Meet me here on Savvyday 29 Oatmeal 94

devblogs.microsoft.com/oldnewthing/20200304-00

March 4, 2020

Raymond Chen

The `InternetTimeToSystemTime` function takes an HTTP time/date string and converts it to a windows `SYSTEMTIME` structure. A customer noticed that the `InternetTimeToSystemTime` returns strange results when given strange data.

Input	Result	Notes
`Sat, 29 Oct 1994 09:43:31 GMT`	October 29, 1994 at 09:43:31 GMT	As expected
`Sat 29 Oct 1994 09:43:31 GMT`	October 29, 1994 at 09:43:31 GMT	Missing comma
`Sat 29 Oct 1994 9:43:31 GMT`	October 29, 1994 at 09:43:31 GMT	Missing leading zero
`Sat Oct 29 9:43:31 1994`	October 29, 1994 at 09:43:31 GMT	Flipped month/day and trailing year
`Sat 29 Oct 1994 9:43`	October 29, 1994 at 09:43:00 GMT	Missing seconds and time zone
`Sat 29 Oct 1994`	October 29, 1994 at 00:00:00 GMT	Missing time
`Sat 29 Oct 94`	October 29, 1994 at 00:00:00 GMT	Two-digit year
`Savvyday 29 Oatmeal 94`	October 29, 1994 at 00:00:00 GMT	Horrible spelling errors
`1`	March 4, 2020 at 15:00:00 GMT	Returns current time

What’s going on?

The `InternetTimeToSystemTime` function tries really hard to make sense out of what you give it. This sometimes leads to the absurd, like treating *Savvyday* as if it were a misspelling of *Saturday* and *Oatmeal* as if it were a misspelling of *October*.

The `InternetTimeToSystemTime` is not a validator. It's a best-guess parser. The expectation is that you are giving `InternetTimeToSystemTime` a string that you got from an HTTP header, and you need to make as much sense out of it as you can, per Postel's Principle.[1]

Back in Windows 7, the feature team tried to make the parser more strict. This effort was a total disaster, because applications in practice were using the function to parse all sorts of things that didn't even pretend to adhere to the RFC. For example, a photo processing shell extension used this function to parse dates, and the attempt to enforce strict validation caused the shell extension to stop working entirely.

Consequently, all the changes were backed out, and the parser to this day remains as tolerant of malformed dates as it was when it was originally written. The generous parsing is now a required feature.

[1] There are those who believe that Postel's Principle is wrong.

Raymond Chen

Follow