IMF API Data Structure

Prepared for Turret Labs

October 18, 2016

This example gives additional guidance on how the IMF API is structured and uses the API to plot Brazilian exports over the last ten years. To make a chart entirely from API data and text, a second URL is needed to get the title, units, footer, and frequency. It may be preferrable to hard key (or reference to a table) every component of the chart except for the data, but this example shows how each text component of a chart can be obtained from the API, in a programmatic way.

More information on the IMF Data API The IMF offers guidance on using their data services. The API is not stable, so check the IMF data services news if something that was previously working starts to give an error messages.

URL summary Data and reference dates only (for example: August 2016, 16989.08):
`http://dataservices.imf.org/REST/SDMX_JSON.svc/CompactData/IFS/M.BR.TXG_FOB_USD`
Data structure and everything else (large dataset) for all IFS series:
`http://dataservices.imf.org/REST/SDMX_JSON.svc/DataStructure/IFS`

1 Retrieving data and text for the chart

The URL of the data (first URL in the summary above) has six parts:

1. http://dataservices.imf.org/REST/SDMX_JSON.svc/: the IMF JSON RESTful Web Service;
2. CompactData: the **method**; this will be changed to DataStructure to get the title, units, etc;
3. IFS: the **series name**, which is used to find the title, units, etc;
4. M: the **frequency**; there are only a few options: Annual (A), biannual (B) quarterly (Q), monthly (M), weekly (W), or daily (D);
5. BR: The **country code**; how to get the full country name from the code is included below; and
6. TXG_FOB_USD: the **indicator** tells which dataset to retrieve and is used to get the title, units, etc.

In this example with Python, data are first collected from the data URL (CompactData method), then the DataStructure **method** retrieves text for the chart title and labels.

```
In [1]: # Import libraries
        import requests
        import pandas as pd

        # Define the six parts above for our request.
        # This is just a decomposition of the URL
        webserv = 'http://dataservices.imf.org/REST/SDMX_JSON.svc/'
        method1 = 'CompactData/'
        method2 = 'DataStructure/'
        series = 'IFS'
        frequency = 'M'
        country = 'BR'
        indicator = 'TXG_FOB_USD'
```

```
In [2]: # The URL for data, put back together, is therefore:
        URL_data = webserv + method1 + series + '/' + frequency +'.' + country +'.' + indicator

        # This line gets data from the above URL using the requests package
        data = requests.get(URL_data).json()

        # Load data into a pandas dataframe, called 'brx' for Brazillian exports.
        # The parts in square brackets are the navigation within the JSON data
        # to the data of interest.
        brx = pd.DataFrame(data['CompactData']['DataSet']['Series']['Obs'])

        # Show the last five observations
        brx.tail()
```

```
Out[2]:         @OBS_VALUE @TIME_PERIOD
        627  15374.374167      2016-04
        628   17571.14694      2016-05
        629  16743.258811      2016-06
        630  16330.547668      2016-07
        631  16989.086582      2016-08
```

1.1 Retrieving chart text using the DataStructure method

The key to obtaining information about the data itself, which can be used for chart titles, is to use the IMF data services DataStructure **method**. Navigating the new URL JSON results gives the series and indicator names, units, frequency, and country names, which are used as string fields for our chart. The critical new URL is:
http://dataservices.imf.org/REST/SDMX_JSON.svc/DataStructure/IFS

The data in the second URL is defined as data_2 below.

Series name as the chart source First, retrieve the full name of the series to footnote the chart source.

```
In [3]: # the URL for DataStructure method is simply:
        URL_seriesinfo = webserv + method2 + series

        # Get data from the new URL and call it data_2
        data_2 = requests.get(URL_seriesinfo).json()

        # Save series name -- again, square brackets are
        #  navigation within the JSON data
        series_name = str(data_2['Structure']['Concepts']['ConceptScheme']['Name']['#text'])
        chart_source = 'Source: International Monetary Fund, ' + series_name
        print chart_source
```

```
Source: International Monetary Fund, International Financial Statistics (IFS)
```

Country name Next, obtain the text string for the full country name using the country ID from the original data request, ‘BR’:

```
In [4]: # Country information is here:
        ifs_countries = pd.DataFrame(data_2['Structure']['CodeLists']['CodeList'][2]['Code'])

        # Keep only Brazil, which is 'country'
        ifs_countries = ifs_countries[ifs_countries['@value'] == country]
        idx = ifs_countries.index.values[0]

        # Country name string
        chart_country = str(ifs_countries['Description'][idx]['#text'])
        print chart_country
```

```
Brazil
```

Frequency The same process as above:

```
In [5]: # Frequency does not vary by dataset, and is found here:
        ifs_freq = pd.DataFrame(data_2['Structure']['CodeLists']['CodeList'][1]['Code'])

        ifs_freq = ifs_freq[ifs_freq['@value'] == frequency]
        idx = ifs_freq.index.values[0]

        # Country name string
        chart_frequency = str(ifs_freq['Description'][idx]['#text'])
        print chart_frequency
```

```
Monthly
```

Indicator This is the important one for the chart title. It gives the indicator name and unit type, and follows the same process as before:

```
In [6]: # Same DataStructure URL, but indicator information is here:
        ifs_indicator = pd.DataFrame(data_2['Structure']['CodeLists']['CodeList'][3]['Code'])

        ifs_indicator = ifs_indicator[ifs_indicator['@value'] == indicator]
        idx = ifs_indicator.index.values[0]

        # Chart unit multiplier string
        chart_indicator = str(ifs_indicator['Description'][idx]['#text'])
        print chart_indicator
```

```
Goods, Value of Exports, Free on board (FOB), US Dollars
```

Unit multiplier Last one is the unit multiplier number, which is actually shown in the original data URL. The number corresponds to the number of zeros to add to obtain the actual value. So 6 is millions, 9 is billions, 3 is thousands, etc. Here is how to get the number:

```
In [7]: unit_multiplier = data['CompactData']['DataSet']['Series']['@UNIT_MULT']
        print unit_multiplier
```

```
6
```

For completeness, here is how to verify the unit_multiplier and include the its text string in the chart:

```
In [8]: # DataStructure method URL
        ifs_unit_mult = pd.DataFrame(data_2['Structure']['CodeLists']['CodeList'][0]['Code'])

        ifs_unit_mult = ifs_unit_mult[ifs_unit_mult['@value'] == unit_multiplier]
        idx = ifs_unit_mult.index.values[0]

        # Chart title string
        chart_unit_mult = str(ifs_unit_mult['Description'][idx]['#text'])
        print chart_unit_mult
```

```
Millions
```

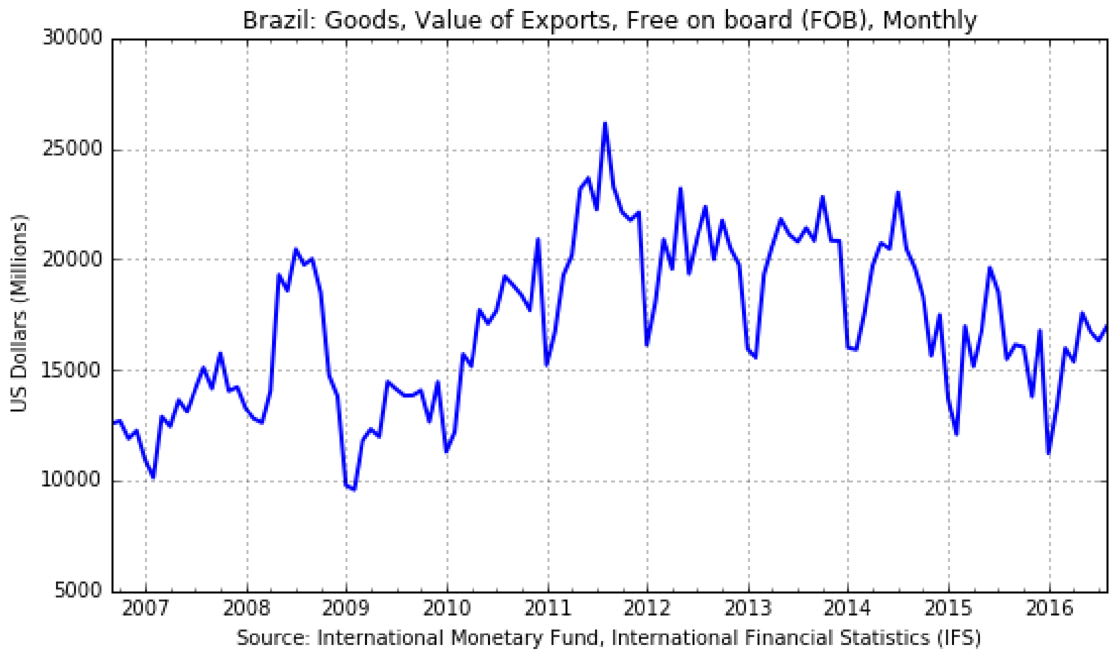
2 Creating the chart

There are many options for arranging the chart components. One example is the following:

```
In [9]: # Clean JSON data
        brx['@OBS_VALUE'] = brx['@OBS_VALUE'].astype(float)
        rng = pd.date_range(pd.to_datetime(brx['@TIME_PERIOD'][0]),
                            periods=len(brx.index), freq='M')
        brx = brx.set_index(pd.DatetimeIndex(rng))
        brx = brx[512:]
```

```
In [10]: # Packages for creating plots in python
         import matplotlib as mpl
         import matplotlib.pyplot as plt
         %matplotlib inline

         # Plot exports
         brx.plot(grid=True, figsize=(9, 5), color="blue",
                  linewidth=2, legend=False)
         plt.xlabel(chart_source)
         plt.ylabel(chart_indicator[45:] + " (" + chart_unit_mult + ")")
         plt.title(chart_country + ": " + chart_indicator[:44] +
                   ", " + chart_frequency);
```


Brazil: Goods, Value of Exports, Free on board (FOB), Monthly
US Dollars (Millions)
Source: International Monetary Fund, International Financial Statistics (IFS)