This is a FAQ guideline on working with the AZURE Text Analytics API

Goal

You should be able to run the following command in a terminal.

```
curl -X POST
https://francecentral.api.cognitive.microsoft.com/text/analytics/v2.1/languages -H "Content-Type: application/json" -H "Ocp-Apim-Subscription-Key:1234567890abcdefghij" --data " { 'documents': [{ 'id': 1, 'text':'Good morning it is a beautiful day today.'}]}"
```

and get the following result”

```
{"documents":[{"id":"1","detectedLanguages":[{"name":"French","iso6391Name":"fr","score":1.0}]}],"errors":[]}
```

Dissecting the API Request

The API request can be decomposed along the following components:

- curl : the command line to send the request
- the API endpoint:
  "https://francecentral.api.cognitive.microsoft.com/text/analytics/v2.1/languages", which is the url the request is sent to.
  The endpoint is composed of
  - the domain: https://**francecentral**.api.cognitive.microsoft.com. The location part of the domain (here: **francecentral**) depends on the global azure environment settings and on the API version
  - the API version **/v2.1**. At the time of writing, the API is available in 2 versions v2.1 and v3.1. Not all locations are available for v3.1.
  - The path to the requested service: here **text/analytics**/v2.1/**languages**
- The header with the -H flag:
  - -H "Content-Type: application/json" : tells the API that the data sent is formatted as json
  - -H "Ocp-Apim-Subscription-Key:1234567890abcdefghij": the user key. Here the real key has been replaced with the value: 1234567890abcdefghij
- The data which is a json formatted string
  - { 'documents': [{ 'id': 1, 'text':'Bonjour je voudrais une baguette madame si il vous plait'}]}
  - it contains a list f documents, each one identified with a unique id.

    - to send 2 documents for instance the data would look like
    - { 'documents': [{ 'id': 1, 'text':'Bonjour je voudrais une baguette madame si il vous plait'}, { 'id': 2, 'text':'Hello, I would like a coffee with cream please'}]}

The structure of the request is simple : endpoint, header, user credentials and data formatted as json, but the details matter.

Setup - Happy Path

We want to query the language detection API of the Azure service.

Requirements:

- Go to portal.azure.com
- Search for cognitive services
- You should already have a subscription through the OC credentials. You can check your subscription here: https://portal.azure.com/#blade/Microsoft_Azure_Billing/SubscriptionsBlade
- You are on the market place page
- Search for text analytics
- Create the resource
- Fill out the form. This where you specify
    - The name of the resource (for instance: oc-azure-faq-20200212)
    - The location (francecentral)
    - The pricing Tier (choose the smallest one: F0 5k transactions per 30 days)
    - And the resources group (choose the one available in the drop down)
- Once the resource has been deployed click on go to resource
- You should see your endpoint and your API key:
- Copy them both to build your API command line requests

Location, key and endpoint are dependent

A key element is that endpoint, location and key are related and interdependent.

The service and location determines the endpoint and the key can only be used for that location, endpoint and that service. In other words, you cannot use the same key to query another service or location. If you get an error message, as a result of an API request, double check that these 3 elements are coherent.

Using the right API version: v2.1 vs v3.0

At the time of writing, the text analytics API exists in 2 versions: 2.1 and 3.1 (preview).
The stable version is 2.1 and this is the one you should be using.
The main difference is that some locations are not yet available for the 3.1 version. For instance francecentral is only available for v2.1.
This means that the following request will fail:

```
curl -X POST
https://francecentral.api.cognitive.microsoft.com/text/analytics/v3.1/languages -H "Content-Type: application/json" -H
"Ocp-Apim-Subscription-Key:1234567890abcdefghij" --data " {
'documents': [{ 'id': 1, 'text':'Good morning it is a
beautiful day today.'}]}"
```

Since the request is using the V3.0 to access the `francecentral` location.
A list of available locations is available on these pages:

- V2:
  https://westcentralus.dev.cognitive.microsoft.com/docs/services/TextAnalytics-v2-1/operations/56f30ceeeda5650db055a3c7
- V3:
  https://westcentralus.dev.cognitive.microsoft.com/docs/services/TextAnalytics-v3-0-Preview-1/operations/Languages

Location endpoint vs private endpoint

When you create the resource, you specify the name of the resource. Here we set the name of the resource to: oc-azure-faq-20200212
You can either use that name or the location of the resource in the endpoint.
So the same key will work for both requests:

```
curl -X POST
https://francecentral.api.cognitive.microsoft.com/text/analytics
/v2.1/languages -H "Content-Type: application/json" -H
"Ocp-Apim-Subscription-Key:8c7faf0490094a83b953bc4512a2eede"
--data " { 'documents': [{ 'id': 1, 'text':'Good morning it is
a beautiful day today.'}]}"
```

```
curl -X POST
https://oc-azure-faq-20200212.cognitiveservices.azure.com/text/an
alytics/v2.1/languages -H "Content-Type: application/json" -H
"Ocp-Apim-Subscription-Key:8c7faf0490094a83b953bc4512a2eede"
--data " { 'documents': [{ 'id': 1, 'text':'Good morning it is
a beautiful day today.'}]}"
```

Note the extra .api in `francecentral.api.cognitive.microsoft.com` when using the location based domain. Its not `francecentral.cognitive.microsoft.com`

Splitting shell commands across multiple lines

The command line is long and it would be pleasant to split it across several lines with \

So instead of:

```
curl -X POST
https://francecentral.api.cognitive.microsoft.com/text/analytic
s/v2.1/languages -H "Content-Type: application/json" -H
"Ocp-Apim-Subscription-Key:1234567890abcdefghij" --data " {
'documents': [{ 'id': 1, 'text':'Bonjour je voudrais une
baguette madame si il vous plait'}]}"
```

We would write

```
curl -X POST
https://francecentral.api.cognitive.microsoft.com/text/analytic
s/v2.1/languages \
-H "Content-Type: application/json" \
-H "Ocp-Apim-Subscription-Key:1234567890abcdefghij" \
--data " { 'documents': [{ 'id': 1, 'text':'Bonjour je voudrais
une baguette madame si il vous plait'}]}"
```

This should work fine. However, in some cases, notably with a zsh shell on mac osx this request was not understood by the API and failed.

The solution is to remove the line breaks and \ and write the command line on a single line.

Properly formatting the --data

The documentation specifies that the text should be sent as a document and each text should have an id. Basically the correct data structure to send N texts is:

```
{"documents": [
        {"id" : 1, "text": "some text"},

        ...

        {"id" : N, "text": "some other text"}

]}
```

This has been tested and works.

Some previous version of the documentation does not indicate the necessity of sending the text as an array of documents. This is a grey area and sometimes (sending on the API version and endpoint location), sending the data simply as (1 document):

```
{"text": "some text"}
```

will work.

But this is unstable and may result in errors at some point later. The best practice is to use the document structure with an **id** field.

Quotes

The examples above all follow the same pattern

- `curl  flag "key:value ", flag "key: value",`

Where flag is either --data, -H or -X POST.

In particular, the data part includes single quotes around the text and the keys of the document:

```
--data " { 'documents': [{ 'id': 1, 'text':'Good morning it is a beautiful day today.'}]}"
```

If the text also contains single quotes, (for instance: `Good morning it's a beautiful day today.`) the the single quote in the text must be escaped by adding a \ before the quote as such:

```
curl -X POST
https://oc-azure-faq-20200212.cognitiveservices.azure.com/text/analytics/v2.1/languages -H "Content-Type: application/json" -H "Ocp-Apim-Subscription-Key:8c7faf0490094a83b953bc4512a2eede"
```

```
--data " { 'documents': [{ 'id': 1, 'text':'Hello, it\'s a
beautiful day today.'}]}"
```

Or you can simply remove all quotes from the original text, this should not impair the performance of the language detection service.

Interpreting Error messages

Error messages such as the ones listed below are not to be taken literally but as a mere indication that something is not right with the request. They sometimes can be a bit misleading, indicating for instance that the problem lies with the API key when the culprit was the domain name.

```
{ "statusCode": 401, "message": "Unauthorized. Access token is
missing, invalid, audience is incorrect
(https://cognitiveservices.azure.com), or have expired." }
```

API Key

Securing your API key

It's important to understand that sharing the API key or making it public by mistake can be dangerous. Anyone with your API key can run queries on the Azure service and exploit the student's account with potential impact on billing and a risk of unlawful usage.

The API key should be regularly deleted and a new one recreated to avoid disclosure risks.

There are many resources on API key security, for instance here, here or here.

Format

The API key is 32 characters long.

On some documentation, examples of a valid query is written between brackets <>:

curl -X POST

```
https://francecentral.api.cognitive.microsoft.com/text/analytics/v2.1/languages
```

```
-H "Content-Type: application/json"

-H "Ocp-Apim-Subscription-Key:<resource API key>" --data " {
'documents': [{ 'id': 1, 'text':'Hello, it is a beautiful
day.'}]}"
```

The brackets are not part of the request and should be removed

The correct syntax is:

- "Ocp-Apim-Subscription-Key:1234567890abcdefghij"

and not

- "Ocp-Apim-Subscription-Key:<1234567890abcdefghij>"

Resources

Text analytics documentation

- https://docs.microsoft.com/en-us/azure/cognitive-services/text-analytics/how-tos/text-analytics-how-to-language-detection
- https://docs.microsoft.com/en-us/azure/cognitive-services/text-analytics/how-tos/text-analytics-how-to-call-api

You can try out the API and verify for instance your key and the endpoint on this page

- For V3: https://westeurope.dev.cognitive.microsoft.com/docs/services/TextAnalytics-v3-0-Preview-1/operations/Languages/console
- For V2: https://francecentral.dev.cognitive.microsoft.com/docs/services/TextAnalytics-v2-1/operations/56f30ceeeda5650db055a3c7/console