**Build a Telegram Bot for Arduino IoT Cloud4**

**1. Components Needed:**

1. Arduino MKR1010

2. Arduino IDE (1.8.5 or later)

3. Telegram

4. Micro-USB cable

5. Wi-Fi connection

6. LED

7. LDR

8. Servo Motor

9. Jumper cables

10. Breadboard

**2. Software Used:**

1. Arduino IDE 1.8.10

2. Telegram

**3. Workshop Skill Break-up**

80% software 20% Hardware

**4. Workshop Overview**

In this workshop, we will be using an Arduino MKR 1010 board with built-in Wi-Fi connectivity to set up a simple IoT automation system.

**5. Workshop Steps**

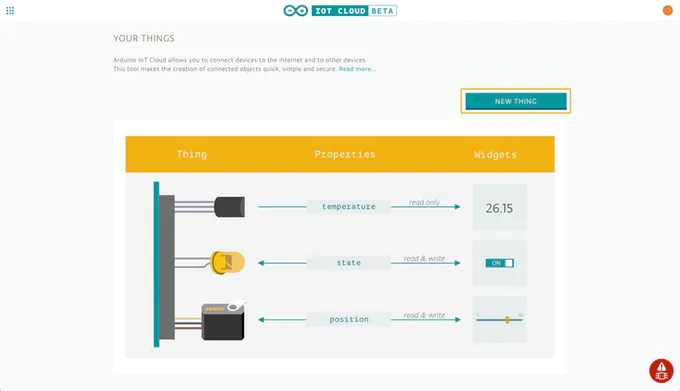
**5.1.** **Arduino IoT cloud**

The first method being discussed is setting up this automation system on Arduino IoT cloud. Due to the compilation time limitation, we will only be compiling for an LED and DHT11 temperature sensor and use widgets to control them.

* Setup:

1. Create an account on Arduino Create
2. Click on Arduino IoT Cloud
3. Follow the instructions on the screen to install Arduino Create Plugin and to setup your WiFi MKR 1010 Board

* LED and DHT11 sensor:

1. Once the Arduino MKR WiFi 1010 Board is set, create a new **‘Thing’**. 
2. Click ‘**Add Property**’ to create a new variable that can also be controlled by the cloud. We need three properties for this task.
3. ‘light’ would be the property for the LED as well as its variable name.

The value type is int and the range is between **0 and 255**.

Ensure **‘read and write**’ is checked.

1. ‘temperature’ would be the property for the DHT11 sensor as well as its variable name.

The value type is float and the range is between **-100 and 100**.

Ensure ‘**read only**’ is checked.

1. ‘Humidity’ is another property for the DHT11 sensor as well as its variable name.

The value type is float and the range is between **-100 and 100**.

Ensure ‘**read only**’ is checked.

1. Click ‘**Edit Sketch’**
2. On the left click **‘Libraries’**
3. Click **‘Custom’**
4. Import libraries from the github repository ‘**All Libraries**’ file. We need:

- Adafruit\_Sensor.h

- DHT.h

1. From the github repository, find the code to be used called **“IoTCloud\_code**”
2. Delete the existing code in the sketch and copy paste the “IoTCloud\_code”
3. In sketch, in the ‘Secret’ Tab, enter the SSID and password of your WiFi network
4. Click ‘Upload’ button at the top
5. Click ‘Go to IoT Cloud’ at the top
6. Go to ‘Dashboard’ in order to view your properties as widgets that you can control

**Telegram Bot For Arduino**

**Components Needed:**

1. Arduino MKR WiFi 1010

2. Jumper Wires

3. BreadBoard

4. LED

5. LDR

6. 1k ohm Resistor and a 470 ohm Resistor

7. Servo Motor

8. Temperature/Humidity Sensor DH11

**Software Used:**

1. Arduino IDE

2. Telegram

**Workshop Skill Break-up**

50% Hardware and 50% Software

**Workshop Overview**

The objective of this workshop is to control our Arduino board using a Telegram bot, for instance user can control an LED by sending a command to his bot such as “/ON“ which would turn the LED on. In this workshop, we will use the Telegram bot to control the brightness of an LED, give sensor readings for LDR and Temperature/Humidity Sensor, and control a servo motor.

**Workshop Steps**

Step 1: Install [Arduino IDE](https://www.arduino.cc/en/main/software).

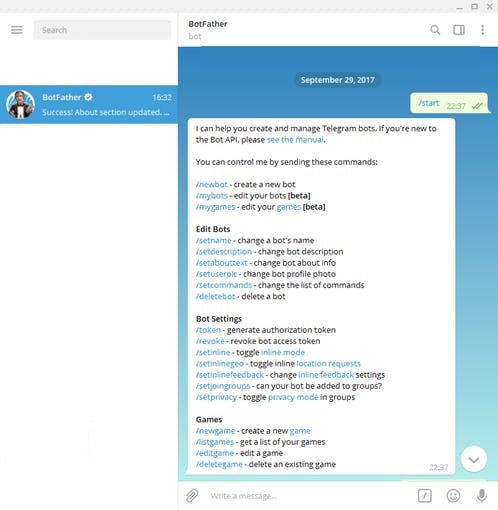
Step 2: Connect your Arduino Board to the computer and install the drivers.

Step 3: Install all [libraries](https://github.com/The-Assembly/Arduino-IOT/tree/master/All%20Libraries) needed.

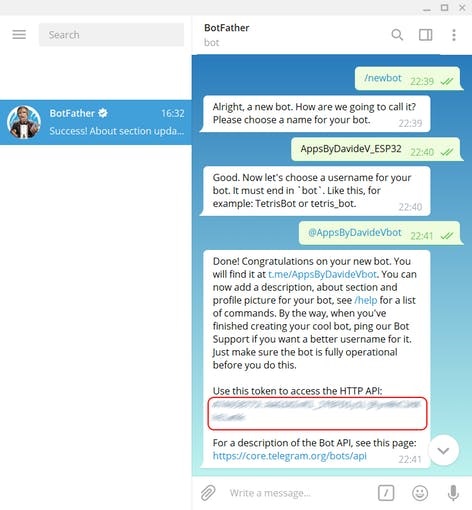
Step 4: Install [Telegram](https://desktop.telegram.org).

Step 5: Now we will try to set the Telegram bot, after you sign up for an account, go to the search bar and type **@BotFather**.

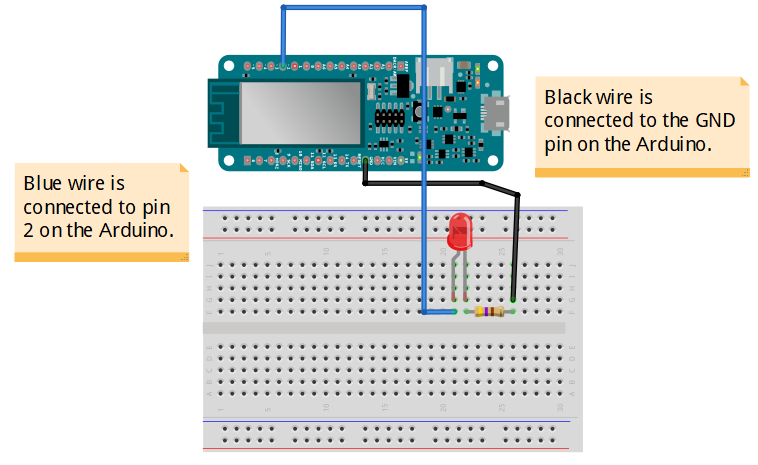
Step 6: Start a chat with BotFather by typing **/start**, after the **/start** command you will see the help list.



Step 7: Type **/newbot**, then type the name of your bot. BotFather will then ask you for a username. Your bot’s username must end in ‘**bot**’. If both the name and username are valid, you will receive a long string of chars called **TOKEN**. This is your personal ‘key’ used to communicate with Telegram servers (will be used in the Arduino code).

.

Step 8: Before we begin, you can access the full code [here](https://github.com/The-Assembly/Arduino-IOT/tree/master/Code%20for%20IDE%20(Telegram%20Bot)), note that I won’t be posting full lines of code in this tutorial, I will only be mentioning the important parts of the code that come with each connection we do. Now let’s start with the first connection which is the LED, this is how the circuit is connected (use the 470 ohm resistor):



Also, just a quick note, on this part of the code:



you will have to add your WiFi’s network and password. And on this part of the code:



you will have to add your BOT Token given to you by the BotFather.

Now, back to the LED, before the setup function, we added:



and inside the setup function, we added:



which sets the LED pin as output.

Now, inside the loop function, this is the bit of code we added:



We added an **if-statement**, this line of code: if (bot.messages[i].text == "/ON") means if YOU, the user, texted your bot the command **/ON** (to turn the LED on), then we will be adding 255 to the value of pulse we already initialised, therefore, pulse = 0 + 255 = 255. And this line of code: bot.sendMessage(bot.messages[i].chat\_id,"LED is ON!", ""); will program your bot to inform you that the LED is on. As for this line of: Serial.println(pulse); this will print the value of the pulse returned into the serial monitor. The second condition is when you text your bot the command **/Increase** (to increase the LED’s brightness), if the pulse of the LED smaller than or equal to 255, then we increment by 30, else the pulse will equal 255.

We are still not done, inside the loop function we have more lines of code:

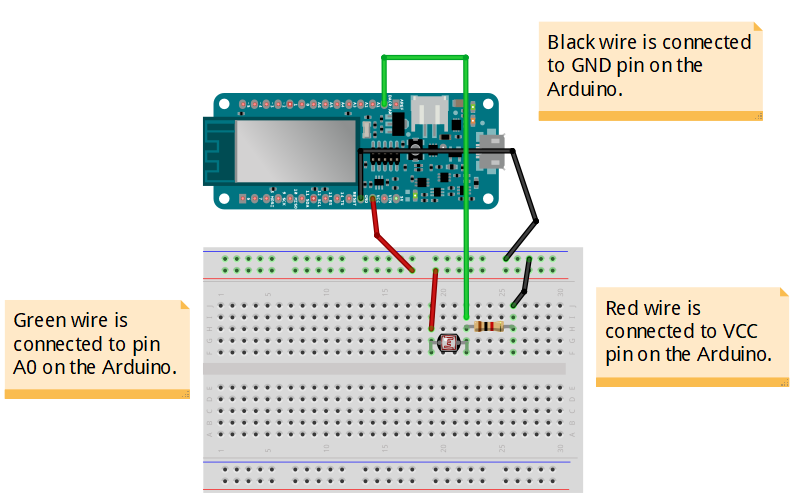


If you texted the bot the command **/Decrease** (to decrease the LED’s brightness), and the pulse is larger than or equal to 30, then we will decrement by 30. Else, the pulse will be 0. And if you texted the bot the command **/OFF**, the pulse will be equal to 0 and the LED will be turned off. Finally, at the end of the loop function, outside the **if-statement**, we added this line of code:



which writes an analog value (PWM wave) to the LEDPIN which is pin 2.

Step 9: Now let’s jump into the LDR connection (here we used the 1k ohm resistor):



Before the setup function, we added only two lines of code:



and:



which is just a declaration. In the setup function we have not added any new line of code. However, in the loop function we did. Starting with this line of code:



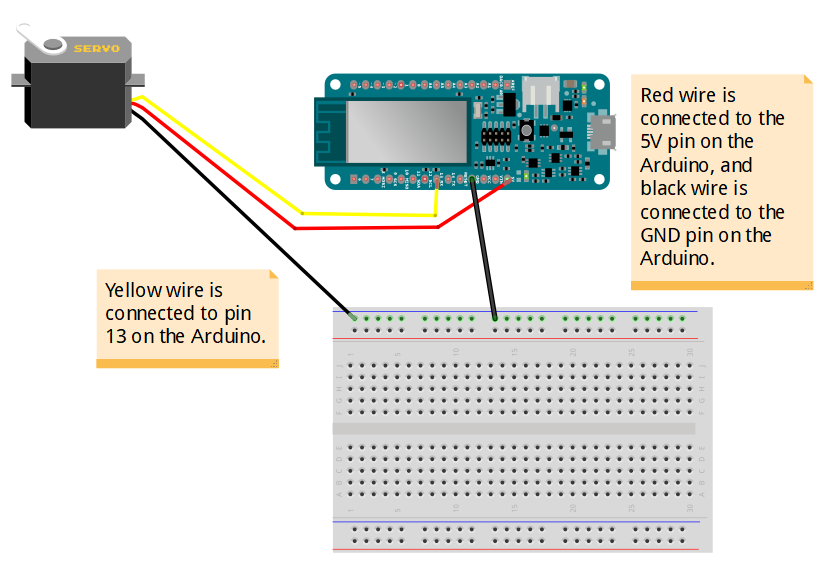
the analogRead() function reads the analog pin value from the specified analog pin which in this case is A0.

The rest of the code still inside the loop function is:



If you text your bot the command **/LDR**, there are four statements to follow, if the LDR value is less than or equal to 200 then your bot will inform you that it is DARK (if you place your hand on the LDR, so it won’t be exposed to any source of light) and it will attach the analog value which what the String(LDRvalue) returns. as for the three statements, these lines of code are pretty straightforward and self-explanatory, because they follow the same logic as the first statement. If you expose your LDR to your phone’s flashlight, and then sent your bot the command **/LDR**, here the LDRvalue will most likely be larger than 800, and hence your telegram bot will inform you that it is FULL DAY LIGHT along with the analog value of your sensor.

Step 10: Now for the Servo Motor connection:



Viewing the code [here](https://github.com/The-Assembly/Arduino-IOT/blob/master/Code%20for%20IDE%20(Telegram%20Bot)/IDE_Servo/IDE_Servo.ino), notice that we have installed the servo library, hence the #include <Servo.h>, now before the setup function, we add these two lines of code: #define servoPin 13 and Servo myservo; (creates servo object to control the servo). Inside the setup function, we added:

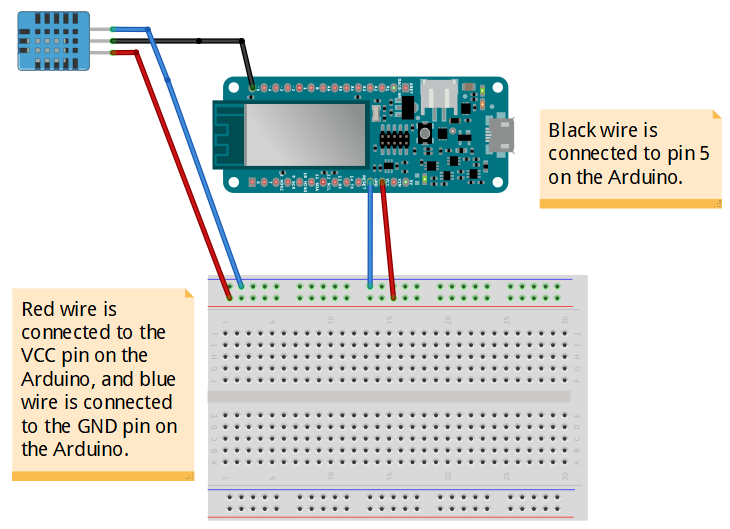


the first line attached the servo on pin 13 to the servo object, and the second line sets the servo position at 0. Now, here are the lines of code we added inside the loop function:

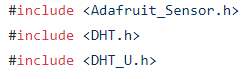


The code is pretty simple. If you texted your bot the command **/ONservo**, this will set the servo position at 90 degrees, if you texted you bot the command **/OFFservo**, this will set the servo position at 0 degrees, and if you texted your bot the command **/TRYservo**, your servo will change its position between 0 and 90 degrees twice.

Step 11: Finally, for the Temperature/Humidity sensor connection:



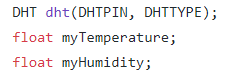
For the Temperature/Humidity sensor, take a look at the code [here](https://github.com/The-Assembly/Arduino-IOT/blob/master/Code%20for%20IDE%20(Telegram%20Bot)/IDE_Temperature_Humidity/IDE_Temperature_Humidity.ino), notice that we added three new libraries:



Now, before the setup function, these are the lines of code we added:

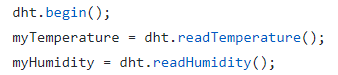


the DHTTYPE is just the type of sensor we are using, and:

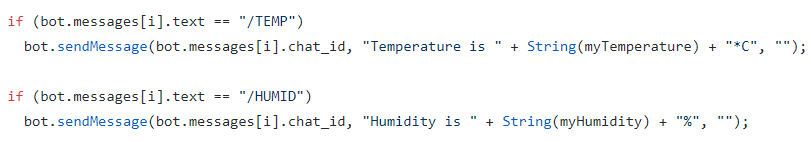


In the setup function we have not added any new code.

As for the loop function, we added these lines of code:

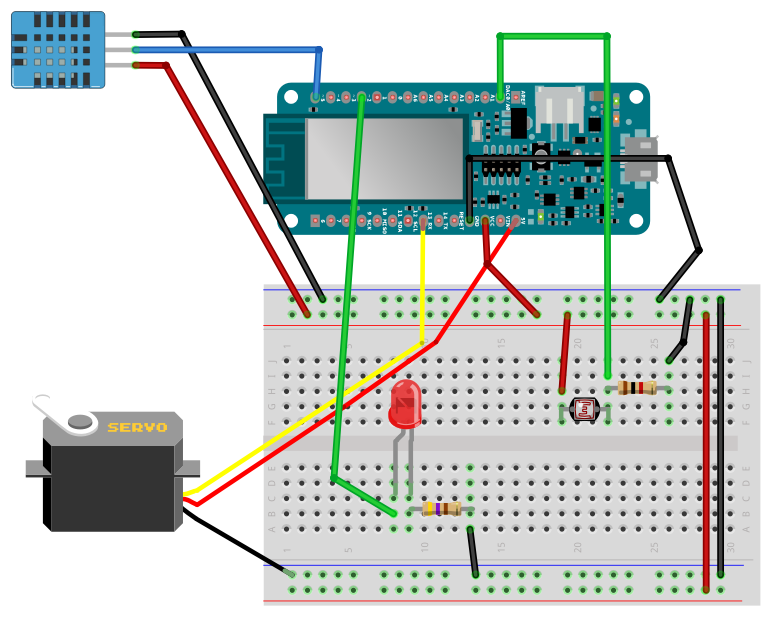


this will read the value of the temperature and humidity from the sensor. And:



the code is pretty simple, if you texted your bot the command **/TEMP**, your bot should return to you the temperature value, and if you texted your bot the command **/HUMID**, your bot should return to you the humidity value.

Finally, this is the final circuit:



and [here](https://github.com/The-Assembly/Arduino-IOT/blob/master/Final%20Code/IDE_FINAL_all_/IDE_FINAL_all_.ino) you can access the full code.

**References and Tutorials:**

1. <https://github.com/The-Assembly/Arduino-IOT>