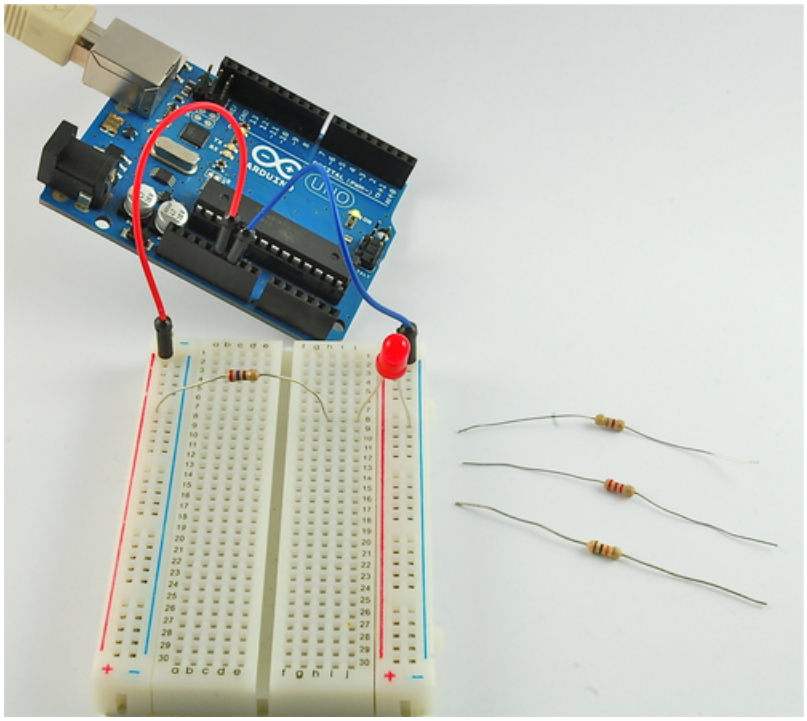
adafruit learning system

Arduino Lesson 2. LEDs

Created by Simon Monk

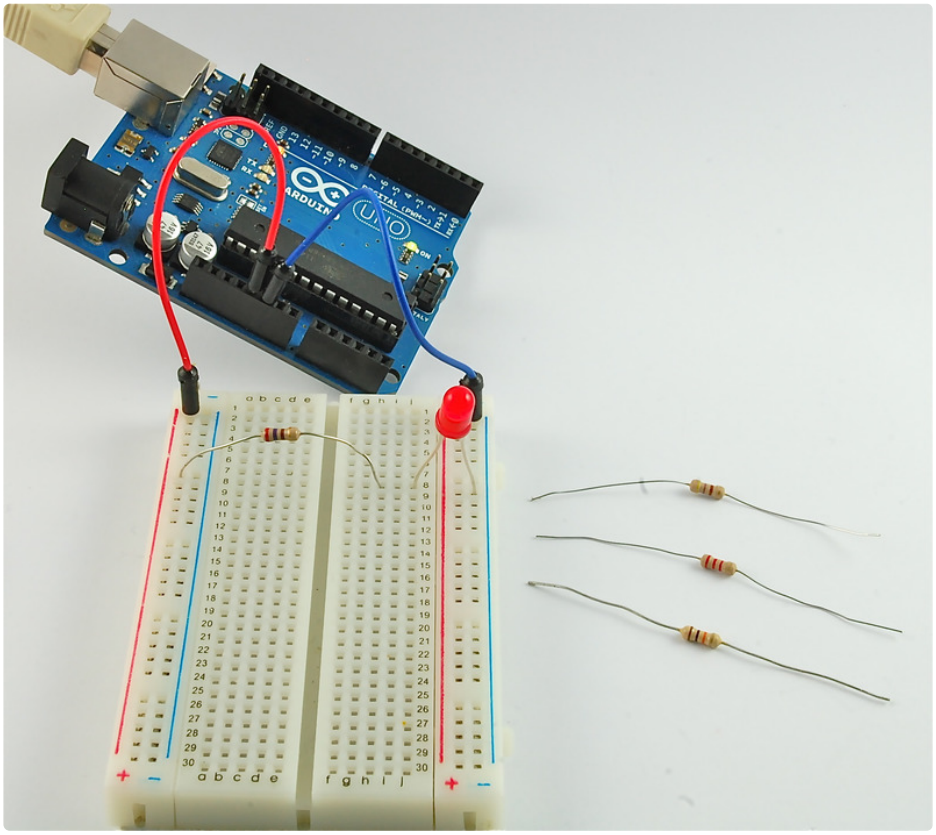


Last updated on 2021-10-22 10:56:27 AM EDT

Guide Contents

Overview

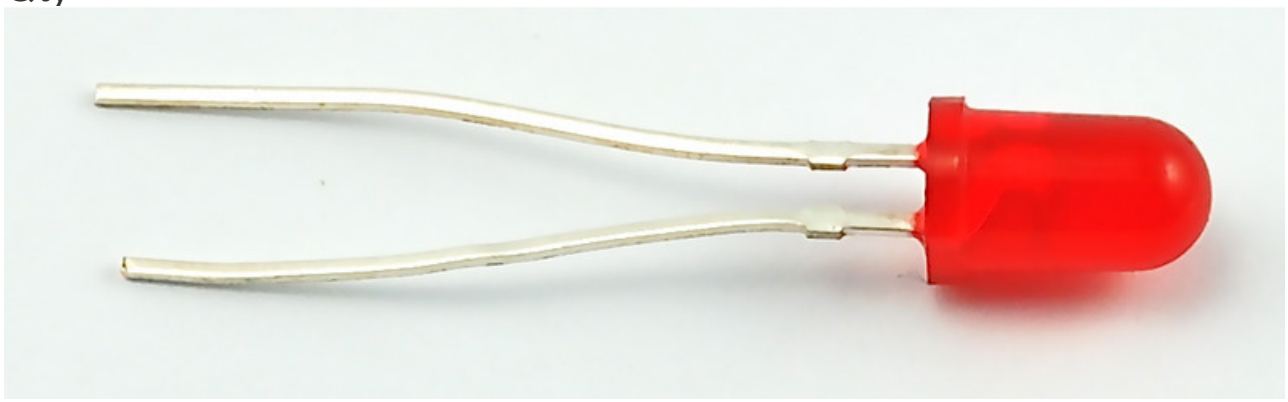
In this lesson, you will learn how to change the brightness of an LED by using different values of resistor.



Parts

To carry out the experiment described in this lesson, you will need the following parts.

Part
Qty



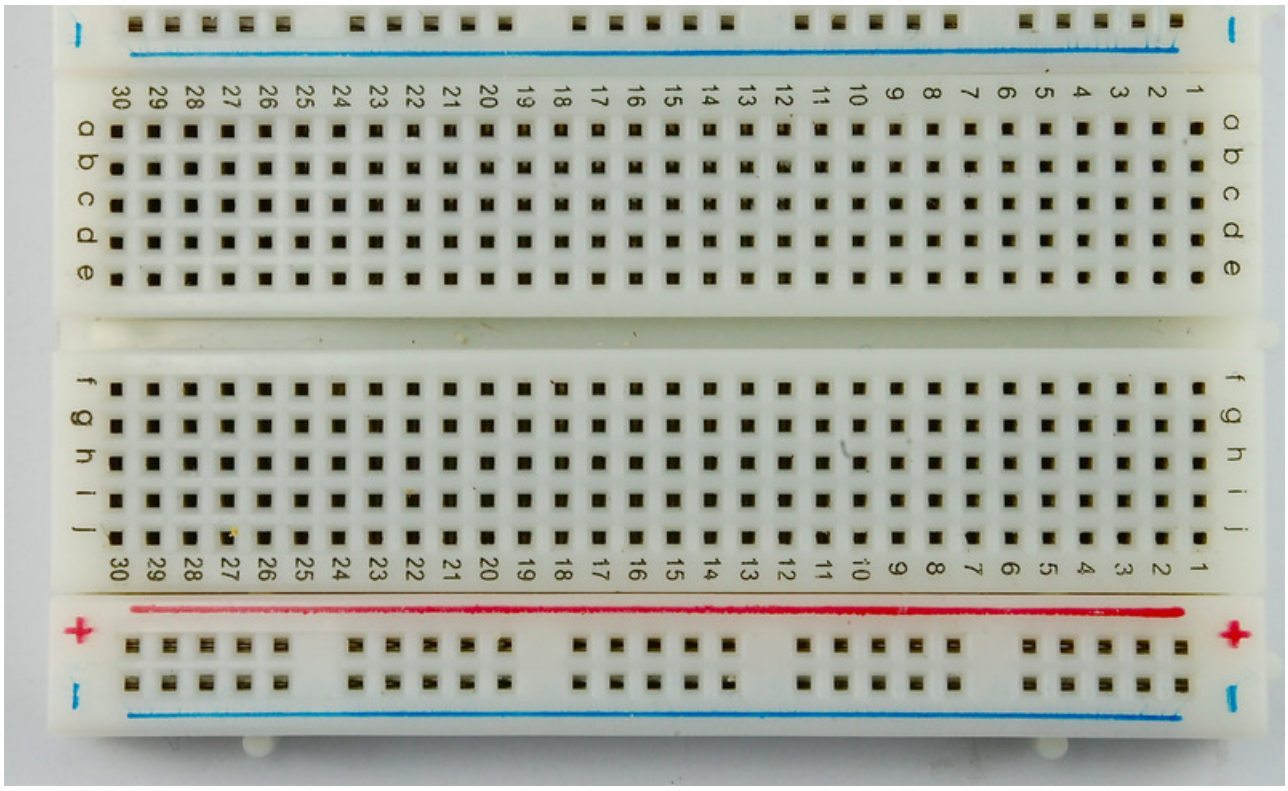
5mm Red LED 1

270 Ω Resistor (red, purple, brown stripes) 1

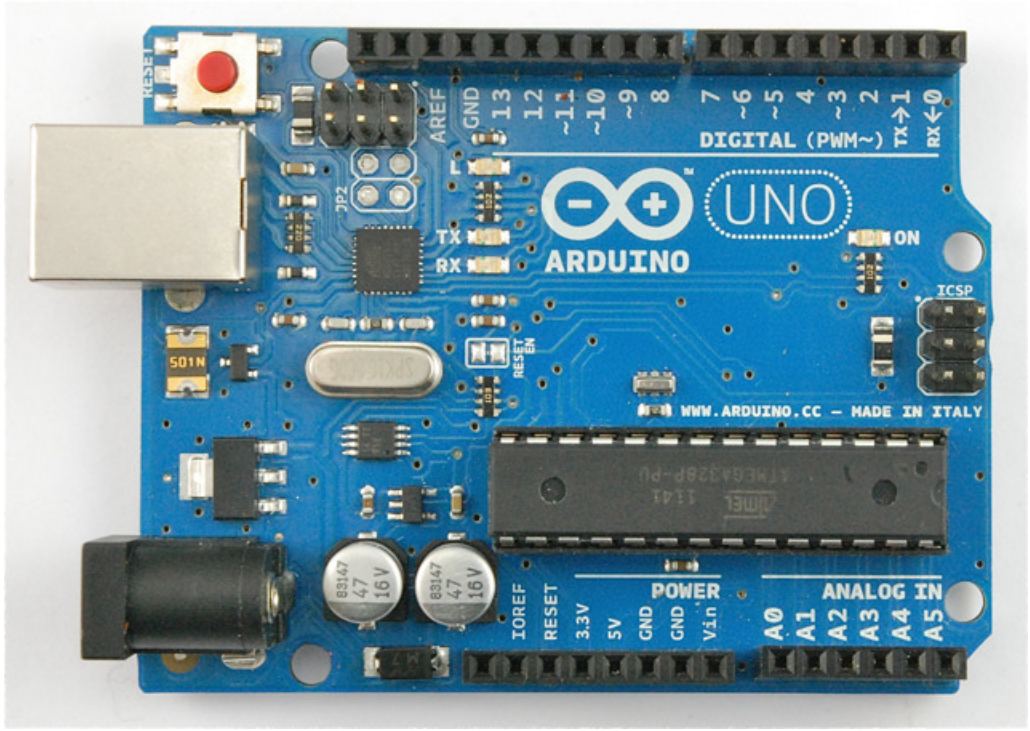
470 Ω Resistor (yellow, purple, brown stripes) 1

2.2 kΩ Resistor (red, red, red stripes) 1

10 kΩ Resistor (brown, black, orange stripes) 1



Half-size Breadboard 1



Arduino Uno R3 1

Jumper wire pack 1

LEDs

LEDs make great indicator lights. They use very little electricity and they pretty much last forever.

In this lession you will use perhaps the most common of all LEDs a 5mm red LED. 5Mm refers to the diameter of the LED and as well as 5mm, other common sizes are 3mm and the large fun 10mm LEDs.

You cannot directly connect an LED to a battery or voltage source. Firstly, because the LED has a positive and a negative lead and will not light if they are the wrong way around and secondly, an LED must be used with a resistor to limit or 'choke' the amount of current flowing through the LED - otherwise the LED could burn out!

If you do not use a resistor with an LED, then it may well be destroyed almost immediately, as too much current will flow through the LED, heating it and destroying the 'junction' where the light is produced.

There are two ways to tell which is the positive lead of the LED and which the negative.

- Firstly, the positive lead is longer.
- Secondly, where the negative lead enters the body of the LED, there is a flat edge to the case of the LED.

If you happen to have an LED that has a flat side next to the longer lead, you should assume that the longer lead is positive.

Resistors

As the name suggests, resistors resist the flow of electricity and the higher the value of the resistor, the more it resists and the less electrical current will flow through it. We are going to use this to control how much electricity flows through the LED and therefore how brightly it shines.

But first, a bit more about resistors.

The unit of resistance is called the Ohm, which is usually shortened to Ω the Greek letter Omega. Because an Ohm is a low value of resistance (it doesn't resist much at all), we also give the values of resistors in kΩ (1000 Ω) and MΩ (1000,000 Ω). These are called kilo-ohms and mega-ohms.

In this lesson, we are going to use four different values of resistor, 270Ω, 470Ω, 2.2kΩ and 10kΩ. These resistors all look the same, except that they have different colored stripes on them. These stripes tell you the value of the resistor.

The resistor color code works like this, for resistors like this with three colored stripes and then a gold stripe at one end.

Each color has a number, as follows:

- Black 0
- Brown 1
- Red 2
- Orange 3
- Yellow 4
- Green 5
- Blue 6
- Purple 7
- Gray 8
- White 9

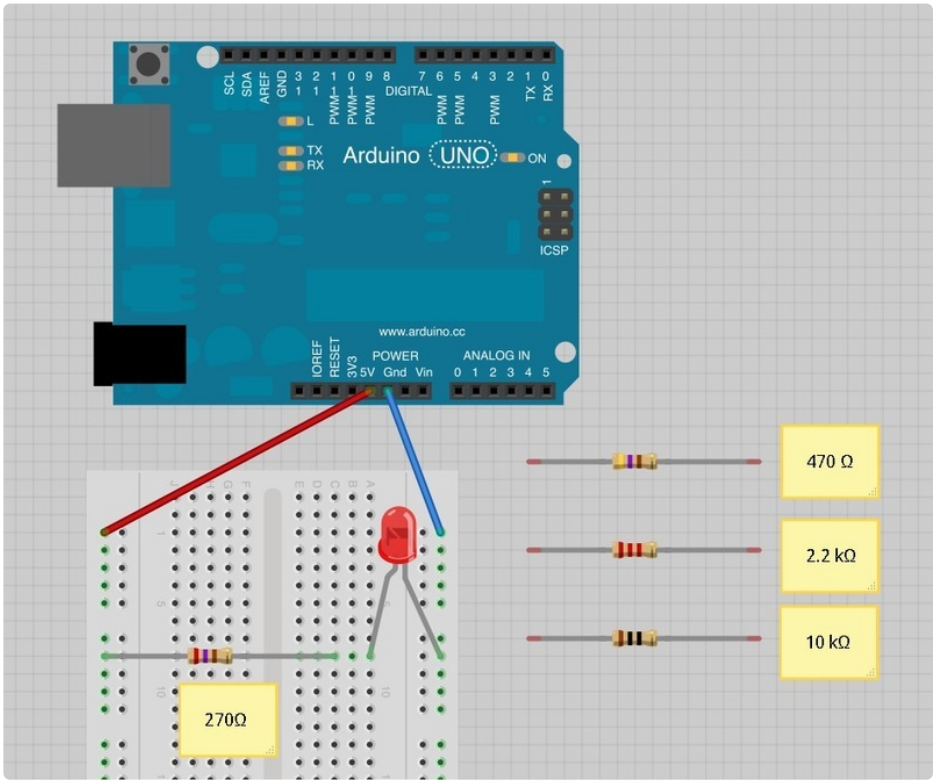
The first two striped are the first two digits of the value, so red, purple means 2, 7. The next stripe is the number of zeros that need to come after the first two digits, so if the third stripe is brown, as it is in the photograph above, then there will be one zero and so the resistor is 270Ω.

A resistor with stripes brown, black, orange is 10 and three zeros so 10,000 Ω in other words 10 kΩ.

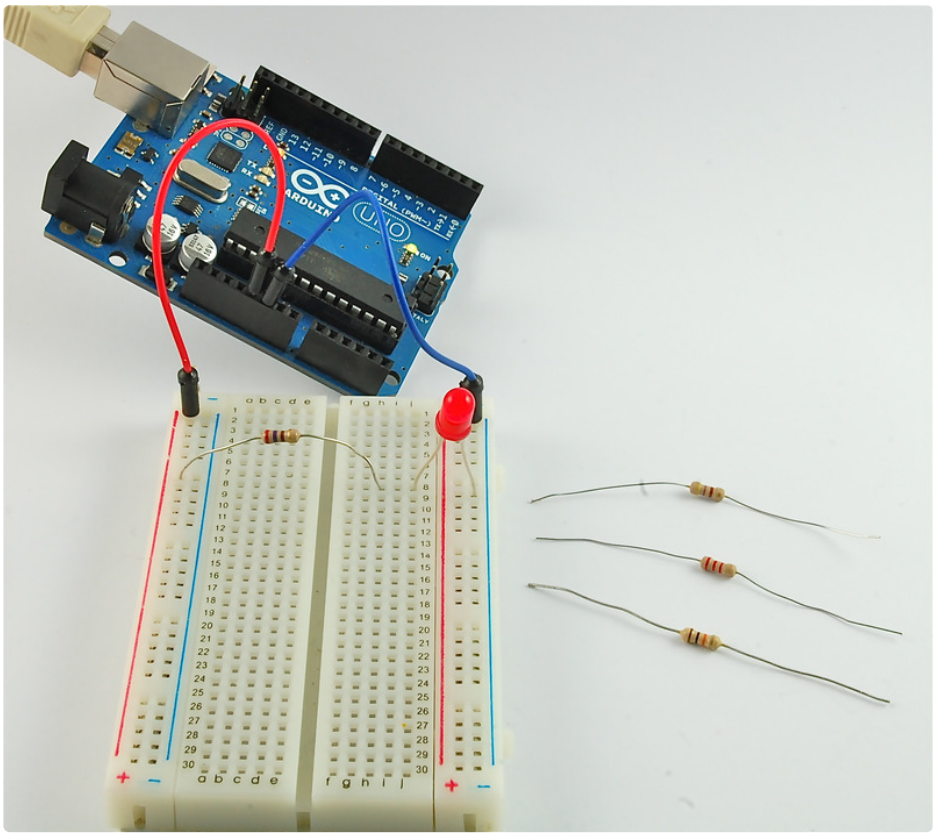
Unlike LEDs, resistors do not have a positive and negative lead. They can be connected either way around.

Breadboard Layout

Connect up your stripboard as shown below, using the 270Ω resistor.



The Arduino is a convenient source of 5 Volts, that we will use to provide power to the LED and resistor. You do not need to do anything with your Arduino, except plug it into a USB cable.

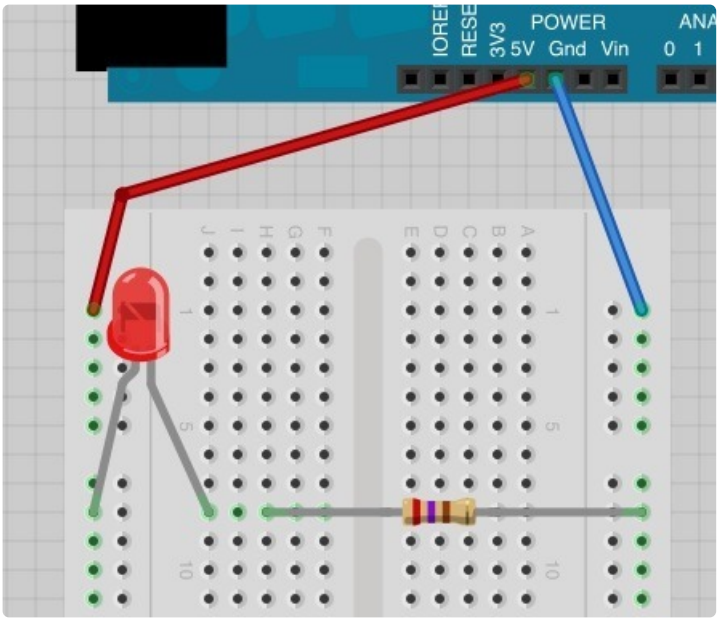


With the 270 Ω resistor in place, the LED should be quite bright. If you swap out the 270 Ω resistor for the 470 Ω resistor, then the LED will appear a little dimmer. With the 2.2kΩ resistor in place the LED should be quite faint. Finally, with the 10 kΩ resistor in place, the LED will be just about visible. Pull the red jumper lead out of the breadboard and touch it into the hole and remove it, so that it acts like a switch. You should just be able to notice the difference.

Turning out the lights might help even more.

Moving the Resistor

At the moment, you have 5V going to one leg of the resistor, the other leg of the resistor going to the positive side of the LED and the other side of the LED going to GND. However, if we moved the resistor so that it came after the LED, as shown below, the LED will still light.

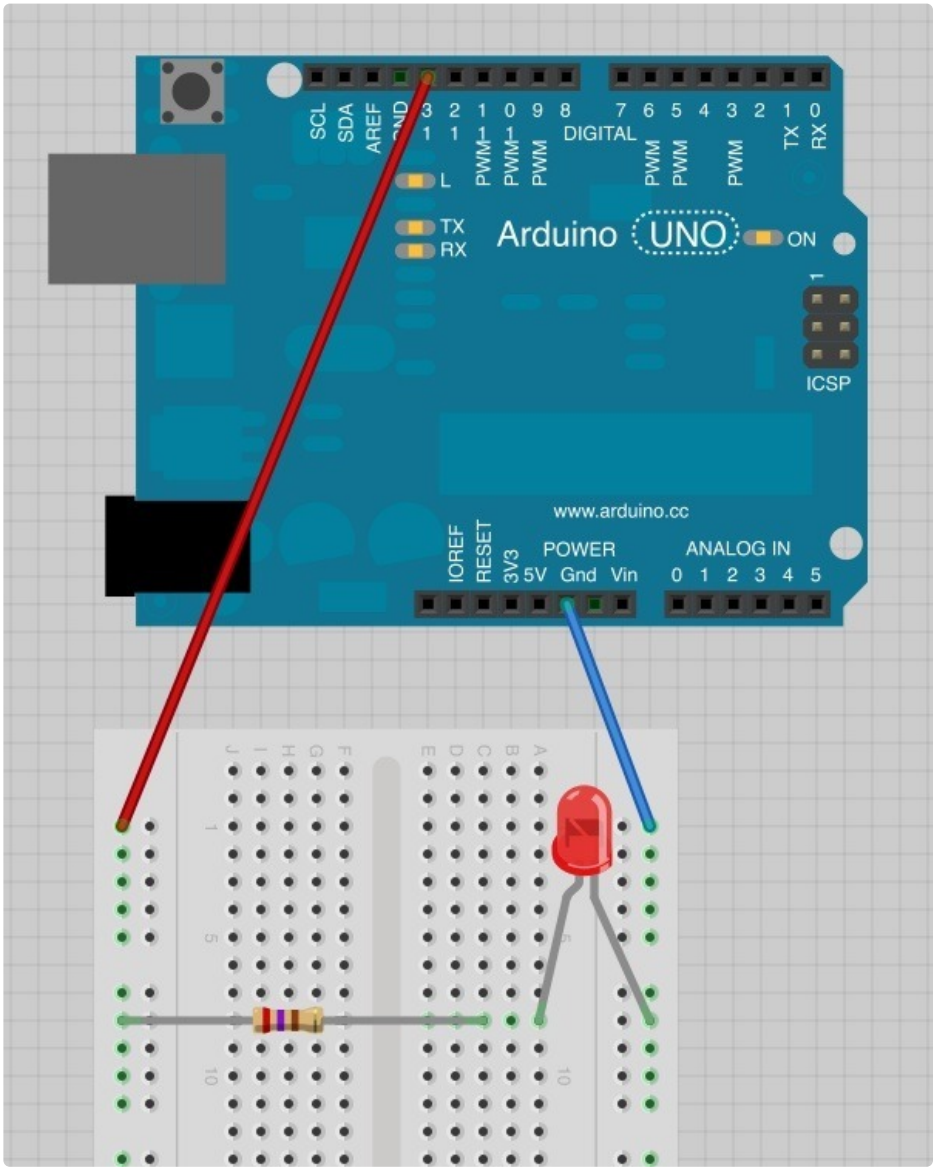


Note, you will probably want to put the 270Ω resistor back in place.

So, it does not matter which side of the LED we put the resistor, as long as it is there somewhere.

Blinking the LED

With a simple modification of the breadboard, we could attach the LED to an output pin of the Arduino. Move the red jumper wire from the Arduino 5V connector to D13, as shown below:



Now load the 'Blink' example sketch from Lesson 1. You will notice that both the built-in 'L' LED and the external LED should now blink.

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

Lets try using a different pin of the Arduino – say D7. Move the red jumper lead from pin D13 to pin D7 and modify the following line near the top of the sketch:

```
int led = 13;
```

so that it reads:

```
int led = 7;
```

Upload the modified sketch to your Arduino board and the LED should still be blinking, but this time using pin D7.

In the next lesson, we will be using LEDs again, this time, the Arduino will be controlling the LED.

https://adafru.it/aUx

https://adafru.it/aUx

About the Author

Simon Monk is author of a number of books relating to Open Source Hardware. The following books written by Simon are available from Adafruit: Programming Arduino (http://adafru.it/1019), 30 Arduino Projects for the Evil Genius (http://adafru.it/868) and Programming the Raspberry Pi (https://adafru.it/aM5).