NAME - AASHISH BUDANIA

Date of Submission - 12 Sept 2018
NITK Surathkal, Karnataka

Problem Statement -

As a benchmarking company, there are frequent requirements on converting cost related data which may be available in the different currency based on where the Client operates, to a base Currency that BlueOptima uses for benchmarking. This requires BlueOptima to record and store historical values of all Currency values and their conversion rates, going as far back as start of 2010.

I am required to collect mid-market rates for all Currency available from a reliable source and store the information in a format that can easily be used in an application to convert cost from one currency to another.

I am only expected to identify and store mid-market day rates.

The application should be designed and implemented to:

1. Collect historical values and store it
2. Collect values day on day and update the data store with new values

As deliverables following should be provided at the end of the work sample:

1. Source code of a working solutions
2. Document explaining setup, build and execution steps
3. Document covering the reasons for source being used
4. Brief explanation of implementation and control flow
5. Details of data structure design and benefits of the design
6. How to use data to convert cost from one currency to another

SOLUTION -

Goals of the Application -

1. The name of the application can be given as *CurrencyConverter* as it is used to convert the currency from one country's currency to another country's currency on a daily basis.
2. It is also used to store the values of currency change and their increment and decrement in the currencies in comparison to each other over a time period.
3. It stores the values in json format to have more availability of the data for frequent use whenever the data is required.
4. Values to be extracted on daily basis are also appended to the existing results.
5. Simple structure of storage and minimum data redundancy is achieved during implementation.

1. SOURCE CODE -

Source code is sent as an attachment in this mail, please find the python file named ***main.py*** and to get the maximum available currencies ***all-codes.csv*** is also attached that has 436 currencies. For few of the countries it's not available and those countries are discarded from the list while implementation.

Some of the used libraries -

pandas - to deal with csv file and to select a specific column as country_code
lxml - to process the html page content in better way to scrap the data
requests - to make http request using URL of webpage
pickle - to store the historical data

After execution of main.py for a time period (between user defined start_date and end_date) the historical data is written to historical.p file in the json format as shown below -

```
payload={
        "INR":{
                "base" : "USD",
                "historicalData":{
                        "2017-10-10" : "65.249001",
                }
        },
        "AED":{
                "base" : "USD",
                "historicalData":{
                        "2017-10-10" : "3.672501",
                }
        }
}
```

Base currency on which BlueOptima operates is considered as USD (US Dollar), although it can be selected manually based on the requirement or user's choice.

Once the data is dumped into the pickle few lines of the code can be commented those are the part of writing the data between range over a period of time, because once the data is written to the file we don't need to extract the same data again, it's time consuming and worthless too.

These lines are -

```
payload={} #global access
startDate = datetime.date(2010, 9, 12)
endDate = datetime.date(2018, 9, 12)
#the above two lines are telling a period of last 8 years
getDate(startDate,endDate)
pickle.dump( payload, open("historicalData.p", "wb" ) )
```

2. EXECUTION STEPS -

1. All the files should share the same folder or path should be valid for each file in case of these are not maintained in a single folder.
2. To access all the libraries, they should be installed in the system.
   -To install using python2.7 try -> pip install pkg-name
   -To install using python3.5 or python3.6 try -> pip3 install pkg-name
3. For execution use -
   $ python3 main.py (in case of python3.5 or python3.6)
   else
   $python main.py
4. To see the output or historical data you can use payload that is global dictionary or all the data will be stored to historical.p and it can be loaded to payload and displayed.

3. SOURCE USED -

1. *http://fxtop.com/*
2. *https://github.com/datasets/currency-codes/blob/master/data/codes-all.csv*

After a lot of research this website was selected to apply currency converter tool.

Reasons for its selection -

1. found most active and effective as it has historical datasets
2. wide range of currency data over the years
3. Supports more than 400 country's currency codes
4. Open source data (No pricing at all)
5. Ip doesn't get blocked while scraping :P.
6. These sources can be trusted for as providers of correct information.
7. It considers multiple currencies for a single country if available for example for Bhutan it's BTN as well as INR(Indian Rupees) is also supported.

4. CONTROL FLOW and EXECUTION -

The python script named as *main.py* takes only two parameter starting and ending date of your choice and provides you the historical data for all the country code on the base currency on which blueOptima operates as USD (US Dollar, it can be changed manually as well).

The data is dumped in json format using pickle.

It can be loaded as -

payload=pickle.load(open("historicalData.p", "rb")) ***#loads data into payload***
payload["AUD"]["historical"]["2018-09-11"] ***#tells the AUD value in USD on 2018-09-11***

Important -> format for date is 2018-09-11 instead of 2018-9-11

5. DATA STRUCTURES and BENEFITS -

1. To store the list of country codes python list is used, that’s same as an array.
2. To store the historical data pickle is used in python for better storage.
3. Historical data is stored in json format for more availability and for quick response.
4. Data structures, those are easily accessed and handled with python are given priority in order to implement this problem statement.

6. CURRENCY CONVERTER -

It takes two parameter as source and target currency code and amount to be converted from one currency to another, and to append the conversion result to historical data.

```
#payload=pickle.load(open( "historicalData.p", "rb" ) )
currencyConverter('AUD, 'INR', 267643)
#print(payload)
```

The above lines of code convert 267643 australian dollars to indian rupees.

Thank You