App1: The Other Side of the World

Outside packages:

# folium # Map tool

from folium import Map, Marker, Popup

Map = Map( \*list of lat and lng\* ) # show to locations on the world map

point = Marker( location = [ \*list of lat and lng\* ], \*popup = ‘ \*str\*’ )

point.add\_to(Map) #create a / a list of marker on the location

Popup = Popup( \*str \* )

popup .add\_to(Map) # add a popup window

Map.save( \* str\* .html ) # save the map instance into a HTML file, named ‘\* str\*.html’

# datetime, pytz, timezonefinder # Time tool

from datetime import datetime

from pytz import timezone

from timezonefinder import TimezoneFinder

tf= TimezoneFinder()

tf.closest\_timezone\_at( \*lat= FLOAT, lng= FLOAT\* ) # return str of the \*LOCATION

birth= datetime(\*year, \*month, \*day, \*\*..) # create a time

death= datetime(\*year, \*month, \*day, \*\*..)

(death - birth).days # time can be subtracted

datetime.now() # No para needed if pytz is not imported & return the local time

tz= timezone( “\* LOCATION \*” ) # [HERE](https://stackoverflow.com/questions/13866926/is-there-a-list-of-pytz-timezones) to see the entire list of Timezone

datetime.now(tz) # return the LOCATION time.

# sunnyday # Weather tool

from sunnyday import Weather

# step 1, go to [*openweathermap.org*](https://openweathermap.org/)to get your APIkey

My key = “ f6f970013f0f39ea27a183b04c0bd0ba ”

weather= Weather(apikey= “ \* APIKEY \* ”, lat= \*FLOAT, lon = \*FLOAT, \*city= “ \*CITY\* ”)

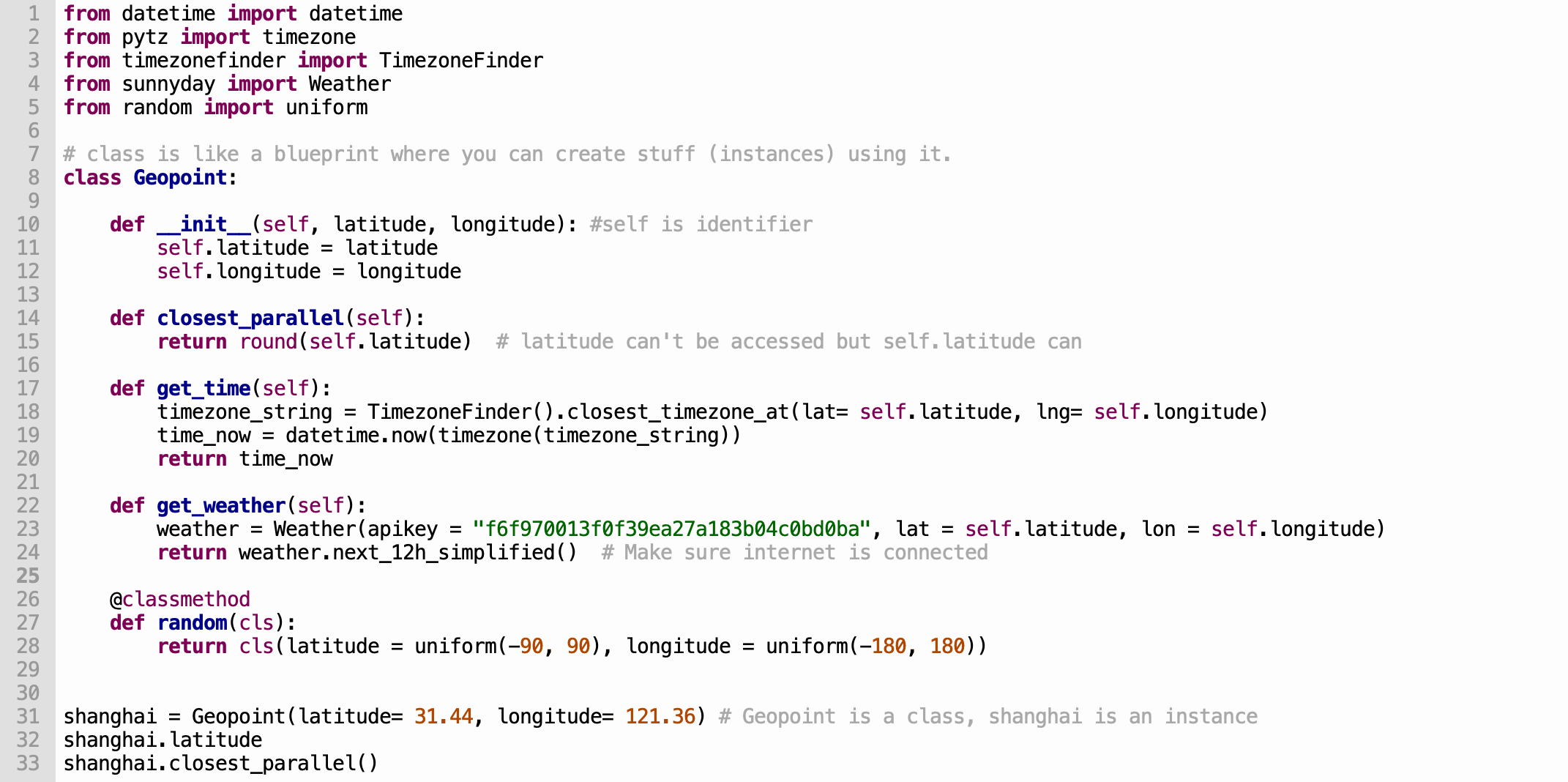
weather.next\_12h\_simplified() #return a list of weather in the following 12hrs

# random # random number tool (built\_in\_pack)

from random import uniform

uniform(\*int, \*int) # return random FLOAT between

Glossary (Class):



# Class

line 8 ~ 28

# Instance

shanghai

# Method

All of the defs (method usually will have output)

# Constructor

def \_\_init\_\_ (...) (to construct the instance, usually be \_\_init\_\_ )

# Instance method

line 10 ~24 (to be applied to the instance)

\*Note: \_\_init\_\_ is both constructor and instance method

# Class method \*

def random(cls) (not related to an instance, but aim to create an object instance)

\*Note: need *@classmethod* decorator

# Parameters

def\_\_init\_\_**(self, latitude, longitude)**:

\*Note: refer to the context in method

# Arguments

Geopoint( **latitude=** 31.44, **longitude=** 121.36 )

\*Note: refer to the context in call

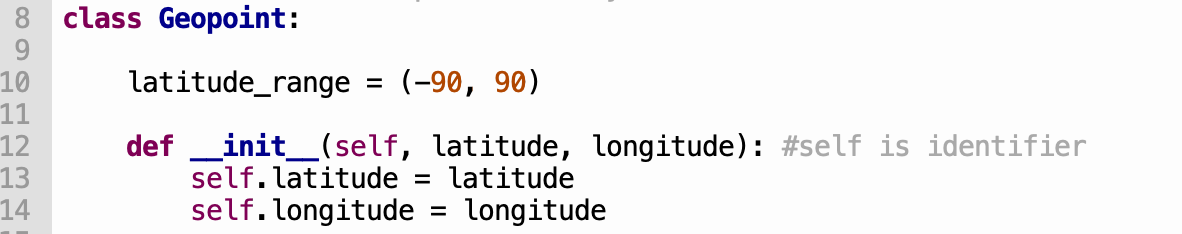
# Argument values

Geopoint( latitude= **31.44**, longitude= **121.36** )

# Instance variables

self.XXX ( values referring to the object instance inside the class )

# Class variable \*



line 10 (built inside the class level, **\*\*it won’t change as instance variables change\*\*** )

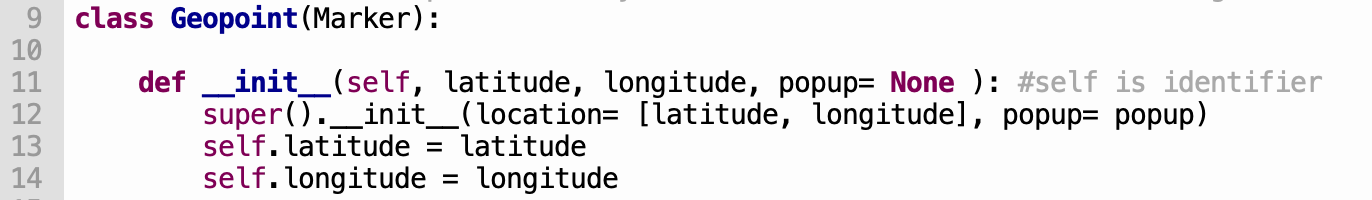
\*Note: class variables belong to both class and instance. **To call it by using “self.XX”**

# Attributes

line 32& 33. ( Operations can be made )

Inheritance\* :

# Parent & Child:



Parent: Marker

Child: Geopoint

# Functions:

Child will inherit all methods and properties from its parent.

String (Supplement):

dir(str) to see all method, help(str.XX) to see all explanation

# Method:

name= ‘john smith’

name.capitalize()

name -> ‘John smith’

name.title()

name-> ‘John Smith’

name.lower()

name-> ‘john smith’ # same with str.upper()

name.split() or name.split(‘ ’)

name-> [‘john’ ‘smith’] --> list

name.count(‘h’) -> 2

name.strip(‘h’) -> ‘john smit’ #only remove all the leading or trailing char or whitespace

name.replace(‘ ’, ‘\_’)

name-> ‘john\_smith’

number= ‘2’

number.zfill(2) #fill with 0, para is the width

number-> ‘02’

# Magic Method: (\_\_XX\_\_())

name= ‘john ’

name.\_\_add\_\_(‘smith’) <--> name + ‘smith’

name-> ‘john smith’

name.\_\_contains\_\_(‘o’) -> True

name.\_\_eq\_\_(‘mary’) <--> name == ‘mary’ -> False

name.\_\_getitem\_\_(2) <--> name[2] -> ‘h’

# Slicing:

filename= ‘nice\_taco.jpg’ # slicing starts at 0 from L->R, starts at -1 from R->L

filename[:3] -> ‘nic’ # upperbound included & lowerbound excluded

filename[-3:] -> ‘jpg’

# Print ‘f-string’ Trick: \*

user= ‘John’

sentence = ‘User’ + user + ‘has logged in.’

sentence = f‘User {user} has logged in.’ \*\* RECOMMANDED\*\*

List, Tuple & Range

# Difference:

List ---> mutable & Tuple ---> immutable

# Common:

Slicing method applies to them ALL.

HTML coding:

# Insert a pic:

<img src= ‘ \*Image URL\*’ width= \*int > # right click a img, select ‘Copy Image Address’

# Horizontal line:

<hr>