breadware

Lab 1: Introductory Project to Breadware

Exploration of Breadware's IoT Development Tools

Overview

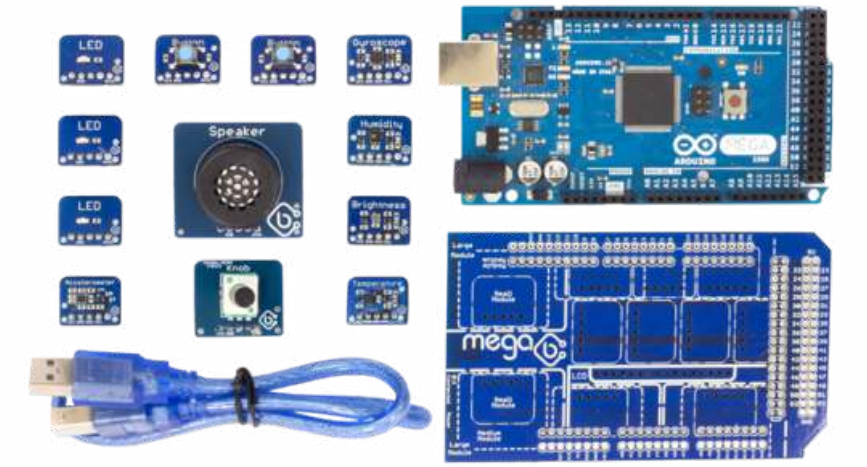
The goal of this lab is to become familiar with the Internet of Things prototyping tools available in the Breadware Development Toolkit. Specifically, you will experiment with the Mega-B Development Kit, which includes an Arduino Mega, a Mega-B Bread™Board, and 13 prototyping modules ranging from environmental sensors to manual inputs. Next, you will become familiar and oriented with the Breadware IDE, which allows you to modularly program your prototype, develop a custom mobile application, and establish a dashboard to stream and analyze your data.

To get started prototyping with the Breadware platform, you will need to acquire a Mega-B Bread™Board and Breadware modules.

Prerequisites/Materials required:

☐ **Hardware**

- Arduino Mega or Nano processor
- Breadware modules
- Mega-B Bread™Board
- USB 2.0 cable Type A Male to Type B Male
- An iOS device if using the Bread™Connect App or Breadware Dashboard



Breadware Hardware Descriptions

- **Mega-B Bread™Board**: A powerful modular prototyping shield for the Arduino Mega.
- **Accelerometer:** The Accelerometer module is suitable for monitoring motion, accelerations, and tilt.
- **Brightness:** The Brightness module is suitable for lux monitoring systems.
- **Button:** The Button module is a small tactile button switch to enable simple manual interaction with your device.
- **Gyroscope:** The Gyroscope module is suitable for monitoring rotation on the X, Y, and Z axes.
- **Humidity:** The Humidity module is suitable for building connected weather and humidity indicators.
- **Knob:** The Knob module will allow you to set a value on a continuum within your device.
- **LED:** The RGB LED (Light Emitting Diode) module can add colored light indicators to a project.
- **Microphone:** The Microphone module is suitable for performing wireless sound detection.
- **Speaker:** The Speaker module generates alert tones at various frequencies.
- **Temperature:** The Temperature module is suitable for building connected thermometer applications

☐ **Breadware IDE**

1. Visit https://dev.breadware.com
2. Register for a free student Breadware account
3. Log in to create, save, and edit your projects as well as view your dashboards.

☐ **Download the Bread™Connect App Software**

Currently only supported on iOS. You can find the download here.
https://itunes.apple.com/us/app/bread-connect/id1172516219?mt=8

Log into the Bread™Connect App with the same account information you used to log into the Breadware IDE.

☐ **Download the Breadware Flasher**

If the Breadware Flasher is not already installed on your computer, you will need to download it from the Breadware site.

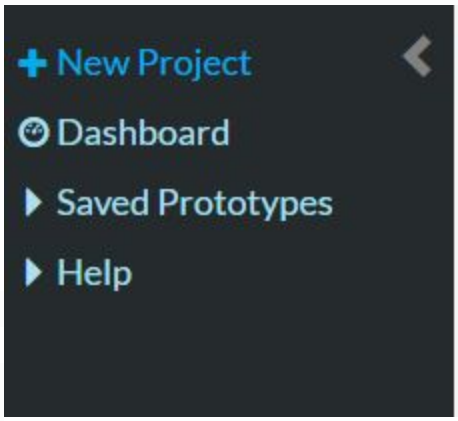
1. Visit docs.breadware.com
2. Click on “Downloads” in the directory on the left-hand side of the page
3. Download and install the Breadware Flasher for either Windows or macOS
4. Log into the Breadware Flasher with the same account information you used to log into the Breadware IDE.

Part One: Building Your First Project

This guide will walk you through the basics of building an IoT device using the Breadware IDE by getting a simple project running on a Bread™Board. This first project will be enabling a **button-controlled LED.** The tutorial will guide you through using the Breadware IDE to turn on a LED when a button is pressed using a BreadBoard.

1a. Create a New Project

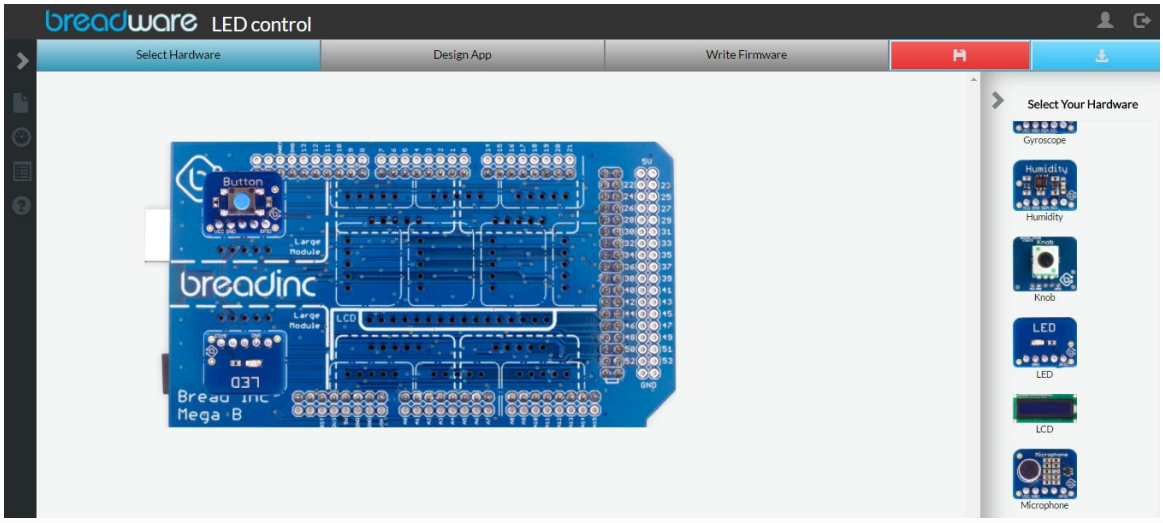
Open the Breadware IDE at https://dev.breadware.com and sign in/register with your account if you have not already done so. Start a new project by expanding the left side navigation bar (if not already expanded) and clicking the "New Project" button.



1. Begin by clicking "New Project" and give your project a name.
2. Select the type of Bread™Board you are using. **Note:** The code generated using the Breadware IDE will only work on the Bread™Board type you selected.
3. Choose a pre-existing template if desired. For this walkthrough you should choose a blank template to start from scratch.

1b. Adding Hardware Using the Hardware Builder

Navigate to your project's hardware builder by clicking "Select Hardware" in the navigation bar at the top of the page (under your project's name).

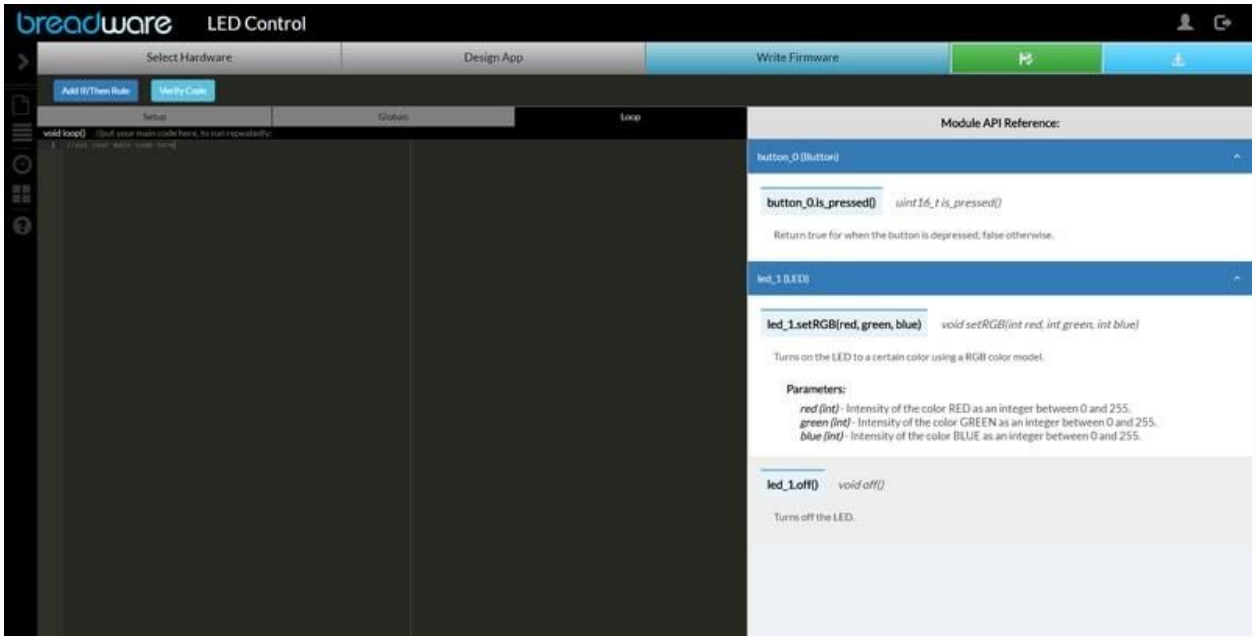


1. Open the hardware module menu on the right side of the page (if not already opened).
2. Add a button onto the Bread™Board by dragging and dropping the button module into its desired position. **Note:** Wherever you place the Breadware module in the Breadware IDE must match where you place it on the physical Bread™Board.

3. When you drop the Breadware module onto the Bread™Board you must also give it a name. The name will be used to write the program for the board, so give it a name that you will understand. **Note:** If you don't care about the name just press enter through the popup and a generated name will be provided.
4. Click "Create" to finish adding the Breadware module onto the Bread™Board.
5. Repeat steps 1-4, but instead of a button add an LED to another open module slot on the Bread™Board.

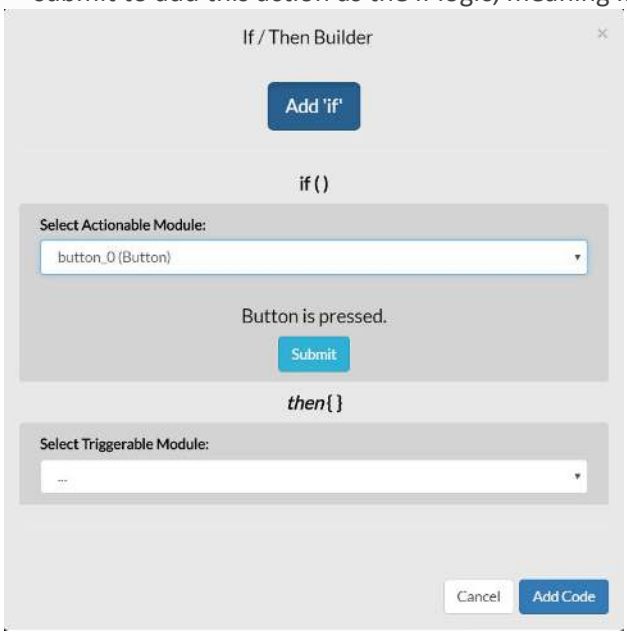
1c. Writing Code Using the Firmware Editor

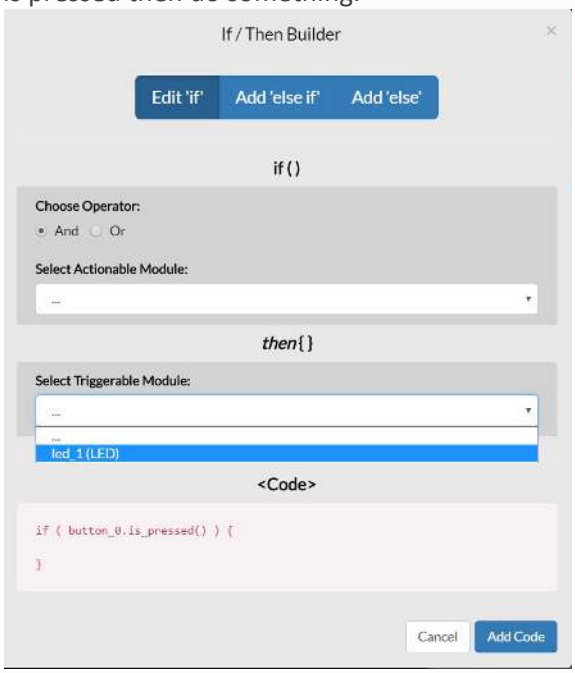
Navigate to your project's firmware editor by clicking "Write Firmware" in the navigation bar at the top of the page. This is where you will begin to write the program that will run on your Bread™Board. If you are familiar with C/C++ programming, you can jump directly into writing the code by using the module functions documentation on the right-hand side of the screen for reference. If you are new to C/C++ programming or not sure where to start, follow the next steps.



1. Click on the **"If/Then Builder**" at the top of the firmware editor page to begin. The "If/Then Builder" is used to generate desired code using if-then building blocks.
2. The "If/Then Builder" uses a logical statement to describe what the device should do (if blank is met, then do something). Begin by selecting the actionable module as the button.

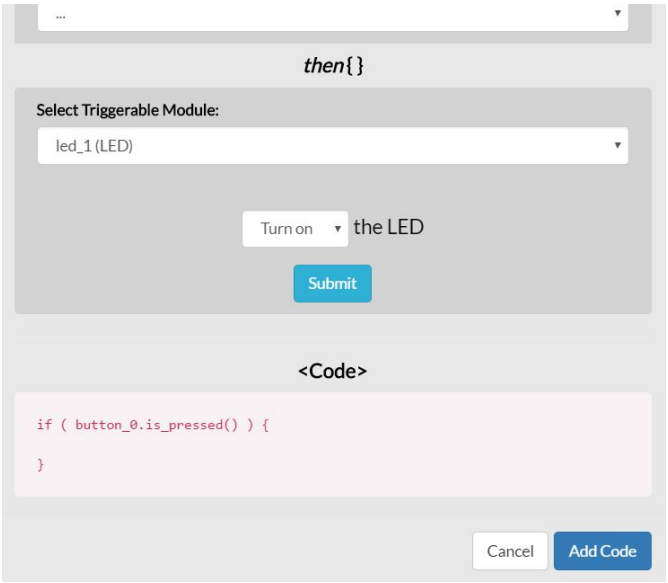
3. The button module has two states, or actions, button **is pressed** and **not pressed**. Select "**Button is pressed**". Click submit to add this action as the if logic, meaning if the button is pressed then do something.





4. Next, **set the triggerable module** as the LED.
5. The LED module has two actions; **on** and **off**. Select the "**on**" action and the color you wish to turn the LED on. Click **submit** to add this action as the then logic, meaning turn on the LED if the if condition in step 3 is met.
6. After clicking submit for both the If/Then rules, you should now see a preview of your code at the bottom of the popup. It should be similar to the code below, if so click the "Add Code" button to add the code to your project. The numbers inside the setRGB function may be different if you selected a different color than blue.

```
if ( button_0.is_pressed() ) {
led_1.setRGB(0,0,255);
}
```

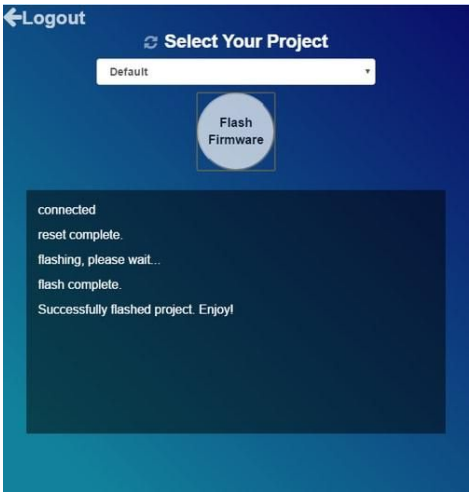


1d. Flashing Code onto a Bread™Board

The last step is to flash (upload) the code you wrote using the firmware editor onto your Bread™Board. The following steps will show you how to flash code using the Breadware Firmware Flasher desktop application. If you have issues with the Breadware Firmware Flasher, the Breadware IDE is also compatible with the Arduino IDE.

1. First **save your project** as this is necessary before you're allowed to compile and flash your project.
2. **Verify your firmware code is valid** by clicking the "Verify Code" button in the firmware editor. If there's an error, please look at your firmware code closely and fix the error before continuing.

3. After saving and verifying your project, **open the Breadware Firmware Flasher**.
4. If you haven't done so already, **Log into your account** using your Breadware credentials.
5. **Connect the Bread™Board** to a USB port on your machine.
6. **Select the project** you want to flash onto the Bread™Board.
7. Click the **"Flash Firmware"** button on the desktop application to begin flashing your code onto the Bread™Board.
8. Congratulations! You just programmed your Bread™Board with your first project. Make sure the **Breadware modules** that you used in this project (button and LED) are on the physical board in the positions that you placed them in the "Select Hardware" page. Verify the code works by clicking the button to turn on the LED.



---

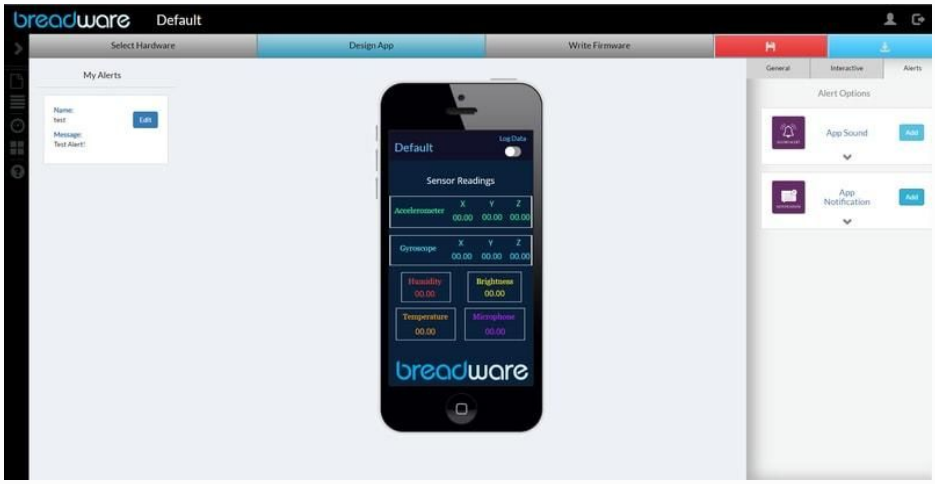
continued on next page (8)

Part Two: Mobile App Controlled LED

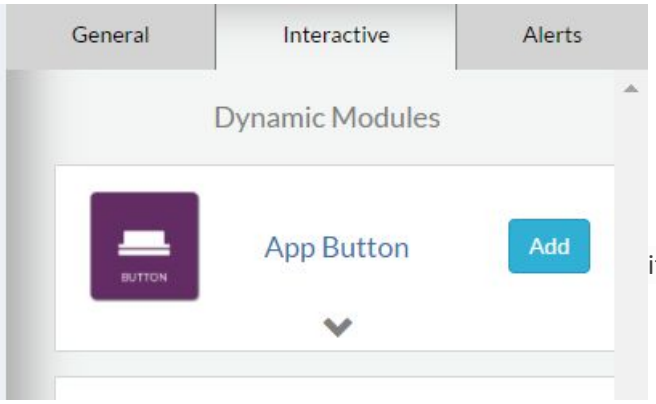
The second part of this tutorial continues to build on top of the button-controlled LED project built in part one. This section will guide you through using the **mobile app builder** in the Breadware IDE to control your LED. Your hardware button already turns on the LED, so now we will create a remote interface with the device in your mobile app that will turn off the LED.

2a. Using the Mobile App Builder

Navigate to your project's mobile app builder by clicking "Design App" in the navigation bar at the top of the page (under your project's name).

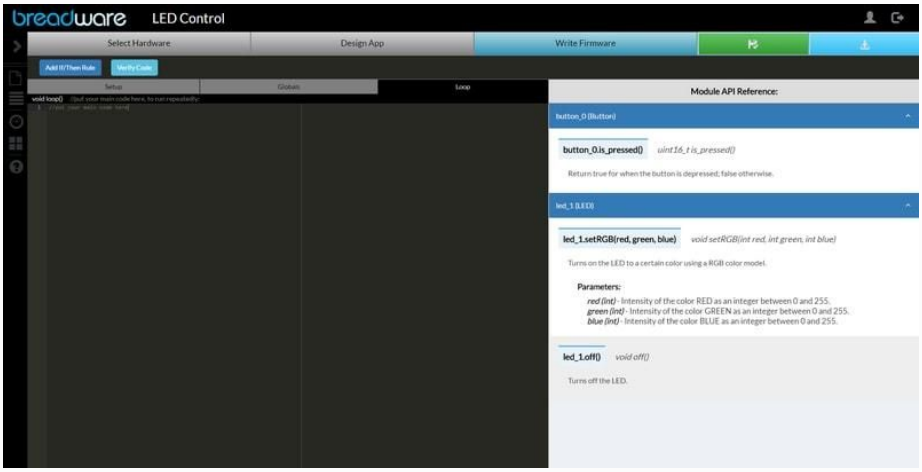


1. **Open the mobile module menu on the right side** of the page (if not already opened).
2. **Select the "Interactive" tab** in the menu and locate the mobile app button module.
3. **Add a mobile button** onto the Bread™Board by clicking the "Add" button for the mobile button module and give it a name if desired. ***Note:** By default the mobile app modules are added to the center of the mobile app device, but you can move the position of the module by clicking and dragging the module to a new location.*

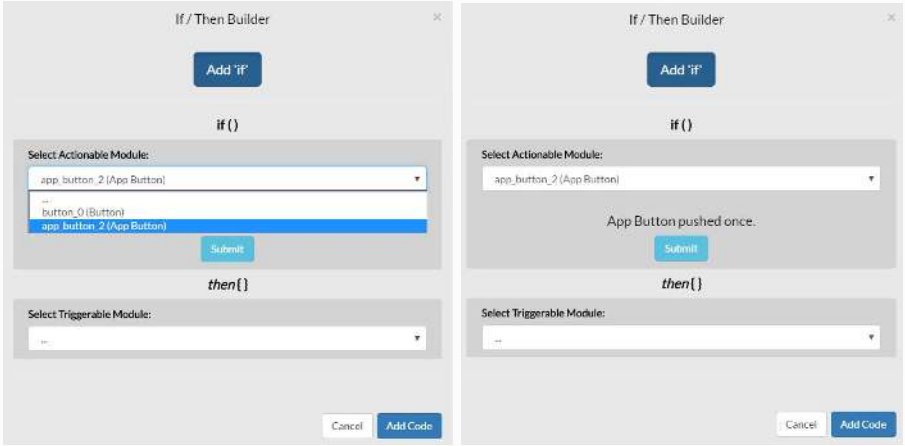


2b. Writing Code Using the Firmware Editor

Navigate to your project's firmware editor by clicking **"Write Firmware"** in the navigation bar at the top of the page. This is where you will begin to write the program that will run on your Bread™Board. If you are familiar with C/C++ programming, you can jump directly into writing the code by using the module functions documentation on the right-hand side of the screen for reference. If you are new to C/C++ programming or not sure where to start, follow the next steps.



1. Click on the **"If/Then Builder"** at the top of the firmware editor page to begin. The "If/Then Builder" is used to generate desired code using if-then building blocks.
2. The "If/Then Builder" uses a logical statement to describe what the device should do (if blank is met, then do something). Begin by **selecting the actionable module as the app button.**
3. The mobile app button module has only one action and it should be **"App Button pushed once"**. Click submit to add this action as the if logic, meaning if the mobile app button is pressed then do the then action.



4. Next **set the triggerable module as the LED.**

5. The LED module has two actions, on and off, **select "off".** Click submit to add this action as the then logic, meaning turn off the LED if the if condition in step 3 is met.

6. After clicking submit for both the If/Then rules, you should now see a preview of your code at the bottom of the popup. It should be similar to the code below, if so **click the "Add Code"** button to add the code to your project.

```
if ( app_button_2.read() ) {
  led_1.off();
}
```

7. Your firmware code should now look like the code below if you started from the beginning of the tutorial.

```
if ( button_0.is_pressed() ) {
  led_1.setRGB(0,0,255);
}

if ( app_button_2.read() ) {
  led_1.off();
}
```

2c. Testing Your Code

1. **Follow the same steps as part 1** of the tutorial to flash the firmware code onto a BreadBoard.
2. To use the mobile app you created using the Breadware IDE, **open the Bread™Connect** mobile app and log in with your Breadware credentials.
3. Verify you have Bluetooth enabled on your mobile phone. **Connect to your Bread™Board** by clicking on your device in the list of Bluetooth devices.
4. Test out your completed project by using the hardware button to turn on the LED and the button on the mobile app to turn off the LED.
5. **Congratulations! You just made a mobile app to interact with your Breadware hardware.** Try changing up your code to turn the LED on using a custom color by directly editing the code instead of using the If/Then builder (See the LED module references on the right). Or instead of using the mobile app button to turn off the LED, try using the hardware button when it's not pressed by using the "else" statement in your code.