The Potentiometer

In this guide, you'll be introduced to a new component and some new programming concepts. The component that you'll be working with is the *potentiometer*, sometimes known informally as a *pot*.

A *potentiometer* is a resistor that supports a range of values, and are usually adjusted by a knob. Volume controls on audio equipment are commonly potentiometers, so you've likely had experience interacting with this component.

Understanding how potentiometers work opens the door for several fun experiment ideas, such as being able to control LED brightness with the turn of a knob, or making a synthesizer where the knob controls the pitch!

The goal of this guide is to become familiar with using *analog input*: a range of input values from 0 to 1023. The potentiometer will be used to send this range of input into the Arduino.

You've already worked with *analog output* when you created the fading LED. Remember that the Arduino used *PWM* to rapidly turn an LED on and off, creating a fading effect.

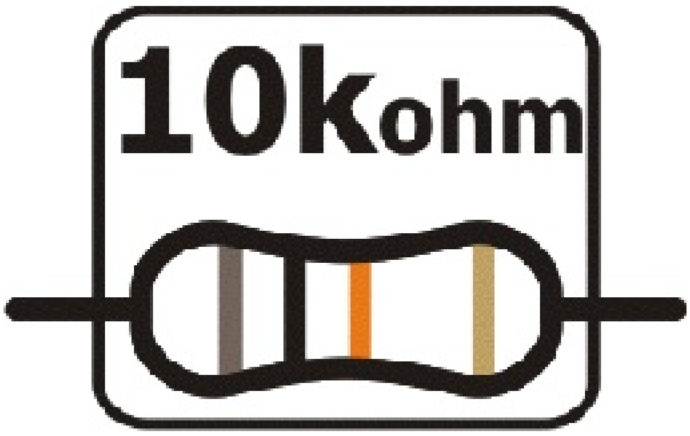
You've also worked with *digital input* with the pushbutton. The pushbutton used *digitalRead* to determine the state of the button, which could be either HIGH or LOW. With *digital input* there are only two possible states. *Analog input* has several supported values, and is used to read a *range* of input values!

The best way to understand this project is with a project, so let's get to the experiment!

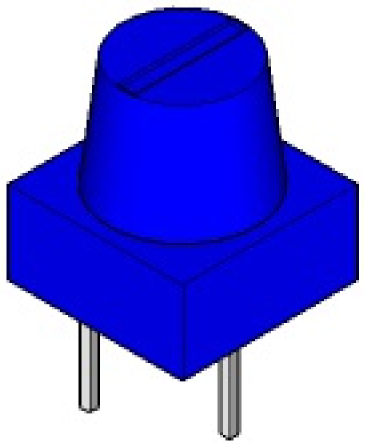
Project: Dialing in the LED

Supplies

Reminder: The 10k Ohm resistor is the brown/black/orange resistor. Here is an image:



- Uno and breadboard
- Jumper wires

- **(New!) Potentiometer**: The potentiometer is a knob. The one in your kit is a blue plastic knob, although other pots are sometimes metallic. Once the potentiometer is mounted, turning the knob will generate the range of input. Here is a picture of the one in your kit:
  - 
- 10k Ohm resistor for the potentiometer
- LED circuit supplies:

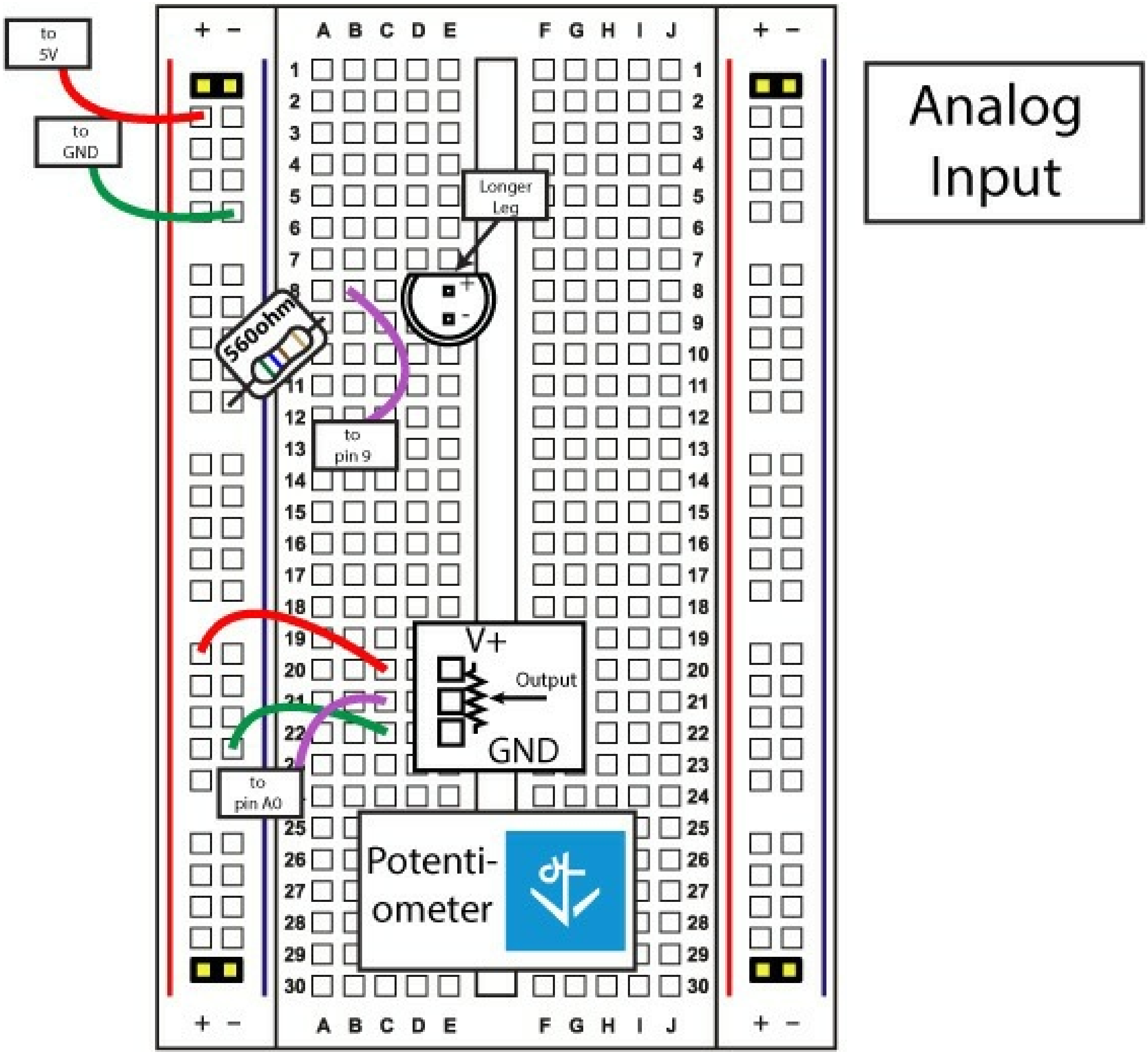
- LED and 220 (or 560) Ohm resistor

Steps

The project steps are broken down into the circuit setup and the sketch. Let's start with the circuit.

Circuit

Use this breadboard diagram as your guide:



1. Connect the 5V (Power) on the Arduino to the + rail of the breadboard.

2. Connect the GND on the Arduino to the - rail of the breadboard.
3. Wire up the LED circuit that you've previously done. If you need help, look back to one of the previous projects.
4. Attach the potentiometer to the breadboard. Place it so that the pins are closest (on the left side) to the Arduino. The edge of the potentiometer can hang over into the middle of the breadboard where the rails split.
5. Use 1 jumper wire to connect the power rail of the breadboard to the **V+ (top pin)** on the potentiometer.
6. Use 1 jumper wire to connect the output (middle pin) of the potentiometer to the **A0 analog input** on the Arduino board. Make sure to use the **A0** input and not a digital input.
7. Use 1 jumper wire to connect the **GND (bottom pin)** of the potentiometer.

Sketch

Create a new sketch with the following code and then upload it to your Arduino. For right now, go ahead and use the code below as a guide. Make sure to write the code yourself instead of copying it- this may seem like an unnecessary step, but actually taking the time to write the code will help you internalize it and get a better feel for how it's written.

There are two new commands in the code: **analogRead()** and **map()**. These will be explained in more detail below. For right now go ahead and just write the code in its entirety.

```
int ledPin=9; //Variable to store pin of LED
int potentPin=A0; //Variable to store pin of potentiometer
int potentValue=0; //Variable to store last known value of po
int brightnessValue=0; //Variable to store LED brightness
```

```
void setup() {
  // put your setup code here, to run once:
  pinMode(ledPin, OUTPUT); //Setup LED pin for output
}

void loop() {
  // put your main code here, to run repeatedly:
  potentValue=analogRead(potentPin); //Read the value of the
  brightnessValue=map(potentValue,0,1023,0,255); //Map the p
  analogWrite(ledPin,brightnessValue); //Set the brightness
}
```

New Commands

The sketch for this project introduced some new commands. Here's what they do:

analogRead()

analogRead(value) analogRead() is used to read the input from a pin and return a value between 0 and 1023.

map()

map(value, fromLow, fromHigh, toLow, toHigh) The map command re-maps a number from one range to another. The range of inputs returned from the potentiometer is 0 to 1023, whereas the range of brightness for the LED in the code is 0 to 255.

Why map()? You may notice an issue here: The maximum values for these ranges don't match. The potentiometer values range from 0 - 1023 whereas the LED accepts values from 0 - 255. The minimum values are both 0, but the maximums are different. Since this is the case, you need to **map** the value from the potentiometer to a value on the LED.

map() takes 5 inputs. Look at line 14 in the code:

Here is the parameters that map takes as input: map(value, fromLow, fromHigh, toLow, toHigh)

```
brightnessValue=map(potentValue,0,1023,0,255); //Map the pot
```

Remember, *potentValue* is a variable storing the last known value of the potentiometer. This is the value that will be mapped, and is the first parameter in the map() command.

The *fromLow* value is mapped to a value in *toLow*, the fourth input parameter. In the example, since the minimum values are the same for the potentiometer reading and the LED brightness, *fromLow* and *toLow* are both 0.

The *fromHigh* value is mapped to a value in *toHigh*, the fifth input parameter. Note that since the maximum values are different for the potentiometer reading and the LED brightness, the *fromHigh* and *toHigh* values are different.

Now that you've set this up via **map()**, all the values in between will be corresponding as well so that when you move the potentiometer, you'll then see changes in the LED brightness.

Going Further

This is just the basic outline of how to create a project using a potentiometer. Once you're ready, move on to the *Potentiometer Experimenter Challenge*!