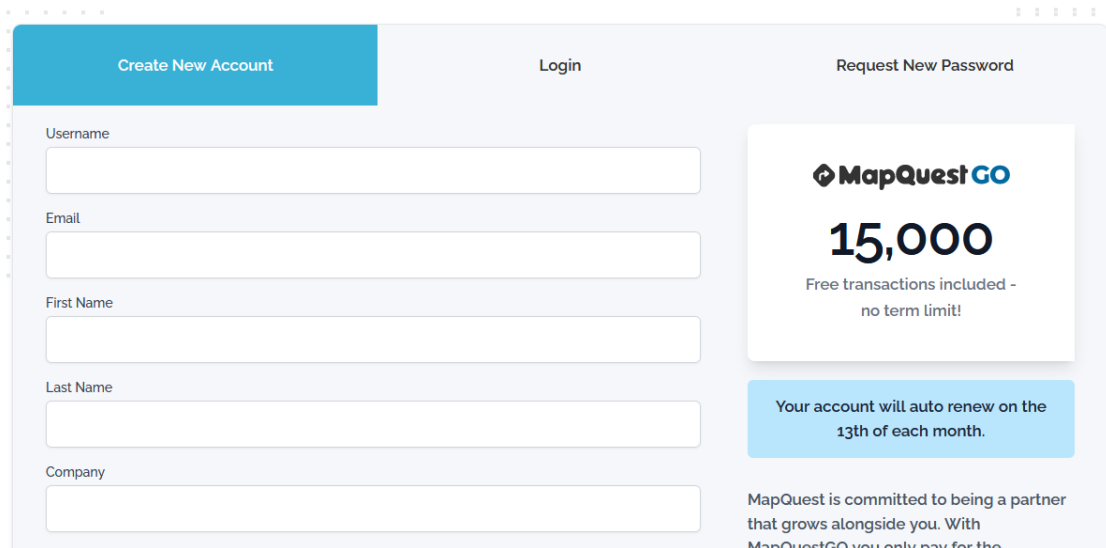


Paso 1 (opcional): Antes de poder construir nuestra aplicación, deberemos conseguir para nuestra API una llave. Pueden usar la llave proporcionada en este documento.

- Vamos a: <https://developer.mapquest.com/>
- Damos click en la opción **Login** en la página y luego **Create new Account**.



- Llenamos la forma y en la opción **Company** ingresamos: **Universidad ICESI**.

Paso 2: Abrimos VS Code o nuestro IDLE de python. Luego creamos una carpeta Repo para este curso y crearemos un archivo nuevo con nuestro nombre-API.py y luego comenzaremos a crear nuestro script.

Importar módulos para la aplicación

Debemos importar dos módulos de la librería de python los cuales son requests y urllib.parse. El módulo request nos ayuda a poder utilizar funciones para obtener data JSON desde una URL y la librería urllib.parse nos da una variedad de funciones que nos habilitan el analizar y manipular la data JSON que recibiremos desde una petición(request) a un URL.

```
1 import urllib.parse
2 import requests
```

Creacion de URL request a MapQuest API.

El primer paso en crear nuestra petición API es construir nuestra URL que nuestro script(aplicación) usará para hacer su API call inicial. La URL es una combinación de las siguientes variables:

- **main_api**: la URL principal que estamos accediendo
- **orig**: el parámetro que indica el punto de origen

- Combinaremos las cuatro variables que creamos(`main_api`, `orig`, `dest`, `key`) para hacer el formato de petición URL. Usaremos el método `urlencode` para hacer el formato correctamente de la dirección.

```
import urllib.parse
import requests

main_api = "https://www.mapquestapi.com/directions/v2/route?"
orig = "Cali, CO"
dest="Palmira, CO"

#USAR TU LLAVE OBTENIDA de el sitio developer de MapQuest
key = "S5QBdzNi6AtLvUfVdRTf5XRsrhn6LyYM"
url = main_api + urllib.parse.urlencode({"key":key, "from": orig, "to": dest})
json_data=requests.get(url).json()
print(json_data) #mostrar respuesta de peticion
```

Probando la petición URL

Guardaremos nuestro script con nuestro nombre nombre-API-respuesta.py y luego lo haremos correr para ver el resultado.

[illegible]

El resultado obtenido debe ser parecido al que veremos arriba o al que actualmente estamos efectuando en el laboratorio, Debemos recordar que esta respuesta es un diccionario de python con dos valores **key/value** .

Mostrar el URL y ver el Status code de nuestra petición JSON.

Ahora que ya sabemos que la respuesta a nuestra petición funciona y regresa la información deseada vamos a eliminar la función print que

contiene dentro la variable `json_data`, En su lugar personalizamos la data que recibimos de la siguiente manera.

```
10
11 json_data = requests.get(url).json()
12 print("URL: " + (url))
13 json_status = json_data["info"]["statuscode"]
14 |
```

Crearemos una variable llamada `json_status` la cual nos ayudará a filtrar el contenido dentro de `json_data`, Recordemos que ahora el contenido es un diccionario de python por lo tanto sigue las convenciones de este lenguaje

Añadiremos una condicional para poder mostrar el Status code de la API call en vigencia.

```
15 if json_status == 0:
16     print("API Status: " + str(json_status) + " = A successful route call.\n")
```

Procederemos a personalizar aún más nuestro script y habilitar user input para modificar el punto de partida y el punto de llegada.

Ahora instalar una extensión en Chrome/Firefox para poder visualizar de mejor manera nuestro formato JSON

Copiamos el URL que nos muestra de output cuando ejecutamos nuestro script. Pegamos este link en nuestro navegador y la extensión JSONview nos ayudará a poder visualizar la información más comprensible. Podemos dar click en el símbolo de menos para poder contraer la información y veremos que nos mostrará dos diccionarios principales(root dictionaries)

Mostrando información de duración y distancia.

Regresaremos a nuestro navegador y buscaremos el tiempo que nos tomará llegar de un punto a otro, La llave asignada a este valor se llama **formattedtime** la cual está dentro del diccionario **route**.

```

21  if json_status == 0:
22      print("API Status: " + str(json_status) + " = A successful route call.\n")
23      print("Trip Duration: " + (json_data["route"]["formattedTime"]))
24

```

Personalizamos aún más nuestra aplicación manipulando la información de nuestro diccionario de Python. Observar la estructura de la respuesta en nuestro navegador. Veremos qué Route es un diccionario y legs es uno de sus contenidos la llave legs contiene una lista incrustada y dentro de esta lista podemos ver un diccionario, Por sintaxis sabemos que las listas son objetos ordenados por un index en este caso legs contiene solamente un index el cual es 0, si desplegamos la información de este diccionario veremos que uno de sus key/value es maneuvers, veremos como el contenido de maneuvers tiene como valor una lista y dentro 7 diccionarios incrustados. Analizar la información.

Ahora que entendemos la estructura de la data que tenemos podemos hacer uso de un for loop para poder filtrar la información narrativa la cual nos dará direcciones de el como llegar a nuestro destino.

El resultado final se verá de la siguiente manera:

```

21  if json_status == 0:
22      print("API Status: " + str(json_status) + " = A successful route call.\n")
23      print("Trip Duration: " + (json_data["route"]["formattedTime"]))
24      print("Miles: " + str(json_data["route"]["distance"]))
25      print("Fuel Used (Gal): " + str(json_data["route"]["fuelUsed"]))
26      for each in json_data["route"]["legs"][0]["maneuvers"]:
27          print((each["narrative"] + " (" + str("{:.2f}".format((each["distance"]))) + " km")))
28

```

Investigar otros status code, y personalizar el código para mostrar una respuesta cuando el status code no sea 0.

Postman

Es una de las herramientas más populares para hacer pruebas eficaces a nuestra API con postman podremos hacer uso de los métodos GET, POST, PUT y PATCH en este laboratorio veremos los métodos GET y PUT.

<https://www.mapquest.com/directions/from/co-493843492/to/co-362957182>
<https://api.chucknorris.io/jokes/categories>

