \'$a\' (granted, \ is a more uniform choice for an escape char, but not really an option under Windows)

The wildcard thing is really bad, what if you WANT to have it match hundreds of files? You'd better hope the shell can handle really long lines, and that the program doesn't interpret file names differently based on their position in the argument list.

Petr Kadlec
17 Apr 2004 11:13 AM
#
There is always some "what if you want...". In Unix, the rules are very clear and consistent. If you want some special behaviour, noone prevents you from using either some custom wildcards, or quoting the params.
Note that even the order in which the files are listed in the * expansion is defined (ASCIIbetical), so that a UNIX-enabled program which has problems because of shell expansion is probably written by someone who just ignores known facts.
But as you have said, it's just a matter of taste -- both ways have their pros and cons...

Anonymous Coward
17 Apr 2004 11:34 AM
#
It is also normal for Windows programs to not be able cope with multiple files specified on the command line. For example "notepad file1.txt file2.txt" doesn't open notepad on two files. It instead tries to open one file named "file1.txt file2.txt".

There is exactly one place I like the Windows wilcard convention. Try to do this on Unix:
rename *.log *.txt

Surprisingly, even "ren *.htm *.html" works correctly, as does the reverse.

Petr Kadlec
17 Apr 2004 1:45 PM
#
Well, I am afraid your example is not the best one. :-) In fact, this deficiency in Unix-style expansions is so obvious that there _is_ a rename tool on Linux (whereas a single-file renaming is done using the mv command), which is able to do all those tricks. (Like "rename .htm .html *.htm".)

josh
17 Apr 2004 7:35 PM
#
I think my point was that Unix does *not* pass input on verbatim to the program. The hacks to get around this have been slightly more uniform than those to get around DOS weirdness, but it was still not the best choice.

Multiple files is a matter for the application rather than the shell, but you can easily (erm... or not) fix that deficiency in the shell: "for %i in (file1.txt file2.txt) do notepad %i" Notepad's an SDI app anyway...

Florian
19 Apr 2004 11:51 AM
#
Well, Unix shells *do* pass input on verbatim to the program if the user does desire this. But the you have to rely on the program to know how to handle * as it's input. Same is under DOS, just that you will have to follow the clear rules of the shell to call >foo '*'< instead of >foo *<.

The ren *.log *.txt example is a nice one where under Unix you would need an extra tool like mmv in order to do that (i.e. if you don't want to type something like: bash$ for file in *.log ; do mv $file ${file/%.log/.txt} ; done). But that is not so different from DOS, either, as the "ren" command also has to have the logic to interpret parameters "*.log *.txt" built in, doesn't it?

Anonymous Coward
19 Apr 2004 11:54 AM
#
I wouldn't exactly call the rename command intuitive (and I have been using Unix since 1989). Also note that it replaces the *first* occurrence of the pattern mentioned. That would cause grief if files had double extensions, or if the final wildcard didn't match the first argument (eg if it was index.* instead of *.htm).

As for notepad being SDI etc, it does quite happily accept multiple files via drag and drop (but only displays the first). Arguably the command line behaviour should be the same, although I susupect it is a the way it is to cater for people who didn't bother to quote the command line arguments.

I also found this KB article about being able to force "correct" matching for long extensions. I believe it is implemented by putting a tilde in the extension. The article says it is for NT4. I haven't tried it on XP.

http://support.microsoft.com/default.aspx?scid=kb;en-us;164351

And another note on renames, the SMB protocol supports a rename request, including wildcards. We had a really hard time trying to figure out the correct semantics. Think of things like *foo* to *foo, ????foo? to *.foo, *.foo to bar.* etc (and don't forget to mix in long and short names).

Petr Kadlec
20 Apr 2004 6:29 AM
#
Yeah, non-trivial renaming is probably done using a special program in some programming language. :-)
Actually, the best tool for renaming I use is the rename function of Total Commander. Well, you can't use REs (yet), but its capabilities like "[N3-4][N1-5]" are well enough for most of my needs...

Diego
14 May 2004 12:35 PM
#
Can I expect ALL short names to be EXACTLY 8 chars long (with an optional extension of 3 chars)?

Raymond Chen
14 May 2004 12:46 PM
#
Nope. No such guarantee. And I can even name a scenario where it happens, but I'm not going to say what it is because my point is that you can't make any assumptions, not to list specifically all the exceptions. (Otherwise people will be tempted to make assumptions and just check for the exceptions.)

Raymond Chen
8 Jul 2004 1:07 AM
#
Commenting closes after two weeks.