Twitter Thread by foone

foone
@Foone

It is 2018 and this error message is a mistake from 1974.
This limitation, which is still found in the very latest Windows 10, dates back to BEFORE STAR WARS. This bug is as old as Watergate.

When this was developed, nothing had UPC codes yet because they'd just been invented.
Back when this mistake was made, There was only one Phone Company, because they hadn't been broken up yet. Ted Bundy was still on the loose. Babe Ruth's home run record was about to fall.

When this bug was developed, Wheel of Fortune hadn't yet aired. No one had seen Rocky Horror. Steven Spielberg was still a little-known directory of TV films and one box-office disappointment. SNL hadn't aired yet. The Edmund Fitzgerald was still hauling iron ore.

WHEN THIS STUPID MISFEATURE WAS INVENTED, THE GODFATHER PART II HAD JUST OPENED IN THEATERS.

So, why does this happen? So Unix (which was only 5 years old at this point) had the good idea of "everything is a file" which mean you could do things like write to sockets, pipes, the console, etc with the same commands and instructions.

This idea was brought into CP/M by Gary Kiddal in 1974.
You could do neat things with it like copy data off the serial port into a text file, or print a textfile right from the command line!

This is done in unix by having special files existing in special folders, like /dev/tty for the console or /dev/lp0 for the first printer.
You can get infinite zeros from /dev/zero, random bytes from /dev/random, etc!

but here's the problem: CP/M is designed for 8-bit computers with very little memory, and no hard drives. At best you've got an 8" floppy drive.
So directories? you don't need 'em. Instead of directories, you just use different disks.

but without directories you can't put all your special files over in a /dev/ directory.
So they're just "everywhere", effectively.
So if you have FOO.TXT and need to print it, you can do "PIP LST:=FOO.TXT" which copies foo.txt to the "file" LST, which

is the printer.

and it works where ever you are, because there are no directories! it's simple.

but what about extensions? Here's the problem: programs like to name their files with the right extension.

so if you're running a program and it goes "ENTER FILENAME TO SAVE LISTING TO" you could tell it LST to print it or PTP to punch it out to tape (cause it's 1974, remember?)
but the program might try to put .TXT on the end of your filename! LST.TXT isn't the printer, right?

Nah. It is. These special devices exist at all extensions, so that this works. so if "CON" is reserved to refer to the keyboard, so is CON.TXT and CON.WAT and CON.BUG

Eh. It's a hack, but it works, and this is just on some little microcomputers with 4k of ram, who cares?

Well CP/M caught on widely through the late 70s and early-80s. It was one of the main operating systems for business use. It defined an interface which meant you could write CP/M code on a NorthStar Horizon and run it on a Seequa Chameleon.

The lack of a portable graphics standard kept it out of the games market for the most part (though there are Infocom releases) so it was mainly business users.
But it was big, so naturally IBM wanted it for some "PC" project they were doing in early 1980

So IBM intended to launch the IBM PC with several operating systems, and were expecting CP/M to be the "main" one. But CP/M for the x86 didn't come out until 6 months after the IBM PC launched... and it cost 240$ vs 40$ for DOS.

so the vast majority of users ended up using Microsoft's PC-DOS, which was an evolution of a new OS developed by Seattle Computer Products.
MS purchased Tim Paterson's project and developed it into PC-DOS (which later became MS-DOS, if you're not aware)

Tim Paterson's OS was called "QDOS", for "Quick and Dirty Operating System". It was basically written because CP/M didn't have an x86 version yet, and an attempt to solve some of the limitations of CP/M.
It was definitely inspired by CP/M, in a lot of ways.

One of those main ways was keeping the idea of special files and no directories, because that was a useful feature of CP/M. So QDOS and PC-DOS 1.0 have AUX, PRN, CON, LPT, etc, too!

For PC-DOS 2.0 released in 1983 for the new IBM XT, Microsoft significantly revamped PC-DOS. The IBM XT featured a hard drive, so PC-DOS needed directories support.
You need them to keep your massive 10mb hard drive organized, obviously!

But here's the problem: Users have been using these special files since PC DOS 1.0 release two years earlier. Software has been written that uses them! batch files have been written that support them.
with directories, Microsoft could now make a C:\DEV folder... but they didn't.

For what wouldn't be the last time, Microsoft sacrificed sanity for backwards compatibility:
Special files are in EVERY DIRECTORY with EVERY EXTENSION.
So your "DIR > LPT" trick to print the directory listing doesn't break because you're in C:\DOS instead of A:\

But we're not running DOS 2.0, of course...
And when Windows 95 was released, it was built on top of DOS. So it naturally inherited this behavior. (Windows 1/2/3 similarly did, but Win95 was much more an OS than they were)

But hey, we're not running Windows 95 anymore! The current branch of windows is based on Windows NT, not Win95.

But Windows NT wanted compatibility with DOS/Windows programs. And XP merged the two lines.
So these special files still work, FORTY FOUR FUCKING YEARS LATER

Feel free to try it yourself! Open explorer, do "new text file", and name it con.txt
aux.txt
prn.txt

it'll tell you NOPE

So because of Gary Kiddal going "Special files representing hardware devices! That's a neat idea, Unix. I'll borrow that idea and try to hack it into my toy-computer OS" so long ago that people born that year can have children old enough to drink... we can't name con.txt

Microsoft gives the official list here:
CON, PRN, AUX, NUL, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, and LPT9

https://t.co/1eYQVBKO65

For extra fun, accessing C:\con\con (or C:\aux\aux) on win95 would cause it to bluescreen instantly. That was hilarious back in 1995, because it was a 21-year-old bug! Imagine some misdesign hanging on that long?

Bonus: Here's a picture of Tim Paterson at the VCF:W this August, giving a talk on the history of DOS.

And if you want the backstory for I got into this mess where I have a file I can't copy:
These special-device names are implemented at the OS level, rather than the filesystem level. So they're perfectly valid NTFS filenames, and I was using an NTFS drive in linux.

And apparently OS/2 didn't implement these special names either, cause IBM shipped some opengl headers as AUX.H on one of the OS/2 devcon disks.

So today I was trying to backup this NTFS drive onto my main PC and WHOOPS CAN'T COPY ALL FILES CAUSE OF BUGS OLDER THAN MOST PEOPLE READING THIS

OK sorry I was stuck in hospitals and/or asleep since I wrote this.
A couple follow ups:
1. The CP/M inventor's name is "Gary Kildall", not "Gary Kiddal".
Sorry, I posted this at 5am after being in the hospital for like... 8 hours?

2. CP/M actually didn't do these special names as simply as I described them, which is a fact I either never learned or had since forgot.
It actually required them to be followed by a colon, as if they were a drive name.
So PRN: is the printer, PRN is not.

3. CP/M didn't implement these at the OS layer like DOS did! They were just included in PIP, the file copy command. So you couldn't do the DOS trick of telling a program to save to PRN.TXT to print it.
I didn't mean to imply CP/M did, just DOS, but I don't think I made this clear

4. PC DOS 1 didn't actually have redirection or pipes, so you couldn't do the redirects I suggested. I forgot that. They were added in PC DOS 2.0 in 1983.
PC DOS 1 did support copying to/from special files though, so my general point was correct, even if my example was confusing

In any case, thanks for the response this thread got! I wasn't expecting it to blow up as big as it did, this was really just me being way too tired and rambly and being amazed at getting home to an error message caused by a decision from 44 years ago.

and if it wasn't clear, my intent was never to be like "WINDOWS SUCKS" with this. Backwards compatibility is, in general, a good thing. In fact, I'd like MORE backwards compatibility, not less.

But I was just flabbergasted at hitting a bug from 44 years ago, while running Windows 10 and copying from a USB 3.0 SSD to another SSD.

It's like you're living on a space station and get trampled by a horse.

tl;dr: