Does the windows FILETIME structure include leap seconds?

Asked 16 years, 3 months ago Modified 5 years, 7 months ago

Viewed 6k times

20

The `FILETIME` structure counts from January 1 1601 (presumably the start of that day) according to the Microsoft documentation, but does this include leap seconds?

windows datetime filetime leap-second

Share

Improve this question

Follow

edited Oct 30, 2018 at 20:17

Matt Johnson-Pint
241k ● 75 ● 462 ● 607

asked Sep 24, 2008 at 23:24

Lawrence D'Anna
3,114 ● 2 ● 23 ● 27

Leap Seconds in Windows. Anyway Windows 10 will support leap seconds – phuclv Aug 3, 2018 at 2:05

@phuclv - Thanks, I added an answer below that details these changes. – Matt Johnson-Pint Oct 26, 2018 at 20:28

7 Answers

Sorted by: Highest score (default)

18

The question shouldn't be if `FILETIME` includes leap seconds.

It should be:

> Do the people, functions, and libraries, who interpret a `FILETIME` (i.e. `FileTimeToSystemTime`) include leap seconds when counting the duration?

The simple answer is *"no"*. `FileTimeToSystemTime` returns seconds as `0..59`.

The simpler answer is: "*of course not, how could it?*".

My Windows 2000 machine doesn't know that there were 2 leap seconds added in the decade since it was released. Any interpretation it makes of a `FILETIME` is wrong.

Finally, rather than relying on logic, we can determine by direct experimental observation, the answer to the posters question:

```
var
    systemTime: TSystemTime;
    fileTime: TFileTime;
begin
```

```
    //Construct a system-time for the 12/31/2008 11:59
    ZeroMemory(@systemTime, SizeOf(systemTime));
    systemtime.wYear := 2008;
    systemTime.wMonth := 12;
    systemTime.wDay := 31;
    systemTime.wHour := 23;
    systemtime.wMinute := 59;
    systemtime.wSecond := 59;

    //Convert it to a file time
    SystemTimeToFileTime(systemTime, {var}fileTime);

    //There was a leap second 12/31/2008 11:59:60 pm
    //Add one second to our filetime to reach the leap
    filetime.dwLowDateTime := fileTime.dwLowDateTime+1
100ns = 1s

    //Convert the filetime, sitting on a leap second,
time
    FileTimeToSystemTime(fileTime, {var}systemTime);

    //And now print the system time
    ShowMessage(DateTimeToStr(SystemTimeToDateTime(sys
```

Adding one second to

```
12/31/2008 11:59:59pm
```

gives

```
1/1/2009 12:00:00am
```

rather than

```
1/1/2009 11:59:60pm
```

Q.E.D.

Original poster might not like it, but god intentionally rigged it so that a year is not evenly divisible by a day. He did it just to screw up programmers.

Share Improve this answer Follow

edited Mar 9, 2016 at 10:49
phuclv
41.6k ● 15 ● 177 ● 519

answered Aug 30, 2010 at 18:10
Ian Boyd
256k ● 264 ● 907 ● 1.3k

1 Your windows 2000 server can *know* the leap seconds added since it was released if it consults an NTP server and keeps track of the LI field in their response. – Pacerier Jun 16, 2013 at 19:48

1 @Pacerier Unfortunately that only tells the OS if a leap second will be inserted at the end of the day. Unfortunately NTP server's don't report a list of all leap-seconds that have happened retroactively since NTP was invented; or all leap seconds i missed before i installed my OS, or any days my server was off. – Ian Boyd Jun 17, 2013 at 19:05

I wasn't saying that NTP reports a list of all leap seconds that happened. Your windows 2000 server can know the leap seconds added since it was released if **it** consults an NTP server **and** keeps track of the LI field in their response. The OS reads the field provided by the NTP server, and it does the *tracking*, perhaps by saving it to the system etc. Apparently, the OS can do much much more, examples include consulting an online web service from Microsoft every once in a while to ensure its records aren't tempered... – Pacerier Jun 18, 2013 at 20:00

...by malicious NTP servers, or simply to do a data refresh when it hasn't NTP-ed for *x* duration. – Pacerier Jun 18, 2013 at 20:02

@Pacerier What happens if my computer was not on when a leap second happened? – Ian Boyd Jun 19, 2013 at 3:01

9

There can be no single answer to this question without first deciding: What is the Windows FILETIME actually counting? The Microsoft docs say it counts 100 nanosecond intervals since 1601 UTC, but this is problematic.

No form of internationally coordinated time existed prior to the year 1960. The name UTC itself does not occur in any literature prior to 1964. The name UTC as an official designation did not exist until 1970. But it gets worse. The Royal Greenwich Observatory was not established until 1676, so even trying to interpret the FILETIME as GMT has no clear meaning, and it was only around then that pendulum clocks with accurate escapements began to give accuracies of 1 second.

If FILETIME is interpreted as mean solar seconds then the number of leap seconds since 1601 is zero, for UT has no leap seconds. If FILETIME is interpreted as if there had been atomic chronometers then the number of leap seconds since 1601 is about -60 (that's negative 60 leap seconds).

That is ancient history, what about the era since atomic chronometers? It is no better because national governments have not made the distinction between mean solar seconds and SI seconds. For a decade the ITU-R has been discussing abandoning leap seconds, but they have not achieved international consensus. Part of the reason for that can be seen in the [javascript on this page]() (also see the delta-T link on that page for plots of the ancient history). Because national governments have not made a clear distinction, any attempt to define the count of seconds since 1972 runs the risk of being invalid according to the laws of some jurisdiction. The delegates to ITU-R are aware of this complexity, as are the folks on the POSIX committee. Until the diplomatic issues are worked out, until national governments and international standards make a clear distinction and choice between mean solar and SI seconds, there is little hope that the computer standards can follow suit.

Share Improve this answer Follow

answered Oct 5, 2009 at 3:33

user183800

9

[Here]()'s some more info about why that particular date was chosen.

> The FILETIME structure records time in the form of 100-nanosecond intervals since January 1, 1601. Why was that date chosen?

> The Gregorian calendar operates on a 400-year cycle, and 1601 is the first year of the cycle that was active at the time Windows NT was being designed. In other words, it was chosen to make the math come out nicely.
>
> I actually have the email from Dave Cutler confirming this.

Share Improve this answer Follow

edited May 18, 2019 at 7:54
Abel Cheung
496 ● 5 ● 15

answered Aug 9, 2009 at 0:50
Chris Smith
5,446 ● 31 ● 30

1 this doesn't answer the question: "Does the windows FILETIME structure include leap seconds?" – phuclv Oct 27, 2018 at 1:59

Where you get the value from determines whether it includes leap seconds or not; this representation of time doesn't care about leap seconds because there is no section for partial minutes to overflow. It is simply a linear count of time periods since an epoch. The concept of "leap seconds" only makes sense in the context of a more complex calendar/clock. My answer is meant to illustrate that property of the situation. – Chris Smith Oct 29, 2018 at 4:01

The answer to this question used to be no, but has changed to: **YES, sort of, sometimes**...

4

Per [the Windows Networking team blog article](#):

> Starting in Server 2019 and the Windows 10 October [2018] update time APIs will now take into account all leap seconds the Operating System is aware of when it translates FILETIME to SystemTime.

As there have been no leap seconds issued since the time of this feature being added, the operating system is still unaware of any leap seconds. However, when the next official leap second makes its way into the world, Windows computers that have this new feature enabled will keep track of it, and thus `FILETIME` values will be offset by the number of leap seconds on the computer at the time they are interpreted.

The blog post goes on to describe:

> No change is made to FILETIME. It still represents the number of 100 ns intervals since the start of the epoch. What has changed is the interpretation of that number when it is converted to SYSTEMTIME and back. Here is a list of affected APIs:
>
> - GetSystemTime
> - GetLocalTime
> - FileTimeToSystemTime

> - FileTimeToLocalTime
> - SystemTimeToFileTime
> - SetSystemTime
> - SetLocalTime
>
> Previous to this release, SYSTEMTIME had valid values for wSecond between 0 and 59. SYSTEMTIME has now been updated to allow a value of 60, provided the year, month, and day represents day in which a leap second is valid.
>
> ...
>
> In order receive the 60 second in the SYSTEMTIME structure a process must explicitly opt-in.

Note that the opt-in applies to the behavior within the functions listed on how a `FILETIME` is mapped to a `SYSTEMTIME`. Regardless of whether you opt-in or not, the operating system will still offset `FILETIME` values according to the leap seconds it is aware of.

With regard to compatibility, the article states:

> Applications that rely on 3rd party frameworks should ensure their framework's implementation on Windows is also calling into the correct APIs to calculate the correct time, or else the application will have the wrong time reported.

And also provides a links to an earlier post which describes how to disable the entire feature, as follows:

> ... you can revert to the prior operating system behavior and disable leap seconds across the board by adding the following registry key:
>
> - `HKLM:\SYSTEM\CurrentControlSet\Control\LeapSecondInformation`
> - Type: "REG_DWORD"
> - Name: Enabled
> - Value: `0` Disables the system-wide setting
> - Value: `1` Enables the system-wide setting
>
> Next, restart your system.

Share Improve this answer Follow

answered Oct 26, 2018 at 20:27

Matt Johnson-Pint
241k ● 75 ● 462 ● 607

2

Leap seconds are added unpredictably by the IERS. 23 seconds have been added since 1972, when UTC and leap seconds were defined. Wikipedia says "because the Earth's rotation rate is unpredictable in the long term, it is not possible to predict the need for them more than six months in advance."

Since you'd have to keep a history of when leap seconds were inserted, and keep updating the OS to keep a reference of when they had been inserted, and the difference is so small, it's fair not to expect a general-purpose OS to compensate for leap seconds.

In addition, regular clock drift, of the simple electronic clock in your PC compared to UTC, is so much larger than the compensation required for leap seconds. If you need the kind of precision to compensate for leap seconds, you shouldn't use the highly-inaccurate PC clock.

Share Improve this answer Follow

answered Sep 25, 2008 at 13:25
Mike Dimmick
9,792 ● 2 ● 26 ● 48

0

According to this comment windows is totally unaware of leap seconds. If you add 24 * 60 * 60 seconds to a FILETIME that represents 1:39:45 today, you get a FILETIME that represents 1:39:45 tomorrow, no matter what.

Share Improve this answer Follow

answered Sep 27, 2008 at 5:01
Lawrence D'Anna
3,114 ● 2 ● 23 ● 27

-2

A very crude summary:

UTC = (Atomic Time) + (Leap Seconds) ~~ (Mean Solar Time)

The MS documentation says, specifically, "UTC", and so should include the leap seconds. As always with MS, your mileage may vary.

Share Improve this answer Follow

answered Sep 24, 2008 at 23:42

Brent.Longborough
9,775 ● 10 ● 44 ● 62