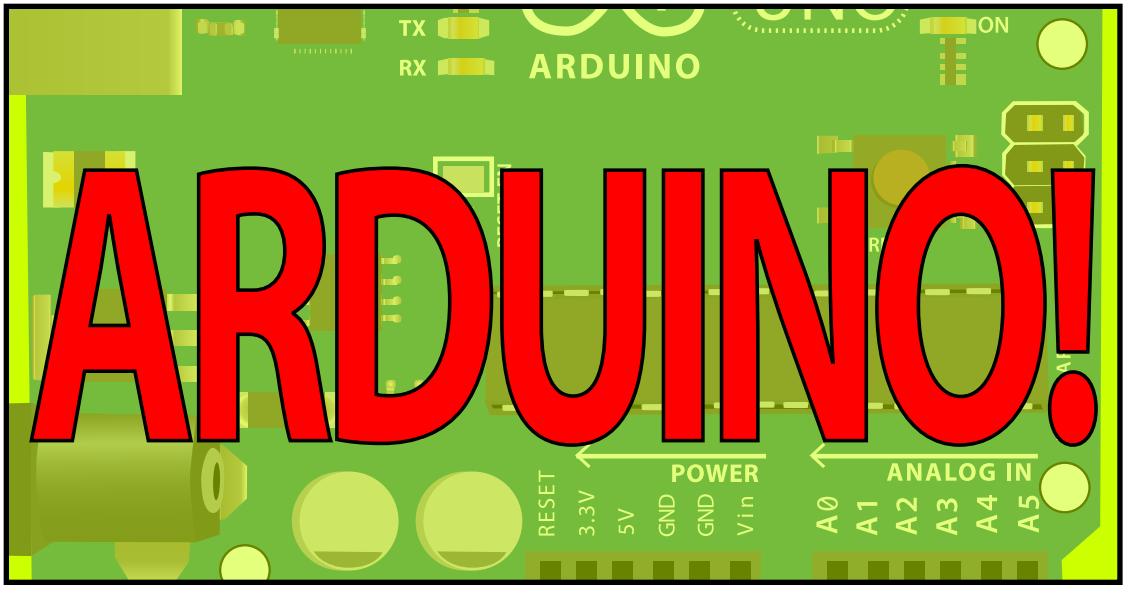
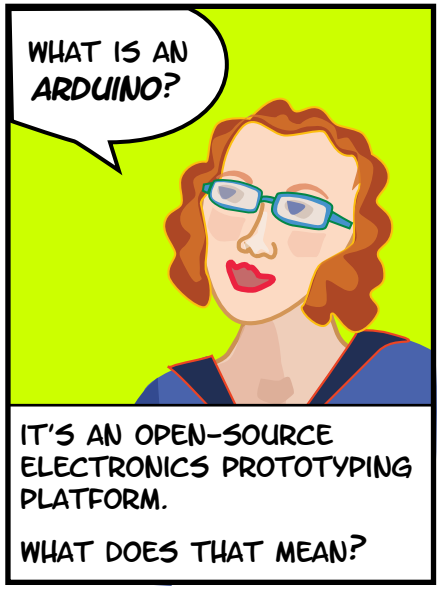


ARDUINO!

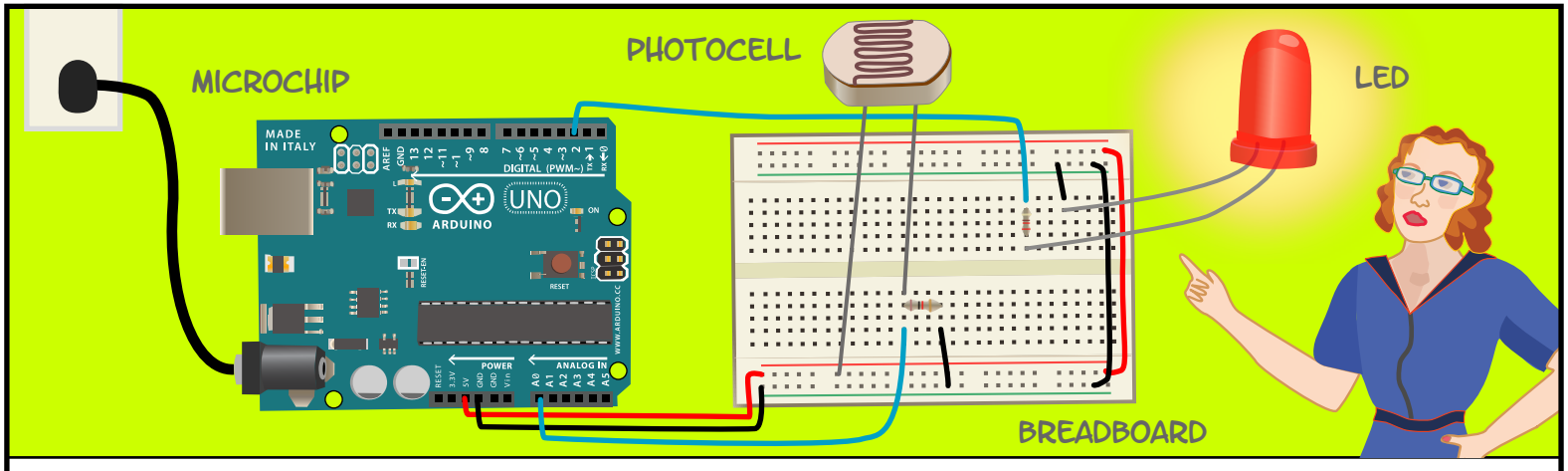


OPEN SOURCE- "RESOURCES THAT CAN BE USED, REDISTRIBUTED OR REWRITTEN FREE OF CHARGE. OFTEN SOFTWARE OR HARDWARE."

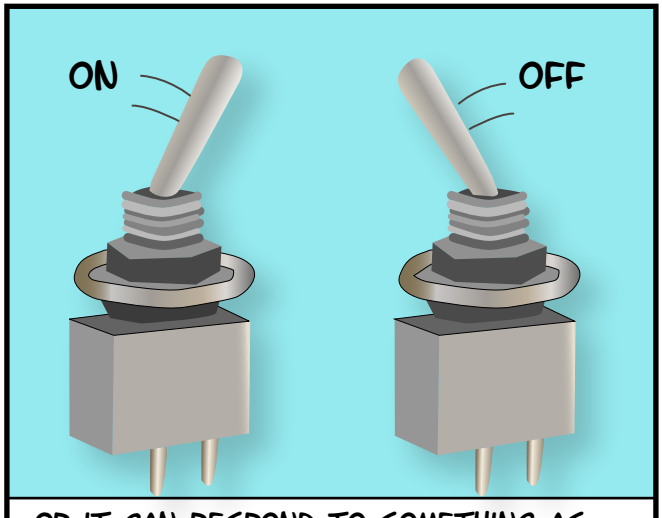
ELECTRONICS- "TECHNOLOGY WHICH MAKES USE OF THE CONTROLLED MOTION OF ELECTRONS THROUGH DIFFERENT MEDIA."

PROTOTYPE- "AN ORIGINAL FORM THAT CAN SERVE AS A BASIS OR STANDARD FOR OTHER THINGS."

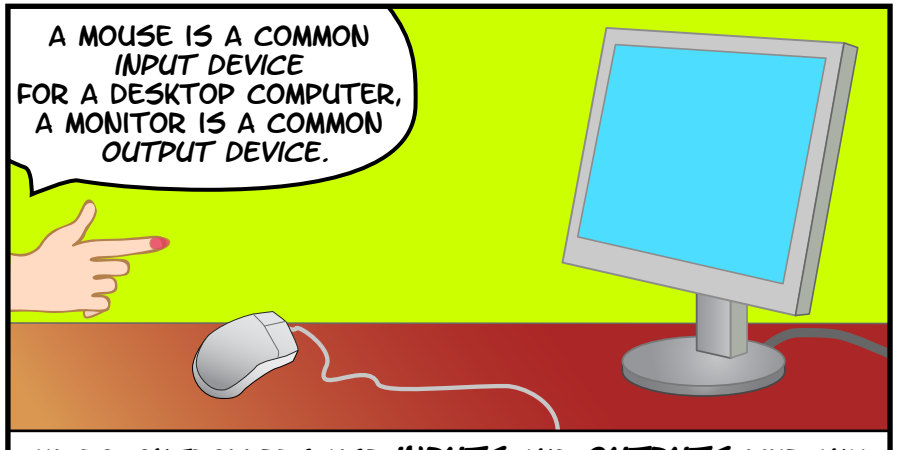
PLATFORM- "HARDWARE ARCHITECTURE WITH SOFTWARE FRAMEWORK ON WHICH OTHER SOFTWARE CAN RUN."



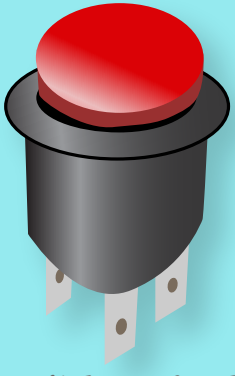
AN ARDUINO IS A ***MICROCHIP***, WHICH IS A VERY SMALL COMPUTER THAT YOU CAN PROGRAM TO RESPOND TO THINGS. IT CAN MEASURE CONDITIONS (LIKE HOW MUCH LIGHT THERE IS IN THE ROOM). IT CAN CONTROL HOW OTHER OBJECTS REACT TO THOSE CONDITIONS (ROOM GETS DARK AND AN LED TURNS ON).



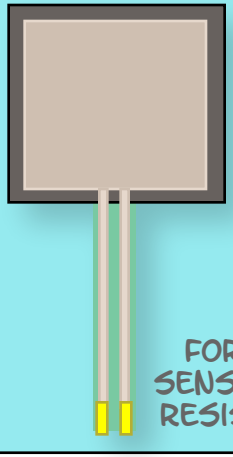
OR IT CAN RESPOND TO SOMETHING AS SIMPLE AS THE PRESS OF A SWITCH.



MICROCONTROLLERS USE ***INPUTS*** AND ***OUTPUTS*** LIKE ANY COMPUTER. INPUTS CAPTURE INFORMATION FROM THE USER OR THE ENVIRONMENT WHILE OUTPUTS DO SOMETHING WITH THE INFORMATION THAT HAS BEEN CAPTURED.



MOMENTARY SWITCH



FORCE SENSITIVE RESISITOR

A SWITCH OR A SENSOR COULD BE AN INPUT INTO THE ARDUINO.

DC MOTOR

ANY OBJECT WE WANT TO TURN ON AND OFF AND CONTROL COULD BE AN OUTPUT. IT COULD BE A MOTOR OR EVEN A COMPUTER.

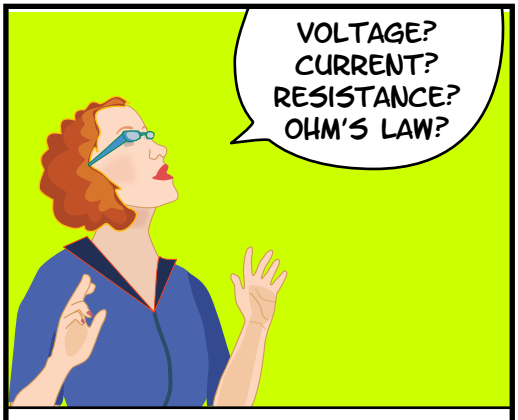
WHATS THE DIFFERENCE BETWEEN DIGITAL AND ANALOG INPUTS AND OUTPUTS?

INPUTS AND OUTPUTS CAN BE ***DIGITAL*** OR ***ANALOG***. DIGITAL INFORMATION IS BINARY- IT IS EITHER TRUE OR FALSE. ANALOG INFORMATION IS CONTINUOUS, IT CAN HOLD A RANGE OF VALUES.

DIGITAL INFORMATION IS ***DISCRETE*** AND ***FINITE***. ALL INFORMATION IS DESCRIBED IN ***TWO STATES***, *1* OR *0*, ***ON*** OR ***OFF***.

ANALOG INFORMATION IS CHARACTERIZED BY ITS ***CONTINUOUS*** NATURE. IT CAN HAVE AN ***INFINITE*** NUMBER OF POSSIBLE VALUES.

A SWITCH IS A DIGITAL INPUT, A SENSOR IS AN ANALOG INPUT. THE RANGE OF AN ANALOG SENSOR IS LIMITED BY ITS CONVERSION TO DIGITAL DATA.



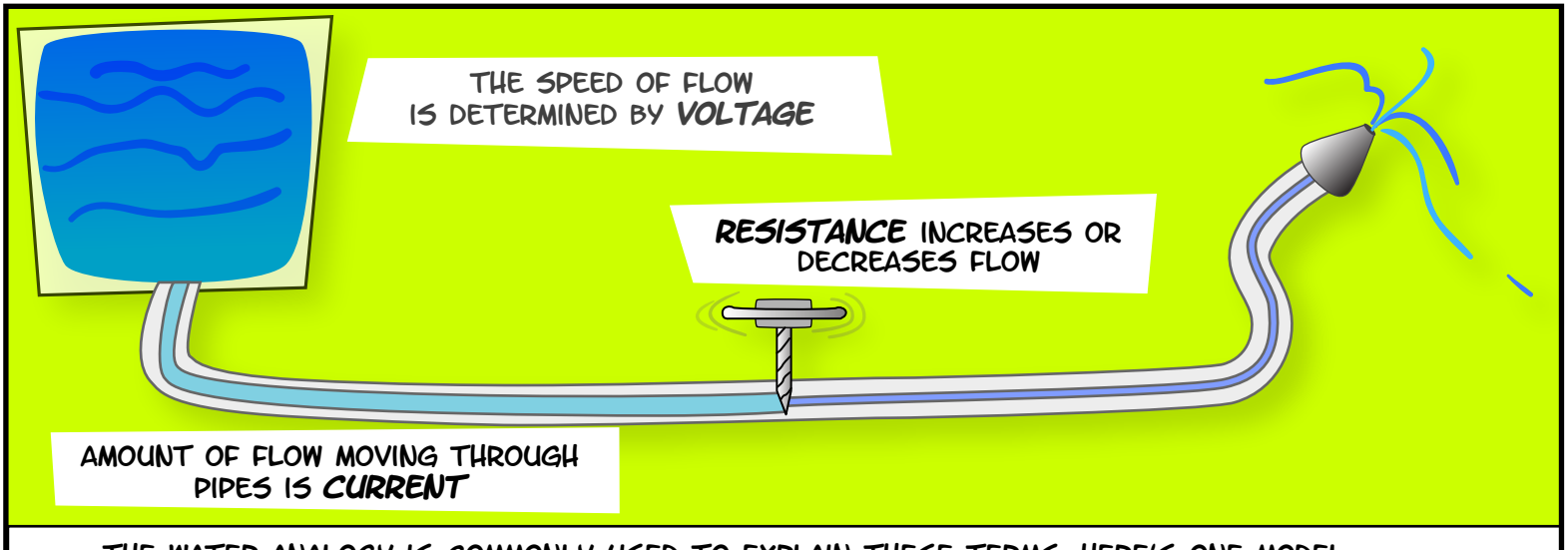
BEFORE WE PLUG IN THE ARDUINO, WE WILL REVIEW A FEW TERMS AND PRINCIPLES THAT HAVE TO DO WITH HOW ELECTRICITY (AND THEREFORE ELECTRONICS) WORKS.

VOLTAGE (V) IS A MEASURE OF ELECTRICAL POTENTIAL. IT IS MEASURED IN **VOLTS.**

CURRENT (I) IS THE AMOUNT OF FLOW THROUGH A CONDUCTIVE MATERIAL. IT IS MEASURED IN **AMPERES** OR **AMPS.**

RESISTANCE (R) IS A MATERIAL'S OPPOSITION TO THE FLOW OF ELECTRIC CURRENT. IT IS MEASURED IN **OHMS.**

ELECTRICITY IS THE FLOW OF ENERGY THROUGH A CONDUCTIVE MATERIAL.



THE WATER ANALOGY IS COMMONLY USED TO EXPLAIN THESE TERMS. HERE'S ONE MODEL.

OHM'S LAW

CURRENT = VOLTAGE/RESISTANCE
(I = V/R)

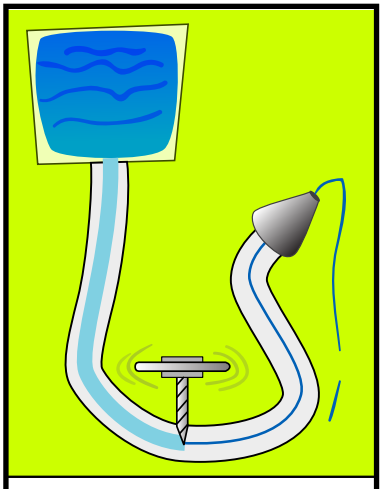
OR

RESISTANCE = VOLTAGE/CURRENT
(R = V/I)

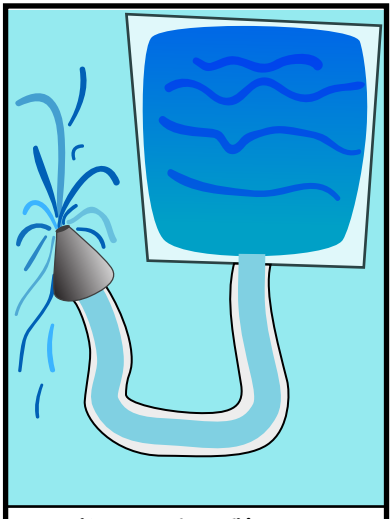
OR

VOLTAGE = RESISTANCE * CURRENT
(V = R*I)

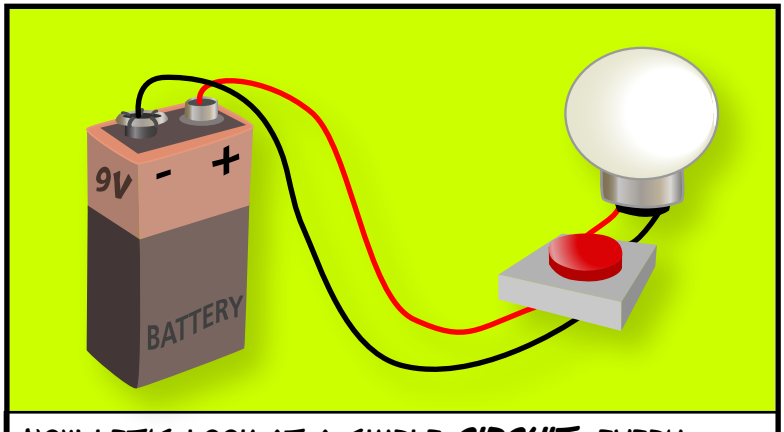
THERE IS A RELATIONSHIP BETWEEN VOLTAGE, CURRENT AND RESISTANCE, DISCOVERED BY GEORG OHM, A GERMAN PHYSICIST.



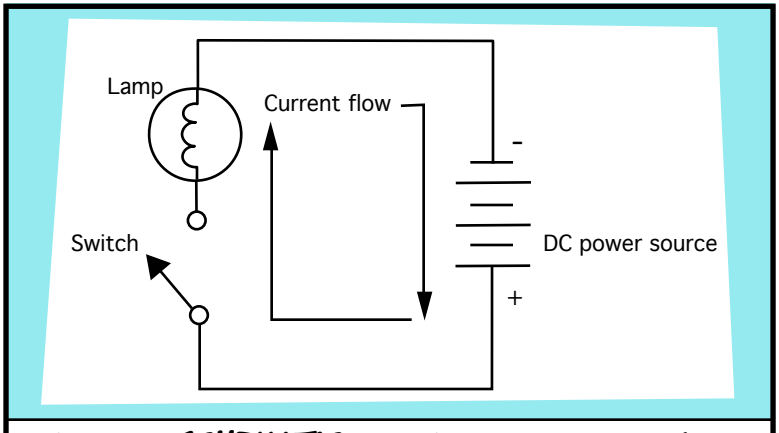
FOR EXAMPLE, INCREASE THE RESISTANCE, LESS FLOW.



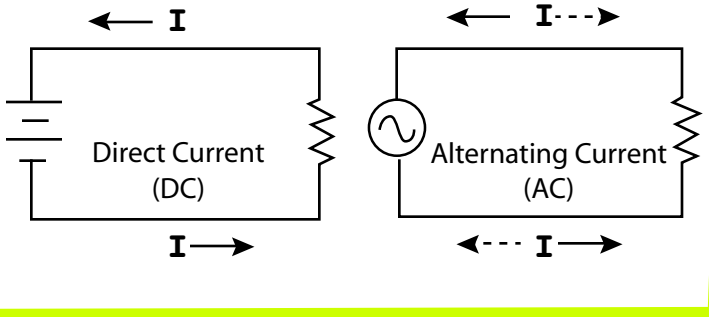
OR INCREASE THE POTENTIAL, MORE FLOW.



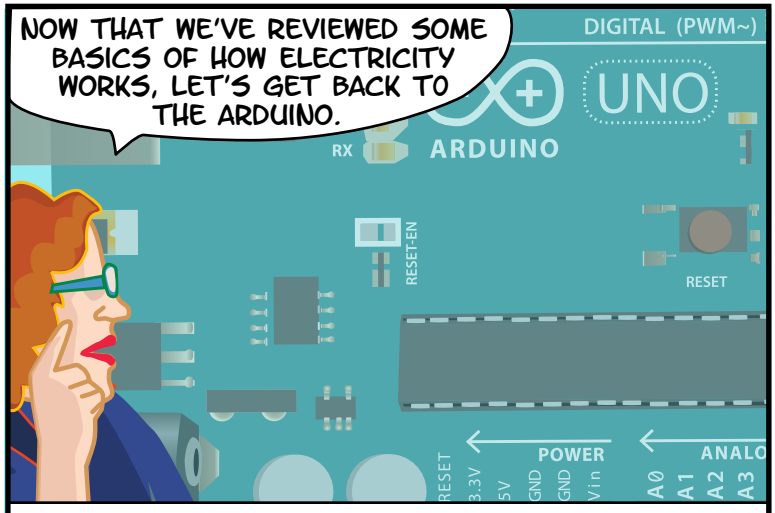
NOW LET'S LOOK AT A SIMPLE ***CIRCUIT***. EVERY CIRCUIT IS A CLOSED LOOP THAT HAS AN ***ENERGY SOURCE*** (BATTERY) AND A ***LOAD*** (LAMP). THE LOAD CONVERTS THE ELECRICAL ENERGY OF THE BATTERY AND USES IT UP. THIS ONE HAS A SWITCH TOO.



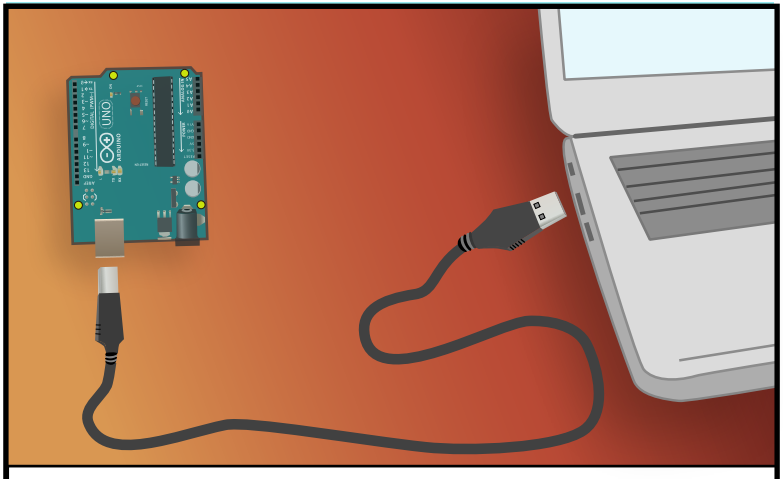
THIS IS A ***SCHEMATIC*** OF THE SAME CIRCUIT (IT REPRESENTS THE CIRCUIT USING SYMBOLS FOR THE ELECTRONIC COMPONENTS). WHEN THE SWITCH IS CLOSED, CURRENT FLOWS FROM THE POWER SOURCE AND LIGHTS THE LAMP.



THERE ARE TWO COMMON TYPES OF CIRCUITS, ***DIRECT CURRENT*** AND ***ALTERNATING CURRENT***. IN A DC CIRCUIT, THE CURRENT ALWAYS FLOWS IN ONE DIRECTION. IN AC, THE CURRENT FLOWS IN OPPOSITE DIRECTIONS IN REGULAR CYCLES. WE WILL ONLY TALK ABOUT DC CIRCUITS HERE.



THE ARDUINO WILL NEED POWER TO RUN. WE WILL NEED TO ATTACH IT TO A COMPUTER TO PROGRAM IT.



ATTACHING THE ARDUINO TO A COMPUTER WITH A USB CABLE WILL SUPPLY THE 5 VOLTS OF POWER WE NEED AND ALLOW US TO START PROGRAMMING.

YOU'LL HAVE TO DOWNLOAD AND INSTALL SOFTWARE TO PROGRAM THE ARDUINO. IT IS AVAILABLE FROM THE URL ABOVE FREE OF CHARGE. THE ARDUINO SOFTWARE RUNS ON THE MAC OS X, WINDOWS AND LINUX PLATFORMS.

FOR INSTRUCTIONS ON HOW TO INSTALL ARDUINO SOFTWARE ON A MAC:

HTTP://WWW.ARDUINO.CC/EN/GUIDE/MACOSX

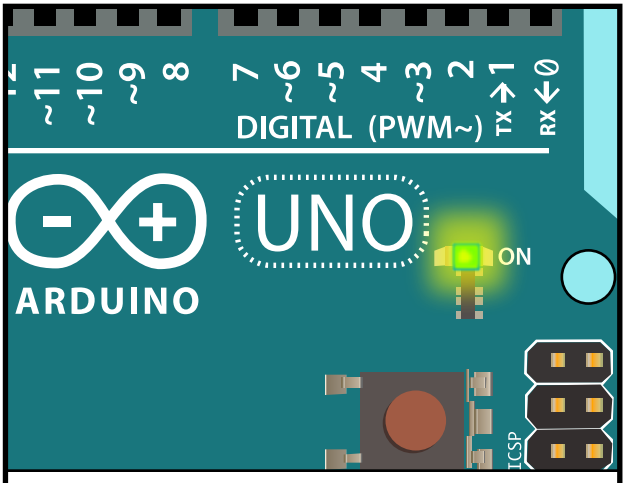
FOR INSTRUCTIONS ON HOW TO INSTALL ON WINDOWS:

HTTP://WWW.ARDUINO.CC/EN/GUIDE/WINDOWS

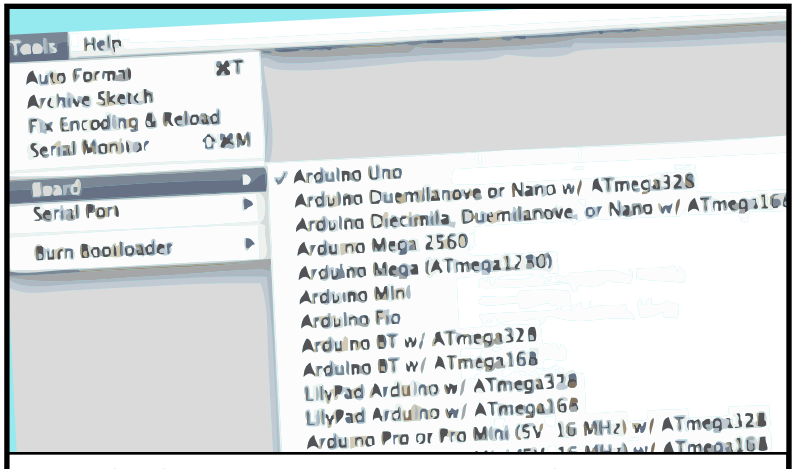
FOR INSTRUCTIONS ON HOW TO INSTALL ON LINUX:

HTTP://WWW.ARDUINO.CC/PLAYGROUND/LEARNING/LINUX

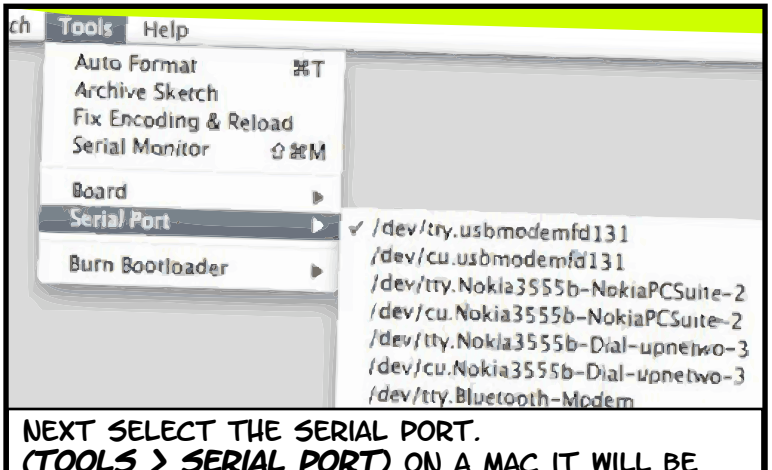
GO TO THE URLS ABOVE FOR DETAILED INSTRUCTIONS ON INSTALLING THE SOFTWARE ON THESE PLATFORMS.



WHEN YOU HAVE INSTALLED THE SOFTWARE, CONNECT THE ARDUINO. AN LED MARKED *ON* SHOULD LIGHT UP ON THE BOARD.

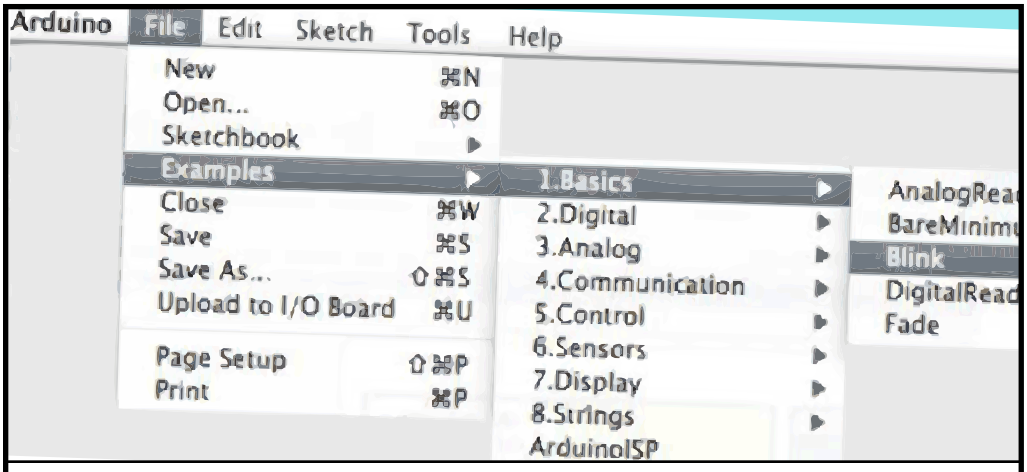


LAUNCH THE ARDUINO SOFTWARE. IN THE TOOLS MENU, SELECT THE BOARD YOU ARE USING *(TOOLS > BOARD)*. FOR EXAMPLE, *ARDUINO UNO*.

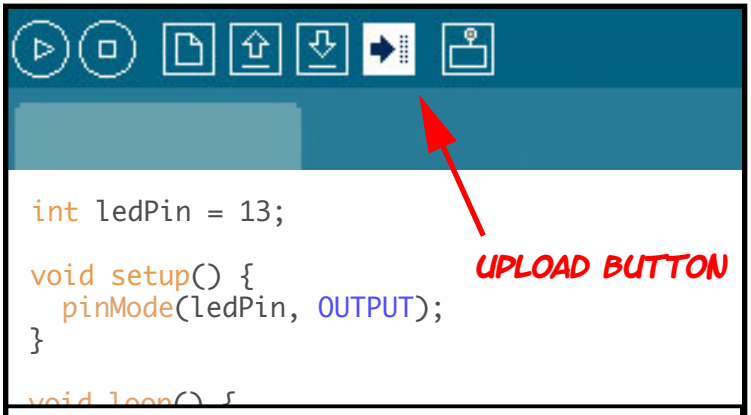


NEXT SELECT THE SERIAL PORT. *(TOOLS > SERIAL PORT)* ON A MAC IT WILL BE SOMETHING LIKE */DEV/TTY.USBMODEM*. ON A WINDOWS MACHINE, IT WILL BE *COM3* OR SOMETHING LIKE THAT.

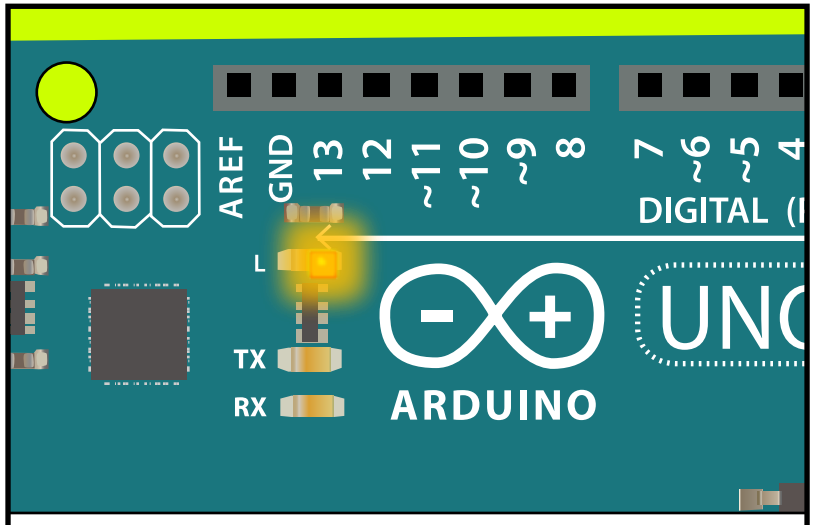
WHEN YOU DOWNLOADED THE ARDUINO SOFTWARE, YOU DOWNLOADED AN ***IDE***. IT COMBINES A TEXT EDITOR WITH A COMPILER AND OTHER FEATURES TO HELP PROGRAMMERS DEVELOP SOFTWARE.



THE ARDUINO IDE ALLOWS YOU TO WRITE ***SKETCHES***, OR ***PROGRAMS*** AND UPLOAD THEM TO THE ARDUINO BOARD. OPEN THE ***BLINK*** EXAMPLE IN THE FILE MENU. FILE > EXAMPLES > 1.BASICS > BLINK.



TO UPLOAD THE SKETCH TO THE ARDUINO BOARD, CLICK THE ***UPLOAD BUTTON*** ON THE STRIP OF BUTTONS AT THE TOP OF THE WINDOW. SOME MESSAGES WILL APPEAR IN THE BOTTOM OF THE WINDOW, FINALLY ***DONE UPLOADING***.



THE LED AT PIN 13 ON THE ARDUINO STARTS BLINKING.

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has LED connected on most Arduino boards
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

A SKETCH, LIKE A PROGRAM WRITEN IN ANY LANGUAGE, IS A SET OF INSTRUCTIONS FOR THE COMPUTER. IF WE LOOK CLOSELY AT THE BLINK SKETCH, WE SEE THERE ARE 2 MAJOR PARTS, ***SETUP*** AND ***LOOP***.

SETUP: HAPPENS ONE TIME WHEN PROGRAM STARTS TO RUN

LOOP: REPEATS OVER AND OVER AGAIN

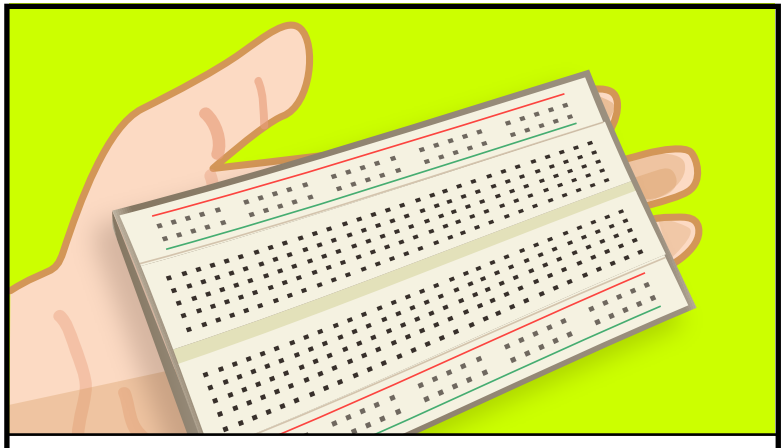
THESE ARE BOTH BLOCKS OF CODE CALLED ***FUNCTIONS*** THAT EVERY SKETCH WILL HAVE. THEY ARE BLOCKED OUT BY CURLY BRACES { }.

HTTP://ARDUINO.CC/EN/REFERENCE/HOMEPAGE

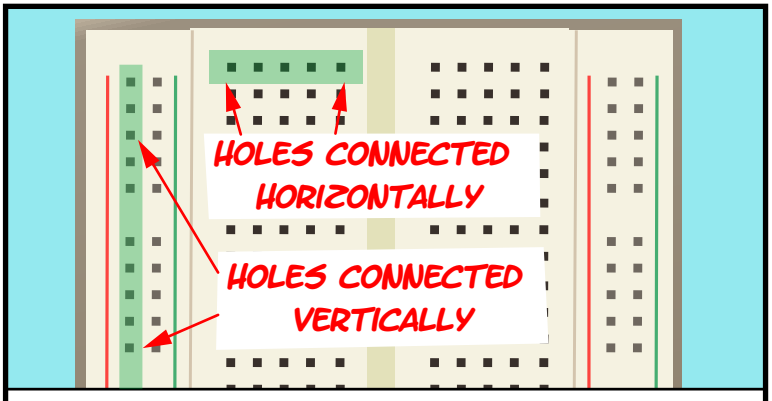
CHECK OUT THE ARDUINO WEBSITE FOR THE ARDUINO REFERENCE GUIDE AND MANY OTHER RESOURCES TO LEARN THE LANGUAGE.

```
void setup() {      //DECLARES BLOCK OF CODE
  pinMode(13, OUTPUT); //SETS PIN 13 TO OUTPUT
}                     //END BLOCK OF CODE


void loop() {      //DECLARES BLOCK OF CODE
  digitalWrite(13, HIGH);  //SETS PIN 13 HIGH
  delay(1000);              //PAUSE 1 SECOND
  digitalWrite(13, LOW);   //SETS PIN 13 LOW
  delay(1000);              //PAUSE 1 SECOND
}                          //END BLOCK OF CODE
```

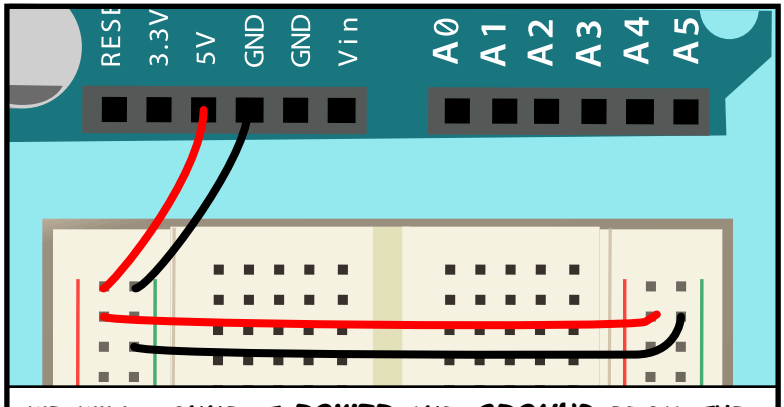
FOR NOW, LET'S LOOK AT THIS SIMPLE SCRIPT LINE BY LINE & SEE WHAT EACH LINE DOES.



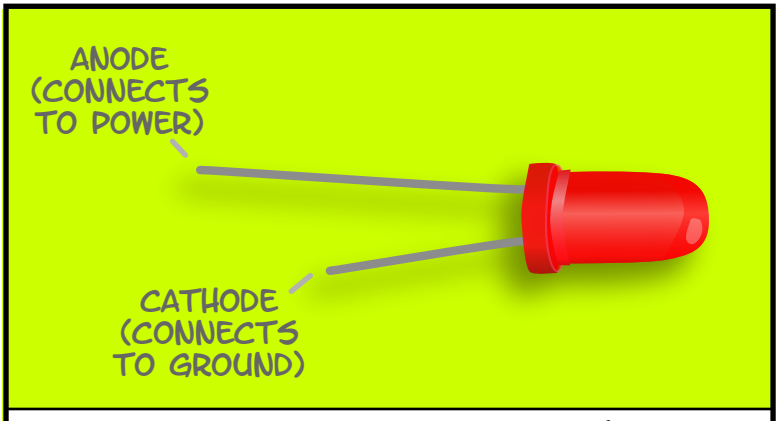
HOW DO WE CONTROL OBJECTS THAT ARE NOT ON THE ARDUINO BOARD? WE WILL CONNECT THE ARDUINO TO A ***SOLDERLESS BREADBOARD***. THIS WILL allow US TO QUICKLY SET UP AND TEST CIRCUITS.



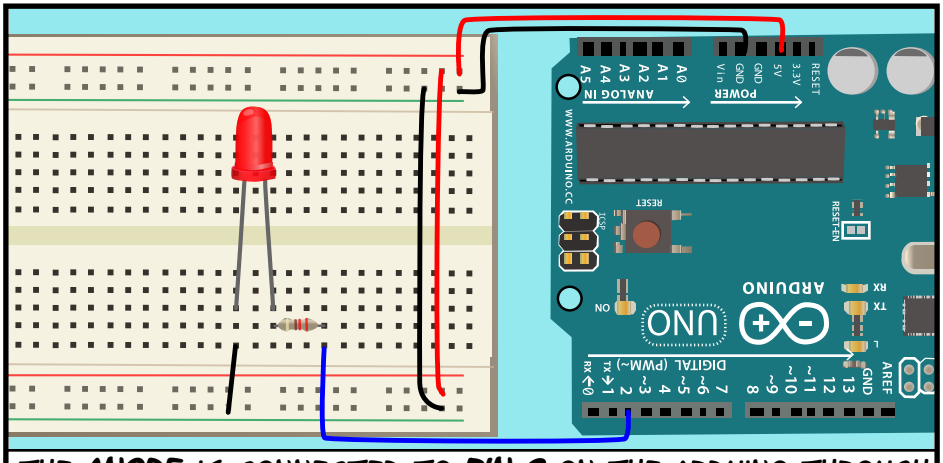
THIS BREADBOARD HAS 2 ROWS OF HOLES RUNNING DOWN THE LEFT AND RIGHT SIDE, AND 5 ROWS OF HOLES ON EITHER SIDE OF A MIDDLE INDENTATION. THE SIDE ROWS ARE CONNECTED ***VERTICALLY***, EACH ROW OF 5 HOLES IN THE MIDDLE ARE CONNECTED ***HORIZONTALLY***.



WE WILL CONNECT ***POWER*** AND ***GROUND*** FROM THE ARDUINO BOARD TO THE VERTICALLY CONNECTED STRIPS ON THE LEFT AND RIGHT WITH 22 GAUGE WIRE. OTHER COMPONENTS CAN BE ATTACHED TO THE HOLES IN THE MIDDLE AND TO POWER AND GROUND AS NEEDED.



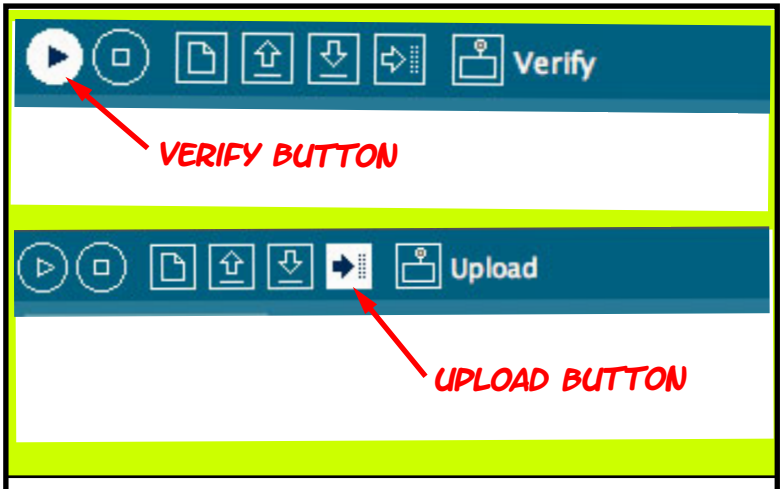
WHEN CURRENT FLOWS THROUGH A ***LED (LIGHT EMITTING DIODE)*** IN THE RIGHT DIRECTION, IT LIGHTS UP. WE'LL ATTACH AN LED TO THE BREADBOARD, THEN TO THE ARDUINO SO WE CAN CONTROL IT WITH CODE.



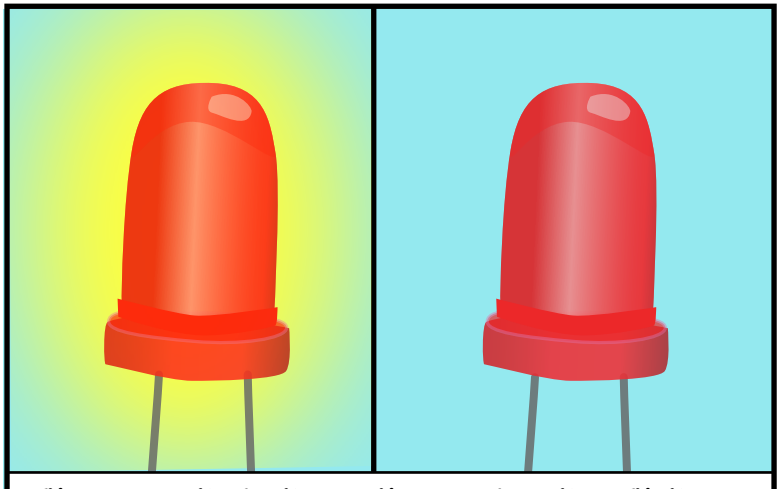
THE **ANODE** IS CONNECTED TO **PIN 2** ON THE ARDUINO THROUGH A **220 OHM RESISTOR**. THE **CATHODE** IS CONNECTED TO **GROUND**. PINS 2 THROUGH 13 CAN BE CONFIGURED AS DIGITAL INPUTS OR OUTPUTS. CLICK **NEW** BUTTON TO START A SKETCH.

```
void setup() {
  pinMode(2, OUTPUT);
}

void loop() {
  digitalWrite(2, HIGH);
  delay(500);
  digitalWrite(2, LOW);
  delay(500);
}
```

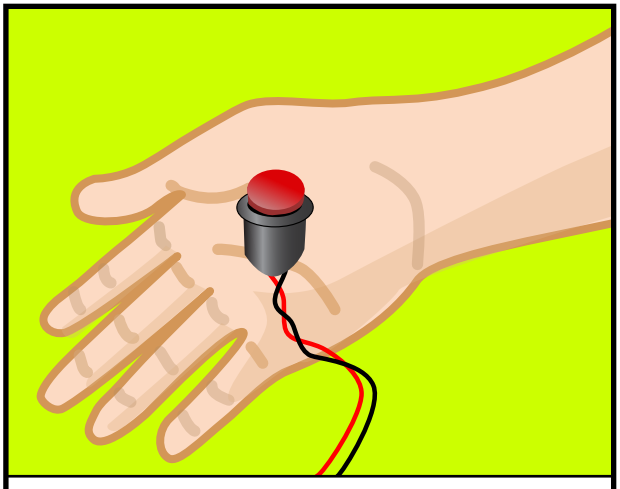
IN **SETUP**, WE SET PIN 2 TO BE AN OUTPUT. IN **LOOP**, FIRST WE SET PIN 2 HIGH WHICH LIGHTS THE LED. DELAY PAUSES 500 MILLISECONDS, OR HALF A SECOND. WHEN PIN 2 IS SET LOW, THE LED GOES OFF, WE PAUSE ANOTHER HALF SECOND.



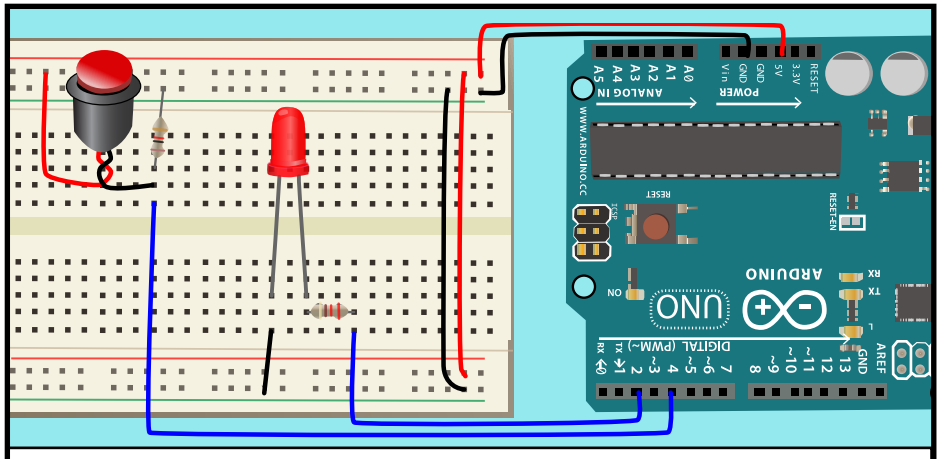
CLICK VERIFY ON THE MENU TO CHECK YOUR CODE. IF THERE AREN'T ANY ERRORS, CLICK UPLOAD TO PUT YOUR PROGRAM ON THE ARDUINO.



THE LED BLINKS ON FOR HALF A SECOND, THEN BLINKS OFF FOR HALF A SECOND, OVER AND OVER AGAIN.



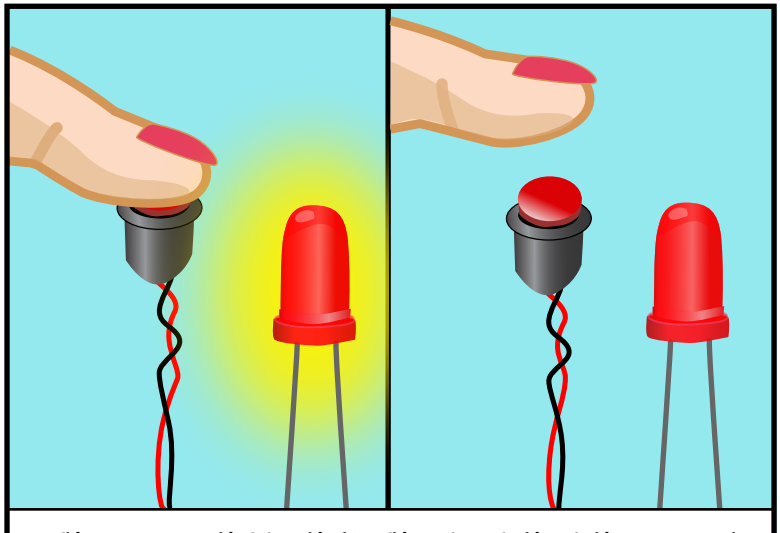
NEXT WE WILL ADD A SWITCH, A DIGITAL INPUT, SO WE CAN TURN THE LED OFF AND ON.



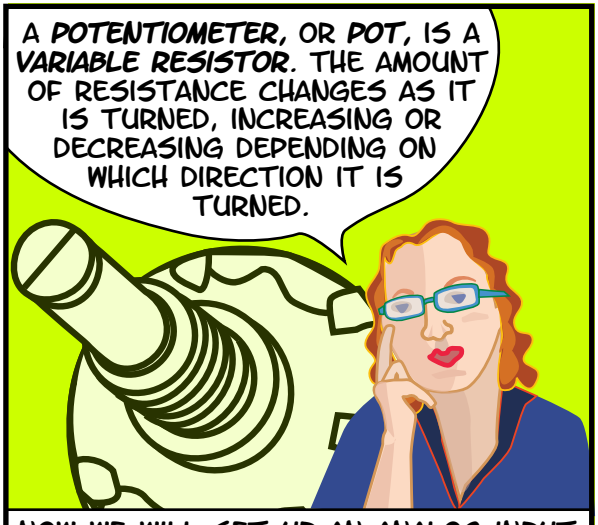
CONNECT ONE END OF A MOMENTARY SWITCH TO PIN 4 ON THE ARDUINO, WITH A 10K RESISTOR CONNECTED TO GROUND ATTACHED TO THE SAME END. ATTACH THE OTHER END TO POWER. WE WILL LEAVE THE LED ATTACHED TO THE SAME PIN.

```
void setup() {
  pinMode(2, OUTPUT);
  pinMode(4, INPUT);
}

void loop() {
  if(digitalRead(4)){
    digitalWrite(2, HIGH);
  }else{
  digitalWrite(2, LOW);
}
```

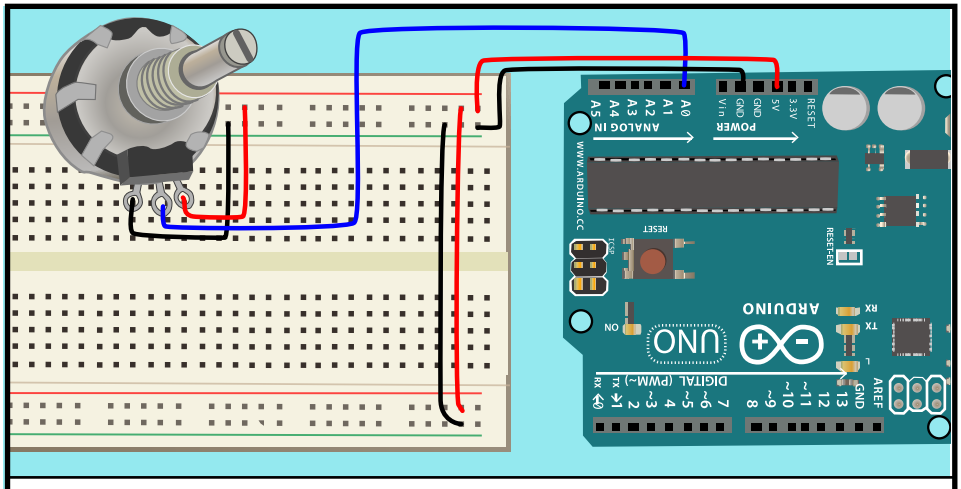
NEXT WE'LL WRITE THE CODE. IN SETUP, WE DECLARE PIN 2 AN OUTPUT AND PIN 4 AN INPUT. IN LOOP, WE USE AN IF STATEMENT, IF WE READ PIN 4 AS HIGH, WE SET THE LED PIN TO HIGH, OTHERWISE WE SET THE LED PIN TO LOW, TURNING IT OFF.



THE LED LIGHTS WHEN THE SWITCH IS HELD DOWN.



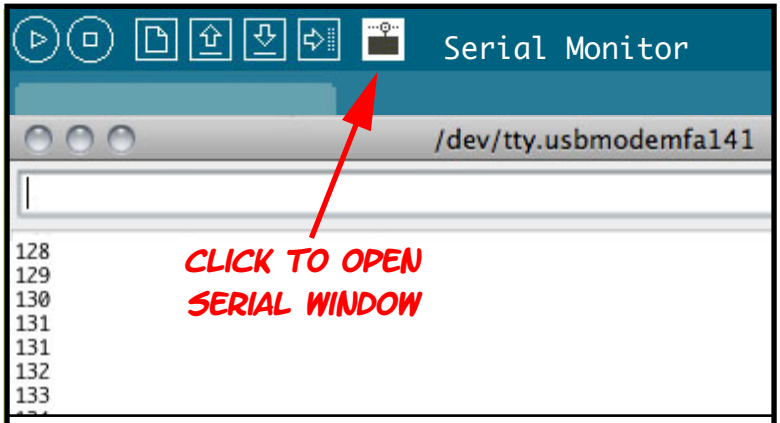
NOW WE WILL SET UP AN ANALOG INPUT. WE'LL USE A POTENTIOMETER.



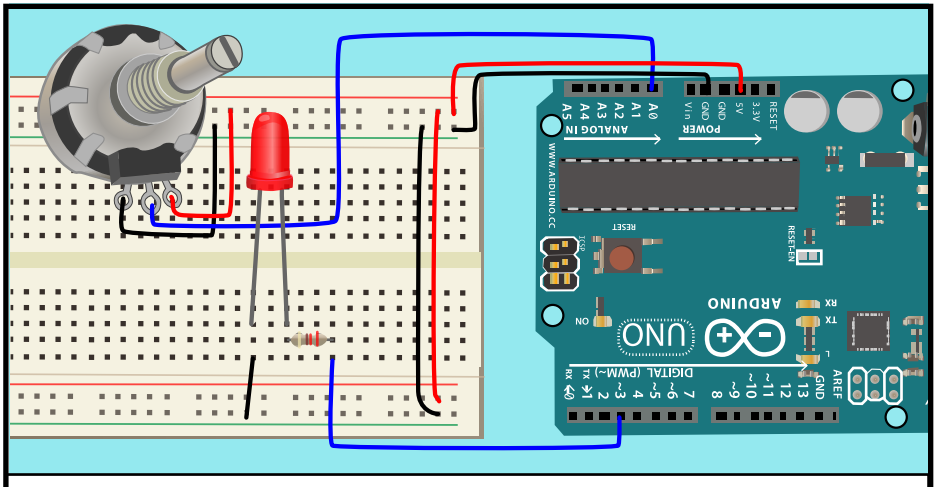
ATTACH THE MIDDLE PIN ON THE POTENTIOMETER TO ANALOG PIN A0. ATTACH ONE END OF THE POT TO POWER, THE OTHER TO GROUND.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println(analogRead(A0));
}
```

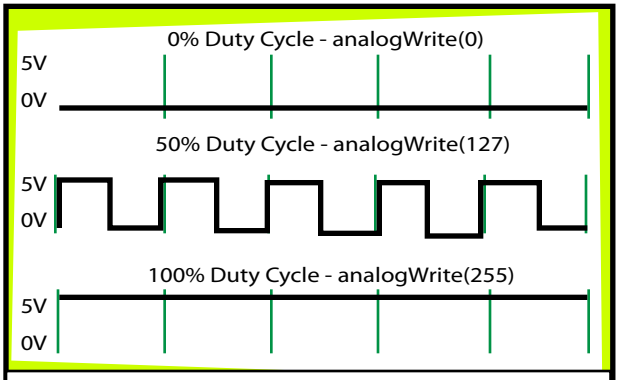
FIRST WE WILL LOOK AT THE RANGE OF VALUES WE GET BY TURNING THE POT USING THE ***SERIAL MONITOR***. IN OUR CODE, WE INITIALIZE THE SERIAL OBJECT IN SETUP, SETTING A BAUD RATE OF 9600. IN LOOP, WE READ THE VALUE FROM ANALOG PIN A0 AND PRINT IT TO THE SERIAL OBJECT USING THE ***PRINTLN*** FUNCTION,



AFTER YOU HAVE UPLOADED THE SCRIPT TO THE ARDUINO, CLICK THE ***SERIAL MONITOR*** BUTTON IN ORDER TO SEE THE VALUES AS YOU TURN THE POT. A WINDOW WILL OPEN, AND YOU WILL SEE VALUES RANGING FROM 0 TO 1024 AS THE POT IS TURNED.



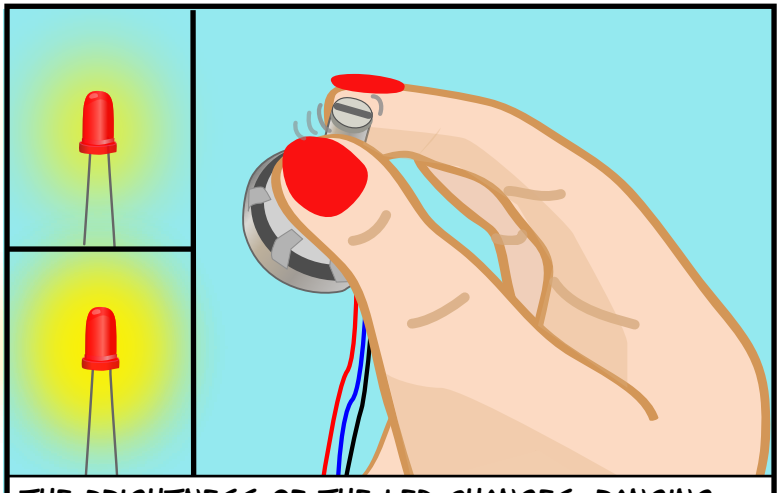
LET'S USE THE CHANGING VALUES WE RECEIVE FROM THE POT AS A DIMMER TO LIGHT AN LED. PUT THE LED BACK INTO THE BOARD, ATTACHED TO THE ARDUINO AT PIN 3.



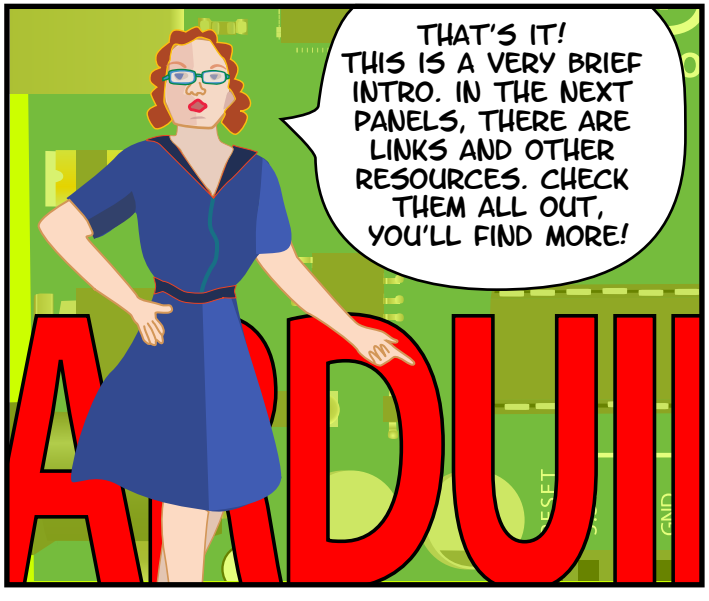
WE'LL USE ***PULSE WIDTH MODULATION*** (PWM). THIS IS A METHOD OF SIMULATING AN ANALOG VALUE BY MANIPULATING THE VOLTAGE, TURNING IT ON AND OFF AT DIFFERENT RATES, OR ***DUTY CYCLES***. YOU CAN USE PWM WITH PINS 3, 5, 6, 9, 10, AND 11.

```
int sensorValue = 0;

void setup() {
  pinMode(3,OUTPUT);
}

void loop() {
  sensorValue = analogRead(A0);
  analogWrite(3, sensorValue/4);
}
```

FIRST WE CREATE A ***VARIABLE*** TO STORE THE VALUE OF THE POT. IN SETUP WE MAKE PIN 3 AN OUTPUT. IN LOOP, WE STORE THE VALUE WE HAVE READ FROM PIN A0 IN OUR VARIABLE. THEN WE WRITE THE VALUE TO PIN 3, OUR LED PIN. WE HAVE TO DIVIDE THE VARIABLE BY 4, SO WE WILL HAVE A RANGE OF VALUES FROM 0 TO 255, OR A ***BYTE***.



THE BRIGHTNESS OF THE LED CHANGES, RANGING FROM COMPLETELY OFF TO VERY BRIGHT AS YOU TURN THE POT.



LINKS

SOFTWARE

SOFTWARE DOWNLOAD
HTTP://WWW.ARDUINO.CC/EN/MAIN/SOFTWARE
LANGUAGE REFERENCE
HTTP://ARDUINO.CC/EN/REFERENCE/HOMEPAGE

SUPPLIES

SPARKFUN ELECTRONICS
HTTP://WWW.SPARKFUN.COM/
ADAFRUIT INDUSTRIES
HTTP://ADAFRUIT.COM/
MAKER SHED
HTTP://WWW.MAKERSHED.COM/
JAMECO ELECTRONICS
HTTP://WWW.JAMECO.COM/

TUTORIALS

ARDUINO SITE TUTORIALS
HTTP://WWW.ARDUINO.CC/EN/TUTORIAL/HOMEPAGE
LADY ADA
HTTP://WWW.LADYADA.NET/LEARN/ARDUINO/
INSTRUCTABLES
HTTP://WWW.INSTRUCTABLES.COM/TAG/TYPE-ID/CATEGORY-TECHNOLOGY/CHANNEL-ARDUINO/

BOOKS

GETTING STARTED WITH ARDUINO BY MASSIMO BANZI
MAKING THINGS TALK: USING SENSORS, NETWORKS, AND ARDUINO TO SEE, HEAR, AND FEEL YOUR WORLD BY TOM IGOE
PHYSICAL COMPUTING: SENSING AND CONTROLLING THE PHYSICAL WORLD WITH COMPUTERS BY DAN O'SULLIVAN & TOM IGOE
ARDUINO COOKBOOK BY MICHAEL MARGOLIS

ALL TEXT AND DRAWINGS BY JODY CULKIN
FOR MORE, CHECK OUT JODYCULKIN.COM

SPECIAL THANKS TO TOM IGOE, MARIANNE PETIT, CALVIN REID, THE FACULTY AND STAFF OF THE INTERACTIVE TELECOMMUNICATIONS PROGRAM AT NYU, PARTICULARLY DAN O'SULLIVAN, DANNY ROZIN AND RED BURNS. THANKS TO CINDY KARASEK, CHRIS STEIN, SARAH TEITLER, KATHY GONCHAROV & ZANNAH MARSH.

MANY, MANY THANKS TO THE ARDUINO TEAM FOR BRINGING US THIS ROBUST AND FLEXIBLE OPEN SOURCE PLATFORM.

AND THANKS TO THE LIVELY, ACTIVE AND EVER GROWING ARDUINO COMMUNITY.

INTRODUCTION TO ARDUINO BY JODY CULKIN IS LICENSED UNDER A CREATIVE COMMONS ATTRIBUTION-NONCOMMERCIAL-SHAREALIKE 3.0 UNPORTED LICENSE.