Using the Natural Language API from Google Docs

GSP126

Google Cloud Self-Paced Labs

Overview

What you'll learn

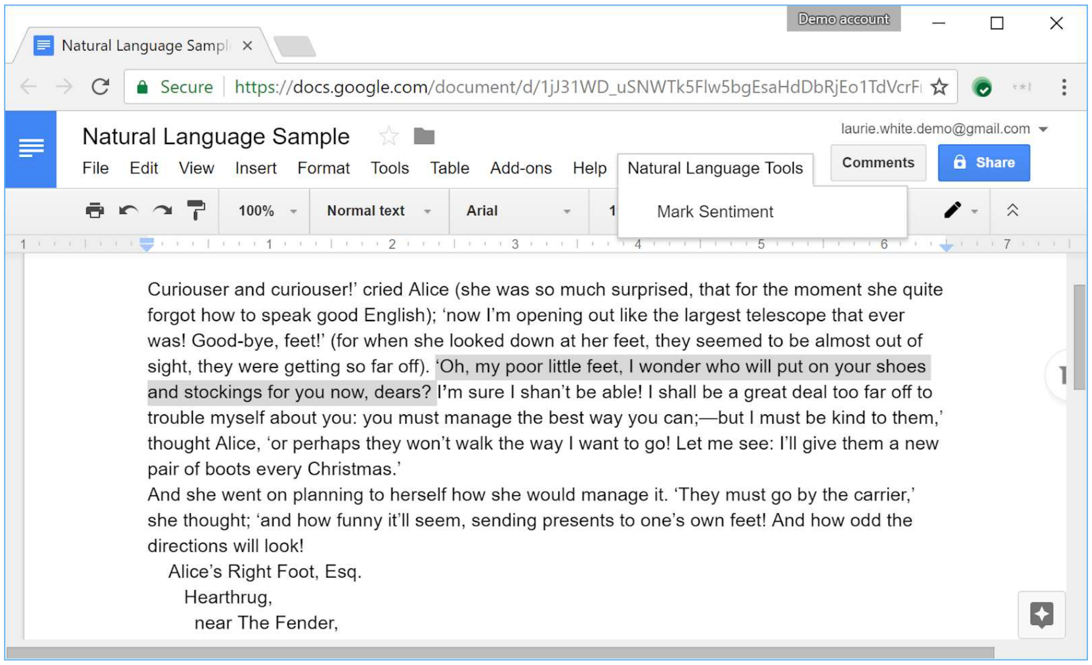
- How to call the Natural Language API from Google Docs
- How to add menus to Google Docs
- How to recognize and work with selected text in Google Docs

Introduction

The Natural Language API is a pretrained machine learning model that can analyze syntax, extract entities, and evaluate the sentiment of text. It can be called from Google Docs to perform all of these functions.
This lab will walk you through calling the Natural Language API to recognize the sentiment of selected text in a Google Doc and highlight it based on that sentiment.

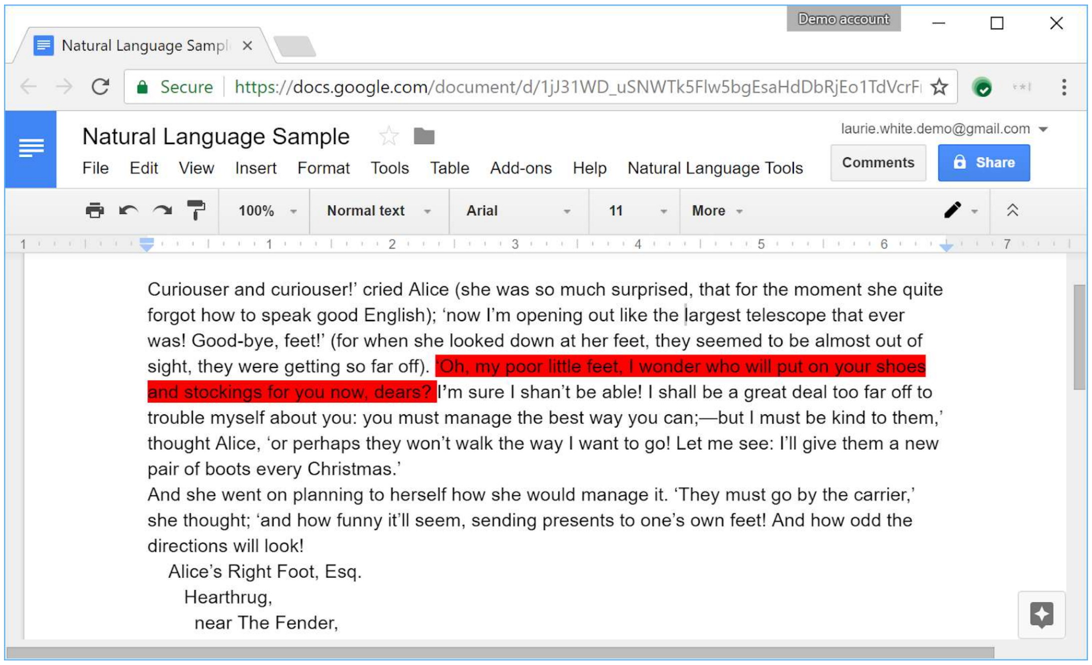
What are you building?

Once this lab is complete, you will be able to select text in a document and mark its sentiment, using a menu choice, as shown below.


Natural Language Sample
Demo account
Secure
https://docs.google.com/document/d/1jJ31WD_uSNWTk5Flw5bgEsaHdDbRjEo1TdVcrF
Natural Language Sample
laurie.white.demo@gmail.com
File Edit View Insert Format Tools Table Add-ons Help Natural Language Tools
Comments
Share
100%
Normal text
Arial
Mark Sentiment

Curiouser and curiouser!' cried Alice (she was so much surprised, that for the moment she quite forgot how to speak good English); 'now I'm opening out like the largest telescope that ever was! Good-bye, feet!' (for when she looked down at her feet, they seemed to be almost out of sight, they were getting so far off). 'Oh, my poor little feet, I wonder who will put on your shoes and stockings for you now, dears? I'm sure I shan't be able! I shall be a great deal too far off to trouble myself about you: you must manage the best way you can;—but I must be kind to them,' thought Alice, 'or perhaps they won't walk the way I want to go! Let me see: I'll give them a new pair of boots every Christmas.'
And she went on planning to herself how she would manage it. 'They must go by the carrier,' she thought; 'and how funny it'll seem, sending presents to one's own feet! And how odd the directions will look!
Alice's Right Foot, Esq.
Hearthrug,
near The Fender,

Text will be highlighted in red for negative sentiment, green for positive sentiment, and yellow for neutral sentiment.


Natural Language Sample
Demo account
Secure
https://docs.google.com/document/d/1jJ31WD_uSNWTk5Flw5bgEsaHdDbRjEo1TdVcrF
Natural Language Sample
laurie.white.demo@gmail.com
File Edit View Insert Format Tools Table Add-ons Help Natural Language Tools
Comments
Share
100%
Normal text
Arial
11
More

Curiouser and curiouser!' cried Alice (she was so much surprised, that for the moment she quite forgot how to speak good English); 'now I'm opening out like the largest telescope that ever was! Good-bye, feet!' (for when she looked down at her feet, they seemed to be almost out of sight, they were getting so far off). 'Oh, my poor little feet, I wonder who will put on your shoes and stockings for you now, dears? I'm sure I shan't be able! I shall be a great deal too far off to trouble myself about you: you must manage the best way you can;—but I must be kind to them,' thought Alice, 'or perhaps they won't walk the way I want to go! Let me see: I'll give them a new pair of boots every Christmas.'
And she went on planning to herself how she would manage it. 'They must go by the carrier,' she thought; 'and how funny it'll seem, sending presents to one's own feet! And how odd the directions will look!
Alice's Right Foot, Esq.
Hearthrug,
near The Fender,

This lab is focused on calling the Natural Language API from Google Docs.

Setup and Requirements

Qwiklabs setup

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

What you need

To complete this lab, you need:

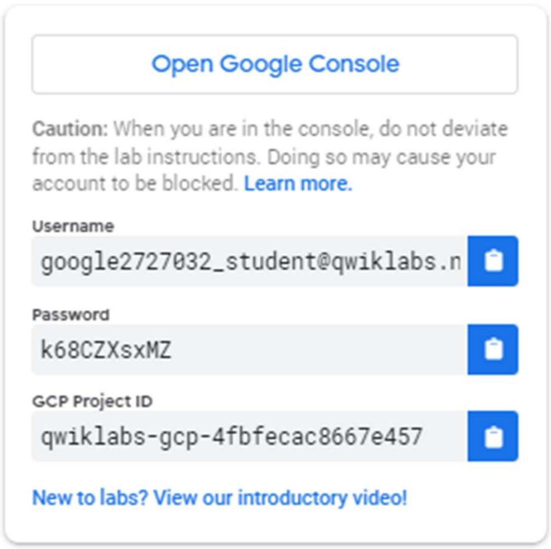
- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.

Note: If you already have your own personal Google Cloud account or project, do not use it for this lab.

Note: If you are using a Pixelbook, open an Incognito window to run this lab.

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.



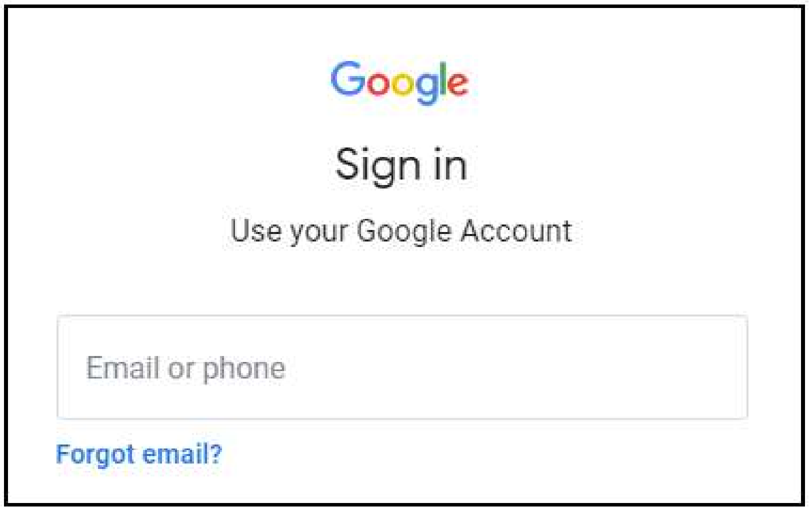
Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. Learn more.

Username
google2727032_student@qwiklabs.n

Password
k68CZXsxMZ

GCP Project ID
qwiklabs-gcp-4fbfecac8667e457

2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



Google

Sign in

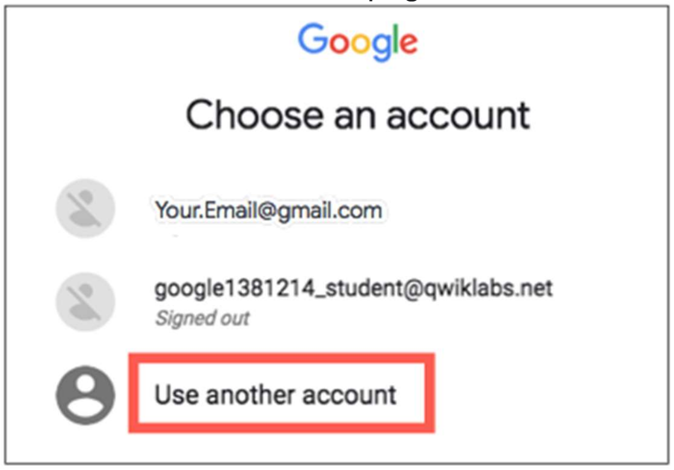
Use your Google Account

Email or phone

Forgot email?

Tip: Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another Account**.



Google

Choose an account

Your.Email@gmail.com

google1381214_student@qwiklabs.net
Signed out

Use another account

3. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

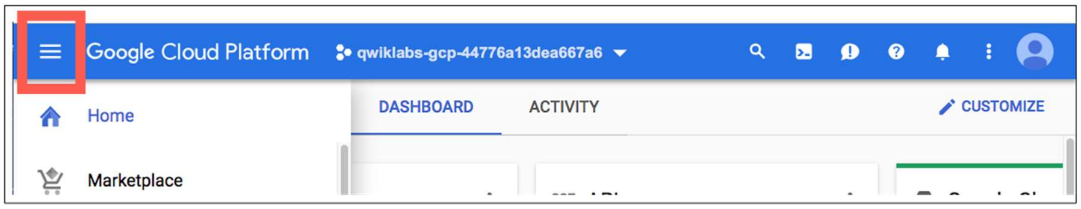
   Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

4. Click through the subsequent pages:

   - Accept the terms and conditions.
   - Do not add recovery options or two-factor authentication (because this is a temporary account).
   - Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

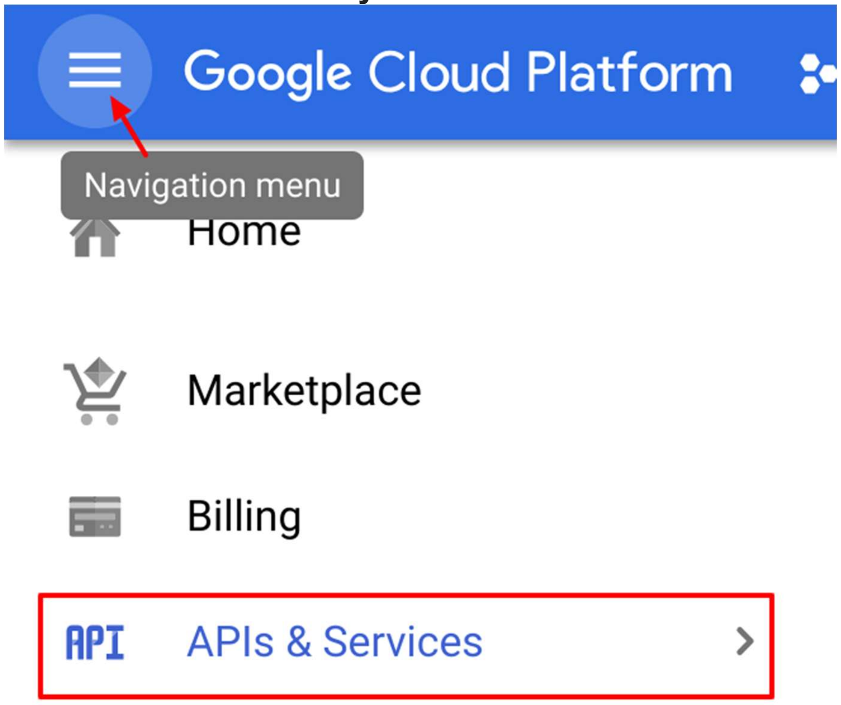
Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.



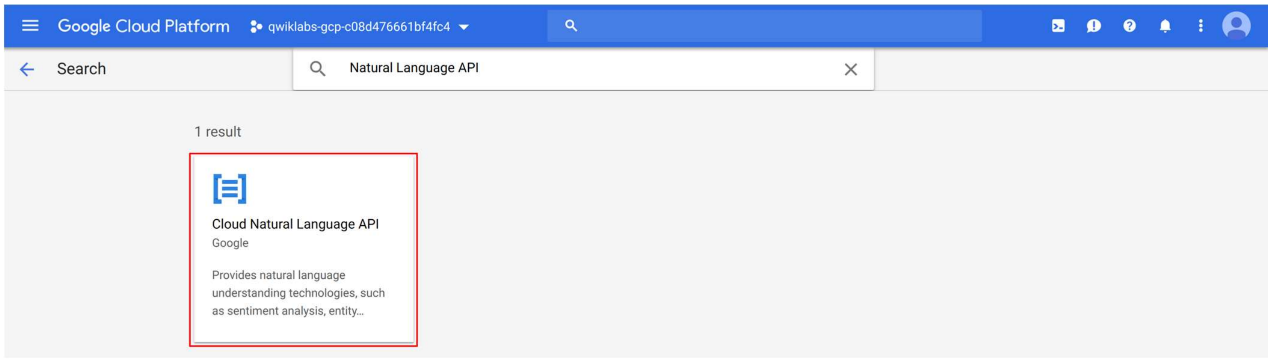
Enable the Natural Language API

Before you start, let's make sure that the Natural Language API is enabled.

1. In the Google Cloud console, select **Navigation menu** > **APIs & Services** > **Library**:



2. Search for **Natural Language API** and click on the API to enable or to confirm that the API is enabled.



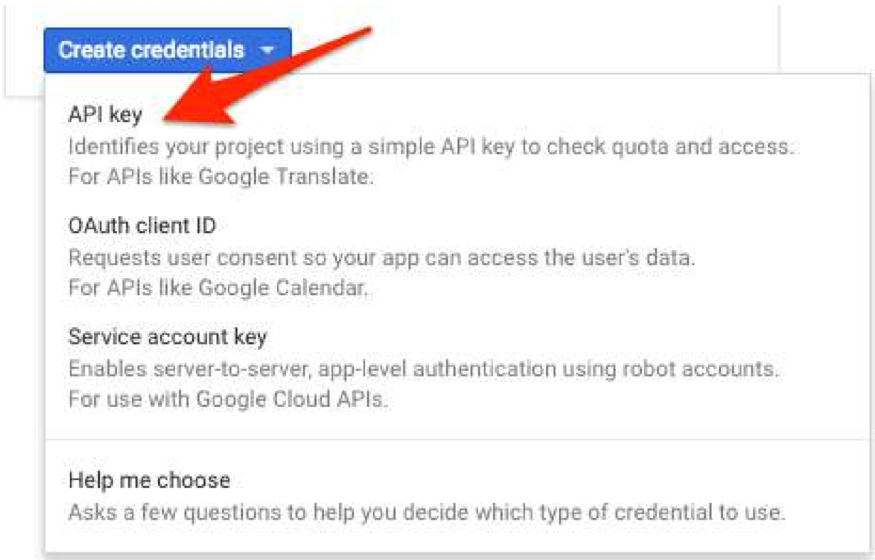
Cloud Natural Language API
Google
Provides natural language understanding technologies, such as sentiment analysis, entity...

Get an API Key

You'll need to generate an API user key to pass in the request URL.

1. To create an API key, seclect **Navigation menu** > **APIs & Services** > **Credentials**:

2. In the left pane, click on **Create credentials** and select **API key**:



Create credentials

API key
Identifies your project using a simple API key to check quota and access.
For APIs like Google Translate.

OAuth client ID
Requests user consent so your app can access the user's data.
For APIs like Google Calendar.

Service account key
Enables server-to-server, app-level authentication using robot accounts.
For use with Google Cloud APIs.

Help me choose
Asks a few questions to help you decide which type of credential to use.

3. Copy the API key to a text file or a Google Doc to use in a later step. Click **Close**. Once you have the API key you are ready to move into Google Docs.

Set up your Google Doc

Before you can call the Natural Language API, you'll make an Apps Script program to create the menu, link it to a function to mark the text, and extract the text from the user selection.

1. Create a new Google Doc.
2. From within your new document, select the menu item **Tools > Script editor**. If you are presented with a welcome screen, click **Blank Project**.

3. Delete any code in the script editor and paste in the code below. This code will create a menu item, extract the text from the current selected text, and highlight the text based on its sentiment. It does not call the Natural Language API yet.

```
/**
* @OnlyCurrentDoc
*
* The above comment directs Apps Script to limit the scope of file
* access for this add-on. It specifies that this add-on will only
* attempt to read or modify the files in which the add-on is used,
* and not all of the user's files. The authorization request message
* presented to users will reflect this limited scope.
*/

/**
* Creates a menu entry in the Google Docs UI when the document is
* opened.
*
*/
function onOpen() {
  var ui = DocumentApp.getUi();

  ui.createMenu('Natural Language Tools')
    .addItem('Mark Sentiment', 'markSentiment')
    .addToUi();
}
/**
* Gets the user-selected text and highlights it based on sentiment
* with green for positive sentiment, red for negative, and yellow
* for neutral.
*
*/
function markSentiment() {
  var POSITIVE_COLOR = '#00ff00';  //  Colors for sentiments
  var NEGATIVE_COLOR = '#ff0000';
  var NEUTRAL_COLOR = '#ffff00';
  var NEGATIVE_CUTOFF = -0.2;   //  Thresholds for sentiments
  var POSITIVE_CUTOFF = 0.2;

  var selection = DocumentApp.getActiveDocument().getSelection();
  if (selection) {
    var string = getSelectedText();

    var sentiment = retrieveSentiment(string);

    //  Select the appropriate color
    var color = NEUTRAL_COLOR;
    if (sentiment <= NEGATIVE_CUTOFF) {
      color = NEGATIVE_COLOR;
    }
    if (sentiment >= POSITIVE_CUTOFF) {
      color = POSITIVE_COLOR;
    }

    //  Highlight the text
    var elements = selection.getSelectedElements();
    for (var i = 0; i < elements.length; i++) {
      if (elements[i].isPartial()) {
        var element = elements[i].getElement().editAsText();
        var startIndex = elements[i].getStartOffset();
        var endIndex = elements[i].getEndOffsetInclusive();
        element.setBackgroundColor(startIndex, endIndex, color);

      } else {
        var element = elements[i].getElement().editAsText();
        foundText = elements[i].getElement().editAsText();
        foundText.setBackgroundColor(color);
      }
    }
  }
}
/**
 * Returns a string with the contents of the selected text.
 * If no text is selected, returns an empty string.
 */
function getSelectedText() {
  var selection = DocumentApp.getActiveDocument().getSelection();
  var string = "";
  if (selection) {
    var elements = selection.getSelectedElements();

    for (var i = 0; i < elements.length; i++) {
      if (elements[i].isPartial()) {
        var element = elements[i].getElement().asText();
        var startIndex = elements[i].getStartOffset();
        var endIndex = elements[i].getEndOffsetInclusive() + 1;
        var text = element.getText().substring(startIndex, endIndex);
        string = string + text;

      } else {
        var element = elements[i].getElement();
        // Only translate elements that can be edited as text; skip
        // images and other non-text elements.
        if (element.editAsText) {
          string = string + element.asText().getText();
        }
      }
    }
  }
  return string;
}

/** Given a string, will call the Natural Language API and retrieve
  * the sentiment of the string.  The sentiment will be a real
  * number in the range -1 to 1, where -1 is highly negative
  * sentiment and 1 is highly positive.
*/
function retrieveSentiment (line) {
//  TODO:  Call the Natural Language API with the line given
//         and return the sentiment value.
  return 0.0;
}
```

More details about Apps Script are available at the App Script site.

4. Select the menu item **File** > **Save all**. Name your new script **Natural Language Tools** and click **OK**. (The script's name is shown to end users in several places, including the authorization dialog.)

5. Return to your document. Add text to your document. You can use the sample that comes from Alice in Wonderland on Project Gutenberg (copy and paste the `Plain Text UTF-8` version into the document), but feel free to use any text you wish.
6. Reload the document to see the new menu, **Natural Language Tools**, which you created, appear in the Google Docs toolbar.

7. Select text and then the **Mark Sentiment** option from the Natural Language Tools menu. The first time you select this option, you will be prompted to authorize the script to run. Click **Authorize**, and then log in with your credentials.

8. Allow Natural Language Tools to view and manage documents that this application has been installed in.

9. Once the script is authorized, the selected text will be highlighted in yellow, since the stub for sentiment analysis always returns 0.0, which is neutral.

Call the Natural Language API

Once your program can extract text from the selection and highlight it, it's time to call the Natural Language API. All of this will be done in the body of the `retrieveSentiment` function.

Find all the details of the Natural Language API here.

1. Return to the **Tools** > **Script editor** in Google Docs.

2. In the `retrieveSentiment` function, remove the current lines and add a variable to contain your API key, which you saved in the "Get an API key" section.

```
  var apiKey = "your key here";
```

3. Create a variable that will hold the URL of the Natural Language API with your API key appended to it.

```
var apiEndpoint =
'https://language.googleapis.com/v1/documents:analyzeSentiment?key='
+ apiKey;
```

4. Build a structure from the line passed into the function that holds the text of the line, along with its type and language. Currently the only supported language is English.

```
  var docDetails = {
    language: 'en-us',
    type: 'PLAIN_TEXT',
    content: line
  };
```

5. Build the entire data payload from the document details by adding the encoding type.

```
  var nlData = {
    document: docDetails,
    encodingType: 'UTF8'
  };
```

6. Create a structure containing the payload and the necessary header information.

```
  var nlOptions = {
    method : 'post',
    contentType: 'application/json',
    payload : JSON.stringify(nlData)
  };
```

7. Make the call, saving the response.

```
  var response = UrlFetchApp.fetch(apiEndpoint, nlOptions);
```

8. The response will be returned in JSON format, so parse it and extract the score field, if it exists. Return either that field or 0.0.

```
  var data = JSON.parse(response);
  var sentiment = 0.0;
  //  Ensure all pieces were in the returned value
  if (data && data.documentSentiment
          && data.documentSentiment.score){
     sentiment = data.documentSentiment.score;
  }

  return sentiment;
```

The complete code to retrieve the sentiment is below.

```
function retrieveSentiment (line) {
  var apiKey = "your key here";
  var apiEndpoint =
'https://language.googleapis.com/v1/documents:analyzeSentiment?key='
+ apiKey;

  //  Create a structure with the text, its language, its type,
  //  and its encoding
  var docDetails = {
    language: 'en-us',
    type: 'PLAIN_TEXT',
    content: line
  };

  var nlData = {
    document: docDetails,
    encodingType: 'UTF8'
  };

  //  Package all of the options and the data together for the call
  var nlOptions = {
    method : 'post',
    contentType: 'application/json',
    payload : JSON.stringify(nlData)
  };

  //  And make the call
  var response = UrlFetchApp.fetch(apiEndpoint, nlOptions);

  var data = JSON.parse(response);

  var sentiment = 0.0;
  //  Ensure all pieces were in the returned value
  if (data && data.documentSentiment
          && data.documentSentiment.score){
     sentiment = data.documentSentiment.score;
  }

  return sentiment;
}
```

9. Save your script, reload the document, and try out the full program. You may need to re-authorize with your credentials to enable the new functionality. Select different sections of your document to see how the sentiment may differ over its parts.

Congratulations!

You've created a Google Doc and called the Natural Language API to analyze the sentiment of selected portions of the document.

Finish Your Quest

This self-paced lab is part of the Workspace Integrations Quest. A Quest is a series of related labs that form a learning path. Completing this Quest earns you the badge above, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in this Quest and get immediate completion credit if you've taken this lab. See other available Qwiklabs Quests.

Take your next lab

Continue your Quest with Build a Complete Database Web App with App Maker, or check out these suggestions:

- The Apps Script CLI - clasp

Next Steps / Learn More

Continue your Google Cloud learning with these suggestions:

- Take more labs. Learn more about the Natural Language API by taking labs, like Entity and Sentiment analysis with the Natural Language API. Or take something completely different like Rent-a-VM to Process Earthquake Data.
- Start a Quest. A Qwiklabs Quest is a series of related labs that form a learning path. Completing a Quest earns you a digital badge, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. See available Qwiklabs Quests.

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. Our classes include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. Certifications help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated February 16, 2021

Lab Last Tested February 21, 2019