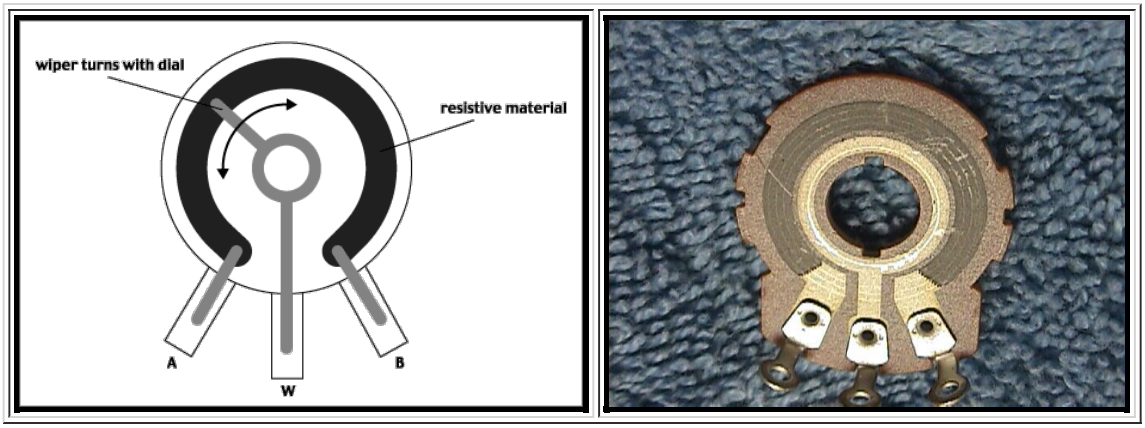
Arduino Lesson: Potentiometer and Analog Input

Setup a laptop, Arduino and USB cable as before. You will also need a small breadboard, some jumper wires, and one small potentiometer (aka "pot").

Potentiometer

A potentiometer is a device with (typically) three pins and a shaft which can be rotated and which is used to create a variable voltage.

Here is a diagram and a picture of the inside of one:



Typically, the outer two pins (marked "A" and "B" above) are connected to power and ground, and then a wiper (marked "W") moves around from one to the other.

When it is rotated all the way to the A pin, the W pin will have the same voltage as the A pin; and when it is rotated all the way to the B pin, the W pin will have the same voltage as the B pin; and in between, the W pin will have a voltage proportional to how far between A and B it is.

For example, at the half-way point it will have a voltage half-way between the voltages on the A and B pin.

Thus, if we connect our Arduino ground (0 volts) to the A pin, and power (5 volts) to the B pin, and then connect the W pin to a volt meter, it will show 0 volts when the wiper is rotated all the way to the left, 5 volts when rotated all the way around to the right, and 2.5 volts when rotated to the mid point. And rotated to the 3/4 point, it will be 3.75 volts (3/4 of 5); and so on.

When you have read and understand all of the above, sign off this line: _____

Power & Ground:

Most batteries label their terminals (connection points) "+" and "-", plus and minus.
In computer electronics we often use the words "Power" and "Ground" instead; they are almost exact synonyms for "plus" and "minus".
Power is + and Ground is -. We can think of the electric current as flowing out of the power terminal (+), and going around through the circuit and returning to the ground (-) terminal. _____

Find the Arduino pin labelled "5V". There is only one such pin. This is the power terminal. _____

Find the Arduino pins labeled "GND". There are several. These are all ground terminals. You can use any or all of them identically. _____

Digital vs. Analog Pins:

The Arduino has a set of pins on the right side labelled 0, 1, 2, ... through 13; these are all digital pins.

And on the left side of the Arduino board is a set of pins marked "A0", "A1", "A2", "A3", "A4", and "A5". These are analog pins.

For our purposes here, the major difference is that digital pins can only be zero or one, (HIGH or LOW, on or off), whereas analog pins can take on any value from 0 to just over 1000.

For example, you have already seen "digitalWrite(13,LOW);" and "digitalWrite(13,HIGH);" used to turn off and on an LED.

In this lesson we will experiment with the analog pins.

Find the Arduino pin labelled "A0". This is the first of the six analog input pins.

Find a colored jumper cable -- use any color except black or white. Connect it to the Arduino A0 pin, and leave the other end "hanging" where it will not touch anything.

Click on "**File --> Examples --> Basic --> AnalogReadSerial**" and upload it.

Startup the **Serial Monitor** (it's the icon in the upper right that looks like a magnifying glass. It says "Serial Monitor" when you mouse over it). Once you have started it up you should see a new window appear with numbers spewing out at a high rate. They will range from as low as 0 to as high as 1000. (Actually as high as 1023).

Watch the values being printed out. These numbers represent the voltage at the tip of the A0 jumper wire. At the moment, it is "hanging" and the voltage numbers you are seeing are just what that wire is picking up from the air. (There is electricity all around us from radio waves, power wires in the walls, and even from nearby human beings.

Hold the end of the A0 jumper between two fingers. Do the numbers change just by touching the wire? Let go and watch the numbers change again. Let go and hold repeatedly and observe the changing values. Your body is acting as a source of electricity! The numbers may change even when you just put your hand ***near*** the wire. Try *licking* your fingers first!

Now, push the free end of the A0 jumper into one of the Arduino GND pins. The numbers should go to zero immediately!

Now, remove the free end from GND and push it into the 5V pin instead. The numbers will go to 1023 immediately! 1023 is the largest number our Arduinos can measure, and represents 5 volts, the highest voltage our Arduinos use. WARNING: The jumper should now be going from A0 to 5V, NOT from GND to 5V. If you do that you will *fry* the Arduino almost immediately! If you feel the wire get warm, rip it out fast!

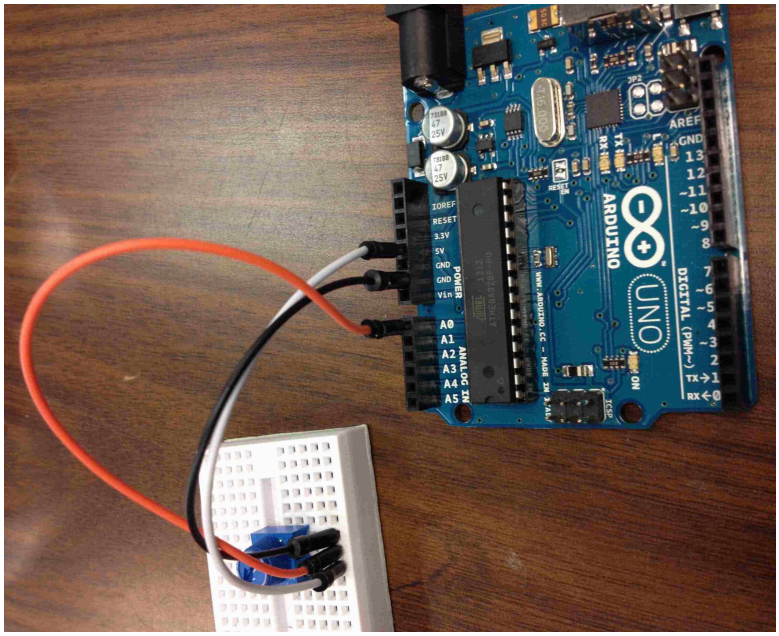
Now get a breadboard and a potentiometer. Insert the Potentiometer so that each of its 3 pins are in a separate breadboard row. (See the picture below). Be careful not to bend or break any of the tiny pins of the pot!

Insert the end of the A0 jumper wire into the breadboard in the same row as the center pin of the potentiometer.

Get a white jumper wire. Insert one end of it into the same row as either of the outer pins of the pot. Insert the other end of the wire into the Arduino 5V power pin.

Get a black jumper wire. Insert one end of it into the same row as the OTHER outer pin of the pot. Insert the other end of the wire into an Arduino ground pin.

It should look like this,
with the white power and black ground pin jumpers on the outside pins of the pot,
and the colored A0 jumper in the center pin of the pot.



Now, with the "AnalogReadSerial" program still running, watch the numbers as you turn the pot! What do you observe?

(Note: the very tiny pots we have are hard to turn and tend to come loose from the breadboard. Deal with it! ;-)

When you rotate it all the way toward the ground pin, its middle pin will deliver 0 volts and print 0. When you turn it all the way toward the power pin, its middle pin will deliver 5 volts and print 1023. And in between those two ends, as the voltage varies between 0 and 5 volts, and the numbers will vary between 0 and 1023.

Can you make it print exactly 500? 250? 750?

Get it? Got it? Good!

END OF LESSON 3!

A FURTHER COOL PROGRAM MODIFICATION:

In the AnalogReadSerial program, find the line that says:

Serial.println(sensorValue);

And change it to this:

```
Serial.print(sensorValue/204.8);
for (int i=0 ; i<sensorValue/6 ; i++) { Serial.print(" "); }
Serial.println("*");
```

Be very careful. Every semicolon and paren is important! Case matters. Also note that the middle Serial.print has a blank space between the two quotes. This is important! And the last line says "println" not "print" in it, that "ln" stands for "new line".

That is all. This bit of "code" prints out the sensor value as a "wave" on the screen. Perhaps you can see why they call it "code" now! It's more than a bit obscure! A full explanation of the above three lines is beyond the scope of this course! For now, just copy it and use it!

==========
Math nerds only:

If you are curious about the math... The analog "sensorValue" goes from 0 to 1023 as the pot sweeps from ground to full power. 1023 is one less than 1024, which is 2 to the 10th power. The analog sensors on Arduino are 10 bit sensors, and the maximum value you can get in 10 bits is 1023.

But the actual voltages go from 0 to 5. The value 204.8 equals 1024/5, and so while sensorValue goes from 0 to 1023, sensorValue/204.8 goes from 0 to 5.

And then why divide it by 6 in the second line? This code prints "sensorValue/6" spaces followed by an asterisk. Most laptops only have about 170 or less characters per line. The maximum value of sensorValue is 1023, and 1023 spaces would overflow our screen lines! But 1023/6 is about 170. If your computer screen is narrower than 170 columns, you can increase the 6 to 7 or 8, and if you have a wider screen, you can decrease the 6 to 5 or less. Get it? Got it? Good!