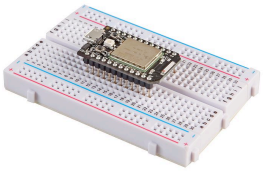


Drones/Sensors Workshop
Dr. Cindy Royal
Texas State University - San Marcos
School of Journalism and Mass Communication

Using Sensors

Particle Photon - we are using the Particle Photon, a wifi development kit for creating connected products. The kit includes various features including the chip with wifi and various digital and analog connectors, a breadboard for mounting the chip and making connections, USB Power connector, LED lights, Photo Resistors, Temperature sensor and various capacitors. The company was previously named Spark and the previous product was called Core. The Photon replaced the Core, but the functionality is very similar. The Photon Maker Kit costs only $19. https://store.particle.io/?product=particle-photon

Set Up
The kits are already set up for you with department accounts. But here is the process that we used, should you want to purchase a sensor kit yourself and try these steps after the workshop.

Start by finding the Photon in its pouch. It is already mounted on the breadboard. Also, find the USB power connector. You connect this to the computer with the small end into the chip. Be sure that the Photon star symbol is facing the top.

The setup instructions can also be found here:
https://docs.particle.io/guide/getting-started/start/photon/

You need:

- Photon and Breadboard with USB connector (connecting to laptop or desktop)
- Wifi
- Particle app downloaded to iPhone or Android (you will each do this on your own phone prior to coming to class, so you can login to work on the projects).

Each of these kits has a gmail account and particle account associated with it. You will be given that information in class.

Connect the Photon to the computer via the USB connection. As soon as it is

plugged in, the LED on your device should begin blinking blue. If your device is not blinking blue, hold down the SETUP button.

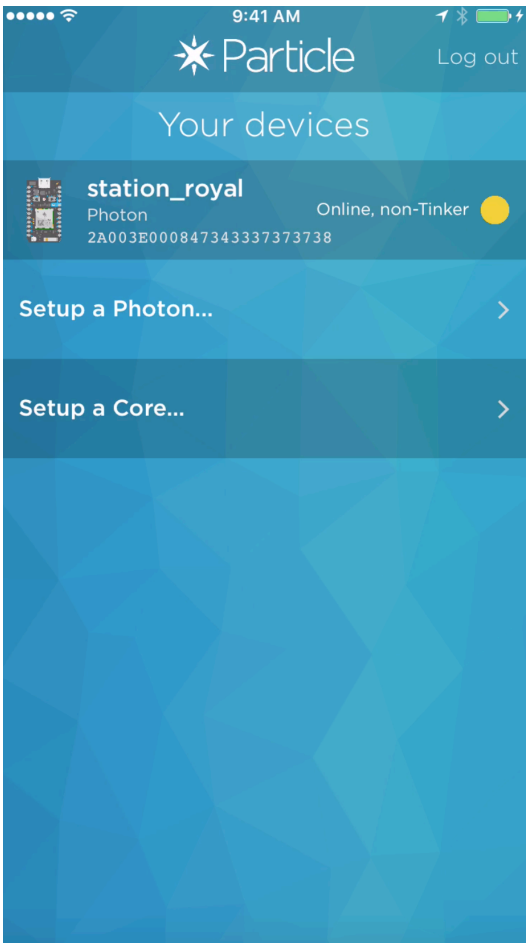
If your device is not blinking at all, or if the LED is burning a dull orange color, it may not be getting enough power. Try changing your power source, but connecting to your computer via USB should be fine. The USB connection is solely for power. The Photon will be communicating over wifi with the app and with the Internet.

Download the Particle App on your smartphone. It's free. Go through your app store. There are versions for iOS and Android. You will be given the credentials for signing in during class. This part will all be done for you in advance, so when you login you will see your station in the list.

Setup a Photon
Make sure your smartphone is connected to the Internet. Choose Setup a Photon. Tap Home--> Settings --> Wifi and choose the network associated with your Photon. The four-digit code after Photon- is unique to your device. Select the wifi network you are on, providing password. Wait a few seconds for the device to connect. Give your photon a unique name (ours are already set up).

Working with the Photon (this is where we will pick up in the workshop)

When you login to the Particle app, you should see the Photon listed with Your Devices on the Tinker screen. If the status is "Online, not Tinker," you will need to Flash the Tinker software to your device. Click the yellow button and choose Flash. Wait while the software is flashing. If it says Offline, you may have to logout of Particle and log back in. You should see Online next to your device in the list and the status LED should be slowing breathing cyan.



Click the device to enter the Tinker software. Click to get past the welcome screen if you see one. You will then see a schematic of the pins on the device.

The D7 pin comes already wired to a small blue LED on the face of your device. When you set the power of the D7 pin to high, this LED turns on. Let's do that now in the Tinker screen.

Tap D7 then digitalWrite in the popup. Now when you tap the D7 circle the tiny blue LED should turn off or on. High is on, Low is off. It may take a second or two to communicate with the device. To stop using a button, hold on to it for a second and it will return to its original state.

Congratulations, you just blinked an LED over the Internet, using your Particle device!

We're going to be doing a lot more with the Photon, but this assures that everything is communicating correctly. It is basically the "Hello World" of sensors.

Device Modes
Here are some of the modes of your device, based on the color and behavior of the status LED.

Connected: When it is **breathing cyan**, your device is happily connected to the internet. When it is in this mode, you can call functions and flash code.

Firmware Update: If your device is **flashing magenta**, it is currently loading an app or updating its firmware.

Connecting to Internet: If your device is **flashing green**, it is trying to connect to the Internet.

Connecting to Cloud: When the device is in the process of connecting to the cloud, it will rapidly f**lash cyan.**

WiFi Off: If your device is **breathing white,** the Wi-Fi module is off.

Listening Mode: When your device is in Listening Mode, it is waiting for your input to connect to the wifi. Your device needs to be in Listening Mode in order to begin connecting with the Mobile App or over USB. To put your device in Listening Mode, hold the SETUP button for three seconds, until the RGB LED begins flashing blue.

Tinker Interface: The app consists of 16 pins in vertical rows - 8 analog pins on the left, 8 digital pins on the right. These pins represent the 16 GPIO (General Purpose Input and Output) pins on your device.

- digitalWrite: Sets the pin to HIGH or LOW, which either connects it to 3.3V (the maximum voltage of the system) or to GND (ground). Like we did above with the D7 pin.
- analogWrite: Sets the pin to a value between 0 and 255, where 0 is the same as LOW and 255 is the same as HIGH. You could use analogWrite to dim an LED, as an example.
- digitalRead: This will read the digital value of a pin, which can be read as either HIGH or LOW.
- analogRead: This will read the analog value of a pin, which is a value from 0 to 4095, where 0 is LOW (GND) and 4095 is HIGH (3.3V). All of the analog pins (A0 to A7) can handle this. analogRead is great for reading data from sensors.

We did digitalWrite above with the D7 pin and onboard LED. Now let's wire something and get an LED to brighten and dim.

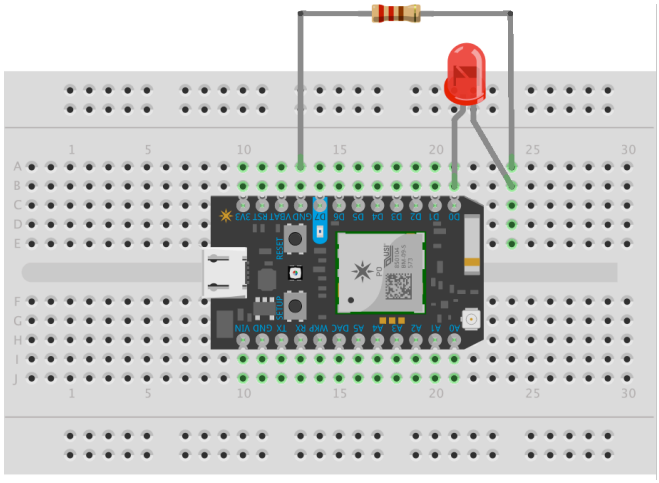
Wiring an LED
Your kit comes with several packages of materials. We have pared down what came in the Photon Kit to focus on the things you will need for this exercise. Find the LED lights and small resistors.

In this example, we'll plug an LED into D0 and change its brightness with analogWrite.

The breadboard allows you to connect the items to pins while you prototype an application. When you want to actually use a sensor, you will solder the wires to the proper location on the chip.

You can access the pins through any of the holes in the breadboard next to it. The Photon chip is on the breadboard when you unpackage it, but it can be moved around, if necessary to have access to more connections on one side or another.

Put the long wire of the LED into a hole adjacent to D0. Put the shorter wire in another hole away from the Photon in the same row. Find the small resistors with the brown, red and black handles in the middle. Then wire a resistor from the same row as the long wire to the GRN wire. This grounds the circuit.

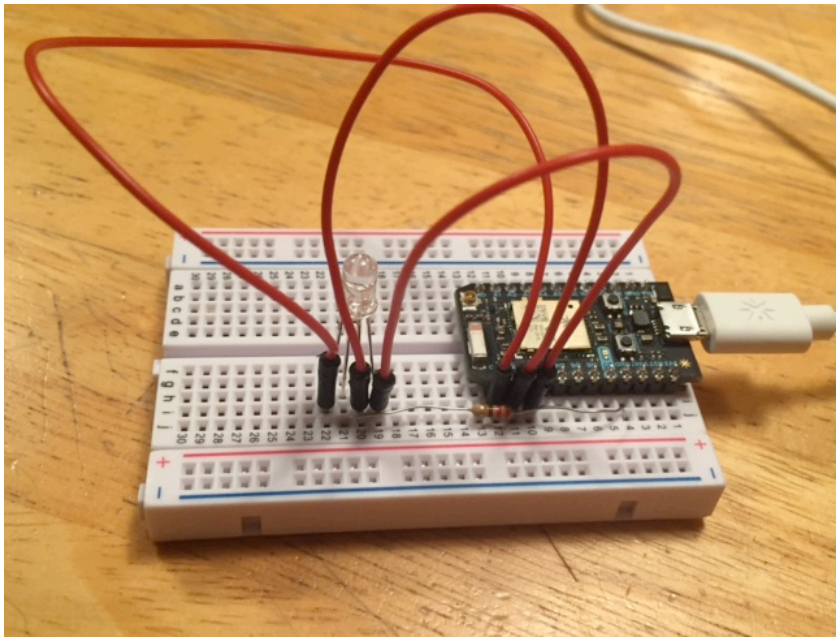


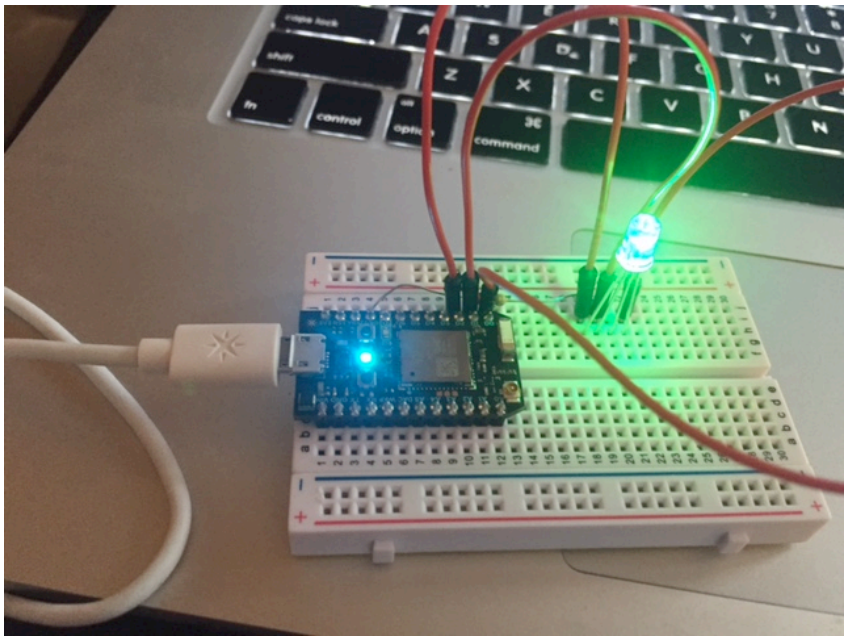
You will get used to these wiring diagrams. If you have done this correctly, you should see the LED getting brighter or dimmer as you use the slider next to D0. This works because we wired the LED to D0 and grounded it. Ground is necessary so the circuit has a return path.

The exercise above is basically taking the voltage going to the LED and dividing it depending on the value of the slider to make it brighter or dimmer.

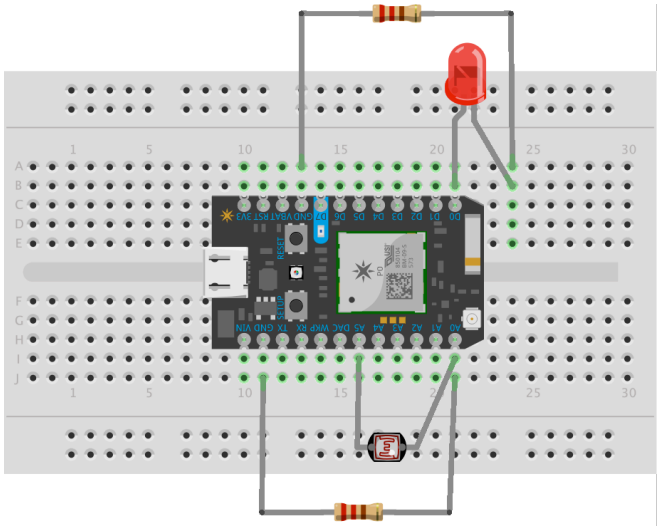
Wiring the RGB LED

Depending on time, we may or may not do this in the workshop. There is an RBG LED included in the kit. This allows you to connect each color and then mix the color of the output with the sliders on the app. It has four wires, one longer than the rest. The longer wire should be connected to GND. This allows you to wire up different controls for Red, Green and Blue to create different colors by sliding the respective sliders. You will use the same kind of resistor as above to connect to GND, then use colored wires from the three on the LED to D0, D1, D2, as below. Give it a try.





Using a Photo Resistor
Next we will read from a sensor. We'll be using the Photo Resistor, so find the item with two wires that looks like it has a squiggly wave on a brown head.



Wire the photo resistor to A5 and A0. Keep these in the same row on the breadboard.

Put a small resistor from A0 to GND. Keep these in the same row on the breadboard.

You can leave your LED connections on the D side as is.

Tap the A5 pin in Tinker and set it to digitalWrite and HIGH.

This essentially gives us a consistent power source from A5 that will go to our photo resistor.

Now tap A0 and set it to analogRead. You will see a number reflecting brightness of light.

Now cover the photo resistor and tap A0 again. See the difference?

Using the Web IDE (Interactive Development Environment)

That was fun, but there are a lot more things we can do with the Photon if we use a bit of code. Luckily, Particle provides a development environment with code samples we can run and test out.

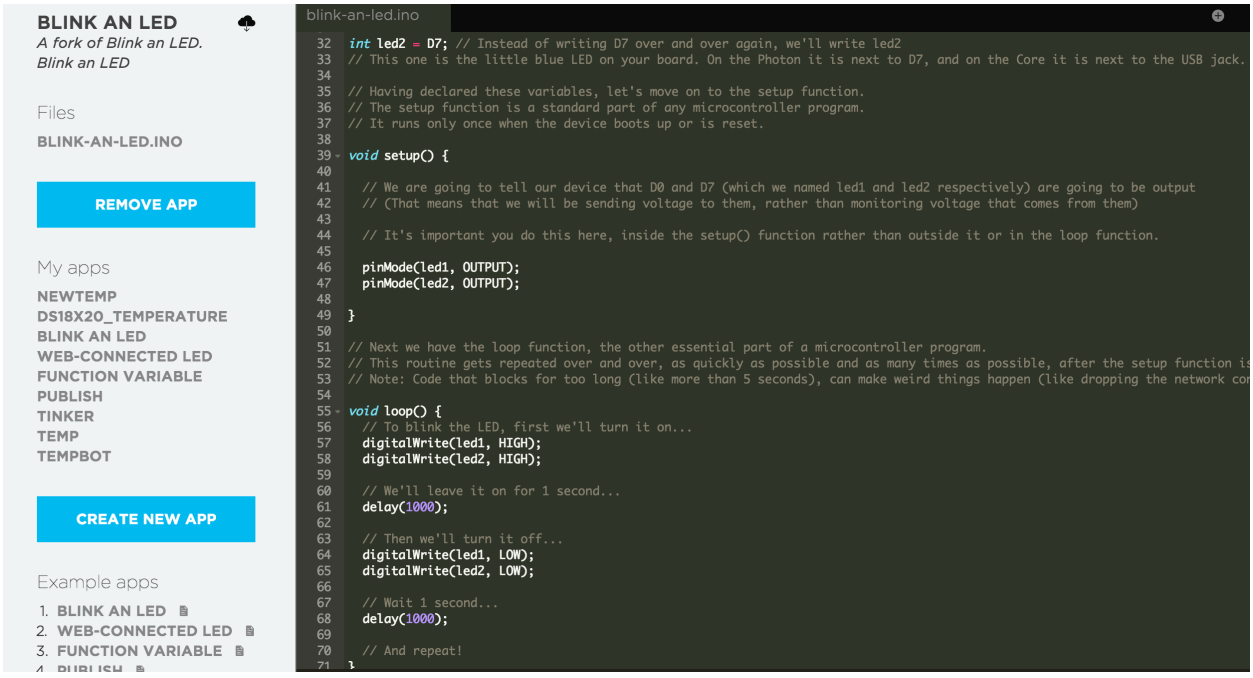
Go to build.particle.io and sign in with the account for your device.

The interface includes buttons for:

- Flash: Flashes the current code to the device. This initiates an over-the-air firmware update and loads the new software onto your device.
- Verify: This compiles your code without actually flashing it to the device; if there are any errors in your code, they will be shown in the debug console on the bottom of the screen.
- Save: Saves any changes you've made to your code.

You will see code on the right side of the screen.

There are some example apps already available for us to try out. We will be forking (copying) each example so we can run and modify for our devices.



The following examples are also available here:
https://docs.particle.io/guide/getting-started/examples/core/

Blink an LED
Click on the example app Blink an LED and choose to fork it. You can use the Save button to save it, the Verify button to verify that the code is correct and the Flash button to test it.

Try it. Let it run through the cycle. It should show both LEDs blinking.

Find the delay in the code. Change it. Verify it and Flash. See what happens.

Web-Connected LED
Now let's try the Web-Connected LED. Find the code and fork it. Save it.

Open a text editor on your computer (TextEdit is fine, although make sure to turn off Rich text in the preferences). Create an html page that looks like this. It is a basic html page with a form with two radio button inputs and a submit button. There is a sample called led.html in the GitHub account github.com/cindyroyal/dronessensors.git.

The action takes you to the API call for the LED status.

```
<!-- Replace your-device-ID-goes-here with your actual device ID
and replace your-access-token-goes-here with your actual access token-->
<!DOCTYPE>
<html>
  <body>
  <center>
  <br>
  <br>
  <br>
  <form action="https://api.particle.io/v1/devices/your-device-ID-goes-
here/led?access_token=your-access-token-goes-here" method="POST">
    Tell your device what to do!<br>
    <br>
    <input type="radio" name="args" value="on">Turn the LED on.
    <br>
    <input type="radio" name="args" value="off">Turn the LED off.
    <br>
    <br>
    <input type="submit" value="Do it!">
  </form>
  </center>
  </body>
</html>
```

As the comment says, replace your device id and access token. You can find your device id under the Device button on the Build screen. Click on the device. Copy and paste that number into the html code.

Click on the Settings icon at the bottom of the Build page to get your Access Token. Replace that in the html code. Make sure you are inputting this info after the "=" and before the closing ".

Let's name it led.html. Save it on the desktop, so it is easy to find.

You need to verify and Flash the code in the Build screen first. Wait for it to complete the cycle.

Go to that file in a browser. File, Open File.

Try it to see if it works. You should see the LED turn on when you select on. The page then goes to the api for the LED setting, letting you see the result. Hit the back button to keep trying. Were you able to turn the LED on and off from your computer?

The data sent via the request is what is shown on the screen in the API. The value is either 0 or 1. That's digital. On or off.

Click the Code icon if you need to get back to the Code page.

Function and Variable with Photo Resistors
Your photo resistor should still be wired up.

Choose the Function Variable code, fork it and save it.

Make sure the LED and photo resistor are pointing at one another. Gently push one toward the other.

Verify and Flash the code in the Build screen.

Put this address in your browser, changing the device number and access token, as you did before. This code is slightly different than what we used before, accessing the analogvalue, rather than the LED status.

https://api.particle.io/v1/devices/your-device-ID-goes-here/analogvalue?access_token=your-access-token-goes-here

As you change the brightness that the photo resistor receives, every time you make a request to the api, you should see a different value. Carefully cover the breadboard. Then use the html page you created to turn the LED on, check the api. Then off, check the api again for the result value.

Motion Detector
This app will publish a status based on when the light beam between the photo resistor and the LED is broken.

Find the Publish code. Fork it. Verify and Flash it. Wait for the LED light to come back on.

Go to the Particle Dashboard at dashboard.particle.io and make sure you are logged in with your login info. Click the log button. If you put your finger between the LED and photo resistor, you should see it logging the activity.

Tinker
If you want to return the device to the Tinker state, you can do this by running the Tinker script. This lets you use Tinker on your device while running a script. For example, you can control the brightness of the LED and see how that changes the value from the photo resistor on A0.

Just fork, verify and flash the Tinker script. This is the same as “flashing” code from the Web IDE.

Temperature Exercise
The final exercise we will do is the modified Temperature Readings Exercise provided here by Matt Waite on his GitHub account. https://github.com/mattwaite/NICAR15SparkCoreHandsOn. But I have everything you will need on my GitHub account at github.com/cindyroyal/sensordemo.

We'll be wiring up the Photon with the thermometer, which you should be able to find with your materials in the Photon Kit. It is a small black device with three wires. If you can read the writing on it, it has the letters "TMP" and some numbers on it.

Remove all the previous wiring. Goodbye to LED and Photo Resistors.

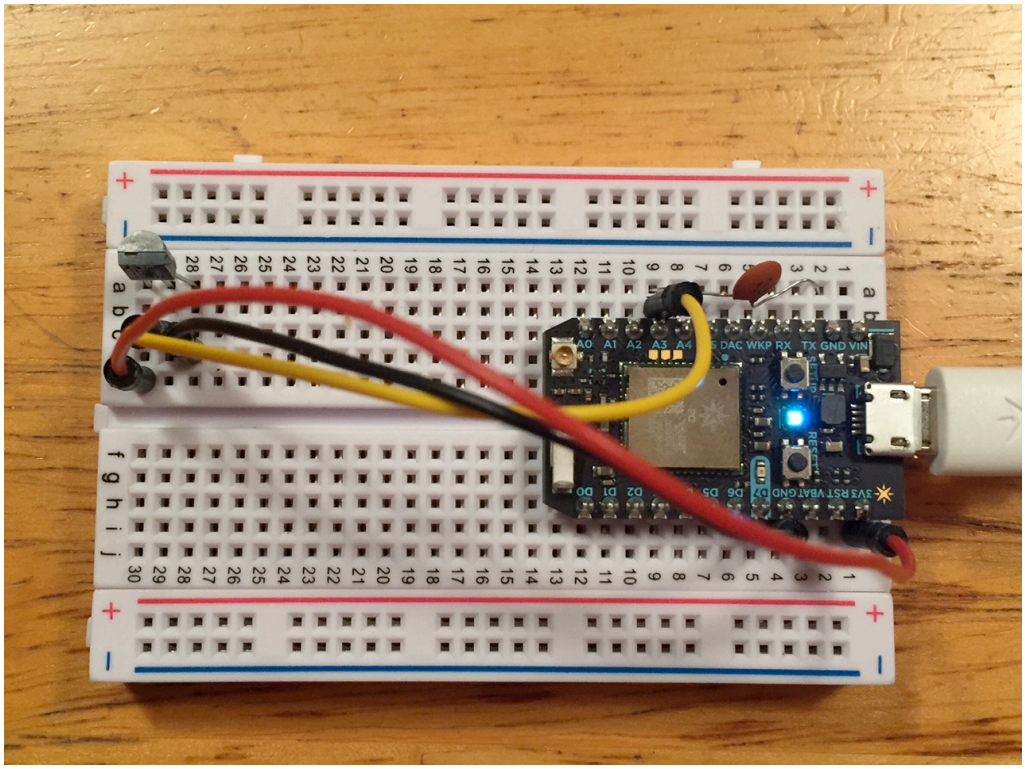
In addition to the thermometer, get three colored wires, preferably red, black and yellow. This is how we will create the circuit for the Photon to be able to log temps.

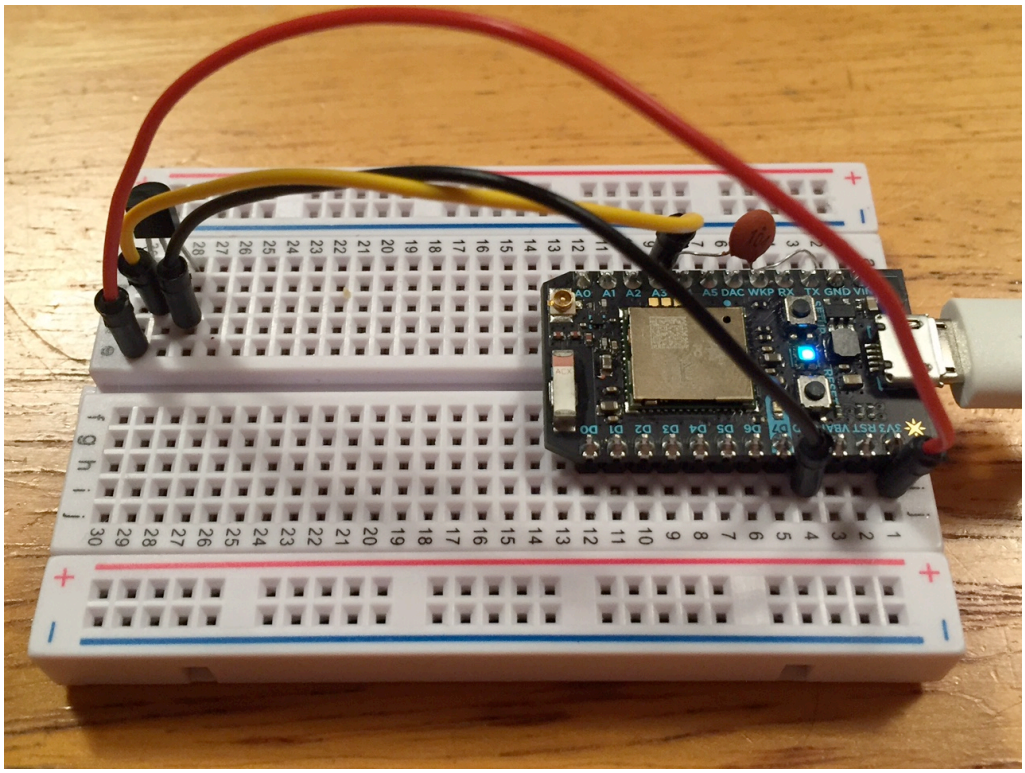
Carefully wire the Photon as follows:

Have the breadboard with the power connector facing to the right.

Put the Thermometer off to the side on the breadboard. Make sure the flat side of the thermometer is facing you. You can use the breadboard's a28, a29, a39 holes.

We will connect each of the wires on the thermometer to the appropriate place on the chip.





The black wire is the ground. Put one side in d30 and the other in a hole next to GRN on the D side of the chip.

The red wire is the power. Use the d28 hole and the other side to 3V3 on the chip.

The yellow is used for the input. Connect it to b29 and the other end to A3.

We also need to add a small capacitor to reduce noise from the input. This helps us get more accurate temp readings. Find the capacitor with the terra cotta colored circle on it that says 104.

The colors of the wires don't really matter. They all do the same thing. But it is good to use different colors to reduce confusion.

Once you have the chip wired properly, in the GitHub account, find the tempbot.ino file. This file will track the temperature and report to an API call. Copy the file from GitHub and use Create New App and add that code to your account. Save, Verify and Flash it.

Now go to this API call.
https://api.particle.io/v1/devices/your-device-code/temperature?access_token=your-access token

Make sure you use your device code and your access token where specified. Notice the change in the API call is for “temperature”. Our previous calls were for “led” and “analogvalue”. This is the variable right before the “?”.

You find your device code under the Devices button on the Particle Build page. You find the Access Token under the Settings button.

Check that out. Did it work? Do you see a value? It may not be exactly right, due to the sensitivity of the temperature sensor, but you should be getting something close to a room temp.

Logging Temps
Last exercise. In our code, there are a couple of lines:

```
// Google API needs it in a string, so make it a string
  String temp = String(temperature);

  // This is the bit that pushes it to Particle, which sends it to IFTTT,
which sends it to Google Spreadsheets. No lie. This is it.
  Particle.publish("temperature", temp);
```

The first line converts the temperature to a string variable. The second line

publishes to IFTTT (account already set up for you based on your device), which then pushes it to a Google Drive spreadsheet (already set up for you).

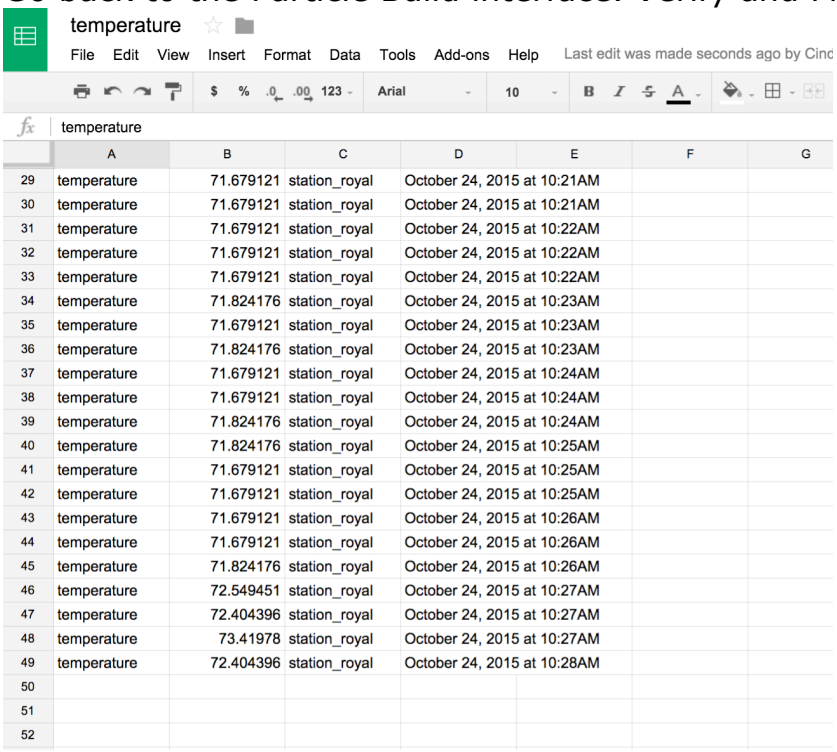
Note: The code we used from examples use Spark.publish and Spark.variable. With the company name change, Particle.publish and Particle.variable now work and will be updated in the future. I have updated this in our tempbot.ino code.

Go to this already created recipe to log events to Google Drive at https://ifttt.com/recipes/244508-log-events-in-google-drive . Login with your credentials given at the beginning of the workshop. After you login, you have to go back to the url above.

Here you will connect your Particle account and your Google account. Luckily, all these logins are the same. Allow Google Drive to access this gmail (make sure it is the workshop gmail and not your own. If you have to logout of your gmail, do so before this step).

The only thing you have to do here is to set up the If (Event Name) to temperature. Select your device in the dropdown. The rest should be fine. It will create a folder called IFTTT, then a folder called events in the Google Drive account. It will begin logging in the temperature spreadsheet. Make sure you Add the recipe. You should be able to see this under My Recipes.

Go back to the Particle Build interface. Verify and Flash the tempbot.ino file.


temperature

File Edit View Insert Format Data Tools Add-ons Help Last edit was made seconds ago by Cinc

$ % .0 .00 123 Arial 10 B I S A

temperature

	A	B	C	D	E	F	G
29	temperature	71.679121	station_royal	October 24, 2015 at 10:21AM			
30	temperature	71.679121	station_royal	October 24, 2015 at 10:21AM			
31	temperature	71.679121	station_royal	October 24, 2015 at 10:22AM			
32	temperature	71.679121	station_royal	October 24, 2015 at 10:22AM			
33	temperature	71.679121	station_royal	October 24, 2015 at 10:22AM			
34	temperature	71.824176	station_royal	October 24, 2015 at 10:23AM			
35	temperature	71.679121	station_royal	October 24, 2015 at 10:23AM			
36	temperature	71.824176	station_royal	October 24, 2015 at 10:23AM			
37	temperature	71.679121	station_royal	October 24, 2015 at 10:24AM			
38	temperature	71.679121	station_royal	October 24, 2015 at 10:24AM			
39	temperature	71.824176	station_royal	October 24, 2015 at 10:24AM			
40	temperature	71.824176	station_royal	October 24, 2015 at 10:25AM			
41	temperature	71.679121	station_royal	October 24, 2015 at 10:25AM			
42	temperature	71.679121	station_royal	October 24, 2015 at 10:25AM			
43	temperature	71.679121	station_royal	October 24, 2015 at 10:26AM			
44	temperature	71.679121	station_royal	October 24, 2015 at 10:26AM			
45	temperature	71.824176	station_royal	October 24, 2015 at 10:26AM			
46	temperature	72.549451	station_royal	October 24, 2015 at 10:27AM			
47	temperature	72.404396	station_royal	October 24, 2015 at 10:27AM			
48	temperature	73.41978	station_royal	October 24, 2015 at 10:27AM			
49	temperature	72.404396	station_royal	October 24, 2015 at 10:28AM			
50							
51							
52							

Then go to the Google Drive (drive.google.com) associated with your device and see if it is logging temps every 20 seconds (delay in code is set to 20000 milliseconds).

Congratulations, you made it to the end of the workshop. Behold all that you have accomplished!