Memoria Práctica 1
Comunicaciones

Leopoldo Cadavid Piñero

Febrero 2022

Índice

1. Introducción **1**

2. Ejemplo básicos **2**
2.1. Openweather 2
2.2. REData 6

3. Ampliaciones **7**
3.1. /info 8
3.2. /tiempo_hoy 8
3.3. /caracas 10
3.4. /presio 10
3.5. /cripto 11

4. Link al Github **12**

1. Introducción

En la siguiente memoria se procederá a explicar las pruebas y resultados obtenidos tras probar el uso de las APIS vistas en clase.

Primero veremos los ejemplos básicos dados en clase para el uso de estas APIS y programas para el manejo de estas:

- `Openweather`

- `REData`
- `Google Colab`
- `Node Red`

Posteriormente veremos en otro apartado la ampliación elegida en la práctica, en la cual se ha trabajado con la API de la app telegram para crear un bot que se conecte con Node red para obtener información de distintas APIS y proporcionarla al usuario con a través de distintos comandos.

2. Ejemplo básicos

2.1. Openweather

Lo primero que hemos hecho con Openweather, tras crear una cuenta y obtener nuestra API key, es crear un link con nuestro id y la ciudad de la cual queremos obtener la información.

```
http://api.openweathermap.org/data/2.5/weather?q=Alicante,
ES&mode=json&units=metric&APPID=339f553dbde09f5130892ce20de1dccf
```

Si introducimos el link en el navegador veremos el archivo JSON de esta manera:

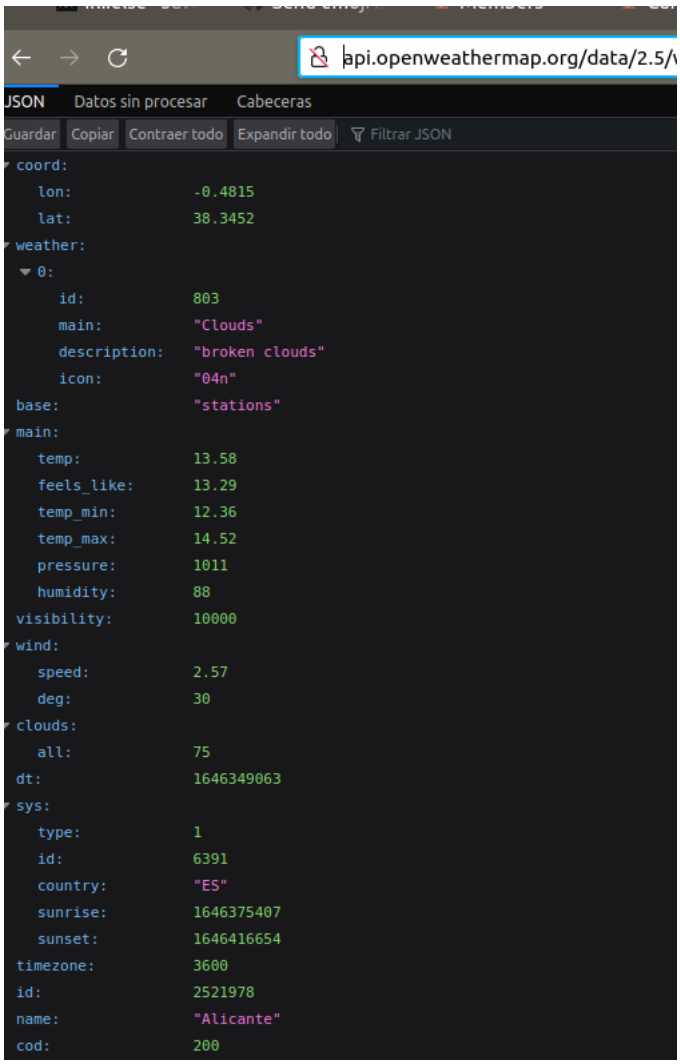

Figura 1: Archivo JSON de la API en el navegador

Una forma de tratar el archivo para tener la información de una forma más limpia sería utilizando un código de python para para tratar el JSON. Esto lo podemos hacer desde la herramienta Google Colab:

```python
##Leopoldo Cadavid
import datetime
import json
import urllib.request

def time_converter(time):
    converted_time = datetime.datetime.fromtimestamp(
        int(time)
    ).strftime('%I:%M %p')
    return converted_time

def url_builder1(city):
    user_api = '339f553dbde09f5130892ce20de1dccf'  # Obtain yours form: http://openweatherr
    unit = 'metric'  # For Fahrenheit use imperial, for Celsius use metric, and the defaul
    api = 'http://api.openweathermap.org/data/2.5/weather?q='     # Search for your city ID
    full_api_url = api + str(city) + '&mode=json&units=' + unit + '&APPID=' + user_api
    print(full_api_url)
    return full_api_url

def data_fetch(full_api_url):
    url = urllib.request.urlopen(full_api_url)
    output = url.read().decode('utf-8')
    raw_api_dict = json.loads(output)
    url.close()
    #print(raw_api_dict)
    return raw_api_dict

if __name__ == '__main__':
    try:
        now=datetime.datetime.now()
        datos=data_fetch(url_builder1('Caracas,ES'))
        print(datos)
        print(now)
        for key in datos['main']:
          print(key, ":", datos['main'][key])
        for key in datos['wind']:
          print(key, ":", datos['wind'][key])
        for key in datos['clouds']:
          print(key, ":", datos['clouds'][key])
        for key in datos['sys']:
          print(key, ":", datos['sys'][key])
        print(time_converter(1612681284))
        print(time_converter(datos['sys']['sunrise']))
        print(time_converter(datos['sys']['sunset']))
    except IOError:
        print('error')
```

```
http://api.openweathermap.org/data/2.5/weather?q=Alicante,ES&mode=json&units=metric&APPID=3
{'coord': {'lon': -0.4815, 'lat': 38.3452}, 'weather': [{'id': 801, 'main': 'Clouds', 'desc
2022-02-10 12:57:23.936467
temp : 14.46
feels_like : 13.95
temp_min : 12.62
temp_max : 17.13
pressure : 1028
humidity : 76
speed : 1.54
deg : 130
all : 20
type : 1
id : 6391
country : ES
sunrise : 1644476311
sunset : 1644514436
07:01 AM
06:58 AM
05:33 PM
```

Figura 2: Código python y salida desde Colab

La siguiente forma de tratar los datos de la api de Openweather es mediante la ampliación `Node Red`. Esta ampliación es muy usada en el ámbito de las conexiones entre dispositivos, páginas, etc, ya que mediante una interfaz sencilla e inttuitiva podemos conseguir conectar y tratar información de distintas apis y combinar estas.

En la siguiente imagen veremos el flow hecho para obtener el archivo JSON de Openweather mediante el nodo especifico de Openweather para Node Red.

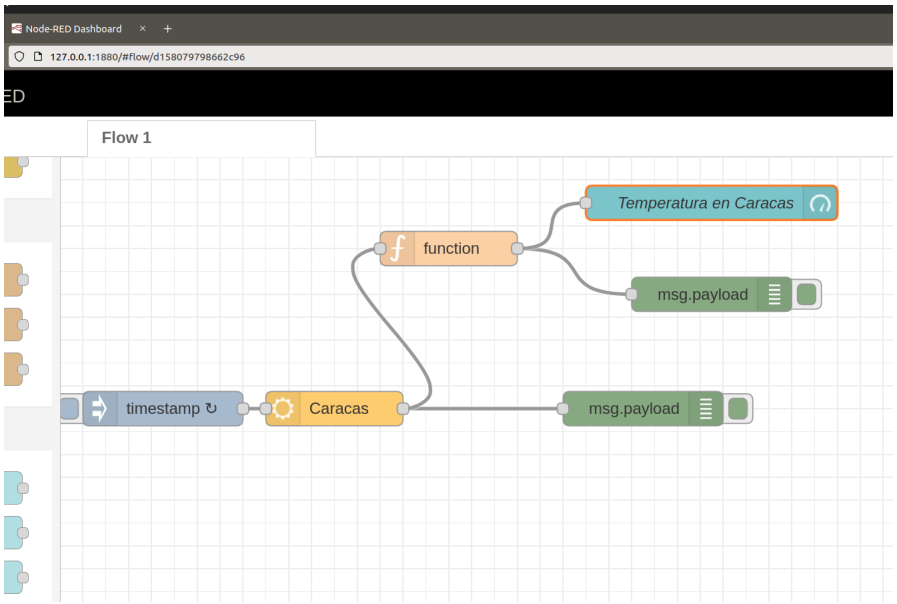

Figura 3: Flow de nodered para obtener la temperatura en Caracas

El siguiente código sería el que usamos para obtener el contenido de json y lo metemos pasamos al nodo Gauge:

```
variable = msg.payload
msg.payload = variable.tempc
return msg;
```

Y cuando accedemos a la UI de nodered observamos el resultado:

Figura 4: Gauge en formato donut

2.2. REData

La otra api que hemos usado en clase ha sido la del servicio REData, la cual proporciona información relacionada con la red electrica española. En ella podemos obtener información de diversos ámbitos (mercado, balances, importaciones...). En el siguiente ejemplo vemos una implementación en node-red, usando el bloque httpRequest para sacar los archivos JSON.

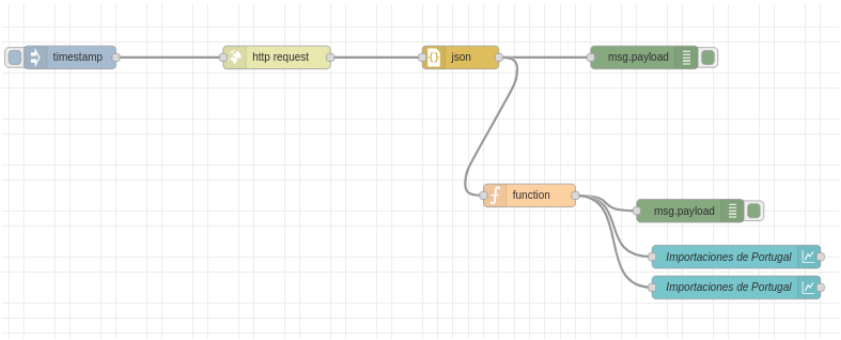

Figura 5: Flow de uso de REData

Y aquí podemos ver la salida en la interfaz de la dashboard.

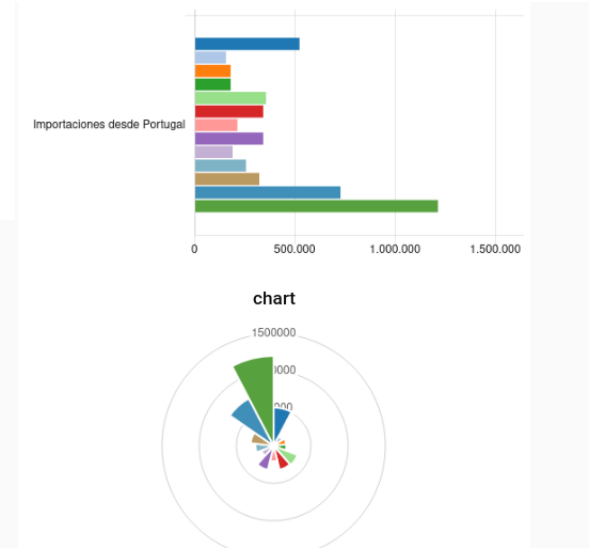

Figura 6: Charts de las importaciones de energía desde Portugal en 2019 por meses

3. Ampliaciones

La ampliación realizada para la práctica consiste en un bot de telegram que reune todas las APIS vistas y además hace uso de otras nuevas, como la de Telegram, manejandolo todo desde Node Red.

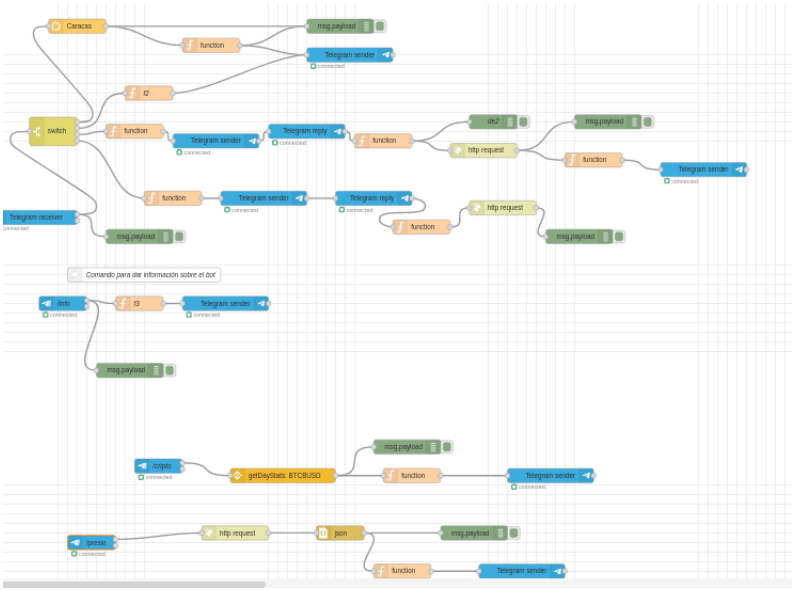

Figura 7: Flow de nodered del bot de telegram

3.1. /info

Este comando proporciona un mensaje con información sobre el bot. Hace uso de los bloques receiver y sender de la api de telegram en nodered.

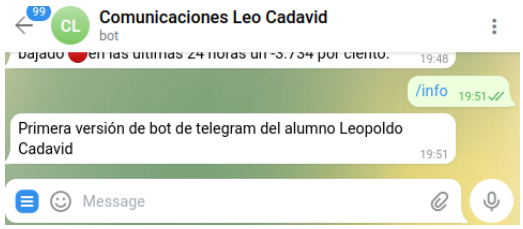

Figura 8: Ejemplo de la conversación con el comando /info

3.2. /tiempo_hoy

Comando que usa la api de Openweather para obtener el tiempo en la ciudad que se requiera. El código utilizado para procesar la información y

crear el mensaje es el siguiente:

```
variable = msg.payload
variable2 = variable.main.temp -273
variable2 = Math.round(variable2)
if(variable.weather[0].main == "Clouds"){
    cielo = "nublado "
}
if(variable.weather[0].main == "Clear"){
    cielo = "despejado "
}
if(variable.weather[0].main == "Rain"){
    cielo = "lluvioso "
}
if(variable2<=10){
    variable3= "";

}else{
    if(variable2<=20){
        variable3= "";
    }else{
        variable3= "";
    }

}
msg.payload={
    chatId: 1081269815,
    type: "message",
    content: "La temperatura en "+ variable.name + " es:  "
    + variable2 +  " grados centigrados " + variable3 + "\n" +
    "Podemos ver el cielo " + cielo

}
return msg;
```

Figura 9: Tiempo en la ciudad de Granada

3.3. /caracas

Este comando realiza el mismo proceso que el anterior 3.2, con la diferencia interna de que en vez de usar el nodo `httpRequest` usaremos el propio nodo de la api de Openweather:

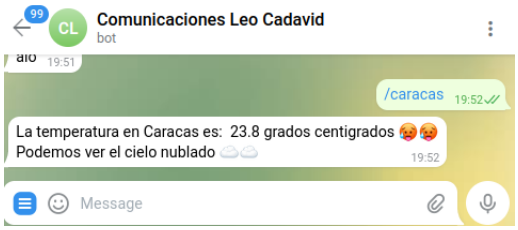

Figura 10: Tiempo en la ciudad de Caracas, Venezuela

3.4. /presio

En ejemplo hacemos uso de la api de REData para sacar el precio actual del megavatio/hora:

Figura 11: Precio a día de hoy del megavatio

3.5. /cripto

Por último, se ha creado un comando que, conectando con el nodo de la API de Binance, nos da el precio y la variación en las últimas 24 horas de este. Veamos el código para procesar el mensaje:

```
variable = msg.payload
if(variable.priceChangePercent <= 0){
    cambio = ""
    cambiado = "bajado "
}
else{
    cambio = ""
    cambiado = "subido "
}
msg.payload={
    chatId: 1081269815,
    type: "message",
    content: "El precio de bitcoin es: " + variable.lastPrice + " dólares,"
    + " el precio ha " + cambiado + cambio + "en las últimas 24 horas un " +
    variable.priceChangePercent + " por ciento."
}

return msg;
```

Y ahora podemos observar la conversación con el bot:

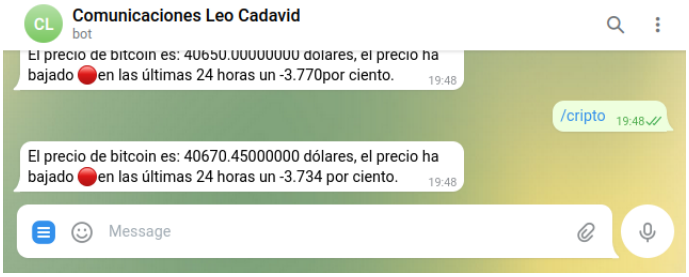

Figura 12: Precio de bitcoin y variación

4. Link al Github

Como se comentó en clase, se ha ido actualizando un repositorio de Github con las capturas JSON y códigos usados en la práctica para tener bien documentado todo en caso de que el profesor requiera consultarlo para la evaluacion de la práctica. el link al repositorio es el siguiente:

`https://github.com/leocadpin/Comunicaciones_Leopoldo_Cadavid`