Understanding the Starter Code

Optional Internet of Things Presentation

fb_analyze.py

This section will cover the starter code that is provided for in fb_analyze.py

Get your environment set up

- This script uses the Facebook API and the Pattern API for Python.
  - Pattern will only run on Python 2.7, so you'll need to install that for Pattern to work.

Installing Python 2.7 on Windows

You can install Python 2.7 from the following link: https://www.python.org/downloads/release/python-2710/

- Download “Windows x86 MSI Installer” and run it
- Open “Command Prompt” and type cd “C:/Python27” to verify that it worked
- Set the path so that you can install the other packages
  - In “Command Prompt” type **path=%path%;C:/Python27**

[Optional] Install a Text Editor

My personal favorite is the Sublime Text 2

- You can download it at https://www.sublimetext.com/2

Other options are:

- Vim (for crazy people): http://www.vim.org/download.php#pc
- Atom (for hipsters): https://atom.io/
- Python IDLE (comes installed with Python 2.7)

Running Programs in Sublime

There are two ways to run programs in Sublime:

- In a python file, type Ctrl+B
- In “Command Prompt” navigate to the directory of the file and type “python *file_name.py*”

Pattern Overview

Pattern is a powerful API that does Natural Language Processing.

- Available in English, Spanish, German, French, Italian, and Dutch (Sorry, no Portuguese)
- Sentiment Analysis for English, French, and Dutch
- Identifies parts of speech, conjugates verbs, etc.

Pattern also has modules for gathering internet data

Installing Pattern

If you have pip…

```
sudo pip install pattern
```

Otherwise…

1. Go to http://www.clips.ua.ac.be/pages/pattern
2. Download the zip and extract all the files
3. Navigate to the directory in “Command Prompt”, and run python setup.py install

How to do Sentiment Analysis with Pattern

Sentiment analysis using Pattern can be done easily, with minimal code. The follow two lines are all that's necessary to analyze the sentiment of a sentence:

```
from pattern.en import sentiment

sentiment("I love using the Pattern API!")
```

This will return a tuple with the first value representing the polarity (how positive or negative) and the second will be subjectivity

Installing the Facebook API

Download the Facebook API using pip:

```
sudo pip install facebook-sdk
```

Or go to through the repo

1. Go to: https://github.com/pythonforfacebook/facebook-sdk
2. Download zip, and extract all files
3. Navigate to the download directory and run in Command Prompt:

```
python setup.py install
```

Facebook API Tutorial

There are three methods:

- `graph = facebook.GraphAPI(access_token=your_access_token)`
  - This will return a GraphAPI object from which you can use the other methods.
- `graph.get_connections(id=some_id, connection_name=type)`
  - This takes an id (which we'll cover later) and a connection type, which can be posts, likes, photos, etc.
  - It will return a dictionary with two keys: paging (which you don't need) and data (which is a list of the information you want)
- `graph.get_object(id=some_id)`
  - Takes in an id
  - Returns a dictionary of information that about the id you requested

Walking through fb_analyze.py

First, we need to initialize the Facebook API using our access token

```
token = “your_access_token”

graph = facebook.GraphAPI(access_token=token)
```

Getting posts

Now, we want to get the posts for the Id that we chose. There's a method in the facebook API that will do that for us.

```
graph.get_connections(id="id", connection_name="posts")
```

The id can be the id of anything: a person, a page, or even a post.

The connection_name can be things like "posts", "photos", "likes", etc...

Getting posts, part 2

get_connections() returns a dictionary with two keys.

- “paging” - this contains information on how to get the next set of information
- “data” - this contains the data requested

For this reason, we want to extract the data key.

```
post_data = get_connections(id=”id”, connection_name=”posts”)[“data”]
```

Getting the index using sys.argv

Now, we want to extract a certain post at a certain index. I chose to use system variables to get the index so that the Galileo could ask for different posts via the arduino sketch.

You can access the system variables with:

```
sys.argv[some_index]
```

The first one is always the name of the file. The second one is the variable that you pass in.

Getting the index using sys.argv

Since we're passing the index as the first system variable, we can get it through

```
index = int(sys.argv[1])
```

Note that we need to cast it to an int because the system variables are all strings.

Checking to see that something was passed

What happens if we don’t include a system variable for the index? We can check to see if it’s included and specify a default value using the ternary operator. The ternary operator has the syntax

variable = value if (conditional) else (default value)

For the system variable example:

```
index = int(sys.argv[1]) if len(sys.argv) > 1 else 0
```

Checking for content

Posts in the Facebook API are dictionaries with the following keys:

- “id” - this is the id of the post
- “created_time” - this is the time the post was created
- “message” - this is the status update
- “story” - this is the event associated with the post, for example, changing your profile picture is a story.

Not every post has a message and not every one has a story

Checking for content

We want to cycle through the posts in the event that the post we selected has no message.

```
while “message” not in post_data[index]:
```

This will cycle through the posts, but we need to update the post that’s being checked. To do this, we update index.

```
    index += 1

    index = index % len(post_data)
```

Get the status

Now we have a dictionary that represents a post that we know has a message. To get the text, we have to extract the message key.

```
status = post_data[index]["message"]
```

Analyze the sentiment

Now for the easy part, analyzing the sentiment.

```
sent = sentiment(status)
```

Remember that sentiment returns a tuple. We want the first value because we're concerned with the polarity of the post. So we'll get the 0 index of that return value.

```
sent = sentiment(status)[0]
```

We'll write this result to a text file, which the Galileo will use.

SentimentSweep.ino

The starter code for SentimentSweep.ino contains a method called readSentiment() that will be covered in this section.

What does readSentiment() do?

This function serves to obtain the result from fb_analyze.py and return a float that represents the sentiment value.

Call fb_analyze.py

First we must convert the index integer to a string that fb_analyze.py can use

```
String indexStr = String(index);
```

We can now execute fb_analyze.py and open the file it creates, called sentiment.txt

```
system("python /media/mmcblk0p1/fb_analyze.py " + indexStr);
File sentimentText = SD.read("/media/mmcblk0p1/sentiment.txt');
```

Call fb_analyze.py

First we must convert the index integer to a string that fb_analyze.py can use

```
String indexStr = String(index);
```

We can now execute fb_analyze.py and open the file it creates, called sentiment.txt

```
system("python /media/mmcblk0p1/fb_analyze.py " + indexStr);
File sentimentText = SD.read("/media/mmcblk0p1/sentiment.txt');
```

Convert sentimentText to a string

Now we want to initialize an empty string that we will use to hold our sentimentText infomation.

```
String sentimentString = “”
```

We then add each character from the file to our string:

```
while(sentimentText.available()){

    sentimentString +=(char) sentimentText.read();

    }
```

Convert sentimentText to a string

To convert a string to a float, we must first convert the string to an array of characters.

```
char floatBuffer[32];

sentimentString.toCharArray(floatBuffer,sizeof(floatBuffer));
float sent = atof(floatBuffer);
```

This code is not very intuitive, but it results in a float that holds our value

Finish it up!

Now we just need to return our float value and close out of the sentimentText file that we opened.

```
sentimentText.close();
return sent;
```