Project: scraping data for ISIN and Derivatives from yourMoney website

Type ISIN : Master data table and Interest data table

Type Derivative: master table and Eispa table

Website link :

https://www.yourmoney.ch/ym/search/extended?searchTerm=**US48128GYT11**

<https://www.yourmoney.ch/ym/details/45354246,622,333#Tab0>

python script working for this one : yourMoney\_refwithEusipa.py

Code description:

This code does the following:

1. Imports some Python libraries that are needed to run the code.
2. Defines some functions that help to find and extract data from a financial website.
3. Reads a text file containing a list of financial product identifiers.
4. Uses the functions to scrape data from the website for each identifier in the text file and stores the results in a pandas DataFrame.
5. Cleans up the column names in the DataFrame.
6. Saves the DataFrame to an Excel file.

In summary, the code helps to collect data about financial products from a website and store it in a file that can be used for further analysis.

Challenges : website link , used API to create query URL. With the help of API , I got listing id and then created query URL

Added type column to get type of security that if interest table is there security is bond else security is derivative.

Sathish : 14 feb : issue : df was showing isin from website . in website for some security data was there but isin was blank. So I replaced isin column in df with identifier of portfolio.

Mouna : 16 Feb : Mouna requested date should be in dd/mm/yyyy instead of mm/dd/yyyy that was

Provided in Website

Code added :

**try**:  
 df[**'Issue Date'**] = pd.to\_datetime(df[**'Issue Date'**], format=**'%m/%d/%Y'**).dt.strftime(**'%d/%m/%Y'**)  
**except** (KeyError, ValueError):  
 df[**'Issue Date'**] = pd.NaT

learning from today’s task:

To convert a date column in a pandas DataFrame from "MMDDYYYY" format to "DDMMYYYY" format, you can use the **pd.to\_datetime()** method with the **format** parameter.

This code first uses **pd.to\_datetime()** to convert the "date\_col" column to a pandas datetime object, using the **format='%m%d%Y'** parameter to specify the input format.

The **dt.strftime('%d%m%Y')** method is then used to convert the datetime object back to a string, this time using the desired output format of "DDMMYYYY".

df['date\_col'] = pd.to\_datetime(df['date\_col'], format='%m%d%Y').dt.strftime('%d%m%Y')

The error you're seeing suggests that the input data in the "MMDDYYYY" format contains at least one instance where the date is missing or represented as a dash ("-").

You can handle this error by either removing the problematic values or replacing them with a default value. Here's an example of how you can replace the problematic values with "NaT" (not a time) using the **errors** parameter of the **pd.to\_datetime()** method:

df['date\_col'] = pd.to\_datetime(df['date\_col'], format='%m%d%Y', errors='coerce').dt.strftime('%d%m%Y')

The **errors='coerce'** parameter will set the problematic values to "NaT". The **.dt.strftime('%d%m%Y')** method is then used to convert the datetime object back to a string, this time using the desired output format of "DDMMYYYY".

By assigning the result back to the "date\_col" column, the column will now contain the dates in the desired format, with "NaT" for any problematic values.

If the column is not present in the DataFrame, you can create it using the **assign** method. Here is an example:

df = df.assign(issue\_date=pd.to\_datetime(df['Issue Date'], format='%m%d%Y', errors='coerce').dt.strftime('%d%m%Y'))

import pandas as pd

# Assume the data is in a CSV file and we want to read it into a DataFrame

df = pd.read\_csv('data.csv')

# Convert the date column to datetime format

df['date'] = pd.to\_datetime(df['date'], format='%m/%d/%Y')

# Convert the date column back to string format with the desired format

df['date'] = df['date'].dt.strftime('%d/%m/%Y')