

**Anya
Prosvetova**
@Anyalitica



Tableau DataDev
Ambassador

-

Tableau & Alteryx
Consultant

Tableau Prep

How to connect to APIs using the Script step and Python

What is an **API**?



Client application

API

Server

Data formats when working with APIs

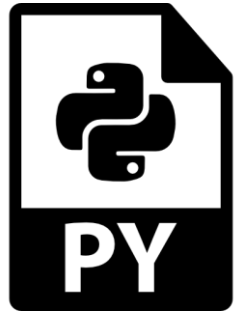
XML – Extensible markup language

JSON – JavaScript Object Notation

The example of the response in JSON format on the right shows that the data is stored in **name and value pairs**.

```
{
  "orders":
  [
    {
      "orderno": "748745375",
      "date": "Jun 30, 2008 1:54:23 AM",
      "trackingno": "TN0039291",
      "customer" :
      {
        "custid": "11045",
        "fname" : "Sue",
        "lname" : "Hatfield",
        "address" : "1409 Silver Street",
        "city" : "Ashland",
        "state" : "NE",
        "zip" : "68003"
      }
    }
  ]
}
```

Connecting to APIs in Tableau Prep



Python script



Script step in your Tableau Prep flow

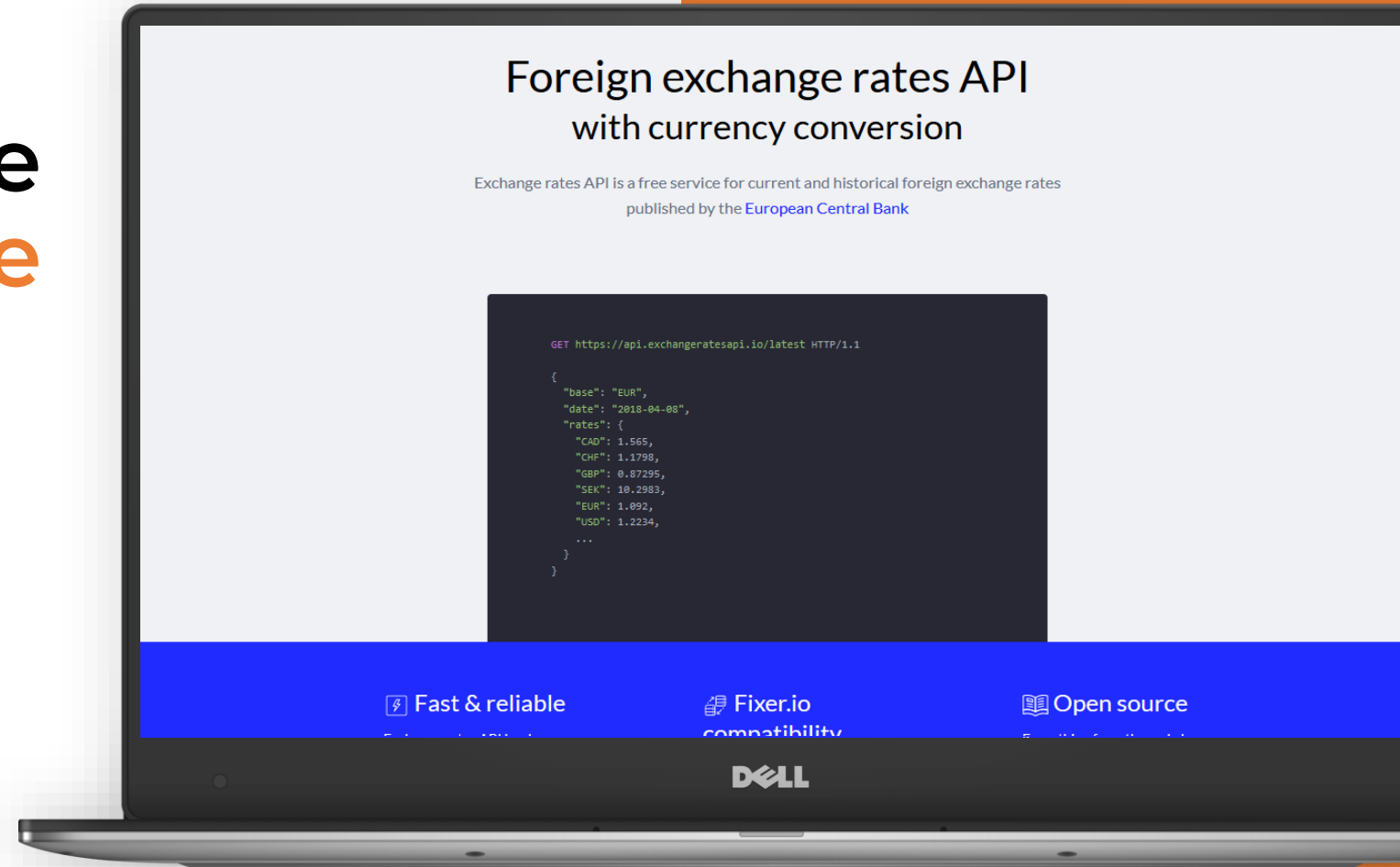


TabPy server



Connecting to the Foreign Exchange rates API in Tableau Prep

exchangeratesapi.io



Step 1. Read the [API's documentation](#)

Step 2. Get the [URL](#)

https://api.exchangeratesapi.io/history?start_at=2020-10-01&end_at=2020-10-31&symbols=GBP

Step 3. Write your [Python script](#)

<https://github.com/anyalitica/api-connect-tableau-prep>

Step 4. Install and launch [TabPy server](#)

Step 5. Connect [Tableau Prep & TabPy server](#)

Step 6. Add the [Script step](#) in your Tableau Prep flow

Step 1. Read the **API's documentation**

Foreign exchange rates API with currency conversion

Exchange rates API is a free service for current and historical foreign exchange rates published by the [European Central Bank](#)

```
GET https://api.exchangeratesapi.io/latest HTTP/1.1
```

```
{
  "base": "EUR",
  "date": "2018-04-08",
  "rates": {
    "CAD": 1.565,
    "CHF": 1.1798,
    "GBP": 0.87295,
    "SEK": 10.2983,
    "EUR": 1.092,
    "USD": 1.2234,
    ...
  }
}
```

Fast & reliable

Exchange rates API has been designed and tested to handle thousands of request per second. Also constantly monitored.

Fixer.io compatibility

Built in Fixer.io compatibility so you can keep all the libraries you already like and use daily.

Open source

Everything from the code base, homepage to the deployment process is opensource and free to use under a permissive MIT license.

Step 2. Get the URL

https://api.exchangeratesapi.io/history?start_at=2020-10-01&end_at=2020-10-31&symbols=GBP

Request only historical exchange rates for **GBP** for the period between **1 October** and **31 October 2020**.

The default base rate is **Euro**.

Request specific exchange rates by setting the symbols parameter.

```
GET https://api.exchangeratesapi.io/latest?symbols=USD,GBP HTTP/1.1
```

Get historical rates for a time period.

```
GET https://api.exchangeratesapi.io/history?start_at=2018-01-01&end_at=2018-09-01
```


Step 3. Write your Python script

```
1 import requests
2 import pandas as pd
3
4 # create function to get data from the API
5 def get_exchange_rates(input):
6     # connect to the API to get exchange rates for GBP during October 2020. The base currency is EUR.
7     response = requests.get("https://api.exchangeratesapi.io/history?start_at=2020-10-01&end_at=2020-10-31&base=EUR&currencies=GBP")
8     # create new variable 'rates' for the response from the API
9     rates = response.json()
10    # create a Pandas dataframe from the 'rates' level of the response
11    df=pd.DataFrame(rates['rates'])
12    # create a new variable 'rates_table' to hold the dataframe and transpose the table
13    rates_table = df.T
14    # rename the index column to be called 'Date'
15    rates_table=rates_table.rename_axis('Date')
16    # turn index to a column
17    rates_table.reset_index(inplace=True)
18    # rename the 'GBP' column to be called 'EURGBP'
19    rates_table = rates_table.rename(columns={'GBP':'EURGBP'})
20    # change data types
21    rates_table['EURGBP'] = rates_table['EURGBP'].astype('float')
22    rates_table['Date'] = rates_table['Date'].astype('str')
23    return rates_table
24
25 # define the columns and their data types that are brought into Tableau Prep
26 def get_output_schema():
27     return pd.DataFrame({
28         'Date' : prep_string(),
29         'EURGBP' : prep_decimal()
30     })
```

Data type in Tableau Prep Builder	Data type in Python
String	Standard UTF-8 string
Decimal	Double

Function in Python	Resulting data type
prep_string ()	String
prep_decimal ()	Decimal
prep_int ()	Integer
prep_bool ()	Boolean
prep_date ()	Date
prep_datetime ()	DateTime

Step 4. Install and launch TabPy server

a) Install TabPy using the Command Line

```
pip install tabpy
```

b) Start TabPy using the Command Line

```
tabpy
```

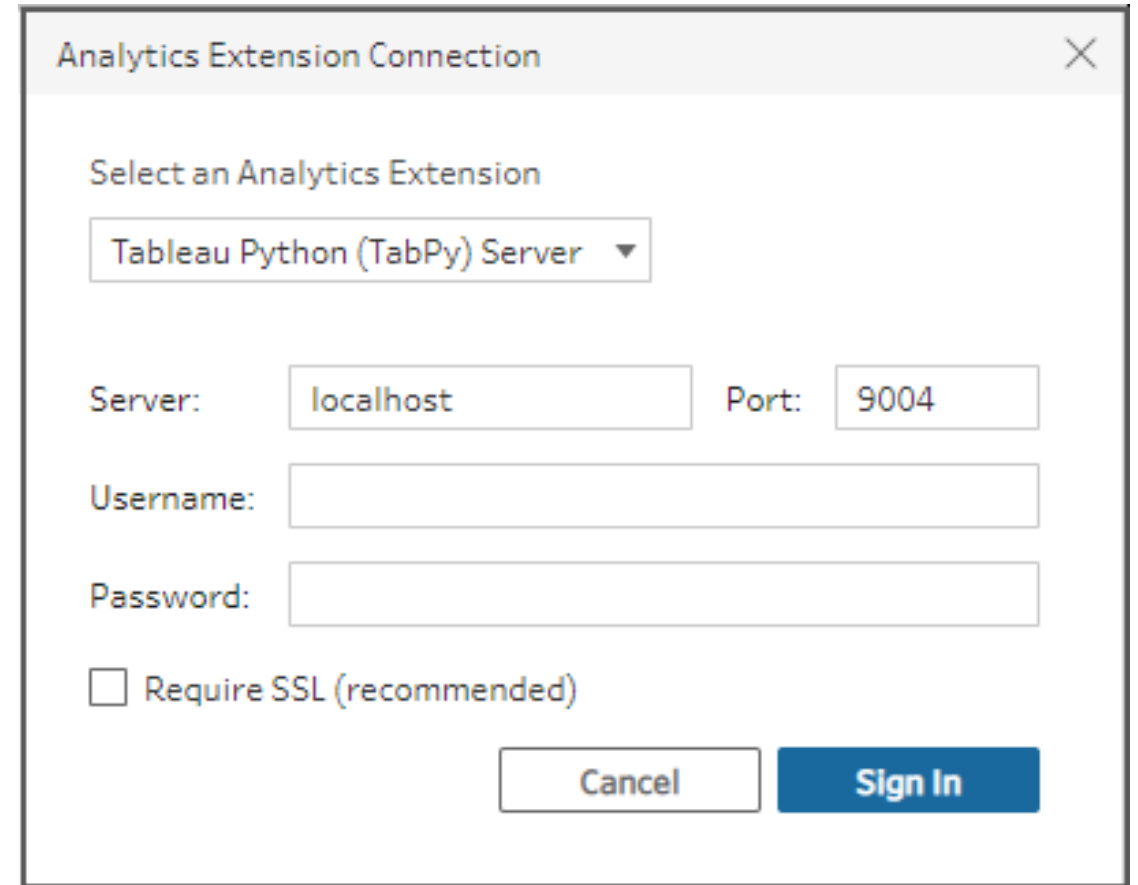
```
Anaconda Prompt (Anaconda3) - tabpy
(base) C:\Users\Anna Prosvetova>tabpy
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter port set to "9004" from default
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter server_version set to "0.8.9"
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter evaluate_timeout set to "30" f
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter upload_dir set to "c:\users\an
ackages\tabpy\tmp\query_objects" from default value
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter transfer_protocol set to "http
2020-10-27,12:44:16 [DEBUG] (app.py:app:221): Parameter certificate_file is not set
2020-10-27,12:44:16 [DEBUG] (app.py:app:221): Parameter key_file is not set
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter state_file_path set to "c:\use
ite-packages\tabpy\tabby_server" from default value
2020-10-27,12:44:16 [INFO] (app.py:app:280): Loading state from state file c:\users\an
ackages\tabpy\tabby_server\state.ini
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter static_path set to "./" from d
2020-10-27,12:44:16 [DEBUG] (app.py:app:292): Static pages folder set to "C:\Users\Ann
2020-10-27,12:44:16 [DEBUG] (app.py:app:221): Parameter TABPY_PWD_FILE is not set
2020-10-27,12:44:16 [INFO] (app.py:app:311): Password file is not specified: Authentic
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter log_request_context set to "fa
2020-10-27,12:44:16 [INFO] (app.py:app:327): Call context logging is disabled
2020-10-27,12:44:16 [DEBUG] (app.py:app:215): Parameter max_request_size_in_mb set to
2020-10-27,12:44:16 [INFO] (app.py:app:110): Initializing TabPy...
2020-10-27,12:44:16 [DEBUG] (selector_events.py:selector_events:53): Using selector: S
2020-10-27,12:44:16 [INFO] (callbacks.py:callbacks:42): Initializing TabPy Server...
2020-10-27,12:44:16 [DEBUG] (state.py:state:147): Collected endpoints: {}
2020-10-27,12:44:16 [INFO] (app.py:app:113): Done initializing TabPy.
2020-10-27,12:44:16 [INFO] (app.py:app:67): Setting max request size to 104857600 byte
2020-10-27,12:44:16 [INFO] (callbacks.py:callbacks:62): Initializing models...
2020-10-27,12:44:16 [DEBUG] (state.py:state:147): Collected endpoints: {}
2020-10-27,12:44:16 [INFO] (app.py:app:93): Web service listening on port 9004
```

Step 5. Connect Tableau Prep & TabPy server

Help > Settings and Performance >

Manage Analytics Extension

Connection



Analytics Extension Connection

Select an Analytics Extension

Tableau Python (TabPy) Server ▼

Server: localhost Port: 9004

Username:

Password:

Require SSL (recommended)

Cancel Sign In

Step 6. Add the **Script step** in your Tableau Prep flow

The screenshot displays the Tableau Prep interface. At the top, a flow diagram shows an 'Input' step connected to a 'Python Script' step. Below this, the 'Python Script' step is selected, showing its configuration and output.

Python Script 2 fields 18 rows

Filter Values... Create Calculated Field...

Settings Changes (1)

Connection type

- Rserve
- Tableau Python (TabPy) Server

Server
Connection to localhost: 9004

File Name
exchange-rates-api.py

Function Name
get_exchange_rates

Include the schema function "get_output_schema" in your script to define the fields and data types that are returned. [Learn more](#)

Date 18	EURGBP 18
01/10/2020	0.90273
02/10/2020	0.90395
05/10/2020	0.90535
06/10/2020	0.90588
07/10/2020	0.90591
08/10/2020	0.90598
09/10/2020	0.90673
12/10/2020	0.90675
13/10/2020	0.90723
14/10/2020	0.90754
15/10/2020	0.90755
16/10/2020	0.9081

Learn more about integrating Tableau Prep & Python



Read about using Python scripts in Tableau Prep flow

https://help.tableau.com/current/prep/en-us/prep_scripts.htm



Read TabPy documentation

github.com/tableau/TabPy



Practice with public APIs

github.com/public-apis/public-apis



Watch Jack Perry's video on TabPy & Tableau Prep Conductor

youtube.com/watch?v=QGj-xXgO8Qs

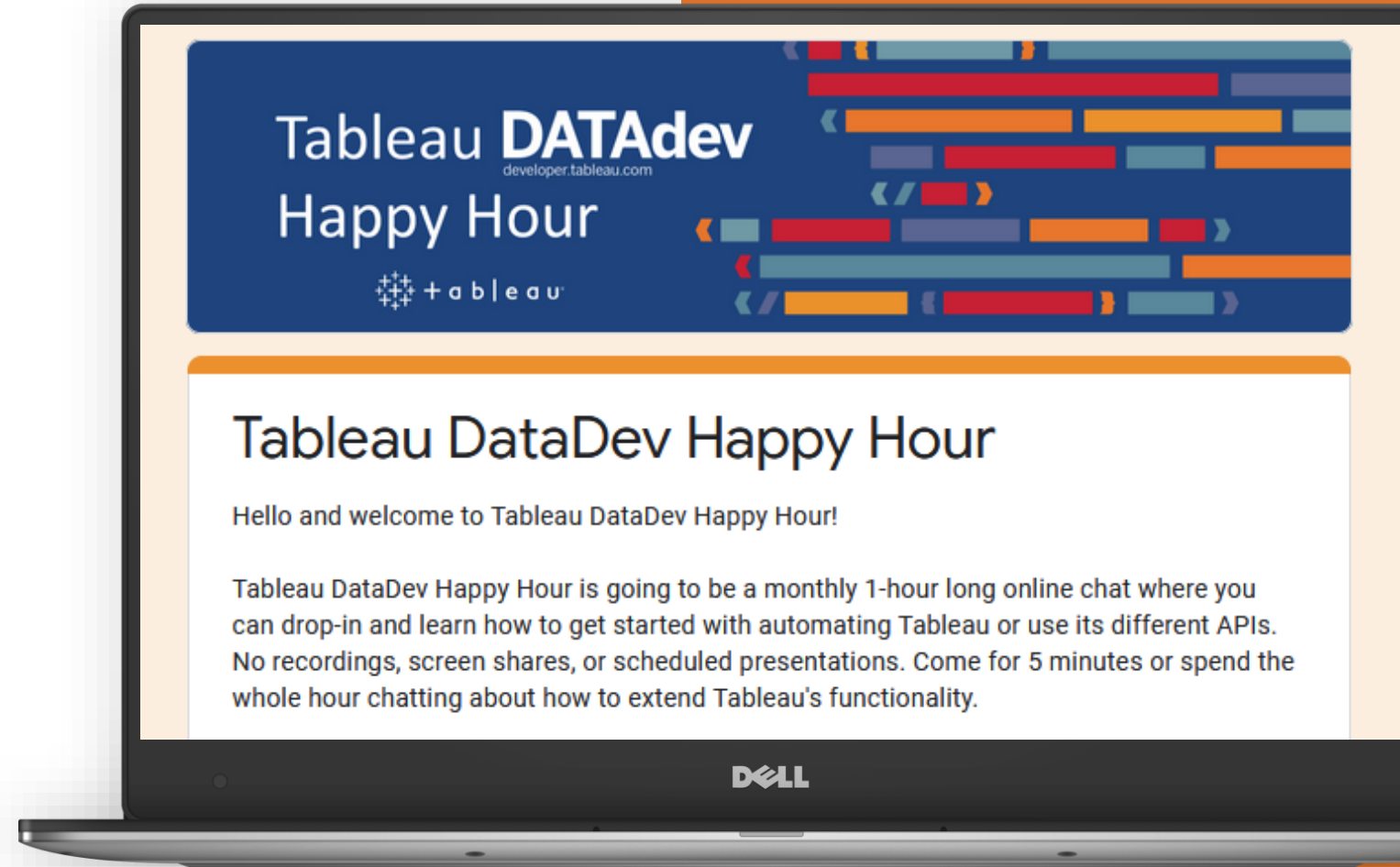


Join Tableau Developer Program

www.tableau.com/developer

Join me at Tableau DataDev Happy Hour!

bit.ly/TableauDataDevHappyHour



Questions?

GET IN TOUCH!

 @Anyalitica

 prosvetova.com

Thank you!

