Smart quotes: The hidden scourge of text meant for computer consumption

devblogs.microsoft.com/oldnewthing/20090225-00

February 25, 2009

Raymond Chen

Smart quotes—you know, those fancy quotation marks that curl “like this” ‘and this’ instead of standing up straight “like this” ‘and this’—are great for text meant for humans to read. Like serifs and other typographic details, they act as subtle cues that aid in reading.

But don’t let a compiler or interpreter see them.

In most programming languages, quotation marks have very specific meanings. They might be used to enclose the text of a string, they might be used to introduce a comment, they might even be a scope resolution operator. But in all cases, the language specification indicates that the role is played by the quotation mark U+0022 or apostrophe U+0027. From the language’s point of view, the visually similar characters U+2018, U+2019, and U+02BC (among others) are completely unrelated.

I see this often on Web sites, where somebody decided to “edit” the content to make it “look better” by curlifying the quotation marks, turning what used to be working code into a big pile of syntax errors.

I even see it in email. Somebody encounters a crash in a component under development and connects a debugger and sends mail to the component team describing the problem and including the information on how to connect to the debugger like this:

> WinDbg –remote npipe:server=abc,pipe=def

Or maybe like this:

> Remote.exe “abc” “def”

And you, as a member of the team responsible for that component copy the text out of the email (to ensure there are no transcription errors) and paste it into a command line.

```
C:\> Remote.exe “abc” “def”
```

and you get the error

```
Unable to connect to server ôabcö
```

What happened? You got screwed over by smart quotes. The person who sent the email had smart quotes turned on in their email editor, and it turned “abc” into “abc”. You then got lulled into a false sense of security by the best fit behavior of `WideCharToMultiByte`, which says *I can’t represent “ and ” in the console code page, but I can map them to " which is a close visual approximation, so I’ll use that instead*. As a result, the value you see on the command line shows straight quotes, but that’s just a façade behind which the ugly smart quotes are lurking.

I’ve even seen people hoist by their own smartly-quoted petard.

> I can’t seem to access a file called `aaa bbb.txt`. The command
>
> type “aaa bbb.txt”
>
> results in the strange error message
>
> The system cannot find the file specified.
> Error occurred while processing: “a.
> The system cannot find the file specified.
> Error occurred while processing: x.txt”.
>
> Why can’t I access this file?

Somehow they managed to type smart quotes *into their own command line*.

So watch out for those smart quotes. When you’re sending email containing code or command lines, make sure your editor didn’t “make it pretty” and in the process destroy it.

Exercise: What is wrong with the WinDbg command line above?

Bonus chatter: PowerShell is a notable exception to this principle, for it treats all flavors of smart quotes and smart dashes as if they were dumb quotes and dumb dashes. From what I can tell, you have this guy to thank (or blame).

Raymond Chen

Follow