Accessing-Google-calendar-using-Python In this project, we use the Google calendar API to integrate google calendar with Python.

The Calendar.py file, prints out the upcoming 10 events from your calendar to the console.

The Calendar2.py file, adds the event that you want (with the details that you specify) to your google calendar.

The other required infos have been given inside the python file.

Do not forget to turn on the "Allow less secure apps" option in your google account settings.

In [ ]:

```
pip install google-api-python-client                #this library will help us to interact withany kind of google api
```

```
Requirement already satisfied: google-api-python-client in /usr/local/lib/python3.6/dist-packages (1.7.12)
Requirement already satisfied: six<2dev,>=1.6.1 in /usr/local/lib/python3.6/dist-packages (from google-api-python-client) (1.15.0)
Requirement already satisfied: httplib2<1dev,>=0.17.0 in /usr/local/lib/python3.6/dist-packages (from google-api-python-client) (0.17.4)
Requirement already satisfied: google-auth>=1.4.1 in /usr/local/lib/python3.6/dist-packages (from google-api-python-client) (1.17.2)
Requirement already satisfied: uritemplate<4dev,>=3.0.0 in /usr/local/lib/python3.6/dist-packages (from google-api-python-client) (3.0.1)
Requirement already satisfied: google-auth-httplib2>=0.0.3 in /usr/local/lib/python3.6/dist-packages (from google-api-python-client) (0.0.4)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.6/dist-packages (from google-auth>=1.4.1->google-api-python-client)
(4.1.1)
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.6/dist-packages (from google-auth>=1.4.1->google-api-python-client)
(0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4; python_version >= "3" in /usr/local/lib/python3.6/dist-packages (from google-auth>=1.4.1->google-api-
python-client) (4.6)
Requirement already satisfied: setuptools>=40.3.0 in /usr/local/lib/python3.6/dist-packages (from google-auth>=1.4.1->google-api-python-client) (49.
1.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.6/dist-packages (from pyasn1-modules>=0.2.1->google-auth>=1.4.1->g
oogle-api-python-client) (0.4.8)
```

OAUTH 2.0 SETUP

In [ ]:

```
from apiclient.discovery import build               #it builds service objects for any kind of google api
from google_auth_oauthlib.flow import InstalledAppFlow  #set up oauth2.0 credentials
```

SCOPES

In [ ]:

```
scopes=['https://www.googleapis.com/auth/calendar']
```

In [ ]:

```
flow = InstalledAppFlow.from_client_secrets_file("client_secret.json.json", scopes=scopes)
# create a flow and add the path to client_secret file
credentials=flow.run_console()   #ask me open this link and sign in to my account and enable the permission to see edit and share calendars.
# thsiwill generate a token for that particular user who has gave permssions
```

```
Please visit this URL to authorize this application: https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=85361175655-qmb26l
a9h1bqnk01lovuum00me24qf4b.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3Aoob&scope=https%3A%2F%2
Fwww.googleapis.com%2Fauth%2Fcalendar&state=0Bg06GlGa84D4ZkrNXzy47OxJZemMp&prompt=consent&access_type=offline
Enter the authorization code: 4/2QGFvg_TodXos77p16r0E57PZh3_O335h_VA-T8W5qVgxvnnbEp9rOw
```

In [ ]:

```
import pickle    #We don't have to execute the above block again because we have saved the tokens (aka the credentials) into a pickle (.pkl) file ca
lled "token.pkl",so that now we can directly load the credentials (tokens) from it.'"
 #we can't run the above commands each and every time so that we can the credentials object in a pickle file
```

In [ ]:

```
pickle.dump(credentials, open("token.pkl", "wb"))    #wb-write binary mode
                                                     #dump-puts the credentials into token.pkl
```

In [ ]:

```
credentials=pickle.load(open("token.pkl","rb"))
```

In [ ]:

```
credentials
```

Out[ ]:

<google.oauth2.credentials.Credentials at 0x7fbe22813240>

In [ ]:

```
service = build("calendar","v3", credentials=credentials)   #we are using the build func frm apiclient.discovery to build the calendar api,v3 is the versic
n
                                                            # and i will pass the credentials using OAuth 2.0 access and refresh tokens.
```

GET MY CALENDARS

In [ ]:

```
result=service. calendarList().list().execute()                    #we can know the functions allowed for service obj frm API REF
                                                                   #this will generate the api obj in which we have to use list func
                                                                   #we are executing the api request actually
```

In [ ]:

```
result['items'][0]                          #tis is my primary calendar tis and ID is most probably our mailid
```

Out[ ]:

```
{'accessRole': 'owner',
 'backgroundColor': '#a47ae2',
 'colorId': '24',
 'conferenceProperties': {'allowedConferenceSolutionTypes': ['hangoutsMeet']},
 'defaultReminders': [],
 'etag': '"1595830269105000"',
 'foregroundColor': '#000000',
 'id': 'c_08bbfprc1fmqu4h6equ1g6672c@group.calendar.google.com',
 'kind': 'calendar#calendarListEntry',
 'selected': True,
 'summary': 'My calendar',
 'timeZone': 'Asia/Kolkata'}
```

In [ ]:

```
calendar_id=result['items'][0]['id']
print(calendar_id)
```

```
c_08bbfprc1fmqu4h6equ1g6672c@group.calendar.google.com
```

GET CALENDAR EVENTS

In [ ]:

```
#passing the calendar api into list#get the event through events list api
result = service.events().list(calendarId=calendar_id, timeZone='Asia/Kolkata').execute()
```

In [ ]:

```
result
```

Out[ ]:

```
{'accessRole': 'owner',
 'defaultReminders': [],
 'etag': '"p32cbtck2sfmek0g"',
 'items': [],
 'kind': 'calendar#events',
 'nextSyncToken': 'CJi-soLj7OoCEJi-soLj7OoCGAU=',
 'summary': 'My calendar',
 'timeZone': 'Asia/Kolkata',
 'updated': '2020-07-27T06:11:08.895Z'}
```

GET NEW CALENDAR EVENTS

In [ ]:

```
from datetime import datetime,timedelta
```

In [ ]:

```
start_time = datetime(2020, 7, 28, 11,15,00)
end_time = start_time + timedelta(hours=24)
timeZone = 'Asia/Kolkata'
```

In [ ]:

```
event = {
  'summary': 'It is my star day',
  'location': 'Coimbatore',
  'description': 'Today is my star day!',
  'start': {
    'dateTime': start_time.strftime("%Y-%m-%dT%H:%M:%S"),     #in string pattern the time is generated for tat we are using strf func
    'timeZone': 'Asia/Kolkata',
  },
  'end': {
    'dateTime': end_time.strftime("%Y-%m-%dT%H:%M:%S"),
    'timeZone': 'Asia/Kolkata',
  },
  'recurrence': [
    'RRULE:FREQ=YEARLY;COUNT=2'
  ],
  'attendees': [
    {'email': 'pradeepa.19is@kct.ac.in'},
  ],
  'reminders': {                                  #here we specify when do we want the reminder for the event
    'useDefault': False,
    'overrides': [
      {'method': 'email', 'minutes': 24 * 60},
      {'method': 'popup', 'minutes': 10},
    ],
  },
}
```

In [ ]:

```
service.events().insert(calendarId=calendar_id, body = event).execute()
```

Out[ ]:


```
{'attendees': [{'email': 'pradeepa.19is@kct.ac.in',
   'responseStatus': 'needsAction'}],
 'conferenceData': {'conferenceId': 'rwa-pmmg-cbg',
  'conferenceSolution': {'iconUri': 'https://lh5.googleusercontent.com/proxy/bWvYBOb7O03a7HK5iKNEAPoUNPEXH1CHZjuOkiqxHx8OtyVn9sZ6Ktl8
hfqBNQUUbCDg6T2unnsHx7RSkCyhrKgHcdoosAW8POQJm_ZEvZU9ZfAE7mZIBGr_tDlF8Z_rSzXcjTffVXg3M46v',
   'key': {'type': 'hangoutsMeet'},
   'name': 'Google Meet'},
  'createRequest': {'conferenceSolutionKey': {'type': 'hangoutsMeet'},
   'requestId': 'hnn41jg0dga5g198h5rbkc2u8g',
   'status': {'statusCode': 'success'}},
  'entryPoints': [{'entryPointType': 'video',
    'label': 'meet.google.com/rwa-pmmg-cbg',
    'uri': 'https://meet.google.com/rwa-pmmg-cbg'},
   {'entryPointType': 'phone',
    'label': '+1 567-246-3437',
    'pin': '199407348',
    'regionCode': 'US',
    'uri': 'tel:+1-567-246-3437'}],
  'signature': 'ADR/mfOylWMePLLrhWYsZoxUE3mh'},
 'created': '2020-07-27T09:33:00.000Z',
 'creator': {'email': 'pradeepa.19is@kct.ac.in'},
 'description': 'Today is my star day!',
 'end': {'dateTime': '2020-07-29T11:15:00+05:30', 'timeZone': 'Asia/Kolkata'},
 'etag': '"3191684760410000"',
 'hangoutLink': 'https://meet.google.com/rwa-pmmg-cbg',
 'htmlLink': 'https://www.google.com/calendar/event?eid=NTVwMXA1cmgwaWIyaGVjbzloZmRpcGdmcW9fMjAyMDA3MjhUMDU0NTAwWiBjXzA4Ym
JmcHJjMWZtcXU0aDZlcXUxZzY2NzJjQGc',
 'iCalUID': '55p1p5rh0ib2heco9hfdipgfqo@google.com',
 'id': '55p1p5rh0ib2heco9hfdipgfqo',
 'kind': 'calendar#event',
 'location': 'Coimbatore',
 'organizer': {'displayName': 'My calendar',
```

'email': 'c_08bbfprc1fmqu4h6equ1g6672c@group.calendar.google.com',
'self': True},
'recurrence': ['RRULE:FREQ=YEARLY;COUNT=2'],
'reminders': {'overrides': [{'method': 'popup', 'minutes': 10},
{'method': 'email', 'minutes': 1440}],
'useDefault': False},
'sequence': 0,
'start': {'dateTime': '2020-07-28T11:15:00+05:30',
'timeZone': 'Asia/Kolkata'},
'status': 'confirmed',
'summary': 'It is my star day',
'updated': '2020-07-27T09:33:00.286Z'}

FOR MY REF:

We need to create some credentials to interact with the api.I need to access and edit my google calendar and for that I need to create oauth client 2.0 id.Auth is is a authorisation methodfrom api key which is a bit more safe and secure.It reqiuires the user's permission to generate a token. This oauth consent screen will be visible to the person viewing our application.Credential can be of many forms like it can be a oauth id ...auth works in 2 ways:we create credentials which are specific.next thing is any user will give permission for accessing some data of them.He will give permisiion for mu google console project.Before adding the scopes we need to enable the calendar api so that we could interact with the calendar.