Revised 2024-01-14T14:18:16Z

Time zones - to be or not to be?

Table of contents -

Time zones - to be or not to be?

Time zones - what are they good for? And the related topics: Are there *any* benefits in using *AM*/*PM* and the *12-hour clock*? Why do we still have several different *date formats*? Can you tell me what time it is?

This text contains my own thorough and comprehensive compilation of the pros and cons (mostly cons to be honest) of our *current timekeeping system*. **The text is being aimed at everyone wanting to have a better understanding of time zones, daylight saving time, date and time formats**. Reading this will give you valuable information and enhance your knowledge about some difficulties with timekeeping, regardless if you are a beginner or an experienced time nerd.

If you have any questions about these topics, email me, and I will try to answer as best I can.

[Please, take me directly to TL;DR, or at least to the summary section with the easy step-by-step solution to all the problems, or the compilation of peculiar quirks section, or the emphasized quotes section.]

> The widespread use of time-zones tends to add considerable complexity to an application.
>
> Time handling in software, a source of confusion for many developers and content authors on the Web.
>
> There is no way you can know from your location alone which time zone applies at some particular point on the face of the earth: you have to ask the people who live there what they have decided.
>
> The following times all refer to the same moment: "18:30Z", "22:30+04", "1130–0700", and "15:00–03:30".

Note: The first version of this article was published in April 2017. The article is a living document, and **the content is continuously revised and updated** (last in January 2024) with new insights, clarifications and examples that I have stumbled upon during my life, when examining these rabbit holes. I recommend you to read everything at least once and thereafter you can have the article as your encyclopedia (or can of worms). This text contains everything you did not know that you wanted to know about time zones. You can rest assured that the vast majority reading this article will understand way more about time handling than before.

Note: [Spoiler alert] You will find out that my answers to the four questions in the introduction above are: "Almost nothing", "No", "I really don't know" and "Sometimes, but *far* from always" respectively.

1 Background

We have all been there.

- "The event starts at 10 o'clock"
- "Call me tomorrow at 4 PM"
- "Your flight will arrive at 08:25"
- "Our international telephone support is open weekdays between 8:30 and 17:30"
- "The game is broadcasted live today at 15:30"

It seems so easy at first, but then you start wonder. *When exactly* is that?

It is easy when you always talk about the same time zone. But if the event is in another time zone? How do you even know if the event is in another time zone? Or if the one you should call is in another time zone? Which time zone is your destination airport in? Is it 08:25 in the morning or in the evening? How long will my flight take? Is their support open now?

The time values seem so simple and clear at first, but when you start thinking about them, they become really ambiguous.

Even answering a simple question like "What time is it?" is sometimes *impossible*.

> Even answering a simple question like "What time is it?" is sometimes *impossible*. ”

Note: As you will see, your *only* chance to be correct when asked "What time is it?", may be to answer "I don't know" or start your answer with "It depends...". Or, as I will explain in this text, you can always use this easy solution.

Note: If you are among those that only interact with people living in your own time zone, or if you don't even know what time zones are and that there are different wall-times around the world, you can keep on living in your bubble and skip reading.
If you belong to the increasing number of people that, in one way or another (perhaps through this thing called the Internet or by travelling), interact with people all around the globe, I think it will be well worth reading further.
You will never look at date and time values in the same way again though, and you will have a hard time to trust your watch and calendar.
So buckle up and prepare for a mind-blowing journey through time and space. While reading, please take time to contemplate, because you *will* need it.

Note: *Wall-time* is local (observed) time. This is the time the people living in a location assume is the current time for them. They (usually) have this time on their clocks on the walls.

Note: If you are saying that our current system of time zones is trivial to understand, then you do not have the full picture, since you most certainly have only scratched the surface. If you, *after reading this text*, still think that time zones are trivial, I truly will salute you.
Yes, I also think that these *basic* concepts of timekeeping should be understandable by all people that have graduated from elementary school, but that is not really the case as you will see, since many things are quite hard to grasp for the majority.
Though, I still emphasize that *all* the concepts in this text should be common knowledge and minimum requirements for everyone that claims to understand the clock and the calendar.

> *All* the concepts in this text should be common knowledge and minimum requirements for everyone that claims to understand the clock and the calendar. ”

1.1 Trying to avoid ambiguity

> A time zone is a set of rules for determining the local observed time (wall time) as it relates to incremental time (as used in most computing systems) for a particular geographical region.

When you add information about the time zone to the list above, the information become less ambiguous:

- "The event starts at 10 o'clock *CET*"
- "Call me tomorrow at 4 PM *IST*"
- "Your flight will arrive at 08:25 *UTC*"
- "Our international telephone support is open weekdays between 8:30 and 17:30 *CET*"
- "The game is broadcasted live today at 15:30 *ET*"

But, there are still problems that need to be solved:

- How do you know how far apart CET, IST, UTC and ET are from your current time zone?

- If I should call her 4 PM IST, what time is that in my current time zone and how do I know that?
- How do I even know which time zone I currently am in?
- When you say IST, is it *Israel Standard Time* or *Indian Standard Time* or *Irish Standard Time* you mean?
- If I currently am in the CEST time zone (daylight saving time of CET), and read the message about opening hours in the CET time zone, I start wondering if the company missed to change to CEST or if they still are using CET?
- Is ET before or after my time zone, i.e. will the game be broadcasted before or after 15:30 my local time?
- Perhaps the game will start to broadcast 15:30 my local time, despite the fact that I am in another time zone than ET?
- Is ET even a correct official time zone?

Note: ET is probably a lazy and confusing way of denoting either EST or EDT.

Yes, often you have a computer, mobile phone or clock that automatically can make these calculations and conversions for you. But why? Why do we have *complex* rules and algorithms just to convert between times in different parts of the world? Would it not be better if we could have a simple, common and unambiguous understanding of time?

And by the way, how do you *easily* know if your computer, mobile phone or clock has switched to the new time zone or if it still shows the time in the old time zone? How can you be sure that your devices are using the correct rules for time zones and daylight saving times? How can you *trust* the time value that you see on these devices? To summarize, *how can you **unambiguously** know if the time you see on your device is the correct time for the location on Earth where you are at the moment?*

> *How can you **unambiguously** know if the time you see on your device is the correct time for the location on Earth where you are at the moment?* ”

Note: If you do not see the problems, here are a few examples of what happens out there in the real world: first, second, third and fourth.

1.1.1 Local time

Sometimes you see the times for events announced on the Internet like "Saturday 2018-10-20 19:30 *local time*". This leaves a big question mark in your face. Now it is up to you to investigate what this "local time" may mean. You may have no information about how many hours before or after your current time zone this "local time" is. As you will learn, it is not an easy task to solve this on your own.

1.1.2 Travelling

If you fly between Stockholm and Los Angeles and go to sleep for a few hours and then wake up and you ask your neighbour what the time is. How does she know what to respond to you? Should she tell you the current time in Stockholm? Or the current time in Los Angeles? Or is it the local time where you are at the moment that you are interested in? And how can she *easily* find out the answer? If you are the pilot, you may in fact be asking about the current Zulu time.

Note: The above is an example of what I call the "trunk of a car" problem.
Imagine you are placed in a trunk. You have a watch on your arm. The car is travelling around for many hours, and you have no idea in which directions. You manage to escape. The question now is: Do you know what time it is? Is it any difference if you have a mechanical watch or a high-tech computerized watch? (Yes, what time it is, may be your smallest problem in this situation, but I think you get the message.)

> Imagine you are placed in a trunk. You have a watch on your arm. The car is travelling around for many hours, and you have no idea in which directions. You manage to escape. The question now is: Do you know what time it is? ”

A real world example from a scheduled flight. Going from Perth → Singapore → Moscow → Stockholm. Starting in Perth 17:30 and landing in Singapore 22:45. Then from Singapore 00:15 and landing in Moscow 06:20. Finally from Moscow 07:35 and landing in Stockholm 07:45. All the times are local times. With this information, it should be easy for you to answer the following questions. What is the total time of the trip? What is the flight time for each leg?

Another example is a flight from Stockholm → Chicago → Denver. From Stockholm 10:20 and landing in Chicago 12:35. Then from Chicago 14:15 and finally landing in Denver 16:00. Total time? Flight time for each leg?

Note: Oh, did I forget to give you the date of the flights. Sorry about that. The date is *mandatory* information if you should be able to get correct answers. No, I am not joking, this is absolutely true. You need the date because some of these cities use Daylight Saving Time (DST) during parts of the year. So even if you (are among the few that) know the time zones or offsets for each of the cities, you still need to know if DST is active during the flight.
If I would give you the date, would it then be an *easy* task to answer the questions? Or do you still need to use the Internet for help?

Note: Even the time between two legs may be hard to calculate. Even if you know your arrival time and your departure time on a specific airport, you can not just take the time difference between those two local time values. There *may* be a DST switch during your stay at the airport.

A flight from Amsterdam to New York. Starting at 13:25 local time and arriving at 15:25 local time. What is the flight time?

Note: Once again, it is *impossible* to give a correct answer without the date. The answer could be either of 7, 8 or 9 hours (or something completely different), depending on what the current DST rules are for Amsterdam and New York.

Note: When I talk about date in the examples above, I mean a complete date, *with year*. It is not enough to say that the flights are February 4, August 23 or December 11. You *must* also define the year. No, I am not joking now either.

Let us take an easy example instead. You fly domestic in the US from Florida to Indiana. You depart from Florida 16:00 (local time) and it is a three hours flight. When will you arrive in Indiana (local time)?

Note: In this case, even if I define the complete date, you lack information and you can not give a correct answer to the question. As you will see you need *detailed information* about which part of Florida you leave and which part of Indiana you enter.

1.1.2.1 Summary of things to consider when flying

You take a six hours flight starting at 11:40 local time. When will it land?

The answer is highly dependent on (at least) three things; (1) the *direction* of the flight, (2) the *speed* of the flight and (3) the *daylight saving time rules* (at both departure and destination).

Think about that again. You have to know direction, speed and daylight saving time rules, in order to know when you will land.

> Think about that again. You have to know direction, speed and daylight saving time rules, in order to know when you will land. ”

If you are flying straight north or south, you may land at 17:40 local time, regardless of your speed. If you fly to the west or east you must take your speed into account. Fast westbound flights may land before 11:40 local time or even the previous day. Fast eastbound flights may land the next day.

Note: Just because you fly straight north or south, it does not mean that you always will land 17:40 local time though, since time zones rules can be very strange (for example India vs China or Argentina vs Peru).

Note: If you cross either of the poles in a north/south flight, you will probably lose or gain about 12 hours in a blink.

Note: If you cross the International Date Line in either direction, your six hours flight will result in even stranger results in local time and local date, since if flying west you may now land the next day and if flying east you may land the previous day. So going in the *same* direction could in fact "travel" you back *or* forth in time, depending on where on Earth you are.

> Going in the *same* direction could in fact "travel" you back *or* forth in time, depending on where on Earth you are. ”

Note: Yes, it is speed of *flight*, not speed of *light* that I am talking about above. The speed of light, together with relativity, will certainly be a factor in a perfect timekeeping system though.

Note: Yes, in order to know the daylight saving time rules, you must of course first know the time zones at departure and destination (which is not always a trivial task to solve).

1.1.3 Communication

I am in Sweden and my wall-time shows 10:00. You are in Finland and your wall-time is showing 11:00. I am on the phone with you and I say that you should call me again at 16:00. When will you call me? When your wall-time is showing 16:00 or when mine is showing 16:00? To avoid confusion, either I have to add "my time" or "your time" or I have to instead say something like "call me in six hours" or "call me in five hours".

It is even worse, if I am in Sweden with wall-time 23:00, and you are in the United States with wall-time 15:00. I say that you should call me again tomorrow at 14:00. If the month is March or October, we may also have to worry about if any of our countries (or to be correct, any *parts* of our countries) have made a transition to/from Daylight Saving Time during the night.

What if I am in Sweden with wall-time 23:00, and you are in eastern Australia with wall-time 08:00. I say that you should call me again *tomorrow* at 09:00. When is that? I may mean "Call me again in 10 hours", which is 09:00 *tomorrow* for me. But for you, 10 hours from now is still *today*, at 18:00. Of course, we also have the "Daylight Saving Time switch during the night" issues, so my wall-time 09:00 tomorrow may actually be 9, 10 or 11 hours away.

Note: I emphasize that if your wall-time is currently 00:00, your wall-time showing 04:00, may in fact be 3, 4 or 5 hours away. It may even be 3.5 or 4.5 hours away.

Note: It is even more complex, as you will understand when reading about how Daylight Saving Time works for northern and southern hemispheres.

What if I am in Stockholm with local time 19:00 and you are in Los Angeles with local time 10:00. You should call me tomorrow at 17:00. During the night you travel to New York and I travel to Dublin. When will you call?

You manage an online meeting with participants all over the world. Your clock is showing 11:00. You tell the participants that you will take a break and continue at 13:00. Many will be confused about how long the break will be and when the meeting will continue. If you instead say that you will take a two hours break, everyone will easily understand when to continue.

1.2 Reflections

Calm down and take a moment and think about all these complexities and difficulties.

2 What exactly is a time zone?

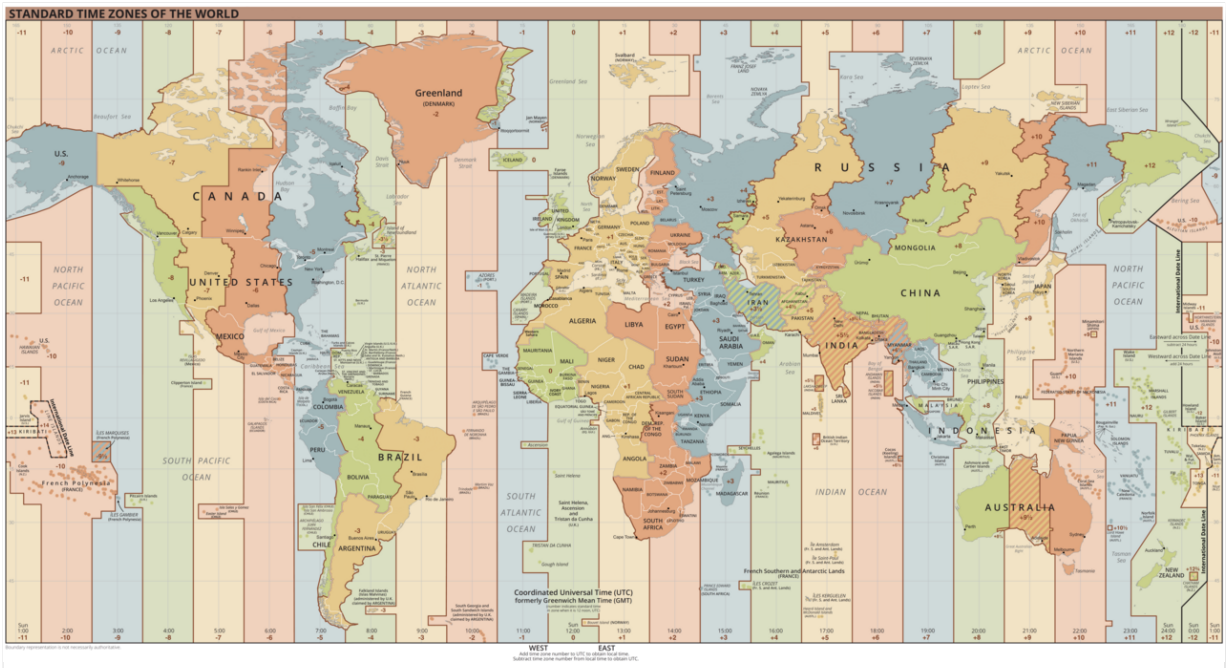
Wikipedia defines a time zone as “a region of the globe that observes a uniform standard time for legal, commercial and social purposes”. As you soon will see, one large problem is *how* to *unambiguously* identify these regions. When failing in doing this, the consequences may have important impacts.

Note: The site timeanddate.com informs that “the term time zone can be used to describe several different things”.

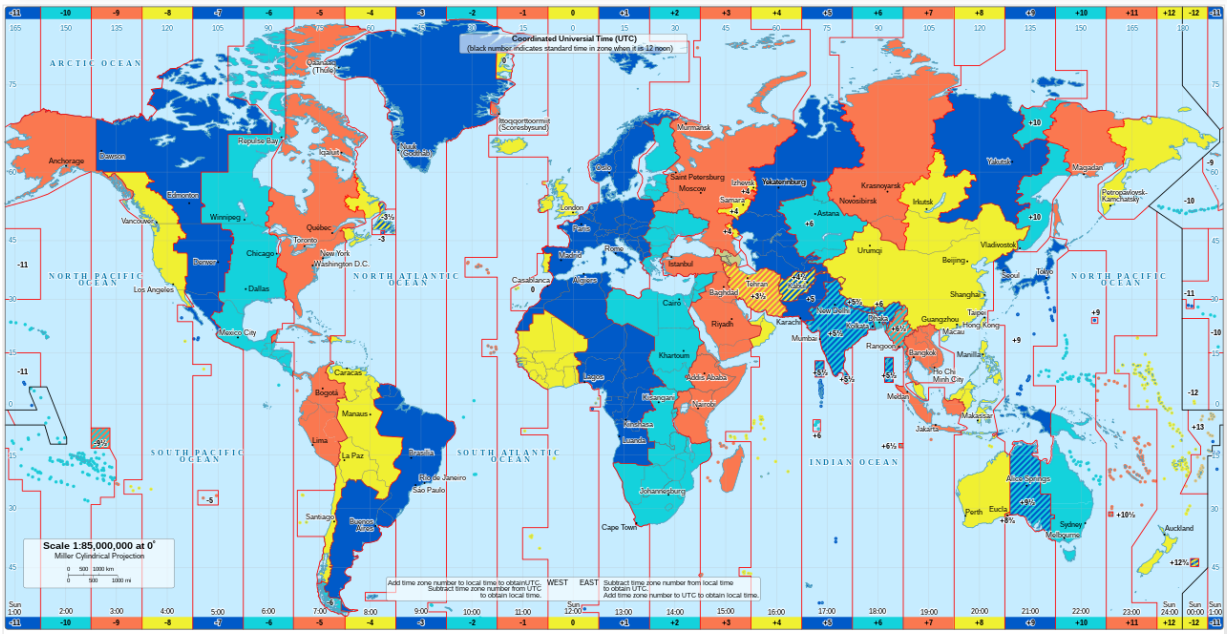
3 Abbreviations, identifiers and offsets

Why do we mix *abbreviations*, *identifiers* and *offsets* for time zones when we want to define an exact point in time? Can we easily convert between those variants?

Note: Sadly, the answer to the second question above will be "No". Sometimes we can't convert at all. From an identifier, you can convert to both of the others, but from an abbreviation or offset you can't convert at all. But beware, you must have a complete timestamp, containing the *date* information, to be able to convert from an identifier, since it is not a perfect one-to-one mapping.
Take the identifier `Europe/Stockholm` as an example. Sometimes this identifier "equals" the abbreviation *CET* which is UTC+01:00. But, sometimes the identifier "equals" *CEST* which is UTC+02:00. It depends on what time of year the time value refers to. So you *must* also know if Daylight Saving Time is active or not for the particular time value.



The time zones in the world.



Alternative view of the time zones in the world.

3.1 Time zone abbreviations

> Time zones are often represented by alphabetic abbreviations such as "EST", "WST", and "CST", but these are not part of the international time and date standard ISO 8601 and their use as sole designator for a time zone is discouraged. Such designations can be ambiguous; [...] Such designations predate both ISO 8601 and the internet era; in an earlier era, they were sufficiently unambiguous for many practical uses within a national context [...], but their ambiguity explains their deprecation in the internet era, when communications more often cannot rely on implicit geographic context to supply part of the meaning.

The list of time zone abbreviations is looooooong. More important, the list is full of duplicates, so the **abbreviations are useless**, since you have to know more than just the abbreviation. It is hard to argue against that fact.

Note: How is it even possible that these abbreviations contain duplicates? When did it all go wrong? How should *anyone* understand what `2021-01-10 15:00 CST` is supposed to mean?

Moreover, *most* people do not have a clue about what the abbreviations stand for and where on Earth they are used. Sometimes you *add* an extra character when DST (Daylight Saving Time) is used, sometimes you instead *change* a character during DST. Most abbreviations use three or four characters, but there are some with two or five characters also.

> *Most* people do not have a clue about what the abbreviations stand for and where on Earth they are used. ”

Note: For example, the standard time zone EET will be EEST during DST, but the standard time zone EST will instead be EDT during DST. You will get an extra point if you know the full names of these abbreviations and another point if you know where on the Earth they are used. Yes, you are correct, I am not a fan of abbreviations anywhere.

Note: A real advert for a livestream event on the Internet from a large tech company says: “Wednesday, May 1 | 9:30 a.m. – 11 a.m. PT”. When is this? If I search among the time zone abbreviations for PT I find nothing. There are both PST and PDT (and a bunch of other abbreviations beginning with P and ending with T), but no PT. So first I have to *guess* that PT probably stands for PST or PDT. Then I have to figure out if DST is active 2019-05-01, but then I need to know more about the actual physical location. After more searching in the advert I find that the location is San Francisco and I end up in complex pages like time offsets in US and time in US. So instead of giving an unambiguous time interval for the event, the company fails hard and leaves everything to the reader. I can not understand why sane-minded people would use abbreviated abbreviations like this.

Note: Sometimes you see a timestamp that looks like `2010-11-12 10:11:12 (Central European Standard Time)`. You may then think that the time zone part is abbreviated as CEST. But no, *Central European Standard Time* is instead abbreviated with CET (the correct name is *Central European Time*). There is an abbreviation CEST though, but it stands for *Central European Summer Time*. So by trying to be more comprehensible by adding "Standard" the time value has instead become even more ambiguous and confusing.

Note: Another example of how you can make your readers really confused, when you do not know how to write things correctly.
The first problem is that they start with “November 13, 2020 at 12 AM EST”, which in itself is confusing. Is it midnight or midday? (Yes, it is midnight.)
The second problem is, do they mean midnight between Thursday and Friday or do they mean midnight between Friday and Saturday? (Yes, it is between Thursday and Friday.)
The third problem is that in the clean(?) list, there is *at least* one item that is not on November 13, but you will have a hard time figuring that out.
The interesting part is that the map they are referring to, shows all these items in a more understandable way, showing local dates explicitly and showing the local times with a 24-hour clock.
So trying to be concise and clear in their text, results in more confusion and you have to look into their map to understand what they actually mean.
People living in the wrong time zones will also do all they can to deceive the systems.

Note: I can not understand why some people that are building brand-new time zone functionality, still are using time zone abbreviations. What are the requirements for these systems?

Note: I really emphasize that you should **avoid using these abbreviations**. The people understanding time zones will be confused when seeing them. The people not understanding time zones, don't know what they represent anyway.

3.2 Time zone identifiers

> The most definitive reference for identifying sets of time zone rules is the TZ database.
>
> In the TZ database, time zones are given IDs that typically consist of a region and exemplar city. An exemplar city is a city in the time zone in question that should be well-known to people using the time zone. For example, the U.S. Pacific time zone has a TZ database ID of "America/Los_Angeles".
>
> A time zone has the history of past rules, present rules, and rules confirmed for the near future.

The list of time zone identifiers is even longer than the abbreviations list. But it is slightly more useful, since there at least are no duplicates. Using the TZ database ~~seems to be~~ **is** extremely complex though, and you need to keep yourself updated to the *never ending* changes. And you rarely see times defined using the identifiers (for example *2018-01-27 14:30 CET* is far more common than *2018-01-27 14:30 Europe/Stockholm*).

Note: When reading about the theory, you have to be careful and distinguish between the terms “time zone” and “timezone”, since they denote two different things:

> Each timezone typically corresponds to a geographical region that is smaller than a traditional time zone, because clocks in a timezone all agree after 1970 whereas a traditional time zone merely specifies current standard time.

Note: There are no duplicates in the list of time zone identifiers, *but* it still contains some strange values. You have some aliases, which is okay, since they are probably used for defining several different cities that in fact use the same time zone (for example "America/Montreal" that is an alias for "America/Toronto" or "Arctic/Longyearbyen" as alias for "Europe/Oslo").
But some of these aliases seem to define the *same* city with different identifiers (for example "America/Indianapolis" as an alias for

"America/Indiana/Indianapolis" or "Asia/Ulan_Bator" as an alias for "Asia/Ulaanbaatar"). So when seeing two different time values, *2019-03-17 08:25 Asia/Ulan_Bator* and *2019-03-17 08:25 Asia/Ulaanbaatar*, you should interpret these as the same.
The list of time zone identifiers also contains many deprecated values.

Note: In the section “User interfaces” below, you can read about the recommended way on how to configure time zone identifiers and why end users should not have to see these cryptic identifier values.

By just using your own brain, do you know what the current time difference is between the capital cities Canberra in Australia ("Australia/Canberra") and Mexico City in Mexico ("America/Mexico_City")?

Note: No, I don't have a clue about the current time difference either. My best guess is about 12 hours and that the difference may vary by two hours depending on what time of the year it is (as you will see when reading about Daylight Saving Time craziness below).

Another thing with using time zone identifiers (and also abbreviations) is that you lay the burden on the receiver of the message to interpret what time it represents. Instead of *you* defining an unambiguous time value for everyone, you are lazy and may make many people around the globe confused.

Yet another thing is that it may be *impossible* to sort values containing time zone identifiers, as you will understand when reading details about daylight saving time later. For example, the "timestamps" `2022-10-30 02:20 Europe/Stockholm` and `2022-10-30 02:45 Europe/Stockholm` can *not* be sorted on the timeline, with just the information you see here.

Note: I put quotes around "timestamps" in the paragraph above, to emphasize that the values really aren't actual timestamps.

3.3 Offsets from UTC

Using *offsets* from UTC, for example `2017-03-25T12:34:56+01:00`, is much better than using time zone abbreviations (and perhaps even better than using time zone identifiers).

Note: I would argue that offsets are far better than both abbreviations and identifiers. If you should be able to understand a time value defined with an abbreviation or identifier, you *always* need the extra step of mapping to the correct offset, but with an offset value you already have the correct numbers. An abbreviation or identifier is really useless on its own.

But if we want to use offsets, why not go all the way and use the UTC time *everywhere* instead? Then we would not have to do calculations in our heads when comparing `2017-03-25T12:34:56+01:00` against `2017-03-25T04:34:56-07:00` in order to see that these timestamps are equal. Or for that matter comparing `2017-03-24T01:00:00+02:00` against `2017-03-23T21:00:00-02:00`. All the following timestamps actually represent the exact same point in time.

```
2017-03-25T23:34:56+12:00
2017-03-25T21:04:56+09:30
2017-03-25T17:34:56+06:00
2017-03-25T17:19:56+05:45
2017-03-25T12:34:56+01:00
2017-03-25T11:34:56+00:00
2017-03-25T08:34:56-04:00
2017-03-25T04:34:56-07:00
2017-03-25T01:34:56-10:00
2017-03-24T23:34:56-12:00
```

That was not completely true, there is one timestamp above that is not equal to the others. Did you spot it? It should be easy to see it, since this is basic time keeping knowledge.

Note: No, it is not `2017-03-24T23:34:56-12:00`, `2017-03-25T21:04:56+09:30` or `2017-03-25T17:19:56+05:45` that is unequal. It is `2017-03-25T08:34:56-04:00` that represents a time that is an hour before all of the others. With the help of tools like NNM Time zones, you can see this for yourself.

Note: I have used the *ISO 8601* extended format for most timestamps in this text, since that is the least error-prone notation the world has come up with so far.

3.3.1 Number of ideal time zones

Since the time zones around the globe should be one hour "wide", each zone should span 15 degrees longitude (15° × 24 = 360°). So you may think that there are 24 offsets spanning 15 degrees each.

But, in fact there are 25 time zones for exact one lap around the globe (UTC+1200, UTC+1100, ..., UTC+0100, UTC+0000, UTC-0100, ..., UTC-1100 and UTC-1200). So in order to get the math working, UTC+1200 and UTC-1200 span only 7.5 degrees each.

Note: 25 time zones is for one lap and with whole hours. As you will see later on in this text, there are far more than 25 time zones (roughly about 40), since they span *more* than one lap and since there (sadly) are many time zones with offsets of 30 and 45 minutes. Of course, the actual number of time zones

used is far, far more than 40 as you saw in the abbreviations and identifiers sections above.
This implies that a "timestamp" like `2019-04-18 09:15` will occur at about 40 *different* occasions.

Note: You can view a list of clocks showing these different local times in the *NNM Clock* dashboard (The wall of time).

Note: Yes, the UTC+1200 and UTC-1200 zones span actually 15 degrees each, but that is because they are "overlapping". In fact they represent the same "zone", so the local time is equal, but the dates are exactly one day apart.
This also implies that two persons in this part of the world may *agree* on the current *local time*, but at the same time they may *disagree* on the current *local date*.

3.4 Deciding correct offset from location

From your physical location on the Earth, it is sometimes impossible to know which offset or time zone you should use. Even if you know your exact longitude and latitude coordinates, there is no perfect tool that will inform you about the correct offset.

3.4.1 Straight line travelers

Some of the ideas with time zones are that (1) when you travel *east*, your local time *increases* when you cross a time zone border and (2) when you travel *west* your local time *decreases* when you cross a time zone border. When you travel *north* or *south* your local time should *stay the same*. The reason behind this is that the local time should stay in "sync" with the solar noon. But there are several examples of really strange results around the world and I will show you some of these.

Note: Yes, if daylight saving time is active in any of these countries during your travel, the result will, most certainly, be different.

Note: As you saw earlier, you have to be careful when crossing the International Date Line though. When going *east* over the line, your local time *decreases* instead. If going *west* over the line, your local time *increases* instead.

Note: There are some caveats when calculating distances from latitudes and longitudes. One degree latitude is *always* about 111 kilometers. But one degree longitude varies from 0 to about 111 kilometers, depending on where on Earth you are.

3.4.1.1 West-east travelers

In theory, when traveling west-east your local time should make a one-hour jump, every 15 degrees.

3.4.1.1.1 The 69th latitude

If you travel the line from coordinate `69.04 (lat), 20.00 (long)` to coordinate `69.04 (lat), 30.00 (long)` you are traveling a distance of 10 degrees longitude (roughly 400 km at this latitude). This is about 2/3 of an ideal time zone, so the time difference would be approximately 40 minutes between your start and end points. Your start point is located in Norway and your end point in Russia, as you can see in NNM Map Viewer.

If you write down the offsets from UTC during your travel you get interesting results. You will cross the following country borders from west to east: Norway, Sweden, Finland, Norway, Finland, Norway, Finland, Russia, Norway, Russia. The standard time zone for Norway and Sweden is UTC+1, for Finland it is UTC+2 and for *this part of* Russia it is UTC+3. Since Norway and Sweden (at the moment) use the same UTC offset, you will have the following *nine*(!) time zones during your trip: UTC+1, UTC+2, UTC+1, UTC+2, UTC+1, UTC+2, UTC+3, UTC+1, UTC+3. Sometimes the difference is one hour, but in the two last switches the difference is two hours.

3.4.1.2 North-south travelers

In theory, when traveling north-south your local time should always stay the same.

3.4.1.2.1 The 27th longitude

Note: The following may not be true after January 2021, if South Sudan change their time zone (from UTC+3 to UTC+2), which you can read about in the *Countries changing time zone* section below. I will leave this example as a reminder though.

If you travel the line from coordinate `5.00 (lat), 27.00 (long)` to coordinate `10.00 (lat), 27.00 (long)` you are traveling a distance of 5 degrees latitude (roughly 550 km). Your start point is located in Congo Democratic Republic and your end point in Sudan, as you can see in NNM Map Viewer.

You get interesting results here also. You will cross the following country borders from south to north: Congo Democratic Republic (UTC+2), Central African Republic (UTC+1), South Sudan (UTC+3) and Sudan (UTC+2). You will have the following *four* time zones during your trip: UTC+2, UTC+1, UTC+3, UTC+2.

3.4.1.2.2 The 23th longitude

Note: You have a similar result as the 27th longitude at the 23.65th longitude.

If you travel the line from coordinate `51.00 (lat), 23.65 (long)` to coordinate `54.00 (lat), 23.65 (long)` you are traveling a distance of 3 degrees latitude (roughly 300 km). Your start point is located in Poland and your end point in Lithuania, as you can see in NNM Map Viewer.

You will cross the following country borders from south to north: Poland (UTC+1), Ukraine (UTC+2 in this part), Belarus (UTC+3) and Lithuania (UTC+2).

Note: If you zoom in on the line, you will see some really interesting time zone switching, for example during a few kilometers your local time will take several two hour leaps (between UTC+1 and UTC+3). This is often the case around the world when a river meanders.

3.4.1.2.3 The 80th longitude

If you travel the line from coordinate `28.00 (lat), 80.50 (long)` to coordinate `72.00 (lat), 80.50 (long)` you are traveling a distance of 44 degrees latitude (roughly 5000 km). Your start point is located in India and your end point in Russia, as you can see in NNM Map Viewer.

You will cross the following country borders from south to north: India, Nepal, India, China, Kazakhstan, China, Kazakhstan and Russia. Since different parts of Russia use different time zones, you will at least have the following *ten* time zones during your trip: UTC+5:30, UTC+5:45, UTC+5:30, UTC+8, UTC+6, UTC+8, UTC+6, UTC+7, UTC+5, UTC+7.

3.4.1.2.4 The -157th longitude

One of the strangest of them all.

If you travel the line from coordinate `1.75 (lat), -157.20 (long)` to coordinate `21.15 (lat), -157.20 (long)` you are traveling a distance of 20 degrees latitude (roughly 2000 km). Your start point is located in Kiritimati island (Kiribati) and your end point in an Hawaiian island (United States), as you can see in NNM Map Viewer.

You will not cross so many country borders, it is most international water out here in the Pacific Ocean. The interesting thing is that you start in UTC+14 and end up in UTC-10. So the local times will be the exact same, but the local dates will be *one day apart*. And you are not even close to the (original) International Date Line.

Note: You will be reading more about Kiribati in this text.

3.4.2 Deciding offset from your country

Trying to get your offset from country alone, will *often* fail. It may work for some, often small, countries. But for many countries it is impossible to get a correct offset, since the country may use *several* time zones. Even for small countries it may be impossible, since daylight saving time may trick you.

Note: Sweden uses only one time zone, but you can not say that Sweden uses offset UTC+01:00, since half of the year the actual offset is UTC+02:00 (due to daylight saving time).
This also means that if you have a company in Sweden, you can not statically say that your opening hours are "10:00 - 18:00 CET". That will most certainly be a complete lie when the rest of Sweden is using CEST. You have to dynamically update your opening hours information during the year, to stay in sync with the daylight saving time switches.
Yes, you may live in the exact same *location* all year, but you may not live in the same *time zone* all year.

> You may live in the exact same *location* all year, but you may not live in the same *time zone* all year. ”

3.5 Local time

Leaving out the time zone marker and instead, in different creative ways, indicate that you mean *local time*, may work for local audiences. But for global audiences it is a real disaster. In these cases you have no direct understanding of what the timestamp stands for.

Note: One format I have stumbled upon is “14 March 12:00lt”. This format was used at an official page at Scandinavian Airlines to inform about when Denmark would close all borders, due to the Corona virus outbreak. So when you really needed to be *very explicit* to avoid confusion, SAS instead used a timestamp with a really ambiguous format. I do not know what “lt” stands for, it could be "local time", "lunch time", "London time", "lazy time" or whatever. If it should denote "local time", why did they not write that explicitly instead of using an obscure abbreviation? And why did they not use (the more familiar) "CET", "Europe/Copenhagen" or "UTC+01:00" in the first place?

Another confusing thing with this page is that you are not completely sure if they are using the 12-hour clock or the 24-hour clock. So the time “12:00lt” may be interpreted as either midnight or noon and all *you* can do is guess.

3.6 Identifiers vs abbreviations vs offsets

Never mix identifiers together with abbreviations or offsets, like `Europe/Stockholm (CET)` or `Europe/Stockholm (UTC+01:00)`, since that will only confuse people. Always keep them apart, unless they are accompanied by a specific datetime value.

Note: You have to remember that a location may have the same time zone *identifier* during a whole year, but the time zone *abbreviation* and *offset* may shift during the same year. For example, if you say that the time zone used in Sweden is `Europe/Stockholm (CET)` or `Europe/Stockholm (UTC+01:00)`, you will tell a lie during the summer. At the moment, Sweden is always using one identifier (`Europe/Stockholm`), but two abbreviations (`CET` and `CEST`) and also two offsets (`UTC+01:00` and `UTC+02:00`).

Note: The more I read about writing a datetime value with either abbreviation (`2020-11-30 08:00:00 CET`), identifier (`2020-11-30 08:00:00 Europe/Stockholm`) or offset (`2020-11-30 08:00:00 +01:00`), I become more and more convinced that the *only* format to use should be with the offset. The offset is the only format that everyone can understand directly. As we already know, the abbreviations are useless, and with the identifiers you have to search for the actual offset value (which thanks to daylight saving time may depend on the time of year).
If you think that the location of an event is important together with the datetime value, then you should write the location explicitly and not assume that CET or Europe/Stockholm is enough.

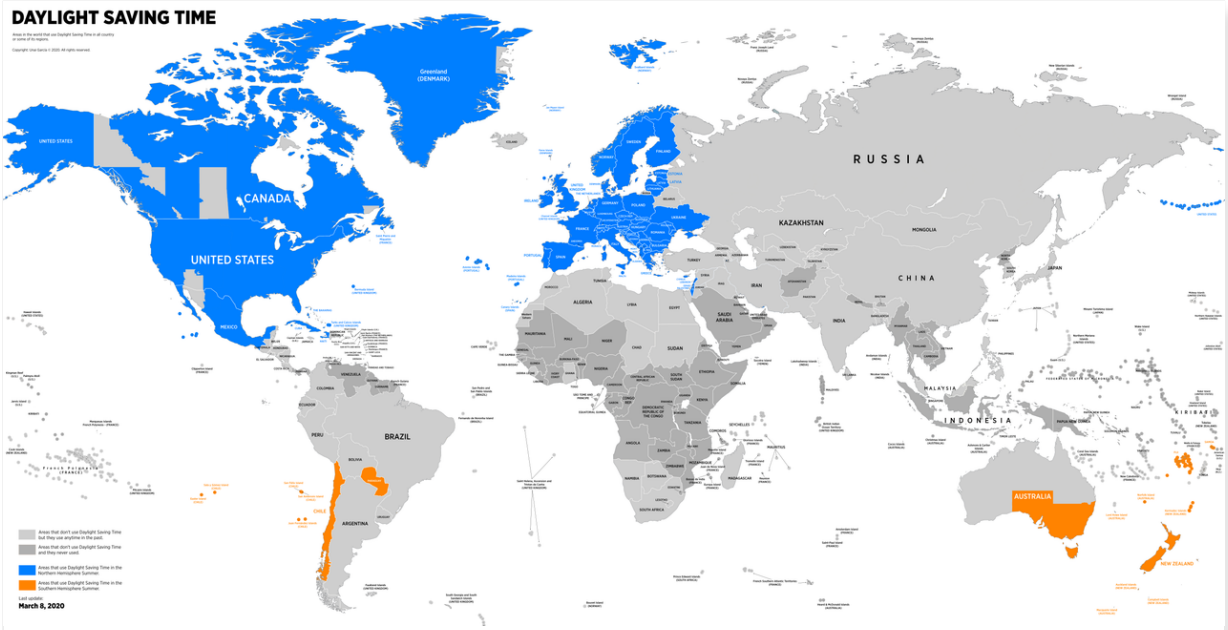
4 Daylight saving time (DST)

> The use of local time for time-stamping records is not recommended for time zones that implement daylight saving time because once a year there is a one-hour period when local times are ambiguous.
>
> Most countries have gone through an amazing list of strange decision-making in this area during this [20th] century. Short of coming to their senses and abolishing the whole thing, we might expect that the rules for daylight saving time will remain the same for some time to come, but there is no guarantee.

If I set up a meeting with you at 02:30 the morning when Sweden changes to Daylight Saving Time in the spring, should we then skip the meeting since 02:30 will never occur that morning? If we instead set up a meeting at 02:30 the morning Sweden changes back to standard time in the autumn, should we then meet twice since 02:30 will occur twice that morning? Yes, these are edge cases, but nevertheless they are real and they are confusing.

By the way, when using daylight saving time, should I move the clock forward or backward? Does DST work the same way in the Southern hemisphere as in the Northern hemisphere?

Note: The correct term is *Daylight Saving Time* and not ~~Daylight Savings Time~~, but if misspelled in quotes or resource links I have left the word *Savings* untouched. I will do my best to use the correct term myself.



Areas in the world that use daylight saving time during some part of the year.

4.1 Absurd complexity

> [Daylight saving times] clock shifts have the obvious disadvantage of complexity.
>
> 02:30:00 on March 30th 2014 might be a valid time, but in some timezones it is not.
>
> Because our ancestors were morons, they opted for a system wherein many governments shift around the local time twice a year for no good reason. And it's not like they do it in a neat, coordinated fashion. No, they do it whimsically, varying the shifts' timing from country to country (or region to region!) and from year to year. And of course, they do it the opposite way south of the Equator. This all a tremendous waste of everyone's energy and, er, time, but it is how the world works and a date and a time library has to deal with it.
>
> DST is controversial. More than 140 countries have used it at some point, but about half of them have since abolished it again.

The complexity of the rules for Daylight Saving Times around the different parts of our globe are *ridiculous* hard to understand. Some of the things you must understand:

- Not all countries use daylight saving time.
- The countries using DST do not switch between standard time and daylight saving time at the same time. Most countries do their switches during the weekends, but some countries, for example Iran, switch in the middle of the week.
- Within some countries the switches are not performed simultaneously.
- If a country uses DST, that does not mean that all parts of the country is using DST.
- The Earth is divided in northern and southern hemispheres.
- The rules are also changing from year to year.

Note: Another example of the complexity and how hard it is to understand all the rules.

4.1.1 Time difference between two places

Thanks to DST, the time difference between two places can vary during a year with confusing results.

4.1.1.1 The Equator as a divider

If the two places are on opposite sides of the Equator, it gets even stranger. Since daylight saving time is used in the "summer", locations in the Northern Hemisphere start to *add* hours to offset at the same time as the locations in the Southern Hemisphere start to *remove* hours from offset. As a result the time difference between two places in the world can vary between *three* distinct values during a year. Let us take Melbourne in Australia (UTC+1000 as standard) and Stockholm in Sweden (UTC+0100 as standard) as an example. We have the following timestamps for some random dates during 2017:

```
Melbourne                    Stockholm
2017-03-25T20:00+11:00  10h  2017-03-25T10:00+01:00
2017-03-26T20:00+11:00   9h  2017-03-26T11:00+02:00
2017-04-03T20:00+10:00   8h  2017-04-03T12:00+02:00
2017-09-30T20:00+10:00   8h  2017-09-30T12:00+02:00
2017-10-02T20:00+11:00   9h  2017-10-02T11:00+02:00
2017-10-30T20:00+11:00  10h  2017-10-30T10:00+01:00
```

The first three of these rows are also depicted in the following screenshots from the NNM Time zones analyzing tool (first, second and third screenshot).

3	TZ identifier	TZ named offset	↓ UTC offset	Diff	Local date	Local time	
1	UTC	UTC	Z	-11h	2017-03-25	09:00:00	Sat
2	**Europe/Stockholm**	Central European Standard Time	**+01:00**	**-10h**	**2017-03-25**	**10:00:00**	Sat
3	**Australia/Melbourne**	Australian Eastern Daylight Time	**+11:00** ☆		**2017-03-25**	**20:00:00**	Sat

Time difference between Melbourne and Stockholm 2017-03-25.

3	TZ identifier	TZ named offset	↓ UTC offset	Diff	Local date	Local time	
1	UTC	UTC	Z	-11h	2017-03-26	09:00:00	Sun
2	**Europe/Stockholm**	Central European Summer Time	**+02:00** ☆	**-9h**	**2017-03-26**	**11:00:00**	Sun
3	**Australia/Melbourne**	Australian Eastern Daylight Time	**+11:00** ☆		**2017-03-26**	**20:00:00**	Sun

Time difference between Melbourne and Stockholm 2017-03-26.

3	TZ identifier	TZ named offset	↓ UTC offset	Diff	Local date	Local time	
1	UTC	UTC	Z	-10h	2017-04-03	10:00:00	Mon
2	**Europe/Stockholm**	Central European Summer Time	**+02:00** ☆	**-8h**	**2017-04-03**	**12:00:00**	Mon
3	**Australia/Melbourne**	Australian Eastern Standard Time	**+10:00**		**2017-04-03**	**20:00:00**	Mon

Time difference between Melbourne and Stockholm 2017-04-03.

Note: I emphasize that if someone asks you about the time difference between Stockholm and Melbourne, the correct answer will be 8, 9 or 10 hours, depending on which time of the year it is. So you will have a really hard time to answer correctly.

Even stranger it will be if comparing Stockholm and Windhoek in Namibia, which both had UTC+01:00 as standard time zone in 2017:

```
Windhoek                     Stockholm
2017-03-25T11:00+02:00   1h  2017-03-25T10:00+01:00
2017-03-26T11:00+02:00   0h  2017-03-26T11:00+02:00
2017-04-03T11:00+01:00  -1h  2017-04-03T12:00+02:00
2017-09-01T11:00+01:00  -1h  2017-09-01T12:00+02:00
2017-09-04T11:00+02:00   0h  2017-09-04T11:00+02:00
2017-10-30T11:00+02:00   1h  2017-10-30T10:00+01:00
```

Note: Thankfully, Namibia has now skipped DST switches and instead uses UTC+0200 all year around.

4.1.2 Hours in a day

Daylight Saving Times also imply that twice a year a "day" is not 24 hours in some countries. In Sweden it will instead be one "day" consisting of 23 hours and one "day" consisting of 25 hours. Most of the countries add one hour when using DST, but in some places 30 minutes is added. So a "day" may also be 23.5 hours or 24.5 hours. Daylight Saving Times can therefore be confusing when talking about durations:

> "PT36H" could be used as well as "P1DT12H" for representing the same duration. But keep in mind that "PT36H" is not the same as "P1DT12H" when switching from or to Daylight saving time.

Note: In the quote above, "PT36H" is a short-form for "a period of 36 hours" and "P1DT12H" is a short-form for "a period of 1 day and 12 hours" (as defined by the ISO 8601 standard).

4.1.3 Time switches due to DST

Thanks to the use of daylight saving time in some of the time zones, we have about 60 distinct occasions per year when a location/area/zone is changing its relation to all the other locations/areas/zones around the world. The total number of time zone identifiers that use DST each year is about 200, so there is roughly 400 switches in total.

Note: The number of switches to/from DST varies from year to year, since countries can not decide if they should use DST or not and may change their minds. The relation between 60 and 400 is that some time zone identifiers switch to/from DST simultaneously (for example large parts of Europe).

4.1.3.1 Several DST switches in the same year

Sometimes, things are complicated even more, when for example religion is mixed into the timekeeping, so in some places we switch to and from "summer time" four times per year (instead of the usual two times per year). Why is religion intertwined with timekeeping?

Note: Moroccan authorities have (with very little forward planning) changed their minds about DST switches. They will still use the confusing Central European Time (CET) time zone though.

Note: I stand corrected, the nightmare in Morocco is not over yet.

> This year's time changes in Morocco are unique on a global scale since Daylight Saving Time (DST) is usually one hour *ahead* of standard time, while Morocco will turn clocks *back* for about one month.

4.1.3.2 Short DST periods

Sometimes the DST is not used for a long period of time.

The DST length for Fiji has been about two months for a couple of years, but in 2020 they came up with the idea that a month is enough. The way that Fiji announces the rules is also interesting.

> Every year, Fiji announces the DST start and end dates in the Government Gazette.

4.1.3.3 Special events

4.1.3.3.1 Sport

Sometimes, large sport events (like the Olympic Games) may change the DST rules. This is as bad as religion mixed into timekeeping.

4.1.3.3.2 School

As if this is not bad enough, the Brazilian authorities have come up with the idea that school exams should be mixed into timekeeping. Sadly, this is not a joke either. So the decisions that Brazil made less than one year before, did not work as planned.

> We wonder how many students will show up one hour early for the test, as a result of mobile phones showing the wrong time.

Note: So the efforts to make the students well-rested before the exams, will instead make the students even more tired, since many of them will be anxious about if their alarm clocks will wake them up correctly on the exam day. Would it not be better to postpone the time for the exams with one hour instead of changing the DST rules with so little forward planning?

Note: It will also be interesting to see how many people that will have trouble to know when a meeting should start during those two weeks. Seems as mechanical watches will be the thing to rely on.

Note: Yes, of course, the Brazilian authorities changed their minds once again, just a few days after their first decision, when they realized that they had not fully understand what they were about to do. But it was a close one this time and I leave this section as a reminder of what can happen anytime.
And if you think carefully about it, the damage is already done. Some time zone libraries may quickly have implemented the first change (that was then rejected), so dependent devices may show the wrong local time anyway.

4.2 Why building obstacles?

Why do people think that it would be so hard to make a change and abolish DST? Take the following quote regarding the EU discussions of DST for example:

> Even when a law to scrap DST has been established, it will likely take some time until it is actually implemented. Updating all systems affected by the change all across Europe is a huge task, and legislators will probably be mindful of that, allowing ample time for the technical implementation.

Why would it be a “huge task” to update “all systems” when abandoning DST? I think it will be quite easy to skip using DST (as you will see in a step-by-step solution in the *Conclusions/summary* section below). A system handling DST today is hopefully dependent on the TZ database (or similar) so the system works correctly. So the **only** task needed is to update the TZ database with the skipped DST for the corresponding time zone rules, therefore the “technical implementation” part will be really easy.

Note: The countries in EU are hopefully about to abolish the DST switches during 2019. One of the propositions seems to be that each country can decide which time zone they will use. Either they use their standard time zone or their "summertime" time zone. But, a country can also choose whatever time zone they want.
What a blessing it would be if all EU countries should choose to use the UTC time zone. All year round. That would truly be pioneering.

> **What a blessing it would be if all EU countries should choose to use the UTC time zone. All year round. That would truly be pioneering.** ”

Note: Another thing regarding the decision about if Sweden should use UTC+0100 or UTC+0200 all year around: Why can't we use UTC+0130 if we have such a hard time for consensus? Arguing that it will be darker in the mornings or in the evenings have no bearing. If it is a problem with darkness, then we must use more fine-grained time zones, so the eastern and western parts of Sweden use different local times. But that will of course not solve any problems at all, since the northern parts of Sweden still would be pitch-dark for many weeks in a row, which no time zone can do anything about. The same problems also exist in for example Norway and Finland. So if we want to avoid darkness when schools start or end in these countries, we have to come up with a completely different solution. Perhaps we should install another Sun at the opposite side of the Earth? Or move all the people towards the Equator?

Note: Some people in south of Sweden and in Denmark are afraid that Denmark and Sweden will choose different time zones and that it will be misunderstandings and confusion with train tables and working hours for those that commute. But wait, isn't that already a "problem" in the north of Sweden and Finland? And around each and every time zone border around the world? In fact, isn't that one of the main problems (and main purposes) with time zones? If it is a real problem with different local times, shouldn't we get rid of all time zones then?

> Some people in south of Sweden and in Denmark are afraid that Denmark and Sweden will choose different time zones and that it will be misunderstandings and confusion with train tables and working hours for those that commute. But wait, isn't that already a "problem" in the north of Sweden and Finland? And around each and every time zone border around the world? In fact, isn't that one of the main problems (and main purposes) with time zones? If it is a real problem with different local times, shouldn't we get rid of all time zones then? ”

4.3 Summary for DST

Perhaps DST is the most dreadful invention in our current timekeeping system.

> Perhaps DST is the most dreadful invention in our current timekeeping system. ”

I am not saying that we should not use daylight saving time, but we should *not* have to shift our clocks. Instead, we could change the working day times from 08-17 (standard time) to 07-16 (daylight saving time). *When* we do things should of course be decided locally, but what time it is when doing these things should be globally understandable.

Note: If you have not been convinced yet, just look at these wonderful videos:

- Daylight Saving - Movie Trailer
- the sequel Daylight Saving: Spring Forward - Movie Trailer

- Daylight Saving Time - How Is This Still A Thing?
- United Swing States of America - Arizona's Daylight Saving Time Opt-Out
- Changed Timezones (The West Wing)
- dst summertime berlin clock
- My cousin Oskaar

> It would put an end to the twice-a-year routine of moving the clock either an hour ahead or an hour behind – something a business magazine calls "humanity's dumbest ritual of all time."
>
> In temperate northern countries, DST usually starts late March/early April and ends late October/early November; exact start dates vary by country. Equatorial nations typically use no DST; southern nations will use dates that match their local summer. It's not unheard of for an individual province or state — or even a piece of one province — to opt out of a DST scheme in effect in the rest of the same nation. Due to the nature of daylight saving time the difference in time zones may vary during the year as one country doesn't have daylight saving time while the other does, or both have it but start at different times. However due to increasing commerce and international communication via the internet and other nearly instantaneous modes, there are increasing efforts to harmonize those things, especially among direct neighbors or political entities with good relations with each other.

Note: If you want to enhance your knowledge around Daylight Saving Time issues and get a better understanding for time zones in general, try out my analyzing tool NNM Time zones.

Note: One of the best sources for information about DST is the site #LockTheClock - Stop Changing Clocks for Daylight Saving Time. The posts are really well written and educational. Much focus on getting rid of DST switches in the United States, but the situations in the rest of the world are also discussed.

Note: The Daylight Saving Time Explained video explains DST in an easy way.

5 Time zone conversions

These calculations become more complicated near a daylight saving boundary.

These conversions are complex, and hence a failure.

6 Noon and solar noon

One of the reasons to have time zones, is that no matter where you live on the Earth, when your clock is 12:00 (noon) the Sun would be at its highest altitude in the sky (solar noon or zenith).

But, since each time zone (often) spans 15 degrees this will not be completely true for most of the places on Earth and therefore nothing you can rely on. The solar noon, for most places, will instead be somewhere in the span 11:30 to 12:30. For some places though, for example in Spain and China, the solar noon will be way outside this one-hour interval.

Note: Setting the clocks after the solar noon was a good idea until the 19th century. But now it is the 21st century and this is *not* a good idea anymore. Furthermore, the current time zones misuse the concept of solar noon.

6.1 Solar noon vs solar midnight

A solar noon, hopefully, occurs exactly once every date. But a solar midnight does not occur for every date. Some dates may even have two solar midnights.

6.2 Solar noon vs sunrise/sunset

The time difference for solar noon between two locations will always be the same. The time difference for sunrise (or sunset) between those locations may vary a lot during the year.

7 Celebrating a new year

The administration of when a specific place on Earth will enter a new year is tremendously error prone. Every year when media shall make lists of countdown times to the new year for different places the lists contain errors. So you can not rely on the lists without double checking them.

The fact that, every new year, there are more than 30 occasions/points on the time-line that represent a "new year" is hard to understand.

8 Problems within large countries

I will show some examples of existing problems for some large countries. Either when the country has decided to use several time zones or when the country has refused to use several time zones.

8.1 Countries with several time zones

Some countries use many different time zones. The complexity of these solutions vary from country to country.

8.1.1 Australia

Australia has five different (standard) time zones (or perhaps it is just three). These are UTC+0800, UTC+0845, UTC+0930, UTC+1000 and UTC+1030.

There are also a couple of places in Australia, that use separate times than the states/territories they lie in.

Furthermore, some of the states/territories use Daylight Saving Time, but not all. For some of the time zones the DST is one hour and for at least one time zone the DST is 30 minutes.

> Australia has multiple time zones. Some of them are half-hour and quarter-hour time zones. Not all states and territories use DST.

Just looking at the table for time zones in Australia makes me dizzy.

How the people in Australia know what the time difference is between two random places *in their own country* on a random date is an unsolved mystery to me.

> How the people in Australia know what the time difference is between two random places *in their own country* on a random date is an unsolved mystery to me. ”

8.1.2 United States of America

United States of America has nine different time zones. Or is it eleven? Of course not all of the time zones in US use Daylight Saving Time and to make things even worse the states do not even switch to and from DST simultaneously. The state Arizona is a remarkable example in how things can go wrong with DST.

Note: All the discussions in this section (and many of the other sections) may be completely wrong from 2023, if the United States decides to go for permanent DST.
As I wrote about the EU countries above, the same would be true for the United States: What a blessing it would be if all US states should choose to use the UTC time zone. All year round. That would truly be pioneering.
It will be very interesting during 2022-2023 to see which of all the alternatives the US will end up with for their solution, and in which way the time zone map will be reorganized. This may have a huge impact for everyone around the world that handles time in some way.

> If driving the correct route from the Arizona state border through both Navajo and Hopi areas to the other side one can end up changing one's clock 7 times!

Let us take the three cities Chicago (UTC-0600), Los Angeles (UTC-0800) and Honolulu (UTC-1000) as an example. We have the following interesting table for the switch to DST, that happened 2017-03-12. There is exactly one hour on the timeline between each row:

```
Honolulu      Los Angeles   Chicago
21:00-10  2h  23:00-08  2h  01:00-06
22:00-10  2h  00:00-08  3h  03:00-05
23:00-10  2h  01:00-08  3h  04:00-05
00:00-10  3h  03:00-07  2h  05:00-05
01:00-10  3h  04:00-07  2h  06:00-05
02:00-10  3h  05:00-07  2h  07:00-05
03:00-10  3h  06:00-07  2h  08:00-05
04:00-10  3h  07:00-07  2h  09:00-05
```

Note: I have skipped the minutes part in the UTC offsets. Instead of using `21:00-10:00` I used `21:00-10`. It would be even harder to understand the table otherwise. `03:00-07:00` could easily be mistaken as a four hour interval (instead of "three o'clock in UTC-0700"), while it is less risk with `03:00-07`.

So before the DST switch, the time difference between Honolulu and Los Angeles was 2 hours and between Los Angeles and Chicago also 2 hours. But since Honolulu did not use DST, during the 150+ days that DST was active, the time difference was 3 hours between Honolulu

and Los Angeles, but still 2 hours between Los Angeles and Chicago.

Note: What is the time difference between New York and Los Angeles? Most of the time it is 3 hours. But, during a couple of hours each year the difference is instead 2 hours and sometimes the difference is 4 hours.

For some unclear reasons, the state Florida currently uses two time zones, CST (Central Standard Time) and EST (Eastern Standard Time). How do you know what local time to use if you are in the north-west parts of Florida, if you are unfamiliar with the detailed information about the state? Texas is also using two time zones, MST (Mountain Standard Time) and CST (Central Standard Time), within the state. As a matter of fact, several other states are also using two time zones, for example Indiana.

> With businesses on one side of the street being in a different time zone from those on the other.
>
> Indiana businesses have lost hours of productive time with out-of-state colleagues because the time quirks are too confusing to keep track of on a daily basis.

Note: A detailed information page about time zones in Florida can be quite complex. And this is just to keep track of what the time is in this small part of the world.

Note: Imagine two persons standing a few meters apart in the north-west parts of Florida and discussing what time it is. That could be a really interesting conversation (especially during the daylight saving time switches nights).

> A person may take knowing the local time for granted, but an official review revealed that there is no single, accurate map showing the nation's time zones and local observance of Daylight Saving Time.

Yes, it is also a mystery to me how the people in the United States of America know what the time difference is between two random places *in their own country* on a random date.

> Yes, it is also a mystery to me how the people in the United States of America know what the time difference is between two random places *in their own country* on a random date. ”

8.1.3 Russia

Russia has eleven different time zones. Currently they are not using Daylight Saving Time at all, which is a good decision. Even though Russia has more time zones than Australia and United States of America, it is far less complex *at the moment* to decide time differences within Russia than within the other two countries.

8.1.3.1 The 2010s

One problem in Russia though is that during the 2010s the time zones have been reduced and shuffled around from year to year. Russia used nine time zones during a couple of years and during this time they completely skipped to use the UTC+4 offset. Many locations had to switch their actual time zone or offset back and forth.

> Russia has a history of time zone changes.

Note: If you want some details, please read:

- Plan Announced to Reduce Russia’s Time Zones
- Russia Reduces Number of Time Zones
- Russia Proposes International System of Time Zones
- Russia Returns to Standard Time All Year
- Russia Changes Several Time Zones

So yes, it is also a mystery to me how the people in Russia know what the time difference is between two random places *in their own country* on a random date.

8.1.4 Mexico

Mexico has four different time zones. The time changes within Mexico during a year are hard to understand.

> With its varying Daylight Saving Time (DST) schedules, Mexico's time zone situation frequently causes confusion.

> A few northern Mexican border towns, such as Tijuana and Juárez (Ciudad Juárez), start DST on March 11 along with the US and Canada.
> However, most of Mexico starts DST on Sunday, April 1, 2018. A few Mexican cities, like Hermosillo and Cancun, do not observe DST at all.

8.2 Countries with one time zone but spanning several

There are also some countries that span several time zones, but have chosen to use only one time zone for the whole country.

8.2.1 China

China is a large country from west to east and spans more than 60 degrees of longitude and therefore China could have used five time zones. Still, China uses the same time zone (UTC+0800) in the whole country.

So in the westernmost parts of China, the solar noon can be after three o'clock in the afternoon. Some of the population in these westernmost parts unofficially use UTC+0600 as their time zone, which can give some interesting conversations:

> While the Uyghur population usually go by Xinjiang Time, known colloquially as “local time”, other ethnicities, like the Han Chinese, normally refer to Bejing Time. This means that visitors asking for the current time in the streets of the region's capital Ürümqi might get 2 conflicting answers, depending on whom they ask.

Note: I do think though, that China has made a good decision with using only one time zone. And if they can, why can't we do it all around the globe...

8.2.1.1 Border between China and Afghanistan

If you wander about aimlessly in the nature reserve at the China and Afghanistan border, you may have to adjust your clock 3.5 hours from one step to another, if you want to keep up with the correct local time.

> The border marks the greatest terrestrial time zone difference on Earth, with a 3.5 hour difference between Afghanistan's UTC+4:30 and China's UTC+08:00.

8.2.2 Norway

Norway is a small, but wide, country. Norway spans three different time zones, but uses just one. The standard time used everywhere is UTC+0100, although the westernmost parts could use UTC+0000 and the easternmost parts UTC+0200.

Since Norway lies relatively far from the equator and is stretched in southwest-northeast direction, this have some interesting effects.

8.2.2.1 Comparing Bergen and Kirkenes

If we compare the local times for sunrise and sunset in Bergen (a city in the southwest part) and Kirkenes (a city in the northeast part) during some interesting dates in 2018:

	Kirkenes		Bergen	
	Sunrise	Sunset	Sunrise	Sunset
January 28	09:03	13:23	09:04	16:39
March 18	05:09	17:08	06:46	18:48
May 6	02:06	21:51	05:21	21:50
November 14	08:36	12:51	08:36	16:09

Note: View Bergen and Kirkenes in NNM Map Viewer (with longitudes -7.5, 7.5, 22.5 and 37.5 marked, i.e. indicating the ideal time zones for UTC+0000, UTC+0100 and UTC+0200).

Note: The difference in longitude between Bergen and Kirkenes is roughly 25°, which implies that the solar noon in Kirkenes is 100 minutes before the solar noon in Bergen.

January 28 and November 14, the Sun rises at the same local time at both places, but sets more than three hours later in Bergen, thanks to the tilted Earth. May 6, it is the other way around, the Sun sets the same time, but rises three hours earlier in Kirkenes.

March 18, the "daylength" is 12 hours at both places, although Kirkenes is more than 90 minutes "ahead of" Bergen.

In Kirkenes, during a part of the summer the Sun is "up all day" and during a part of the winter the Sun is "down all day", which makes the city a great example of that you can not conclude if the Sun is up or not from local time alone.

Note: The above "local time issues" in Norway are of course not "time zone problems", they are more "tilted globe" issues.

9 Countries changing time zone

Sometimes a country, for whatever reason they want, decides to change the time zone (UTC offset) they are using. For example Kiribati (1995), Samoa (2011), North Korea (2015) and South Sudan (2021).

Note: The forward planning is far too often not so good and the information about when and how quite ambiguous, as for example for South Sudan. This will of course make many systems around the world out of sync for a time, before they have had a chance to catch up. Or as their spokesperson (missing that with great power comes great responsibilities) said:

> Everybody is under the duty to adjust whether internationally. Even the airlines have to adjust according to what we are saying. So everybody is under the duty to adjust.

Note: The understanding of the impact on the rest of the world and how important it is with official announcements is sometimes very low.

Note: Ten minutes past the time zone change for South Sudan 2021-01-31 I did some random verifications on a handful sites that claim they tell you the correct local times around the world. Searched for "current time south sudan". Some of them had the correct `2021-01-31 23:10 CAT` and some still had the erroneous `2021-02-01 00:10 EAT`. One site that had the correct local time, showed the incorrect "time zone" (EAT (UTC+03) instead of CAT (UTC+02)). Another site indicated that South Sudan had started DST February 1, but with DST active it should be UTC+04 and not UTC+02. Yet another shows correct offset and correct time, but fails with the abbreviation. Google's own result, shown as the first and most prominent result in the list, showed the erroneous local time.
So the results were `2021-01-31 23:10 CAT`, `2021-02-01 00:10 EAT`, `2021-01-31 23:10 EAT` and `2021-01-31 23:10 UTC+02` (and DST active).
How should anyone be able to understand anything?
Yes, my own site NNM Time zones also shows the erroneous local time. But I already explicitly stated that “the only thing you can be sure of is that errors exist”. In a few days, without me doing anything, it will magically indicate the correct local time again for South Sudan when all third-party libraries, browsers and computers get synced to the latest version of TZ database. I am mostly glad about this error, since it only emphasizes the problems with time zones.

Note: Two days after the time zone change for South Sudan there are still many errors on the "top sites". Google is now explicitly showing the correct offset (UTC+02) in their result, but still displaying the local time in the UTC+03 offset. Everything is really confusing at the moment.
There are no indicators anywhere that systems are out-of-sync (or in-sync for that matter). Since this is how all(?) systems work, how can you be able to trust any values you find out there? How do you know that a system uses the correct version of time zone rules?

Note: After more than three days, Google is now finally returning the correct local time for South Sudan to me. Many of the datetime sites are still displaying the wrong time though.

Note: After more than four weeks, you can still not be sure of what the local time in South Sudan is. Many sites now show the correct local time, but still many of these sites say that the time zone is Eastern African Time and not the correct Central African Time.

Note: I was completely wrong about how long it would take until my site NNM Time zones would show correct local time for South Sudan. It was not days, it took over five weeks, until it showed correct information (regarding offset and abbreviation) in my computer/browser (I did not have to do anything myself though).
So for way over a month, the data was completely wrong, but there was no indication whatsoever about that, because there is no *easy* way of knowing that.
Just imagine how many sources on the Internet that are broken in the same way from time to time.
Of course, I have no idea about how correct the information is for people using other operating systems and browsers.

9.1 Parts of countries changing time zone

Sometimes a *part* of a country, for whatever reason they want, decides to change the time zone they are using. For example the states Maine and Massachusetts (in the United States of America) and the Volgograd region (in Russia).

9.2 Just skipping DST will be no problem though...

The problems described above with getting out of sync when countries change time zone with little forward planning, may of course also be present when countries decide to skip daylight saving time. For example, Samoa decided, with a margin of ten (!) days, that they should skip DST.

> Much of Samoa woke up in confusion today as mobile phones, computers, and other cellular-enabled devices displayed the time as 10:00 when it was in fact 09:00. Many reportedly arrived at church early because their cell phones incorrectly sprang forward.

Note: When, 2021-09-25, asking Internet for the current local time in Samoa, I now get different answers. For the four top results to my search query, two sites inform me correctly and two sites inform me incorrectly. The first and most prominent result, written in large font at the top, is (why am I not surprised) incorrect.

Note: In my experience, when it comes to sites that you can count on whenever the time changes (whether it is DST rules changes or offset changes), the timeanddate.com seems to be one of those that is correct most of the time. Many of the other sites are slow on updates, including Google, and way too often display erroneous information. The thing you always have to understand, is that it is difficult to know if the time they display really is the correct current time.

9.2.1 Changing the date for DST switching must always be okay...

With a margin of eleven (!) days, Palestine decided that they will end DST one day earlier than previous plans.

> Unfortunately this distribution process takes time. I expect that most cell phones and other computing devices in Palestine will not be updated before winter time begins next week, so they will display the incorrect time for a day or more. To avoid similar problems in the future, it would be helpful if the Palestinian government could give sufficient notice of when daylight-saving changes will occur.

Note: What is happening at the moment? Are there no grownups involved in decisions anymore?

Note: Now Palestine thinks that twenty days should be enough for forward planning. Have they not learned anything?

Note: One year later and Palestine continues to confuse us all.

9.2.1.1 Is a new record set by Lebanon?

In 2023, no one in Lebanon (or anywhere in the world), know what the actual time in Lebanon is. Once again, it seems to be religion causing confusion and chaos.

> This has propelled the Mediterranian country into **time zone confusion and conflict**.

Note: Where is the world heading, with all these stupid decisions from "leaders"? I thought that the worst decisions about daylight saving time was in the past, but the last couple of years have sadly proven me wrong.

10 Computer systems

When you see a timestamp *without* a time zone or offset notation (e.g. `2017-04-13T20:30:40`), in an application, a web page or even in an analog paper, you start wonder if the timestamp is converted to your local time zone (as defined by your application settings, browser settings, mobile settings or computer settings) or if it is in some "standard" (read random) time zone. How should you interpret the timestamp? A timestamp without a proper notation of time zone, is that really a timestamp at all?

Note: No, of course a timestamp without time zone notation can not be a proper timestamp (see details in the *Converting date and time values* section below).

If you on the other hand, always would have to show the time zone or offset notations for timestamps (e.g. `2017-04-13T20:30:40+08:00`), the information become cluttered.

Since the time zone used can depend on your application settings, your browser settings, mobile settings and/or your computer settings there is always a risk that you will have to look at timestamps with different time zones, depending on which application you currently are using.

This also means that you may have to configure your time zone in your computer settings, in your browser(s) settings and in all your applications settings. You may also have to readjust all these settings if they get out of sync for some reason. Most certainly, there is no trivial way of seeing *all* your configured values, so you have to quest for them all around your systems.

Note: If I see a "timestamp" like `2019-02-03 14:15:00` in my own computer and someone ask me what point in time that value represents, I *always* have to answer "I have not the slightest idea".
That would in fact always be my answer whenever I see "timestamps" missing vital information.

Furthermore, if a person shows you a "timestamp" value that lacks time zone and offset notation on *her* computer, you must also know exactly how *she* has configured *all* her settings, otherwise you have no chance of understanding the timestamp value you are seeing.

Note: As a matter of fact, even if you know all of her settings, you can not fully understand the value, since you will not have a clue about *which* of the settings, if any, that is used. Unless you dig deep into the program code.

> The AngularJS datetime filter uses the time zone settings of the browser. The same application will show different time information depending on the time zone settings of the computer that the application is running on. Neither JavaScript nor AngularJS currently supports displaying the date with a timezone specified by the developer.

Computer systems that need to handle timestamps with different time zones are (almost) guaranteed to contain errors.

Note: "Arrow is a sensible and human-friendly approach to dates, times and timestamps." Are they kidding? It is only another proof of the complexity of how hard it is to handle dates, times and timestamps when you mix them with time zones. No, I have not read the complete page, my brain exploded just by scrolling down to the end. I emphasize that this is only one (1) of all the libraries to one (1) of the programming languages out there. No other library is of course any better. I also can not trust libraries that in their documentation and examples use deprecated time zone identifiers like "US/Pacific".

Note: This is the reality for many users of computer systems and applications.

10.1 Configuring time zone

Configuring your time zone in computer systems is often a confusing task. Even when using applications from global companies, you can not fully understand what you have configured. Far too often when configuring your time zone you have options like `Europe/Stockholm (UTC+01:00)`. During standard time it would be okay, but since Stockholm is using DST half of the year, this option value is instead very confusing to put it mildly. If the option would be shown as `Europe/Stockholm (UTC+02:00)` during DST it would be (a bit more) understandable, but when using `Europe/Stockholm (UTC+01:00)` it will be ambiguous, since you can not understand if your time values are shown in UTC+01:00 or UTC+02:00.

Note: Using `Europe/Stockholm (UTC+01:00)` in Sweden during DST is not just ambiguous, in fact it is completely wrong.
You have to remember that an identifier and an offset are *not* the same. The identifier `America/New_York` is sometimes "equal" to the UTC-05:00 offset, but that same identifier is sometimes instead "equal" to UTC-04:00. So you should not mix identifiers and offsets statically together as in `America/New_York (UTC-05:00)`.
In one way or another, if you really must mix identifiers and offsets together, you have to indicate if DST is currently active for an option. It could for example be `America/New_York (UTC-04:00)` or `America/New_York (UTC-05:00)*` during DST. Otherwise you have to stick to only using the identifier `America/New_York`.

10.1.1 Using only abbreviations or offsets

If the application only lets you use abbreviations or offsets when configuring your time zone, you *must* manually adjust these yourself everytime a DST switch occurs. There is no way that the application can understand if it automatically should adjust for DST or not. The abbreviations and offsets have no knowledge about DST rules.

Note: We take the interesting state Arizona (US) as an example. If you can configure `America/Phoenix` (or `America/Denver`) as time zone, then everything *may* work fine. If you only can select `MST`, `MDT` (or the scary `MT`), or just the UTC offsets, you are in trouble. If you selected `MST`, the application can *not* assume that you want to be in `MDT` half of the year. The application can *not* assume that you do not want to be in `MDT` either. So you are the only one knowing when to change, so you have to do this yourself. Did I say that these abbreviations are useless?

Note: I emphasize this with another example. If your application lets a user select CET as time zone, you will probably have many users with incorrect time zone half of the year. But you can not automatically change to CEST when Europe changes to DST. There are some African countries that also use CET and they do not switch to CEST, as you will see in the *Misleading time zones* section below. So how should your application be able to know what to do?

10.1.2 Configuring vs displaying

When *configuring* time zones, you should probably only allow *identifiers*, to minimize the problems with daylight saving time. When *displaying* datetime values you should probably not use identifiers, but rather the actual offset. Please, do not use the **useless** "identifiers", for example `US/Pacific`, `US/Eastern`, `Pacific Standard Time`, `Eastern Standard Time`, `Central European Time`, `PST`, `MST`, `EET`, since these have no information about DST switches, which will most certainly not be what the users want.

Note: As an example if you have a user in New York, let her configure the time zone as `America/New_York` (never as `EST` or `US/Eastern`) and when displaying actual timestamps to her, use the format `2020-12-10 15:00 -05:00`. This way, if she for example would have a digital presentation, everyone around the world can understand exactly what that timestamp really mean. Using the display formats `2020-12-10 15:00 America/New_York` or `2020-12-10 15:00 EST` instead confuse most people.
Once again, you have to remember that an identifier tells you *nothing* about what the actual offset is for a specific datetime value and the offset is probably the most important part for people reading the value.

Another problem with skipping to show the actual offsets when displaying values to a user is that the values may be misinterpreted. If you display a list of logged events, you may be really confused.

```
2021-11-01 02:55 Email sent
2021-11-01 04:05 SMS sent
```

Most of us will probably interpret this as the SMS was sent 70 minutes after the email. But what if there was a DST transition between these two events? If the transition was into DST, the SMS may have been sent only 10 minutes after the email. If the transition instead was from DST, the SMS may actually have been sent 130 minutes after the email. How should you understand this if you don't see the actual offsets?

Note: No, it is not enough to show the identifier or abbreviation, since most of us don't have a clue about when exactly there is a DST transition for every identifier/abbreviation. With an identifier, you don't display any information at all about if DST is active or not. With an abbreviation, even if it is two different values, the reader may not easily understand which of the values that is denoting DST.

10.1.3 User interfaces

The time zone identifiers should not be used directly and the end user should not have to see these strange values when selecting a time zone. This is the recommendation from the maintainers of the TZ database. These people recommend user interfaces that, for example, looks like the configuration mechanism in macOS, where you choose a zone from a map.

> The choice of spelling should not be important to end users, as the tzdb spelling is not intended to be visible to them. End users should see their preferred spelling which would typically be Київ, but could also be Kyiv, Kænugarður, Κίεβο, 基輔, or whatever else is appropriate for the user's locale. [...] Admittedly it is all too common for software applications to expose strings like "Europe/Kiev" to users who prefer a different name.

Unfortunately, most of the user interfaces I have seen, use some sort of select list, with area and location.

Note: The majority of us will probably understand the globe/map, and we can select a correct zone even if we are travelling to new places around the world. But the majority of us will not understand which location to use even if we will be presented with the area, since most of us are not fluent in city names around the world. Remember though, you should not always select the identifier/city that is *closest* to your current location. Many times you have to select an identifier/city that is further away, in order to get the *correct* time zone rules.

Note: The following great quote from a maintainer of the TZ database, indicates what they have to endure. If people only could understand how much more complex the world would be if we did not have the TZ database at all. I can really relate to what he says :)

> What people's opinions are about what they wish is irrelevant: what matters is what the reality is in the region, and we reflect that, we don't decide that. They might also wish we used the calendar they use, but we don't, and don't care, and won't change, even although the support and capability has been available for decades, and many rules would make more sense expressed in their native calendars.
>
> It always favours what is used by those people who reside in the region, whether they have the government they would choose or not. Many of us might also prefer to have different governments, or use different time zones than those in effect where we reside. If

you really dislike a time zone or a government, you will have to try to change it, or else move elsewhere, otherwise you will just have to put up with what you have.

The users of the data are the downstream systems using the project, and they are not pushing us to make any of these changes, they deal with their issues internally, and have to put up with the project making random unplanned changes, because of a few emails about spelling in a language not native to them, that would better be automoderated out of notice.

The naming is **meant** to be an internal identifier, and I personally hold the opinion that they should never be changed, until association or comprehension is misleading or negligible, as they are irrelevant outside of this project.

I see a lot of code in a lot of projects with crappy, insane, or even inverted identifiers, but I don't waste my time complaining about those usages, or submitting mass variable renaming patches.
I just hope that the contributors or maintainers will sensibly refactor those usages when sufficient maintenance is required, or confusion has been caused.

If downstreams or other sites expose or misuse those identifiers, the complainers should be told to complain about wherever they see them, as they chose to expose or misuse those identifiers.

Those downstreams, few of whom likely contribute to this project, rather it is the reverse where this project contributes freely to their business, as is our preference, nor do most acknowledge, nor care about this project, as the information is in the public domain, seem to do a better job of saying **NO** to complainers, who also have no vested interest in this project, do not make any contributions, and do not really care, except about petty political campaigns, to which they have been stirred up by ineffectual politicians, where the best they can do is provoke people to argue about spellings in a language they don't use, nor know much of.

Having caved to Ukraine, I would not be surprised to see more campaigns, to press for more politically motivated changes, that useless politicians can point to as accomplishments when nothing else is progressing.

That is why I am a verbose advocate of telling complainers to submit patches or PRs for consideration, fork the sources on github, patch the software they use to suit themselves, or **go pound sand**!

10.1.4 Summarizing problems with UI/UX

I have not yet seen a perfect solution for selecting time zone. Most of the solutions out in the wild are actually really awful, and it is hard to understand if the selected time zone will work correctly for you.

Note: There are no perfect solutions, because the problem is way more complex than people think, and often they have not understood even the basic requirements for how to use the selected time zone in their application.

10.1.4.1 Useless values

As already stated in this chapter, one problem is that many solutions let you select values like:

- `EST`
- `PDT`
- `Pacific Standard Time`
- `Pacific Time`
- `Western European Time`
- `UTC+3`
- `UTC-8`
- `UTC-08:00`
- `(UTC+01:00) Amsterdam, Berlin, Copenhagen, Oslo, Stockholm, Vienna`

I don't think this will work for the user, since you have no information about the rules for Daylight Saving Time for these values. The user will be stuck with the offset connected to that value, which for many people around the world will be incorrect and/or create confusion.

If using these values in a calendar application, a user that have selected `CET` will probably not understand why the application is not automatically handling Daylight Saving Time for her. The application can not just change to `CEST` either, because that may be a wrong assumption.

10.1.4.2 Which time zone should I select?

Another problem is that it is sometimes difficult for a user to understand which time zone to select. It is not always the *nearest* location that should be selected. You need to understand the borders of the time zones, so you pick the *correct* one. This is far from trivial for many locations around the world.

Note: I emphasize that often you should *not* select the time zone identifier that is closest to you. Close, but no cigar. For example, if your location is in the south of Sweden, both Copenhagen and Berlin are closer to you, but it is still Stockholm you have to select as identifier, if you want to have a correct time zone.

10.1.4.3 Recommendations

My recommendation for user interfaces in computers and mobile phones is to only use the identifiers (tz/IANA) that follow the pattern `Area/Location`, together with the almighty `UTC` (Zulu). If using these you will minimize the problems with Daylight Saving Time switches, since these time zones have rules for DST. As soon as you start adding other values, you and your users will be confused.

Note: As you probably know by now, my recommendation is actually to only use `UTC`, which also implies that there is no need for selecting time zone. That would, of course, be the perfect solution to the problem.

Note: Using all the military time zones in the selection is not valuable either, since `Zulu` will be enough.

Note: You have to understand the requirements for what the configured time zone should be used for. If the time zone is going to be used for automatically converting values for the users, you can't use "time zones" that force the users to manually change them from time to time, to keep up with DST switches.

You should also always be explicit about the actual time zone used, when displaying values that relate to a specific time zone. Don't make the user think and guess!

Note: Yes, it still is the same as writing explicit units for lengths, weights and temperatures, which you probably/hopefully will never leave out.

Note: I would love if I in my computer and mobile, only had **one** (1) place, where I had to configure my time zone, and all the applications would use that configuration. I really don't like the solution that we sometimes need to do these annoying configurations in all of our applications.
I have already configured my time zone in macOS, so why is that value not used everytime date and time values are displayed?

10.1.4.3.1 Users not understanding time zones

To be able to configure a time zone, you must understand what your time zone is. If you configure your time zone erroneously, the application can never function correctly for you. At the same time, users not understanding their own time zone, will probably not care about date and time values they see in their applications anyway.

Note: How can it be that we think we know what the local time is, when we don't have a clue about what time zone we are in? Many people thinking that they should select UTC-08:00, PST or Pacific Time, will probably end up with incorrect local time. If you don't know your correct time zone, how did you manage to set your wall-clock in the first place?

> How can it be that we think we know what the local time is, when we don't have a clue about what time zone we are in? ”

11 Converting date and time values

> A time without a timezone is as dangerous as a measurement without units.

Many people do not know how important it is to define a correct time zone when they are defining a timestamp.

> Many people do not know how important it is to define a correct time zone when they are defining a timestamp. ”

If you have a timestamp *without* a time zone or offset notation (e.g. `2017-04-13T20:30:40`), you can *never* convert that to a specific user's time zone. In fact, that timestamp isn't a timestamp at all. The "timestamp" you have is not representing *one* point on the timeline, but instead *several* distinct points on the timeline, and should therefore be treated as a "local observed time" value. So trying to convert this "timestamp" value into a user's time zone will *always* be incorrect.

Yes, if your timestamp has a time zone or offset notation, you can convert it to a user's time zone. That is because the timestamp represents a unique point on the timeline. But, it is still a needlessly complex operation and often only add confusion (as described in the *Computer systems* section above).

Note: If you want to be correct when presenting a timestamp, you *must* define the UTC offset (or something similar). Otherwise, it would be like presenting a mass as 115, a distance as 203, a duration as 48.7 or a speed as 37.6, which make no sense at all to anyone.

Note: I can't count the number of times I have been told that we need to convert a time or date value into the user's time zone, so it will be easier for the user to understand. When I ask what the time zone is for the original value, they answer that the original value lacks time zone information.
It is as if I should be required to present the user's age, without having information about the user's date of birth. Or converting the weight of a package into other units, without having the original weight with *any* unit.
We can do a lot as developers, but we can not "do magic with the knees" as we say in Sweden.

11.1 Date, time, datetime and timestamp

There are several types of "date and time values" that should be handled with care, since they are different. Being lazy and mix these values implicitly, will cause confusion.

There are **local** variants and **zoned** variants. Local dates, local times and local datetimes lack information about time zone. It is like a length measurement lacking unit. So *do not try to convert these local values to a user's time zone*, since you will probably always do it wrong. To be able to convert to other time zones, you need a correct time zone from the beginning. Think of time zones as the "unit" for time values.

Note: Always be suspicious when reading date and time values that lack information about time zones. You *must* treat these values as local date and time. You can *never* treat them as timestamps.

Note: You should also be suspicious when reading date and time values *with* time zone information, since the time zone may not be correctly used.

Note: Local versus locale. These are two completely different things. *Locale* defines *language* (and some other things) for a user. *Local* corresponds to *location* on Earth.

Note: "Plain" is sometimes used instead of "local". So plain date is the same as local date.

11.1.1 Date

A *date* can be either a local date or a zoned date.

11.1.1.1 Local date

A *local date* is of the format `yyyy-mm-dd`. No time part is defined. No time zone is defined.

A local date represents an *interval* (about 50 hours) on the timeline.

11.1.1.2 Zoned date

A *zoned date* is of the format `yyyy-mm-ddZ`. No time part is defined. A time zone is explicitly defined.

A zoned date also represents an *interval* on the timeline. But for zoned dates the interval is about 23-25 hours on the timeline.

Note: You can thank the Daylight Saving Time inventors for the extra complexity of 23-25 hours.

11.1.2 Time

A *time* can be either a local time or a zoned time.

11.1.2.1 Local time

A *local time* is of the format `hh:mm:ss`. The precision (seconds, milliseconds) may vary. No date part is defined. No time zone is defined.

A local time corresponds to the wall-clock time and therefore represents an *infinite number of points* on the timeline.

11.1.2.2 Zoned time

A *zoned time* is of the format `hh:mm:ssZ`. No date part is defined. A time zone is explicitly defined.

A zoned time also represents an *infinite number of points* on the timeline. But for zoned times the points are "fewer".

11.1.3 Datetime

A *datetime* can be either a local datetime or a zoned datetime.

11.1.3.1 Local datetime

A *local datetime* is of the format `yyyy-mm-dd hh:mm:ss`. The precision (seconds, milliseconds) may vary. No time zone is defined.

A local datetime corresponds to the wall-clock datetime and therefore represents a *finite number of points* (about 50 points) on the timeline.

11.1.3.2 Zoned datetime

A *zoned datetime* is of the format `yyyy-mm-dd hh:mm:ssZ`. A time zone is explicitly defined.

A zoned datetime represents a *unique point* on the timeline and may therefore be treated as an actual timestamp.

11.1.4 Timestamp

There is no breakdown into local timestamp and zoned timestamp.

A *timestamp* is of the format `yyyy-mm-ddThh:mm:ss.sssZ`. The precision (seconds, milliseconds) may vary. A time zone is always explicitly defined.

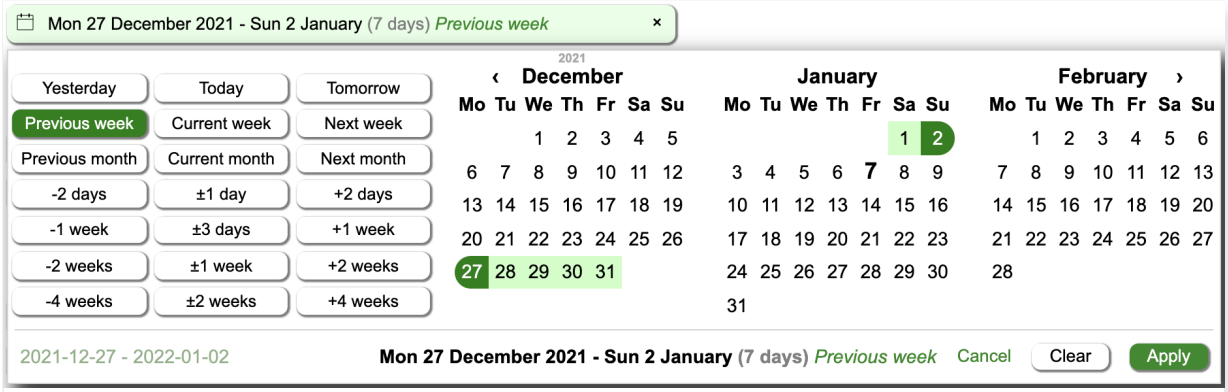
A timestamp represents a *unique point* on the timeline. A timestamp can be converted to any time zone.

Note: "Instant" is sometimes used instead of "timestamp". So an instant is the same as a timestamp.

11.1.5 Date is outdated

Since the word *date* is so confusing in its own, you should avoid using it. Be clear of what you really mean, by using *local date* or *zoned date*.

Note: An example of a tool that is explicit about local date is NNM Local date range picker.

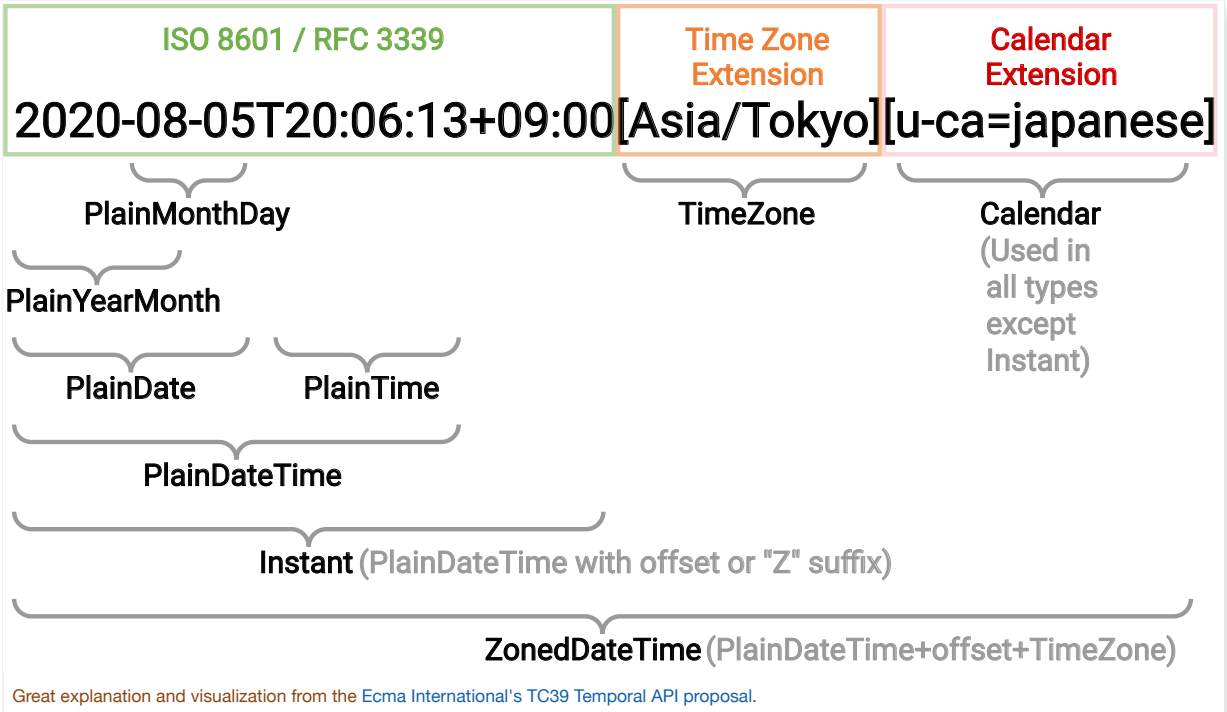


Example of the *NNM Local date range picker* tool.

Note: In programming, you often see Date objects, and often these are really confusing since it is hard to understand what they *really* represent. Sometimes they represent actual timestamps (the JavaScript Date object is a widely used example).

> It's important to keep in mind that while the time value at the heart of a Date object is UTC, the basic methods to fetch the date and time or its components all work in the local (i.e. host system) time zone and offset.
>
> The venerable ECMAScript Date object has a number of challenges, including lack of immutability, lack of support for time zones, lack of support for use cases that require dates only or times only, a confusing and non-ergonomic API, and many other challenges.The venerable ECMAScript Date object has a number of challenges, including lack of immutability, lack of support for time zones, lack of support for use cases that require dates only or times only, a confusing and non-ergonomic API, and many other challenges.

Note: Ecma International's TC39 has a really interesting and promising Temporal API proposal that hopefully will make it much easier to work with date and time within JavaScript.



Great explanation and visualization from the Ecma International's TC39 Temporal API proposal.

11.2 Writing for the Internet

When you publish texts on the World Wide Web, you have to think twice when you are writing date and time values. Your text may be read by people anywhere on the globe, so *you* have to go that extra mile in order to not leave your readers in frustration.

It is not enough to use datetime values like `2019-04-14 09:35`, because that will probably be ambiguous to everyone. If it is important that the value represents an actual timestamp, you have to add an (unambiguous) unit. Otherwise, if the exact time is not so important, you can perhaps lower the precision and only show the date part `2019-04-14`. This will of course give your value a 50 hours timespan (as described in the *Dates - are they too long?* section below), but at least your readers will not have any false assumptions on your "timestamp".

Note: You have to understand that the (local) datetime values `2018-01-28 23:40` and `2018-01-30 01:40` can represent the *exact* same time on the timeline, depending on where you live. Yes, it should be January *28* and January *30*. You can see this for yourself with NNM Time zones.

Note: Some people have understand the confusion and have been taking responsible actions.

> For an online readership, words such as 'tonight' and 'yesterday' can be confusing and misleading - so we have dropped them.
>
> We now have many millions of readers around the world, for whom the use of yesterday, today and tomorrow must be at best confusing and at times downright misleading.
>
> If you need to know exactly when a story was published, the website will continue to carry the precise day, hour and minute that it was launched ("Friday 18 February 2011 10.00 GMT"), which is much more accurate than any time reference we could insert into a story.

If you publish texts on the Internet, you have to remember though, that you should *always* inform the readers about when your text was last updated. Leaving content on the Internet with no time information about when it was created and revised is a scourge (as bad as leaving out best-before information on food).

11.3 Sorting datetime values

Let's say you have a list of datetime values like

```
2020-09-05 10:30
2020-09-04 22:15
2020-09-05 05:10
2020-09-06 01:55
2020-09-05 22:20
2020-09-05 18:45
```

Sometimes a list like this can be sorted along the timeline, but sometimes the list can not be sorted. If the values refer to the *same* location they may be sorted, but if the values refer to different locations you can not know their interrelations (as long as the offsets are unknown for you).

Note: The list *may* be sorted, if the values refer to the same location. But remember, there is no guarantee for that, since Daylight Saving Time may be used in that location.

Note: It is like sorting weights by just using the numbers. If you should sort the weights 5, 3, 10, 8, and 4, you may do this if all of them have the same (implicit) unit, but it will be impossible if different units are used.
You may say that it is stupid to list weights in this way without their explicit units. You are absolutely right, and you have paid attention so far. I can safely rest my case...

12 Times and dates in the future

Time-related things that will happen in the future are even more complex to handle, since you can not know what the time zone rules will be in the future. The rules for a country can be altered without anyone else having anything to say about it. So the exact point on a timeline for a timestamp in the future (with respect to a specific time zone), will sometimes be moved on the timeline (or get a new "local observed time" value), if that specific time zone alters its rules.

Take for example the coming timestamp `2028-07-23 12:50 Europe/Stockholm`. It *may* represent the same point on the timeline as `2028-07-23 12:50 CEST` or `2028-07-23 12:50 +02:00`. But, if Sweden has abandoned Daylight Saving Time by then, it could instead represent `2028-07-23 12:50 CET` or `2028-07-23 12:50 +01:00` (or something completely else for that matter). When using a time zone identifier for denoting a datetime in the (distant) future, you can *never* say that there is exactly *x* days, *y* hours and *z* minutes until this datetime occurs, because you have no idea of what the rules will be in the future.

> As a rule of thumb: events that have happened in the recent past can safely be saved as UTC. Examples are timestamps for when an email has been sent or a database record has been saved. Your system knows the time in UTC and can save it in UTC. And later you can always convert it to any other time zone.
>
> But in the case of future datetimes such as meetings: converting to UTC and saving UTC + timezone name is not resistant to changes in time zone rules. And remember, time zone rules change all the time.
>
> When creating an application that can store values in the future, including recurring events, you'll need to make additional data fields to ensure that you can reconstruct the user's intentions and adapt future time values to changes in time zones and time zone rules (especially alteration of daylight saving/summer time start and stop).

The Time Zone News is a depressing page to read if you want to avoid problems in the future.

Note: It is "interesting" to read about the process for changing time zones in the United States. In the section "Substantive Requirements", you learn that you must consider television and radio broadcasts, newspapers, bus and train services, airports, where residents work, the community's economy and much more, when deciding a good time zone to use.

Note: A great blog post about scheduling events. Some really good comments. Sadly also some bad comments by people not reading the full post before commenting, but those are kindly handled by the blog post author.

12.1 Scheduling events in calendars

There are many things to think about when planning event times.

12.1.1 Short events

If you plan an event or a meeting to happen between 10:00 and 15:00 in your local time, you probably want this to be interpreted to the correct local times for other persons. For example, if you are in UTC+3, you want to see the time as 10:00-15:00 in your calendar. A person in UTC+1 should instead see 08:00-13:00 in her calendar and a person in UTC should see it as 07:00-12:00. Some people may see it as 23:00-04:00 or 16:00-21:00, but the event/meeting is *always* showing exactly when it is on the timeline. This often works fine and people can understand what the start and end times for the event/meeting are.

Note: You may have to adjust for the actual *date* though, since that may be different from your local date.

12.1.2 All-day events

But when we come to the concept of all-day events, you may get confused. What do we even mean with an "all-day event"? If it denotes a birthday, that is not so crucial if being misplaced on the timeline, but if the all-day event defines a vacation day, how should we interpret it? If I say that I will be on vacation January 8 and put it as an all-day event in my calendar, what should other people see? Should my vacation be marked for January 8 in their calendars also? Or should it be marked with a 24-hour slot, correctly placed on the timeline? The first approach may be understandable, but completely wrong on the timeline. The second approach may be confusing, but at least correct on the timeline.

Note: The problem is that my all-day event on January 8 may in fact occur January 7 or January 9 for some other people, depending on where on Earth we are. For most people it will occur starting in January 7 and ending in January 8 or starting in January 8 and ending in January 9. Here you can read about a problem, and I do not fully agree on the solutions discussed (eg “an all-day event it would float as an all-day entry regardless of the creators time zone” and “all day events should indeed always appear on the same date, and without any time or timezone shown”).

12.1.3 Rescheduling issues

Imagine you have planned an event at a particular time in your time zone. Imagine your time zone offset will be `UTC-05:00` at that time. If, for some reason, your time zone alters its rules, so the offset suddenly instead will be `UTC-04:00` or `UTC-06:00`. How will you handle this?

One solution, and probably the most common, is to (implicitly) let the event move one hour along the timeline, so the event actually will happen one hour before/after the original planned event time.

Another solution, probably better if the majority of participants are in other time zones, is to *keep* the original event time according to the timeline. In this way only the (few) people in your time zone suffer from a new event time and the majority will not have their scheduled event moved in their calendars.

Note: Let us make it crystal clear. You schedule an event for 1000 participants in another time zone. Your time zone is `UTC-05:00`. The participants are in `UTC+03:00`. You schedule the event to Monday next week at `08:00 UTC-05:00`, which "equals" `16:00 UTC+03:00`. Sadly, the offset for your time zone changes to `UTC-04:00`. If you decide to let the event happen at `08:00 UTC-04:00`, *the 1000 persons will have their event rescheduled* to `15:00 UTC+03:00`. If you instead move the event time for yourself, to `09:00 UTC-04:00`, *only you will suffer*.

12.1.3.1 It is even worse than you may think

When it comes to scheduling events in the future, both for short events and all-day events, there are more problems involved when participants are from different time zones. It may come as a surprise, but you really *can not* schedule future events in a perfect way when different time zones are involved.

Since the rules for time zones can (and will) change randomly and sometimes with really short notice, there is no way to make sure that a future time *X* in my time zone, will be equal to *Y* in your time zone. It may actually be that *X* in my time zone will be equal to *Y* plus/minus *Z* hours in your time zone on that date.

As described above, I have some alternatives to handle this, but not in a perfect way. Either I can keep the event fixed to the timeline where the event was placed from beginning. Or I can keep the event fixed to my local time. Or I can keep the event fixed to your local time. For all of these alternatives there probably will be misunderstandings between us and often this will result in "double bookings" for one of us.

13 Nautical time zones

To make things even worse we have the nautical time zones. To be able to know what time it is, you now have to distinguish between if you are *on land*, *in territorial water* or *in international water*. It is hard to be a human.

Note: Nautical time zones are at least more predictable, since they follow the longitudes and are not skewed.

14 Military time zones

You also need to know about military time zones.

> The military time zone system ensures clear communication in a concise manner, and avoids confusion when coordinating across time zones.

Note: I do *not* agree with the above quote about the military time zone system. I would argue that the system adds unnecessary confusion/complexity and often leads to misunderstandings. The *only* useful military time zone is Zulu.

Thankfully, the only time zone they really seem to care about nowadays is the *Zulu* time zone, since all the others (A-Y) just complicate things and add confusion.

15 Unofficial time zones

In some countries (for example in China as you read about above) unofficial time zones are in use in some parts of the country.

16 Misleading time zones

Some countries use time zones that are quite misleading. For example, Algeria and Tunisia, use the CET time zone. CET stands for Central European Time in this case, but Algeria and Tunisia are countries in Africa. Egypt, another country in Africa currently uses Eastern European Time (EET).

Note: Reading about all the (bad) decisions in Egypt for the last 15 years makes me dizzy.

16.1 Same same, but actually different

Another implication with this strange use of time zone abbreviations is that countries using for example CET do not have to be in the same offset all year around. Not all the countries using CET as standard time zone use Daylight Saving Time. So during half of the year Algeria and Sweden use the same time zone, CET, but during the other half of the year they are in different time zones, CET and CEST respectively.

Note: If you have an application that lets your users select time zones like CET, which I strongly discourage, you can not automatically adjust the values to CEST during summer, since you will have no idea if CEST should be used or not. On the other hand, in this case some of the users will themselves have to adjust to and from CEST manually twice a year, which will not work either. The same problem exists for PST/PDT, EST/EDT, MST/MDT, ACST/ACDT, EET/EEST, etc.

16.2 Winter time

There is also something known as winter time. For example Ireland is using this, so the Irish Standard Time (IST) is used in the summer, and when most of Europe switch from DST to standard time, Ireland instead switch from standard time to "winter time".

17 Mixed time zones

In Israel and Palestine, sometimes there is also confusion about what the current time is.

> Drivers passing each other on Route 60 drove in different time zones, depending on the color of their license plates.

18 Different calendar systems

Don't get me started on this one. I will just skip it for now.

Note: One interesting thing about the Gregorian calendar. You can use either the CE/BCE (Common Era / Before the Common Era) or AD/BC (Anno Domini / Before Christ) notation. The CE/BCE is often used when you do not want to indicate the relationship to Christ. But, the biggest relationship is still there, since the whole calendar is tightly coupled to the birth of Christ.

> Because AD and BC hold religious (Christian) connotations, many prefer to use the more modern and neutral CE and BCE to indicate if a year is before or after year 1.

Note: I hope we will think outside all boxes and change to something like one of these:

- The 37818.
- 13 equal sized months.
- The Hanke-Henry Permanent Calendar.

19 Date formats

> That search should search midnight september 6th to midnight on September 7th. But it doesn't. It searches from June 9th to July 9th.

Not directly related to time zones, but nevertheless something that is really confusing: Why do we have different date (and time) formats around the world? The date values `2017-04-13`, `04/13/17` and `13/04/17` can all represent the same date, depending on which part of the world you live in.

If you are used to the `mm/dd/yy` format, how can you even know that if you see a date `01/02/03`, is it *January 2, 2003* in your date format or is it instead *February 1, 2003* in the `dd/mm/yy` format (or perhaps even *February 3, 2001* in some lazy `yy/mm/dd` format) you look at? It is quite a difference.

Note: How do you even know which century to use? It could denote 1903 or 2103 or any other year instead of 2003.

> Visitors to a web site from varying locales may be confused by date formats. The format MM/DD/YY is unique to the United States (but sometimes used in Canada, too, which can obviously create some confusion there). Most of Europe uses DD/MM/YY. Japan uses YY/MM/DD. The separators may be slashes, dashes or periods. Some locales print leading zeroes, others suppress them. If a native Japanese speaker is reading a US English web page from a web site in Germany that contains the date 03/04/02 how do they interpret it?

Note: The date format examples `yyyy-mm-dd`, `mm/dd/yy` and `dd/mm/yy` are of course only a small subset of the complex list of all the date formats that have been invented around the world. Understanding the date and time notation in the United States is not trivial.

> Date and time notation in the United States differs from that used in nearly all other countries.
>
> This order [year-month-day] is also used within the Federal Aviation Administration and military because of the need to eliminate ambiguity.

Note: As a matter of fact, the formats `mm/dd/yy` and `dd/mm/yy` are often used without zero-padded values for month and day of month. So instead of the date `06/07/19` or `07/06/19` many countries use `6/7/19` or `7/6/19`. The best part of the last sentence is that you have no idea if I meant June 7 or July 6. The difference in how you (mis)interpret `1/12/19` is over *300 days*.

> The best part of the last sentence is that you have no idea if I meant June 7 or July 6. ”

Note: Just by reading a few articles, I have seen formats that would display 6 July 2021 as:

- `6/7/21`
- `7/6/21`
- `06/07/21`
- `07/06/21`
- `2021-07-06`
- `21-07-06`
- `06-07-21`
- `07-06-21`
- `6.07.2021`
- `06.07.2021`
- `06.07.21`
- `060721`
- `210706`
- `20210706`
- `2021.07.06`
- `2021/07/06`

Note: Some of the most horrible formats for me are `21-07-06`, `06-07-21` and `07-06-21`, which always make me shiver.

Note: A real example from a Swedish car-dealer: “The offer expires 190831”. Saving a few characters, instead of being unambiguous and write 2019-08-31.

Note: "Best before" marks for groceries are a nightmare to decode. Formats like `mm/dd/yy`, `yy/mm/dd`, `dd/mm/yy`, `yy.mm.dd`, `dd.mm.yy`, `yy-mm-dd`, `yymmdd`, `ddmmyy`, etc are used. And it is up to you to try to understand them. If you find a can marked "Best before 09/10/22" would you dare to eat its content in the year 2020?

I would not eat it. Since I have no idea of which format the can-company has used.
It is like having a sign on an elevator saying: "Max weight: 150". For sure, I shouldn't enter that elevator.

> If you find a can marked "Best before 09/10/22" would you dare to eat its content in the year 2020?

Note: When visiting a hospital I filled in my date of birth details according to their confusing example format. When I later saw what was registered in their systems, they had mixed up the month and day of month (which made me almost two months younger).

Note: No, it will not be better from the year 2032. It will still be impossible to distinguish the day and month (except for the day values 13-31).

Note: Why are so many switching date format, just because they write texts in English? If you, for example, write "2021-09-17" in a Swedish text and translate it to English, you *do not* have to write "17/09/21" (or "09/17/21"). It may still be a Swedish audience your text addresses. And how should your audience easily and unambiguously be able to understand if you are using the American English or British English format in this case?

The ISO 8601 standard is a step in the right direction since:

> Date and time values are ordered from the largest to smallest unit of time.

Therefore, one of the advantages with ISO 8601 is that date, time and datetime values can be easily sorted. Try doing that with dates in the `mm/dd/yy` or `dd/mm/yy` formats.

> The ISO 8601 format YYYY-MM-DD [...] is intended to harmonize these formats and ensure accuracy in all situations. Many countries have adopted it as their sole official date format.

Note: In this article I have, by intention, mixed writing date and time formats sometimes with uppercase letters and sometimes with lowercase letters. For example `yyyy-mm-dd hh:mm:ss`, `YYYY-MM-DD HH:MM:SS` or even `yyyy-MM-dd HH:mm:ss`.
When using any format, you *must* read the documentation how it really should be used in your particular case, otherwise you *will* end up with irritating and embarrassing bugs.

19.1 Not following defined format

Sometimes the format may be defined as `mm/dd/yyyy`, but values like `2/3/2022` (lacking padded zeroes) are considered valid. Other times the format may be defined as `m/d/yyyy`, but values like `02/03/2022` (with padded zeroes) are considered valid. If you have defined a format, make sure to have proper and understandable validation, otherwise your users will get confused.

19.2 Understanding dates written vertically

When you have a format like `m/d/yy` or `d/m/yy` and display dates in columns in a table, the data may be really hard to read, since the lengths of the dates will vary (from six to eight characters). Below are some examples where the dates are *sorted increasingly*.

m/d/yy	m/d/yy	d/m/yy	d/m/yy	yyyy-mm-dd
1/1/22	1/1/22	1/1/22	1/1/22	2022-01-01
1/11/22	1/11/22	11/1/22	11/1/22	2022-01-11
9/1/22	9/1/22	1/9/22	1/9/22	2022-09-01
9/11/22	9/11/22	11/9/22	11/9/22	2022-09-11
10/1/22	10/1/22	1/10/22	1/10/22	2022-10-01
10/11/22	10/11/22	11/10/22	11/10/22	2022-10-11
12/1/22	12/1/22	1/12/22	1/12/22	2022-12-01

Note: The dates above are either left- or right-aligned, but the problem is the same. Compare the readability for `m/d/yy` or `d/m/yy` columns with the `yyyy-mm-dd` column.

19.3 Implicitly configured date formats

Never mix the configuration of a user's date format with the configuration of her language. Trying to be smart and implicitly set her date format from her choice of language, will always irritate people. Language and date format should be kept completely separated and a user should always be able to configure them independently.

> Language and date format should be kept completely separated.

A user may want to use Swedish, but not have date values like `7/4 -19` thrown in her face. She may still want to see understandable date values like `2019-04-07`. The same user may as well want to use English, but not be forced to see date values like `04/07/19` (or `07/04/19` if you select the "wrong" English language).

Note: You would not come up with the idea to force strange length units onto users, when they select language, would you? If a Swedish user selects English as language, would you implicitly force her to see miles, yards and inches, instead of kilometers, meters and centimeters?

> You would not come up with the idea to force strange length units onto users, when they select language, would you?

Note: Why is it so hard to get things right when a user should configure her date/time settings in applications?
When trying to use Teams (from one of the world's largest tech companies), you are supposed to be able to configure your language, date format, time format and time zone. But I have no clue how to get it right. Their documentation says “Selecting your preferred language and region also updates your date and time zone”. How can they even mix language with time/date? I want to have English as language, but if I choose "English (US)" I have to stick with the useless `mm/dd/yy` date format and of course the horrible AM/PM notation for time values. If instead using "English (UK)" I get the date format `dd/mm/yy`, which is also useless. There is no way that I can configure English as language and *ISO 8601* as date/time format.
As I understand it, choosing the language "English (US)" is also an implicit selection of the region (which I guess is "US" in this case). So when region is "US" I am forced to use `mm/dd/yy` as described above. But which time zone do I get in this case? Implicitly setting a time zone from region "US" can *never* work. Or am I missing something?
Why is configuring date and time formats still a problem in 2019? Why are we still mixing languages with date/time formats? Who came up with the bright idea that when choosing your language, your date and time formats are implicitly chosen (and can not be changed independently)? As so many already pointed out, why not *always* use *ISO 8601* as the default format for everyone? Why having the horrible "US" format as default for a global product?
Trying to define the date and time format in Twitter is not doable. I want to have English as language here also, but then the dreadful AM/PM notation with its 12-hour clock show up. I can not find a way to distinguish the language from the date and time format. When switching to Swedish, I get a time format that at least uses the 24-hour clock, but the language then is, well, Swedish.

Note: When trying to add an appointment in Outlook (from the same large tech company as Teams above) I literally get the following when trying to choose the time: `2019-05-29 08:00 (UTC+01:00) Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna`. The problem is that Stockholm and probably all the other cities use Daylight Saving Time that time of the year, so the actual offset May 29 is in fact UTC+02:00. So will the booking actually be from 08:00 or 09:00 that day? I don't know, I can only hope for the best, but I am not in control.

Note: An invitation to an event for a Swedish audience says “Wednesday, May 20, 2020 6:00 PM to 8:00 PM CET”. I still have no idea when exactly that was, since Sweden was using CEST that day.

Note: Why can't I in my computer/phone, once and for all, configure my formats for date and time, and then these formats are *always* used, regardless which site or application I am using? Why do I still have to configure user settings in each and every application? Why must I go into each and every application that I use, and configure the date format, the time format and the time zone? Why can't each and every application use the configured *computer/phone* settings? Why do system developers think that I would want to have *different* formats for date and time in my own computer/phone? Why do I have to see the annoying formats `MM/DD/YY` and `H:MM AM/PM` just because I want to use English as *language*?

> Why can't I in my computer/phone, once and for all, configure my formats for date and time, and then these formats are *always* used, regardless which site or application I am using? Why do I still have to configure user settings in each and every application? Why must I go into each and every application that I use, and configure the date format, the time format and the time zone? Why can't each and every application use the configured *computer/phone* settings? Why do system developers think that I would want to have *different* formats for date and time in my own computer/phone?

Note: The global solution Office 365 informs me with two "timestamps". The first, `5/28/2021 12:00:00 AM (UTC)`, is explicit and understandable (sadly it uses an irritating format though). The second, `5/27/2021 10:30:24 PM`, is really confusing. Is the second value also defined in UTC? Or is the second value defined in some other time zone and if so which time zone is used? How should I be able to understand the relationship between those two values?

Note: The widespread application Rundeck also has an interesting approach when listing "timestamps".
Looking at the first row in the list. In their left column they use `21-05-26 05:56`. Since no indicator of AM/PM is used, I assume that it denotes 05:56 in the morning. But in the right column they display `5:56 PM`, which make me confused. Is the `21-05-26 05:56` actually referring to `21-05-26 17:56`?
Looking at the last row, the left column displays the date as `21-05-21 04:28` and the right column displays `05/21/2021`. What is happening? Two crazy and annoying date formats are used at the same time and for the time value `04:28`, I have no idea if it actually refers to `16:28` or if it really is `04:28`, since now there is no AM/PM notation anywhere.

20 AM/PM notation

How was it now? Was it 12 AM that indicates midnight and 12 PM that indicates midday? Or the other way around? Why starting a new day with 12 which is the highest value in the sequence?

> As seen from members of the 24-hour world, the order 12,1,2,...11,12,1,2,...,11 as mapped onto 0,1,2...,23 is not only confusing, it is nearly impossible to make people believe that 13 hours have elapsed from 11 AM to 12 AM.

When using the 24-hour clock, times are usually written with exactly five characters, for example `09:45` or `22:20`. When using 12-hour clock, you need up to three more characters and the length varies, for example `9:45 AM` or `10:20 PM`. If you want to use a more correct notation you need two characters more, for example `9:45 a.m.` or `10:20 p.m.`.

Are there any benefits of using AM/PM? I can't come up with any. Not a single one. In my opinion it is only confusing to use 1-12 AM/PM notation instead of 0-23 to describe the hour of the day and there are no mitigating circumstances.

> Are there any benefits of using AM/PM? I can't come up with any. Not a single one. ”

> In the practice of medicine the 24-hour clock is generally used in documentation of care as it prevents any ambiguity as to when events occurred in a patient's medical history.
>
> The 24-hour clock is commonly used there [12-hour clock countries] only in some specialist areas (military, aviation, navigation, tourism, meteorology, astronomy, computing, logistics, emergency services, hospitals), where the ambiguities of the 12-hour notation are deemed too inconvenient, cumbersome, or dangerous.

When I read about AM/PM and 12-hour clock the text contains many words like *ambiguity* and *confusion*. So I rest my case.

> The loss of 12-hour clocks will annoy a few people for a time, but there is nothing quite like shaking a bad habit for good.

Note: Hilarious discussions about 12:00 PM and 12:00 AM.

Note: Perhaps writing AM/PM or am/pm is wrong. Perhaps the correct way should be to write a.m./p.m. or A.M./P.M. (at least in some language in some part of the world).

Note: I think that the biggest problem and most confusing part with AM/PM notation is that the sequence is numbered 12, 1, 2, ..., 10, 11 and not 0, 1, 2, ..., 10, 11. If using 0 instead of 12 there would be more logic in using AM/PM and it would conform more to the rest of the date and time values. The months are not numbered 12, 1, 2, ..., 10, 11 and the minutes or seconds are not numbered 60, 1, 2, ..., 58, 59. Yes, there is a subtle difference between the date values and the time values, since the months and days of month start at 1, while the hours, minutes and seconds start at 0.

Note: Another problem is that the AM/PM is often left out and should implicitly be understood by context. That is like leaving the month out and just present the day of the month. Saying "I see you on the 17th" may work, as well as saying "I see you 8:30", but there may be misunderstandings involved.

Note: Why are so many switching to AM/PM notation, just because they write texts in English? If you, for example, write "19:00" in a Swedish text and translate it to English, you *do not* have to write "7:00 pm". It may still be a Swedish audience your text addresses.

Note: Yes, there are also the 6-hour clock, the 30-hour clock and numerous other variants, but please, let us not dig deeper into those.

Note: One of the better usages of a 12-hour clock face is when used as clock position.

21 Languages

I really want to emphasize problems arising when you are trying to be smart and use a specific format when writing texts in a language. Whether it is date format or using AM/PM.

If you are writing a text in English, it is *really* hard for your audience to understand if it is written in American English, British English, Indian English, Australian English, etc. And if the readers don't know that, how should your readers be able to understand a date written like 06/07/08? How should they be able to understand if you are using the AM/PM notation or not, when writing a time like 05:00? Why leaving the interpretations to the readers, when you, as the author, easily can do it much better for them?

22 Problems with watches

You may think that you can always trust a good old watch, but lo and behold, you are in for a surprise.

> You may think that you can always trust a good old watch, but lo and behold, you are in for a surprise. ”

22.1 Mechanical watches

You may have invested in a manual mechanical watch that is precise and keeps the pace perfectly day after day. Unfortunately, you can *never* trust its value when you are travelling. If your clock is not connected to the Internet or GPS or if it lacks an embedded computer, you will *never* know how to adjust your clock. Your watch does not have access to the rules contained in the TZ database (or similar). To summarize, **you can tell everyone the exact duration of your trip, but you may not be able to tell anyone what the current time is**.

> **You can tell everyone the exact duration of your trip, but you may not be able to tell anyone what the current time is**. ”

Note: Yes, you can (hopefully, but not certainly) get correct local time from the wall-times in the airport, train station etc or ask someone or ask the Internet. But, you are dependent on something/someone else to know the current time and can not use your mechanical watch alone.

22.1.1 Problems for non-travellers

In fact, even if you are at the *exact* same spot all the time, you can not trust the value on your precious watch. Either the country you are in can change its time zone rules or there may be a Daylight Saving Time switch, so your watch suddenly will be an hour (or so) off. This will of course be true for every type of mechanical clock (including wall-clocks and perfectly paced Swiss-made ones). You have to remember, you can not hide from the disastrous DST demon.

> You have to remember, you can not hide from the disastrous DST demon. ”

22.2 Manual vs automatic watches

When you have a manual watch, you must set it correctly yourself. You also need to adjust the time manually when changing time zone or when DST switches occur. You are in control of the time you see, and you can trust the time value and corresponding time zone, since it will not change without your help. Even if you don't see any explicit time zone value, you can (almost) be sure that the watch is showing the correct time for the time zone you set it for.

Note: You can not be sure that the time is correct for your *location* though (as discussed in the previous section).

When you have an automatic watch, you may not need to set or adjust it yourself. The watch may get correct information from satellites, wifi, radio signals, etc. But if you don't see an explicit time zone value, you can not trust the time value at all. This is because you may have entered a different time zone, it may have been a DST switch, or the watch may (erroneously) have switched to another time zone. So in order to minimize the occasions you see an incorrect time value on your watch, you need to see the current time zone as well.

Note: Of course, the time zone must be expressed with offset to UTC, since an identifier or abbreviation is not enough, due to the fact that your automatic watch may have implemented an incorrect or outdated version of the time zone rules. As an example of the problems for automatic watches, think about the mornings you wake up after a DST switch. If you look at the time value on your watch, mobile phone or computer, you have no idea if it is the correct local time until you have double checked the actual time zone value.

Note: An example of problems encountered when your device will switch erroneously to the wrong time zone. No, it is no better with other brands, so please stop laughing.

> So when an iPhone says it’s 12:46 p.m., you expect that to be true.
>
> **Use another timekeeper.** This may be a frustrating solution, but it can keep you from missing important appointments. Use the in-room hotel clock, wear a watch to keep the time accurate, or buy a cheap and portable alarm clock to carry with you.

A wonderful example of a person thinking outside the box and taking advantage of the local times:

> I travel to India every month...They are 5.5 hours ahead of GMT....this is a great advantage when I'm there (when I've set my watch to Indian time) if I want to know the time in the UK, I just turn my watch upside-down.

23 Airports and other locations

"Why do airports manage to handle which time zone they are in?", you may ask. First, an airport always has its defined location. Second, the number of airports is a small finite number. So each airport around the world can easily be mapped to a specific time zone. Even if there should be a million airports (which there is not), it is a manageable number to handle.

It is the same with addresses. The set of addresses, although much larger than the set of airports, is also a finite set. Even if there are many billions of addresses around the world, you can map each address to a specific time zone. But now it is getting a bit harder to manage the mappings.

Note: This is the reason that it often is no problem choosing dates when you want to book a hotel or a flight. There are many hotels and airports around the world, but they all know their time zone.

The problem is how to handle all the locations *between* defined addresses. Since the time zones are no straight lines between the poles, you can not use your east-west position to map to your current time zone. So you need a mapping from your coordinates to a time zone. But how fine-grained should this coordinate system be so you *unambiguously* know what the local time is?

> But for most systems, the user will need to choose a time zone. There are always edge cases in which even very good external data cannot resolve the time zone accurately.

Note: The little island Märket is an interesting example of a country border that also serves as a time zone border. Think of the precision that a GPS watch must have to show correct local time (NNM Map Viewer). By walking *less than 200 meters* from west to east you are in the following standard time zones: UTC+01, UTC+02, UTC+01, UTC+02, UTC+01 and UTC+02.

Note: Another example is the three-country cairn between Norway, Finland and Russia (NNM Map Viewer). When walking around in this area your clock has to keep up with switching between UTC+01, UTC+02 and UTC+03. The marked circle in the map has a radius of ~400 meters.

You don't even have to cross a country border. If you are in the northwest areas of Florida or somewhere around El Paso in Texas, how would you know what (local) time it is?

24 Aviation

To avoid confusion and misunderstandings, the aviation uses UTC (Zulu) time.

Note: Some videos explaining Zulu time: Pilot's Guide to Zulu Time and Zulu Time.

> The pilot (and the ATC) use the UTC (Coordinated Universal Time). The flight's departure and arrival are in terms of the local times at the respective airports.
>
> Weather reports, flight planning, and flight clearance times are all calculated based on UTC to maintain smooth operations and minimize issues associated with daylight saving time or crossing time zones.
>
> Some years ago I had the actual unfortunate (or fortunate depending on perspective) experience of a flight that had a stopover prior to arrival at my location. All THREE times were for 2:00 AM local - but the FUN part was it was on the day where daylight savings time changed so no-one knew the actual UTC time because of everything being listed in local times. They just gave me a 4 hour window as best they could do.
>
> Many times a route will cross timezone so to avoid confusion about which timezone a time is in controllers and pilots will use zulu time by default. It also avoids daylight savings issues.
>
> As aircraft may be operating over several time zones during a cross country or international flight, it makes sense to file flight plans with departure and arrival times listed in Zulu time for easy reference.
>
> This is especially true in aviation, where "Zulu" is the universal standard. This ensures that all pilots, regardless of location, are using the same 24-hour clock, thus avoiding confusion when flying between time zones.

Note: So all the pilots in the world already know how to cope with the UTC on a daily basis.

25 A proposal for alteration

What if everyone around the world should use the same notation of time? This is the million dollar question. Yes, most parts/people in the world, would have to rethink and make a change. But that should just take a few weeks, and thereafter it should be a walk in the park for everyone using the same time everywhere.

Note: This is my proposal, which is *one* among many other proposals. I emphasize that it does not really matter what we use in the future, the important part is that we make a change to something better than the timekeeping we use today. Or as Absurd Minds put it in their song “Now we hear the call”: “Time is ticking by. Get started straight away. Or will you wait another day? Time is ticking by. The right time is today. Just don't care what others say!”.

Note: I will use London (due to the history of Greenwich and GMT/UTC) as the "starting point" of a new date in my examples, but I could, of course, have chosen any place on the Earth. Perhaps the "international date line" would be a better choice.

If all of us should use the UTC time zone when talking about time, it will be (almost) no change for the people living in London. The normal workday would still be 08-17 for them. For people in Tokyo, the normal workday would instead be something like 23-08. For people in Los Angeles, the normal workday would be something like 16-01. In Stockholm 07-16, Helsinki 06-15, and so forth.

Instead of you saying that you work between 08-17, which will be confusing for everyone outside your time zone, you may say that you work 23-08, which will be *unambiguous to everyone*. The people in your time zone will still understand you with no effort, since their working day is also 23-08.

An online meeting scheduled for `2020-04-13T20:00Z`, will start at `20:00` for *everyone*.

A six hours flight starting at `08:20` local time will *always* land at `14:20` local time. It does not matter if you are flying north, east, south or west. It does not even matter if your speed is 1500 km/h or 200 km/h. The clock will always be `14:20` wherever you land. And the other way around, a flight starting at `16:35` local time and landing `19:55` local time is *always* taking 3 hours and 20 minutes.

The date line will be no more, since the date always will be the *same* everywhere. There will not be any strange 24 hours leaps and you would not be able to travel back and forth in time anymore.

25.1 Dates - are they too long?

Nowadays, when we are using time zones, each date has a span of 48 hours. When January 1 starts in Wellington it will take almost 24 hours before Honolulu starts the same date. And each date, in every spot of the Earth, is 24 hours (the DST issues are ignored here for "simplicity"). So therefore each date spans 48 hours, since we during a period of 24+24 hours have a specific date somewhere around the globe.

But that is not completely true. At the moment, it is more like 50 hours. Since some countries (for example Kiribati) have come up with the (not so great) idea that they want to be first entering a new date instead of being last. So, the time zones around the world actually span from UTC+1400 to UTC-1200. When we start a new date, for example January 29, in UTC+1400, we have

```
2018-01-29T00:00+14:00
2018-01-28T22:00+12:00
2018-01-28T10:00+00:00
2018-01-27T22:00-12:00
```

When we fast-forward 24 hours, there are still some time zones that have not yet entered January 29

```
2018-01-30T00:00+14:00
2018-01-29T22:00+12:00
2018-01-29T10:00+00:00
2018-01-28T22:00-12:00
```

After another 24 hours it will still be January 29 in some time zones

```
2018-01-31T00:00+14:00
2018-01-30T22:00+12:00
2018-01-30T10:00+00:00
2018-01-29T22:00-12:00
```

Two hours later we have finally passed January 29 everywhere

```
2018-01-31T02:00+14:00
2018-01-31T00:00+12:00
2018-01-30T12:00+00:00
2018-01-30T00:00-12:00
```

Note: There are some mitigating circumstances for Kiribati if I understand it correctly. Before inventing UTC+1400 and UTC+1300, Kiribati used three time zones (UTC+1200, UTC-1100 and UTC-1000), since the International Date Line went straight across the country. As you see, these were not consecutive. Instead the time difference was 23 hours between some parts of the country. Nowadays the country uses UTC+1200, UTC+1300 and UTC+1400 instead, with a maximum time difference of two hours within the country. But it still is an increase in complexity for timekeeping in the world. And hopefully Kiribati will not start to use Daylight Saving Time.
But (yes, there is always a but), you have to understand what Kiribati instead could have done though. They could have solved their problem by changing the UTC+1200 to UTC-1200 and none of us should have had UTC+1300 and UTC+1400 thrown upon us.

The invention of UTC+1400 implies that for an hour or two in each 24-hour period there are *three* consecutive dates around the globe that are all valid at the same time. So for example, the following timestamps represent the *exact same* point on the timeline

```
2018-01-30T01:59:59+14:00 (Tuesday)
2018-01-29T23:59:59+12:00 (Monday)
2018-01-29T11:59:59+00:00 (Monday)
2018-01-28T23:59:59-12:00 (Sunday)
```

So if you would ask three random people, what date it was when they saw a specific celestial phenomenon, you *may* receive three different answers, and they would all be correct.

Note: The three answers could be something like this. Person in UTC+1300: "It happened just after midnight Tuesday morning". Person in UTC+1200: "No, it happened just before midnight Monday evening". Person in UTC-1200: "No, no, no, I am sure it happened just before midnight Sunday evening". You can view this in NNM Time zones.

Note: A real example really present 2019-07-20 is the confusion involved about when mankind first landed on the Moon. Eagle, the lunar module from Apollo 11, landed on Sunday July 20, 1969. That is correct for Europe, Africa and America, but for most of Asia and Australia the landing was at Monday July 21, 1969. The exact time in UTC for the landing was `1969-07-20T20:17:40Z`.
The time for the first step on the Moon, a few hours later, is also confusing. In UTC it was `1969-07-21T02:56:15Z`. This corresponds to July 21 for most of the world, but for USA it was still July 20.

Even if we should get rid of the strange UTC+1300 and UTC+1400 time zones, we still would have an interesting fact. Thanks to the UTC+1200 to UTC-1200 span, there *always* are at least two consecutive dates valid at the same time (since UTC+1200 and UTC-1200 *never* can be on the same date). The following timestamps represent the *exact same* point on the timeline

```
2018-01-29T23:59:59+12:00
2018-01-29T11:59:59+00:00
2018-01-28T23:59:59-12:00
```

And one second later they look like this

```
2018-01-30T00:00:00+12:00
2018-01-29T12:00:00+00:00
2018-01-29T00:00:00-12:00
```

Note: Another real example is that when talking about the *moment in time* when equinox takes place you may have to say: “September Equinox takes place on September 22 or 23 depending on your time zone”.

If we instead should use the same notation of time all over the world, a specific date should be valid for 24 hours, which may make more sense.

25.1.1 Birthday problems

You may think that a person that is born 2018-05-25 is older than a person born 2018-05-26. But that may not always be true. Quite often in fact, your assumption is wrong. The reason is that if the first person's birthplace is "west of" the second person's birthplace, this may happen. Of course, as you just learned above, a person born 2018-01-30 may be *older* than a person born 2018-01-28.

> You may think that a person that is born 2018-05-25 is older than a person born 2018-05-26. But that may not always be true. Quite often in fact, your assumption is wrong. ”

Note: The term "west of" should be interpreted as an UTC offset that is smaller. An offset -5 is "west of" -4 and +5. An offset +7 is "west of" +8 and +12.

25.1.1.1 How old are you?

Now an intellectual experiment. If birth dates and death dates are defined as local dates, what will be the answers to the questions below?

Four persons, A, B, C and D are born at the same moment. A and B are born in Samoa and C and D are born in American Samoa. The local birth date (and time) for A and B is `Sunday 2021-01-31 00:30` and for C and D it is `Friday 2021-01-29 23:30`. The reasons are of

course that Samoa has DST that time of the year and therefore uses UTC+14 and American Samoa "always" uses UTC-11.

Note: View surroundings in NNM Map viewer.

Note: Samoa will not use DST anymore (at least not for the moment).

Now A travels the short distance from Samoa to American Samoa and C travels from American Samoa to Samoa. So B and C are now in Samoa and A and D in American Samoa. They have all lived twenty hours, so local datetime in Samoa is `Sunday 2021-01-31 20:30` and in American Samoa it is `Saturday 2021-01-30 19:30`. All of a sudden they all die at the same moment.

What will the inscriptions on their gravestones be? Are the following correct assumptions? If not, what should the correct inscriptions be?

- Person A was born `2021-01-31` and died `2021-01-30`. A negative age?
- Person B was born `2021-01-31` and died `2021-01-31`. Not even a day old?
- Person C was born `2021-01-29` and died `2021-01-31`. Two days old?
- Person D was born `2021-01-29` and died `2021-01-30`. One day old?

Even though the four persons lived exactly twenty hours, from the gravestone inscriptions you may think different.

Note: You can elaborate yourself by using the NNM Time zones tool.

Note: If they instead had been born on two ships located in international water, just north of Samoa and American Samoa respectively, and A and C switch ships, would the inscriptions be different?

25.2 What about days, week, weekends and holidays?

My question is: do we still really need them?

25.2.1 Days

If we all should use the same time, how should we treat days? If London is the first place to enter a new day, what would that mean for Stockholm that lies ~23 hours after London? Should Stockholm also change the day/date at the same time as London? What about Wellington, that lies ~12 hours after London? If the "8 to 17" working day in Wellington should instead be 20:00-05:00, should the day/date change in the middle of a working day?

But do we really need the concept of day, when talking about times? Day, and also night, are just ambiguous definitions extremely local to every place on the Earth. If we still want to keep days, we just have to rethink that they don't always start at 00:00 and end at 23:59 in most places.

25.2.1.1 Ambiguous definitions related to location

It is the same with dawn, morning, noon, afternoon, dusk, evening, midnight, today, yesterday, tomorrow, etc. Those are strongly related to your location.

Another problem is of course that it may be confusing using the term midnight, since people may wonder which midnight you mean:

> If a date/time is referred to as "at midnight on Friday, October 20th" the intention could be either midnight the beginning of the day or midnight at the end of the day.

Timekeeping should not have anything to do with location specific observations. Timekeeping should be precise. Day, night, dawn, morning, noon, afternoon, dusk, evening, midnight, today, yesterday, tomorrow are ambiguous.

25.2.1.1.1 Today, yesterday and tomorrow

If your boss asks you to present all the major events that happened in your company *yesterday*, how will you solve your task? If your company is located in New York, would you use New York's time zone then? What if your company is spread all over the US? Would you make yesterday span 29 (or perhaps 30) hours instead of 24 hours then? What if your company is global? Would you use the 50 hours span then? If your boss is located in Tokyo and you are in New York, which time span would you then use in your search?

Note: When asked for events the last 24, 48 or 72 hours your task is understandable, because that would be exactly the same for New York, the US and globally. But when asked for events yesterday, today, previous week, previous month, etc, the task becomes ambiguous.

Using *today*, *yesterday* and *tomorrow* as time span identifiers in searches is confusing. Should a person in Tokyo get the same result as a person in New York? If the person in New York searches in the time span *yesterday* and then she travels to Los Angeles and performs the same search, should the result be the same?

Instead of these ambiguous searches, the *user* should have to define exactly what she looks for. It should be the user that defines the time span, not a product owner or system developer assuming/guessing what *today*, *yesterday* and *tomorrow* are supposed to mean.

Note: Yes, the form would be a bit more complex to fill in, but now your query would be understandable. If you live in New York and want to search for events that happened yesterday, you may have to define the time span as `2018-10-17T00:00:00-04:00` - `2018-10-18T00:00:00-04:00` (or better yet `2018-10-17T04:00:00Z` - `2018-10-18T04:00:00Z`), but *you* are in charge of your query. As a bonus, if you instead want the result for Los Angeles you just change your time span to `2018-10-17T07:00:00Z` - `2018-10-18T07:00:00Z` (and you do not have to travel to Los Angeles or change obscure time zone settings in your computer).

Note: As a specific example of a problem. Suppose you have a global service for searching among events. For "simplicity" the servers are running in UTC. One user is located in time zone UTC+07, another user in UTC and a third user in UTC-05.
At `2019-05-01T01:25Z` all three users are asked to get the events from *yesterday*. The UTC+07 user has a local time of `2019-05-01T08:25+07:00`, so she will get the events from 2019-04-30. The UTC user has a local time of `2019-05-01T01:25Z`, so she will also get the events from 2019-04-30. But the UTC-05 user has a local time of `2019-04-30T20:25-05:00`, so she will instead get the events from 2019-04-29.
Of course, the problems how exactly 2019-04-29 and 2019-04-30 should be interpreted in server time for the search query (as discussed above in this section) are also present.

25.2.1.1.1.1 Yesterday

When you try to use relative time in a smart way, you may get results that are really confusing. For example, if you list events and instead of using the actual timestamps you use texts like "1 hour ago", "5 hours ago", "2 days ago", "10 days ago". That is fine.

But, then you start using "yesterday" also. Sadly you make the erroneous assumption that yesterday corresponds to the values between "24 hours ago" to "48 hours ago" (or something completely different, it really does not matter). One problem though, is that for some people, one second ago may be *yesterday*, while for some other people, 24 hours ago may still be *today* (since there are many days that are at least 25 hours). The lesson learned is that you can *never* interpret "x hours ago" into "yesterday", "previous week", etc, without knowing the *exact current local time for the person reading the information*. Actually, and of course, you also need to know the *time zone (rules) for the current location of that person*.

Note: In Stack Overflow there are some interesting discussions: Consider the client's time zone when showing time under the questions, Wrong display of a question's "last active" property and See time in a different timezone than UTC. I think that the choice at Stack Overflow to always use UTC is a good choice, but I really think that the way they are displaying "yesterday" is only confusing and should be removed. The answer "This is by design" (to the second question) with the 24-48 hours explanations, is misleading and erroneous in my opinion.

25.2.1.1.2 Summer, autumn, winter and spring

If I say that I will go to Egypt next summer, do you know when that is? Or if I say that I will go to South Africa next summer? Or if I say that I will go to Kenya next summer?

For Egypt, you may assume, since you may know that I live in Sweden, that I mean the summer in the Northern Hemisphere. That *may* be a correct assumption, since Egypt also resides north of the Equator.

For South Africa it gets a bit more complicated. How can you know if I mean the Swedish summer or the summer in South Africa? Your assumption may now be completely wrong.

For Kenya it gets even more complicated and interesting. Can you talk about summer and winter in Kenya? Half the country lies above the Equator and the other half lies below. So what do I mean if I should say summer in Kenya?

Note: For which countries in Africa can you talk about summer and winter? For which countries do it make no sense to talk about summer and winter? The only thing I am certain of, is that it confuses me.

Note: If I, instead of saying that I will go next summer, say that I will go next January (or next June), you will directly understand what I mean, regardless of which country it is.

25.2.2 Week and weekends

Is the concept of Monday to Sunday, weekends normally being Saturday to Sunday and Monday to Friday being treated as "working days" still something we need? We have not been able to come to consensus if a week should start on Monday or Sunday (or some other day).

25.2.2.1 Previous, current and next week

Previous week, *current week* and *next week*. What does these mean? If it is Sunday May 19, is the previous week then Monday May 6 to Sunday May 12 for those of us that define a week to be Monday to Sunday? And for those of you defining a week as Sunday to Saturday, is previous week then instead Sunday May 12 to Saturday May 18? If you say that we should set up a meeting in next week, do you mean Monday May 20 to Sunday May 26 or Sunday May 26 to Saturday June 1?

Note: If it is Sunday May 19 in your time zone and Monday May 20 in my time zone, what will our interpretations of next week be? If I am a Monday-Sunday week person and you are a Sunday-Saturday week person, I will interpret next week as Monday May 27 to Sunday June 2 and you will interpret it as Sunday May 26 to Saturday June 1. But if I am Sunday-Saturday week person and you a Monday-Sunday week person, I will interpret next week as Sunday May 26 to Saturday June 1, while you interpret it as Monday May 20 to Sunday May 26.
If both of us are Monday-Sunday week persons I will interpret next week as Monday May 27 to Sunday June 2, while you interpret it as Monday May 20 to Sunday May 26.
It may be hard to find a day for a meeting under these circumstances.

25.2.3 Holidays

Holidays, for example Christmas, Easter and Midsummer are often celebrated due to different cultures and religions and what we celebrate around the globe varies a lot. If we still want to celebrate these, we can of course keep doing that. But must we really have them "built in" to our timekeeping system?

25.2.4 When should a new year be celebrated?

A new year should of course be entered when London goes from December 31 23:59:59 to January 1 00:00:00. The *big celebration* would be this second, but other celebrations around the globe could be adjusted to other times in the following 24 hours.

25.3 Counter-arguments from pessimists

There are many questions and counter-arguments, some of them quite naive, that come up from fossils when alternative timekeeping systems are proposed. Here are some of them.

25.3.1 Learning a new way of timekeeping will be impossible

Why? If the humanity has learned the current complex way of handling time, why should it be so hard to learn another more intuitive system?

Note: I truly think we (the humanity) haven't learned the current time handling system though, since there are so many misunderstandings.

> Despite the existence of time zones and strange daylight saving time regimes around the world, most people are blithely unaware of their own time zone and certainly of how it relates to standard references.

Note: There are already millions of pilots and flight controllers around the world that have learned and understand Zulu-time.

25.3.2 UTC abbreviation is not understandable to most users

I can assure you that abbreviations like PST, PDT, PT, EST, EDT, ECT, ET, CET, CEST, AST, AWST, AZOST, AZOT, YAKT, YEKT, EEST, BST, KALT, IDLW, NT, THA, ULAST, CXT, MIT, WAT, WIT, WET, WEST, WTF, ... are *way* more confusing to most users, compared to the UTC abbreviation.

Yes, the abbreviation for the time zone *you* live in, is of course trivial for *you* to understand, but for most of us it is far from trivial to understand *your* time zone abbreviation. Furthermore, *you* will probably have a really hard time understanding the majority of the other time zone abbreviations around the globe.

Note: A lot of people know that UTC corresponds to an offset of 00:00, but there are *very* few people that know the offsets for all the other abbreviations. And if you do not know the offset, the abbreviation is useless for you.

Note: If it is not enough with all the "official" abbreviations, you can always invent your own cryptic abbreviation like in the following quote. Where did MTN come from? What was wrong with using either MT/CT/PT/ET or MST/CST/PST/EST for the American time zones and not mix them?

> We've converted the 4 PM ET time to several different time zones including MTN, CT, PST, GMT, and CET.

Note: Many people do not know how to write a correct abbreviation either. Several times I have seen CEDT being used for Central Europe Daylight Time together with CEST being used, erroneously, for Central Europe Standard Time.

Note: Another thing is that many people fail to indicate if DST is active or not. For example for Sweden, many, many invitations and "timestamps" are written with CET, instead of CEST during DST. For those of us reading these messages, we can *never* be sure what time the writers really think they have

defined (since CET is a valid abbreviation all year round). This "one hour ambiguity" is so common to see all over the world.

Note: Using MT (or ET/PT/CT etc) instead of MST or MDT is really confusing in datetime values. MT gives no indication of what the actual offset is, without knowing exactly which location it relates to. Take for example the lovely American state Arizona. If I denote a datetime value as `2020-07-19 MT`, how should you unambiguously know if MT represents UTC-07 or UTC-06?

> The term Mountain Time (MT) is often used to denote the local time in areas observing either Mountain Daylight Time (MDT) or Mountain Standard Time (MST).
> In other words, in locations observing Daylight Saving Time (DST) during part of the year, Mountain Time is not static but switches between MDT and MST.

Note: One of the really large tech companies still thinks that this is a good way of informing a worldwide audience about an event time in the year 2022. I am still confused.

The goal is not to make it easy for *you* to understand a timestamp that *you* define. The goal is instead to make it easy for *everyone* else to understand a timestamp that *you* define. We also want to avoid the error-prone conversions between different time zones.

> The goal is not to make it easy for *you* to understand a timestamp that *you* define. The goal is instead to make it easy for *everyone* else to understand a timestamp that *you* define. ”

Note: You have to understand, that the burden should lie on the *single* person defining the timestamp, not on the *many* persons trying to interpret the timestamp. If the person defining the timestamp has to think an additional 10 seconds, that will still be negligible compared to if thousand persons have to think an additional second each. You do the maths.

25.3.3 UTC+00:00 time notation is not understandable to most people

My experience is that a timestamp defined in `UTC+00:00` is easier to understand for *most people*, compared to, for example, a timestamp defined in `UTC-08:00`, `UTC-05:00`, `UTC+03:00` or `UTC+05:45`.

Note: If you have followed along you understand that using offsets are often better than using identifiers or abbreviations. For example, with a timestamp defined as `2020-09-13 10:00 (+02:00)`, most people directly understand *when* that is. Compare that to using `2020-09-13 10:00 (Europe/Stockholm)` or `2020-09-13 10:00 (CEST)`, where many people need to quest the Internet in order to understand when exactly it is. And if you already are using the offset notation, it is a small adjustment for you to make it even easier for all others by using `2020-09-13 08:00 (+00:00)`.

25.3.4 The current timekeeping system has been working for many centuries

I agree that it worked "well" in the 19th and most of the 20th centuries. But since the computers entered the scene, the timekeeping system has not worked so good and ever since the Internet was turned on it has been way too much confusion involved. So no, I do not agree that the current timekeeping system is still working. It is time to step up and into the 21st century.

25.3.5 All clocks are built for the current way of handling time

Are they really? I think the clocks are either quite dumb or heavily dependent on the TZ database (or similar functionality) for showing the correct time. A clock works in a 24-hour environment today, and will most certainly continue to work. When we get rid of DST the clocks will work even better than today, since it will *always* be 24 hours per day.

Note: Always 24 hours is of course not completely true, we still have leap seconds and similar things to account for, but I think you get the essence of my answer.

25.3.6 How do I know when I have to go up in the morning when travelling?

If you come to a new place today, you have to inform yourself (in some way or another) about the correct local time, by using offset from UTC. In the future, you would have to inform yourself of the correct "working hours" instead, by using offset from UTC. It is no difference.

25.3.7 It is no problem understanding local times from context

If you have the slightest doubt about what the current context is, you will be completely lost in understanding the time value you see. In my experience (local) time values often lack information about context. It is way too much assumptions involved, and as you may know: do not assume, because it makes an ass of you and me.

25.3.8 It will be an enormous cost to change the system

What will the cost be if we keep the current system? If you look forward ten years, what will be the best solution? Which system is "cheaper" in the long run? I think that we will have a large initial cost if changing system, but in just a few years time break-even will be reached.

Note: Ten years from now we will look back and wonder why we did not change the system much earlier.

> Ten years from now we will look back and wonder why we did not change the system much earlier. ”

25.3.9 I am so accustomed to the current timekeeping system

This argument is also known by its similar friend: *I have gotten used to it by now*.

Well, that is not really an argument, just laziness. We were accustomed to a flat Earth, silent movies, cassette tapes, vinyl records, floppy disks, typewriters, horse-and-buggy and no telephones also, but we seem to manage to live with some new systems in those areas. Not so long ago, we did not even have the Internet.

Note: In Sweden we were accustomed to drive our cars on the left side of the roads, but we manage to drive on the *right* side nowadays.

Note: Just because the majority of us are accustomed to a solution, that can *never* be an argument that we should stick with that solution forever. If, for example science, come up with a better solution or discover new facts, we would truly be stupid if we do not take these discoveries into account when deciding paths into the future. Think about the old discussions of flat Earth versus spherical Earth or geocentric versus heliocentric. Should we have stuck with the flat Earth and geocentric beliefs that were mainstream?

25.3.10 I have always started working at eight o'clock

See the answer above. The "I-have-always"-persons have lost all their valid arguments. It was not the "I-have-always"-persons that invented the wheel or put their feet on the Moon.

Note: A perfect example of an "I-have-always"-person, with questionable argumentation and a wonderful concluding remark about programmers: “Programmers exist to make life more convenient for me; I don't exist to make life easier for programmers”. I love the stubbornness in the argumentation though and would love to see how the person manages, if extending the "global" business to China or Russia, with the same logical reasoning.

Note: If you can not cope with starting to work at some other time, you can ask your employer about still starting at eight o'clock. Keep in mind though, that you may look like an idiot.

25.3.11 It will never work

The "It-will-never-work"-persons are probably the same set of people as the "I-have-always"-persons, so the same answers apply.

25.3.12 How can I know if it is okay to call my friend on the other side of the globe?

If you today are living in Los Angeles and want to know what time it is in Tokyo, you ask the Internet something like "Current time in Tokyo" and get a result like "The current time in Tokyo is 2018-03-10 10:23". Your current time in Los Angeles will at that moment be something like "2018-03-09 17:23".

With the proposed system you will still ask the Internet, with a slightly modified question like "Time difference between Los Angeles and Tokyo", and get a result like "Tokyo is always 7 hours after Los Angeles". If it is morning in Los Angeles, you will know it is night in Tokyo, and if it is evening in Los Angeles it would be day in Tokyo.

Note: Instead of saying that Tokyo is 17 hours *before* Los Angeles (as we usually do with our current timekeeping system), we will say that Tokyo is 7 hours *after* Los Angeles. Since the clock is the same everywhere, we should inform about the smaller distance between the cities. The International Date Line would not exist anymore, and no location will be the first to enter a new date.
Another example is that instead of saying Honolulu (United States) is 22 hours *after* Auckland (New Zealand), we will say that Honolulu is 2 hours *before* Auckland.
Of course, in this context, "before" and "after" have nothing to do with local times. It is just a way of describing that the solar noon for a location is before/after the solar noon in another location. The (local) times will be the same at all places.

If you instead live in Stockholm and want to know about Tokyo you will get an answer like "Tokyo is always 8 hours before Stockholm". So if it is morning in Stockholm, you know it will be day in Tokyo, and if it is evening in Stockholm it is night in Tokyo.

As a bonus, when you know the normal working hours for Tokyo, you will always know if it is okay to call your friend in Tokyo. Just by looking at your watch. No matter where on Earth *you* are at the moment. The time you see on your watch would be the same time as in Tokyo.

Note: It is no difference, you still need to understand how a globe facing a light source works, in this particular case the rotating Earth orbiting around the Sun. As a bonus, after your first Internet-search, you do not have to bother the Internet again, since you now know the time difference to Tokyo. Most certainly your Internet requests will also be accompanied with informative tools similar to NNM Time zones, NNM Meeting time across time zones or Every Time Zone, so you can make a qualified guess if it is okay to call your friend.

25.3.12.1 Caller vs receiver

Another problem, already present today with the mobile phones, is that your friend from Tokyo may be somewhere else at the moment. So if she is in Europe it may be inappropriate to call her on the mobile, but you will have no clue about that.

When using mobile phones (or any other communication devices), the decision if it is okay or not should *always* lie on the receiver and not the caller. If the receiver does not want calls, she can switch off her phone. This is especially true when travelling around the world.

Note: The western parts of Spain and the eastern parts of Macedonia use the same time zone, despite the fact that the Sun rises and sets about two hours later in Spain. Would it be appropriate to make a call to a person in either of these places when the time is 07:00? You have two hours of "appropriateness span" that you must take into consideration and can not rely on the local time alone. In China, this "appropriateness span" would actually be at least five hours at the moment.

Note: I emphasize that I really think that the caller should always make her call (or send her email/sms/fax) whenever she wants. If the receiver does not want to be disturbed, the receiver will have to set her phone/computer to silence.
It is like calling a company. If the company does not want to answer my call, I end up in an answering machine informing me about when I can contact the company. I do not have to investigate in advance if it is okay to call the company at the current time.
If you call a phone connected to a building and you know that this phone is at a specific location where it is "appropriate" at the moment you must think again. The calls to the hard-wired phone *may* be forwarded to a mobile phone that can be anywhere in the world at the moment. Nowadays, as a caller, you have to take into account that when you call a (home/office/mobile) number, the actual receiver may not be where you think she is.
The *only* thing the caller is certain of, is that she wants to contact the receiver. All other aspects, for example if it is appropriate for the receiver, are assumptions and guessing.

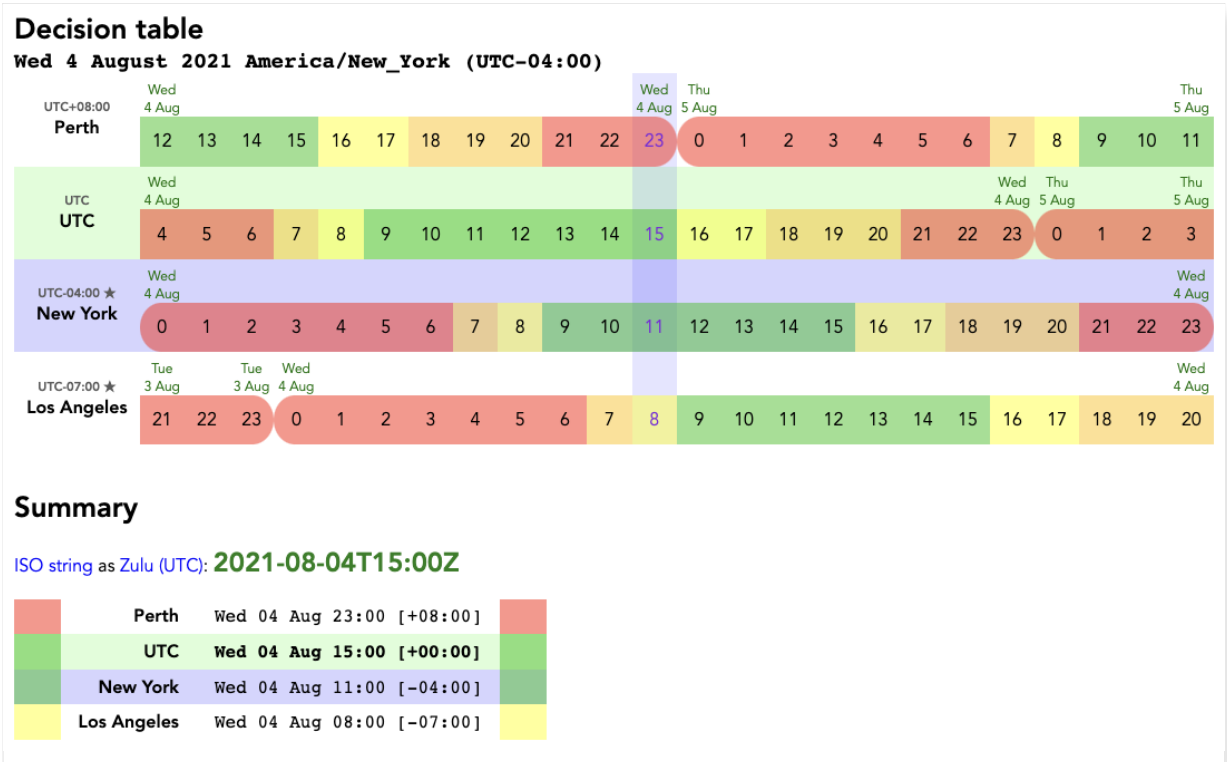
Note: Most importantly, deciding when it is appropriate to contact people should not be dependent on our timekeeping system.

25.3.13 How can I schedule meetings with participants around the globe?

The problems will still be the same when you need participants that are spread around the world. It will still be a nightmare (pun intended) to plan, since if you have participants from all of the 24 ideal time zones, there will *always* be in the middle of the night for some of them. You can not avoid that problem, no matter how you keep track of the time.

Note: One thing about scheduling people around the world that we can do better, regardless of which timekeeping system we use, is to not assume so much. We should not assume that people work between 08-17, just because that is the "normal" office hours. Instead we should be more precise.
As an example, Amy may be working 06-15, Bob may be working 07-16, Sara may be working 08-17, Tom may be working 09-18 and Lisa may be working 10-19. So when scheduling a meeting with these, you should not assume that Amy is available at 16:00 or that Lisa is available at 09:00, just because you think that is "normal". If you take these personal working times into considerations, you may in fact sometimes easier find a spot for scheduling a meeting.

Note: A helpful tool for scheduling world-wide meetings may be NNM Meeting time across time zones.



Example of using the *NNM Meeting time across time zones* tool for planning.

25.3.14 But then I will not know when it is light or dark

Whether it is light or dark at a location has nothing to do with what (local) time it is (as you learned above when reading about Kirkenes in Norway). And the ambiguous words "day" and "night" have *nothing* to do with light and dark.

> Whether it is light or dark at a location has nothing to do with what (local) time it is. ”

The Sun is not always up from 06:00 until 18:00. At the moment in Stockholm (latitude 59), at 15:30 a normal day in the middle of December, it is pitch-dark outside. Some places are completely dark for many days in a row, and some places are completely light many days in a row. There are few locations on Earth where the Sun sets and rises the same time every day.

Note: Now it is the end of May and in Stockholm it instead is light also at 22:00.

Note: It is still this rotating globe, with its tilted axis of rotation, that defines when it will be dark or light at a location. You can not use the local time alone (or the UTC time for that matter), you *must* also know your location on Earth (especially the latitude) and the time of year. If the local time is 23:45, it does not imply that it is dark where you are.

Furthermore, since the time zones are so stretched and misused in some countries, the solar noon (and therefore the "day" and "night") is far from 12:00 in many places.

Note: You must always inform yourself about the normal opening hours for stores, restaurants, businesses, governments, etc for your current location. You can not assume that stores will open 08:00 or that lunchtime for restaurants are 11-14. You have to understand and follow the local folkway.

Note: All of you saying that there is no problem with darkness in the mornings or evenings, please be aware of the fact that this may not be true for all the people in the world. Just because you are lucky enough to live in a place where it mostly is light when you are supposed to be awake, there are many people living in areas where this is not true. For many people there is a large difference in the "day" length in the summer compared to winter. You have to climb out from your tiny bubble and see the problem from other peoples perspective.

Note: Let us compare the maximum difference between day lengths in some locations during the year 2020.
In Nairobi, Kenya, near the Equator, the maximum difference between all the day lengths (12:03 (min) - 12:12 (max)) is *less than ten minutes*.
In Zagreb, Croatia, just above latitude 45, the maximum difference between all the day lengths (08:40 (min) - 15:44 (max)) is *more than seven hours*.
In Stockholm, Sweden, just below latitude 60, the maximum difference between all the day lengths (06:05 (min) - 18:37 (max)) is *more than 12 hours*.
In Kirkenes, Norway, just below latitude 70, the maximum difference between all the day lengths (00:00 (min) - 24:00 (max)) is *24 hours*.
On the northern hemisphere, most of the United States lie *below* latitude 45, while most of Europe, Russia and Canada lie *above* latitude 45. On the southern hemisphere, the vast majority of countries lie within the Equator and latitude 45.

Note: At the extremes, at the poles, the Sun rises once per year and sets once per year. The Sun is up for about six months and then disappears for about six months.

25.3.15 Our country uses the same time zone as our neighbour for economic reasons

If all the countries in the world use the same time zone the economy will be even better, if reasoning in the same way. So I rest my case.

25.3.16 But then I may have to work between 19 and 04

Some people may have to start their working "day" at one date and end it on the next date. But many millions of night-shift people already have this "problem" at the moment. So that will still be manageable.

25.3.17 How do I know if it is a working day in another country?

This is a good question. How should you be able to know if it is a "Monday to Friday" and that the people are normally working? Many of us are assuming that Monday to Friday is the working days and that Saturday and Sunday are the days off, as discussed above in the *Week and weekends* section.

In some way, each country must decide if they should be "before" or "after" the London day of week. This is *exactly* as it works today. Countries are before or after other countries in the "day of week cycle". But the reason that Japan currently is before Sweden in the "day of week cycle" is just random custom, it could easily have been the other way around. So yes, we still need a way to decide the "order" of the countries in some way.

Another thing about the working day issues is that there are numerous holidays spread around the world, for example national days, bank holidays, religious holidays, etc, that you have to keep up with. These special days are of course different from country to country, so this is

already a problem we have to deal with.

25.3.18 24 time zones around the Earth is perfect

If using time zones together with timekeeping is a good idea, why do we not use more of them? One of the reasons for time zones is that the solar noon should be near 12 o'clock everywhere. If we instead divide the Earth in 96 slices (15 minutes per zone), 360 slices (4 minutes per zone) or 1440 slices (1 minute per zone), every place on the Earth would have a local time that would be more accurate in relation to solar noon. If you don't think this would be an even better idea, why would you think 24 slices is a good idea in the first place?

> If using time zones together with timekeeping is a good idea, why do we not use more of them? ”

Note: If you argue that 96, 360 or 1440 slices would be impractical (for any reason), that would be the *exact* same argument I would use about 24 slices compared to *one* slice.

Note: The TZ database and our clocks will *not* have any (additional) problems handling 96, 360 or 1440 slices, so please exclude them from your arguments.

Note: The post So You Want Continuous Time Zones is a perfect example of time zone problems. Read the text, and think of "24 time zones" when reading "continuous" and think of "UTC/Zulu" when reading "discrete". You will see that the text is an excellent advocate for skipping also the current 24 time zones.

Note: One good thing with using 1440 slices would be that you could always say that the clock is 13:37 and you would always be correct.

25.3.19 Waking up in the morning at 2 PM will be silly

If using the same time everywhere, we *must* stop using AM and PM notations, since they per definition will be even more confusing than today. There will be no *ante meridiem*. Or *post meridiem*. At least not combined with timekeeping. The hour when you wake up is just an arbitrary number, and if it says 1, 2, 3, 5, 8, 13 or 21 should not matter at all.

25.3.20 We can not agree on if we should use permanent "winter" or "summer" time

I think the current time zones are way too wide. An ideal time zone spans 15° longitude. The Sun rises one hour earlier in the eastern parts than in the western parts of such a zone. So if a business in the eastern part has opening hours of 8-17, that would be like 7-16 in the western parts (and 7:30-16:30 in the central parts). Many, many discuss if we should have it dark in the morning or evening. The real discussions though, should be about why we still think it is a good idea that people living a few hundred meters apart, should have local times that sometimes are several hours apart.

Note: If using 96, 360 or 1440 slices, as discussed above, this would of course not be such a big problem.

Why not let the businesses, schools, etc use a more fine-grained system, say 10 minute slots? In the eastern parts they start at 8:00, then 8:10, 8:20, 8:30, 8:40, 8:50 and in the west they start at 9:00. You will have *no problem* understanding *your* local start times and no one far away care about when you start your *local* activities (it is your *global* activities others may care about). And there will not be any "one hour leaps" between close locations on different sides of a time zone border.

> No one far away care about when you start your *local* activities. ”

If you need to contact a person directly, perhaps by calling, in any of these locations, that is just like today. You need to look up the opening hours, with the help of the Internet. If the opening hours are 7:00-14:00, 7:20-14:20 or 15:50-22:50 should not matter to you, since you directly will know when that is.

Note: Of course, it would be even better for locations at high latitudes (yes, I mean latitudes and not altitudes), to actually shorten opening hours (and working hours) during the darker winter months. Once again, it is this tilted Earth, that make things difficult when you are too far from the equator.

Note: So far, no one favouring either side has come up with a convincing argument of how to handle the one hour leap of east and west side of a time zone border. So as long as we have one hour zones this will always be an issue.
Please, convince me about why a person living near a time zone border, when she moves hundred meters to the other side, her clock is suddenly best suited to be one hour earlier (or later).
This is one of the questions for which I really would like to see a credible answer.

> Please, convince me about why a person living near a time zone border, when she moves hundred meters to the other side, her clock is suddenly best suited to be one hour earlier (or later). ”

Note: In this article you can read that “We [...] demand a policy that supports optimal human health”. Okay, but why is standard time better than daylight saving time? Or vice versa? If we are aiming for optimal human health, how can we permit one hour leaps in local time? These one hour leaps are *exactly* like choosing between standard time and daylight saving time. Or am I missing something crucial?
If we *really* care about human health and if we obey that it is important that the solar noon and noon coincide, the correct solution is to get rid of these one hour leaps. Continuing with 24 time zones around the world will *not* solve this problem.

Note: When people discuss these issues, they often refer to the local times when the Sun rises or sets. But why do they refer to these changing values? For many places on the Earth, these local times vary quite a lot during a year (as you read above in the section *But then I will not know when it is light or dark*). If you want to discuss pros/cons for either side, why don't refer to the less fluctuating solar noon local time values?

25.3.20.1 Which time zone offset is best?

With our current solution for time zones, you can never have a valid argument that UTC+05:00 is better than UTC+06:00 for a given time zone. As I see it we can prove this with mathematical induction.

Imagine that someone is saying that location A is best suited to use UTC+05 and that location B is best suited to use UTC+06. If A is 15° west of B, this may be true and also understandable. The solar noon for A happens 60 minutes after solar noon for B.

But if A instead is 0.1° west of B? Then the solar noon for A happens 24 seconds after solar noon for B. Surely, A and B should be more suited to use the same offset in this case. If a specific offset is best for B, the best offset for A can not be 60 minutes off (or 59:36 minutes).

So the base case holds for location A and B that are 0.1° apart. Offset X is best for A and offset X is also best for B.

The induction step is to repeat all this. If you repeat this, you will *always* end up in the conclusion that if a person is *best* suited for a specific offset, then if that person moves 0.1° further west, the *best* offset (in one hour steps) must still be the same. You can not argue that an offset of "minus-one-hour" is *better* when moving 0.1° longitude.

I have not seen a valid counter argument that if a person living in A, if that same person instead lived in B, her local time should be changed 60 minutes, and that should still be the *best* solution.

If you argue that my base case does not hold, then I can safely rest my case about time zones, since then the same offset can *never* be the *best* solution for two locations, if we continue to use these one-hour-leaps.

25.3.20.1.1 There is a solution

The more I read about the "war" of which time zone that is best for a region, the more convinced I get that one solution is to add more time zones. Use 96, 360 or 1440 time zones around the world. Otherwise you can never argue that this or that is better, since there are such a difference between west and east side of a time zone.

Note: When I read this article, my conclusion is that we should have many, many more time zones in the world to cope with the SAD (Seasonal Affective Disorder) problems.

> Some time zones are fatter than others, and most are arbitrary, rarely related to state boundaries and often zigzagging along mountain ranges and rivers. The time zones do not represent the reality of the solar day, as it is anchored to sunrise at dawn.

Note: Imagine that we should only have *one* time zone for the whole world, and that we should still force people to do things at the same time. If we should say that schools start 08:00 and ends 15:00, that would be perfect for a few, but completely ridiculous for most of us.
This is exactly what we currently are doing within the ~24 time zones.

An even better solution would be to get rid of time zones and only use one, and then let regions decide locally for when things should happen.

25.3.20.2 The latitude problem

The extra problem that is present in high latitudes is hard to do anything about. Every location on Earth sees the Sun roughly the same amount of hours during a *year*. But the amount of hours during a *day* may be very, very different between locations and this is one of the reasons of why it is so hard to come to consensus of when schools should start and end.

Note: The poles see the Sun as much as a location on the Equator during a year. Half the time the Sun is visible and half the time it is hidden. The difference is that at the Equator the period of sunrise-sunset-sunrise is 24 hours, but at the poles the period is 12 months. Or in other words, at the Equator the Sun is up for 12 hours and then hidden for 12 hours, but at the poles the Sun is up for 6 months and then hidden for 6 months.

25.3.21 I want to choose myself when to do things

This is a stupid argument and has nothing to do with timekeeping. You can still choose to do whatever you want, whenever you want.

> People say, 'Oh, if we went universal time, that would mean we'd have be opening businesses when it's dark outside.' No, your business would go like it does now, with the sun. In New York or Baltimore, if you open normally at 9 a.m., that would be 14:00 [2 p.m.] on your watch.
>
> We do not make a single time because that would confuse people. In one place, people would go to school or work in the dark and the others in the light. It’s so much better to keep different times so that people have normal levels of light daily.

Note: Are some people really that simple-minded? Do they really think that we should stop deciding locally when to do things locally, just because we change our way of keeping time? I answer these questions myself: Sadly, those people truly exist.

Note: You have to understand that the time would be the same everywhere, but that does not imply that we must do everything at the same time.

25.3.22 Summary for counter-arguments

Yes, *you* have to rethink and relearn. Some parts may be a little harder to understand in the beginning, but a majority will always be a lot easier. Do *you* want to be a part of the problem, or a part of the solution?

> The point of a common time is not to force everyone to do everything at the same time. It's to allow us to communicate unambiguously with each other about when we are doing things.

Note: The quote above defines the core of my reasoning about a common time, so please read it once again carefully.

> The difference from today [if the whole world used a single GMT-based time] is that if you were putting together a London-LA conference call at 21:00 there'd be only one possible interpretation of the proposal. A flight that leaves New York at 14:00 and lands in Paris at 20:00 is a six-hour flight, with no need to keep track of time zones. If your appointment is in El Paso at 11:30 you don't need to remember that it's in a different time zone than the rest of Texas.

One of the best answers to the question "Regarding timezones, what are the pros and cons of just having one worldwide?" that I have stumbled upon so far:

> For the pros; The time locally will be the same everywhere. The military and the aviation industry operate with one time zone. They call it Zulu time and it is set as Greenwich mean. It really doesn’t make any difference what time it is locally because your daily life will run with these anyway. So if the sun rises at 1200 hours locally your day will start then. There is no reason why it has to be 8:00 AM at morning everywhere. The cons: it will take some psychological adjustment but that is easily accomplished.

26 Compilation of some peculiar quirks

I will compile lists for *some* of the strange things that have been discussed in the text. These are just a few of the many things that may have been jaw-dropping to you.

26.1 Time zones

- Time zone abbreviations are completely useless.
- Time zone identifiers are also useless (in their own).
- A date, like `2020-01-02`, represents a 50 hours interval on the timeline. If you define an identifier, like `2020-01-02 Europe/Stockholm`, the date represents a 24±1 hours interval (remember the disastrous DST). If you define an abbreviation, like `2020-01-02 CET`, the date represents a 24 hours interval (sadly, you can not be sure of exactly which interval though).
- There are about 40 different local datetime values that are valid at the same time. This also means that a datetime value like `2019-01-02 03:04:05` is representing about 40 different points on the timeline.
- Many local dates during a year span 23 or 25 hours.
- There are always at least two dates valid at the same time around the globe and sometimes there are three valid dates.
- Even if you are in the exact same location, your local time zone may change. This may happen even if daylight saving time is not used at your location.
- If your local time is 00:30, it may take 2, 3 or 4 hours until your local time is 03:30. It may even take 2.5 or 3.5 hours.
- Imagine your local time right now is five hours "before" the local time of your friend in another time zone. In less than an hour from now, your local time *may* instead be three, three and a half, four, four and a half, five and a half, six, six and a half or seven hours "before" your friend's local time. Did I hear DST craziness?
- There is no way that you can get your correct current local time from latitude and longitude alone.
- You can not from country alone, get your correct time zone.
- Noon on your clock and solar noon are sometimes many hours apart.
- Your current local time depends on your current location and the current time of the year.
- You can never trust a local time value that you see on your devices. It does not matter if it is a Swiss-made wall-clock, manual watch, digital watch, smart watch, computer or anything else. The time you see may not be the correct local time for your current location on Earth. These problems will show up especially when traveling, but they may also exist if you stay on the exact same spot.
- The time difference between cities in USA fluctuates during a year. This is true for Honolulu and Los Angeles, but also for New York and Los Angeles. Most often the time difference between New York and Los Angeles is three hours, but sometimes the difference is two or four hours.

26.2 AM/PM

- Well, everything is confusing when it comes to the 12-hour clock together with AM/PM (compared to the 24-hour clock), so just avoid using it.

26.3 Date formats

- When you see a date like `06/07/08`, you can *never* be sure what it denotes. It can denote June 7 2008, July 6 2008, July 8 2006 or August 7 2006. Or perhaps even June 8 2007 or August 6 2007. It can also denote June 7 1908, June 7 2108, July 6 1908, July 6 2108, July 8 1906, July 8 2106, August 7 1906, August 7 2106, June 8 1907, June 8 2107, August 6 1907 or August 6 2107. It can actually denote dates in any of the years ..., 1706, 1707, 1708, 1806, 1807, 1808, 1906, 1907, 1908, 2006, 2007, 2008, ...
- Date formats can not be deciphered from language alone, since date formats have nothing to do with languages. Date formats can not be deciphered from a country either, since a country can use many different date formats at once.

27 Conclusions/summary

Yes, the world **would**, in my humble opinion, be a better place if we stop using different time zones around the globe and instead start using the **same** time (without any time zone notation) all over the world. We should also get rid of the ambiguous 12-hour clock together with the superfluous AM/PM notation. Of course we should also use a common date format.

Complexity, **confusion**, **ambiguity**, **problems** and **errors** are some of the most common words you see when reading about time zones, 12-hour clock and different date formats. I can not remember that I ever have seen a set of rules with so many exceptions as when reading about time zones. Instead of struggling with these problems, why not get rid of them, once and for all?

> **Complexity**, **confusion**, **ambiguity**, **problems** and **errors** are some of the most common words you see when reading about time zones, 12-hour clock and different date formats. ”

To paraphrase myself: Timekeeping is science, time zones are nonsense.

To paraphrase Kaufmann and Freedman: Indeed, time zones are *not* a science at all, but merely a collection of superstitions and hokum that tries to use some of the terminology of timekeeping but which rejects the logical thinking that is at the heart of science.

27.1 Step-by-step solution

But wouldn't it be very difficult to do this? No, I *really* don't think so. In fact **I think it would be quite easy**. A step-by-step formula could be:

1. Daylight Saving Time is a complicated and confusing part when talking about time zones. The time difference between two locations may vary during a year. Getting rid of DST is a first step (that's one small step for a man, one giant leap for mankind). At this point, since we are not using DST anymore, the time difference between two locations is static.
2. Get rid of the ambiguous and useless time zone abbreviations.
3. Let us also throw away the AM/PM and 12-hour clock notations.
4. Start using the date format `yyyy-mm-dd` everywhere. At this point, we use the ISO 8601 formats for date, time and datetime values all around the globe.
5. Only use time zone identifiers to *identify* which UTC offset to use. For example, "Europe/Stockholm" will *always* map to UTC+0100 (remember that DST is not used anymore so it is a one-to-one mapping now). The *only* format you will see for a timestamp in Sweden will then be like `2020-01-02T10:20:30+01:00`. You will *never* have to see the insufficient time zone indicators like `CET`, `CEST` or `Europe/Stockholm` anymore in the timestamps.
6. Now, the step to only use UTC+0000 everywhere would be almost trivial.

Note: *All* these steps will be easy to accomplish, since the TZ database already has functionality that handles all this. Several times during the previous years, countries have decided to abandon DST or have decided to switch to other offsets. That is well-tried, everyday functionality for the TZ database.

Note: If there are systems that can not manage these changes, these systems are already broken since they have made erroneous assumptions about how to handle different time zones in the first place.

27.2 Unambiguous time

In *theory*, it is a nice idea with time zones. In *practice*, with the current solution, it is a real nightmare.

We have come so far nowadays, that when we talk about time, there should not be *any* confusion involved. There should not be *any* translation/conversion problems at all. When you see a time value, you should directly understand *exactly* when that is, without doing any calculations. Don't make me think!

The complexities and errors related to time zone problems in computer systems will be gone and the **immense cost** of maintaining these systems will be heavily reduced. There will not be any confusion about whether or not a time value should be interpreted as "local time", since all time values will be universal.

It is also a *very* interesting fact that when time values are really important not to mess up, such as with aviation, the UTC+0000 (Zulu time) is most often used, regardless of where on the Earth you are. No ambiguous offsets, time zones or local times are used, since we can not accept misunderstandings with catastrophic consequences. Why do we continue to accept misunderstandings in less critical areas?

> Why do we continue to accept misunderstandings in less critical areas? ”

If you should ask someone what time it is right now on a random place on Earth, I do not think that there is *anyone* that always can answer you correctly, without the help of some kind of computer. And even if they have help from a computer, they will most certainly be incorrect sometimes.

Mechanical watches will be useful again since you can trust them.

The pressure on Stack Overflow will probably go down with 50%, when people don't have to work with this madness.

The only "problem" remaining should be that some people would have to learn to go to work at another time of day, but for me that is not a problem, just an opportunity. Which complexities in life do you think that our coming generations should have to live with?

Perhaps no perfect solution exists, but we should at least strive for something better than we have today.

The Earth is global and worldwide. Why do we still use and handle time in the same way as we did in the 19th century? Why is the answer to the question "What time is it?" dependent on our *physical location* on the Earth? In fact, the answer to the question is **also** dependent on the *time of year*! What have we done to deserve this? How can our current system of timekeeping be more complicated than rocket science? Time zones do not improve or simplify our lives. The time is out of joint.

27.3 Moral of the story

> This is your world, these are your people. You can live for yourself today, or help build tomorrow for everyone.

Time zones must be destroyed. Q.E.D. And they all lived happily ever after...

> **Time zones must be destroyed.** ”

Note: If you have spotted *any* errors in my examples, calculations and conversions in the text, those are just additional arguments in proving my point about time zones (but of course I will be thankful if you inform me about the errors). Please, also inform me if you find any blurry reasoning, so it can be straightened out.

Note: The reason my text contains so many occurrences of the words "ambiguity", "complexity", "confusion" and "assumption" (and their similar friends), is because I want to emphasize that there is so much ambiguity, complexity, confusion and assumption involved when using time zones.

Note: If you think that I have used the term *time zone* for several different concepts (region, abbreviation, identifier, offset) in the text, you are probably right. Once again, it is to show how the term is used in the real world and I often get quite confused trying to understand what the authors actually mean.

Note: You could argue that the timestamp `2018-01-16T01:23:45+08:00` is not a *point* on the timeline, but merely an interval, since the precision is only seconds. But I think it is precise enough for this text. If you lower the precision even more you can of course argue that a date `2018-01-16` or even a year `2018` is also a timestamp, and I can agree with that.
Nevertheless, two important things to remember about timestamps are; (1) a time `01:23:45` without date (and offset) information can *never* be a timestamp, since it represents an infinite set of points on the timeline; (2) a time `2018-01-16T01:23:45` without offset information can *never* be a timestamp, since it represents several points on the timeline.

Note: My goal is to keep the content up-to-date, but as you now know, the time zone rules used for a location can be changed anytime and it is impossible to keep up with all changes. So after the different sections were written, the rules may have changed and the content may therefore be misleading. For example, at the moment Florida and North Korea have plans in changing their time zones. Of course, these out-of-date problems are also additional arguments in proving my point about time zones, so I am not the one crying.

Note: I want to emphasize, that I mean that time zones should be abolished when used together with timekeeping. *When you see a time value, it should be obvious to everyone when that is*. Of course, we will still need "time zones" in order to translate to "normal local working hours", to know when it is "normal" to wake up and go to sleep, to know when it is dark/light and other parts related to local custom and culture. The Earth will still rotate as long as we live here...

Note: The more I read about all these topics, and I have read *a lot* now, the more convinced I become that there must be better alternatives for them. The number of edge cases when working with date and time or time zones is infinite. For *every* time/date value I see, it would work as good using Zulu time and in many, many of these cases it would even work better with Zulu time. When it comes to different formats and AM/PM it is *always* worse to use anything else than ISO 8601.

Note: Since you have read this far, now it is time for you to visualize me dropping the mic...

28 Something similar

28.1 Time zones

To get a grasp of how hard and complex time zone conversions are, we can make a comparison with the problems encountered converting between metric and imperial units.

28.1.1 The assignment

Say, for example, that we occasionally must convert between meters (`m`) and yards (`yd`). 1 `yd` equals 0.9144 `m`. This conversion will *always* be the same, since the relation is *always* 1 vs 0.9144. Though this conversion is rather annoying to have to do, it is still easy.

28.1.1.1 The intellectual experiment

Now consider, that a country change the relation between `yd` and `m`. So, for example, from April 4, the relation should instead be 1 vs 0.9182 and October 27 the country change the relation back to 1 vs 0.9144.

Further consider, that *different* countries change the relation on *different* dates (some countries will not change the relation at all). Furthermore, *different* countries change the relation *differently*, defined by complex rules. Some country will change the relation to 1 vs 0.9167 and some other will change to 1 vs 0.9194.

It is getting hard to follow, but we still have some more complexity to add.

Now imagine that a country change the relation between *different* areas *within* the country. For example, the United States change the relation to 1 vs 0.9156 in some state and to 1 vs 0.9178 in another state. To make it a bit more complicated, we also have to imagine that *within* some states the relation will not be the same everywhere.

Furthermore, the dates that the relation changes will occur in every place of the world, will of course be *different* from year to year. So the country changing the relation April 4 and October 27 will use some completely different dates the next year.

Now it is time for you to do the simple math. On a random day of the year, convert 250 meters defined in Stockholm (Sweden) to the correct number of yards in New York (United States). Now you are *starting* to get a grasp of the complexity with time zones and our current timekeeping system.

28.1.2 Another view

If you should leave out the unit when measuring weights, lengths, currencies or temperatures, you should probably be kicked out from your work. A weight of 34, a length of 19, a monetary amount of 750 or a temperature of 5 are not understandable at all. Compare it to 34 kg, 19 inches, USD 750 and 5° Fahrenheit. Or 34 pounds, 19 cm, SEK 750 and 5° Celsius. Imagine a house-builder not using units on the length values, or a chemist not explicitly communicate temperature units.

But when it comes to defining times, we accept that people leave the unit. The unit in this case is of course the time zone part of the time value. But it is *quite a difference* between the following time values:

```
2017-12-18 19:20:21
2017-12-18 19:20:21 Europe/Stockholm
2017-12-18 19:20:21 Europe/London
2017-12-18 19:20:21 EET
2017-12-18 19:20:21 AWST
2017-12-18 19:20:21 UTC-08:00
2017-12-18 19:20:21 UTC+03:00
```

So just writing `2017-12-18 19:20:21` for a timestamp will *never* be enough. And of course, writing `2017-12-18 19:20:21 (local time)` is as bad, as it will be ambiguous and force the reader to investigate further. You would not write "The box weighs 34 (local unit)" or "The price is 50 (local currency)", would you?

28.1.2.1 Missing informing about DST

Using the abbreviation CET instead of CEST when DST is active in Sweden, is as bad as mixing different units. For example, if you define a timestamp as `2019-04-25 13:30 CET`, when you instead actually mean `2019-04-25 13:30 CEST`, is like saying 57 yards instead of 57 meters. Or saying 32° Celsius instead of 32° Fahrenheit. You *will* appear as a fool.

> If you truly want to remain clueless, it's fine to say "ET" or "eastern time" (likewise for CT, MT, and PT); do so, and you're just being imprecise, not wrong.

Note: The above quote is from the article 'EST' and 'EDT' Are Different Time Zones. I do not agree that it is just imprecise, and not wrong. Saying "ET" is like leaving the unit altogether. The length is 57. Is it meter or yards? The temperature is 32°. Is it Celsius or Fahrenheit?
From my point of view, it is at least as bad to use ET, when you actually mean either EST or EDT. But as you hopefully understand by now, using time zone abbreviations is always really stupid.
The comment section of the mentioned article is really "fun" (or should I say depressing) to read. One of the comments says: "I write tech support docs, and always use ET, CT, etc. I hate seeing "EST" in a static document. Don't they know it's *wrong* for half the year?". Does the person commenting know that always using ET is *wrong* the whole year?

28.1.2.2 How many datetime units are there?

Many of us understand that there are several different units that can be used for weights, lengths, currencies and temperatures. But not so many understand that for datetime values there are many more (currencies may have more though). Perhaps you should look at the different time zones as metric prefixes for the length unit meter. So meter, kilometer, millimeter, etc would be "similar" to time zones offsets like +00:00, -01:00, +01:00, etc. The thing you must know though, is that there are about 40 different time zone units used at the moment and if you are not explicit which of those you mean, everyone reading your values will be confused.

28.1.3 Yet another view

Many also understand that currencies are dynamic values and that the relationship between different currencies varies over time. But when it comes to local time in different time zones, there are not so many that understand that the relationship is dynamic.

The relationship between local times is similar to the relationship between currencies. *1* USD may be worth *10* SEK, but a month later *1* USD may instead be worth *9* SEK. The local time `09:00 America/New_York` may be "equal" to `15:00 Europe/Stockholm`, but a month later `09:00 America/New_York` may instead be "equal" to `14:00 Europe/Stockholm`.

28.2 AM and PM

Is using AM and PM a good idea? Let us think about it.

Using AM and PM is like dividing each day in two equal parts. Instead of using the markers AM and PM, we could instead use 1/2 and 2/2 respectively, denoting first half and second half. It would be the exact same thing. This results in the following table, where first column contains some random time values in 24-hour notation and where the values in each row represent the same time of day in different notations.

```
00:20     12:20 AM     12:20 1/2
01:55      1:55 AM      1:55 1/2
06:20      6:20 AM      6:20 1/2
11:00     11:00 AM     11:00 1/2
12:05     12:05 PM     12:05 2/2
15:00      3:00 PM      3:00 2/2
19:25      7:25 PM      7:25 2/2
23:10     11:10 PM     11:10 2/2
```

If dividing the day is a good idea, let us try to divide it even further, since that would be even better. We divide in four, i.e. we use the markers 1/4, 2/4, 3/4 and 4/4. This mean that each part is six hours (instead of 12 hours). Of course, we will use the sequence 6, 1, 2, 3, 4 and 5 to denote the hours, so we follow the same pattern as for the familiar 12-hour clock.

```
00:20     12:20 AM     12:20 1/2      6:20 1/4
01:55      1:55 AM      1:55 1/2      1:55 1/4
06:20      6:20 AM      6:20 1/2      6:20 2/4
11:00     11:00 AM     11:00 1/2      5:00 2/4
12:05     12:05 PM     12:05 2/2      6:05 3/4
15:00      3:00 PM      3:00 2/2      3:00 3/4
19:25      7:25 PM      7:25 2/2      1:25 4/4
23:10     11:10 PM     11:10 2/2      5:10 4/4
```

So if you now say "Call me tomorrow at three o'clock", I would be even more confused about when that is. Do you still think AM and PM together with the 12-hour clock is a good idea compared to the 24-hour clock?

28.2.1 Dividing the year

Another example could be to denote our 12 months with 1-6 FH and 1-6 LH instead of 1-12. FH stands for first half and LH stands for last half. Or perhaps even more similar: 6 FH, 1 FH, 2 FH, 3 FH, 4 FH, 5 FH, 6 LH, 1 LH, 2 LH, 3 LH, 4 LH and finally 5 LH.

Note: Instead of FH and LH we could use AM and PM here also, standing for "ante Midsummer" and "post Midsummer".

So January 14 would instead be defined as 6 FH 14 and December 24 as 5 LH 24.

Note: Yes, I know that 12 in a 12-hour clock actually stands for zero (the first hour), while in the month example the first month (January) is denoted with 1 (and not 0), which means that all the following months are renumbered. But, I hope you see the big picture.

28.3 Date formats

Is using `mm/dd/yy` or `dd/mm/yy` as date formats a good idea? Let us think about it.

Using the *date* formats `mm/dd/yy` or `dd/mm/yy` would be like using the *time* formats `mm:ss:hh` or `ss:mm:hh` (instead of `hh:mm:ss`).

> Using the *date* formats `mm/dd/yy` or `dd/mm/yy` would be like using the *time* formats `mm:ss:hh` or `ss:mm:hh`. ”

The following table contains rows that represent the exact same time values in different time formats. The rows are *sorted increasingly*.

```
hh:mm:ss    mm:ss:hh    ss:mm:hh
01:23:45    23:45:01    45:23:01
05:30:00    30:00:05    00:30:05
14:15:16    15:16:14    16:15:14
19:59:59    59:59:19    59:59:19
20:00:00    00:00:20    00:00:20
20:00:01    00:01:20    01:00:20
23:45:01    45:01:23    01:45:23
```

Note: You would probably have a hard time proving your sanity if you should start to use the `mm:ss:hh` or `ss:mm:hh` formats.

Another way of thinking about it could be that you use another format for writing floating point numbers. You can for instance write the decimal part before the integer part. So if I would say that a bag weighs 17.54 kg, you could instead say that it weighs 54.17 kg and we would both mean the same thing.

Do you still think that other date formats than `yyyy-mm-dd` is a good idea?

28.3.1 Missing vital information

Writing a date, without specifying the format used, is like leaving the unit (as discussed above). The unit part in this case is the format.

If you write dates like `10/11/12` or `23/02/24`, how should anybody understand what you mean?

If you instead add the "unit", everything will be easily understandable, `10/11/12 (yy/mm/dd)` or `23/02/24 (dd/mm/yy)`. Without the "unit" information, the date is completely useless.

Note: I would argue that it is not really a date you have written, if only writing "Best before `23/02/24`". It is just three random numbers to me (or eight random characters). This is also the same as writing "Max weight: 1200" in the elevator, where you completely failed in writing a weight. Or writing "It is freezing cold outside, just 15 degrees", where the readers will not understand anything about the actual temperature.

Note: The only format you can understand implicitly is `yyyy-mm-dd`, since I still haven't seen anyone crazy enough to use `yyyy-dd-mm` for writing dates.

29 Quotes

Quotes from me and quotes from people all over the world. The emphasized quotes are my own, while the additional quotes are from others.

29.1 Emphasized quotes from text

Here is a collection of all the emphasized quotes that you have in the text.

> Even answering a simple question like "What time is it?" is sometimes *impossible*.

> *All* the concepts in this text should be common knowledge and minimum requirements for everyone that claims to understand the clock and the calendar.

> *How can you **unambiguously** know if the time you see on your device is the correct time for the location on Earth where you are at the moment?*

> Imagine you are placed in a trunk. You have a watch on your arm. The car is travelling around for many hours, and you have no idea in which directions. You manage to escape. The question now is: Do you know what time it is?

> Think about that again. You have to know direction, speed and daylight saving time rules, in order to know when you will land.

> Going in the *same* direction could in fact "travel" you back *or* forth in time, depending on where on Earth you are.

> *Most* people do not have a clue about what the abbreviations stand for and where on Earth they are used.

> You may live in the exact same *location* all year, but you may not live in the same *time zone* all year.

> **What a blessing it would be if all EU countries should choose to use the UTC time zone. All year round. That would truly be pioneering.**

> Some people in south of Sweden and in Denmark are afraid that Denmark and Sweden will choose different time zones and that it will be misunderstandings and confusion with train tables and working hours for those that commute. But wait, isn't that already a "problem" in the north of Sweden and Finland? And around each and every time zone border around the world? In fact, isn't that one of the main problems (and main purposes) with time zones? If it is a real problem with different local times, shouldn't we get rid of all time zones then?

> Perhaps DST is the most dreadful invention in our current timekeeping system.

> How the people in Australia know what the time difference is between two random places *in their own country* on a random date is an unsolved mystery to me.

> Yes, it is also a mystery to me how the people in the United States of America know what the time difference is between two random places *in their own*

country on a random date.

How can it be that we think we know what the local time is, when we don't have a clue about what time zone we are in? ”

Many people do not know how important it is to define a correct time zone when they are defining a timestamp. ”

The best part of the last sentence is that you have no idea if I meant June 7 or July 6. ”

If you find a can marked "Best before 09/10/22" would you dare to eat its content in the year 2020? ”

Language and date format should be kept completely separated. ”

You would not come up with the idea to force strange length units onto users, when they select language, would you? ”

Why can't I in my computer/phone, once and for all, configure my formats for date and time, and then these formats are *always* used, regardless which site or application I am using? Why do I still have to configure user settings in each and every application? Why must I go into each and every application that I use, and configure the date format, the time format and the time zone? Why can't each and every application use the configured *computer/phone* settings? Why do system developers think that I would want to have *different* formats for date and time in my own computer/phone? ”

Are there any benefits of using AM/PM? I can't come up with any. Not a single one. ”

You may think that you can always trust a good old watch, but lo and behold, you are in for a surprise. ”

You can tell everyone the exact duration of your trip, but you may not be able to tell anyone what the current time is. ”

You have to remember, you can not hide from the disastrous DST demon. ”

You may think that a person that is born 2018-05-25 is older than a person born 2018-05-26. But that may not always be true. Quite often in fact, your assumption is wrong. ”

The goal is not to make it easy for *you* to understand a timestamp that *you* define. The goal is instead to make it easy for *everyone* else to understand a timestamp that *you* define. ”

Ten years from now we will look back and wonder why we did not change the system much earlier. ”

Whether it is light or dark at a location has nothing to do with what (local) time it is. ”

If using time zones together with timekeeping is a good idea, why do we not use more of them? ”

> No one far away care about when you start your *local* activities. ”

> Please, convince me about why a person living near a time zone border, when she moves hundred meters to the other side, her clock is suddenly best suited to be one hour earlier (or later). ”

> **Complexity**, **confusion**, **ambiguity**, **problems** and **errors** are some of the most common words you see when reading about time zones, 12-hour clock and different date formats. ”

> Why do we continue to accept misunderstandings in less critical areas? ”

> **Time zones must be destroyed**. ”

> Using the *date* formats `mm/dd/yy` or `dd/mm/yy` would be like using the *time* formats `mm:ss:hh` or `ss:mm:hh`. ”

29.2 Additional quotes

The following quotes enhance the understanding of both pros and cons and remember that these quotes are just a small part of all discussions going on.

All the credits go to the original authors for all the quotes in this text, both in this section and everywhere else. Many thanks to all of you for your valuable contributions.

> Working with time-related data can be complex because values are related to calendars and timekeeping rules, which themselves may be somewhat arcane. One of these complexities in working with time-related data is the effect of time zone on the data.
>
> Computer systems tell time differently than people do. So it is helpful to understand how time works within computers as well as in the real world in order to get a handle on the things that can go wrong.
>
> Date and time values based on incremental time are time-zone-independent, since at any given moment it is the same time in UTC everywhere.
>
> Since floating time values are often dates without any associated hours, minutes, or seconds, the resulting incremental time for these fields is often set to zero, exacerbating the problem: all time zones west of the prime meridian will consider a floating time to be the previous day.
>
> Content and query authors are warned that comparing or processing dateTime values with and without time offsets may produce odd results and such processing should be avoided whenever possible. Generating content that omits zone offset information (where it exists) is a recipe for errors later.
>
> It is good practice to use an explicit zone offset wherever possible. If one is not available, best practice is to use UTC as the implicit zone offset for conversions of this nature.
>
> When creating content, use UTC for your time values whenever possible so that values from discrete sources can be compared more readily.
>
> If no UTC relation information is given with a time representation, the time is assumed to be in local time. While it *may* be safe to assume local time when communicating in the same time zone, it is ambiguous when used in communicating across different time zones. Even within a single geographic time zone, some local times will be ambiguous if the region observes daylight saving time.
>
> A lot of technology today seems to ignore the complexities of world time, timezones and daylight savings.
>
> There are many horror storries (sic!) of wasted time and money, hard to find bugs and “but it works on my machine”-type situations that could have been avoided if the programmers simply stopped asking the server for its “local time”.
>
> Broken Hill is in New South Wales, but uses Australian Central Standard Time (ACST) which is 30 minutes behind the rest of the state.
> Eucla in Western Australia is 45 minutes ahead of the rest of the state. It has its own time zone called Australian Central Western Standard Time (ACWST).
>
> Lawmakers in Brazil have decided to move the start of Daylight Saving Time (DST) from 2018, effectively shortening the DST-period in the country by two weeks.
>
> If you live in a country with Daylight Saving Time, your computers are (hopefully) changing time on their own twice a year. This is fine for your personal devices but a server with time-stamped log and/or backup files may run into issues twice a year during the switch.

A time zone is a region on Earth that has a uniform, legally mandated standard time. As legal definitions of zones can vary wildly and change often, a database or lookup table is often required to properly apply time zone rules.

UTC, also called Zulu time, is used everywhere on Earth by astronomers and others who need to state the time of an event unambiguously.

Daylight Saving Time is the period when a time zone shifts its time forward in the spring (usually by one hour) and then shifts back in the fall. This creates many programming issues dealing with the lost hour in the spring, and the extra hour in the fall.

This can lead to problems because some people think that you can always convert in either direction [between UTC and Local Time], which is false for any time zone that observes daylight saving time.

Governments of the world often make changes to how they want to follow DST. You cannot assume that the current rules have always applied in the past or always will apply in the future.

They [time zones] were a good idea at the time [19th century], but in the modern world they cause more trouble than they are worth.

It is genuinely annoying to schedule meetings, calls, and other arrangements across time zones.

Today, however, we are very accustomed to the idea that time zone boundaries should be bent for the sake of convenience and practicality. That means we should move to the most convenient and most practical time system of all — a single Earth Time for all of humanity.

It's not a safety issue. It's a coordination issue. And universal time is already spontaneously being adopted from the ground up for lots of practical reasons. And when the railways and the pilots demanded new time zones, they didn't ask anyone. It was a necessity.

The transition is easy to make, countries have gone metric, and that's a much bigger switch than universal time.

From an astronomical point of view, the time is the same everywhere in the world right now.

The American Journal of Cardiology found a 10 percent increase in heart attacks on the Monday and Tuesday after the March weekend shift to DST. The Journal of Applied Psychology found an increase in the number and severity of injuries suffered by miners after the annual switch. And Accident Analysis & Prevention predicts a 13 percent reduction in pedestrian fatalities and 3 percent reduction in vehicular fatalities if the U.S. would stop resetting its clocks twice a year.

It’s amazing how many meetings get derailed because the times or dates were miscommunicated. For example, when writing a date, 4/9/18 could either be April 9 or September 4. Avoid potential misunderstandings by writing out the full date: April 9, 2018. Also ensure everyone knows what time the meeting is in their local time zone. This gets even trickier around daylight savings time, where different countries change at different times! To avoid any confusion, when sending meeting invites, include all the relevant time zones and be specific (e.g., 7 a.m. NY time, 1 p.m. (13:00) Paris time).

More power to Arizona for protesting one of the most annoying ideas humankind has ever come up with. [...] Not all of our choices can be as great as the choice to ignore daylight saving time, evidently.

In January, 2015, a lawmaker proposed establishing daylight saving time. It was met with shock and outrage.

Local solar time was fine when almost all activity was local. Today, much activity is global and one time is called for.

Once you see this inevitable correspondence between the whole circle and the whole day, you’re liable to get confused by some of the current information design out there, when it’s based on the 12 hour dial.

Our system of time zones defies geographic reality, and means that natural time and civil time don’t always coincide.

You may miss an important obligation, or connections with scheduled transport, simply by not understanding what will be the correct local time as you travel.

Starting a 12–15 hour flight from the U.S. west coast to Japan or Hong Kong in late evening can land you there in the morning **two** calendar days later.
If starting the reverse course by midday, you may in a way "travel back in time" as you land earlier than you started. For example a typical flight from Sydney to LA will take off at lunchtime and land early in the morning on the same calendar date!
If your travel has time zone complexities or possible impacts on your health or comfort, consult an expert as you plan it.

Some time-zones seem to defy logic and were mostly drawn by national or regional governments to make commerce and administration easier.

As there is no universally agreed point of the year to change from standard to daylight saving time, there may also be "fluctuations" of up to several weeks when one country has already changed and the other hasn't. If you are traveling during that time or calling home, make sure to inform yourself of the local time at both your destination and point of origin.

Going west from France lets you stay in the same time zone (when you "should" have to change from Central European time to UTC) but going north from France to Britain you will have to change time-zone.

I am not about to stay at work until 11pm every day.

I do business globally. I know if I call my Japanese customers, it is tomorrow there and 13 hours ahead of my local time. If it was 24:00 here and I was ready to leave work, it would be 24:00 there and who the hell knows what part of the day.
Nope, not gonna do it. I know the sun rises between 6 and 7 in the a.m, noon is when the sun is highest in the sky, and the sun sets around 6 to 7 in the pm (roughly). That makes midnight sorta the middle of the night.
This is true no matter where I am on the globe, within reason and with adjustments for sunrise & sunset times due to latitude.

Programmers exist to make life more convenient for me; I don't exist to make life easier for programmers.

Due to daylight saving time, there is a possibility that a time either does not exist, or has existed twice.

Time zone detection is rather tricky to get right.

You *usually* don't need to worry about time zones. Your code runs on a computer with a particular time zone and everything will work consistently in that zone without you doing anything. It's when you want to do complicated stuff *across* zones that you have to think about it.

Time zones are pain in the ass.

Using the time zones can be confusing for several reasons. First, some countries have several different time zones, and those local zones don't always align perfectly with specific longitude lines. Also, remember that with 24 time zones the local time at another location can be anywhere from 0 to 24 hours different from you, and it can even be a different date!

Remember, that Time Zones, Offsets and Daylight Savings rules are not fixed and change (a lot more than you'd expect!).

When testing your applications make sure you test countries in each hemisphere, with both DST on and off. Test also transition from Summer Time to Winter Time and boundary cases (UTC+13) and half-hour time zones, at least.

While I think Daylight Savings is useless, unfortunately we still have to make sure that our applications work correctly until the rest of the world discovers that it's useless.

Little did I know at the time, I booked every appointment for eastern time (Florida) when I live in the central time zone (Texas). Every single meeting was scheduled for the wrong time.
I was in time zone hell.

If you left the Gilbert Islands [...] on a boat and arrived at Baker Island or Howland Island, your watch would still be showing the correct time, but if it has a date display, the date would be one day off.

Time zones are thus a compromise, relaxing the complex geographic dependence while still allowing local time to approximate the mean solar time.

The increase in worldwide communication has further increased the need for interacting parties to communicate mutually comprehensible time references to one another.

Thus from west to east, time zones were: +6:00, +5:30, +5:45, +5:30, +8:00, +6:00, +5:30 and +6:30.

Pentagon officials were two hours late for an important military conference in Alaska because no one knew what time it was out there on the Russian border.

Norway, Finland, and western Russia observe three different time zones. As a result, this border region, near the Russian village of Nautsi, can get chronologically crazy.

So they could observe the arrival of the new millennium not one, not two, but three separate times.

Welcome to the world of standard time, where the only international rule is keeping your clocks in sync with UTC.

Everything else is open season. Everyone chooses their own standard time offset; everyone chooses when to switch to daylight saving time (or even whether or not to use it at all); everyone chooses their own time zone boundaries.

Daylight Saving Time is the greatest continuing fraud ever perpetrated on the American people. And this weekend, the effect of this cruel monster will rear its ugly head again. [...] It's time to stop this insanity.

Good point about the idea that if everyone eliminated DST, we wouldn’t have to struggle with an added problem. Life is hard enough. Then we would only have to deal with time zones which is another animal.

It is amusing that when people specify a time, they tend to forget that they looked at their watches or asked other time-keeping devices at a particular geographic location. The value they use for current time is colored by this location so much that the absence of a location at which we have the current time, renders it completely useless -- it could be specified in any one of the about 30 (semantically different) timezones employed around the planet.

This piece of information is amazingly useless.

The basic problem with time is that we need to express both time and place whenever we want to place some event in time and space, yet we tend to assume spatial coordinates even more than we assume temporal coordinates, and in the case of time in ordinary communication, it is simply left out entirely. Despite the existence of time zones and strange daylight saving time regimes around the world, most people are blithely unaware of their own time zone and certainly of how it relates to standard references. Most people are equally unaware that by choosing a notation that is close to the spoken or written expression of dates, they make it meaningless to people who may not share the culture, but can still read the language. It is unlikely that people will change enough to put these issues to rest, so responsible computer people need to address the issues and resist the otherwise overpowering urge to abbreviate and drop context.

For some reason, precision in time always suffers when people are short of space.

The Roman tradition of starting the year in the month of March has also been lost. [...] This means that month number 7, 8, 9, and 10 suddenly came in as number 9, 10, 11, and 12, but kept their names: September, October, November, December. This is of interest mostly to those who remember their Latin but far more important was the decision to retain the leap day in February. In the old calendar, the leap day was added at the end of the year, as makes perfect sense, when the month was already short, but now it is squeezed into the middle of the first quarter, complicating all sorts of calculations.

Making Februarius the second month, which explains why leap day is at such a funny point in the year.

There is nothing worse than doing an excellent job and then being penalised by the client for not meeting a deadline merely because of ignorance of time zones.

In 2019, this [March equinox] is on March 20, at 21:58 UTC. For locations that are more than two hours ahead UTC, it will be on March 21. This is because of the time zone difference.

These dates [March 19, 20 or 21] are based on the time of the equinox in UTC. Due to time zone differences, locations ahead of UTC may celebrate the March Equinox a day later, and locations behind UTC may celebrate it a day earlier.

It turns out humans have had a long, long history of poorly dealing with time, so when you hammer your head against the wall trying to deal with a timezone bug, well, you're just the last in a long, long line of human beings that are terrible at all this!

This complicated everything, and made obvious the notion that writing timezone code was some of the worst things you have to do in our field.

Birthdays are kind of like floating events; it doesn't really matter where you are on the planet; if the numerical day matches up with what's on your driver's license we're set. Storing these things as a timestamp column instead of just a date column in your database can end up complicating your code down the line.

Also if you find yourself at this critical juncture, you might want to just give up programming entirely, because that sounds horrible.

I don't know if you've noticed this yet, but Americans aren't perfect. We use this fucked up date format, MM/DD/YYYY. Like under no circumstances does this make any sense.

The *Brightest Point in Human History*, also known as ISO 8601.

Some people believed it stole 11 days of their lives.

I can easily maintain a timezone list myself.

This is the second time in the space of three years that Metlakatla tampers with its time zone.

The time-zone map is a hodgepodge.

There's plenty to argue about in cyberspace, as in the real world. We could at least agree on the time.

If an area switches their time zone, events created before we knew about the change might be in the wrong time zone.

That's a pretty bizarre way to divide a day up. We divide it in half, then divide the halves by twelfths, then divide the twelfths into sixtieths, then divide by 60 again, and then convert to a decimal system for the smallest increments. It's no wonder children have trouble learning how to tell time.

I think it's funny to see people complaining about the use of TimeZones because of how they interfere with certain highly specialized situations, as if the entire world, most of whom rarely travel more than a couple hundred miles from home and therefore don't give a rat's ass about TimeZones, should just *drop everything* and change the way they do things just to please programmers and other geeks.

How does one write a date on the Web? There are so many formats available, most of them incompatible with others, that it can be a usability nightmare to choose a date representation when writing for an international, cross-cultural audience, as is the case on the web. Fortunately, there is one solution in the ISO-developed international date format.

Your answer will depend, mostly, on which country you live in.

How can we expect peace in the region when they can't coordinate time change?

This is madness and it must stop.

The purpose of uniform time measures is economic coordination. If economics were not the main concern, we'd all be on our own solar time, when the sun is directly overhead at noon.

But times have changed, and we still have times zones based on commerce and the global hegemony that prevailed in 1883. The modern world is even more integrated, equal and mobile, suggesting we'd benefit from fewer, more stable time zones.

If our sole objective were better economic coordination, we'd move to a single time zone. But you can have too much of a good thing. A single time zone removes people too far from their solar time, so time would start to lose all meaning.

Messing with time zones may upset our fragile global order, especially when time can be a hot political issue.

Now the world has evolved further – we are even more integrated and mobile, suggesting we'd benefit from fewer, more stable time zones. Why stick with a system designed for commerce in 1883?

The longitudinal distance of Alaska is nearly equal to the entire continental United States, yet the state functions, albeit with some tension, on one time zone. China has been on one time zone since 1949, despite naturally spanning five time zones.

Time is already arbitrary, why not make it work in our favor?

As tempting as it may seem, it is best to leave the application-wide time zone as UTC.

When working with APIs, it is best to use the ISO8601 standard, which represents date/time information as a string. ISO8601's advantages are that the string is unambiguous, human readable, widely supported, and sortable.

We can only reset your time zone once, so please choose your new time zone carefully. Requests to change time zones must come from an Admin user, and your time zone may only be shifted eastward – for example, GMT (Greenwich Mean Time) to IST (India Standard Time). If your time zone has already been reset once at your request, we will not be able to change it again.

We feel your pain. Maybe a bit more quietly, since the editors of a productivity-minded blog don't like to reveal they missed a deadline or appointment due to EST/CST/PST confusion. But it's happened to us, it's happened to people we've been waiting on, even though we seemingly live in an age where this should all be figured out.

The thinking and methodology is the same: don't ever rely on your own brain to do the cross-time-zone math.

Since some states (I'm looking at you Kentucky and El Paso Texas) have more than one time zone, always ask the customer what time zone they are in. They may think that you are stupid, but it's better than being late.

[Sorry for the language in this quote] Fuck timezones. Fuck them and especially fuck Daylight Savings Time. And fuck having to conform to all this BS to avoid confusion just because the entire human race is fine with being dangerously insane.

If your software is only used in a single timezone, you might think that timezones are not necessary to think about. But being aware of the timezone is necessary in more situations than what most people assume.

In maths and physics classes in school I was taught to always keep track of the units when doing calculations.

I think that in general it is best if data is not silently created by default. [...] It can be convenient in some cases. But convenience in one situation can be silent creation of bad data in another situation.

A unit is information. And wrong information is worse than a lack of information. Using wrong units is worse than not using any units.

The most important thing we can learn [...] is that there is a many-to-many relationship between time zones and offsets. A time zone is a place that can map to different offsets, depending on what time of year it is or which year it is. At any given moment, an offset is usually shared by multiple time zones.

I can delight in all this mess or tear my hair out, and I choose to delight in it!

We are in Phoenix and for some reason my friends iPhone time zone set to Denver after she rebooted it. So she woke up an hour early today after daylight savings happened in Denver.

My iphpne X just decided it is 20th of November at 12:00 and refuses to change this.

That’s three weeks in a row that my alarm has woken me up an hour early because it thinks it’s 7am when actually 6am !

I set my iPhone to manual and choose my timezone. As soon when I reboot the device it’s again on automatic and then the wrong time zone. Very annoying, they really should fix it once for all.

It is by no means straightforward. You really have to do your homework to work out what the time is.

It is all much more confusing than if all countries stayed on the 24 hour time zone system.

Assume that you live in a region that does not observe daylight saving time (DST), but the Windows setting for that time zone *does* adjust for daylight saving time. In this situation, we recommend that you do not clear the **Automatically adjust clock for Daylight Saving Time** check box in the **Time Zone Settings** dialog box.
If you disable this daylight saving time functionality, Microsoft Outlook 2010 and Microsoft Outlook 2013 may assign the wrong time to a Calendar event, appointment, or meeting request for recipients in other time zones that do observe daylight saving time. Instead, we recommend that you select a time zone that does not observe daylight saving time and that has the same Coordinated Universal Time (UTC) offset as the current time zone.

Approximate answers will do so long as the students get times in the right ballpark.

Note that the times given here will vary from time to time depending on Daylight Saving. New Zealand and Australian Daylight Saving does not always begin and end on the same date. In addition, not all Australian States always go on Daylight Saving.

Russia’s President Dmitry Medvedev wants to reduce the number of time zones across the country to improve the nation’s economy.

He questioned if the time zone differences could impact on efficiently managing the country and the use of technology.

I found it really hard to handle time zone in JavaScript.

I had a valuable and thrilling experience of solving a problem leading to cause more problems.

For this to work, the date and time transferred from the client to the server must be values based on the same offset (usually UTC) or values that also include the time zone data of the client environment.

The return value of this method varies from browser to browser, and the format of the string type can affect the prediction of exact value.

A string like 2015–10–12 12:00:00 returns NaN on Safari and Internet Explorer while the same string returns the local time zone on Chrome and Firefox. In some cases, it returns the value based on the UTC standard.

To get the correct time zone value, you must know the value of the offset at the time of the date (not of the current date).

The problem doesn’t stop here. There is another problem waiting where you won’t get wanted value by simply adding or subtracting offsets.

Therefore Date objects produced using those strings may represent different moments in time depending on the version of ECMAScript supported unless the system is set with a local time zone of UTC. This means that two date strings that appear equivalent may result in two different values depending on the format of the string that is being converted.

It's strange to think that time zones were invented as a way of reducing confusion rather than causing it.

Once upon a time we set the logging for servers in the local time of wherever they were located. This made correlation of events, especially to local computers, consistent and relatively easy. Then the internet was born, and we moved our servers to the cloud and data centers. Suddenly, setting logging to local time made no sense at all.

If you are a small firm and all your administrators and users are in one time zone, logging into that time zone might be more appropriate. If all of the logs are pulled into a central location from various time zones for analysis, you might choose UTC to do a cross analysis.

What is more useful, however, is a definition of time that doesn't vary with location. This is called Universal Time (UT) and is a modern form of Greenwich Mean Time. It is the same everywhere in the Universe.

The most common usage in transport timetables for air, rail, bus, etc. is to use lightface for a.m. times and boldface for p.m. times.

The 24-hour notation is also widely used by astronomers, hospitals, various forms of transportation, and at radio and other broadcast media outlets behind the scenes where scheduling programming needs to be exact, without mistaking AM and PM. In these cases, exact and unambiguous communication of time is critical.

Keep in mind that time zone changes themselves are made at the whim of the world's various governments, and not all changes are made with sufficient notice to make it into these release cycles before their respective effective dates.

It's important to recognize that while an IANA time zone can be mapped to a single Windows time zone, the reverse is not true. A single Windows time zone might be mapped to more than one IANA time zone.

McIlroy nearly missed his singles match that Sunday here at Medinah when he got confused with the time zones.

Keep in mind that a time zone and a time zone offset are two different things. An offset of -08:00 doesn't necessarily mean you are in the US Pacific time zone.

For this reason [pending abolishing of DST], all subsequent clock changes in European countries are marked as preliminary.

Adding working dates to your application always seems like an easy task in theory, until you attempt to do it.

I experienced a total of four different time zones, [...] these time zone changes were way more disorienting than I could have ever expected.

I didn't know when the time changes were coming, and it was jarring every time they did.

By the middle of the trip, I had absolutely no idea what time it was, and I felt like I was in this weird time warp with no bearings to grab hold of.

Accounting for a mid-sleep time zone change and a non-adjusting phone clock caused a huge mess of math calculations and incorrect alarms.

People often forget that computer clocks are adjusted when transitioning to or from Daylight Saving Time.

Two different computers could be given the same integer date value, and they will output different strings.

To be fair, not *everyone* has this problem. It seems someone at Microsoft had the good sense (or the accidental fortune) of *not* following the spec.

You can not parse a string containing a time zone abbreviation and be absolutely certain that it was interpreted correctly.

The fact that there's a time zone change is not big news. There are usually a dozen or so per year. The problem here is the short notice. It takes time to digest, interpret, validate, incorporate, distribute, and adopt these sort of changes. The processes are all in place, but it requires time!

Short notice changes will cause people to miss scheduled appointments. Cumulatively this has a negative net effect on your economy.

But who wants to navigate through the complex set of rules for determining a time zone name?

Whenever a government decides that their time zone should either move to a different base offset, enact or rescind daylight saving time, or change the *transition* dates of when daylight saving time begins or ends, the existing rule becomes invalid, and a new rule goes into effect. You may not realize it, but this happens all over the world, dozens of times every year!

Microsoft time zones can only implement two DST transitions in a given year, so they have problems representing real-world scenarios.

Wouldn't you rather sleep soundly, knowing that you've accounted for DST properly?

If the local time is unimportant in your application, then record the activity with respect to UTC instead of the local time. Daylight saving time does not occur in UTC.

If you haven't noticed, computers are a bit different than people. In particular, they tend to be more precise.

Any use of the `TimeZone` class. For any reason, whatsoever. Period. No, really, I'm serious. Just don't.

But did you know that time zones and DST rules often change for other time zones all over the world? Multiple updates are made every year. Think globally, people!

Just because the input fits in a `DateTime` doesn't mean it's a *valid* date time.

Like many systems, yours probably runs some kind of daily or weekly reports. Are you considering daylight saving time when you evaluate the results?

Ahh, time zones. There are so many wonderful traps to fall into.

Europe handles DST slightly different than the USA. In the USA, each time zone changes at 2:00 AM in their own local time. But in Europe, all participating EU countries transition at exactly the same instant (at 1:00 AM UTC).

Many other really smart developers make similar mistakes, because it's just too easy to do so!

Application logic should never make a direct call to `DateTime.Now` or `DateTime.UtcNow` or `new Date()` or `GetSystemTime` or whatever the equivalent is in your language to get the current date and time.

From a global perspective, one might think that the time zones of the world should be managed by some relatively neutral international body.

Not only does this mean that the clocks need to be changed four times in a single calendar year, but it also means that nobody is fully certain of when the middle two transitions will occur until the government makes an announcement.

Don't try to invent your own time zones, or hard-code a list of time zones into your application.

The likelihood of economic impact would increase significantly. Some people will miss important meetings. Some people will miss their flights. It will likely affect stock trading, traffic patterns, and other aspects of society that are time sensitive.

There will be considerable confusion about what time it is.

Even though pilots and air traffic controllers use UTC, most reservation systems use local times because they interact with *consumers*.

Time is an indefinite concept, an abstract that has to be measured simply to allow us to make progress with others in terms of accounting, levying and other financial matters, arranging meetings over time and distance.

Economic reasons may also change the time zone temporarily. It's the case of daylight saving time, when a country or a region “moves” one hour back or forth to keep more of the working hours under sunlight. Geographically nothing changes, just convention.

People traveling to and from areas of the world that use unusual time zones may become confused or have trouble discerning the local time.

If you can’t see the sun, you don’t kneed [sic!] time zones.

When the Australian dignitaries arrived, one of them said “Where have you been? You’re an hour late.”

From today’s view, there is no particular reason why we adjust the time zones instead of our schedule. Someone in Australia could, for example, just as easily work from 19:00 to 3:00 UTC and go to sleep from 10:00 to 16:00 UTC. It might be a slight inconvenience that the date changes during working hours, though — but that’s nothing you could not get used to.

It all started with those bloody British railroad nuts!

So, we have time zones because, if we didn’t some people would have to go to work in the middle of the night while others wouldn’t go until noon.

Time is the virtually the same around the world. Different time zones are just different ways to express the same time. 4 AM PST, 7 AM EST, 14:00 SAST, and 22:00 AEST are all the same time; they are just different ways of expressing 12 noon UTC.

I just think it could alleviate some challenges that different time zones cause in the globe as we know it now.

Time zones are harder than creating a standard time. Ask yourself, can you name every global time zone and describe its offset? Do you know which are subject to daylight savings and which are not? How do you make appointments with people across these different zones?!

Humans have made a mistake. It’s probably not correctable.

It is called UTC and/or GMT. All you have to do is to convince your local populace to use that instead of their current local time zone.

Noon at 12:00 local is too important to lose.

Who knows, maybe in 30 year's time the English (and the N.Americans) will do away completely with am and pm and catch up with the rest of Europe.

No logical fundamental argument can apply, as a symmetrical argument can always be made in favour of the other choice.

What matters is if you tell some people to meet you somewhere at 12:00 PM, will they all know to meet you a little after 11:59 AM, or will they all be there a little after 11:59 PM, or a combination of both?

Give it 20 years and "am" and "pm" will be considered obsolete by the Americans. The Brits, however, will desperately cling to it for at least 40 more years.

It was a mess. All federal offices were still on standard time. Some businesses stubbornly refused to move their clocks ahead, and nobody could say for sure who was leaving late or early.

Several studies found that the further west people live within a time zone, the more health problems they may experience and the shorter they live on average.

DST commonly lasts from spring to autumn and changes the social clock so that we simply start the social day 1 h earlier relative to the sun clock.

People who have to get up at 0600 h by the sun clock in winter have to get up at 0500 h by the sun clock under DST, despite the social clock showing 0600 h. Essentially, they have to go to work in 1 time zone further to the east. This means that people in Chicago have to work during the office hours of New York, and people in Berlin have the office hours of St. Petersburg.

And of course it doesn't make any difference if we move the clocks so that 3AM is called 7AM — you can't fool the body's clock. It's the internal clock that matters, and that clock mostly cares about when the sun is up.

It's important to remember that initial enthusiasm for year-round DST does not mean that we've felt what it feels like to go through a winter without light in the morning.

Social clocks change extremely slowly — we still decide "when" to do things based on when people worked outdoors.

[...] some states on Standard, some states on DST, some still switching—we end up with a confused patchwork of states that will make everyone a little crazy.

You are now considering a bill that would put Massachusetts squarely in the lead for the state taking the smartest and best step forward in fixing the twice-yearly madness of changing clocks into and out of Daylight Saving Time.

He also realized that he needed a better reason to switch the clocks, so he and his PR team came up with the stuff about the farmers. That's why everyone thinks DST is for the farmers. It's not, and never has been. In fact, they've always hated it. The only reason we think it was for the farmers is one of the greatest PR con jobs of all time.

But the science is now clear. Changing clocks kills people every single year, and will keep killing people until we stop it.

Let the world know that science and research should drive our decisions about how we set the clocks, not "That's the way we've always done it"-ism.

In 1918 one thing started, however, that we still have today: Daylight Saving Time.
One other thing started that year that we also still have today: Hatred of Daylight Saving Time.

This scourge of Daylight Saving Time is bad, and while it would be poetic and lovely for it to end in its 100th year, I want to make sure that when we kill it, it stays dead forever.

The best thing for all of us would be to just stop doing the thing that is deadly, and that's changing clocks twice a year.

Just because we use the word 'standard time' does not mean that it should be accepted as the norm.

There are things you can do to mitigate it, but there's no question that jet lag is a real thing. I'm certainly discovering that in dramatic fashion right now after returning from a trip where I left Sydney at 11 a.m. on a Sunday and landed at 6 a.m. … on the same day … in L.A. And that was after sitting for 15 hours on an airplane.

Perhaps this is Stockholm Syndrome or something, but I actually think the Feds have a point in wanting uniformity of time zones in the whole country. I don't want people in New York to have to figure out if Denver is two or three hours different depending on the time of year. The disruption to transportation is real.

There is no sane reason to keep switching into and out of time zones other than the fact that we started doing it during World War I and somehow never really stopped.

This Monday — right after the "spring forward" change — U.S. citizens will die.

Think about it. Daylight Savings does kind of seem like an elaborate practical joke you might play on freshman in a particularly cruel dorm. Imagine if someone moved your alarm by one hour a couple of times a year... Think about how mad you'd be. Or imagine that you tried that as a joke on your spouse. I know if I tried that as a joke on my wife there's a good chance I'd wake up the next day in the ICU.

The jolt is much worse than, for instance, traveling one time zone away because in that time zone the sun is coming up at a time that's more consistent with your wake-up time.

Did you ever do business with someone in Arizona or Hawaii? If so, you probably expressed some frustration that it's hard to keep track of the number of hours difference from them to you because it keeps changing. (If you said something to them, they undoubtedly responded that it is you that's doing all the changing around. They do have a point.)

If DST clock changing doesn't bother you, just consider yourself lucky. But don't get mad at others because they are working on it.

The sun is going to keep coming up, no matter what we do. But the way that we agree to run the clocks, that is up to us. Here on the weekend of the deadly time change, how about we agree to do what we can to fix that, or at least not be so critical of others who are trying to make a difference?

A time zone border between the countries would have "enormous practical implications."

At the poles themselves, the Sun rises and sets only once each year on the equinox.

If an observer located on either the North Pole or the South Pole were to define a "day" as the time from the maximal elevation of the Sun above the horizon during one period of daylight, until the maximal elevation of the Sun above the horizon of the next period of daylight, then a "polar day" as experienced by such an observer would be one Earth-year long.

Latitude thus affects how early or late the sun rises and sets, but what the clock says depends on a location's longitude within its time zone. The farther east it is, the earlier the sun will rise and set.

Sitting here in my grounded office, it is baffling to think about a place where a single human can decide to create an entire time zone at any instant.

"Time" is just an operational ritual, intended to create the illusion of regularity.

When I toggled off airplane mode, I saw the time jump from 8 P.M. to 9 P.M. in an instant. Time is weird everywhere.

Some services use strictly UTC time to avoid confusion.

Your phone picks up the time automatically based on its cellular/Wi-Fi connection. This usually works perfectly fine, but for a multitude of reasons, it is possible to trip it up now and then.

If you're someone that does a lot of traveling or happens to leave [sic!] near the edge of a time zone, this might also cause issues with the time on your Android phone.

It's worth diving into what time zones are and sharing why they're more mind-bending than most people think!

For software running on servers connected to the internet, the users can be in all kinds of time zones and it is those time zones that are interesting. Not what the time zone on the server happens to be set to.

The easiest solution (probably the only solution) is to use UTC for logging timestamps. You can always convert the times to whatever local time you like when displaying them.

A time zone is a set of rules for handling adjustments and anomalies as practiced by a local community or region. The most common anomaly is the all-too-popular lunacy known as Daylight Saving Time (DST).

Specify a proper time zone name in the format of Continent/Region, such as America/Montreal, Africa/Casablanca, or Pacific/Auckland. Never use the 2-4 letter abbreviation such as EST or IST as they are not true time zones, not standardized, and not even unique(!).

Nearly all of your backend, database, business logic, data persistence, data exchange should all be in UTC. But for presentation to users you need to adjust into a time zone expected by the user.

It doesn't need to be so complicated. Just plant a sundial in front of City Hall and declare **Local Time** wherever you live and we can run our shops restaurants, schools, and all the stuff of our daily lives on the time that suits our natural circadian rhythms.

For conference calls, zoom happy hours, for baseball games, for airplanes, all those things where everyone needs to coordinate the times, use UTC, that's why it's called coordinated.

Computers are laughing at us.

They always have to ask what the time is, and every time they do I say "just add 8 hours" but nearly 9 years later they still keep asking. Make of that what you will...

There is only one time in reality: wherever you are, at this instant, it is the same moment in time. We are simply assigning different names to the same thing — with much resulting complexity and confusion.

It's all a recipe for missed connections, misunderstandings and uncertainty. If you travel about this country on business, you will no doubt have had occasion to curse your computer calendar, having carefully scheduled an 8 a.m. meeting in Toronto the week before in Vancouver, only to find (too late!) the time had been helpfully "updated" to 11 a.m. en route. Airlines know the havoc this could cause, which is why the world's airlines all set their clocks to the same Coordinated Universal Time.

This wouldn't entirely eliminate the daylight savings issue. People in the higher latitudes would still debate how to adjust to the lengthening and shortening of the days with the seasons.

Software development is hard. Time zones are hard. Dealing with time zones in software development? Yeah, **harder**.

Time zones are one of the most complicated topics you'll find while developing software. They're complex by themselves, and even more when you throw some code into the mix.

The time is one hour off.

Assume that you live in a region that does not observe daylight saving time (DST), but the Windows setting for that time zone does adjust for daylight saving time.

This list [of time zones that do not observe daylight saving time] may change at any time.

The times and locations of a series of future meetings are stored as a pair of strings: one for the calendar date and wall-clock time, and one for the time zone. They cannot be stored as an exact time because between now and the time when the event happens, the time zone rules for daylight saving time could change — for example, Brazil abolished daylight saving time in 2019 — but the meeting would still be held at the same wall-clock time on that date. So if the time zone rules changed, the event's exact time would change.

`Date` has been a long-standing pain point in ECMAScript.

A `Temporal.Instant` represents a fixed point in time (called **"exact time"**), without regard to calendar or location.

A `Temporal.ZonedDateTime` is a timezone-aware, calendar-aware date/time type that represents a real event that has happened (or will happen) at a particular exact time from the perspective of a particular region on Earth.

A `Temporal.DateTime` represents a calendar date and wall-clock time that does not carry time zone information.

Converting between wall-clock/calendar-date types [...] and exact time types [...] can be ambiguous because of time zones and daylight saving time.

The core concept in Temporal is the distinction between **wall-clock time** (also called "local time" or "clock time") which depends on the time zone of the clock and **exact time** (also called "UTC time") which is the same everywhere.

Wall-clock time is controlled by local governmental authorities, so it can abruptly change. When Daylight Saving Time (DST) starts or if a country moves to another time zone, then local clocks will instantly change.

The `Temporal.ZonedDateTime` type encapsulates all of the types above: an exact time [...], its wall-clock equivalent [...], and the time zone that links the two.

In both cases, resolving the ambiguity when converting the local time into exact time requires choosing which of two possible offsets to use, or deciding to throw an exception.

Time zone definitions can change. Almost always these changes are forward-looking so don't affect historical data. But computers sometimes store data about the future!

The developer needs to decide how to fix the now-invalid data.

It enables awesomeness. Unfortunately, it is not a reality for JavaScript.

When I didn't specify a time, the value was interpreted as a UTC value, but when I did specify a time it was interpreted as local. This is all... a bit mad.

Because it was the 'reality' of how the web worked – even if it wasn't correct by the definition of the standard, or even particularly logical.

Introduce a new datetime handling object to JavaScript, giving us a clean slate to make the world right.

Look at the United States of America, with its depressingly moronic units instead of going metric, with its inability to write dates in either ascending or descending order of unit size, and with its insistence upon the 12-hour clock, clearly evidencing the importance of the short-term pain threshold and resistance to doing anyone else's bidding.

The bottom line here is that even knowing the time *and* time zone, it's meaningless unless you also know the date.

Even if you have complete and accurate information about the rules, daylight saving time complicates things in surprising ways.

Working with human times correctly is complicated, unintuitive, and needs a *lot* of careful attention to detail to get right.

The good news is that anyone observing the timer will see it smoothly count down towards 0, with no jumps. The bad news is that when it reaches 0, the conference won't actually start – there'll be another hour left. This is not good.

I'm using the IANA ID for simplicity, but it *could* be the Windows system time zone ID, if absolutely necessary.

Now, anyone refreshing the countdown timer for the event will see the counter increase by an hour when the entry is updated. That may look a little odd – but it means that when the countdown timer reaches 0, the conference is ready to start. I'm assuming this is the desired behaviour.

Having a UTC representation makes it easier to provide total orderings of when things happen.

There's a *huge* difference between knowing the time zone that's applied, and knowing the UTC offset in one specific situation.

While the mapping from UTC instant to local time is always completely unambiguous for a single time zone, the reverse mapping (from local time to UTC instant) is *not* always unambiguous.

That assumes that the time zone associated with the event will not change over time. That assumption may not be valid.

There can be a significant delay between the [time zone rule] changes being published and them being available within applications.

My experience is that developers either don't think about date/time details nearly enough when coding, or are aware of some of the pitfalls but decide that means it's just too hard to contemplate.

I'm very well aware of the TZ database, as the author of the Noda Time package that exposes it for .NET developers...

Having slightly incorrect times for a while might be a worthwhile tradeoff compared to the investment it takes to do things correctly.

You really really need to store the time zone if the value you're representing is intended to be considered in one specific zone.

It's primarily for future events, where your current idea of what the UTC offset will be in the future may prove to be wrong.

It's pretty clear what "YYYY-MM-DD HH:MM:SS" means as a date/time format, but that doesn't mean it's the right pattern to put in code.

There are two common issues when understanding what a time zone is to start with. The first is to assume that a UTC offset (e.g. "+8 hours") is the same as a time zone. [...] The second mistake is to think that an abbreviation such as "EST" or "GMT" identifies a time zone.

But even without this interminable flip-flopping, time zone quirks exist the world over.

Is the fact that time zones make very little sense reason enough to scrap them?

A lack of locally adapted official hours for work and school means that those furthest from Beijing get up in darkness, or head to bed while it's still light.

Time zones have been leveraged as a political tool throughout history.

Without the strictures imposed by a particular time zone, different locations would be free to tamper with their local working hours and timetables. Although all the clocks would be set to the same time, business hours would vary by location.

What is important is being connected for whatever purposes. If something stands in the way of being efficient or responsive, those barriers seek to be removed. And time zones are one of those barriers.

Spanish people wake up at the wrong time, eat meals at the wrong time and go to bed at the wrong time.

Yes, but those folks live in the middle of the Pacific Ocean. As things stand, they have an International Date Line to contend with. With our proposal, that will disappear forever. So they gain that!

"See you tomorrow" refers to the sun being overhead, not a change in the calendar date.

The time it takes to glance at the sun position is no different from the time it takes to glance at the clock in a specific time zone on phone applications and websites.

Calendar reform has always failed before. The reason was that all the major proposals included breaking the seven day cycle of the week. That is completely unacceptable to humankind, and that will never happen.

Everywhere in the world except USA and Canada, the week begins on Monday and ends on Sunday.

It will be about as costly as the Y2K problem was. Remember that? But it is a one-time cost, and then we are safe until the year 10,000. Also, since we have just been through Y2K, we are in an ideal position to make a "second adjustment," having already located the software that needs to be adjusted and learned how to do it. Let's not get rusty on this again: strike while the iron is hot, or at least still warm!

When it comes to time zones, if you're still talking about Greenwich Mean Time there's something you should know: it's old news.

It's often only pilots and members of the military who will refer to an event in Zulu time, and usually not when speaking to civilians.

Everyone would know exactly what time it is everywhere, at every moment.

But jumping from 24 time zones to one would be a much larger leap. On some islands in the Pacific, the date would change with the sun high in the sky. People would wake up on Tuesday and go to bed on Wednesday.

Think about how much time and effort are expended each year in redesigning the calendar of every single organization in the world.

Our customers have due dates and times for various things. Since the viewer sees the due date in their native local time, customers are confused over when exactly that is. Throw in Daylight savings time, and no one really knows what time it is anywhere anywhere else anyway.

But are people really going to get up when the sun is high overhead, or leave for work in the middle of the night? [...] This is absurd! If it's 7am in Moscow we know people are getting up and going to work!

You don't need a clock for anything other than coordination with other people and a single timezone is facilitates that goal more than multiple timezones. Animals have no concept of time as far as we know and they get along just fine. We've massively over-complicated it. Ask anyone who writes programs for a living what the worst part of their job is - they will all say timezone calculations.

Thus there can never be only 1 time zone as long as sun's not up all around the world simultaneously.

No one said people will be forced to change their routine, the only thing that changes is a number. [...] Your children will never even know about the old timezones, and will just know that the sun rises at 13:00 in Los Angeles, and school starts at 15:00. It's all just numbers. Your biological clock doesn't even care about what your watch says.

Now I wished the Earth was flat.

This makes perfect sense. Currently you have to specify a time and the zone to which you are referring. If we use the same time zone, you just need to specify the time.

Not have to be bothered with having go through the headache inducing arithmetic that comes along with remembering what time zone they are in, whether that timezone observes DST, *when* it observes DST, when *you* observe DST, etc.

I find both these ideas extremely stupid.

For everyone worried that they wouldn't know what time a country wakes up and goes to work, do you know exactly what time it is in say China, or Russia, or Australia right now with out looking it up? It would take just as much effort to figure out what time it is in another country under our current system as it would to figure out what time people do things under the new system.

It sounds like a fantastic idea, sure, a near to impossible implementation, but being able to know the exact time of other parts of the world without the need of making conversions would be marvelous. We are living in a global village, thinking of something like that is pretty much expected at this level, I hope they find a solution that works out for everyone.

Everyone operates on their usual schedule, the only difference is that the timing at which people operate would be different.

If the planet adopted UTC, there would be confusion for people getting used to the new times, where the greater the change, the greater the confusion. After that though, international travel, meetings, conference calls etc. would be much simpler.

I think the idea is that people would still go about their business during daylight hours, but the clocks would change.

Do these scientists believe that people in the mid pacific will be going to work at 8 every evening so that everyone around the world operates on clock time.

Do these suggestions turn individuals into units of production? I see no benefit to me!

I agree that we should all be running under a single time zone, using military time as not to have to wake up at 3pm somewhere in the world, but rather at 15:00.

Why bother with specifying time by o'clocks?!? Why not specify it by major city currently under the sun? Like, I eat breakfast at half past Sydney, lunch at Moscow, and dinner at half past Halifax. No matter what method is adopted, mental backflips of some sort are required.

I wish they would do something to make time global. I lost a birthday once flying across the Pacific. The plane flew so slow and it took so long that somewhere along the path and crossing the international date line my birthday disappeared. It would not have bothered me except we always got our birthday off, but since that day did not exist in my timeline I could not.

Believe it. Time zones in JavaScript have a long history of being insufficiently supported, and while there are some recent improvements, we still have a long way to go.

While this format allows for time zones between -23:59 and +23:59, the current range of time zone offsets is -12:00 to +14:00, and no time zones are currently offset from the hour by anything other than `00`, `30`, or `45` minutes. This may change at more or less anytime, since countries are free to tamper with their time zones at any time and in any way they wish to do so.

The second problem is the more serious one; with date input supported, the value is normalized to the format `yyyy-mm-dd`. But with a text input, the browser has no recognition of what format the date should be in, and there are many different formats in which people write dates.

The control is intended to represent *a local date and time*, not necessarily *the user's local date and time*.

Why worry about the Y10K problem if it is going to happen many centuries after your death? Exactly because you will already be dead, so the companies using your software will be stuck using your software without any other coder who knows the system well enough to come in and fix it.

Inexperienced users are not expected to select these names unaided.

The Congressman who introduced Chamorro Standard Time preferred "ChST", so lower-case letters are now allowed.

An extra-special case is SET for Swedish Time (*svensk normaltid*) 1879–1899, 3° west of the Stockholm Observatory.

Application writers should note that these abbreviations are ambiguous in practice: e.g., 'CST' means one thing in China and something else in North America, and 'IST' can refer to time in India, Ireland or Israel. To avoid ambiguity, use numeric UT offsets like '-0600' instead of time zone abbreviations like 'CST'.

The `tz` database is not authoritative, and it surely has errors.

From 1891 to 1911 the UT offset in France was legally UT +00:09:21 outside train stations and UT +00:04:21 inside.

The timezone `America/Kentucky/Louisville` represents a region around the city of Louisville, the boundaries of which are unclear.

Even if the time is specified by law, locations sometimes deliberately flout the law.

In short, many, perhaps most, of the `tz` database's pre-1970 and future timestamps are either wrong or misleading. Any attempt to pass the `tz` database off as the definition of time should be unacceptable to anybody who cares about the facts.

Although the `tz` database does not support time on other planets, it is documented here in the hopes that support will be added eventually.

Note that any representation of time in something that isn't `java.time.*` based is by definition broken, as it is in most programming languages - turns out time is a lot more complex than most takes on a library to represent it realize. The fact that `java.time` is in effect the 4th attempt at writing a time library should be ample indication that it's hard to get it right.

Date is a lying liar who lies - it doesn't represent a date, it represents an instant; it is badly named.

You are now dependent on barely defined behaviour of all the various libraries up and down the chain - you're effectively stuck praying that they do the right thing, or delving into exotic settings to try to cajole these libraries into doing what you want.

If you want to store barber appointments and use `Instant` to do it, you *WILL* be an hour late or early sooner rather than later.

But at the end of the day, we do need to set our wake-up alarms, and that can be challenging if we're relying on digital devices.

With DST, we do not change time, we only change *social clocks*; the *sun clock* with its midday and midnight remains the same and dawn and dusk continue their gradual seasonal photoperiodical/day-length changes.

Although DST is mostly during summer months, DST is simply an advance of the social clock (we agree to do everything 1 h earlier) and does not “make it summer”.

DST changes are therefore **NOT** comparable to traveling to different time zones.

The spring victims can be only rescued by abolishing the clock advance.

The effects may be latitude-dependent.

Our *body clocks* do not heed *social clocks* because *body clocks* are based on *sun clocks* and not political laws; political laws cannot determine health – they can only influence it for the better or worse.

Risks increase and longevity decreases from the eastern to the western border of time zones.

The scientific literature strongly argues against the switching between DST and Standard Time.

DST is simply a work-time arrangement, nothing more than a decision to go to school/work an hour earlier. As such, it is not a decision that should be made by the world, by unions of countries (e.g., the EU), or by individual countries, neither at the federal

nor the state level. Work-time arrangements are decisions that a workforce could decide at the company level.

Like just about every competently implemented user-facing web UI on the planet, it uses the timezone of your browser for any displayed times.
Also, like just about every competently implemented backend on the planet, times are stored as de facto timezoneless quantities in UTC.
Doing anything else is the path to madness...

We already do have a global standard time zone, and everybody who cares already uses it: UTC.

The end of the time shift comes as a relief to border communities straddling two time zones and dealing with the capacity for error that accompanies it.

Being an hour early or late for appointments, missing shop closing times (or sitting around waiting for them to open) are par for the course for those not adept at constantly converting between DST and EST.

The reprieve lasts until Sunday, October 3, when the time shuffle starts all over again.

Any software that thinks that "PST" means "8 hours behind UTC" will be broken. As will any software that thinks that "PST" means "7 hours behind UTC", for older timestamps.
Admittedly such software is dicey anyway, but it's out there and it's in widespread use. [...]
The situations for time zone abbreviations in Europe, the UK, Ireland, etc. will be even more complicated if they go through with plans to institute permanent DST or permanent standard time or whatever it is they might do. It surely will be an even bigger mess to name and abbreviate the resulting time zones.
To avoid the resulting fallout, perhaps we should remove all alphabetic time zone abbreviations from tzdb, other than "UTC" and the "xMT" abbreviations which are pretty much immune to this sort of thing and which we need for other reasons.

When you change language in the Teams client. The date/time settings also change.

If they default they should use the only sensible date that cannot be misunderstood, which is why it is a standard ISO8601.

Is there no way to set the date/time format independently of the language? [...] My OS is set up just like I want it, but in Teams it appears to be impossible.

Amazing that a global business continues to make this so difficult. Been chasing this issue since 1981 with them.

This caused considerable political and economic hardships as the nation tried to conduct normal everyday business with one half of its nation a day ahead and the other a day behind.

Date and time formats cause a lot of confusion and interoperability problems on the Internet.

The actual relationship to UTC may be dependent on the unknown or unknowable actions of politicians or administrators.

The problems with two digit years amply demonstrate why all dates and times used in Internet protocols MUST be fully qualified.

Because the daylight saving rules for local time zones are so convoluted and can change based on local law at unpredictable times, true interoperability is best achieved by using Coordinated Universal Time (UTC).

If date and time components are ordered from least precise to most precise, then [...] the date and time strings may be sorted as strings.

Because no date and time format is readable according to the conventions of all countries, Internet clients SHOULD be prepared to transform dates into a display format suitable for the locality.

Timezones are one of the quirkiest things any programmer has and will ever have to code, because they’re a political construct. They were and are decided by largely non-scientific non-technical politicians. But it’s worse than that, timezones themselves change over time. What timezone a place is in, and worse, daylight savings time, and when daylight savings time starts and stops all change, pretty much every year some place on earth.

As we traveled up to Alaska and now back, anyone using “Set Automatically” in their iPhones etc. found their clocks automatically bouncing between an hour more forward than they should be (based on Seattle) and back. Their smarter devices were dumber at the simple task showing the time.

Please don’t schedule a virtual meeting with an overseas colleague this summer by saying you’re free at such-and-such time “EST.”

I am a huge proponent of going to one world time (GMT or Zulu time). It would be so much more convenient to schedule meetings, and would just make the world a better place.

Even if you do time zones, how do you know if your correspondent works 0700-1600, 0800-1700, or 0900-1800? Or, perhaps they skip lunch and leave an hour earlier, or they work very early or late?
We’re moving away from the “9 to 5", and have been since shortly after Dolly sang about it.
Calendar sharing is easy. We’re making our timekeeping needlessly complex to try to preserve something that is already gone.

This is one of my biggest pet peeves. I can’t stand when I go to the customer support page on a website and I see the hours listed as 8:00-9:00 EST. Does this mean the customer service phone are off an hour in the summer? Or is it that there is a typo?

Why would you be specific and indicate Standard time when we are using Daylight right now.
It infuriates me!

If I say I’m calling you at 11 CST but ...GASP...it’s during CDT, no one is truly confused.

Yes, we completely suck at time.

Figuring out what time it is, was, or will be is way too complicated. I am lazy and imprecise and use the simpler “ET”, “CT”, “MT”.

I had this happen a few years ago when I was trying to schedule a call with a colleague. The person said she would call me at 2pm, CST, even though it was June. I emailed her back and asked if she meant CDT, and she said she didn’t know there was a difference! Friggin annoying, especially in a professional setting.

As a network engineer, I do everything in UTC.
DST exists [to] punish users for buying software written by bad programmers.
I really, REALLY wish we’d just make the switch. Airlines and the military get it right with this.

Get rid of time zones and daylight saving time. The entire world switches to Coordinated Universal Time.
It will take a while for people in California to get used to the sun rising when the clock says 2:30 PM but you can always use the “time is a social construct” argument.
When you have to schedule a meeting between London, Montreal and Los Angeles, the boss just says, “Let’s zoom at 4:00 PM” and everybody’s clock says the same thing. No confusion.

Can we just brutally murder DST?

More people need to read this article. I propose making a test on your comprehension of this concept mandatory for getting out of bed in the morning.
The people who insist on saying EST when their time zone is currently using EDT are just advertising their ignorance.
[...] When you use that S or that D I assume that you know what you mean and will respond based on that assumption. Why else would you have been so specific?

I find that many people I deal with are clueless/imprecise and put ST or DT to reference a time, which is a huge problem if they put the wrong thing. I prefer when they just state ET or PT (as an example) because it’s actually less confusing. If we’re in DST and some one says 3PM PT, I know exactly when they’re talking about. If you can’t figure out if you’re in the time of year that’s DST, you have another problem.

This means that there is a one-hour time difference between the two sides of the road.

There are, however, also individuals who struggle intellectually with the concept of time changes. Some people believe that the very act of moving the clock an hour forward implies that we are adding an hour of daylight to the day. They appear to believe that changing the reference point of our clocks somehow influences the rotation of the earth.
The letter shown below [...]. The author blames the dire situation on daylight saving. He claims that the extra hour of sunlight is "slowly evaporating the moisture". Evaporation indeed increases in summer, but changing the clocks is not most certainly not causing the drought.

Perhaps we can use technology to solve the daylight saving time problem. The way we measure time during the day has changed once before; maybe we should change it again.

For something so seemingly simple and intuitive to humans, the interaction of dates & times, and timestamps & timezones, has proven deceptively tricky for many engineers. And just when we think that we’ve gotten it figured out, we are blindsided by nuances and edge cases, and confronted with ambiguous scenarios.

A timezone [...] is used to *derive* a region’s UTC offset. But that offset can change over time.

Just because we’re not displaying times does not make us immune from timezone issues.

It can be tempting to try to parse, manipulate, or format date-times by hand. In most cases, this is a bad idea. Parsers and formatters don’t exist simply to make coding a little easier. They also handle the numerous complexities and edge cases that many of us mere mortals aren’t knowledgeable enough to properly address or, in many cases, aren’t even aware of.

If our homespun approach shows the correct date in a given user’s timezone, it will likely be out of sheer luck.

But rest assured, we will sometimes see errors. Because there are nuances to calculating dates and times. Daylight Savings Time is perhaps the most common—and most annoying—of these nuances.

Working with dates and times in our code can be deceptively tricky and error prone.

One other irony, is UI work is often outsourced to regions that don't even have timezones or daylight savings.

It will also make the reopening of scheduled commercial air service much smoother if we don’t have to be concerned shifting arrival and departure times, which may look like a simple thing but requires some significant logistical adjustments domestically and internationally.

Working with dates in JavaScript sucks. The Date API is extremely clunky, has almost no methods that respect immutability, and is overall just bad.

Looking at this map always shows me that there's no answer that is universally good everywhere. [...] Days are just shorter in the winter, and you have to adjust somehow.

The bottom line is, why do we need to move every single person to an artificial time? [...] Daylight saving time is a human invention, not what our bodies evolved for.

It took me awhile to understand the challenge at hand and, frankly, I don’t want to be dealing with this madness.

To many people, Microsoft Outlook really seems bad at handling time zone changes, and that's being kind.

If you create appointments when the computer clock is set for the wrong time zone, or if you move to a new time zone, you can't change the time zone without messing up the appointment times and all day appointments will span two days. You can export the

calendar to a non-Outlook format (Excel or CSV works well), then change the time zone settings and import the items back into Outlook.

I travel for a living. I make a lot of appointments for the location I will be at when the appointment occurs. Thus I want a calendar that will allow me to specify the time zone of the appointment and always display the time in that time zone.

As far as I can tell, there is no way to satisfy everyone's intuitions.

Between zones, however, wall time arithmetic is basically meaningless; time zones can be considered the "units" of datetime calculations, so subtracting two datetimes in different zones is similar to subtracting X meters from Y feet – the answer is definitely not (X - Y), you need to convert the two quantities to the same units first.

I am fairly convinced that there is no single intuitive approach to the datetime semantics problem – no matter what model a library or language chooses, you can't easily represent everything we mean when we do math on datetimes.

Picking dates and times is a complex task, and designing a user experience that takes into account internationalization, accessibility, and usability across many types of devices is a huge challenge.

Time zones are another huge area of complexity for date and time manipulation.

Correctly manipulating dates and times is *really hard*. Making assumptions about calendar systems, time zones, locales, date and time arithmetic, etc. is a recipe for bugs when users around the world interact with your app.

Dates, times and timezones are troublesome things.

The clock change was therefore postponed for a week, lest people who forgot to reset their clocks before going to bed on Saturday night might misread the time on Sunday, and innocently end up arriving at their local voting station after it had closed, not realising they were an hour late because their clocks were an hour slow.

It's astonishing how many people refuse to accept that non-integer timezones exist, insisting that *"all legal timezones go in hours"*.

But it *won't solve the problem of how to make sense of computer logs*, and how to use them in IT troubleshooting, notably in cybersecurity threat response, where *the sequence in which things happened can be very important indeed*.

Always reduce timestamps to UTC (universal co-ordinated time), thus factoring timezones out of your logfiles, and **always record timestamps in a simple, unambiguous, alphabetically sortable format**.

Times don't need AM and PM (computers can count to 24 at least as easily as you can count to 12), which removes ambiguity.

That `Z` at the end denotes that the date and time shown *have no timezone adjustment applied*, so that any two *Zulu time* log entries can directly be compared to determine the sequence in which they took place.

A timezome [sic!] name with -S- in the middle in North America implies "winter time", where S is for Standard. In Europe and the UK, -S- stands for "summer", and therefore implies daylight saving.

Zulu timestamps are so much easier for human threat-hunting experts to compare at a glance, and so much simpler to work with programmatically, that I can't imagine how anyone could prefer anything else…

So `DD` and `dd` only produce the same answer in January, after which the *day of the year* goes to 32 while the *day of the month* resets to 01 for the first of February.

Don't make assumptions. Just because upper-case `YYYY` denotes the calendar year in some places doesn't mean it always does.

We take calendars for granted, but their design and use of is a mixture of art and science that has been going on for millennia, and still requires great care and attention.

Trust me, when you screw up time, the failures of your implementation will be felt, painfully.

It is worth noting that any time that is affected by daylight saving is not totally ordered.

It appears it was all a big mistake.

Back then it was chaos. So many people would set an appointment and forget Mexico was on a different time part of the year.

I took a look at the itinerary Evan included in his paper trail. He was correct; it didn't explicitly say that the flight would land on March 26. But it also didn't say that it would land on March 25 either [...]. It had no arrival date indicated.

Conversely, it [daylight saving time] is not observed at some places at high latitudes, because there are wide variations in sunrise and sunset times and a one-hour shift would relatively not make much difference.

The very concept of *time* is simply an agreement among people. In our modern world we need to have time as an agreement to coordinate so much of what we do.
My view is that the *time agreement* has a bug, and the bug makes us change the clocks twice a year. One of those changes, the one in the spring, kills people every time, and injures lots of others.
Once we get rid of that bug, what is the best time zone for us to land in? I really don't think there is a perfect answer for any place, and the answer is a bit different in every place.
Anyone who says there is only one correct answer has some other agenda that they are not telling you about.

Having to deal with leap seconds drives me crazy.

We made a mess of time all over the world.

What you do not want is to have to go digging around on an online search using one of these facts, to try and find the correct time zone term that your calendar accepts.

I don’t know what to say, other than this UX work sucks.

Of all the different time zones, *Coordinated Universal Time* (or UTC) is the one reliable, true compass for time definition. It never changes and serves as a **constant time of reference**, in which other time zones are relative.

Designing time-zone selectors can be tricky.

There is no universal way of organizing time zones.

Most users don’t know their offset and struggle to find their time zone when time zones are ordered by offset.

Imagine you are a non-technical person, living in a country, in a city with a date and a time. You have definitely heard of the concept of timezones before. But do you know the name of the timezone you currently live in? Is it `UTC+1`? Is it `CET`?Is it `CEST`? Is it maybe `Europe/Berlin`?

Well, you shouldn’t know such things!

If we prefix every line with the current time (and date) in that specific timezone, it is fairly easy for a customer to find the timezone that they are looking for, as they know what time it is at the moment where they live.

We have not found a suitable grouping of timezones as of today.

A very friendly and straightforward type of feedback for time zone settings is to show the local time according to the selected time zone. Users can simply look at the clock and verify if the displayed time matches their time.

Time zones are dumb. They are a vestigial holdover from history, and we should get rid of them.

We don’t need time zones. Of course, that’s no reason to get rid of them. Time zones make life unnecessarily difficult, and likely cause a significant amount of economic inefficiency. Those *are* good reasons to get rid of them.

Time zones suck for people.

Daylight Saving Time (another terrible time institution in practice around the world), may cost hundreds of millions of dollars in lost productivity in the United States alone.

Time zones suck for computers.

Without time zones, there would never be the potential for ill-defined behavior. Apps wouldn’t have to worry about compensating for time zones of the user or of other apps. Stuff would just work.

Time of day doesn’t mean much near the poles. The sun is up, or it isn’t, and this largely depends on the season.

Abolishing time zones isn’t just something that might or should happen. It’s inevitable.

Time zones are dumb. We don’t need them anymore, and we will eventually get rid of them anyway. Were I king of the world for a day, I would abolish them.

30 References

30.1 Nemo nisi mors (NNM)

- The golden timestamp rules
- Current time
- Constant time
- NNM Time zones
- NNM Meeting time across time zones
- NNM Clock, with the proof-of-concept The wall of time
- NNM Local date range picker
- NNM Map viewer
- Clock angle
- The 37818

30.2 Wikipedia

- Time zones
- AM/PM and 12-hour clock
- 24-hour clock
- 24-hour analog dial
- Decimal time
- Metric time
- Greenwich Mean Time (GMT)
- Coordinated Universal Time (UTC)
- List of time zone abbreviations
- Daylight saving time
- Daylight saving time by country
- Winter time (clock lag)
- Calendar date (especially the *date format* section)
- Date format by country
- Date and time representation by country
- ISO 8601 standard
- Calendar systems
- Noon
- Zenith
- Nautical time
- Time in Australia
- tz database (a.k.a. IANA time zone database, Olson database)
- List of tz database time zones
- International Date Line
- Swatch Internet Time
- New Earth Time
- Märket
- Date and time notation in the United States
- Time in the United States
- Time in Norway
- KISS principle (keep it simple, stupid)

- Time in Europe
- UTC offset
- Midnight sun (Polar day)
- Polar night
- Latitude
- Longitude
- Anywhere on Earth
- Abolition of time zones
- Military time zone

30.3 World Wide Web Consortium (W3C)

- Working with Time Zones (the 20051013 and 20110705 Working Group Note versions)
- Use international date format (ISO)
- Date formats

30.4 Java API

- The `java.time` package

30.5 MDN Web Docs (previously Mozilla Developer Network)

- The JavaScript `Date` object
- The HTML `<time>` element
- Date and time formats used in HTML
- The HTML `<input type="time">` element
- The HTML `<input type="date">` element
- The HTML `<input type="datetime-local">` element

30.6 timeanddate.com

- Time and Date
- Time Zone Converter – Time Difference Calculator
- The World Clock
- Why Do We Have Time Zones?
- Half Hour and 45-Minute Time Zones
- Australia's Time Zones
- Why Is There Only 1 Time Zone in China?
- Daylight Saving Time Statistics
- Egypt and Morocco suspend DST for Ramadan
- The Never Ending DST Debate
- Time Zone News
- 39 New Years in 2019 – New Year Countdown
- What Are Longitudes and Latitudes?
- Equinox: Equal Day and Night, Almost
- March Equinox - Equal Day and Night, Nearly
- The September Equinox
- What Is Universal Time?
- What Is International Atomic Time (TAI)?
- What Is a Time Zone?

- Time Zone Abbreviations – Military Time Zone Names
- The Future of Leap Seconds
- UTC – The World's Time Standard
- US Senate Approves Permanent DST Bill

30.7 Stack Overflow

- Daylight saving time and time zone best practices
- About timezone
- About DST
- How to translate between Windows and IANA time zones?
- A great answer by Basil Bourque to the question "What's the difference between Instant and LocalDateTime?"
- Getting the UTC timestamp in Java

30.8 Code of Matt

- JavaScript Date type is horribly broken
- Handling Birthdays, and Other Anniversaries
- Time Zone Abbreviations
- UTC vs GMT
- What is a Time Zone?
- Five Common Daylight Saving Time Antipatterns of .NET Developers
- JavaScript Date Parsing Changes in ES6
- Happy New LEAP Year!
- On the Timing of Time Zone Changes
- Time Zone Chaos Inevitable in Egypt
- On the Life Cycle of a Leap Year Bug
- Matt Johnson: The Past, Present, and Future of JavaScript Date and Time APIs | JSConf EU 2017

30.9 IANA.org

- Time Zone Database
- Theory and pragmatics of the `tz` code and data

30.10 Misc

- Why You Should Use Timezone Offsets Not Timezone Names
- How to save datetimes for future events - (when UTC is not the right answer)
- Falsehoods programmers believe about time and time zones
- World time is more than GMT offsets
- What everyone except programmers knows about dates
- How more time types prevent bugs and add clarity
- The need for timezone awareness
- Why not to ask the server for its "local time"
- Timezone updates need to be fixed
- The Problem with Time & Timezones - Computerphile
- Here are some of the world's most stupid time zones
- Millenium (sic!) - Date Line Politics
- In Pacific Race to Usher In Millennium, a Date-Line Jog
- The anatomy of time

- So You Want To Abolish Time Zones
- So You Want Continuous Time Zones
- Now
- Explaining Time Zones and Best Practices for Configuring Time on Servers
- The radical plan to destroy time zones
- The case against time zones: They're impractical & outdated
- Imagine a World With One Universal Time Zone
- The Long, Painful History of Time
- Time Zones Aren't Offsets – Offsets Aren't Time Zones
- The advantages of dwelling in a decent timezone
- Time zones were a good idea when they were invented in the 19th century — but today they cause more trouble than they're worth
- Why Don't We Share the Same Time Zone?
- Changing Times
- Lawmakers want one Florida time zone, to make Daylight Saving Time permanent
- 12 insane facts you didn't know about time zones
- Why Does China Have Only One Time Zone?
- Time zones are dumb
- A Plan to Create Just Two Time Zones in the Continental United States
- Time Zones: What If We Got Rid Of Them?
- Why is Singapore in the "Wrong" Time Zone?
- Procedure for Moving an Area from One Time Zone to Another
- Design (of the 24 hour analog dial)
- Discussions around "So you want to abolish time zones"
- Time zone - A complicated complication!
- What would an Earth with no timezones be like?
- What would be the pros/cons for abolishing Timezones, Daylight Savings, and the 12-Hour Clock—thus creating a new world-wide time standard?
- Moment.js Guides
- Moment Timezone Documentation
- Luxon - Time zones and offsets
- Time Zones (and Daylight Savings) in your Infrastructure and Applications
- 5 weird things you didn't know about time zones
- Procedure for Moving an Area from One Time Zone to Another
- Time zone
- The Case for and Against Daylight Saving Time
- Daylight Savings Time: 7 Surprising Things You May Not Know
- The Worst Server Setup Mistake You Can Make
- Time Zones
- Ken Jennings Finds the Spots on the Map Where Time Travel Is Possible
- Time Zone Deviants, Part I: the strangest time zones in the world
- Top 3 Pros and Cons of Daylight Saving Time
- Daylight Saving Time Is America's Greatest Shame
- Why is 11 am + 1 hour == 12:00 pm?
- Times of Day FAQs
- UTC is enough for everyone ...right?
- Falsehoods programmers believe about time
- More falsehoods programmers believe about time; "wisdom of the crowd" edition

- Time to Dump Time Zones
- How Time Works (8 pages)
- Why Aren't We All on the Same Time Zone?
- Is there any technical reason why, in programming, the default date format is YYYYMMDD and not something else?
- China Only Has One Time Zone—and That's a Problem
- The US needs to retire daylight savings and just have two time zones—one hour apart
- America needs to have just two time zones and the world should follow suit
- How Do I Prevent Time Zone Mess-Ups While Traveling?
- Data types, assumptions and how a spacecraft crashed
- There's a Widespread Time Zone Bug in iOS 11 & iOS 12 — Here's What You Need to Know
- How to get updated time zone information for your Apple device
- Why are some countries 30 minutes off the global time zone grid?
- Venezuela and the politics of time
- Call me cuckoo: It's Chavez time in Venezuela
- Handling Time Zone in JavaScript
- 3 simple rules for effectively handling dates and timezones
- The world time zone map
- The world's silliest time zones
- I spent 96 hours on a train from Toronto to Vancouver, and crossing 4 time zones was more disorienting than I ever expected
- One Time Zone for the Whole World?
- Why do we have different time zones on earth? Is time different for different places in the universe?
- Why don't we make a single time for the entire Earth? Instead of having different times for different places, why don't we keep the same time for every place and adjust ourselves accordingly? We can take the time of the GMT as the standard time.
- Mind the gaps
- Why Should We Abolish Daylight Saving Time?
- Why Standard Time is better
- #LockTheClock - Stop Changing Clocks for Daylight Saving Time The official site of the movement to quit changing clocks in and out of DST
- Model Resolution for State Legislatures to Fix Daylight Saving Time Clock-Changing #LockTheClock
- Ditch the switch? Call to go on permanent daylight saving time grows
- Where to hate daylight saving time and where to love it
- Time Has No Meaning at the North Pole
- How to keep time: there's more to time zones than first meets the eye
- It's Time Again To Change The Way We Tell Time
- It's Time to Adopt Cosmic Time, One Time for the Entire World
- Andrew Coyne: Forget the daylight savings time debate, we need to get rid of time zones altogether
- 4 Time Zone Bugs I Ran Into
- Yes, we do know what day it is (but we probably won't say so)
- Temporal
- Temporal Cookbook
- Time Zones and Resolving Ambiguity
- Fixing JavaScript Date – Getting Started
- Fixing JavaScript Date – Web Compatibility and Reality
- Falsehoods programmers believe about time zones
- Coding for Time Zones & Daylight Saving Time — Oh, the Horror
- The world's only ethnic time zone
- People urged to switch off auto date and time settings on their phones
- The mysteries of BCL time zone data

- Storing UTC is not a silver bullet (highly recommended reading)
- Common mistakes in date/time formatting and parsing
- What would happen if we abolished time zones altogether?
- The clocks change tonight – but it won't happen for much longer
- Hanke Henry On Time (with the Hanke-Henry Permanent Calendar)
- Zulu and UTC: the story behind aviation's time zone
- One Time Zone for the World? (with many enjoyable comments)
- Daylight Saving Time and Artificial Time Zones – A Battle Between Biological and Social Times
- Permanent Standard Time vs. Daylight Time — The Ultimate Guide
- Let's Get Rid of Time Zones, Not Just Daylight Savings
- Why Europe Couldn't Stop Daylight Saving Time
- This Furious Icelandic Man Sums Up Exactly How We Feel About Daylight Saving Time
- JavaScript: Working with Time Zones
- Is Luxon the Heir to the Moment.js Throne?
- If So Many People Loathe Daylight Saving Time, Why Hasn't Someone Done Something About It? Well, They've Tried
- How to read time zones on your plane ticket
- The international date line, explained
- Date and Time on the Internet: Timestamps (RFC 3339)
- Temporal: getting started with JavaScript's new date time API
- 'EST' and 'EDT' Are Different Time Zones
- Your Calendrical Fallacy Is... Helping you navigate the insane complexity of calendrically correct date and time operations
- The Reason for Daylight Savings: Blessing or a Nuisance?
- Why is Programming with Dates So Hard? How to think about dates and timestamps, timezones, and offsets
- The Largely Untold Story Of How One Guy In California Keeps The World's Computers On The Right Time Zone. (Well, Sort Of)
- tz database community up in arms over proposals to merge certain time zones
- The Timing of Sunrise Within U.S. Time Zones . . . and SAD
- The Kobayashi Maru of Comparing Dates with Times
- Temporal Date API Ultimate Guide
- The History Of Daylight Saving Time Searching for a good answer to justify our collective madness
- Where Mornings Would Get Darker Under Permanent Daylight Savings Time
- Bringing Coda up to date: 25-hour days and other things I learned when migrating from moment date library
- Semantics of timezone-aware datetime arithmetic
- Date and Time Pickers for All
- How to deal with dates and times without any timezone tantrums…
- Serious Security: The decade-ending "Y2K bug" that wasn't
- Serious Security: What we can all learn from #PiDay
- DOT to map out nation's time zones after report shows no official map exists
- DOT Can Improve Processes for Evaluating the Impact of Time Zone Changes and Promoting Uniform Time Observance [PDF]
- Time in Databases
- How to Handle Database Timezones
- Daylight saving time causes confusion in some parts of Mexico and the U.S.
- My American Airlines flight was on time. How did I arrive on the wrong day?!
- This Guy Says Getting Rid of Time Zones Will Improve Everyone's Life
- Standard Time vs. Daylight Time
- Timestamps and Time Zones in PostgreSQL
- What is a valid use case for using TIMESTAMP WITHOUT TIME ZONE?

- A Giant Leap for the Leap Second. Is Humankind Ready?
- Horrible UX: Selecting Time Zone in Google & Apple Calendar
- On Time, Time Zones, and Software
- A Designer's Guide To Time Zone Selection UX
- It's Time We Addressed Time-Zone Selectors
- Case study: The UX of selecting timezones and the follow-up article How to group time zones
- Which way should I display time zone names to make them easily pickable?
- How to make selecting a timezone more user-friendly?
- Why is subtracting these two epoch-milli Times (in year 1927) giving a strange result?

30.11 Podcasts

- CodePen - Time Zones 2019-02-20
- Stuff You Should Know - Short Stuff: Time Zones 2019-08-07
- The Complete Developer Podcast - DateTime Part 1: History of Time 2017-12-07
- The Complete Developer Podcast - DateTime Part 2: Computing and Time Protocols 2017-12-14
- The Complete Developer Podcast - DateTime Part 3: Best Practices 2017-12-21
- The Complete Developer Podcast - Dates, Times, and User Intents 2021-05-27
- Syntax - Hasty Treat - Temporal Date Objects in JavaScript 2020-10-26
- no dogma podcast - #60 Jon Skeet (part 1), Noda Time 2016-10-17
- TalkPython - Unlock the mysteries of time, Python's datetime that is! 2020-07-04
- The Hanselminutes Podcast - The Problem with DateTime - NodaTime with Matt Johnson 2015-07-24

30.12 Tools

- Every Time Zone
- Time.is
- timezon.es
- GreenwichMeanTime.com
- No Time Zone

31 Now to something completely different

The reason why we decided to have a *year* of 365 *days* is due to the Earth orbiting around the Sun. But the reasons to why we decided to divide the year in 12 months with 28-31 days per month and why we decided that a day should be 24 hours, with 60 minutes per hour and 60 seconds per minute are not so simple to understand or agree with.

But these will be completely different discussions...

First published by Anders Gustafson 2017-04-13 and continuously revised. Many thanks to Dr. Eno Zemit, Prof. Zoe Menti and Prof. Tim Ézoné for valuable discussions and guidance. This compilation is dedicated to Galileo Galilei. Always remember “E pur si muove”.

Page last revised 2024-01-14T14:18:16Z
URL: https://anders.nemonisimors.com
Email: anders *at* nemonisimors *dot* com
GitHub: epursimuove
Twitter: AndersNNM

Nemo nisi mors