It’s About Time

by Benjamin Fleischer

A short journey through time with Ruby

@hazula
github/bf4
bf@benjaminfleischer.com
benjamin@mrskin.com

What is Time?

Time in a computer is:

- Measured as microseconds form the Unix Epoch
  - **(January 1, 1970 00:00:00 UTC)**
  - Some systems allow it to be negative. (A non-portable feature)

- UTC stands for “Coordinated Universal Time”
  - It replaced the designation of GMT (Greenwich Mean Time)

For Ruby

- Ruby dates can be, starting January 1, 4713 BCE, in the format:
  - civil, (aliased to :new)
  - ordinal,
  - commercial,
  - Julian, and
  - standard,

Ruby supports some standard formats (*with require 'time', an extension of Class Time*)

rfc2822 (e-mail)	Thu Apr 01 16:32:45 CST 2004
rfc822 (ARPA Internet Text Messages)	Thu, 01 Apr 2004 16:32:45 -0600
httpdate (RFC 2616, rfc1123-date), always UTC	Thu Apr 01 16:32:45 UTC 2004
iso8601 / xmlschema	2004-04-01T16:32:45-06:00

For Ruby

- Note that iso8601 is the both human-readable and machine, unambiguous, and sortable as a string. It is the format used in the microformats.org specification
- UTC 'timezone' designator is "Z" e.g 1994-11-05T13:15:30Z corresponds to 1994-11-05T08:15:30-05:00
- Use ‘tzinfo’ gem to work with zone
- **Suggestions**
  - Always store your dates or times as iso8601 xmlschema in UTC see http://devblog.avdi.org/2009/10/25/iso8601-dates-in-ruby/ and note http://www.w3.org/TR/NOTE-datetime

Time is all around

But you can turn it back
one hour at 2 a.m. on Oct. 28
when daylight-saving time
ends and standard time begins.

Gotchas:
why we can't have ~~nice things~~ good times

Be careful with:

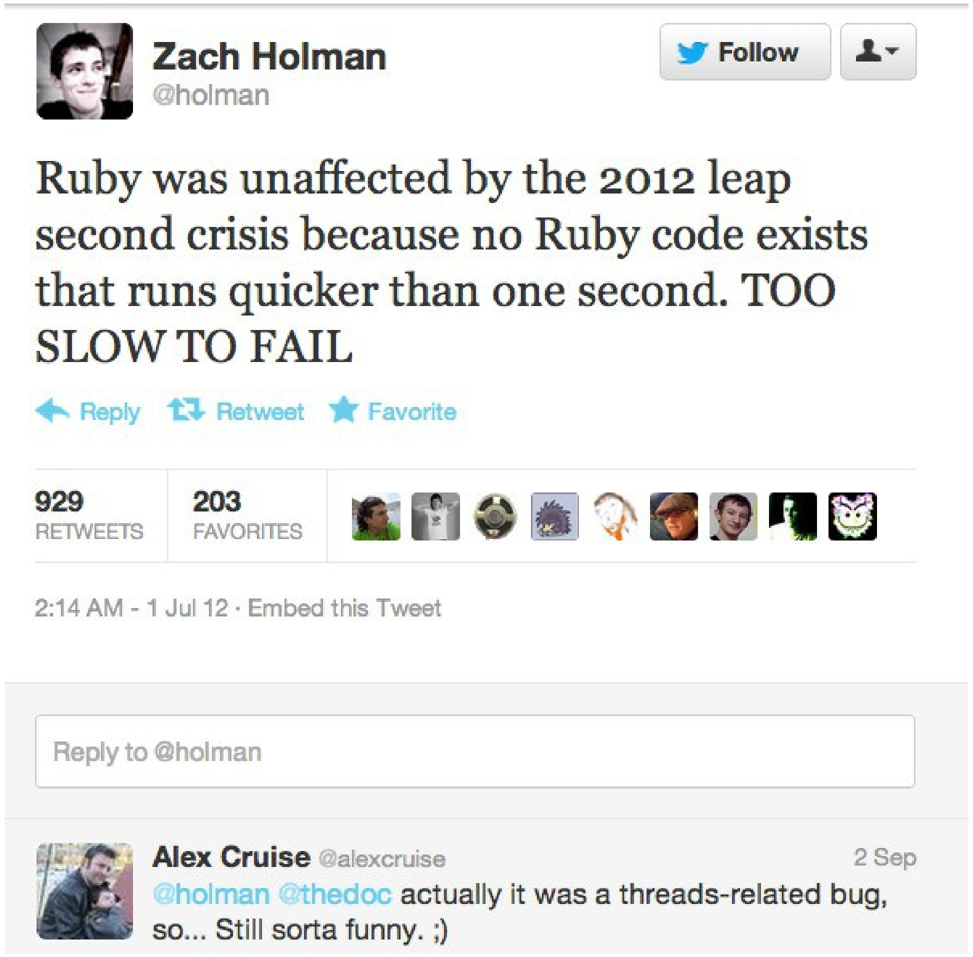
- Daylight savings
  - Spring Forward: On the second Sunday in March 2 a.m. becomes 3 a.m. and daylight time begins. **2 a.m. doesn't exist!**
  - Fall Back: On the first Sunday in November, 2 a.m. becomes 1 a.m. local standard time. **2 a.m. occurs twice!**
- Using localized times in your app, or in a script
  - *12 a.m. Wednesday March 28th in Chicago* is *Tuesday March 27th in Denver* and Palo Alto
- Leap seconds, or other last minute changes
  - (next slide)
- Fuuuuuu! (h/t Peter Cooper ruby19 walkthrough)
  - require 'time'; (else get undefined method `parse' for Time:Class)
  - Time.parse("30/12/2001"): in Ruby 1.8 ArgumentError, '12/30/2001' works
  - in 1.9.2 parses as dd/mm/yyyy .. was mm/dd/yyyy in 1.8!

Gotchas:

why we can’t have ~~nice things~~ good times

require 'time'	1.8	1.9
Time.parse("30/12/2001")	ArgumentError	2001-12-30 00:00:00 -0600
Time.parse("12/30/2001")	Sun Dec 30 00:00:00 -0600 2001	ArgumentError
Time.parse(“1/2/2001”)	Tue Jan 02 00:00:00 -0600 2001	2001-02-01 00:00:00 -0600
Time.parse('2001-01-02')	Tue Jan 02 00:00:00 -0600 2001	2001-01-02 00:00:00 -0600

For Ruby, leap second



Zach Holman
@holman

Ruby was unaffected by the 2012 leap second crisis because no Ruby code exists that runs quicker than one second. TOO SLOW TO FAIL

929 RETWEETS
203 FAVORITES

2:14 AM - 1 Jul 12 · Embed this Tweet

Alex Cruise @alexcruise 2 Sep
@holman @thedoc actually it was a threads-related bug, so... Still sorta funny. ;)

Solutions

- Don’t schedule cron jobs between 1 a.m. and 3 a.m.
  as that time may not exist
- Be careful scheduling cron jobs around midnight
  that depend on the current date
  as the current date changes in different time zones

640k ought to be enough for anyone – “Bill Gates”

- A *32-bit* machine can only go up to 3:14:07 UTC on Tuesday, 19 January *2038* before there’s an integer overflow. Problems began in 2006 in the AOLServer
  - *64-bit* machines should be good until *292 billions years from now*
- **Test your machine bits in ruby**
  - Could be useful if you run specs with dates > 2038 on multiple machines
  - (1.size*8 == 32)

Timezones

```
require 'rubygems' # required in 1.8

require 'tzinfo'

def time_in_zone(time,zone)

  tzinfo = zone.respond_to?(:current_period) ? zone : TZInfo::Timezone.get(zone) rescue guess_tz(zone)

  offset = tz_offset(tzinfo)

  if RUBY_VERSION <  '1.9'

    tzinfo.utc_to_local(time.utc)

  # Note that the Time returned will look like it is UTC (Time.zone will return "UTC"). This is because it is not currently possible to change the offset of an individual Time instance.

  else

    time.localtime(offset)

  end

end

def tz_offset(tzinfo)

  '%.02d:00' % (tzinfo.current_period.utc_total_offset / 60 / 60 ) # e.g. '-06:00'

end

def guess_tz(zone_guess)

  guess = zone_guess.to_s.split('/')[-1]

  TZInfo::Timezone.us_zones.detect {|tz| tz.name =~ /#{guess}/i }

end
```

```
require 'tzinfo'

datetime  = '2012-11-03T10:00:00-06:00'

time = Time.parse(datetime)

tzinfo = TZInfo::Timezone.get('Pacific/Honolulu') # #<TZInfo::TimezoneProxy: Pacific/Honolulu>

offset =  '%.02d:00' % (tzinfo.current_period.utc_total_offset / 60 / 60)  # "-10:00"

localtime_from_utc = tzinfo.utc_to_local(time.utc)

localtime_from_offset = time.localtime(offset) # 1.9 only

localtime_from_utc.xmlschema

 => "2012-11-03T06:00:00Z"

localtime_from_offset.xmlschema

 => "2012-11-03T06:00:00-10:00" # and the time object is changed, too
```

Timezones in Rails

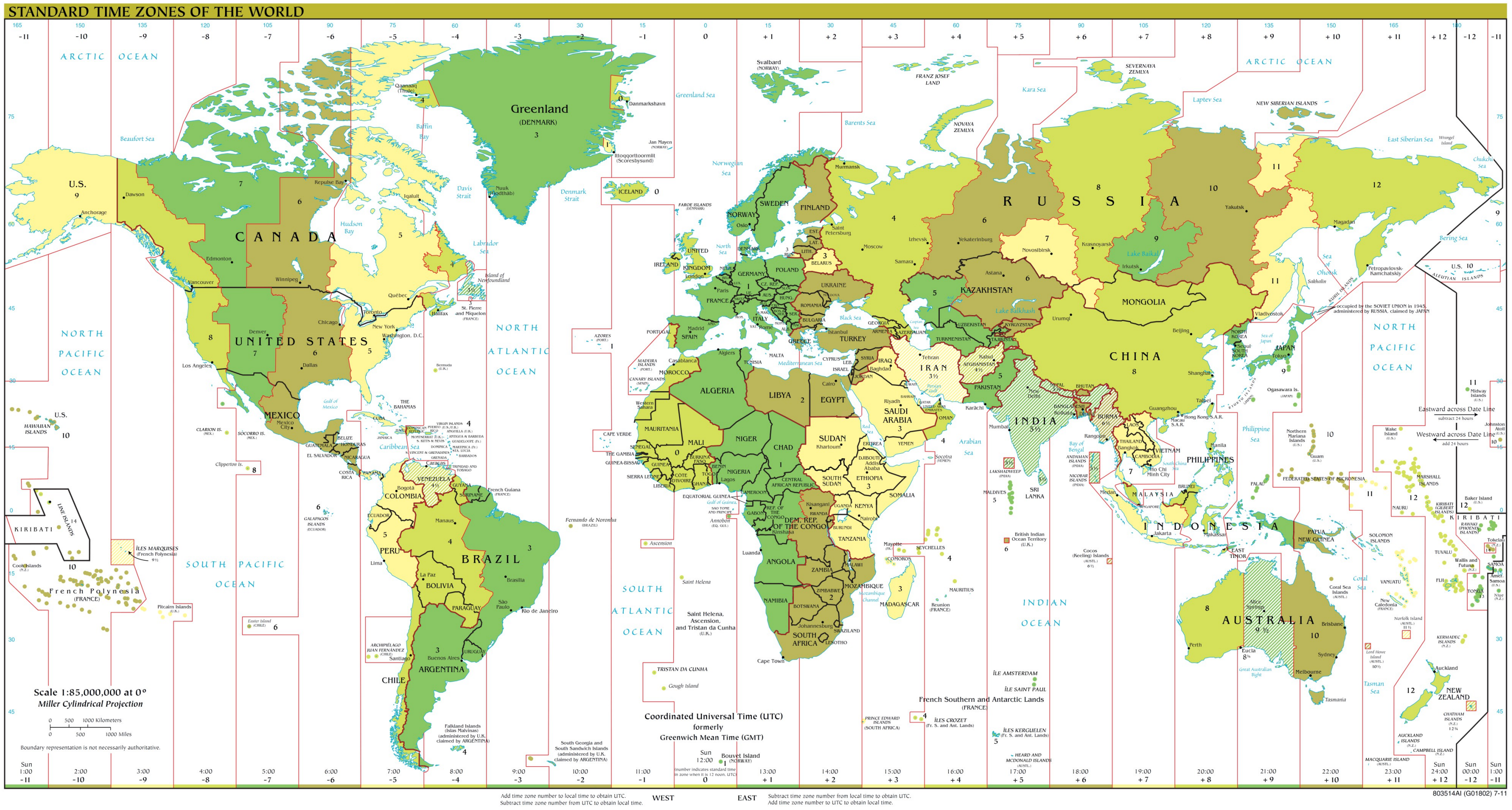
```
def get_tzinfo_zone_from_rails(tz ='America/Chicago')
  zone = rails_friendly_name_zone_from_tz(tz)
  Time.zone = zone
   # Time.zone.class => ActiveSupport::TimeZone
   Time.zone
end
def rails_friendly_name_zone_from_tz( tz =
'America/Chicago' )
      ActiveSupport::TimeZone::MAPPING.detect {|
rails_zone_key,tz_info_name| tz_info_name == tz }.first
end
def the_zone_i_need(rails_friendly_name)
  Time.zone = rails_friendly_name
  Time.zone
end
#  Can only Cceate ActiveSupport::TimeWithZone
instances via TimeZone's +local+, +parse+, +at+ and
+now+ methods.
```

```
def demonstrate_zone(time,tz_info)
  Time.zone = nil
  Time.zone = tz_info
  rails_friendly_name =
rails_friendly_name_from_tz(tz_info)
  Time.zone = rails_friendly_name
  zone = the_zone_i_need(rails_friendly_name)
  zone.parse(time).xmlschema
end
```

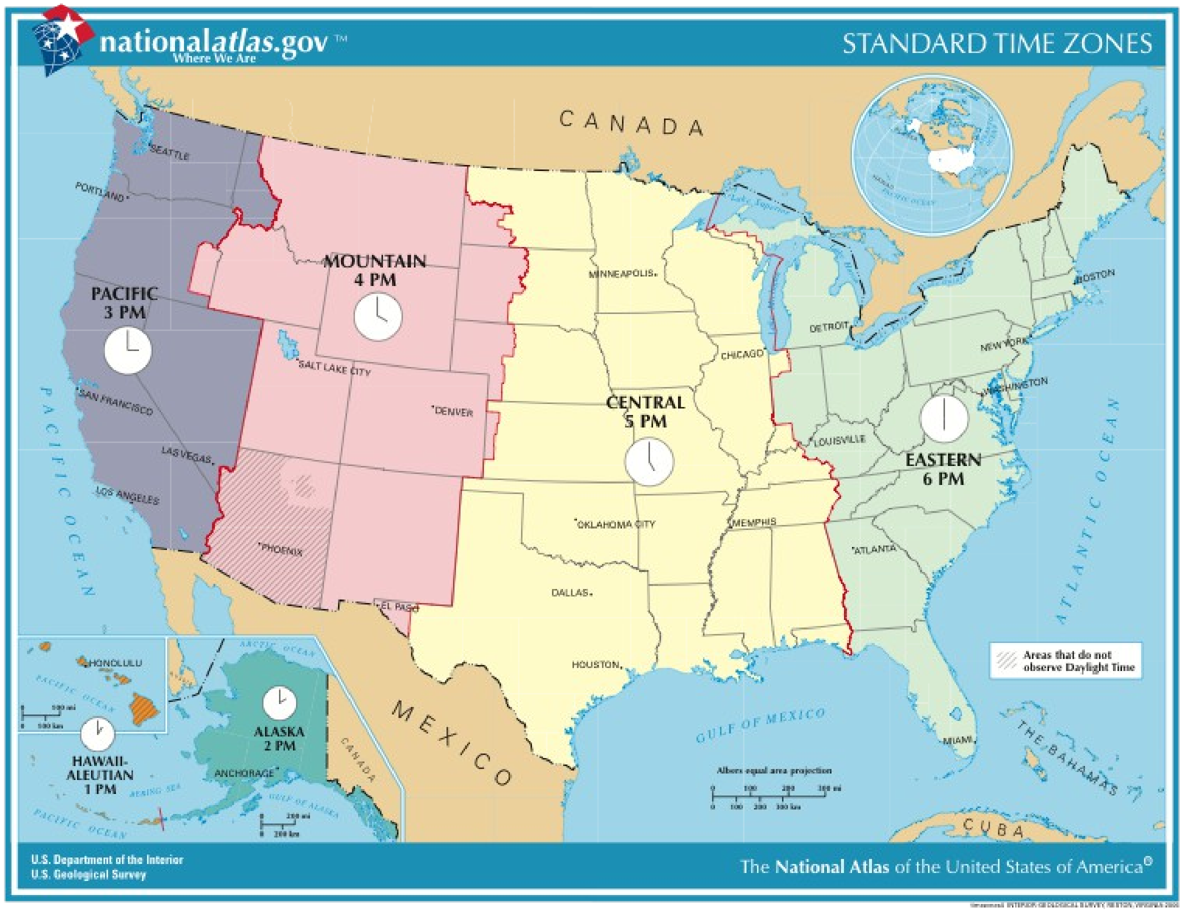
Timezones in Rails

- ActiveSupport::Timezone::MAPPING
  - rails_friendly_zone_name = ActiveSupport::TimeZone::MAPPING.detect {|rails_zone_key,tz_info_name| tz_info_name == 'America/Chicago' }.first
  - The TimeZone class serves as a wrapper around TZInfo::Timezone instances. It allows us to do the following
  - Retrieve and display zones with a friendlier name (e.g., “Eastern Time (US & Canada)” instead of “America/New_York”).
  - Adds Time.zone etc
- ActiveSupport::TimeWithZone
  - A Time-like class that can represent a time in any time zone. Necessary because standard Ruby Time instances are limited to UTC and the system’s ENV[‘TZ’] zone.
  - You shouldn’t ever need to create a TimeWithZone instance directly

World Timezones


STANDARD TIME ZONES OF THE WORLD
Coordinated Universal Time (UTC) formerly Greenwich Mean Time (GMT)
Scale 1:85,000,000 at 0°
Miller Cylindrical Projection
Boundary representation is not necessarily authoritative.
WEST
EAST
Add time zone number to local time to obtain UTC. Subtract time zone number from UTC to obtain local time.
Subtract time zone number from local time to obtain UTC. Add time zone number to UTC to obtain local time.
803514AI (G01802) 7-11

US Timezone


nationalatlas.gov™
Where We Are
STANDARD TIME ZONES
CANADA
PACIFIC 3 PM
MOUNTAIN 4 PM
CENTRAL 5 PM
EASTERN 6 PM
ALASKA 2 PM
HAWAII-ALEUTIAN 1 PM
SEATTLE
PORTLAND
SAN FRANCISCO
LAS VEGAS
LOS ANGELES
SALT LAKE CITY
PHOENIX
DENVER
EL PASO
MINNEAPOLIS
CHICAGO
OKLAHOMA CITY
DALLAS
HOUSTON
MEMPHIS
DETROIT
LOUISVILLE
ATLANTA
NEW YORK
WASHINGTON
BOSTON
MIAMI
HONOLULU
ANCHORAGE
PACIFIC OCEAN
ATLANTIC OCEAN
GULF OF MEXICO
MEXICO
CUBA
THE BAHAMAS
Areas that do not observe Daylight Time
Albers equal area projection
U.S. Department of the Interior
U.S. Geological Survey
The National Atlas of the United States of America®

Citations

- #Timezones
- * World Map http://en.wikipedia.org/wiki/File:Standard_time_zones_of_the_world.png
- * US Map http://nationalatlas.gov/printable/images/pdf/reference/timezones4.pdf
- * US 2005 Energy policy act http://www1.eere.energy.gov/femp/pdfs/epact_2005.pdf
- * TZ database http://en.wikipedia.org/wiki/Tz_database

- # Cartoons
- * You can't stop time http://en.wikipedia.org/wiki/File:Daylightsavings.svg
- * Sleep Cycle http://en.wikipedia.org/wiki/File:SpringFwd-FallBack.jpg
- * Get your hoes ready http://en.wikipedia.org/wiki/File:Victory-Cigar-Congress-Passes-DST.jpeg

- # Time formats
- * http://www.w3.org/TR/NOTE-datetime
- * http://tzinfo.rubyforge.org/doc/files/README.html
- * http://www.twinsun.com/tz/tz-link.htm

- # Quotes
- * Bill Gates http://en.wikiquote.org/wiki/Bill_Gates
- * Zach Holman leap second tweet https://twitter.com/holman/status/219328090021703681

- # Ruby
- ##Rails
- * https://github.com/rails/rails/blob/master/activesupport/lib/active_support/time_with_zone.rb
- * https://github.com/rails/rails/blob/master/activesupport/lib/active_support/time.rb
- * https://github.com/rails/rails/blob/master/activesupport/lib/active_support/values/time_zone.rb

Links

- http://time.is/
- http://strfti.me/
- Ruby code for this talk (WIP)
  https://gist.github.com/3668333

Citations

- #Timezones
- * World Map http://en.wikipedia.org/wiki/File:Standard_time_zones_of_the_world.png
- * US Map http://nationalatlas.gov/printable/images/pdf/reference/timezones4.pdf
- * US 2005 Energy policy act http://www1.eere.energy.gov/femp/pdfs/epact_2005.pdf
- * TZ database http://en.wikipedia.org/wiki/Tz_database

- # Cartoons
- * You can't stop time http://en.wikipedia.org/wiki/File:Daylightsavings.svg
- * Sleep Cycle http://en.wikipedia.org/wiki/File:SpringFwd-FallBack.jpg
- * Get your hoes ready http://en.wikipedia.org/wiki/File:Victory-Cigar-Congress-Passes-DST.jpeg

- # Time formats
- * http://www.w3.org/TR/NOTE-datetime
- * http://tzinfo.rubyforge.org/doc/files/README.html
- * http://www.twinsun.com/tz/tz-link.htm

- # Quotes
- * Bill Gates http://en.wikiquote.org/wiki/Bill_Gates
- * Zach Holman leap second tweet https://twitter.com/holman/status/219328090021703681

- # Ruby
- ##Rails
- * https://github.com/rails/rails/blob/master/activesupport/lib/active_support/time_with_zone.rb
- * https://github.com/rails/rails/blob/master/activesupport/lib/active_support/time.rb
- * https://github.com/rails/rails/blob/master/activesupport/lib/active_support/values/time_zone.rb

I’ve got all the time in the world

- Thanks

MR SKIN Fast-Forwarding to the Good Parts SINCE 1999!

Benjamin Fleischer

@hazula

github.com/bf4

bf@benjaminfleischer.com

benjamin@mrskin.com

http://bit.ly/bf4-talks