Dates

Topic 9.5

Dr. Douglas Stebila

Department of Computing and Software
McMaster University

Spring/Summer 2020

Table of Contents

Introduction

datetime

Timezones

Table of Contents

Introduction

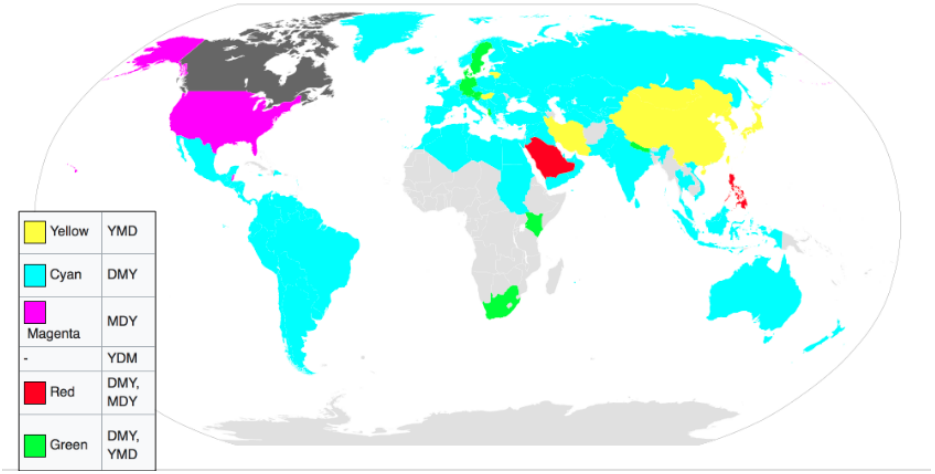
datetime

Timezones

Working with dates

- There are many calendar systems
  - Current: Gregorian, Bengali, Chinese, Ethopian, Hebrew, Hindu, Iranian, Islamic, ...
  - Historical: Julian, ...
- Within a calendar system, there are many complications:
  - Daylight saving time
  - Leap years / leap seconds
  - Time zones
  - Different and possibly ambiguous string representations of dates
  - Y-M-D, D-M-Y, M-D-Y, etc.

Date format by country



The only country in the world where YMD, DMY, and MDY are all commonly used. SAD!

`https://en.wikipedia.org/wiki/Date_format_by_country`

Table of Contents

Introduction

datetime

Timezones

Working with dates and times in Python

The datetime module provides classes to represent dates and times, and to manipulate them

- datetime.date: Class representing dates in Gregorian calendar (y, m, d)
- datetime.time: Class representing a time on an abstract day of 24*60*60 seconds (h, min, sec, microsecond, timezone)
- datetime.datetime: Class representing a date and a time on that date
- datetime.timedelta: Class representing a difference between two time/date objects for arithmetic

Creating a datetime object

```
from datetime import datetime
rightnow = datetime.now()
midterm = datetime(2018, 3, 14, 15, 30)
timeleft = midterm - rightnow
print(timeleft)
print(type(timeleft))
```

Converting datetime objects to strings

- There are many string formatting options to print parts of a datetime object
- Use the strftime method of the datetime object

```
print(midterm.strftime("%Y-%m-%d %H:%M:%S"))
2018-03-14 15:30:00
print(datetime.now().strftime("Today is %A"))
Today is Wednesday
```

https://docs.python.org/3/library/datetime.html#strftime-strptime-behavior

http://strftime.org/

Creating datetime objects from strings

- Can use the strptime method to parse a string into a datetime object
- But need to specify the exact format being parsed
- Uses the same string formatting options as strftime

```
christmas = datetime.strptime("Dec 25, 2017 12:00:00",
                              "%b %d, %Y %H:%M:%S")
print(christmas)
print(type(christmas))
2017-12-25 12:00:00
<class 'datetime.datetime'>
```

Table of Contents

Introduction

datetime

Timezones

Timezones

- Have to be careful if dealing with dates/times that may involve multiple timezones
- Python datetime objects can have a timezone set for them
- And then it is possible to convert between one timezone and another

I won't ask you to work with timezones in this course.

Timezones

- Recommended approach: keep all Python datetime objects in UTC time
- Convert to relevant timezone for printing
- Easiest to use an external package (pytz) to help with timezones
  - Can handle daylight saving time, leap years, etc.
- GMT = Greenwich Mean Time – time in Greenwich, England
- UTC = Coordinated Universal Time – generally accepted international time format, roughly equivalent to UTC

Complex example involving timezones

```
from datetime import datetime, timedelta
import pytz
tz = pytz.timezone('Canada/Eastern')
before = datetime(2018, 3, 11, 6, 59, 59, 0,
tzinfo=pytz.utc)
onesec = timedelta(seconds=1)
after = before + onesec
fmt = '%Y-%m-%d %H:%M:%S %Z%z'
print(before.astimezone(tz).strftime(fmt))
print(after.astimezone(tz).strftime(fmt))
```

Even more complex examples

```
s = '{"screenName": "realDonaldTrump", "id":
"970650759091163137", "time": "2018-03-05T13:22:29.000Z",
"isRetweet": false, "isPinned": false, "isReplyTo": false,
"text": "Why did the Obama Administration start an
investigation into the Trump Campaign (with zero proof of
wrongdoing) long before the Election in November? Wanted
to discredit so Crooked H would win. Unprecedented. Bigger
than Watergate! Plus, Obama did NOTHING about Russian
meddling.", "userMentions": [], "hashtags": [], "images":
[], "urls": [], "replyCount": 28055, "retweetCount":
16933, "favoriteCount": 55177}'
```

```
import dateutil
import json
t = json.loads(s)
day = dateutil.parser.parse(t['time']).strftime("%A")
print(day)
Monday
```

Successfully converts timestamp to a day. But doesn't take into account the timezone – this is the date in UTC timezone, not our local timezone or the original poster's timezone.

https://dateutil.readthedocs.io/