Spark has a rate limit for API calls. This limit is largely dependent on the total traffic, so is not specifically defined at this time. So, what should be done if you want to make API calls very frequently, but don’t want to hit errors by going over the rate limit? The answer is to use the Retry-After Header that is returned with an HTTP 429 error. The Retry-After header defines the number of seconds that should be waited before sending API requests again.

This app pulls the list of rooms available to the user (using a single user or bot’s bearer token), as fast as possible. Normally, it would be recommended to put some kind of wait time between continuous looping requests, to avoid hitting the rate limit. However, since that rate limit might be reached anyway depending on the time waited between requests, it’s important to handle for the possibility.

Basically, we make a call to <https://api.ciscospark.com/v1/rooms> continuously and look for an error. If the error is a status code 429, we check the wait time and sleep for the time specified, the start again. If there is an error, but it isn’t a 429, we break out of the loop because something else is wrong. Maybe we had a bad bearer token. Don’t forget to fill in yours in the variable in the code!

Since this code can run forever if it’s implemented properly. You’ll need to Ctrl+C to kill it!

import urllib2

import json

import time

def **sendSparkGET**(url):

request = urllib2.Request(url,

headers={*"Accept"* : *"application/json"*,

*"Content-Type"*:*"application/json"*})

request.add\_header(*"Authorization"*, *"Bearer "*+bearer)

response = urllib2.urlopen(request)

return response

bearer = *"BEARER\_TOKEN\_HERE"*

while True:

try:

result = sendSparkGET(*'https://api.ciscospark.com/v1/rooms'*)

print result.code, time.time(), result.headers[*'Trackingid'*]

except urllib2.HTTPError as e:

if e.code == 429:

print *'code'*, e.code

print *'headers'*, e.headers

print *'Sleeping for'*, e.headers[*'Retry-After'*], *'seconds'*

sleep\_time = int(e.headers[*'Retry-After'*])

while sleep\_time > 10:

time.sleep(10)

sleep\_time -= 10

print *'Asleep for'*, sleep\_time, *'more seconds'*

time.sleep(sleep\_time)

else:

print e, e.code

break

It’s important to note that it’s possible to send a request after receiving a 429 without getting an error. This is because there are many servers involved with handling a request. Because you’ve hit the rate limit of one, does not mean you’ve hit the rate limit of all. However, you’re probably pretty close, so it’s best to wait for the time described by Retry-After.

The full code can be found here:

<https://github.com/ciscospark/SparkRetryAfterDemo>

As always, If you have any questions, please contact [devsupport@ciscospark.com](mailto:devsupport@ciscospark.com) 24/7/365 - we’re happy to help!

Taylor Hanson, Customer Support Engineer II