

תוכן

2.....	עבודה מול הברוקר
2.....	הורדת התוכנה
3.....	פתיחת חשבון
6.....	עבודה מול הברוקר הגדרות התוכנה וחיבור לקוד
8.....	שימוש בספרייה חיצונית
11.....	טיפים לבחירת ספרייה חיצונית
14.....	ויזואליזציה
14.....	הכנת המידע
18.....	המשך הכנת המידע
20.....	הגרף הראשון שלי
23.....	משמעות ה ax
26.....	שינוי תכונות הגרף ויצירת גרף כפול

שבוע 3

עבודה מול הברוקר

נכנסים לאתר interactive broker:

- 1) מורידים את התוכנה: TWS LATEST בגרסת online (שמעודכנת כל הזמן).
- 2) פותחים חשבון או נכנסים באופן זמני עם כניסת demo.

הורדת התוכנה

אפשר לרשום בגוגל:

interactive brokers tws latest download

מגיעים להתקנת התוכנה ישירות.

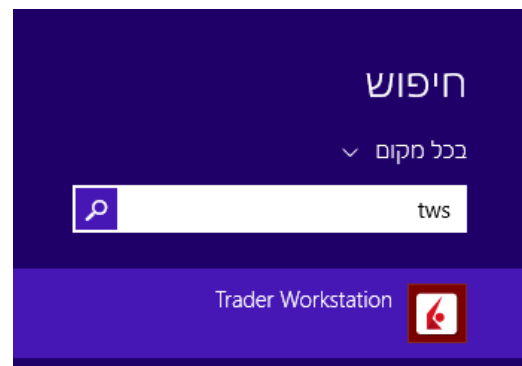
לאחר ההורדה מופיעים באתר צילומי מסך שמראים איך להתקין את התוכנה.

לאחר ההתקנה מתווסף אייקון על שולחן העבודה שדרכו ניתן להתחבר לתוכנה.



או על ידי לחיצה על סמל החלונות/windows בפינה הימנית או השמאלית במסך.

רושמים tws ואז מוצאים את התוכנה Trader Workstation

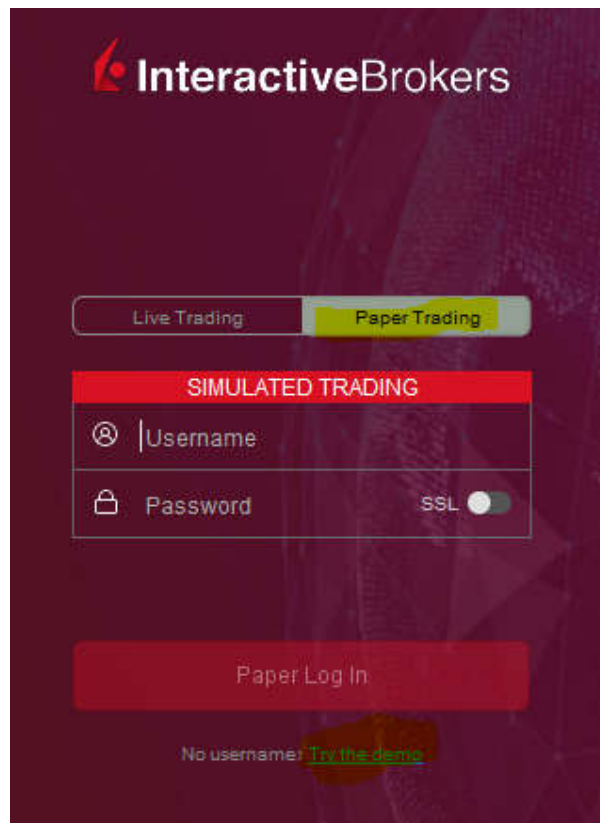


במסך שנפתח בוחרים:

"Paper trading"

אם פתחנו כבר חשבון באתר אז ניתן להתחבר עם יוזר וסיסמא.

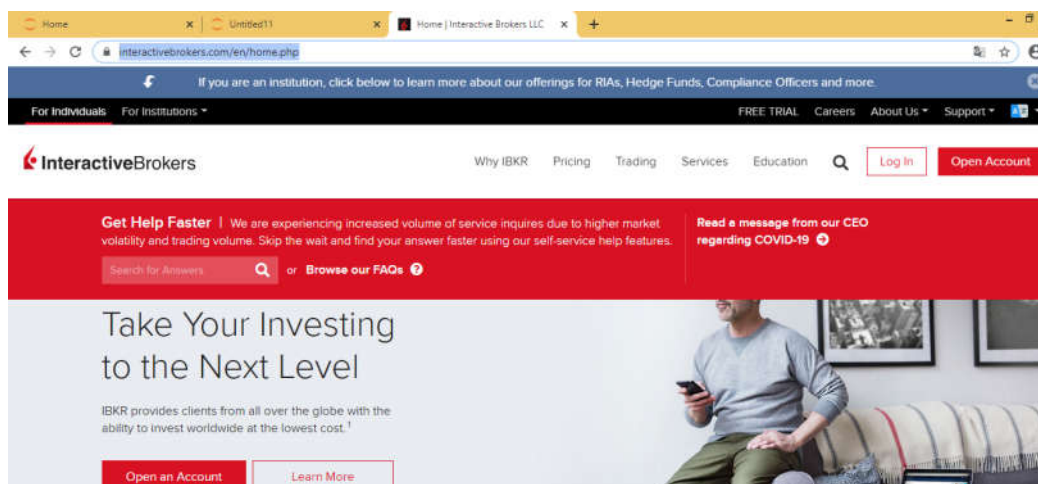
אם רוצים להתחבר באופן זמני דרך כניסת הדמו אז לוחצים על הקישור בתחתית החלון שנפתח "Try the demo".



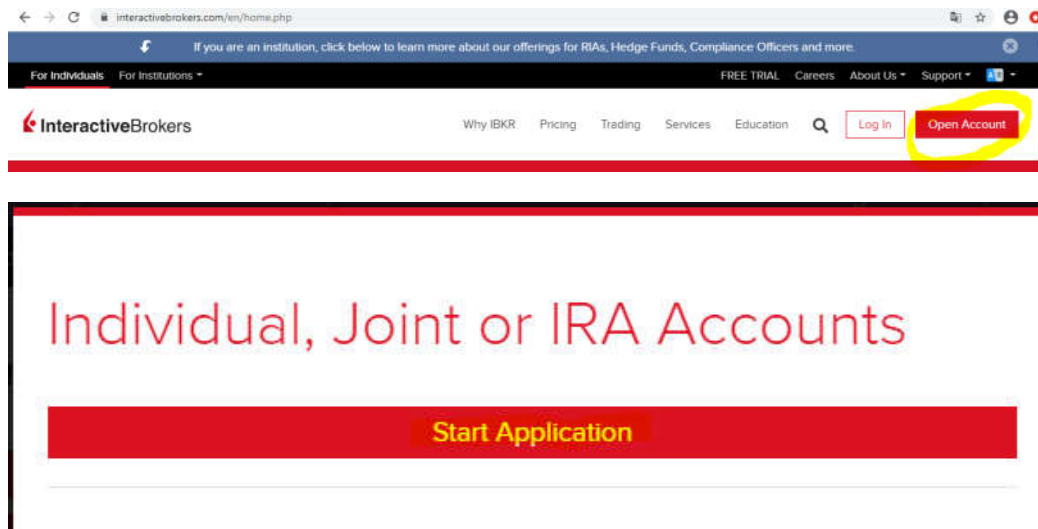
פתיחת חשבון

אתר interactive broker :

<https://www.interactivebrokers.com/en/home.php>



לוחצים על Open Account



מתחילים למלא פרטים

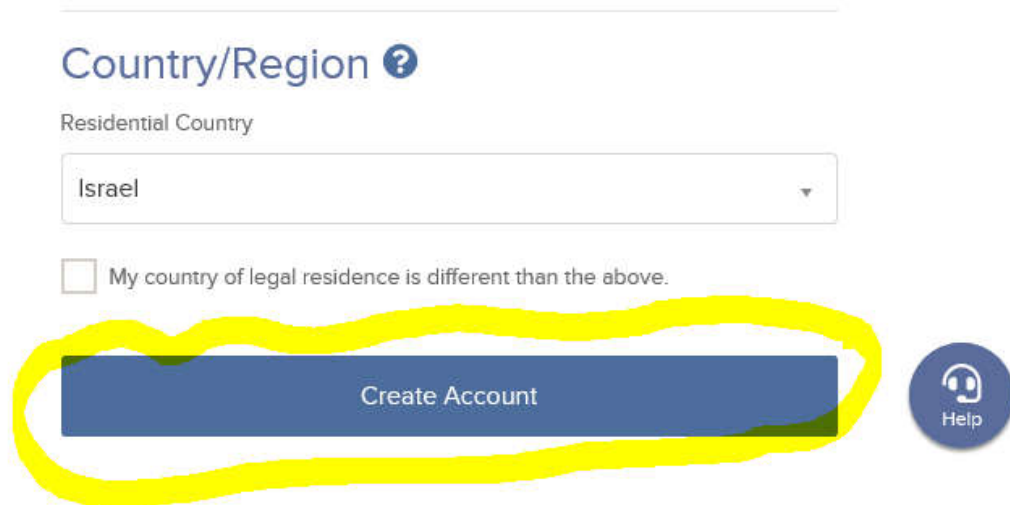
A screenshot of the 'Open an Account' page on InteractiveBrokers. The page is divided into two main sections. On the left, under the heading 'Open an Account', it says 'It's easy. Here's how to get started:' followed by a list of three steps: 1. Create a username and password, 2. Confirm your email address, and 3. Complete the application. Below this list, it says 'For each account holder, you will need:'. On the right, under the heading 'Create a Username and Password', there are three input fields: 'Email Address', 'Username', and 'Password'. Each field has a 'Required' label below it.

ובתחתית הדף ללחוץ על "create account".

אם מאזו שהיא סיבה לא רואים את כפתור זה בסוף הדף אז הפתרונות:

או ללחוץ על f5 כדי שהדף יטען שוב ואז לראות אם מופיע בסוף הדף.

או להיכנס דרך דפדפן מסוג אחר (chrome/explorer).



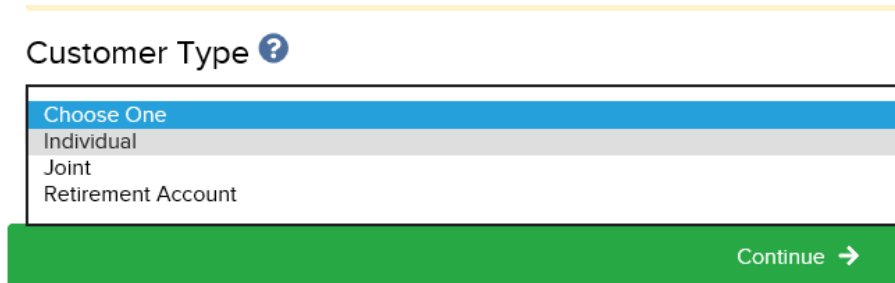
אחרי שלוחצים create account נשלח אימייל מinteractive brokers ורק צריך ללחוץ בו על "verify account" (רצוי לעשות זאת דרך המחשב ולא דרך הטלפון הנייד).

מגיעים למסך שצריך להזין שם משתמש וסיסמא.

מזינים את שם המשתמש והסיסמא שיצרנו במסך פתיחת חשבון.

מגיעים למסך שבו צריכים לבחור איזה סוג לקוח אנחנו.

בוחרים "individual"



****אם יש מסך שנתקע ולא ממשיך לשלב הבא אז לוחצים f5 ואז הוא נטען שוב.

ממשיכים למלא פרטים:

About You

Contact Information

Salutation: Choose

First Name: 1

Middle Name(s):

Last Name: Required

Suffix: None



Why do we collect this information?

We are required to collect the personal information in this application to comply with government anti-money laundering regulations. We understand that

מבקש מספר tax id אפשר לבחור באופציה הזו ואז לא דרוש להכניס מספר.

- ☐ I have a US taxpayer identification number and I will add it to the W8 tax form.
- ☐ The specified country does not issue TINs to its residents.
- ☒ I am not legally required to obtain a TIN from the specified country.
- ☐ TINs are issued, however, account holder is exempt from this requirement under the laws of the specified country.
- ☐ Other
- ☒ Add Tax Residency

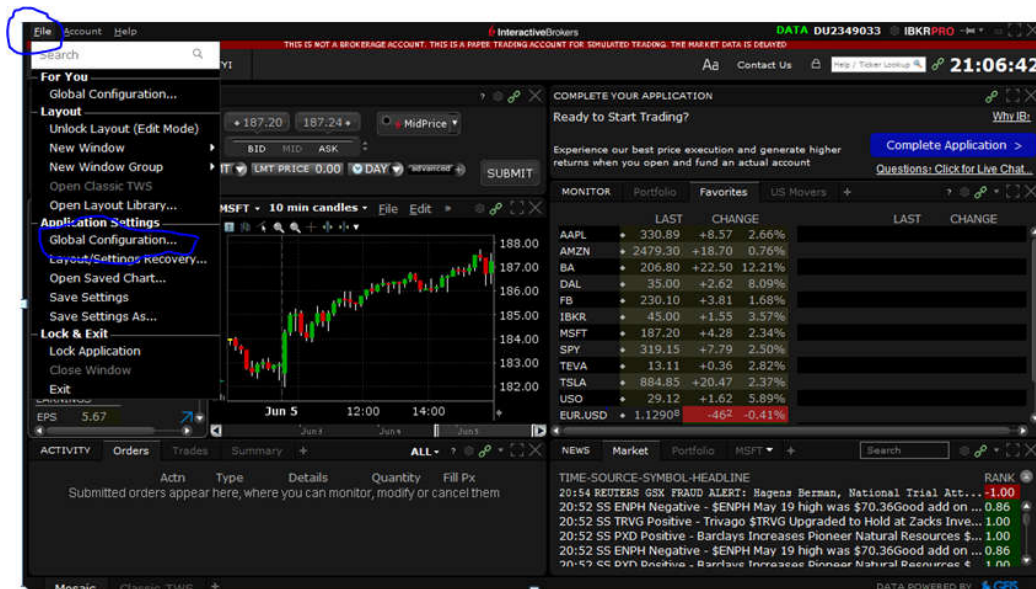
מסיימים את המסך המפרך הזה ועוברים הלאה.....עד שמסיימים.

***אם יש חשבון דמו שלא מתחברים אליו הרבה זמן אז הוא נמחק.

עבודה מול הברוקר הגדרות התוכנה וחיבור לקוד

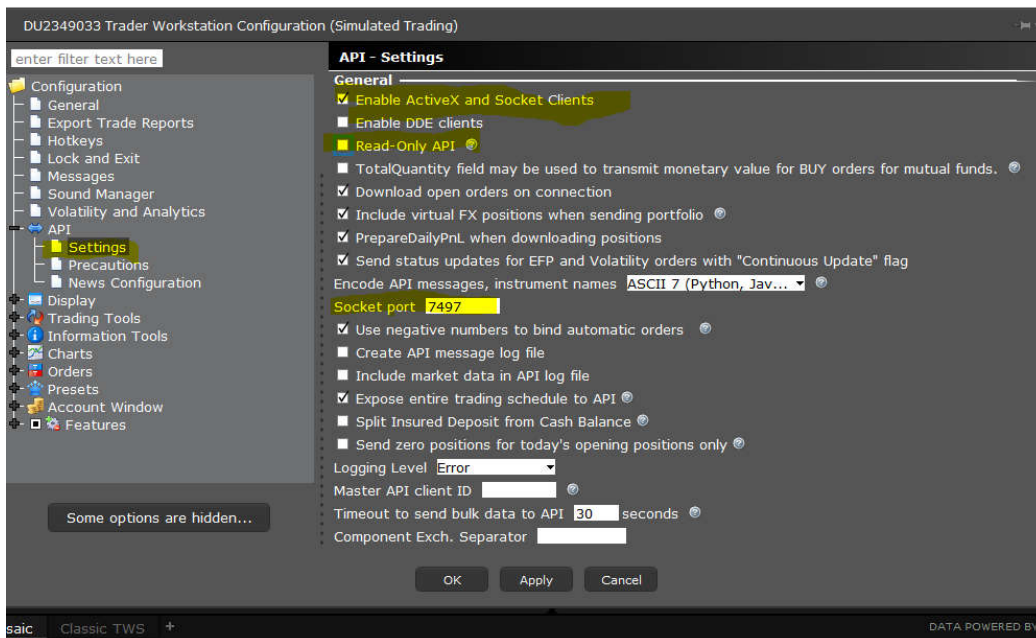
באתר יש יכולת להגיש פעולות, לקבל נתונים, לראות נתונים, לעשות ניתוחים, לעצור תהליכים ועוד... אבל אנחנו נעשה הכל דרך הפיתוח.

נפתח את אפליקציית הברוקר



אם לנלחץ על <file> <-configuration> <-api>
api כאן מאפשר לתקשר בין הפייתון לאפליקציה המקומית וממנה לאתר הכללי של
interactive broker.

- בחלון שנפתח ללחוץ על (+) של <-api>
 (1) להוסיף v ל Enable activex and socket clients
 (2) להוריד v מ api מ read only
 (3) ללחוץ על ok בתחתית החלון על מנת לשמור את ההגדרות.



Socket port = 7497

localhost - זהו string שמומר אוטומטית ל ip 127.0.0.1 שזו כתובת לוקאלית במחשב שלנו.

איך נגשים מה Jupiter לאפליקציית הברוקר המקומית שנגשת אל הברוקר שנמצא בלונדון ומקבלת ממנו את המידע.

כך המידע מתאכסן אצלנו ב jupyter.

שימוש בספרייה חיצונית

חבילה חיצונית:

תוכנת אנקונדה מגיעה עם חבילות שלה. יש אפשרות להוסיף חבילות חדשות.

כדי להתקין חבילה חדשה משתמשים בפקודה pip.

את הפקודה pip מפעילים או דרך ה command line או דרך jupyter

אם רוצים להתקין זאת דרך ה jupyter אז צריך לפני הקו לשים סימן קריאה ואז הוא יודע שהוא מריץ זאת אומנם דרך jupyter אבל מאחורי הקלעים מריץ זאת דרך ה command line.

איך מתקינים דרך jupyter:

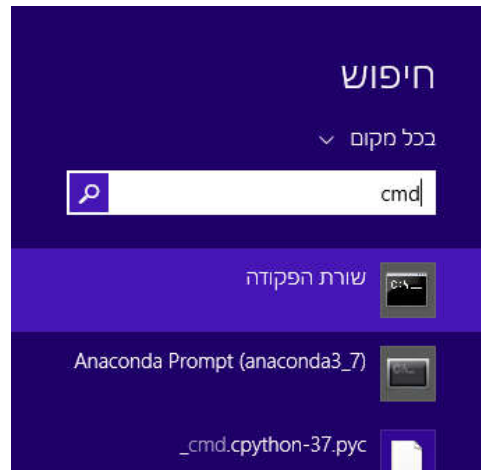
```
In [1]: ! pip install ib_insync
```

```
Collecting ib_insync  
  Downloading ib_insync-0.7.1-py2.py3-none-any.whl (30 kB)  
Installing collected packages: ib_insync  
Successfully installed ib_insync-0.7.1
```

איך מתקינים זאת דרך ה command line:

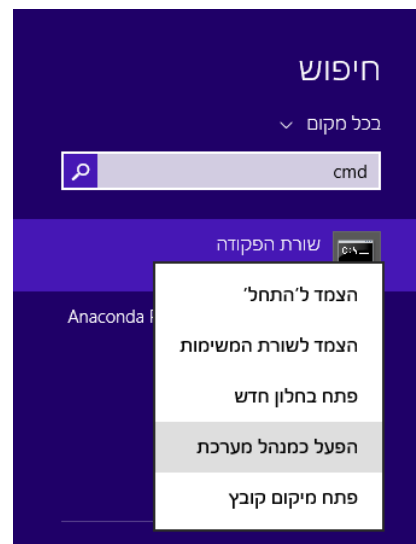
לוחצים על סמל החלונות שבפינה הימנית או השמאלית התחתונה במסך.

רושמים cmd ובוחרים בשורת פקודה או command line.



עדיף ללחוץ קליק ימני על שורת הפקודה או command line ולבחור להפעיל אותה כמנהל מערכת.

למה: כי לא תמיד יש לנו הרשאות מלאות על המחשב שלנו, לצורך הגנה שלא נתקין או נמחק כל מיני דברים שעלולים לפגוע בו. הגנה זו יכולה למנוע מאתנו לבצע התקנה בצורה מוצלחת לכן עדיף לבחור הפעלה כמנהל מערכת כי כמנהל מערכת יש לנו הרשאה מלאה להתקנת דברים במחשב שלנו.



יפתח לנו חלון

```
Administrator: שורת הפקודה
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
C:\WINDOWS\system32>
```

נרשום בו: `pip install ib_insync` הפעם ללא סימן קריאה כי אנחנו כבר נמצאים ב `command line`

```
Administrator: שורת הפקודה - pip install ib_insync
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
C:\WINDOWS\system32>pip install ib_insync
```

הספרייה מותקנת בהצלחה

```
Administrator: שורת הפקודה
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
C:\WINDOWS\system32>pip install ib_insync
Collecting ib_insync
  Downloading ib_insync-0.9.61-py3-none-any.whl (69 kB)
    |#####| 69 kB 209 kB/s
Collecting eventkit
  Downloading eventkit-0.8.6-py3-none-any.whl (31 kB)
Collecting nest-asyncio
  Downloading nest_asyncio-1.3.3-py3-none-any.whl (4.7 kB)
Requirement already satisfied: numpy in c:\users\user\anaconda3_7\lib\site-packages (from eventkit->ib_insync) (1.18.1)
Installing collected packages: eventkit, nest-asyncio, ib_insync
Successfully installed eventkit-0.8.6 ib_insync-0.9.61 nest-asyncio-1.3.3
C:\WINDOWS\system32>
```

מריצים את ההתחברות:

```
from ib_insync import *
util.startLoop() # uncomment this line when in a notebook

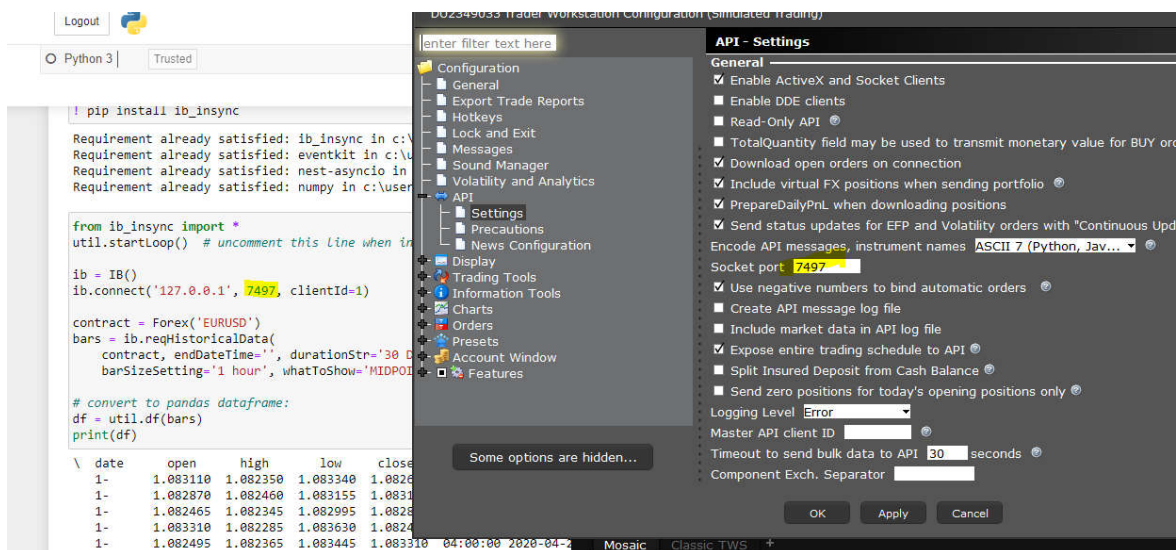
ib = IB()
```

```
ib.connect('127.0.0.1', 7497, clientId=1)

contract = Forex('EURUSD')
bars = ib.reqHistoricalData(
    contract, endDateTime='', durationStr='30 D',
    barSizeSetting='1 hour', whatToShow='MIDPOINT', useRTH=True)

# convert to pandas dataframe:
df = util.df(bars)
print(df)
```

***חשוב לשים לב שמספר port זה בפקודת ההתחברות ובהגדרות אפליקציית הברוקר.



פנינו ל `ib` שזה אובייקט שמאפשר לתקשר עם אפליקציית ברוקר והפעלנו את הפונקציה `reqHistoricalData` וקיבלנו מידע שהמרנו אותו לתוך `data frame`.
נעמיק בנושא זה בהמשך.

טיפים לבחירת ספרייה חיצונית

הספריות הן `open source` ונגישות לציבור כדי שנוכל לסמוך עליהן.

בדרך כלל הספריות מאוכסנות ב `github`

כשננסים לספרייה אפשר לראות מי כתב אותה וניתן לראות את הפרופיל שלו, לראות אם הוא פעיל.

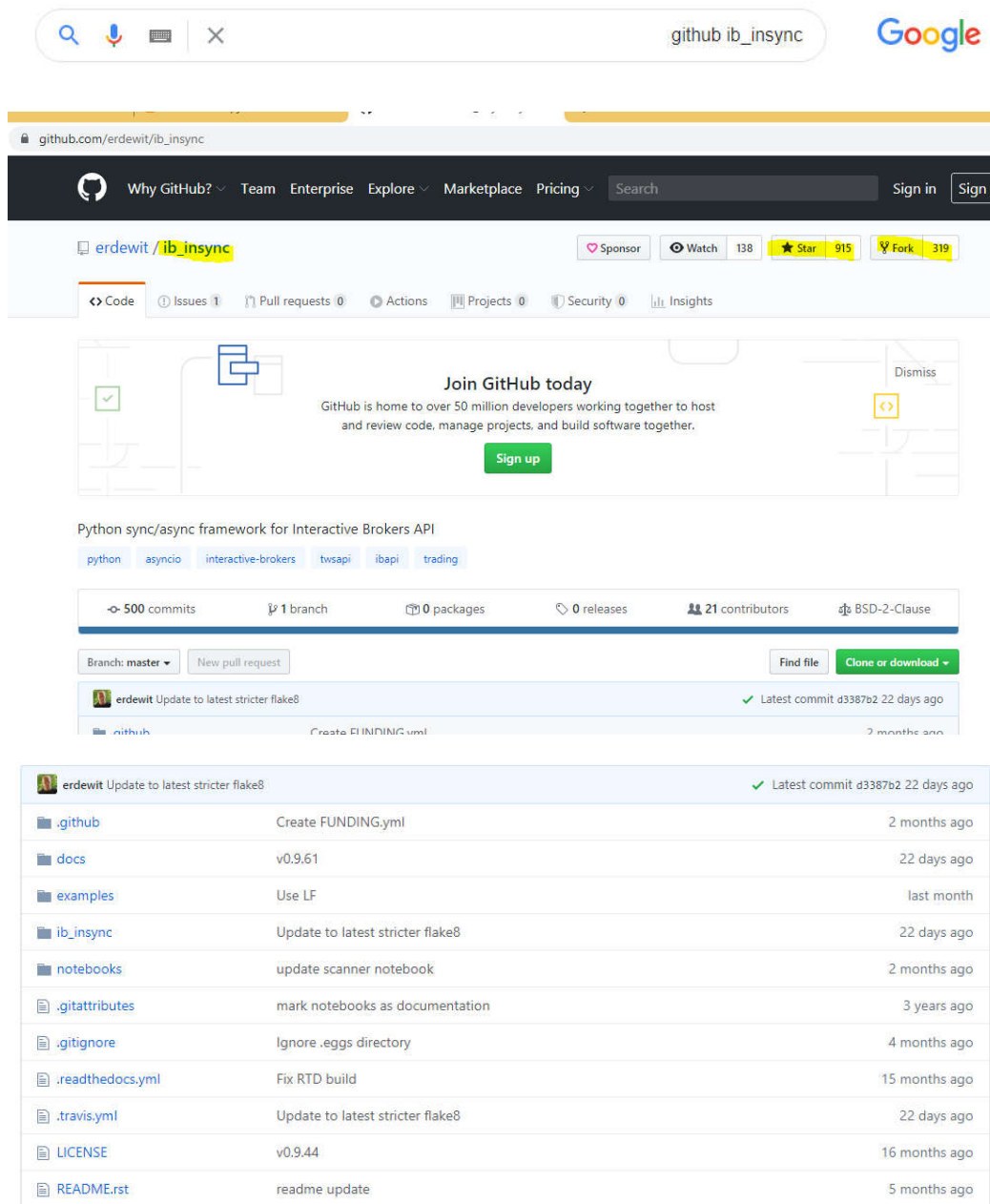
כמות הstar ים – משקף את הבעת הערכה של המשתמשים לספרייה. אם זה נמוך אז זה חשוד.

Fork = העתקת הקוד של הספרייה וביצוע שינויים.

אנחנו רוצים שתהיה לספרייה רמה גבוהה של fork.

צריך שיהיה הסבר על ביצוע ההתקנה, הסבר על הספרייה ודוגמאות.

בספריות טובות ומתוחזקות יש את מידע זה.



github.com/erdewit/ib_insync

Why GitHub? Team Enterprise Explore Marketplace Pricing Search Sign in Sign up

erdewit / **ib_insync** Sponsor Watch 138 Star 915 Fork 319

Code Issues 1 Pull requests 0 Actions Projects 0 Security 0 Insights

Join GitHub today
GitHub is home to over 50 million developers working together to host and review code, manage projects, and build software together.
Sign up

Python sync/async framework for Interactive Brokers API
python asyncio interactive-brokers twsapi ibapi trading

500 commits 1 branch 0 packages 0 releases 21 contributors BSD-2-Clause

Branch: master New pull request Find file Clone or download

erdewit Update to latest stricter flake8 Latest commit d3387b2 22 days ago

.github	Create FUNDING.yml	2 months ago
docs	v0.9.61	22 days ago
examples	Use LF	last month
ib_insync	Update to latest stricter flake8	22 days ago
notebooks	update scanner notebook	2 months ago
.gitattributes	mark notebooks as documentation	3 years ago
.gitignore	Ignore .eggs directory	4 months ago
.readthedocs.yml	Fix RTD build	15 months ago
.travis.yml	Update to latest stricter flake8	22 days ago
LICENSE	v0.9.44	16 months ago
README.rst	readme update	5 months ago

נכנסים ל readme

Introduction

The goal of the IB-insync library is to make working with the [Trader Workstation API](#) from Interactive Brokers as easy as possible.

The main features are:

- An easy to use linear style of programming;
- An [IB component](#) that automatically keeps in sync with the TWS or IB Gateway application;
- A fully asynchronous framework based on [asyncio](#) and [eventkit](#) for advanced users;
- Interactive operation with live data in Jupyter notebooks.

Be sure to take a look at the [notebooks](#), the [recipes](#) and the [API docs](#).

Installation

```
pip install ib_insync
```

For Python 3.6 install the `dataclasses` package as well (newer Python versions already have it):

```
pip install dataclasses
```

Requirements:

- Python 3.6 or higher;
- A running TWS or IB Gateway application (version 972 or higher). Make sure the [API port is enabled](#) and 'Download open orders on connection' is checked.

The `ibapi` package from IB is not needed.

Example

This is a complete script to download historical data:

```
from ib_insync import *
# util.startLoop() # uncomment this line when in a notebook

ib = IB()
ib.connect('127.0.0.1', 7497, clientId=1)

contract = Forex('EURUSD')
bars = ib.reqHistoricalData(
    contract, endDateTime='', durationStr='30 D',
    barSizeSetting='1 hour', whatToShow='MIDPOINT', useRTH=True)

# convert to pandas dataframe:
df = util.df(bars)
print(df)
```

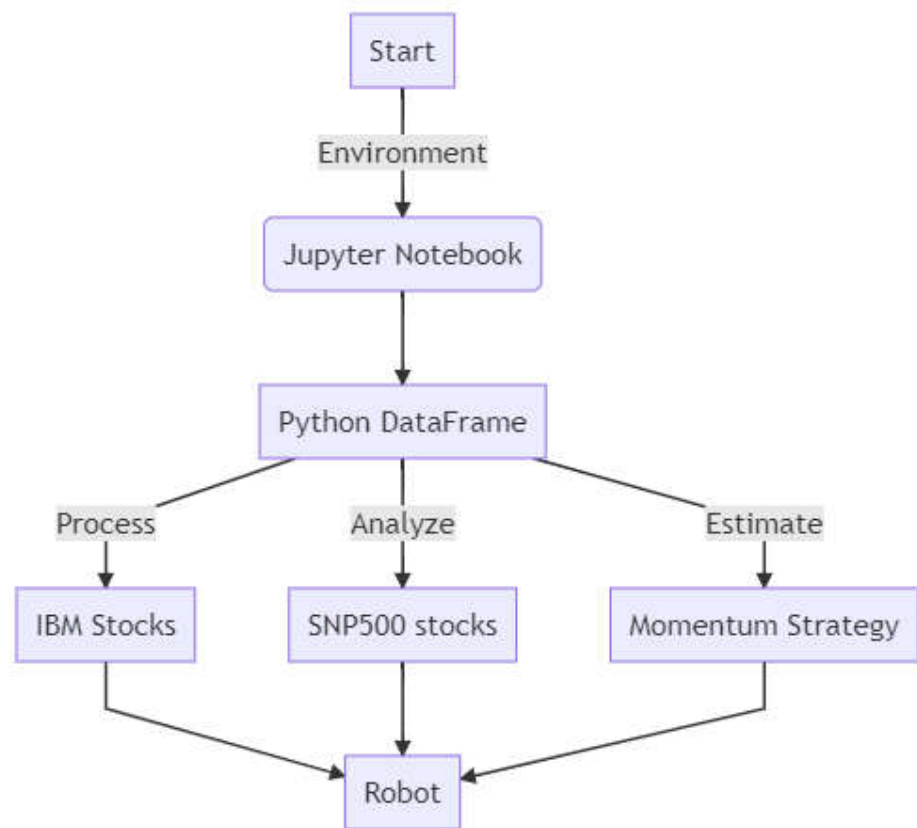
Output:

```
      date  open  high  low  close  volume  \
0  2018-11-10 23:15:00  1.127875  1.128050  1.127735  1.127825  1
```

ניתן לראות גם את הקוד שאנחנו השתמשנו בו...

ויזואליזציה

הדרך שנעבור בקורס



הכנת המידע

דרך לטעון קובץ שנמצא באיזו כתובת url:

מתקינים את ספרייה urllib

שומרים במשתנה את ה url שמוביל לקובץ

ומפעילים את פקודת הייבוא urllib.retrieve

```
In [ ]: !pip install urllib.
```

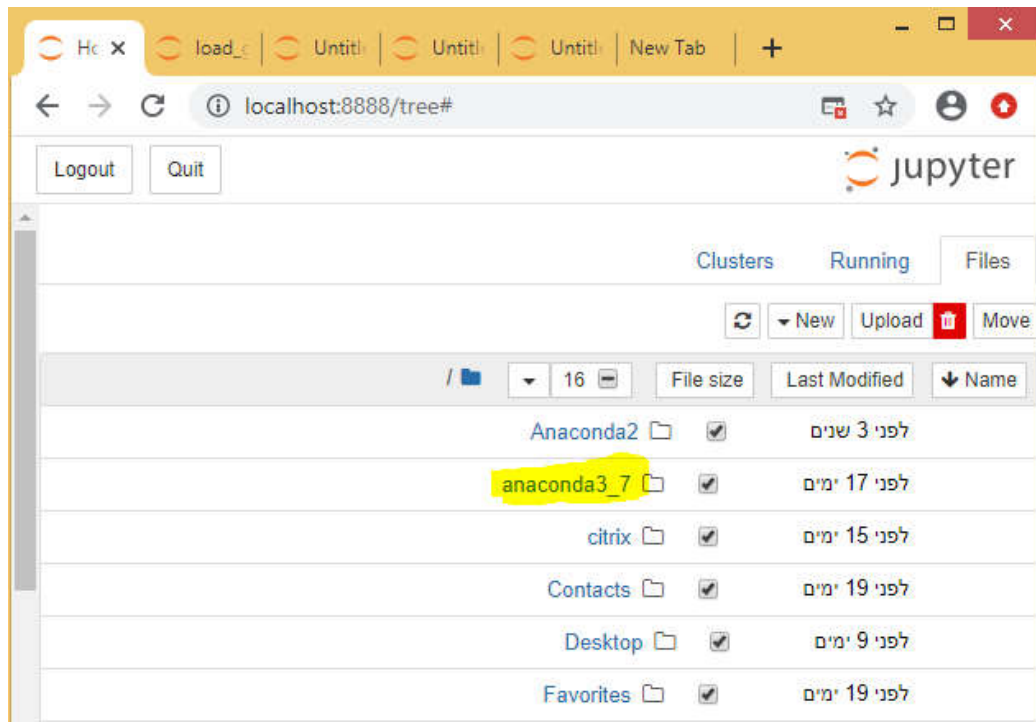
```
In [17]: import urllib
url='https://storage.googleapis.com/algotrading.dataxcelerator.com/all_stock_4yr.csv'
urllib.urlretrieve(url,'all_stock_4yr.csv')
```

```
Out[17]: ('all_stock_4yr.csv', <httplib.HTTPMessage instance at 0x06858D50>)
```

מכיוון שלא רשמנו לו איפה לשמור אותו. הקובץ המיובא נשמר בתיקיית ברירת המחדל של jupyter.

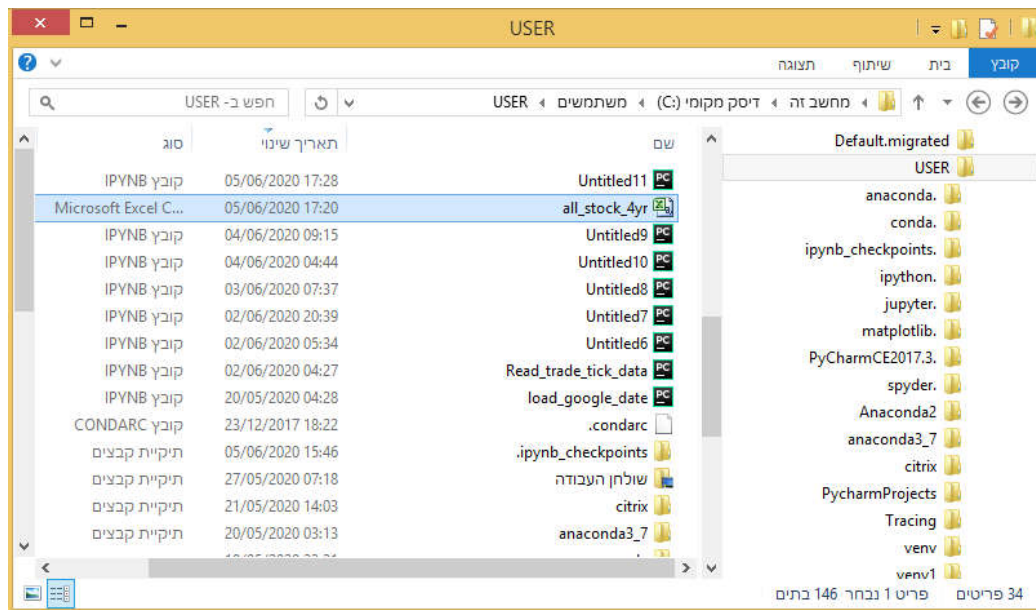
איך יודעים איזו תיקייה זו:

כשפותחים את jupyter נפתח לנו המקום ברירת המחדל של תוכנה זו שבו גם יצרנו את ספריית anaconda בזמן ההתקנה.



המיקום הזה נמצא במחשב הזה כאן:

ובו גם נראה שנשמרות כל המחברות קוד שאנחנו יוצרים.



```
In [19]: import pandas as pd
df2=pd.read_csv('all_stock_4yr.csv')
df2
```

אנחנו יכולים גם לטעון את הקובץ באותה הדרך שטענו עד עכשיו אחרי ששמרנו אותו בספרייה c:\Temp\stock_4yr.csv

```
In [1]: import pandas as pd
df=pd.read_csv('c:\Temp\stock_4yr.csv')
df
```

Out[1]:

	Unnamed: 0	date	open	high	low	close	volume	Name	dt
0	0	08/02/2013	15.070	15.12	14.630	14.75	8407500	AAL	08/02/2013
1	1	11/02/2013	14.890	15.01	14.260	14.46	8882000	AAL	11/02/2013
2	2	12/02/2013	14.450	14.51	14.100	14.27	8126000	AAL	12/02/2013
3	3	13/02/2013	14.300	14.94	14.250	14.66	10259500	AAL	13/02/2013
4	4	14/02/2013	14.940	14.96	13.160	13.99	31879900	AAL	14/02/2013
5	5	15/02/2013	13.930	14.61	13.930	14.50	15628000	AAL	15/02/2013
6	6	19/02/2013	14.330	14.56	14.080	14.26	11354400	AAL	19/02/2013
7	7	20/02/2013	14.170	14.26	13.150	13.33	14725200	AAL	20/02/2013

נבדוק מה שמות העמודות בקובץ

```
In [21]: df.columns
```

```
Out[21]: Index([u'Unnamed: 0', u'date', u'open', u'high', u'low', u'close', u'volume',
u'Name', u'dt'],
dtype='object')
```


אנחנו רוצים לראות את הטבלה ללא העמודה הראשונה "Unnamed:0"

יש כמה דרכים:

כמו שאנחנו יודעים להציג עמודה אחת אנחנו יכולים להציג list של עמודות:

```
In [22]: df[['date', 'Name', 'open', 'close', 'high', 'low', 'volume']]
```

Out[22]:

	date	Name	open	close	high	low	volume
0	08/02/2013	AAL	15.070	14.75	15.12	14.630	8407500
1	11/02/2013	AAL	14.890	14.46	15.01	14.260	8882000
2	12/02/2013	AAL	14.450	14.27	14.51	14.100	8126000
3	13/02/2013	AAL	14.300	14.66	14.94	14.250	10259500

אנחנו יכולים לשמור במשתנה את כל שמות העמודות החל מעמודה 1 (ללא עמודה 0).

ואח"כ להציב אותו בתוך הסוגריים המרובעים כדי שיוציג רק את עמודות אלו.

```
In [23]: df.columns[1:]
```

Out[23]: Index([u'date', u'open', u'high', u'low', u'close', u'volume', u'Name', u'dt'], dtype='object')

```
In [24]: col_names=df.columns[1:]
```

```
In [25]: df[col_names]
```

Out[25]:

	date	open	high	low	close	volume	Name	dt
0	08/02/2013	15.070	15.12	14.630	14.75	8407500	AAL	08/02/2013
1	11/02/2013	14.890	15.01	14.260	14.46	8882000	AAL	11/02/2013
2	12/02/2013	14.450	14.51	14.100	14.27	8126000	AAL	12/02/2013
3	13/02/2013	14.300	14.94	14.250	14.66	10259500	AAL	13/02/2013
4	14/02/2013	14.940	14.96	13.160	13.99	31879900	AAL	14/02/2013
5	15/02/2013	13.930	14.61	13.930	14.50	15628000	AAL	15/02/2013

ועוד דרך:

```
In [26]: df[df.columns[1:]]
```

Out[26]:

	date	open	high	low	close	volume	Name	dt
0	08/02/2013	15.070	15.12	14.630	14.75	8407500	AAL	08/02/2013
1	11/02/2013	14.890	15.01	14.260	14.46	8882000	AAL	11/02/2013
2	12/02/2013	14.450	14.51	14.100	14.27	8126000	AAL	12/02/2013
3	13/02/2013	14.300	14.94	14.250	14.66	10259500	AAL	13/02/2013
4	14/02/2013	14.940	14.96	13.160	13.99	31879900	AAL	14/02/2013
5	15/02/2013	13.930	14.61	13.930	14.50	15628000	AAL	15/02/2013

המשך הכנת המידע

המשימה לסנן את המניה AAL, לסנן חודש מסוים ולראות את vol שהיה באותו חודש.

איך עושים זאת:

נחלץ את היום מעמודה date שהיא string. ניקח ממנה את 2 האברים הראשונים ונמיר אותם למספר כי המיון של string לא תמיד יתנהג כמו שאנחנו רוצים. למשל יכול למיין כך שאחרי 1 יהיה 11 אז עדיף להמיר למספר כדי שלא יהיו טעויות וימוין בטוח לפי סדר רציף. בנוסף, נוסיף זאת כעמודה חדשה בשם day ב data frame שלנו.

```
In [35]: df['day']=df['date'].str[0:2].astype(int)
df
```

Out[35]:

	Unnamed: 0	date	open	high	low	close	volume	Name	dt	day
0	0	08/02/2013	15.070	15.12	14.630	14.75	8407500	AAL	08/02/2013	8
1	1	11/02/2013	14.890	15.01	14.260	14.46	8882000	AAL	11/02/2013	11
2	2	12/02/2013	14.450	14.51	14.100	14.27	8126000	AAL	12/02/2013	12
3	3	13/02/2013	14.300	14.94	14.250	14.66	10259500	AAL	13/02/2013	13
4	4	14/02/2013	14.940	14.96	13.160	13.99	31879900	AAL	14/02/2013	14
5	5	15/02/2013	13.930	14.61	13.930	14.50	15628000	AAL	15/02/2013	15
6	6	19/02/2013	14.330	14.56	14.080	14.26	11354400	AAL	19/02/2013	19
7	7	20/02/2013	14.170	14.28	13.150	13.33	14725200	AAL	20/02/2013	20

נסנן את ה data frame כך שיציג רק מניות עם הסימול AAL.

מתוך אותו data frame מסונן ניקח את עמודה date שהיא מסוג string ונחלץ ממנה רק את התווים שמייצגים חודש ושנה. במקרה שלנו מתו מינוס 7 ומעלה או מתו 3 ומעלה.

נשים את מידע החודש ושנה של מניות AAL בעמודה חדשה בשם year_month.

זה אומר שיהיו ערכים של חודש ושנה רק במקרה שהמניה היא AAL. בשאר המקרים, בשורות של מניות אחרות, אז השדה year_month יהיה ריק.

data frame מילוי מידע
שורה
מאפיין חודש וסדר

In [36]: df['year_month']=df[df['Name']=='AAL']['date'].str[3:]

In [37]: df

Out[37]:

	Unnamed: 0	date	open	high	low	close	volume	Name	dt	day	year_month
0	0	08/02/2013	15.070	15.12	14.630	14.75	8407500	AAL	08/02/2013	8	02/2013
1	1	11/02/2013	14.890	15.01	14.260	14.46	8882000	AAL	11/02/2013	11	02/2013
2	2	12/02/2013	14.450	14.51	14.100	14.27	8126000	AAL	12/02/2013	12	02/2013
3	3	13/02/2013	14.300	14.94	14.250	14.66	10259500	AAL	13/02/2013	13	02/2013
4	4	14/02/2013	14.940	14.96	13.160	13.99	31879900	AAL	14/02/2013	14	02/2013
5	5	15/02/2013	13.930	14.61	13.930	14.50	15628000	AAL	15/02/2013	15	02/2013
6	6	19/02/2013	14.330	14.56	14.080	14.26	11354400	AAL	19/02/2013	19	02/2013
7	7	20/02/2013	14.170	14.26	13.150	13.33	14725200	AAL	20/02/2013	20	02/2013
8	8	21/02/2013	13.620	13.95	12.900	13.37	11922100	AAL	21/02/2013	21	02/2013

איך נראות שורות של מניות שהן לא AAL

23	23	14/03/2013	15.500	16.00	15.500	16.20	5555500	AAL	14/03/2013	14	03/2013
24	24	15/03/2013	16.450	16.54	15.880	15.98	17667700	AAL	15/03/2013	15	03/2013
25	25	18/03/2013	15.800	16.33	15.710	16.29	6514100	AAL	18/03/2013	18	03/2013
26	26	19/03/2013	16.480	16.85	16.410	16.78	11805300	AAL	19/03/2013	19	03/2013
27	27	20/03/2013	17.130	17.33	16.870	17.23	10819800	AAL	20/03/2013	20	03/2013
28	28	21/03/2013	17.210	17.43	16.870	17.00	10740800	AAL	21/03/2013	21	03/2013
29	29	22/03/2013	17.100	17.29	16.770	16.86	8545200	AAL	22/03/2013	22	03/2013
...
605880	618984	16/11/2017	69.970	70.49	69.710	70.41	2055575	ZTS	16/11/2017	16	NaN
605881	618985	17/11/2017	70.240	71.12	70.240	70.79	2355140	ZTS	17/11/2017	17	NaN
605882	618986	20/11/2017	70.800	71.14	70.490	70.95	2268805	ZTS	20/11/2017	20	NaN
605883	618987	21/11/2017	71.030	72.14	70.970	71.36	2725360	ZTS	21/11/2017	21	NaN
605884	618988	22/11/2017	71.330	71.83	70.950	71.07	1437222	ZTS	22/11/2017	22	NaN
605885	618989	24/11/2017	71.170	71.42	70.930	71.29	1200253	ZTS	24/11/2017	24	NaN
605886	618990	27/11/2017	71.490	71.80	71.220	71.54	1997531	ZTS	27/11/2017	27	NaN

שלב אחרון:

נסמן את ה data frame לפי חודש ושנה ספציפיים.

מתוך ה data frame המסונן הזה, נשלוף 3 עמודות:

Date, volume, day ונמייין את התוצאה לפי העמודה day.

נשמור הכל בתוך משתנה שנקרא לו volume.

```
In [45]: volume=df[df['year_month']=='12/2017'][['day','date','volume']].sort_values('day')
volume
```

```
Out[45]:
```

	day	date	volume
1213	1	01/12/2017	6229631
1214	4	04/12/2017	7113822
1215	5	05/12/2017	3594043
1216	6	06/12/2017	2835542
1217	7	07/12/2017	3046954
1218	8	08/12/2017	3817896
1219	11	11/12/2017	2835633
1220	12	12/12/2017	2798179
1221	13	13/12/2017	3035241
1222	14	14/12/2017	4503105
1223	15	15/12/2017	9074165
1224	18	18/12/2017	3236066
1225	19	19/12/2017	3019807
1226	20	20/12/2017	5172839
1227	21	21/12/2017	7125891

הגרף הראשון שלי

לצורך יצירת גרפים נצטרך לייבא את הספרייה matplotlib.

```
%matplotlib inline
import matplotlib.pyplot as plt
```

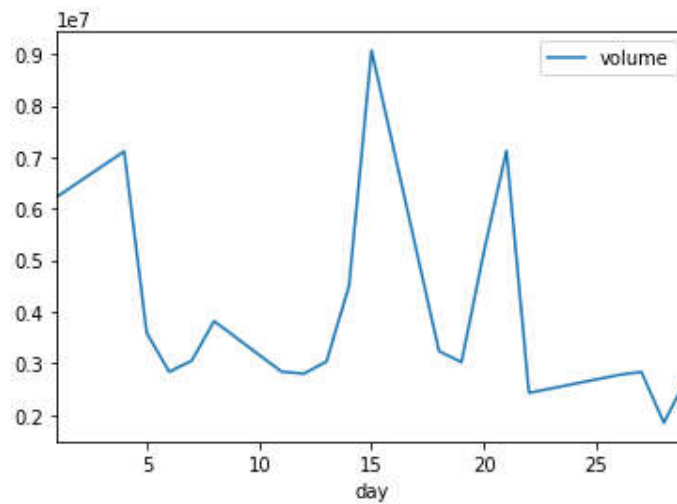
יש פונקציה בשם plot שמקבלת 2 פרמטרים:

x ו y .

גם אם לא עושים השמה בפרמטר כך שיראו מה מקבל x ומה y אז הפונקציה תעבוד. לפונקציה יש סדר שמצפה לקבל קודם את x ואח"כ את y כך שיכולים פשוט לתת בפרמטר ערך ראשון פסיק ערך שני מבלי לרשום: $x = y$. הפונקציה תדע שהערך הראשון הוא x והשני זה y .

```
In [47]: volume.plot(x='day',y='volume')
```

```
Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x1b6ac950>
```



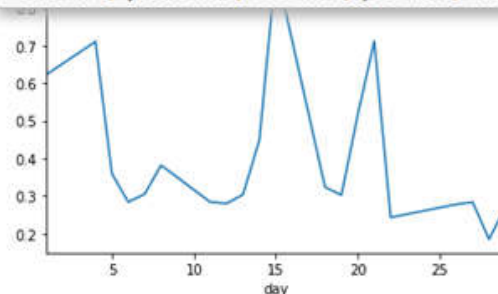
Shift+tab יציג לנו את הקלי help לגבי אותה פונקציה שאנחנו עומדים עליה.

כך נדע פרטים על הפונקציה ומה היא מקבלת בפרמטרים שלה.

```
In [46]: %matplotlib inline
import matplotlib.pyplot
```

```
In [47]: volume.plot(x='day',y='volume')
```

```
Out[47]: Signature: volume.plot(self, x=None, y=None, kind='line', ax=None, subplots=
false, sharex=None, sharey=False, layout=None, figsize=None, use_index=True, title=
None, grid=None, legend=True, style=None, logx=False, logy=False, loglog=False, xt
icks=None, yticks=None, xlim=None, ylim=None, rot=None, fontsize=None, colormap=No
```



אם נלחץ על ה"+" בחלון העזרה שנפתח אז נראה המשך הסברים לגבי אותה פונקציה ודוגמאות.

```
In [46]: %matplotlib inline
import matplotlib.pyplot
```

```
In [48]: volume.plot(x='day',y='volume')
```

```
Out[48]: kind : str
- 'line' : line plot (default)
- 'bar' : vertical bar plot
- 'barh' : horizontal bar plot
- 'hist' : histogram
- 'box' : boxplot
- 'kde' : Kernel Density Estimation plot
- 'density' : same as 'kde'
- 'area' : area plot
- 'pie' : pie plot
- 'scatter' : scatter plot
```

סוגי גרפים פופולריים:

Scatter- גרף נקודות. נוח להשתמש בו כאשר רוצים להשוות בין כמה מערכי נתונים (כמה עמודות)

Line – מציג קו מגמה.

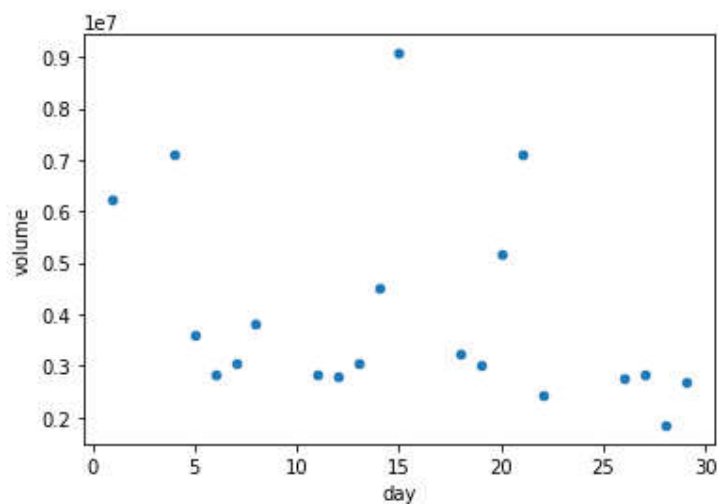
Histogram- מתאר כמות מסוימת של נתונים.

Boxplot- מתאר כמות מסוימת של נתונים שכוללת: ממוצע, חציון, רבעונים ועוד.

Pie- כאשר רוצים להציג יחס אחוזים מהשלם.

```
In [49]: volume.plot(x='day',y='volume',kind='scatter')
```

```
Out[49]: <matplotlib.axes._subplots.AxesSubplot at 0x1a786f30>
```

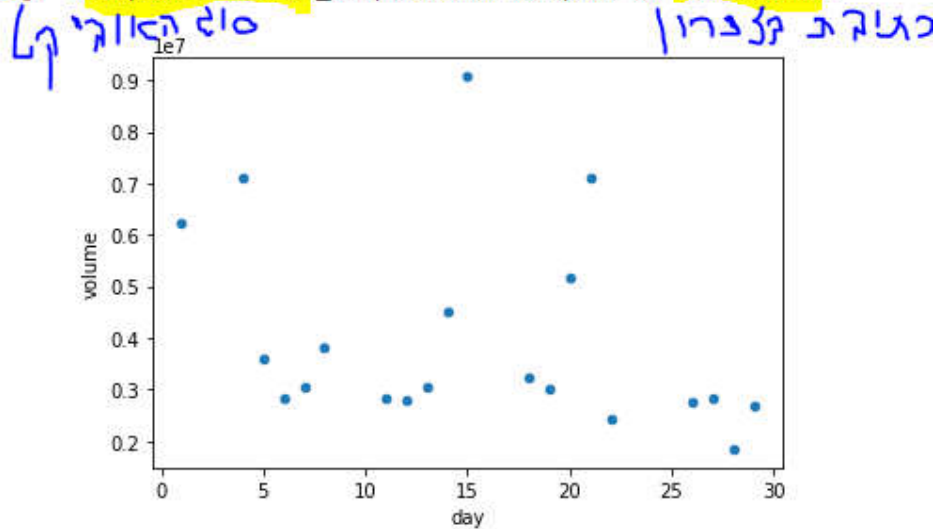


משמעות ה ax

האיבר שחוזר מהפונקציה plot הוא אובייקט מסוג matplotlib.axes

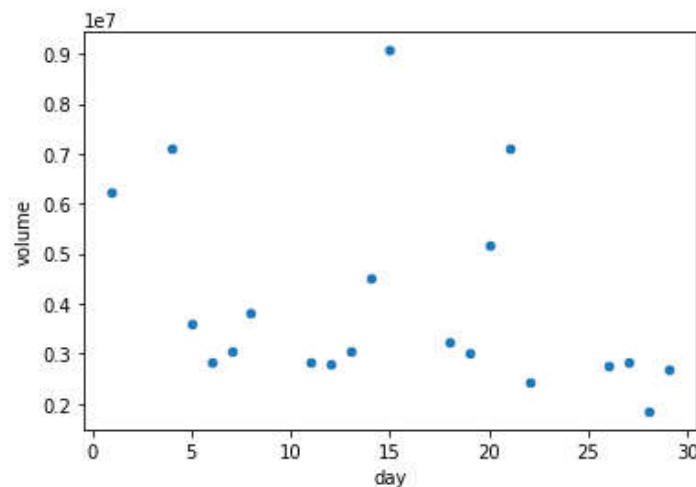
```
In [49]: volume.plot(x='day',y='volume',kind='scatter')
```

```
Out[49]: <matplotlib.axes._subplots.AxesSubplot at 0x1a786f30>
```



אם נשמור את הגרף בתוך משתנה ונפעיל את הפונקציה type נראה זאת.

```
In [52]: volume_plot=volume.plot(x='day',y='volume',kind='scatter')
```



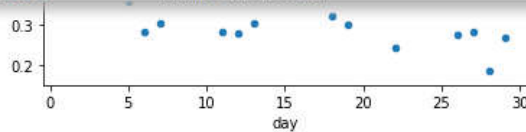
```
In [53]: type(volume_plot)
```

```
Out[53]: matplotlib.axes._subplots.AxesSubplot
```


יש עוד המון פרמטרים שניתן לתת לפונקציה הזו וכך לשחק עם הגרף עד לתצוגה האופטימלית עבורנו.

```
volume.plot(x='day',y='volume',kind='scatter')
```

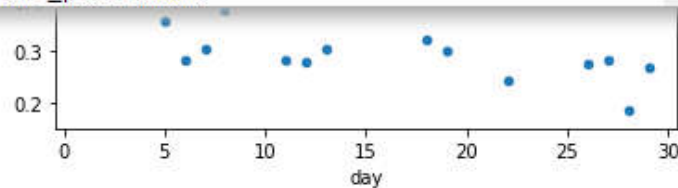
Signature: `volume.plot(self, x=None, y=None, kind='line', ax=None, subplots=False, sharex=None, sharey=False, layout=None, figsize=None, use_index=True, title=None, grid=None, legend=True, style=None, logx=False, logy=False, loglog=False, xticks=None, yticks=None, xlim=None, ylim=None, rot=None, fontsize=None, colormap=None, table=False, yerr=None, xerr=None, secondary_y=False, sort_columns=False, **kwargs)`
Call signature: `volume.plot(x=None, y=None, kind='line', ax=None, subplots=False, sharex=None, sharey=False, layout=None, figsize=None, use_index=True, title=None, grid=None, legend=True, style=None, logx=False, logy=False, loglog=False, xticks=None, yticks=None, xlim=None, ylim=None, rot=None, fontsize=None, colormap=None, table=False, yerr=None, xerr=None, secondary_y=False, sort_columns=False, **kwargs)`
Type: `FramePlotMethods`



אם שומרים את הגרף בתוך משתנה אח"כ ניתן להפעיל עוד פונקציות על אותו משתנה:

```
In [60]: volume_plot=volume.plot(x='day',y='volume',kind='scatter')
volume_plot.
```

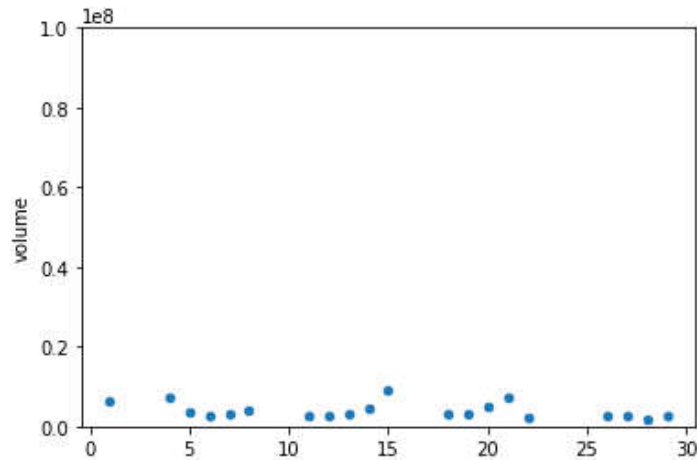
`volume_plot.acorr`
`volume_plot.add_artist`
`volume_plot.add_callback`
`volume_plot.add_collection`
`volume_plot.add_container`
`volume_plot.add_image`
`volume_plot.add_line`
`volume_plot.add_patch`
`volume_plot.add_table`
`volume_plot.aname`



לדוגמא להגביל את המידע שיופיע ב ציר ה y

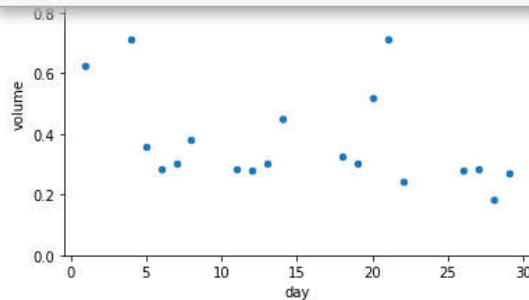

```
In [65]: volume_plot=volume.plot(x='day',y='volume',kind='scatter')
volume_plot.set_ylim(0,100000000)
```

Out[65]: (0, 100000000)



```
In [66]: volume_plot=volume.plot(x='day',y='volume',kind='scatter')
volume_plot.set_ylim(0,10000000)
```

Out[66]: **Signature:** volume_plot.set_ylim(bottom=None, top=None, emit=True, auto=False, **^ + x** ****kw**)
Docstring:
Set the data limits for the y-axis



תוספת מידע על פונקציות ופרמטרים לגרפים:

פונקציות הגרפיקה מתחלקות לשלושה סוגים,

שליטה בגרפים	יצירת גרפים	סימונים ואפיונים
figure subplot zoom hold	<u>2-D</u> plot fill plotyy	legend (2D only) text title box grid axis set, get clabel xlabel ylabel
alpha shading <u>3-D</u> view rotate3d	<u>3-D</u> plot3 surf, surfc mesh, meshz contour, contour3, contourf waterfall cylinder	<u>3-D</u> zlabel colorbar colormap

סוג קו	סוג נקודה	צבע
- solid	. point	b blue
: dotted	o circle	g green
-. dashdot	x x-mark	r red
-- dashed	+ plus	c cyan
	* star	m magenta
	s square	y yellow
	d diamond	k black
	v triangle (down)	
	^ triangle (up)	
	< triangle (left)	
	> triangle (right)	
	p pentagram	
	h hexagram	

שינוי תכונות הגרף ויצירת גרף כפול

סיננו בעבר נתונים למניה מסוימת בחודש מסוים והצגנו את volume שלה בכל יום בחודש.

שמרנו את טבלה זו במשתנה volume ובדקנו איך יראה גרף שיכיל בציר הא את הערכים שבעמודה day ויצג בציר ה y את הערכים שבעמודה volume.

עכשיו ננסה ליצור גרף שמציג בתוכו נתוני שתי עמודות גם open וגם close.

לצורך זה נוסיף לטבלה volume גם את העמודות open ו close.

הקוד לפני הוספת העמודות open ו close:

```
In [67]: df['day']=df['date'].str[0:2].astype(int)
df['year_month']=df[df['Name']=='AAL']['date'].str[3:]
volume=df[df['year_month']=='12/2017']['day','date','volume'].sort_values('day')
volume
```

```
Out[67]:
```

	day	date	volume
1213	1	01/12/2017	6229631
1214	4	04/12/2017	7113822
1215	5	05/12/2017	3594043
1216	6	06/12/2017	2835542
1217	7	07/12/2017	3046954
1218	8	08/12/2017	3817896

לאחר הוספת העמודות open ו close:

```
In [68]: df['day']=df['date'].str[0:2].astype(int)
df['year_month']=df[df['Name']=='AAL']['date'].str[3:]
volume=df[df['year_month']=='12/2017']['day','date','open','close'].sort_values('day')
volume
```

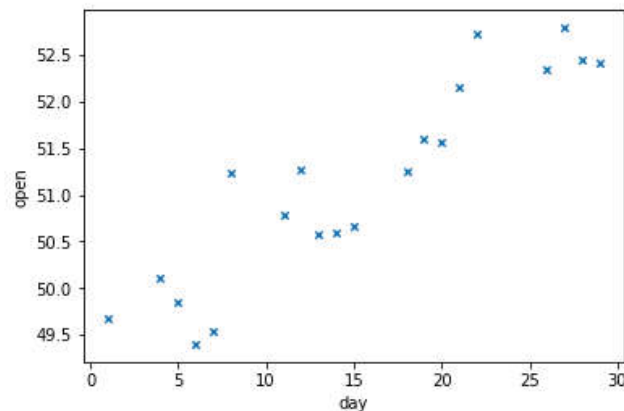
```
Out[68]:
```

	day	date	open	close
1213	1	01/12/2017	49.67	49.00
1214	4	04/12/2017	50.10	49.93
1215	5	05/12/2017	49.85	49.47
1216	6	06/12/2017	49.39	49.61
1217	7	07/12/2017	49.53	50.88
1218	8	08/12/2017	51.23	51.02
1219	11	11/12/2017	50.78	51.30

ניצור גרף מסוג scatter שיציג את ערכי הopen בסימון x ונשמור אותו במשתנה my_ax_plot.

```
In [71]: %matplotlib inline
import matplotlib.pyplot as plt

my_ax_plot=volume.plot(x='day',y='open',kind='scatter', marker='x')
```



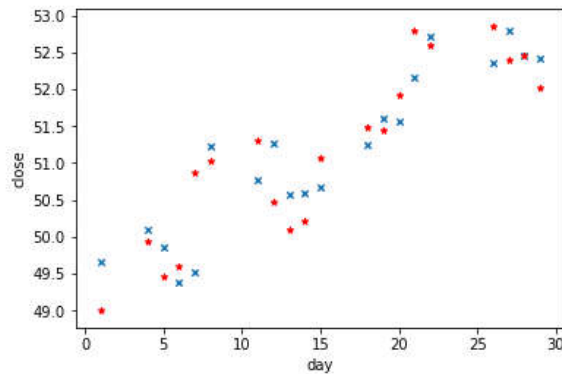
ניצור עוד גרף שיציג את עמודה close אבל יקבל כפרמטר גם את הגרף ששמרנו במשתנה my_ax_plot. בצורה כזו נוכל לראות על גרף אחד ערכים של שתי עמודות (גם open וגם close).

מה רשמנו כאן: שאנחנו רוצים גרף שציר הא הוא העמודה day, ציר ה y הוא העמודה close, סוג הגרף שאנחנו רוצים ליצור הוא מסוג scatter, צבע אדום c='r' (color=red), סימון ערכים בכוכבית ונותנים לגרף זה כפרמטר את הגרף ששמרנו במשתנה my_ax_plot.

```
In [75]: %matplotlib inline
import matplotlib.pyplot as plt

my_ax_plot=volume.plot(x='day',y='open',kind='scatter', marker='x')
volume.plot(x='day',y='close',kind='scatter',c='r',marker='*',ax=my_ax_plot)
```

Out[75]: <matplotlib.axes._subplots.AxesSubplot at 0x1af9efb0>



כדי לצייר נרות נצטרך להשתמש בגרף מסוג boxplot אבל את זה נלמד לעשות דרך ספרייה אחרת בהמשך.