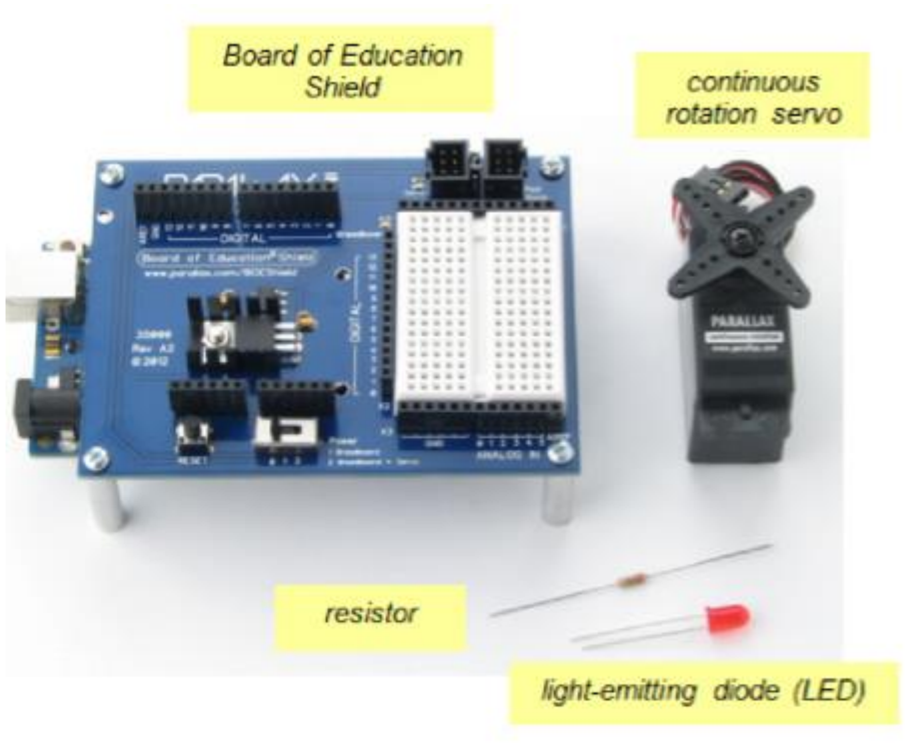
Lab 2. Shield, Resistors and LEDs

In this Lab, you will use the BOE Shield for building and testing circuits with resistors, and light-emitting diodes. Along the way, you'll start learning the basics of building circuits and making the Arduino interact with them.


Board of Education Shield
continuous rotation servo
resistor
light-emitting diode (LED)

Activity 1: BOE Shield Setup

The Board of Education Shield makes it easy to build circuits and connect servos to the Arduino module.

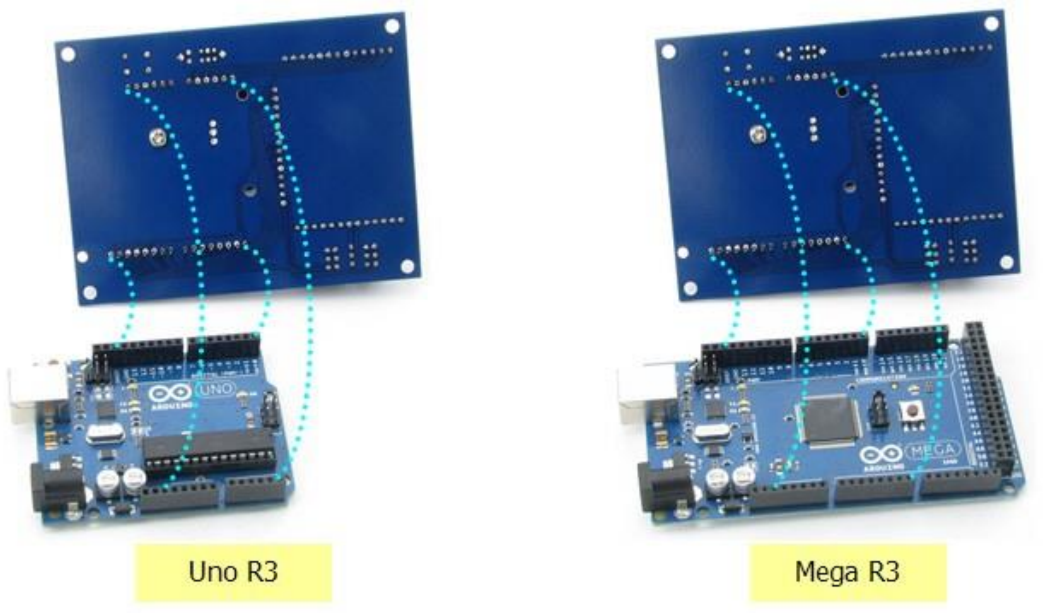
Parts List:

(1) Arduino module
(1) Board of Education Shield
(4) 1″ round aluminum standoffs
(4) pan head screws, 1/4″ 4-40
(3) 1/2″ round nylon standoffs
(3) nylon nuts, 4-40
(3) pan head screws, 7/8″, 4-40

Instructions:

The four groups of pins under the BOE Shield plug into the four Arduino socket headers. There are also three board-connection holes in the shield that line up with holes in the Arduino module, designed to connect the two boards together with screws and nylon standoffs.

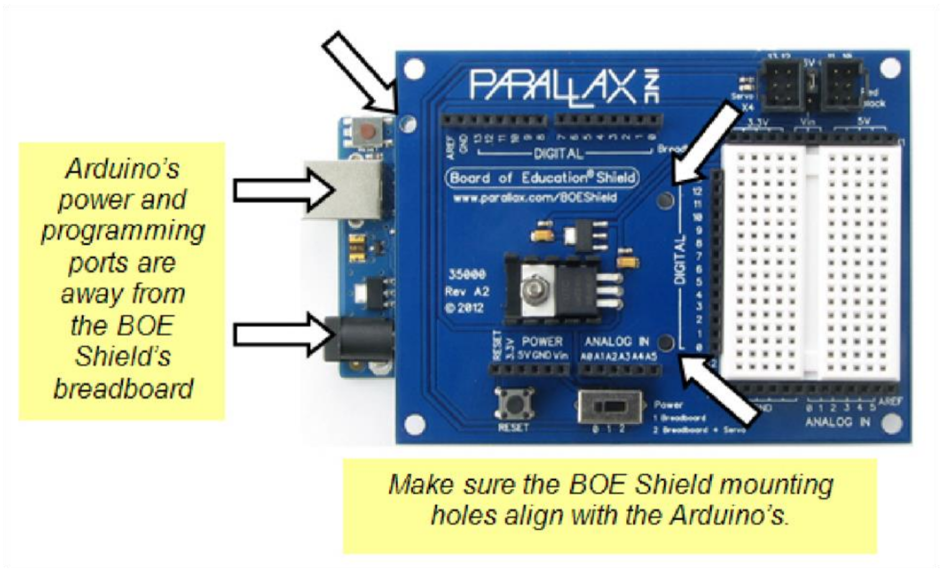
If you have a revision 3 Arduino, it will be labeled UNO R3 or MEGA R3 on the back. R3 boards will have two empty pairs of sockets, closest to the USB and power connectors, after socketing the shield. Earlier versions, such as 2, 1, and Duemilanove, have the same number of sockets as the shield has pins, so there will be no empty sockets left over. If you have an Arduino Mega, the four pin groups will fit into the four headers closest to the USB and power connectors, as shown in the box below.



✓ Disconnect the programming cable from your Arduino module.

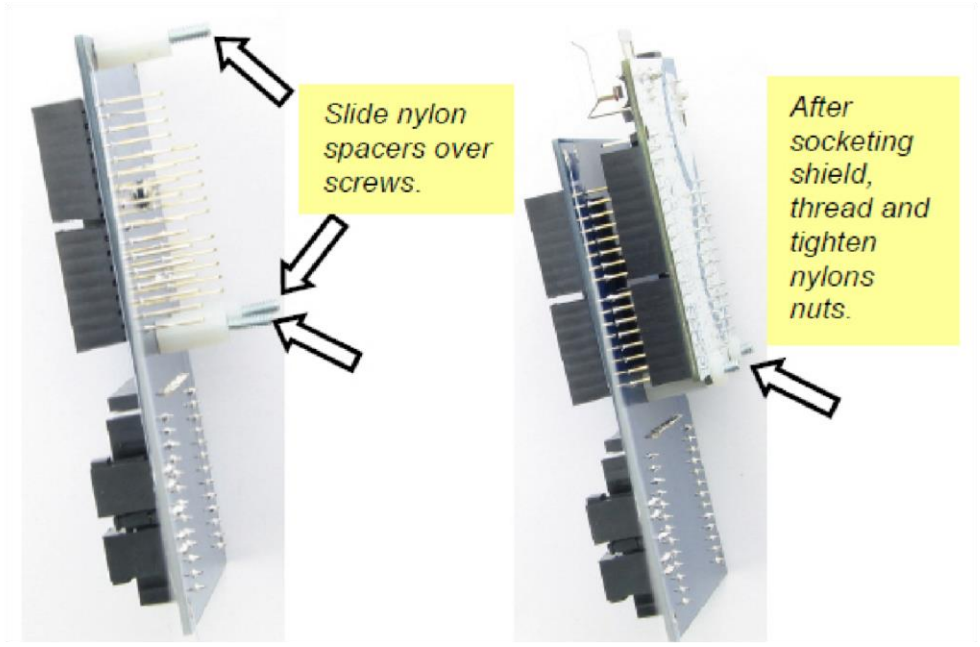
✓ Look closely at your Arduino module and the pins on the BOE Shield to see how the sockets and pins will line up for your particular boards. Note that If you have an Arduino Mega, its USB port and power jack will be close to the edge of the shield, like the image on the right.

Component placement varies a little bit for the different Arduino models; some can only fit one or two nylon standoffs for holding the boards together. This is okay, but you need to find out which holes you can use before socketing the B Education Shield.


Arduino's power and programming ports are away from the BOE Shield's breadboard
Make sure the BOE Shield mounting holes align with the Arduino's.

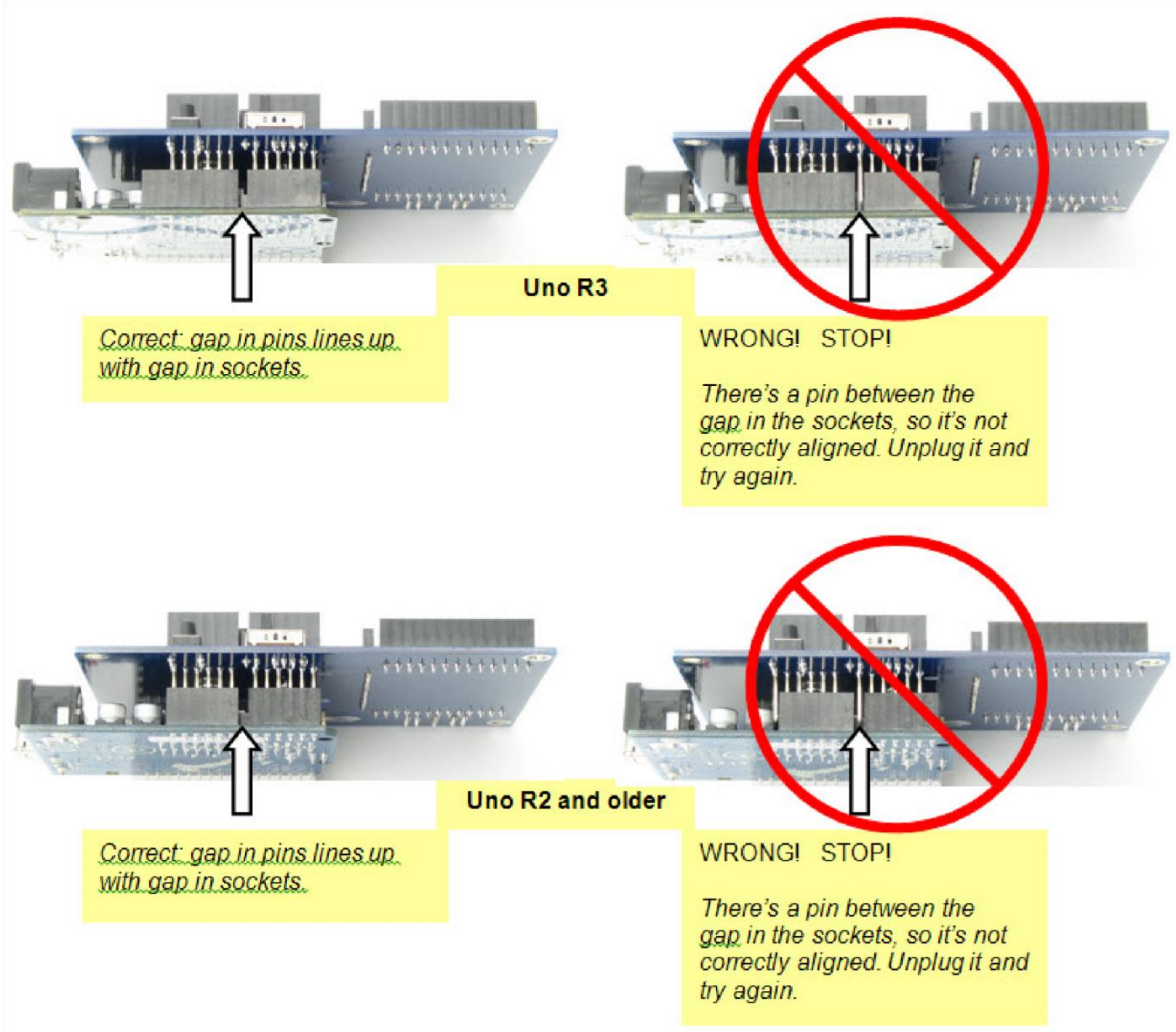
- ✓ Hold a nylon spacer over each mounting hole on your Arduino module, and look through it to see if the spacer can line up with the hole completely.
- ✓ For each hole that works on your Arduino module, insert a 7/8″ screw through the corresponding board-connection hole in your Board of Education Shield.

2


Slide nylon spacers over screws.
After socketing shield, thread and tighten nylons nuts.

- ✓ Slide a nylon spacer over each screw you used.
- ✓ Line up the Arduino module's sockets with the Board of Education Shield's pins.
- ✓ Also line up the 7/8″ screws with the mounting holes in the Arduino board.
- ✓ Gently press the two boards together until the pins are firmly seated in their sockets. The sockets will not cover the pins completely; there will be about 3/8″ (~5 mm) of the pins still exposed between the bottom of the shield and the top of the sockets.

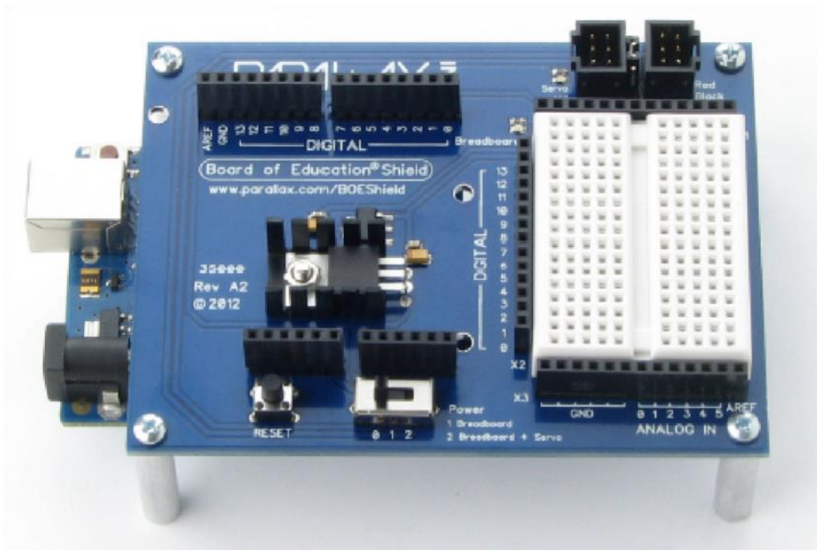
✓ Check to make ABSOLUTELY SURE your pins are seated in the sockets correctly. It is possible to misalign the pins, which can damage your board when it is powered.


Uno R3
Correct: gap in pins lines up with gap in sockets.
WRONG! STOP!
There's a pin between the gap in the sockets, so it's not correctly aligned. Unplug it and try again.
Uno R2 and older
Correct: gap in pins lines up with gap in sockets.
WRONG! STOP!
There's a pin between the gap in the sockets, so it's not correctly aligned. Unplug it and try again.

- Thread a nylon nut over each screw, and tighten gently.

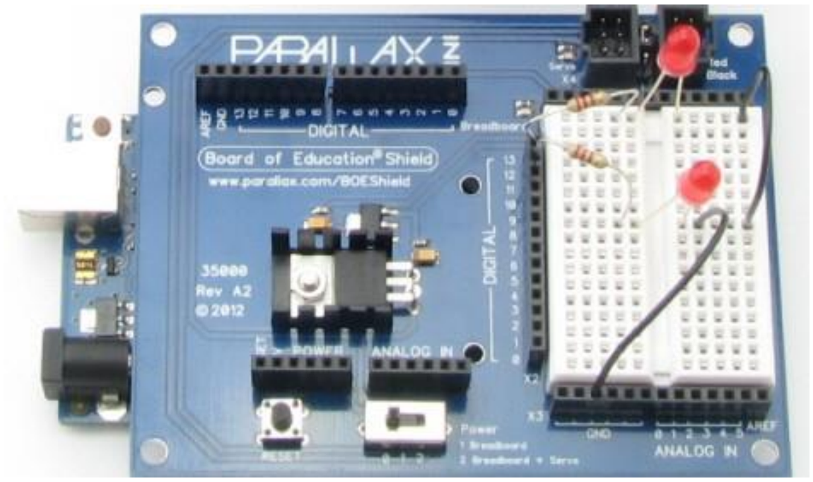
To keep the connected boards up off of the table, we'll mount tabletop standoffs to each corner of the Board of Education Shield.

- Thread a 1/4″ screw through a corner hole on the Board of Education Shield from the top side.
- Thread a 1″ aluminum standoff onto the screw and tighten gently.
- Repeat until all four standoffs are installed.



Activity 2: Build and Test LED Indicator Lights

Indicator lights give people a way to see a representation of what's going on inside a device, or patterns of communication between two devices. Next, you will build indicator lights to display the communication signals that the Arduino will send to the servos. If you haven't ever built a circuit before, don't worry, this activity shows you how.

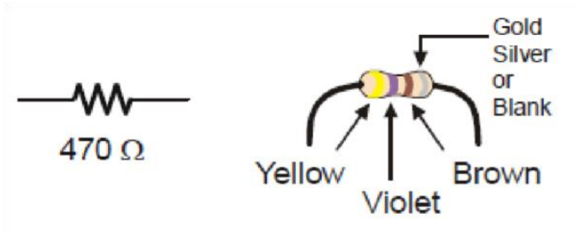


Introducing the Resistor

Introducing the Resistor

A resistor is a component that resists the flow of electricity. This flow of electricity is called current. Each resistor has a value that tells how strongly it resists current flow. This resistance value is called the ohm, and the sign for the ohm is the Greek letter omega: Ω. (Later on you will see the symbol kΩ, meaning kilo-ohm, which is one thousand ohms.)

This resistor has two wires (called leads and pronounced "leeds"), one coming out of each end. The ceramic case between the two leads is the part that resists current flow. Most circuit diagrams use the jagged line symbol with a number label to indicate a resistor of a certain value, a 470 Ω resistor in this case. This is called a schematic symbol. The part drawing on the right is used in some beginner-level texts to help you identify the resistors in your kit, and where to place them when you build circuits.

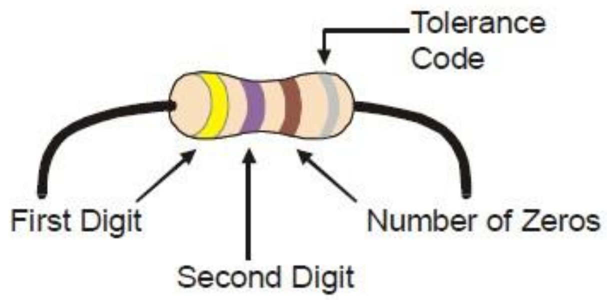


The resistors in your parts kit have colored stripes that indicate what their resistance values are. There is a different color combination for each resistance value. For example, the color code for the 470 Ω resistor is yellow-violet-brown.

There may be a fourth stripe that indicates the resistor's tolerance. Tolerance is measured in percent, and it tells how far off the part's true resistance might be from the labeled resistance. The fourth stripe could be gold (5%), silver (10%) or no stripe (20%). For the activities in this book, a resistor's tolerance does not matter, but its value does.

Each color bar on the resistor's case corresponds to a digit, as listed in the table below. Resistor Color Code Values

Digit	0	1	2	3	4	5	6	7	8	9
Color	black	brown	red	orange	yellow	green	blue	violet	gray	white



Here's how to find the resistor's value, in this case proving that yellow-violet-brown is really 470 Ω:

- The first stripe is yellow, which means the leftmost digit is a 4.
- The second stripe is violet, which means the next digit is a 7.
- The third stripe is brown. Since brown is 1, it means add one zero to the right of the first two digits.

Yellow-Violet-Brown = 4-7-0 = 470 Ω.

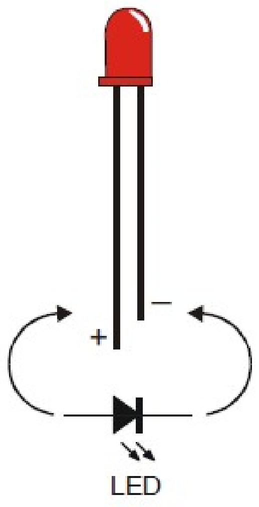
Your Turn

✓ Use the table and picture above to figure out the color code for the 220 Ω resistors you will need for the indicator lights.

Introducing the LED

Introducing the LED

A diode is a one-way electric current valve, and a light-emitting diode (LED) emits light when current passes through it. Since an LED is a one-way current valve, you have to make sure to connect it the right way for it to work as intended. An LED has two terminals: the anode and the cathode. The anode lead is labeled with the plus-sign (+) in the part drawing, and it is the wide part of the triangle in the schematic symbol. The cathode lead is the pin labeled with a minussign (-), and it is the line across the point of the triangle in the schematic symbol.

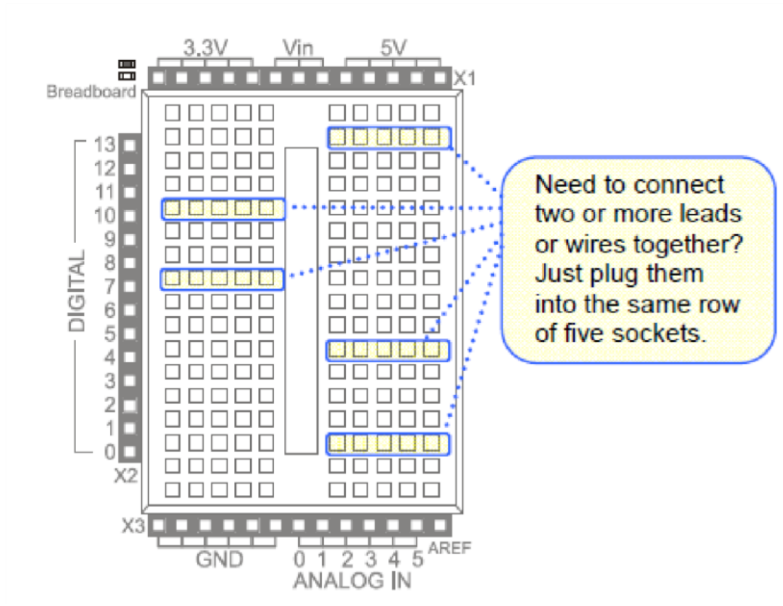


When you build an LED circuit, you will have to make sure the anode and cathode leads are connected to the circuit properly. You can tell them apart by the shape of the LED's plastic case. Look closely at the case—it's mostly round, but there is a small flat spot right near one of the leads, and that tells you it's the cathode. Also note that the LED's leads are different lengths. Usually, the shorter lead is connected to the cathode.

Always check the LED's plastic case.
Usually, the longer lead is connected to the LED's anode, and the shorter lead is connected to its cathode. But sometimes the leads have been clipped to the same length, or a manufacturer does not follow this convention. So, it's best to always look for the flat spot on the case. If you plug an LED in backwards, it will not hurt it, but it won't emit light until you plug it in the right way.

Introducing the Prototyping Area

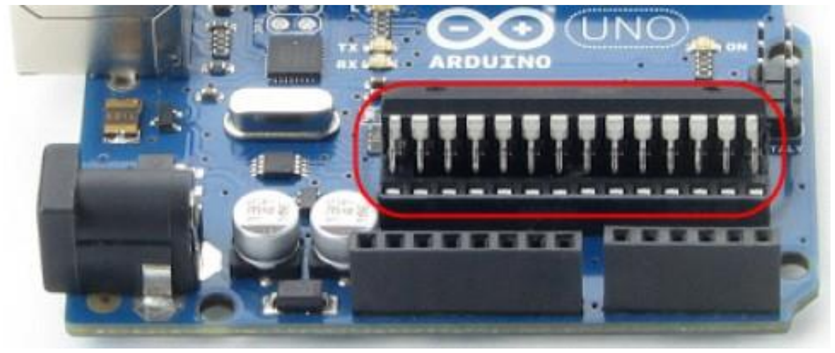
The white board with lots of square sockets in it is called a solderless breadboard. This breadboard has 17 rows of sockets. In each row, there are two five-socket groups separated by a trench in the middle. All the sockets in a 5-socket group are connected together underneath with a conductive metal clip. So, two wires plugged into the same 5-socket group make electrical contact. This is how you will connect components, such as an LED and resistor, to build circuits. Two wires in the same row on opposite sides of the center trench will not be connected.


3.3V
Vin
5V
X1
Breadboard
13
12
11
10
9
8
7
6
5
4
3
2
1
0
DIGITAL
X2
X3
GND
0 1 2 3 4 5
AREF
ANALOG IN
Need to connect two or more leads or wires together? Just plug them into the same row of five sockets.

The prototyping area also has black sockets along the top, bottom, and left.

- Top: these sockets have three supply voltages for the breadboard: 3.3 V, Vin (input voltage from either battery pack or programming cable), and 5 V.
- Bottom-left: The first six sockets along the bottom-left are ground terminals, labeled GND; think of them as a supply voltage that's 0 V. Collectively, the 3.3V, Vin, 5V and GND are called the power terminals, and they will be used to supply your circuits with electricity.
- Bottom-right: The ANALOG IN sockets along the bottom-right are for measuring variable voltages; these connect to the Arduino module's ANALOG IN sockets.
- Left: The DIGITAL sockets on the left have labels from 0 to 13. You will use these to connect your circuit to the Arduino module's digital input/output pins.

Digital and analog pins are the small pins on the Arduino module's Atmel microcontroller chip. These pins electrically connect the microcontroller brain to the board.



A sketch can make the digital pins send high or low signals to circuits. In this chapter, we'll do that to turn lights on and off. A sketch can also make a digital pin monitor high or low signals coming from a circuit; We'll do that in another chapter to detect whether a contact switch has been pressed or released.

A sketch can also measure the voltages applied to analog pins; we'll do that to measure light with a phototransistor circuit in another chapter.

How To: The Basics of Breadboarding (A YouTube video from Parallax Inc.)

LED Test Circuit

LED Test Circuit Parts

(2) LEDs – Red
(2) Resistors, 220 Ω (red-red-brown)

Always disconnect power to your board before building or modifying circuits!
1. Set the BOE Shield’s Power switch to 0.
2. Disconnect the programming cable and battery pack.

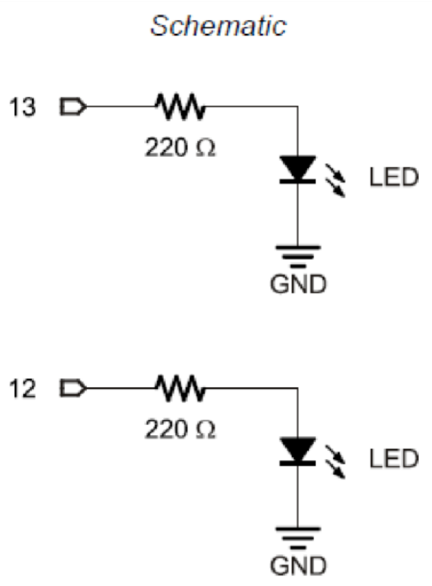
LED Test Circuits

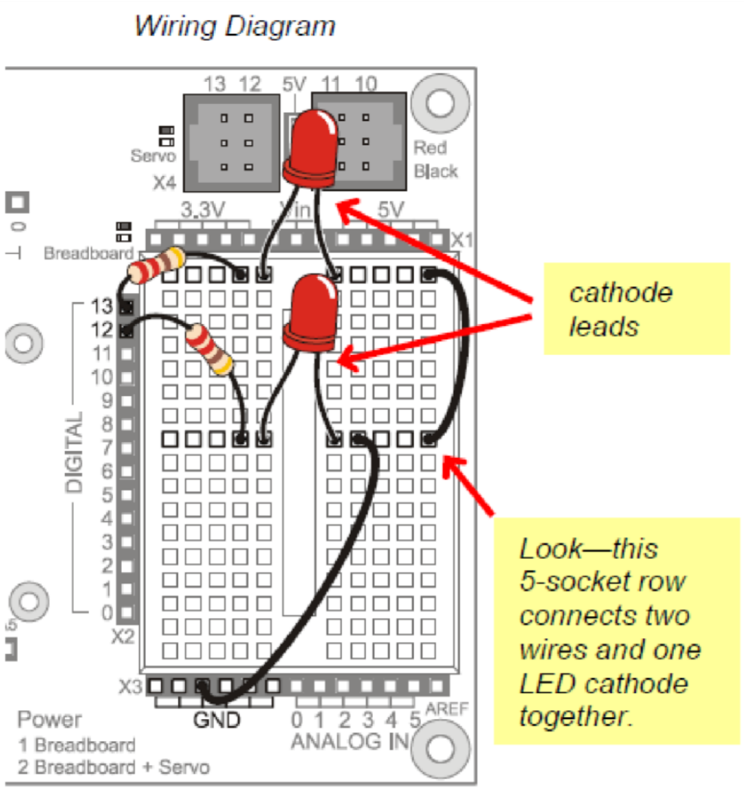
The image below shows the indicator LED circuit schematic on the left, and a wiring diagram example of the circuit built on your BOE Shield’s prototyping area on the right.

✓ Build the circuit shown below. If you are new to building circuits, try to follow the wiring diagram exactly.

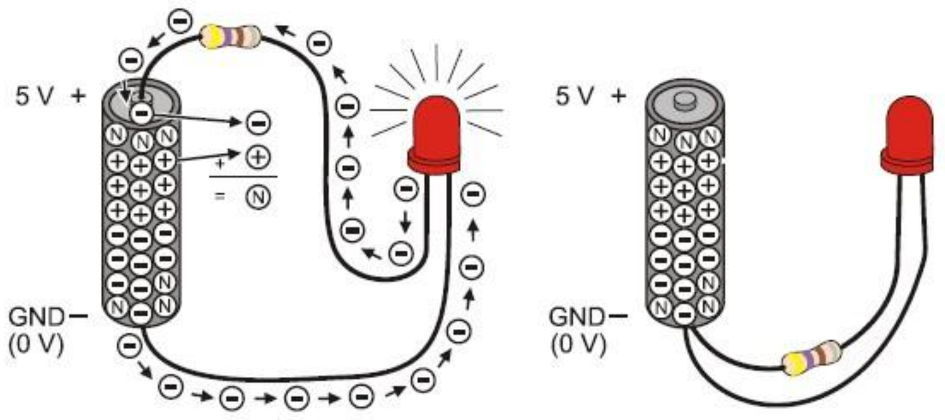
✓ Make sure your LED cathode leads are connected to GND. Remember, the cathode leads are the shorter pins that are closer to the flat spot on the LED’s plastic case. Each cathode lead should be plugged into the same 5socket row as the wires that run to the GND sockets.

✓ Make sure that each longer anode lead is connected to the same 5-socket row as a resistor lead.





The next picture will give you an idea of what is going on when you program the Arduino to control the LED circuit. Imagine that you have a 5 volt (5 V) battery. The Board of Education Shield has a device called a voltage regulator that supplies 5 volts to the sockets labeled 5V. When you connect the anode end of the LED circuit to 5 V, it's like connecting it to the positive terminal of a 5 V battery. When you connect the circuit to GND, it's like connecting to the negative terminal of the 5 V battery.



On the left side of the picture, one LED lead is connectd to 5 V and the other to GND. So, 5 V of electrical pressure causes electrons to flow through the circuit (electric current), and that current causes the LED to emit light. The circuit on the right side has both ends of the LED circuit connected to GND. This makes the voltage the same (0 V) at both ends of the circuit. No electrical pressure = no current = no light.

You can connect the LED to a digital I/O pin and program the Arduino to alternate the pin's output voltage between 5 V and GND. This will turn the LED light on/off, and that's what we'll do next.

Volts is abbreviated V.
When you apply voltage to a circuit, it's like applying electrical pressure. By convention, 5 V means "5 V higher than ground." Ground, often abbreviated GND, is considered 0 V.
Ground is abbreviated GND.
The term ground originated with electrical systems where this connection is actually a metal rod that has been driven into the ground. In portable electronic devices, ground is commonly used to refer to connections that go to the battery supply's negative terminal.

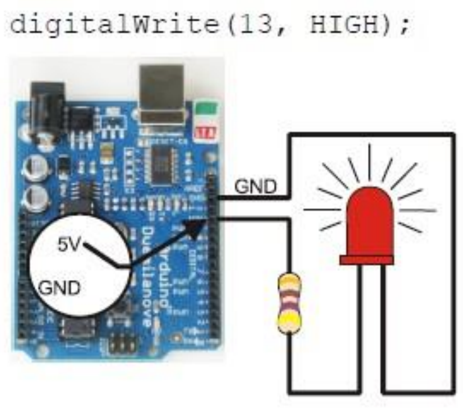
Current refers to the rate at which electrons pass through a circuit. You will often see measurements of current expressed in amps, which is abbreviated A. The currents you will use here are measured in thousandths of an amp, or milliamps. For example, 10.3 mA passes through the circuit shown above.

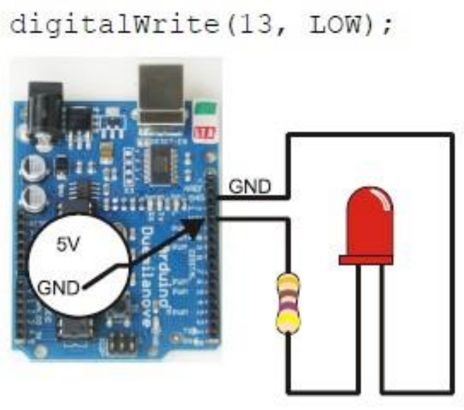
How a Sketch Makes the LED Turn On and Off

Let's start with a sketch that makes the LED circuit connected to digital pin 13 turn on/off. First, your sketch has to tell the Arduino to set the direction of pin 13 to output, using the `pinMode` function: `pinMode(pin, mode)`. The `pin` parameter is the number of a digital I/O pin, and `mode` must be either `INPUT` or `OUTPUT`.

```
void setup()                          // Built-in initialization block
{
  pinMode(13, OUTPUT);                // Set digital pin 13 -> output
}
```

Now that digital pin 13 is set to output, we can use `digitalWrite` to turn the LED light on and off. Take a look at the picture below. On the left, `digitalWrite(13, HIGH)` makes the Arduino's microcontroller connect digital pin 13 to 5 V, which turns on the LED. On the right, it shows how `digitalWrite(13, LOW)` makes it connect pin 13 to GND (0 V) to turn the LED off.





Here's the `loop` function from the next sketch. First, `digitalWrite(13, HIGH)` turns the light on, `delay(500)` keeps it on for a half-second. Then `digitalWrite(13, LOW)` turns it off, and that's also followed by `delay(500)`.

Since it's inside the `loop` function's block, the statements will repeat automatically. The result? The light will flash on/off once every second.

```
void loop()                          // Main loop auto-repeats
{
  digitalWrite(13, HIGH);
// Pin 13 = 5 V, LED emits light

    delay(500);
    // ..for 0.5 seconds
    digitalWrite(13, LOW);
    // Pin 13 = 0 V, LED no light
    delay(500);
    // ..for 0.5 seconds
  }
```

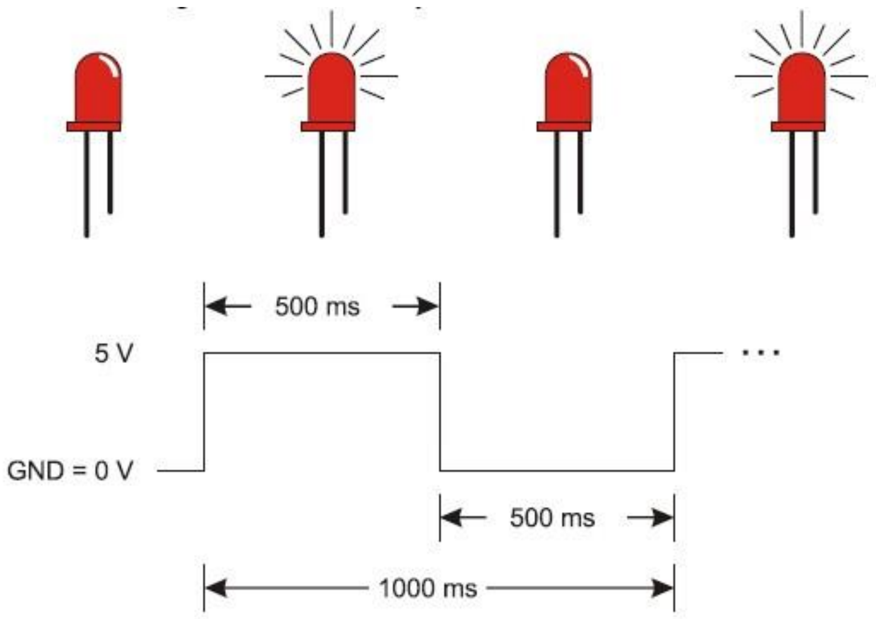
Example Sketch: HighLowLed

- ✓ Reconnect the programming cable to your board.
- ✓ Enter, save, and upload HighLowLed to your Arduino.
- ✓ Verify that the pin 13 LED turns on and off, once every second. (You may see the LED flicker a few times before it settles down into a steady blinking pattern. This happens when reprogramming the Arduino.)

```
/*
HighLowLed
Turn LED connected to digital pin 13 on/off once every second.
 */

void setup()
// Built-in initialization block
{
  pinMode(13, OUTPUT);
// Set digital pin 13 -> output
}

void loop()
// Main loop auto-repeats
{



digitalWrite(13, HIGH);


// Pin 13 = 5 V, LED emits light


delay(500);


// ..for 0.5 seconds


digitalWrite(13, LOW);


// Pin 13 = 0 V, LED no light
delay(500);

// ..for 0.5 seconds

}
```

Introducing the Timing Diagram

A timing diagram is a graph that relates a signal's high and low stages to time. This timing diagram shows you a 1000 ms slice of the `HIGH` (5 V) and `LOW` (0 V) signals from the sketch HighLowLed. Can you see how `delay(500)` is controlling the blink rate?



Your Turn – Experiment with the Blink Rates and Both LEDs

How would you make the LED blink twice as fast? How about reducing the delay function's ms parameters by half?

- ✓
- ✓ Try modifying your sketch to use delay(250). Don't forget to change it in both places!

How far can you reduce the delay before it just looks like the LED is dim instead of blinking on/off?

Blinking the pin 12 LED is a simple matter of changing the pin parameter in the pinMode and two digitalWrite function calls.

- ✓ Modify the sketch so that pinMode in the setup function uses pin 12 instead of pin 13.
- ✓ Also modify both digitalWrite statements in the loop function to use pin 12.
- ✓ Run it, and make sure the pin 12 LED blinks.

You can also make both LEDs blink at the same time.

Add statements to the sketch so that it uses pinMode twice:

```
  pinMode(13, OUTPUT);

// Set digital pin 13 -> output

pinMode(12, OUTPUT);

// Set digital pin 12 -> output
```

….and uses digitalWrite four times:

```
  digitalWrite(13, HIGH);

// Pin 13 = 5 V, LED emits light

digitalWrite(12, HIGH);

 // Pin 12 = 5 V, LED emits light

delay(500);                  // ..for 0.5 seconds
digitalWrite(13, LOW);

// Pin 13 = 0 V, LED no light

digitalWrite(12, LOW);

// Pin 12 = 0 V, LED no light

delay(500);                  // ..for 0.5 seconds
```

Run the modified sketch. Do both LEDs blink on and off together?

How would you modify the sketch again to turn one LED on while the other turns off? One circuit will need to receive a HIGH signal while the other receives a LOW signal.

✓ Try it!

Exercises

1. Write a `loop` function that makes an LED blink 5 times per second, with an on time that's 1/3rd of its off time.