



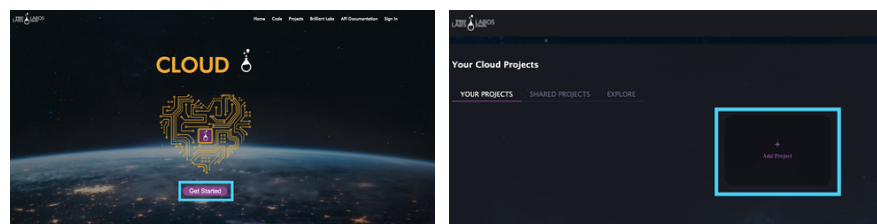
Connecting to the Cloud With your Mars Rover

Connecting to the Brilliant Labs Cloud allows users to have direct point-to-point connectivity. This connection gives Makers a secure connection to create. It's not only a wise cybersecurity service, but it enables Mission:Mars' rovers IoT capabilities. This means, you can control the rover you engineered from your home or anywhere with an internet connection!



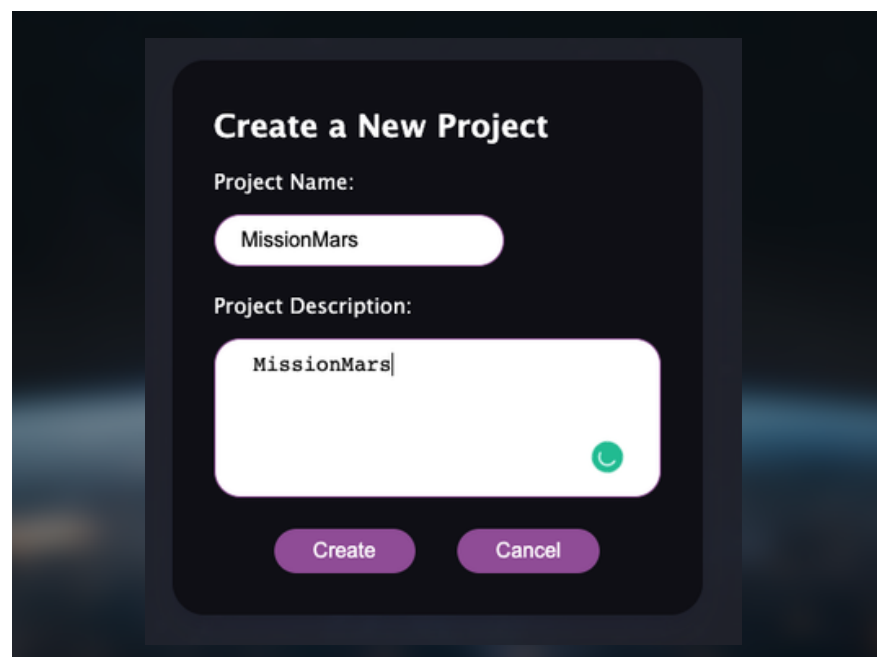
Go To

cloud.brilliantlabs.ca and
select the "Get Started"
button:



Create a new project

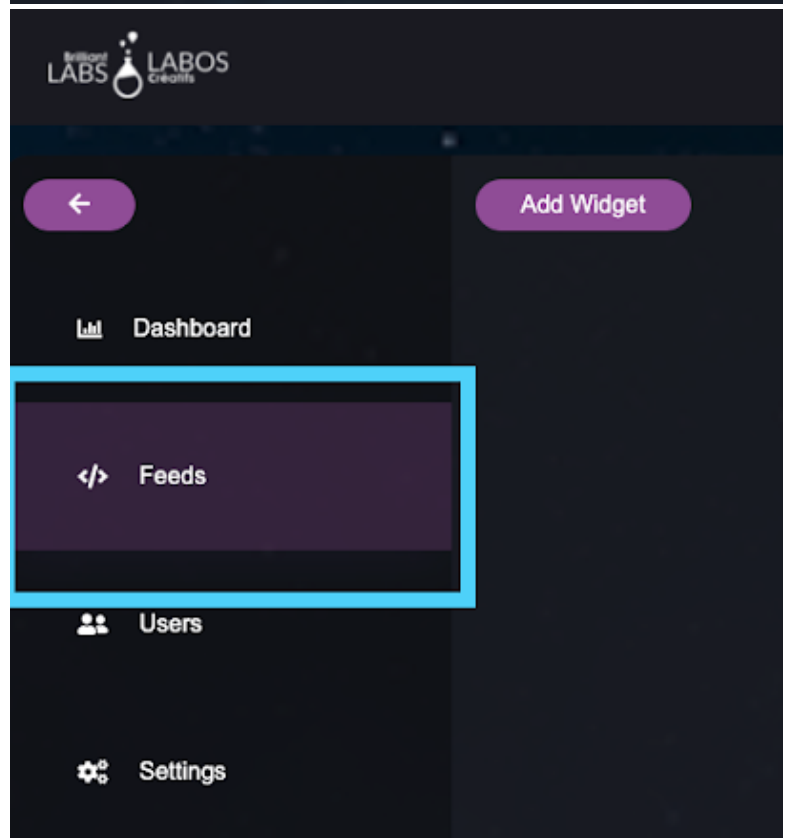
Give your project a name
and description



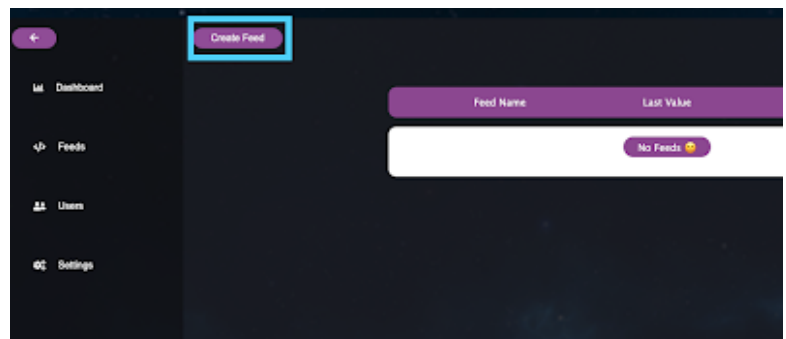
Open your new project



Select the "Feeds" tab

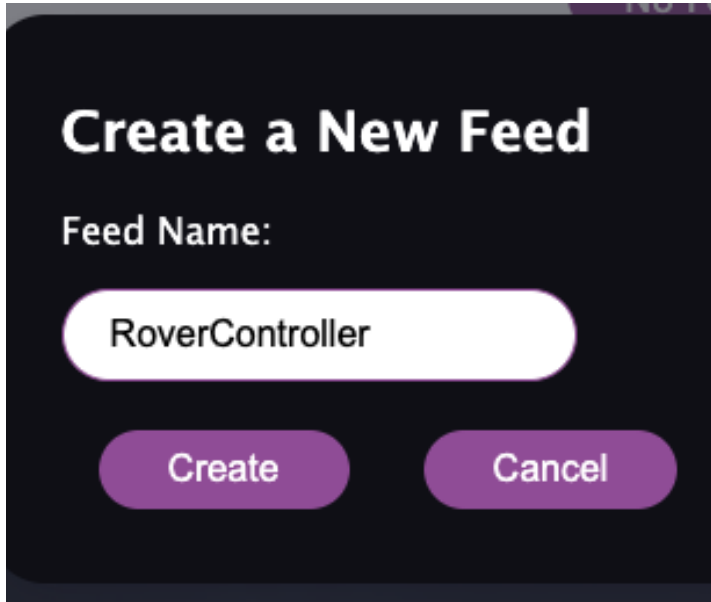


Select "Create Feed"

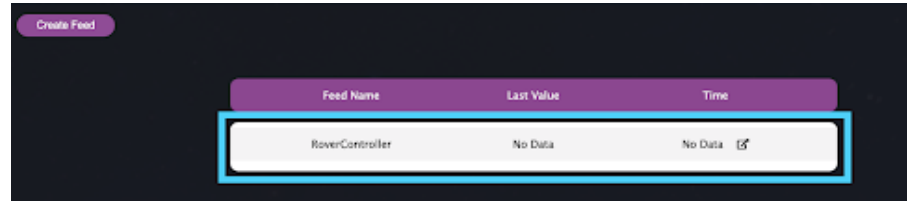


Give your feed any name and select "Create" →

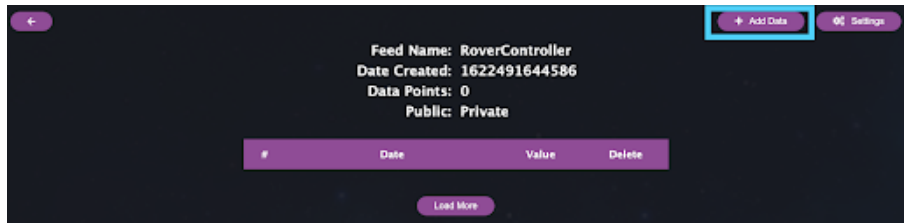
Congratulations! You have successfully created your first feed. A feed is a place where data (numbers like temperature, acceleration or humidity, etc) can be stored. We are going to use this feed to store commands to drive our Rover.



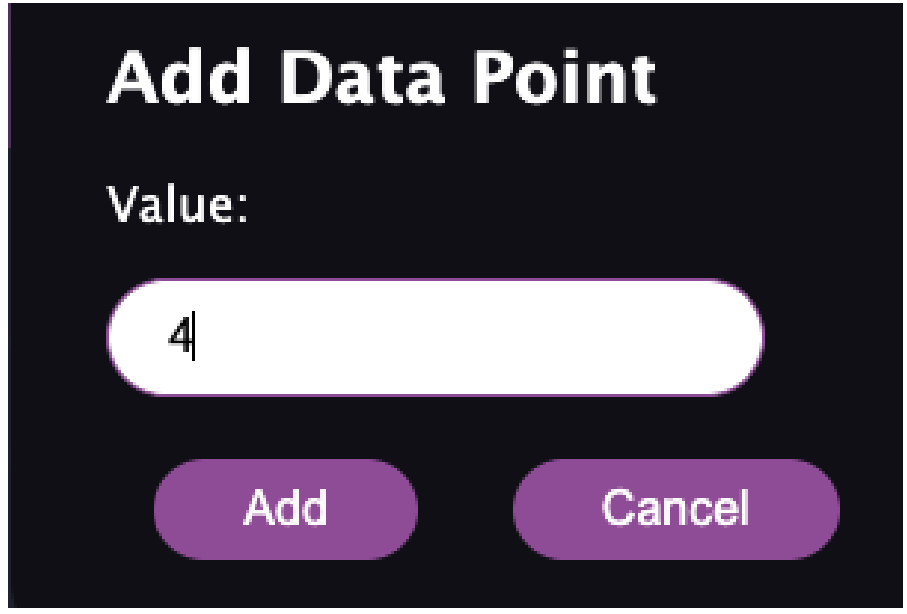
We can add data to our feed one value at a time by clicking on it




Then select the "+ Add Data" button in the top right corner



Here, a value of 4 is being added to our data by entering "4" and selecting "Add"






As you can see, the value has been added to our feed

#	Date	Value	Delete
1	5/31/2021, 5:16:26 PM	4	

Load More

If we do it a few more times, and add some more values, you will see something like this:

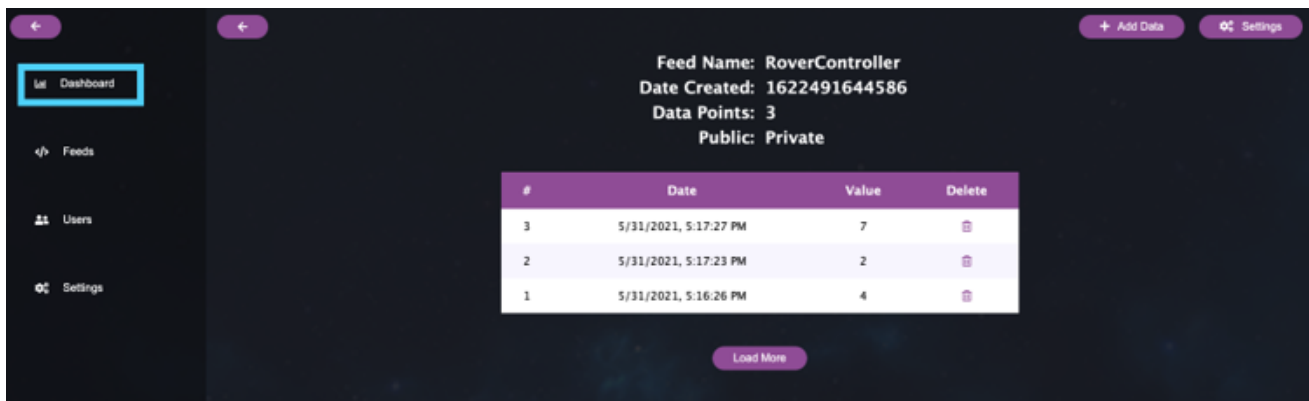
#	Date	Value	Delete
3	5/31/2021, 5:17:27 PM	7	
2	5/31/2021, 5:17:23 PM	2	
1	5/31/2021, 5:16:26 PM	4	

Load More




Each value is stored in the cloud according to when it was sent. Adding data like this can be time-consuming and difficult, especially when trying to use this data to do something like control a Rover, so let's look at an easier way!



Select "Dashboard" on the left side of your screen:



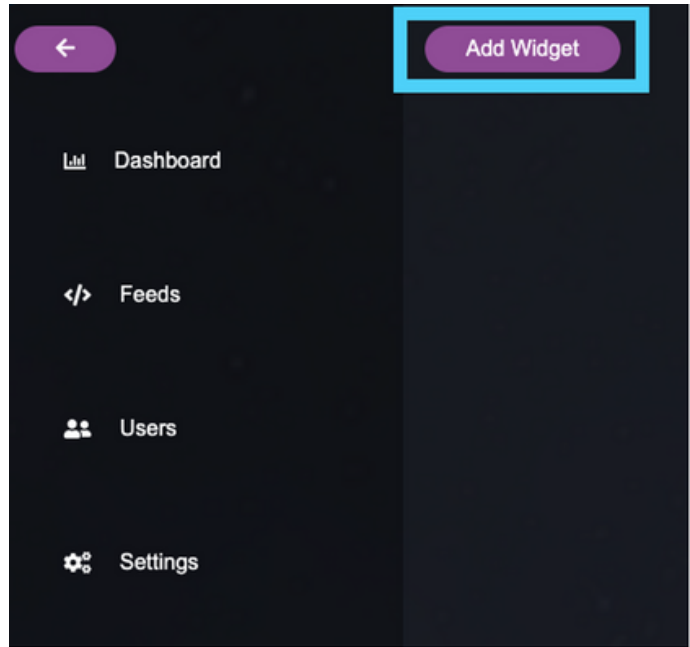
Feed Name: RoverController
Date Created: 1622491644586
Data Points: 3
Public: Private

#	Date	Value	Delete
3	5/31/2021, 5:17:27 PM	7	
2	5/31/2021, 5:17:23 PM	2	
1	5/31/2021, 5:16:26 PM	4	

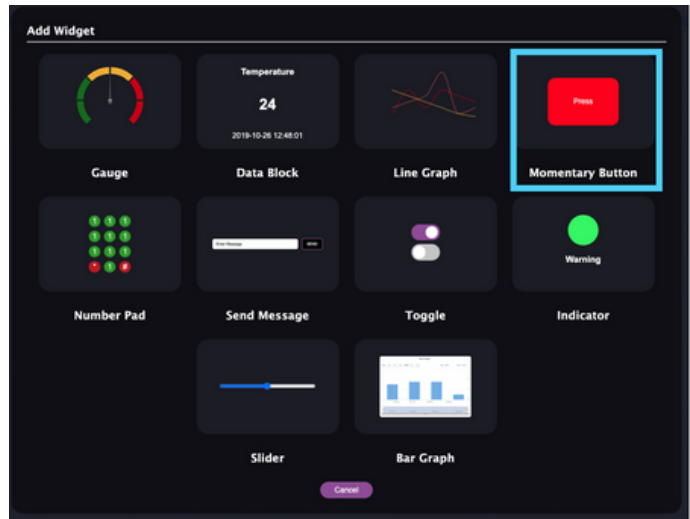
Load More

The dashboard lets us look at our data and even control it with much nicer visuals that are easier to understand. We do this through what are called "Widgets".

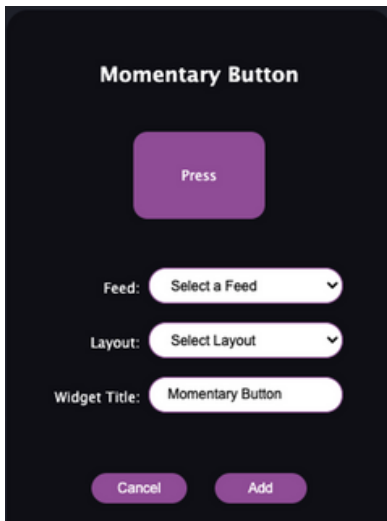
Select "Add Widget" at the top of your screen:



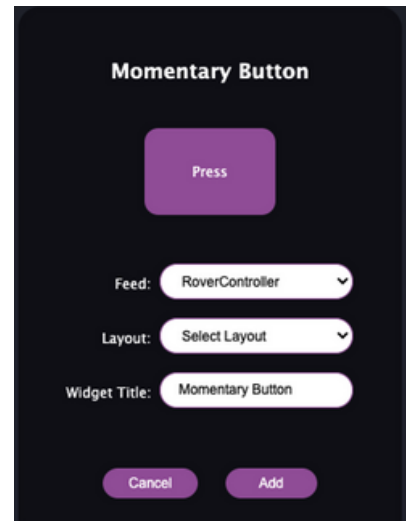
Once open, you will see a variety of great ways you can visualize and control your data. To help us drive our rover, we are going to select the "Momentary Button" on the top right:



Here you will find a few options for setting up your button(s):



The first option is the "Feed". Since we want to use these buttons to control our rover, we are going to select our RoverController feed we just created:



For your button layout, the 1x2x1 layout is recommended, but it's up to you.

Below we've also named our Widget "Rover Remote Control":

We now have 4 parameters we can set for each of the buttons we have:

Button Text: The name of the button. Call it whatever you like.

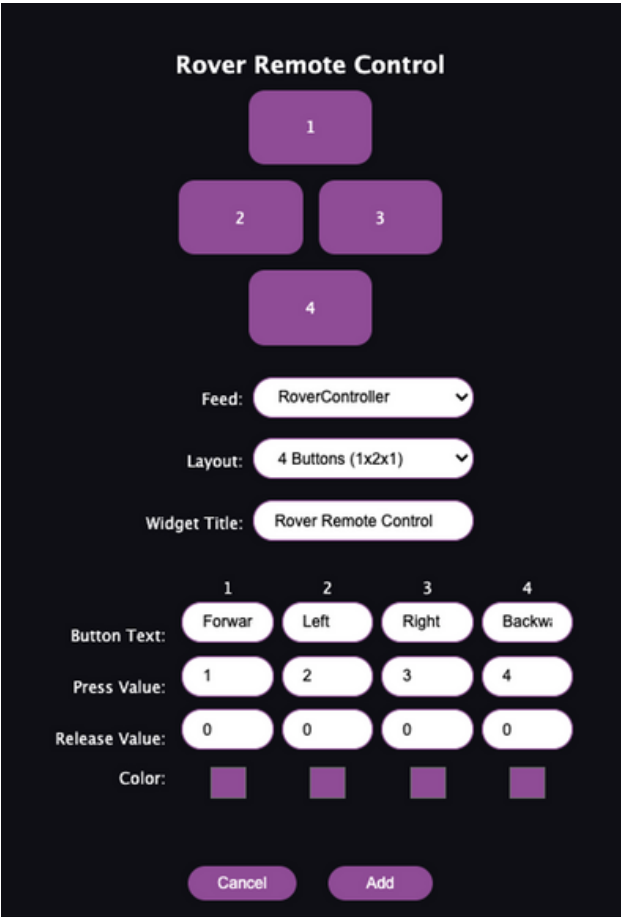
Press Value: The value you want the widget to write to your feed when you press that button

Release Value: The value you want the widget to write to your feed as soon as you let go of that button

Colour: The colour of your button

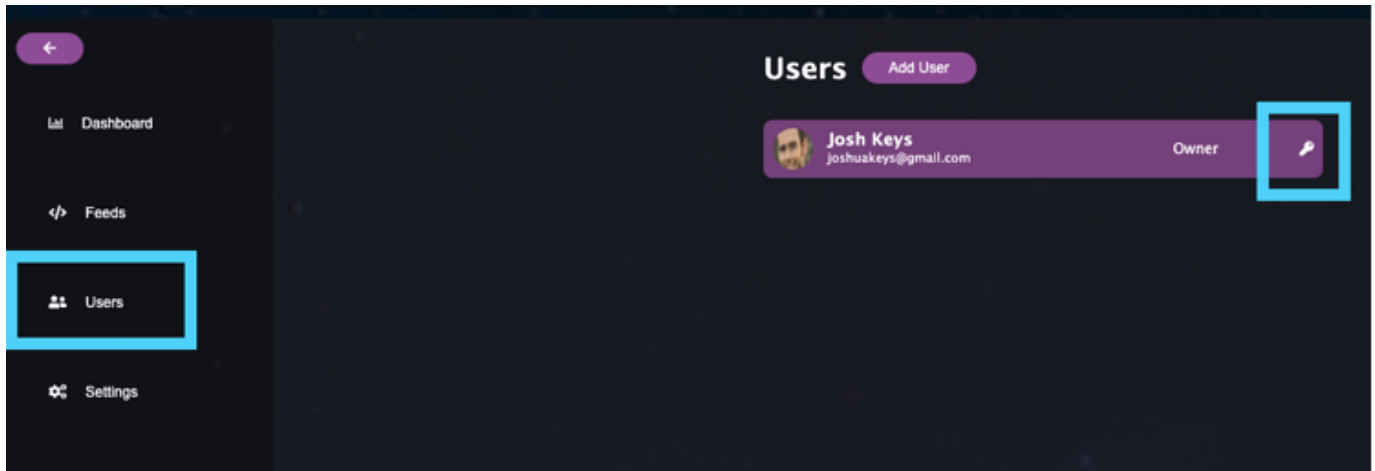
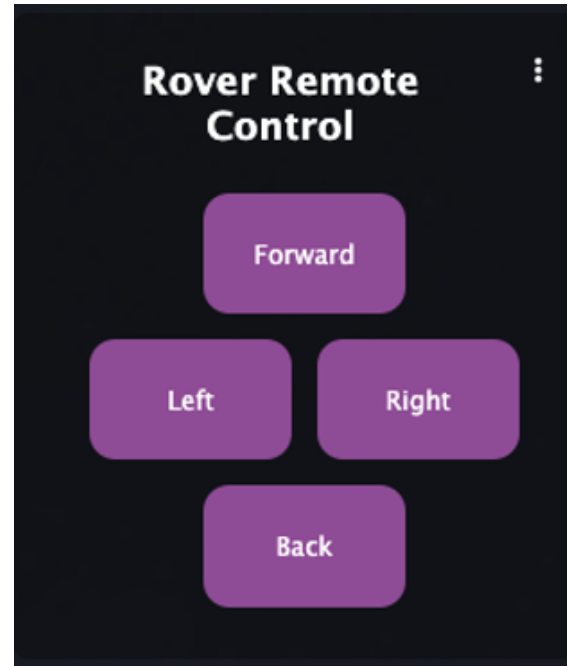
Here is an example setup for controlling our rover: →

When the "Forward" button is pressed, it will write a value of "1" to the Rover Controller Feed. When we decide to let go of the button, it will then write a value of "0". When the "Left" button is pressed, a value of "2" will be written to the RoverController feed. When we let go, it will write 0, and so on.

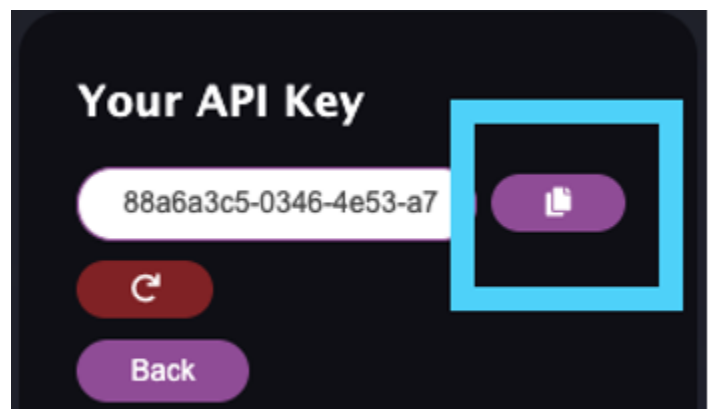


When you're all done setting up your widget, select add and that's it. You now have your WiFi controller built and ready to use by pressing the button directly in your dashboard:

Before we head over to build our code, we are going to need one more piece of information. Even though you have a unique name for your widget and your feed, there are probably a lot of other students using the exact same name. We need a way to distinguish between your widget/feed and everyone else. To do that, every BL Cloud account has a secret password that lets their bBoard connect to their Cloud. We call it an API Key and you can find it by navigating to the "Users" tab and selecting the key icon:

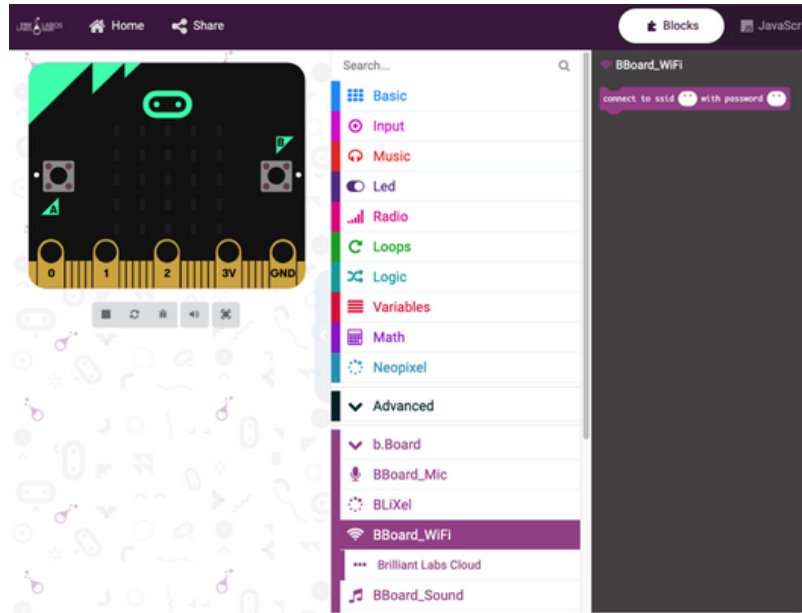


Once selected, a window will open up with your API Key. Press the copy icon as shown below to copy it to your clipboard. Keep this in a secure place as we will need it later.



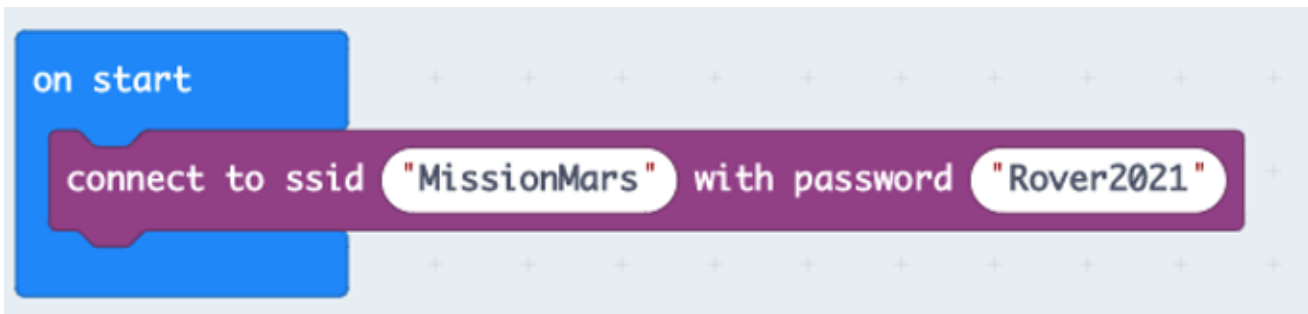
Now it's time to set up the code on the b.Board that is connected to our Rover to move when these buttons are pressed.

Start by going to code.brilliantlabs.ca and opening your rover project. Once in there, navigate to the "b.Board" toolbox and select the "BBoard_WiFi" category:



Select the "connect to ssid.." block and drop it in your "on start" and put your SSID and password in for the internet connection you want your rover to connect to while you are testing it.

Before you ship, make sure to change this to SSID: MissionMars and Password: Rover2021 so it will connect to the network we have set up on Mars.



Now that our rover can connect to the internet, we need it to connect to the Brilliant Labs Cloud to listen for new data from the feed we previously created. To do this, select the "BBoard_WiFi" category followed by the "Brilliant Labs Cloud" subcategory below it.



There are 2 blocks in here for controlling your cloud data:



The publish block is how you send data from the b.Board to your BL Cloud Feed. In this example, we are sending data to a pretend feed called "ExampleFeed" using a username "brilliantlabs@gmail.com" and API Key



There is a lot going on in this block, so let's break it down:



Whenever there is new data on your feed (RoverController in this example), it is written

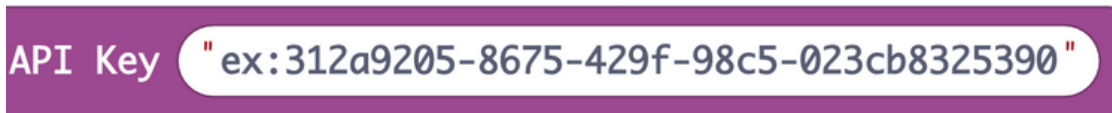
the variable "receivedData". You can drag that red block out to place in your code. You can also let your code know if you plan on sending text to your feed "String", or numbers "Number" with the drop down menu.

This is the name of the feed you created earlier to store all of the commands (numbers) sent when you press on the widget buttons.



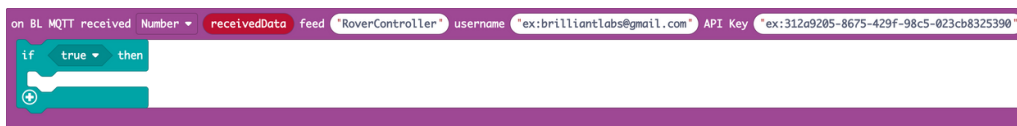
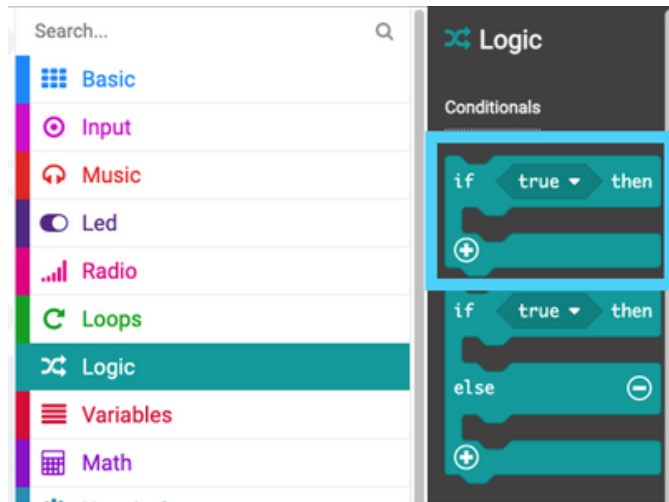
Put the username (your email address) that you use to connect to the cloud.brilliantlabs.ca with

Remember all of those numbers and letters you saved earlier on your cloud? That is a special password or API Key that lets your b.Board write and read data to any feeds in the MissionMars project you created.

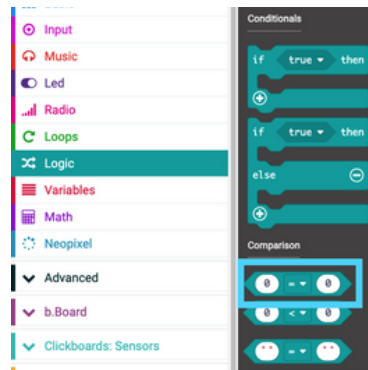


Your rover is now ready to connect to the internet and listen for any new data that is sent to your RoverFeed feed. When a new message is sent, the "on BL MQTT" block we just looked at above will run, but we have no code in it! Let's take a look at what code we can put inside the "on BL MQTT" to control our rover whenever new data arrives.

We first need to apply logic to our rover so it can decide what to do depending on what data it receives from the RoverController feed. To do that, grab the "If true then" block and drop it into your MQTT block.



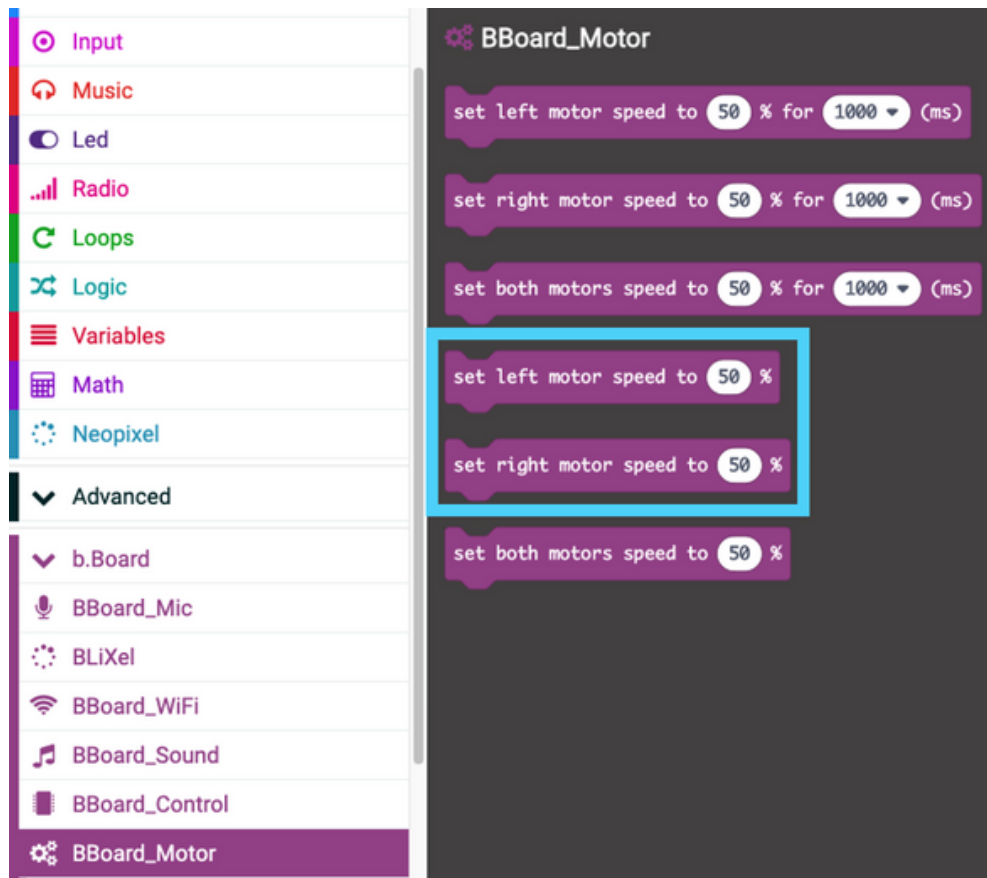
Next, grab the "comparison" block and drop it into the "if" block you just added.



And now take the "receivedData" variable and drag it into your comparison block:



Your rover now has its first piece of cloud logic code. It says "If the message we just received from the cloud (receivedData) is equal to 0, then run the code below". If you recall, we told our Rover Controller widget to send a 0 every time a button was let go, so let's program our rover to stop every time a 0 is received. To do this, navigate to the BBoard_Motor category and grab the "set left motor speed to .." and "set right motor speed to .." blocks, put them inside our "if" block and set their speed to 0 to stop it:



To finish up our controller, duplicate your “if” block and cover each condition for a value that the RoverController feed can send. In the example below, we’ve provided code for having the motors go forward, backward, left, right and stopping according to the values that our widgets sent when the buttons were pressed. Feel free to use this same code to drive your rover, or do something completely different, it’s up to you!

```
on BL MQTT received Number receivedData feed "RoverController" username "ex:brilliantlabs@gmail.com" API Key "ex:312a9205-8675-429f-98c5-023cb8325390"
if receivedData = 0 then
  set left motor speed to 0 %
  set right motor speed to 0 %
+
if receivedData = 1 then
  set left motor speed to 100 %
  set right motor speed to 100 %
+
if receivedData = 2 then
  set left motor speed to -100 %
  set right motor speed to 100 %
+
if receivedData = 3 then
  set left motor speed to 100 %
  set right motor speed to -100 %
+
if receivedData = 4 then
  set left motor speed to -100 %
  set right motor speed to -100 %
+
```

Questions?

If you need a little more help or want to schedule a professional learning session let's connect! We're here to help. [Email Neil@brilliantlabs.ca](mailto:Neil@brilliantlabs.ca)

Join the Mission: Mars Challenge!

Are you on track to complete your Mission: Mars Rover by early May? If so, Brilliant Labs welcomes you to visit or mail your rover to 1 of 5 in-person Provincial School Maker Faires or the Atlantic Virtual Mission: Mars Challenge (June 2nd). This is your chance to showcase your work and participate in up to 10 mission challenges. Each challenge, when completed successfully, will earn points and badges. The Mission: Mars student engineers with the most points will win the Mission: Mars Challenge Showcase!

Download, Register & Book Today!

Get the Mission: Mars Challenge Guide to learn more about what to expect and how the points will be awarded. Plus, don't forget to register and book your Mission Challenge(s) at Brilliantlabs.ca/mission-mars (winners will be announced at the June 9th Atlantic Virtual School Maker Faire). Join the challenge and explore Maker Mars!

