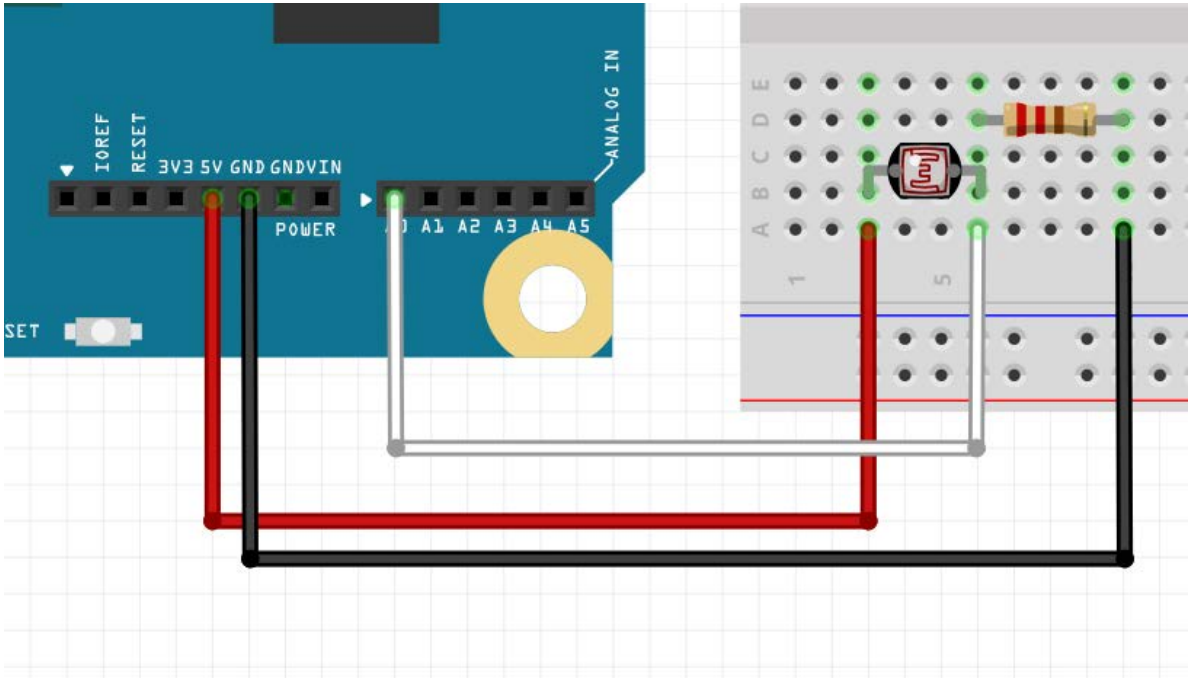
Internet of Things

Publishing Sensor Data
Thingspeak and Emails

Sensor Circuit

- Choose your sensor (light sensor recommended)
- Connect power and ground
- Connect the output pin to A0 (white)



Read the Analog Input

- Go to **File > Examples > 03.Analog > AnalogInput**
- Look through the sketch and read the comments to understand how it works, then upload it to see it in action
- With a light sensor, cover it with your fingers or shine a light at it to see the LED frequency change

AnalogRead() function

- `AnalogRead()` reads the sensor value as an integer between 0 and 1023
- This corresponds proportionally to a voltage. If you're connected to 5V power, then a reading of 1023 = 5V.
- Therefore, a reading of 500 would correspond to:

  500 / 1023 * 5 = 2.44 V

Publishing to Thingspeak

- Thingspeak is an open-source platform for real-time data collection that has an extensive API
- Easy to integrate with Arduino and Galileo
- Go to https://thingspeak.com/ and create an account

Making a Channel

- When you have your account set up, go to the **Channels** tab and click “New Channel”
- Change the **Name** to something descriptive like “Galileo Light Sensor”
- Put something in the description like “Insper Galileo workshop – publishing data”
- Change **Field 1** to reflect what the axis label will be – “voltage”
- Click **Save**, then go to the **API Keys** tab and copy down your key

Using Python to Publish to Thingspeak

- Open the *thingspeak_sender.py* file in a text editor and edit the **API_KEY** variable to contain your API Key from Thingspeak
- Notice that the Python script uses **sys** to take an argument from command line
- Transfer the file to the Galileo via **pscp** and try a test run with a random voltage as the argument, for example:

```
python thingspeak_sender.py 3
```

- Go to your Thingspeak channel online to see the change on your graph

Sending Arduino Data to Thingspeak

- Now that we have a Python script set up, we can edit the Arduino code to send data *to* Python
- Open *data_sender.ino*

Arduino Code: Loop()

- In loop(), if the update rate has passed, it uses **analogRead()** to get data from the sensor pin
- Converts the reading to a voltage
- Passes the voltage into a function I've defined, called **send_voltage**

Arduino Code: send_voltage

- There are a lot of extra parts in this that you don't need to worry about
  - This is because of data types in C – we have to convert between a String and a character array
- The important line is the one that creates the request that we'll execute in Linux
  - The parts before use a helper function to convert the voltage to a string so we concatenate it
  - The parts after convert the String to a character array so it can sent in the **system** command

Running the Code

- Be sure to move or copy *thingspeak_sender.py* to the ***/media/realroot*** directory in Linux
- Upload *data_sender.ino* to the Galileo
- Watch as your graph updates on Thingspeak online in real time!

Using Python to send Emails

- Open the *email_sender.py* file in a text editor
- This uses the smtplib library to login to a Gmail account and send an email
- Replace the **to_address** with your email and the **body** with text of your choice
- Save the file and transfer it to your Galileo. Run it using Python and see yourself receive the email

Project

- Use the Galileo to send yourself an email every time the voltage from your light sensor changes by more than 1 volt
- Have the message text change to indicate whether it got darker or brighter

Project Scaffolding

- Edit the *email_sender.py* file so that it takes a system argument as the message body
- Create a new Arduino file based on *sensor_thingspeak.ino* so that if the voltage has changed by more than 1V, it executes the Python script, and sends an appropriate message as an argument

Notes on Emails

- If you want to send emails from your own Gmail account instead of the Insper one, you will need to allow access to less secure apps by enabling it here: https://www.google.com/settings/security/lesssecureapps