### Trabajar con el diccionario de respuesta

Con la información de la llamada a la API representada como un diccionario, podemos trabajar con los datos allí almacenados. Vamos a generar alguna salida que resuma la información. Es una buena forma de asegurarnos de que hemos recibido la información que esperábamos y de empezar a examinar la información que nos interesa:

**python\_repos.py**

import requests  
  
# Make an API call and store the response.  
--snip--  
  
# Convert the response object to a dictionary.  
response\_dict = r.json()  
❶ print(f"Total repositories: {response\_dict['total\_count']}")  
print(f"Complete results: {not response\_dict['incomplete\_results']}")  
  
# Explore information about the repositories.  
❷ repo\_dicts = response\_dict['items']  
print(f"Repositories returned: {len(repo\_dicts)}")  
  
# Examine the first repository.  
❸ repo\_dict = repo\_dicts[0]  
❹ print(f"\nKeys: {len(repo\_dict)}")  
❺ for key in sorted(repo\_dict.keys()):  
 print(key)

Empezamos a explorar el diccionario de respuestas imprimiendo el valor asociado a 'total\_count', que representa el número total de repositorios Python devueltos por esta llamada a la API ❶. También utilizamos el valor asociado a 'incomplete\_results', para saber si GitHub pudo procesar completamente la consulta. En lugar de imprimir este valor directamente, imprimimos su opuesto: un valor de True indicará que hemos recibido un conjunto completo de resultados.

El valor asociado a 'items' es una lista que contiene una serie de diccionarios, cada uno de los cuales contiene datos sobre un repositorio Python individual. Asignamos esta lista de diccionarios a repo\_dicts ❷. A continuación, imprimimos la longitud de repo\_dicts para ver de cuántos repositorios tenemos información.

Para ver más de cerca la información devuelta sobre cada repositorio, sacamos el primer elemento de repo\_dicts y lo asignamos a repo\_dict ❸. A continuación, imprimimos el número de claves del diccionario para ver cuánta información tenemos ❹. Por último, imprimimos todas las claves del diccionario para ver qué tipo de información se incluye ❺.

Los resultados nos dan una idea más clara de los datos reales:

Status code: 200  
❶ Total repositories: 248  
❷ Complete results: True  
Repositories returned: 30  
  
❸ Keys: 78  
allow\_forking  
archive\_url  
archived  
--snip--  
url  
visiblity  
watchers  
watchers\_count

En el momento de escribir esto, sólo hay 248 repositorios de Python con más de 10.000 estrellas ❶. Podemos ver que GitHub fue capaz de procesar completamente la llamada a la API ❷. En esta respuesta, GitHub devolvió información sobre los primeros 30 repositorios que coinciden con las condiciones de nuestra consulta. Si queremos más repositorios, podemos solicitar páginas adicionales de datos.

La API de GitHub devuelve mucha información sobre cada repositorio: hay 78 claves en repo\_dict ❸. Cuando eches un vistazo a estas claves, te harás una idea del tipo de información que puedes extraer sobre un proyecto. (La única forma de saber qué información está disponible a través de una API es leer la documentación o examinar la información a través del código, como estamos haciendo aquí).

Vamos a extraer los valores de algunas de las claves de repo\_dict:

**python\_repos.py**

--snip--  
# Examine the first repository.  
repo\_dict = repo\_dicts[0]  
  
print("\nSelected information about first repository:")  
❶ print(f"Name: {repo\_dict['name']}")  
❷ print(f"Owner: {repo\_dict['owner']['login']}")  
❸ print(f"Stars: {repo\_dict['stargazers\_count']}")  
print(f"Repository: {repo\_dict['html\_url']}")  
❹ print(f"Created: {repo\_dict['created\_at']}")  
❺ print(f"Updated: {repo\_dict['updated\_at']}")  
print(f"Description: {repo\_dict['description']}")

Aquí imprimimos los valores de varias claves del diccionario del primer repositorio. Empezamos con el nombre del proyecto ❶. Un diccionario entero representa al propietario del proyecto, así que utilizamos la clave owner para acceder al diccionario que representa al propietario, y luego utilizamos la clave login para obtener el nombre de usuario del propietario ❷. A continuación, imprimimos cuántas estrellas ha obtenido el proyecto ❸ y la URL del repositorio GitHub del proyecto. A continuación, mostramos cuándo se creó ❹ y cuándo se actualizó por última vez ❺. Por último, imprimimos la descripción del repositorio.

El resultado debería ser algo parecido a esto

Status code: 200  
Total repositories: 248  
Complete results: True  
Repositories returned: 30  
  
Selected information about first repository:  
Name: public-apis  
Owner: public-apis  
Stars: 191493  
Repository: https://github.com/public-apis/public-apis  
Created: 2016-03-20T23:49:42Z  
Updated: 2022-05-12T06:37:11Z  
Description: A collective list of free APIs

Podemos ver que el proyecto Python con más estrellas en GitHub en el momento de escribir esto es *public-apis*. Su propietario es una organización con el mismo nombre, y ha recibido estrellas de casi 200.000 usuarios de GitHub. Podemos ver la URL del repositorio del proyecto, su fecha de creación en marzo de 2016, y que se ha actualizado recientemente. Además, la descripción nos dice que *public-apis* contiene una lista de API gratuitas que podrían interesar a los programadores.

[anterior](c17_6.html)[Subtema 7 de 20: (Ver todo)](c17.html)[siguiente](c17_8.html)