OAuth with Python (Windows)

* Objective:
  + Needed to have a test case generator to generate the test cases along with test plans with ease. To do this Google Spreadsheet is the good choice to generate test input data and Python is the good choice to retrieve the Google Spreadsheet data and managing with the strings to create the test case and test plan files.
    - So if i can just generate the testing input data, testing function parameter values for example, into Google Spreadsheet → Execute Python test plan and test case generator → Test plan & Test cases generated.
    - Now i just need to create a shell script or also Python script to execute all the test cases indicated maybe inside a log file generated with single execution.
* Staff:
  + Lead Programmer: Samil Chai
  + Junior Programmer: Nick Jang
    - Email: [nickjang114@gmail.com](mailto:nickjang114@gmail.com)
* Start date: 2017/10/02
* End date: 2017/10/02

# Main tutorial:

* 1. <https://developers.google.com/sheets/api/quickstart/python>

# Must install:

* 1. pip 9.0.1 from c:\users\nickj\appdata\local\programs\python\python36\lib\site-packages (python 3.6)

# Google Developer Console setup:

* 1. Use [this wizard](https://console.developers.google.com/start/api?id=sheets.googleapis.com) to create or select a project in the Google Developers Console and automatically turn on the API. Click **Continue**, then **Go to credentials**.
     1. [My Google Developer Console](https://console.developers.google.com/apis/credentials?highlightClient=711100213978-r9sgti5vht063nqavui422bpd7ght635.apps.googleusercontent.com&project=driven-density-181823) (Can just click [this wizard](https://console.developers.google.com/start/api?id=sheets.googleapis.com) and sign in with Google account)
  2. On the **Add credentials to your project** page, click the **Cancel** button.
  3. At the top of the page, select the **OAuth consent screen** tab. Select an **Email address**, enter a **Product name** if not already set, and click the **Save** button.
     1. **For my case, i named it “WY\_PROJ\_BLENDER\_EDITOR” since i am using this to generate test cases for project name “WY\_PROJ\_BLENDER\_EDITOR”**
  4. Select the **Credentials** tab, click the **Create credentials** button and select **OAuth client ID**.
  5. Select the application type **Other**, enter the name "Google Sheets API Quickstart", and click the **Create** button.
     1. **For my case, i named it “WY\_PROJ\_BLENDER\_EDITOR\_OAUTH\_CLIENT”**
  6. Click **OK** to dismiss the resulting dialog.
  7. Click the file\_download (Download JSON) button to the right of the client ID.
  8. Move this file to your working directory and rename it client\_secret.json.

# Install Google API Python client:

* 1. $ pip install --upgrade google-api-python-client

# Running sample code:

* 1. Create a test file “oauthtest.py” and paste the following code:

from \_\_future\_\_ import print\_function

import httplib2

import os

from apiclient import discovery

from oauth2client import client

from oauth2client import tools

from oauth2client.file import Storage

try:

import argparse

flags = argparse.ArgumentParser(parents=[tools.argparser]).parse\_args()

except ImportError:

flags = None

# If modifying these scopes, delete your previously saved credentials

# at ~/.credentials/sheets.googleapis.com-python-quickstart.json

SCOPES = 'https://www.googleapis.com/auth/spreadsheets.readonly'

CLIENT\_SECRET\_FILE = 'client\_secret.json'

APPLICATION\_NAME = 'Google Sheets API Python Quickstart'

def get\_credentials():

"""Gets valid user credentials from storage.

If nothing has been stored, or if the stored credentials are invalid,

the OAuth2 flow is completed to obtain the new credentials.

Returns:

Credentials, the obtained credential.

"""

home\_dir = os.path.expanduser('~')

credential\_dir = os.path.join(home\_dir, '.credentials')

if not os.path.exists(credential\_dir):

os.makedirs(credential\_dir)

credential\_path = os.path.join(credential\_dir,

'sheets.googleapis.com-python-quickstart.json')

store = Storage(credential\_path)

credentials = store.get()

if not credentials or credentials.invalid:

flow = client.flow\_from\_clientsecrets(CLIENT\_SECRET\_FILE, SCOPES)

flow.user\_agent = APPLICATION\_NAME

if flags:

credentials = tools.run\_flow(flow, store, flags)

else: # Needed only for compatibility with Python 2.6

credentials = tools.run(flow, store)

print('Storing credentials to ' + credential\_path)

return credentials

def main():

"""Shows basic usage of the Sheets API.

Creates a Sheets API service object and prints the names and majors of

students in a sample spreadsheet:

https://docs.google.com/spreadsheets/d/1BxiMVs0XRA5nFMdKvBdBZjgmUUqptlbs74OgvE2upms/edit

"""

credentials = get\_credentials()

http = credentials.authorize(httplib2.Http())

discoveryUrl = ('https://sheets.googleapis.com/$discovery/rest?'

'version=v4')

service = discovery.build('sheets', 'v4', http=http,

discoveryServiceUrl=discoveryUrl)

spreadsheetId = '1BxiMVs0XRA5nFMdKvBdBZjgmUUqptlbs74OgvE2upms'

rangeName = 'Class Data!A2:E'

result = service.spreadsheets().values().get(

spreadsheetId=spreadsheetId, range=rangeName).execute()

values = result.get('values', [])

if not values:

print('No data found.')

else:

print('Name, Major:')

for row in values:

# Print columns A and E, which correspond to indices 0 and 4.

print('%s, %s' % (row[0], row[4]))

if \_\_name\_\_ == '\_\_main\_\_':

main()

* 1. Run the test file with following command:
     1. $ python oauthtest.py
  2. This should work perfectly and output:

Name, Major:

Alexandra, English

Andrew, Math

Anna, English

Becky, Art

Benjamin, English

Carl, Art

Carrie, English

Dorothy, Math

Dylan, Math

Edward, English

Ellen, Physics

Fiona, Art

John, Physics

Jonathan, Math

Joseph, English

Josephine, Math

Karen, English

Kevin, Physics

Lisa, Art

Mary, Physics

Maureen, Physics

Nick, Art

Olivia, Physics

Pamela, Math

Patrick, Art

Robert, English

Sean, Physics

Stacy, Math

Thomas, Art

Will, Math

# Running with my Google Spreadsheet:

* 1. Make sure the “client\_secret.json” is with the main Python file.
  2. I need to change few lines of code in order to load and read from my own Google Spreadsheet:
     1. My Google Spreadsheet URL:
        1. This is one of my time report spreadsheet name “WY\_PROJ\_BLENDER\_EDITOR\_04\_20171001\_20171003\_TIME\_REPORT” which has a feed name “**Sheet1**”
           + **Caution: Need to use the feed name in the code. The feed name is shown at the left bottom of the spreadsheet window.**
        2. <https://docs.google.com/spreadsheets/d/1WReZYyjIMcfau-9TfERLZtquK7Tx1kRhHKCTfv1u184/edit#gid=0>
     2. Adjust lines:
        1. CLIENT\_SECRET\_FILE = 'client\_secret.json'
           + Google Developer Console downloaded client secret file path
        2. APPLICATION\_NAME = 'WY\_PROJ\_BLENDER\_EDITOR\_OAUTH\_CLIENT'
           + Google Developer Console OAuth 2.0 created client credential.
        3. spreadsheetId = '1WReZYyjIMcfau-9TfERLZtquK7Tx1kRhHKCTfv1u184'
           + Spreadsheet ID indicated in my Google Spreadsheet URL
        4. rangeName = 'Sheet1!A2:E'
           + Provide the feed name i want to load with.

Eg. Sheet1

* + - * + Provide the section i want to load from the feed with.

Eg. All cells from row A2 to column E.

# Troubleshooting:

* 1. When i first compile and run with above steps reading from my Google Spreadsheet, i got this error:

Traceback (most recent call last):

File "oauthtest.py", line 94, in <module>

main()

File "oauthtest.py", line 81, in main

if values[i][1] != []:

IndexError: list index out of range

* + 1. This was because i was accessing the element where it does not exist, where there are only 1 element in the cell row received from Google Spreadsheet (eg. [“elem1”]) but the code from the tutorial is using the fixed index value to print the output (eg. values[0], values[1]).
    2. So i adjusted the code to print the column elements for those who exist in row element range (where “values” is the row element):

if not values:

print('No data found.')

else:

print(len(values))

if values != None:

for i in range(0, len(values)):

for j in range(0, len(values[i])):

print('%s,' % (values[i][j]), end='')

print("\n")

else:

print("\n")

# Final working code:

from \_\_future\_\_ import print\_function

import httplib2

import os

from apiclient import discovery

from oauth2client import client

from oauth2client import tools

from oauth2client.file import Storage

try:

import argparse

flags = argparse.ArgumentParser(parents=[tools.argparser]).parse\_args()

except ImportError:

flags = None

# If modifying these scopes, delete your previously saved credentials

# at ~/.credentials/sheets.googleapis.com-python-quickstart.json

SCOPES = 'https://www.googleapis.com/auth/spreadsheets.readonly'

CLIENT\_SECRET\_FILE = 'client\_secret.json'

APPLICATION\_NAME = 'WY\_PROJ\_BLENDER\_EDITOR\_OAUTH\_CLIENT'

def get\_credentials():

"""Gets valid user credentials from storage.

If nothing has been stored, or if the stored credentials are invalid,

the OAuth2 flow is completed to obtain the new credentials.

Returns:

Credentials, the obtained credential.

"""

home\_dir = os.path.expanduser('~')

credential\_dir = os.path.join(home\_dir, '.credentials')

if not os.path.exists(credential\_dir):

os.makedirs(credential\_dir)

credential\_path = os.path.join(credential\_dir,

'sheets.googleapis.com-python-client\_secret.json')

store = Storage(credential\_path)

credentials = store.get()

if not credentials or credentials.invalid:

flow = client.flow\_from\_clientsecrets(CLIENT\_SECRET\_FILE, SCOPES)

flow.user\_agent = APPLICATION\_NAME

if flags:

credentials = tools.run\_flow(flow, store, flags)

else: # Needed only for compatibility with Python 2.6

credentials = tools.run(flow, store)

print('Storing credentials to ' + credential\_path)

return credentials

def main():

"""Shows basic usage of the Sheets API.

Creates a Sheets API service object and prints the names and majors of

students in a sample spreadsheet:

https://docs.google.com/spreadsheets/d/1BxiMVs0XRA5nFMdKvBdBZjgmUUqptlbs74OgvE2upms/edit

https://docs.google.com/spreadsheets/d/1WReZYyjIMcfau-9TfERLZtquK7Tx1kRhHKCTfv1u184/edit#gid=0

"""

credentials = get\_credentials()

http = credentials.authorize(httplib2.Http())

discoveryUrl = ('https://sheets.googleapis.com/$discovery/rest?'

'version=v4')

service = discovery.build('sheets', 'v4', http=http,

discoveryServiceUrl=discoveryUrl)

spreadsheetId = '1WReZYyjIMcfau-9TfERLZtquK7Tx1kRhHKCTfv1u184'

rangeName = 'Sheet1!A:E'

result = service.spreadsheets().values().get(spreadsheetId=spreadsheetId, range=rangeName).execute()

values = result.get('values', [])

if not values:

print('No data found.')

else:

print(len(values))

if values != None:

for i in range(0, len(values)):

for j in range(0, len(values[i])):

print('%s,' % (values[i][j]), end='')

print("\n")

else:

print("\n")

if \_\_name\_\_ == '\_\_main\_\_':

main()