Internet of Things

Session 1

November 2015

Materials for This Activity

Materials such as code samples, presentations, executables, etc. can be accessed from:

https://github.com/adeaver/InspernetOfThings

To start today's activity, go to this repository and download the material. You'll need "fb_analyze.py" which is in the Code folder.

Understanding the Starter Code

In order to save time in this project, we’ve provided two starter code files. To understand the code being used in these examples, we’ve included an optional presentation called “Understanding the Starter Code”

Note: You will not be able to run “fb_analyze.py” right away. There are instructions on how to do this in “Understanding the Starter Code” but it is not necessary for the Internet of Things activity.

First Project Overview

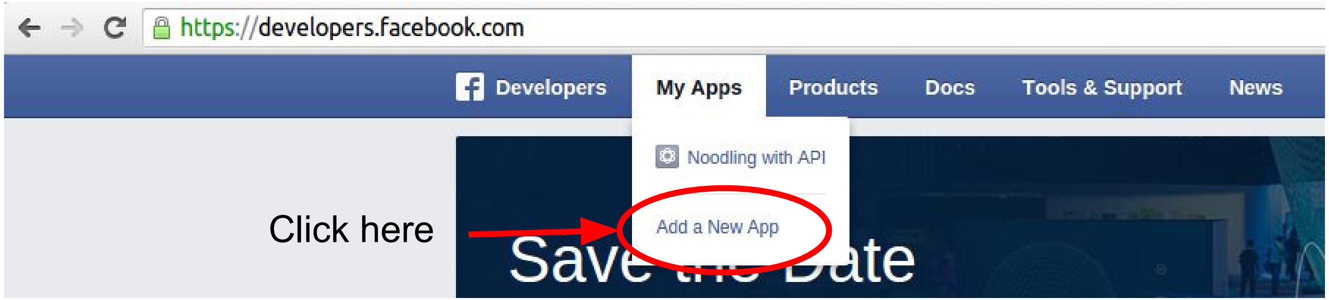
We'll be making a sentiment meter to show the sentiment of Facebook posts that will looking something similar to the image below. We have already provided the python code that we'll use in this project.

POOR
FAIR
GOOD
EXCELLENT

www.campaignasia.com

Getting set up with Facebook

Go to https://developers.facebook.com



A pop-up menu should appear, choose the “website” option

Give your app a name and click “Create new App ID”

Getting set up with Facebook, pt. 2

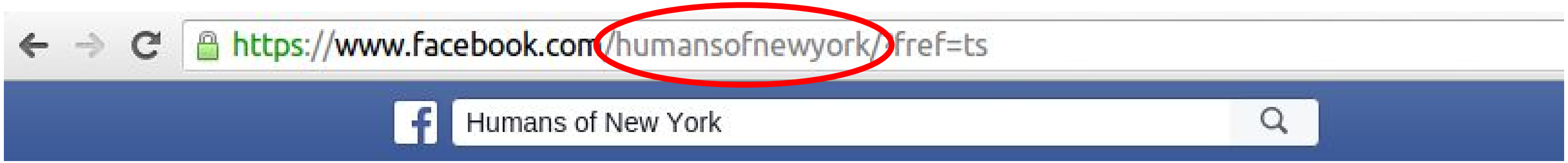
Select any category for your app and accept. On the next page, click “Skip Quick Start”

Now get an access token by going to: https://developers.facebook.com/tools/access_token

Copy and paste the “app token” and replace “your_access_token_here” on line 4 of fb_analyze.py

Finding Content to Analyze

Open facebook and go to a page whose content you want to analyze. Pay attention to the URL. The part in red is the page name.



https://www.facebook.com/humansofnewyork/?fref=ts

Humans of New York

Now go to: https://graph.facebook.com/{page_name}/?access_token=*your_token*

Replace “some_id_here” in line 7 of fb_analyze.py with the id.

NOTE: It is important that this page is in English (you can also use French or Dutch pages, but there’s a little more that needs to be done for this.)

Onto the Hardware!

Connect the Galileo to your computer via the 6 pin USB-Serial Connection.

- If you’re running Windows, you may need to open up the device manager to install the software. You will always need to find the COM port here.

Find the directory of your SD card by running the sddirectory.ino sketch provided in the “Helper Code”

Open up PuTTy (the executable is available on the repository for this activity). Use the Serial connection option, with the port as the COM port you found in the device manager and the rate as 115200.

Getting Galileo Set Up for Python

type **vi /etc/opkg/base-feeds.conf**

- This will put you in vim, to insert text in vim, type “i”, and then copy and paste the following lines (you can find this on the repository):
  - **src/gz all http://repo.opkg.net/galileo/repo/all**
  - **src/gz clanton http://repo.opkg.net/galileo/repo/clanton**
  - **src/gz i586 http://repo.opkg.net/galileo/repo/i586**
- Type “esc” then “:wq!” and press enter.
- Now run the command **opkg update**
- Then type **cd /home/root**
- Type **ifconfig** and pay attention to **inet addr** under **eth0**

Getting Galileo Set Up for Python, pt. 2

In your Command Prompt on your desktop, navigate to the directory where you cloned the Internet of Things repository and go to the “Helper Code” folder and type:

```
pscp -scp ./get-pip.py root@[IP ADDRESS]:/home/root
```

Now, back on the Galileo, type the command **python get-pip.py** this will install pip.

- Note: You may need to update the time and date.
    - To do this, type **date MMddhhmmYY** replacing each part with the current UTC time. For example, November 11th, 2015 at 6pm UTC would look like **1111180015**

Getting Galileo Set Up for Python, pt. 3

On the Galileo, type the commands:

pip install facebook-sdk

pip install --no-cache-dir pattern (this one will take a really long time)

This will install the python packages that we need.

Moving the Python Script to the Galileo

Back on your computer, navigate to the “code” directory. Type the command:

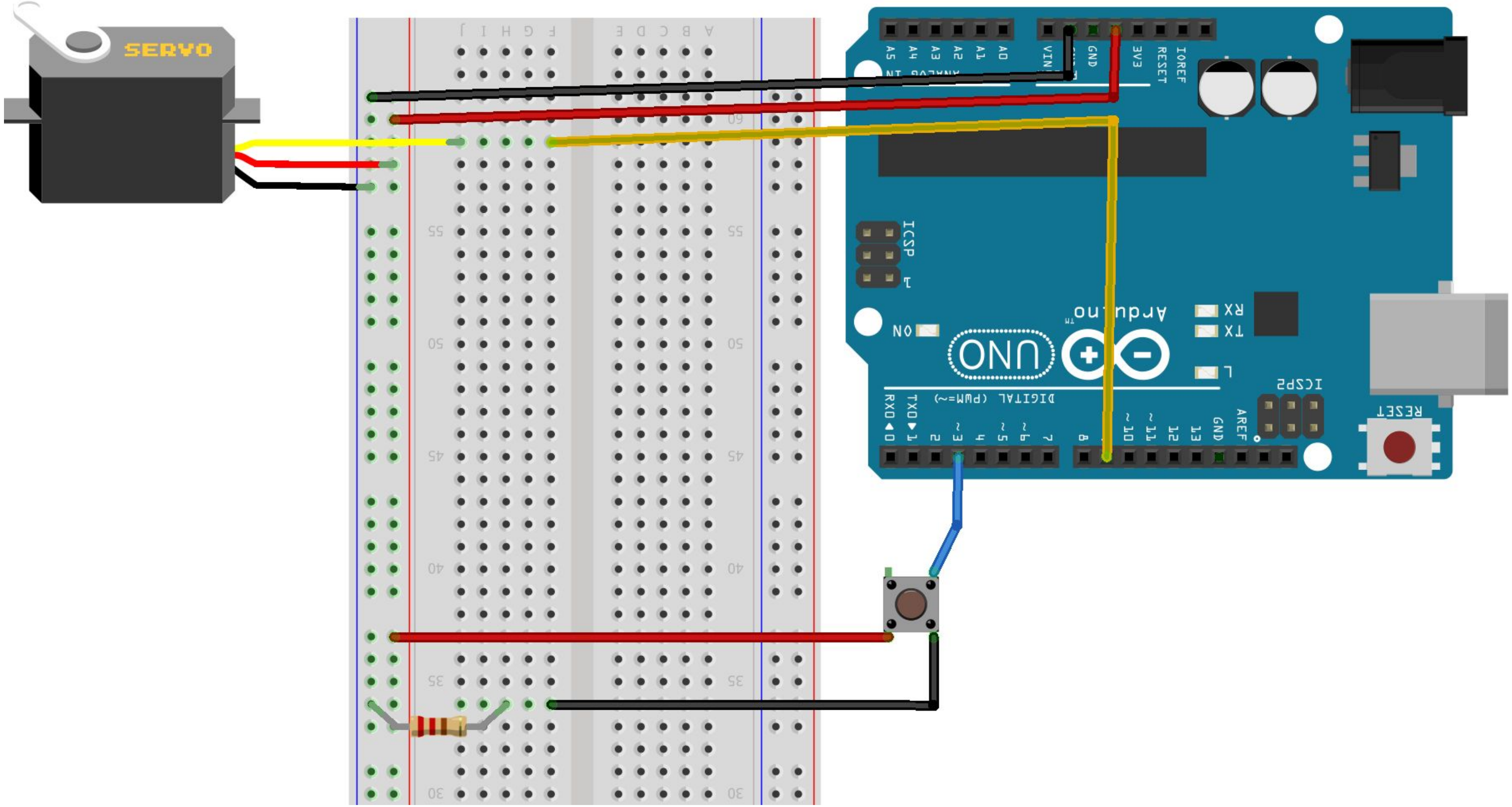
```
pscp -scp ./fb_analyze.py root@[IP ADDRESS HERE]:/home/root
```

Back on the Galileo, type the commands:

```
cd /home/root
```

```
mv ./fb_analyze.py [sd card directory that you found earlier]
```

Circuitry


SERVO
Arduino
UNO
ICSP
ICSP2
RESET
DIGITAL (PWM=~)
ANALOG IN
ON
RX
TX
L
AREF
GND
5V
3V3
VIN
IOREF

Modifying the Arduino Starter Code

The end result -> Rotate the servo every time a button is pushed according to the sentiment value.

High Level approach:

1. Detect when the button is pushed
2. Read the sentiment (implement the method)
3. Determine the position of the servo based on this sentiment
4. Rotate the servo

Part 1: Detecting the Button Push

How do we read the input from the button?

```
int pushed = digitalRead(pushButton);
```

What does the input look like? What means the button has been pressed?

```
if(pushed) { }
```

All of the code will go inside of the if-statement from now on.

Part 2: Reading the Sentiment/Determining the Position

```
float sentiment = readSentiment();
```

This value will be between -1 and 1, how do we convert this to an angle? Note, this must be an integer value.

int angle = (int) (90.0 + (90 * sentiment));

Part 3: Rotate the Servo

The servo object has a method called “write” which will put the servo at a certain angle.

```
myservo.write(angle);
```

If you want to use a different status every time, make sure to increment the index variable.

```
index++;
```

Troubleshooting

Make sure the directory for the python script and SD card are correct. We’ve included an Arduino sketch in the Helper Code folder to do this.