Random joke teller

#!/usr/bin/env python3.7

import requests

import json

from datetime import datetime

def get\_joke():

url = "https://08ad1pao69.execute-api.us-east-1.amazonaws.com/dev/random\_joke"

params = { }

headers = { 'Content-Type': 'application/json' }

response = requests.get(url, headers=headers, params=params)

if response.status\_code == 200:

return json.loads(response.content.decode("utf-8"))

else:

print("\*\*\* ERROR! Response ",response.status\_code," \*\*\*")

return None

answer = input("Do you want to hear a joke? (y/n) ")

while answer == "y":

joke = get\_joke()

print( joke["setup"] )

input()

print( joke["punchline"] )

print("")

answer = input("Do you want to hear a joke? (y/n) ")

City latitude & longitude finder

#!/usr/bin/env python3.7

import requests

import json

from datetime import datetime

def get\_city\_coordinates( location ):

url = "https://geocode.xyz/"+location+"?json=1"

params = { }

headers = { 'Content-Type': 'application/json' }

response = requests.get(url, headers=headers, params=params)

if response.status\_code == 200:

return json.loads(response.content.decode("utf-8"))

else:

print("\*\*\* ERROR! Response ",response.status\_code," \*\*\*")

return None

city = input("Type the name of a city: ")

info = get\_city\_coordinates(city)

print("City: ",info["standard"]["city"])

print("Country: ",info["standard"]["countryname"])

print("Latitude: ",info["latt"])

print("Longitude: ",info["longt"])

Next SpaceX launch

#!/usr/bin/env python3.7

import requests

import json

from datetime import datetime

def get\_next\_spacex():

url = "https://api.spacexdata.com/v2/launches/next"

params = { }

headers = { 'Content-Type': 'application/json' }

response = requests.get(url, headers=headers, params=params)

if response.status\_code == 200:

return json.loads(response.content.decode("utf-8"))

else:

print("\*\*\* ERROR! Response ",response.status\_code," \*\*\*")

return None

next\_launch = get\_next\_spacex()

if next\_launch != None:

# Convert timestamp to human readable date

launch\_timestamp = next\_launch["launch\_date\_unix"]

launch\_dt = datetime.fromtimestamp(launch\_timestamp)

launch\_date\_string = launch\_dt.strftime("%d/%m/%Y %H:%M:%S")

# Print info

print("Next flight: ",next\_launch["mission\_name"])

print("Date/time: ",launch\_date\_string,"(your timezone)")

print("Rocket: ",next\_launch["rocket"]["rocket\_name"])

print("Launch site: ",next\_launch["launch\_site"]["site\_name\_long"])

Swiss public transport info

#!/usr/bin/env python3.7

import requests

import json

from datetime import datetime

# documentation at

# http://transport.opendata.ch/docs.html

def get\_station\_info( location ):

url = "http://transport.opendata.ch/v1/locations"

params = { "query" : location, "type" : "station" }

headers = { 'Content-Type': 'application/json' }

response = requests.get(url, headers=headers, params=params)

if response.status\_code == 200:

return json.loads(response.content.decode("utf-8"))

else:

print("\*\*\* ERROR! Response ",response.status\_code," \*\*\*")

return None

def get\_station\_departures( name, id ):

url = "http://transport.opendata.ch/v1/stationboard"

params = { "station": name, "id" : id }

headers = { 'Content-Type': 'application/json' }

response = requests.get(url, headers=headers, params=params)

if response.status\_code == 200:

return json.loads(response.content.decode("utf-8"))

else:

print("\*\*\* ERROR! Response ",response.status\_code," \*\*\*")

return None

stopname = input("Type the name of a Swiss train or bus stop: ")

# Step 1 - Search for the station, get it's ID

info = get\_station\_info(stopname)

if info != None and "stations" in info:

station = info["stations"][0] # Assume the first result is correct ???

print("Found station: ", station["name"]," with id ",station["id"])

# Step 2 - Use the ID to search for next departures from station

departures = get\_station\_departures( station["name"], station["id"] )

if departures != None and "stationboard" in departures:

print("Next departures from "+station["name"])

for service in departures["stationboard"]:

# Collect info

name = service["name"]

route\_num = service["number"]

category = service["category"]

destination = service["to"]

platform = service["stop"]["platform"]

# Convert timestamp to a human readable date and time

departure\_timestamp = service["stop"]["departureTimestamp"]

departure\_date = datetime.fromtimestamp(departure\_timestamp)

departure\_text = departure\_date.strftime("%H:%M:%S")

# Print info

if platform is None:

print(category+" "+route\_num+" to "+destination+" departs at "+departure\_text)

else:

print(category+" "+route\_num+" to "+destination+" departs at "+departure\_text+" from platform "+platform)

else:

print("Something went wrong :-(")

else:

print("Sorry, no matching station name found!")

Weather for a city

#!/usr/bin/env python3.7

import json

import requests

import datetime

import os

# You will have to sign up for a free API KEY for this one to work

# Go to https://openweathermap.org/

def get\_weather(api\_key, location):

url = "https://api.openweathermap.org/data/2.5/weather?q={}&APPID={}".format(location,   
 api\_key)

params = { }

headers = {

'Content-Type': 'application/json',

'Authorization': '{0}'.format(api\_key)

}

response = requests.get(url, headers=headers, params=params)

if response.status\_code == 200:

return json.loads(response.content.decode("utf-8"))

else:

print("\*\*\* ERROR! Response ",response.status\_code," \*\*\*")

return None

WEATHER\_API\_KEY = "----insert-your-api-key-here----"

city = input("What city do you want the weather for?")

data = get\_weather(WEATHER\_API\_KEY, city)

description = data["weather"][0]["description"]

kelvin = data["main"]["temp"]

celsius = round(kelvin-273.15)

message = "The weather in "+location+" is "+description+" and the temperature is "+str(celsius)

print(message)