Arduino Setup Lab

Introduction

For this first lab, you will be experimenting with your Arduino from your Lafvin Super Starter Kit that you purchased from the Bookstore. If you haven't been able to get a kit, please message your instructor.

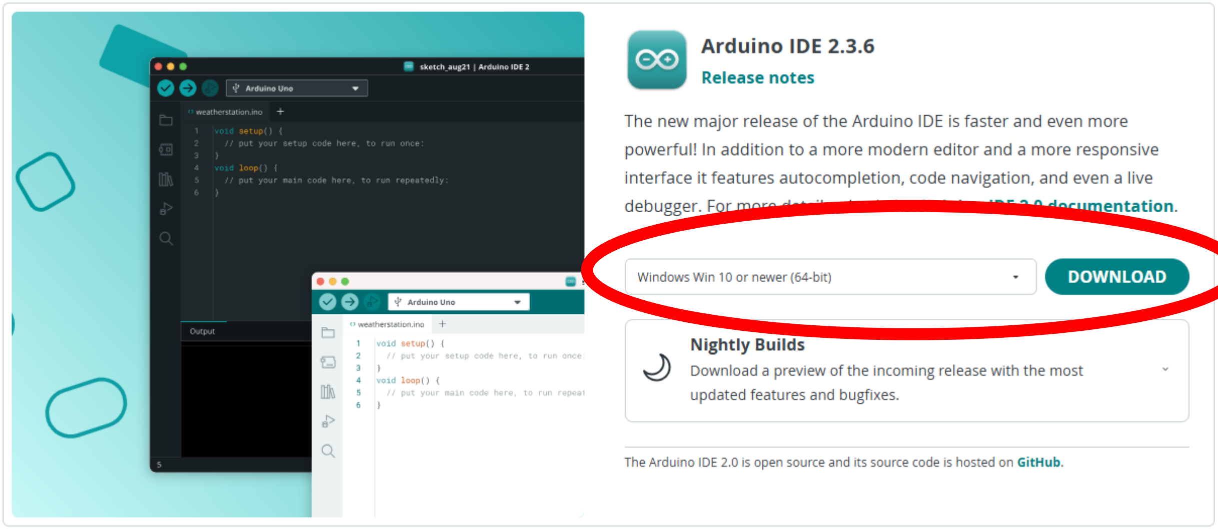
You will be learning how to connect the Arduino to your computer, how to upload code (a sketch) to it, how to use some of the parts from your kit, and how to understand and edit some of the code.

After installing the correct program onto your laptop (Part 1) and verifying the connection to your Arduino (Part 2), this lab has three main parts (Parts 3, 4, and 5). You will connect a button to your Arduino and turn on a light when it is pressed (Part 3). Then you will connect a second button and use it together with the first to accomplish the same task, altering some code logic (Part 4). Finally, you will add in a buzzer that will activate once both buttons are pressed (Part 5). After completing these steps, you will submit a photo of your circuit to I-Learn.

If you run into problems, you can follow along with this TA walkthrough video. You can also review the troubleshooting tips at the end of the instructions, and you may always ask your instructor for help.

Part 1: The Arduino IDE

Please install the Arduino IDE to your computer. The Arduino IDE is an application that allows you to write code on your laptop and then upload that code onto your Arduino. Follow the link to the Arduino website and select the appropriate download for your laptop.



sketch_aug21 | Arduino IDE 2

Arduino Uno

weatherstation.ino

```
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
}
```

Arduino IDE 2.3.6

Release notes

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger. For more d IDE 2.0 documentation.

Windows Win 10 or newer (64-bit) | DOWNLOAD

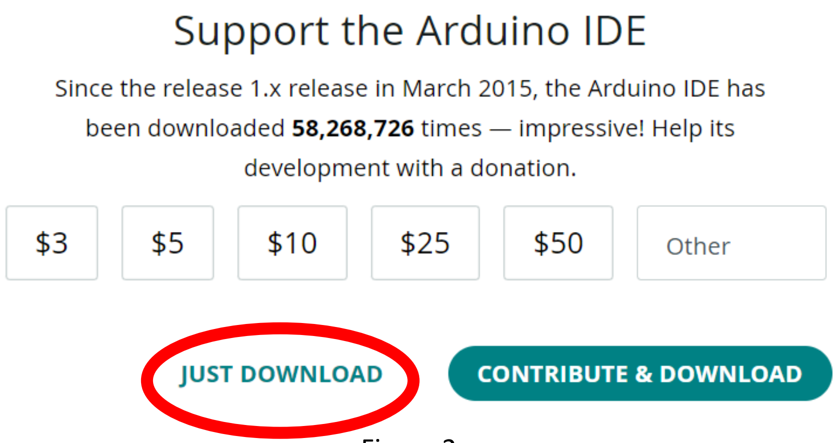
Nightly Builds

Download a preview of the incoming release with the most updated features and bugfixes.

The Arduino IDE 2.0 is open source and its source code is hosted on GitHub.

Figure 1

Once you've selected the download type, click on the "DOWNLOAD" button to move forward with downloading the software. It may or may not ask you do donate – you do *not* need to donate.



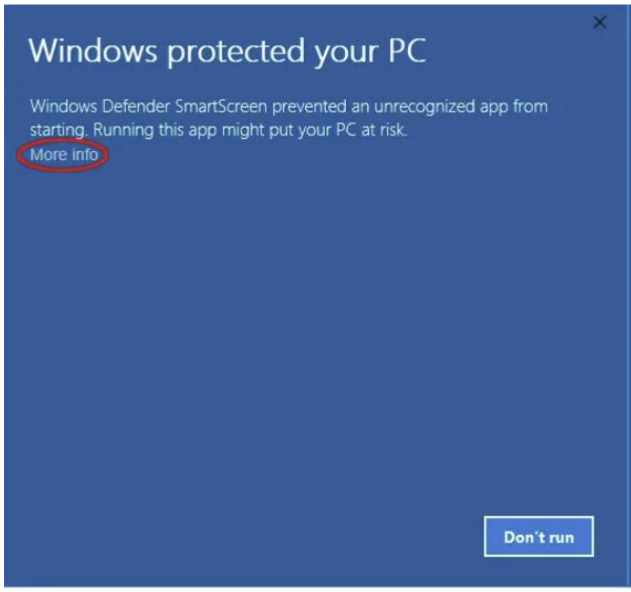
Support the Arduino IDE

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **58,268,726** times — impressive! Help its development with a donation.

Figure 2

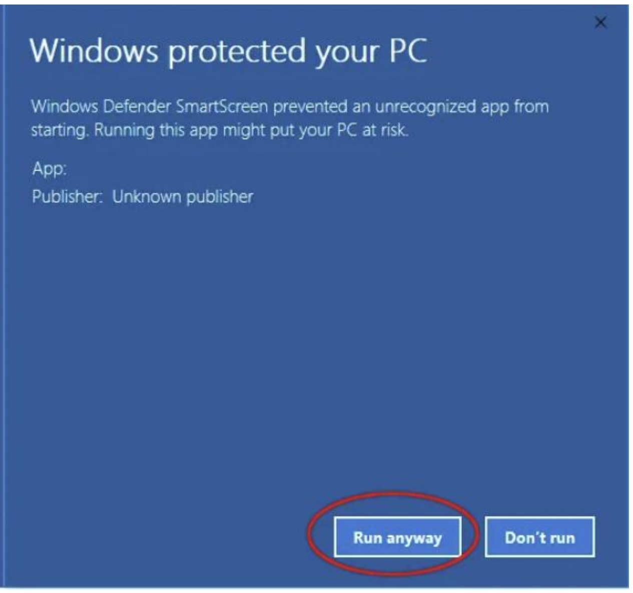
The program may go to your Downloads directory. Once it downloads, double click on the `arduino-ide_2.0.3_Windows_64bit.exe` file (the version number may have changed) to install. (For a Mac, it will be similar to installing other programs on a Mac.) Installation may take a few minutes.

If you see a Windows popup that tries to stop you from installing the program, click "More info" (Figure 3) and then "Run Anyway" (Figure 4) on the popup to continue with the installation.



Windows protected your PC

Windows Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.



Windows protected your PC

Windows Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

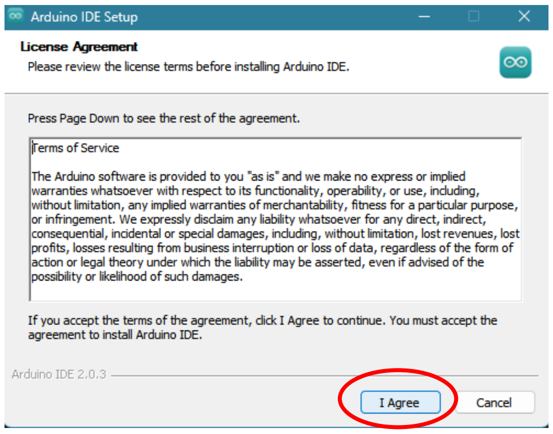
App:

Publisher: Unknown publisher

Figure 3

Figure 4

Once you start the actual installation process, you will just have to click through the options. You shouldn't need to change anything in here unless you want to change the directory for your program (Figure 7). Note: The directory path shouldn't include spaces or non-English characters. Otherwise, just click through, and then click "Finish" to complete the installation (Figures 5-8).


Arduino IDE Setup

License Agreement

Please review the license terms before installing Arduino IDE.

Press Page Down to see the rest of the agreement.

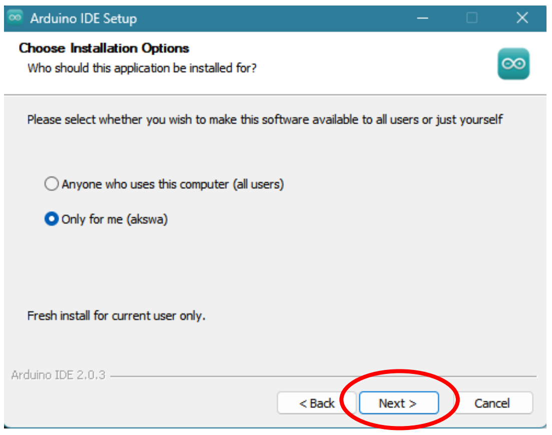
Terms of Service

The Arduino software is provided to you "as is" and we make no express or implied warranties whatsoever with respect to its functionality, operability, or use, including, without limitation, any implied warranties of merchantability, fitness for a particular purpose, or infringement. We expressly disclaim any liability whatsoever for any direct, indirect, consequential, incidental or special damages, including, without limitation, lost revenues, lost profits, losses resulting from business interruption or loss of data, regardless of the form of action or legal theory under which the liability may be asserted, even if advised of the possibility or likelihood of such damages.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Arduino IDE.

Arduino IDE 2.0.3

Figure 5


Arduino IDE Setup

Choose Installation Options

Who should this application be installed for?

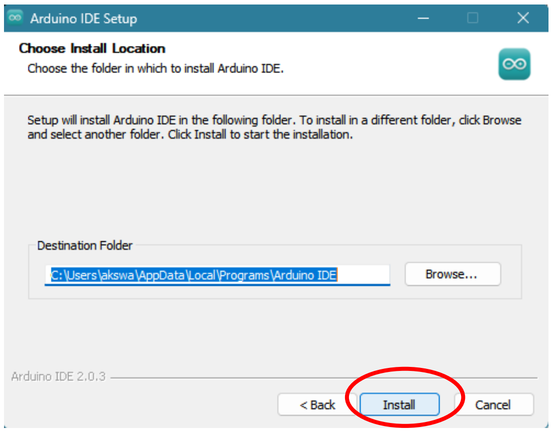
Please select whether you wish to make this software available to all users or just yourself

Anyone who uses this computer (all users) [✓ Only for me (akswa)]

Fresh install for current user only.

Arduino IDE 2.0.3

Figure 6


Arduino IDE Setup

Choose Install Location

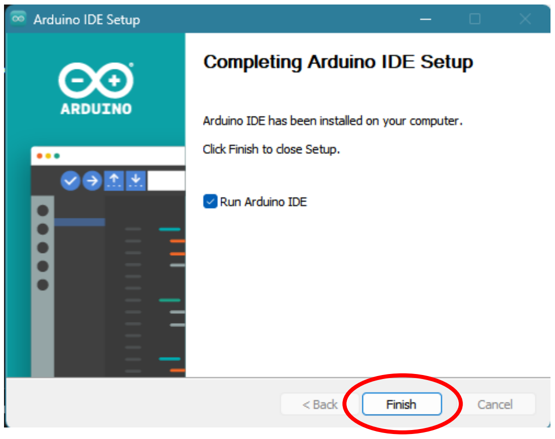
Choose the folder in which to install Arduino IDE.

Setup will install Arduino IDE in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder: C:\Users\akswa\AppData\Local\Programs\Arduino IDE

Arduino IDE 2.0.3

Figure 7


Arduino IDE Setup

Completing Arduino IDE Setup

Arduino IDE has been installed on your computer.

Click Finish to close Setup.

[✓ Run Arduino IDE]

Figure 8

Part 2: Connect and Verify with Example Code ("Blink")

The first thing that you should do for this part is to make sure that your Arduino is connected to your computer with the provided USB cable from your Lafvin Kit (see Figure 9). The USB cable in your kit uses the older USB-A standard. If your laptop doesn't have any older USB-A ports and only has the newer USB-C ports, you will need to purchase an adapter such as the black adapter shown in Figure 10.

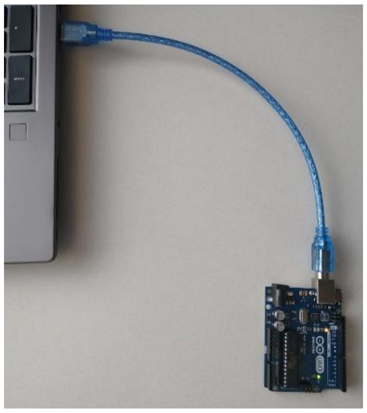


Figure 9

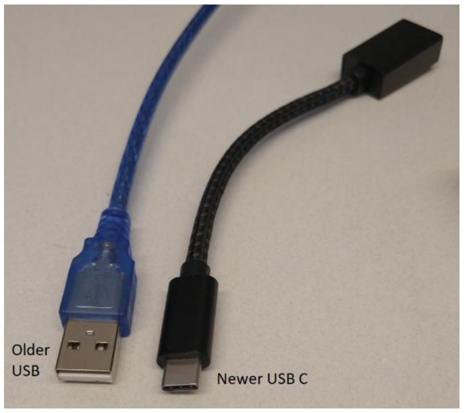


Figure 10

Once you have opened the Arduino IDE program, connect your Arduino board to the IDE by selecting the correct port in the "Select Board" dropdown menu, as shown in the screenshot below. It should be the port that says "Arduino Uno." (The "COM" number may change.)

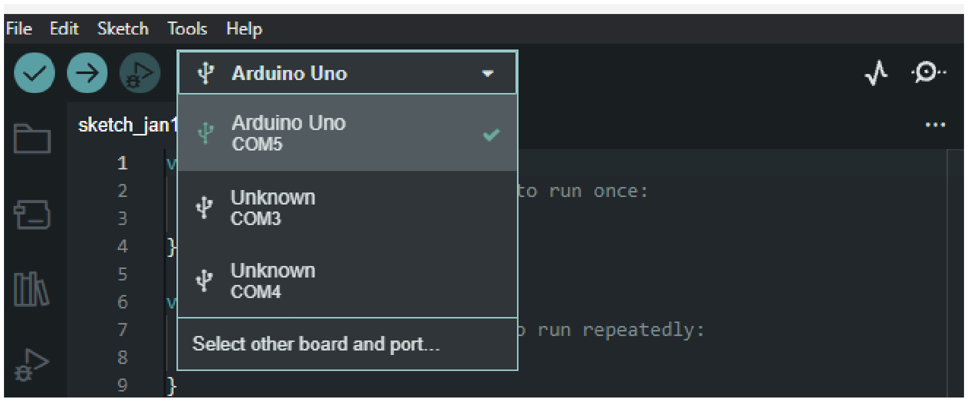


Figure 11

Note: If this step or the next steps don't work, see the end of this file for Troubleshooting Tips.

Next, for your first experiment on the Arduino, you'll be running a program called "Blink". Navigate to File in the taskbar, then go to "Examples", then "01.Basics," then select the "Blink" program.

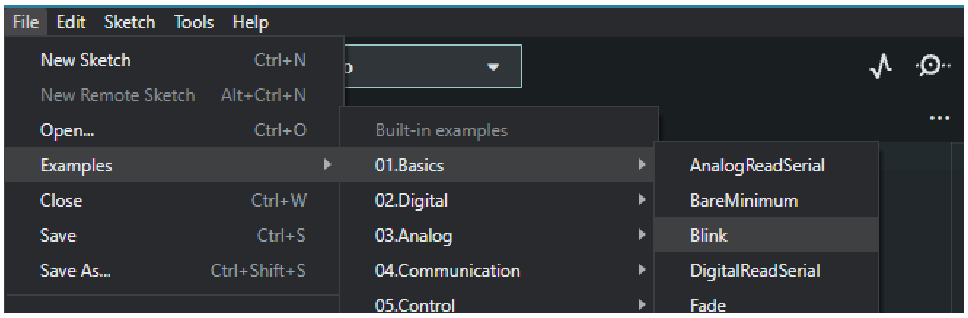


Figure 12

Once you have the Blink sketch open, upload it to your Arduino by clicking the upload arrow at the upper left portion of the IDE (Figure 13) or by going to Sketch -> Upload in the toolbar (Figure 14).

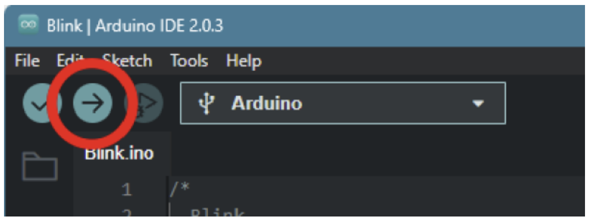


Figure 13

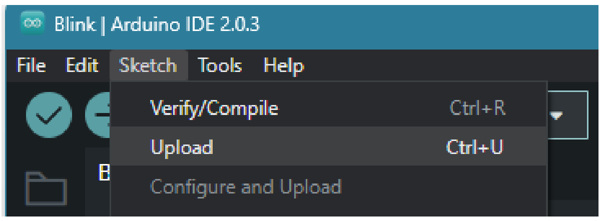


Figure 14

Now just watch as the orange light on your device start to flash at one second intervals. At this point, you can change the code if you want to slow down or speed up the blinks by editing the `delay()` at the bottom of the sketch. Higher numbers will make it blink slower and lower will make it blink faster.

If you cannot get the "Blink" program to work, please review the Troubleshooting Tips at the end of these instructions and/or contact the instructor for help.

Now that you're familiar with how an Arduino and the IDE work, we'll move into the fun part of the Lab!

Part 3: Using a Button

Hardware (physical components) for Part 3

You'll need a few things from your kit, including a button, two male-to-male jumper wires (colors don't matter), and your breadboard. Once you have those parts, install them as seen in the image below.

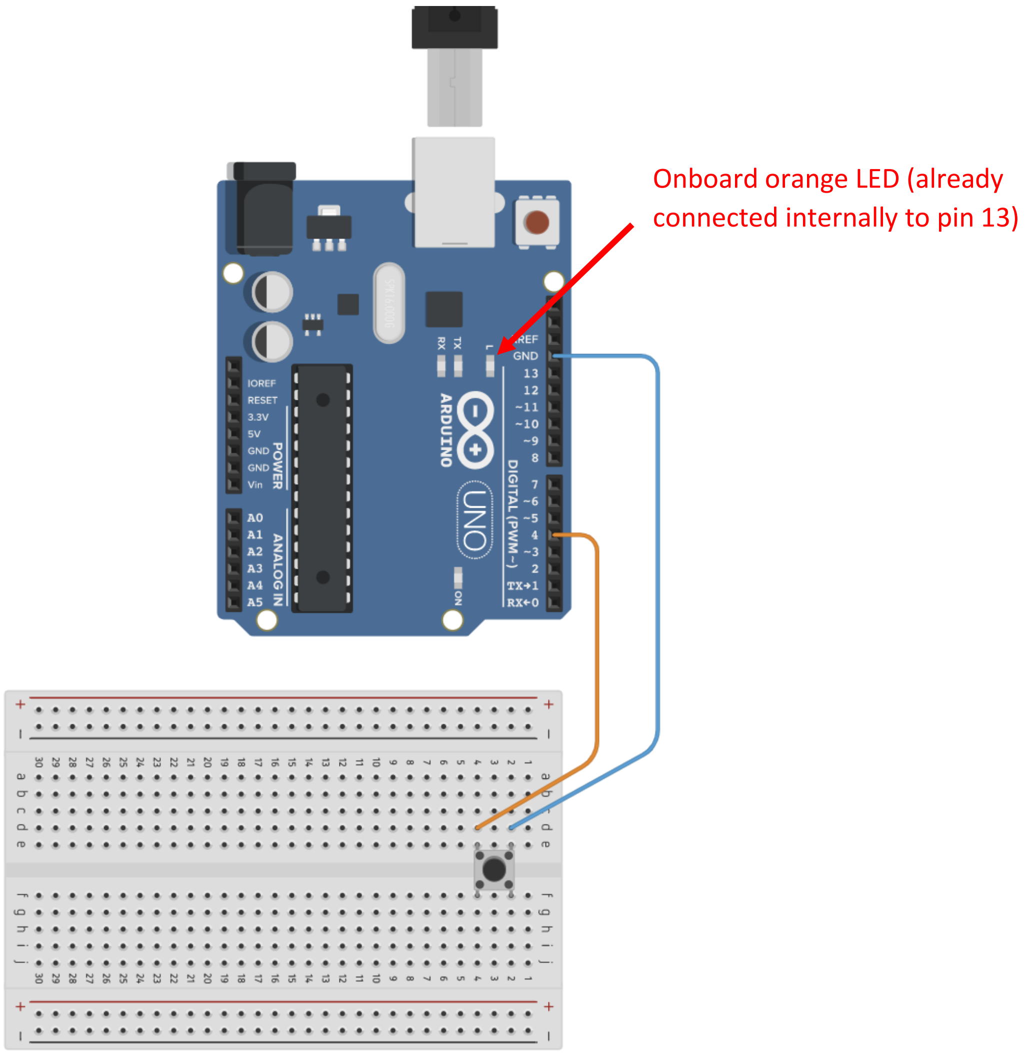


Figure 15

When you are putting the button into the breadboard, place it across the gap in the middle of the board like the button on the left in Figure 16. ***If you try to put it in sideways, it will not fit.***

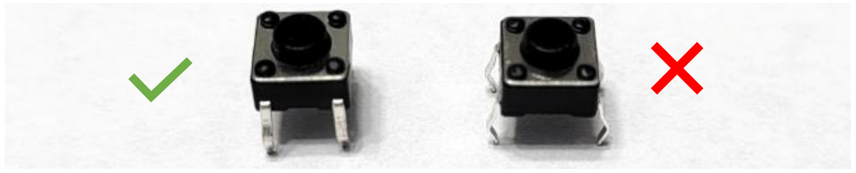


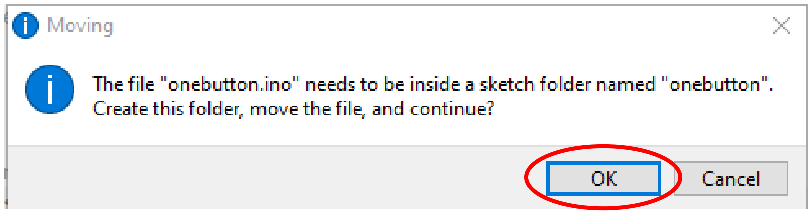
Figure 16

Once you have the button placed in the breadboard, connect it to the Arduino by placing each wire in the same column on your breadboard as one of the button pins. The blue wire should then be connected to the GND (Ground) on your Arduino, and the other wire–whichever color you choose for that–should be connected to pin 4. **Take a photo of your Part 3 hardware setup for I-Learn.**

Software (code) for Part 3

Now that the hardware is installed, we need the software (the code file or "sketch" we load to the board). This code will be executed by the Arduino's *CPU*. Click this link to download the "onebutton.ino" sketch. It will likely go to your Downloads folder.

After downloading the code file, open it from within the Arduino IDE by going to "File" -> "Open" and navigating to and selecting the "onebutton.ino" file. At this point, it may or may not ask you to put the sketch into a new folder (Figure 17). If it does, select OK to that and it will open in your IDE.



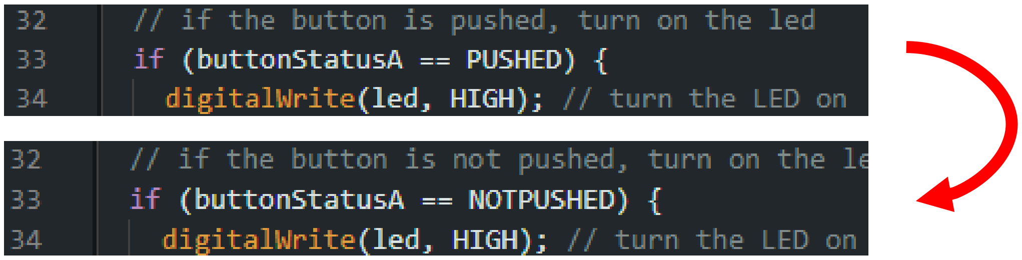
The file "onebutton.ino" needs to be inside a sketch folder named "onebutton". Create this folder, move the file, and continue?

Figure 17

Once it's open, make sure that you are still connected to the correct to the Arduino Uno in the dropdown menu like back in Figure 11. Then click upload like in Figure 13. That will copy the code to the Arduino's *Memory*. If everything was installed and uploaded correctly, at this point you should see the orange LED turn on whenever you press the button.

Now, take a minute to look at the code. You don't need to understand everything, but you may be able to understand some of it. Lines 4 and 5 show that the button is connected to pin 4 and the LED is connected to pin 13. The onboard orange LED is always connected to pin 13; that can't change. But if you wanted to, you could change which pin you wire the button to by changing line 4.

Lines 33 through 37 is where the code checks if the button is pushed. If the button is pushed, it turns on the LED. Otherwise, it turns off the LED. This is code you *will* be changing. Change `PUSHED` to `NOTPUSHED` on line 33 of your code. This will change the code so that the LED will be on when you are *not* pushing the button. As good practice, you should also change the comment above it in line 32.



```
32    // if the button is pushed, turn on the led
33    if (buttonStatusA == PUSHED) {
34      digitalWrite(led, HIGH); // turn the LED on
```

```
32    // if the button is not pushed, turn on the le
33    if (buttonStatusA == NOTPUSHED) {
34      digitalWrite(led, HIGH); // turn the LED on
```

Figure 18

Load the updated code onto your Arduino by saving the file and selecting the "Upload" arrow again. Verify that it now does the opposite of what it did before.

Part 4: Using Two Buttons and Digital Logic

Now that you've successfully used a single button to activate a light on your Arduino, we'll be introducing another button and some digital "logic" in the form of `AND` and `OR`.

Hardware for Part 4

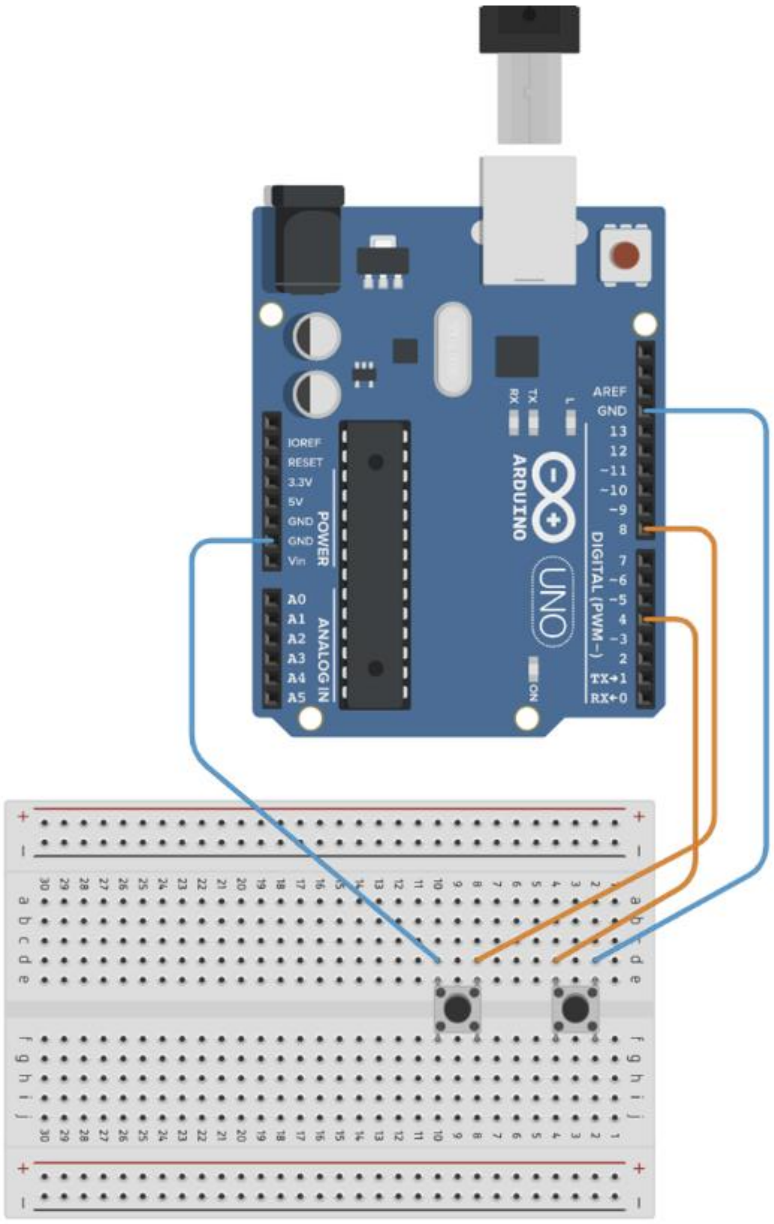


Figure 19

Add a second button. Just like before, make sure that the button fits correctly across the gap in the middle of the breadboard as shown above. Connect one side of the button to a GND pin and other side of the button to pin 8 on your Arduino. It should look something like the image above. **Take a photo of your Part 4 hardware setup for I-Learn.**

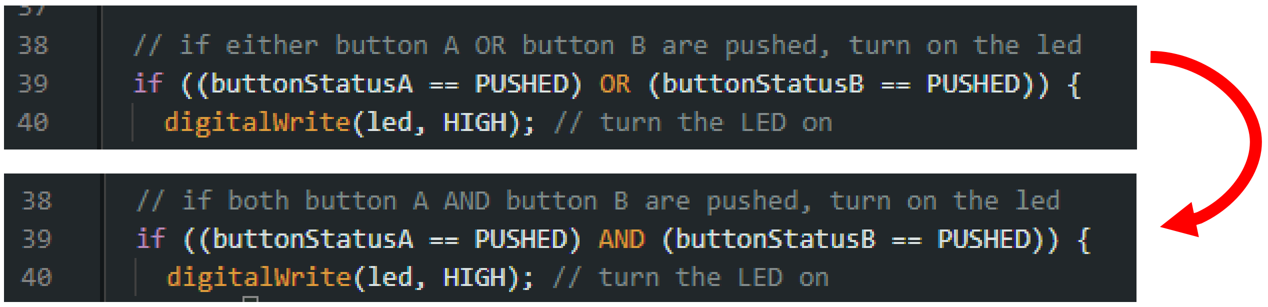
Software (code) for Part 4

Once you have the new button installed, click on this link to download the "twobuttons.ino" sketch.

Go to the Arduino IDE and open the "twobuttons.ino" file using File -> Open. The sketch is similar to the previous code, but with the addition of a new button on pin 8 (line 5) and the digital logic `OR` (line 39).

Make sure that you're still connected to the correct port for your board and then upload the "twobuttons" sketch to your Arduino. If you were able to install everything correctly, you should notice that at this point the light on your Arduino will turn on when you press either button.

For the second part of this section, you will need to change the code from an `OR` to an `AND`. Navigate in the sketch down to line 39 and replace the word `OR` with `AND`. Also update the comment above it on line 38. It should look like this:



```
38    // if either button A OR button B are pushed, turn on the led
39    if ((buttonStatusA == PUSHED) OR (buttonStatusB == PUSHED)) {
40      digitalWrite(led, HIGH); // turn the LED on
```

```
38    // if both button A AND button B are pushed, turn on the led
39    if ((buttonStatusA == PUSHED) AND (buttonStatusB == PUSHED)) {
40      digitalWrite(led, HIGH); // turn the LED on
```

Figure 20

After changing the code, save it, and reupload it to the Arduino. You should now see that the LED will only turn on if both buttonA `AND` buttonB are pressed.

Part 5: Using a Buzzer Output

For this final section, we will add another output device, a buzzer.

Hardware for Part 5

The kit comes with two buzzers. You need the one with the sticker (see Figure 21). If neither buzzer has a sticker on it, you can tell which is the correct buzzer by checking the sides. The one that you want to use DOES NOT have a groove going all the way around it, as seen in the picture below. It will have smooth sides and usually a sticker on top. If your buzzer doesn't work, try the other one.

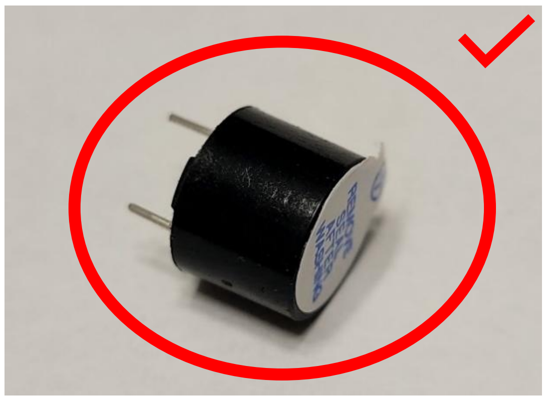


Figure 21

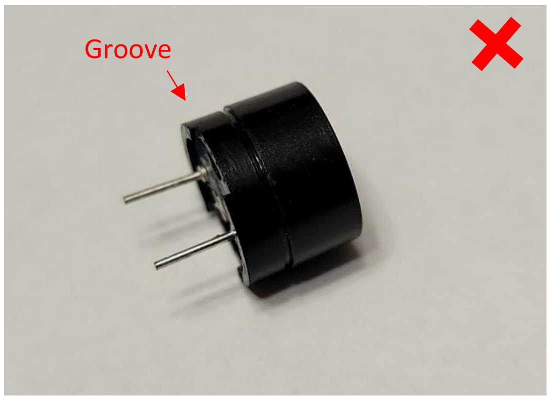


Figure 22

Just like with the buttons, make sure that the buzzer crosses the gap (Figure 22). You also need to be aware of which side of the buzzer is the positive (+) and which side is the ground (-). It will indicate on the buzzer once you remove the sticker from on top of it.

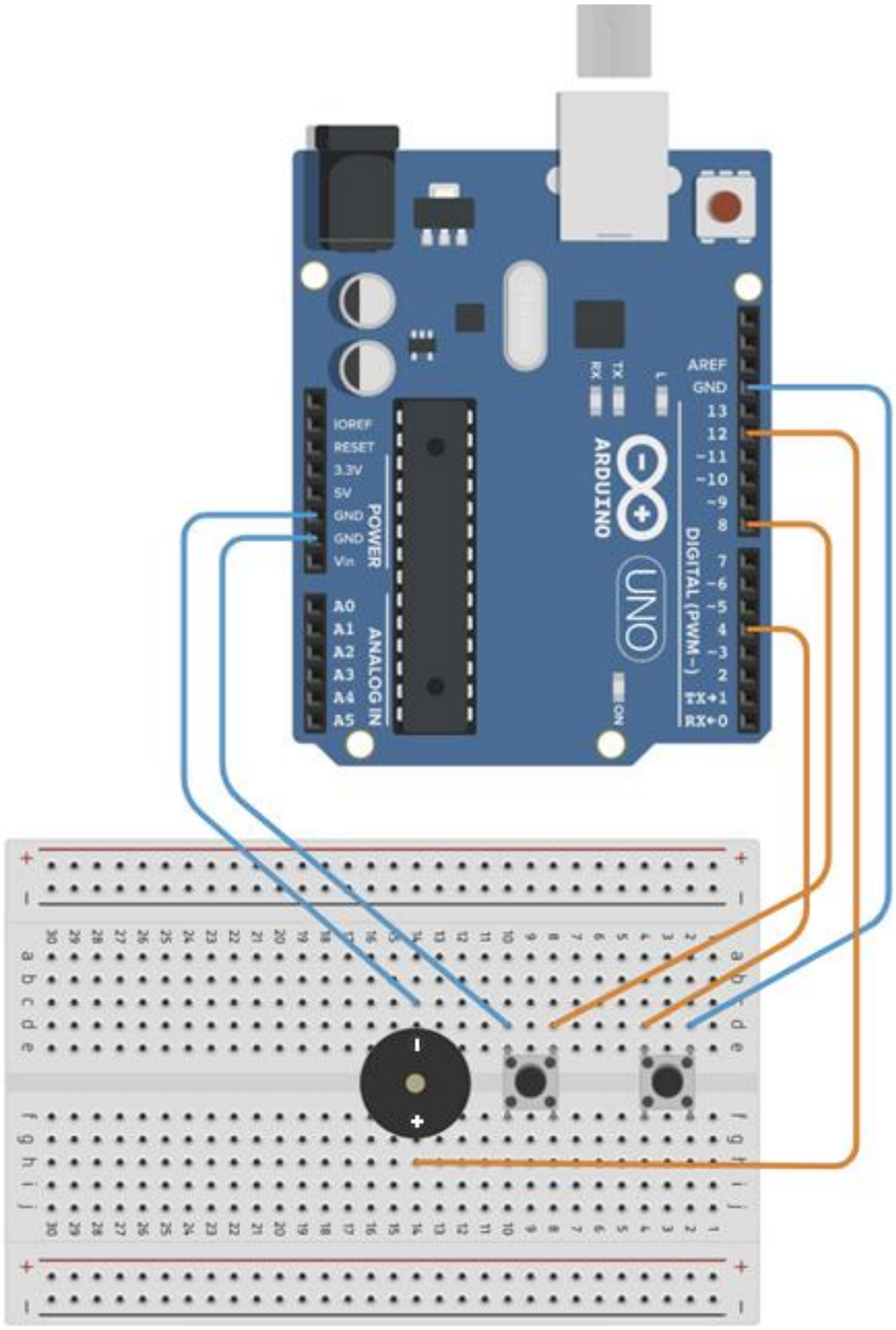


Figure 23

After you have installed the buzzer onto your breadboard, connect the (-) side to a GND on your Arduino and the (+) to pin 12 as seen in the example above.

Software (code) for Part 5

Click on this link to download the "twobuttonsbuzzer.ino" sketch. Return to your Arduino IDE and open the "twobuttonsbuzzer.ino" file. You will notice at this point that the code is almost identical to the previous files except that the buzzer has been added as an output on pin 12 (line 7).

Make sure that you are connected to your Arduino still and then upload the sketch. As long as everything has been installed correctly, when you press both buttons, the buzzer and LED will both activate. (If the buzzer doesn't make a sound, try reversing +/-. Some buzzers come incorrectly labeled.)

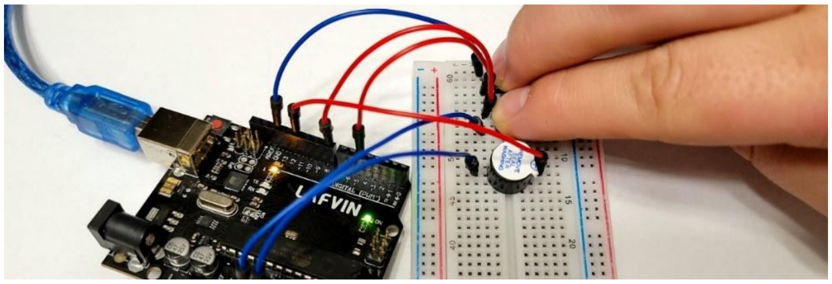


Figure 24

Take a photo of your Part 5 setup for I-Learn *while pushing both buttons and showing the LED on.* Your photo should look like the picture above in Figure 24.

Final Step: Submitting to I-Learn

Your submission for this lab will include the three photos for Part 3, Part 4, and Part 5. Submit them in the "W02.1 Lab Report" assignment and fill out other the questions there. Congratulations, you're finished!

Troubleshooting Tips:

If you were not able to upload code from your laptop to your Arduino, try some of the following steps:

1. Make sure you *didn't* install the "Nightly Builds" (double check which link you clicked at https://www.arduino.cc/en/software).
2. Try rebooting your laptop.
3. The BYU-I Wi-Fi sometimes blocks the server for installing/updating the Arduino drivers. Try using your Wi-Fi at home, a hotspot, or a VPN.
4. Sometimes the newest version of the Arduino drivers can have issues. Try installing an older version of the drivers. Go to "Tools" -> "Board" -> "Boards Manager." On the left, the Boards Manager will appear. Under "Arduino AVR Boards," there is a dropdown list. In the list, select an older version (such as 1.6.22) and then "INSTALL."
5. Sometimes the code files (the .ino files) need to be opened from *within* the Arduino IDE using "File" -> "Open" rather from *outside* the IDE in a file browser/explorer/finder.
6. Make sure your USB ports work by testing a different USB device (such as a flash drive).
7. If all the above tips fail, try installing an older version of the Arduino IDE. On the downloads page (link), scroll down to find the "Legacy IDE (1.8.X)." Once you install the old version, you will select the correct board using "Tools" -> "Board" -> "Arduino AVR Boards" -> "Arduino Uno." Then you will select the correct port by using "Tools" -> "Port" -> and then selecting the port that shows the Arduino Uno. After these steps, then try uploading the code to your Arduino.