Real-time temperature chart with Plotly

Introduction to IoT and Cloud Architectures

GITHUB.COM/CLAUDIUGROZA/MSA

Contents

1 Live temperature chart 5

1.1 Problem statement 5

1.2 Reading the temperature sensor 5

1.3 Posting temperature data to Plotly 5

1.3.1 API registration 5

1.3.2 Plotly SDK 6

1.3.3 Script walkthrough 6

2 Assignments 9

3 Bibliography 11

3.1 References 11

3.2 Image credits 11

1. Live temperature chart

1.1 Problem statement

Let us begin by defining some basic user requirements. We are interested in collecting temperature data for a certain area and then storing that information to a cloud service. Moreover, it comes in handy to visualise the temperature using some time series representations. The workload can be divided into two phases:

1. reading the temperature value from a local sensor
2. posting the temperature value to a chart API

1.2 Reading the temperature sensor

Due to some logistical reasons, we are not able to provide all the required components for building the circuit and implement temperature acquisition step. Alternatively, we use the *sensor.py* stub script that will return an arbitrary value.

1.3 Posting temperature data to Plotly

The cloud service we'll use to upload the temperature values and analyse them using graph representations is named **Plotly** [Plo].

1.3.1 API registration

In order to have access to Plotly's API, you need to create an account. Once you've finished this step, then navigate to user settings page. The following values need to be extracted and copied to *credentials.json* configuration file:

- the username
- generated API key

- generated streaming API(s) tokens

1.3.2 Plotly SDK

To reduce the time you need to actually post data to Plotly's web service, we'll be using the official supported SDK [Api]. Let's make sure that the package is installed on our board:

```
$ sudo pip install chart_studio
```

1.3.3 Script walkthrough

This section will provide additional comments with some regards to *main.py* script.
We make sure to import all the packages we need in our current implementation:

```
import temperature #temperature stub
import chart_studio.plotly as plot #plotly API
import json #extract credentials
import time #delay
import datetime #temp-time chart association
```

The next step is to extract authentication details by using the *credentials.json* file we previously modified:

```
with open('./credentials.json') as credentials_file:
    plotly_user_config = json.load(credentials_file)
```

Now we need to define an object that enables us to interact with Plotly API:

```
plot.sign_in(plotly_user_config["plotly_username"],
             plotly_user_config["plotly_api_key"])
```

The plot object is configured based on user preferences and set up with the streaming token from authentication details:

```
url = plot.plot([
    {
        'x': [],
        'y': [],
        'type': 'scatter',
        'mode':'lines+markers',
        'stream': {
            'token': plotly_user_config['plotly_streaming_tokens'][0],
            'maxpoints': 200
        }
    }], filename='MS-IOT Temp')
print ("View your streaming graph here: ", url)
```

Before posting any data, we need to define a stream object based on the previous plot instance. Don't forget to call *open()* function before starting to stream temperature values:

```
stream = plot.Stream(plotly_user_config['plotly_streaming_tokens'][0])
stream.open()
```

Finally, we can execute a temperature reading and then posting it:

```
temp = temperature.read_temp()
stream.write({'x': datetime.datetime.now(), 'y': temp})
```

2. Assignments

1. Extend the script to execute a temperature reading only when a button was pressed.

2. Modify the script such that line color is green and line width is set to 3 points.

3. Explore Plotly's capabilities for exporting streamed data and then download data into a .csv file format.

3. Bibliography

3.1 References

[Api] *Plotly API documentation*. https://plot.ly/api/. [Online; accessed March-2018]. 2018 (cited on page 6).

[Plo] *Plotly website*. https://plot.ly/. [Online; accessed March-2018]. 2018 (cited on page 5).

3.2 Image credits

- First page illustration.
  http://www.northeastern.edu/levelblog/2018/01/25/guide-iot-careers
- Chapter header background.
  https://blogs.microsoft.com/iot/2015/03/17/simplifying-iot/