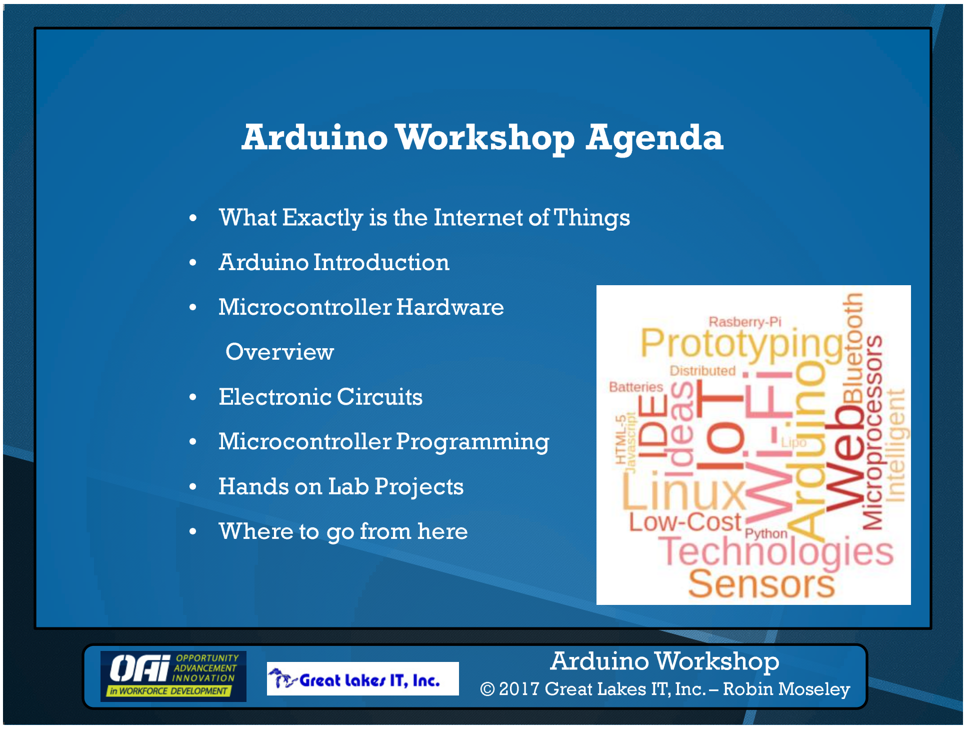
Arduino Workshop

Robin C. Moseley
President & CEO
Great Lakes IT, Inc

OAI OPPORTUNITY ADVANCEMENT INNOVATION in WORKFORCE DEVELOPMENT

OAI Chicago Southland (OCS)
214 Forest Blvd Park Forest, IL 60466
(708)-283-5020
Oaiinc.org

Great Lakes IT, Inc.
Technology Consulting
www.greatlakesitinc.com
1.847.867.2280



Arduino Workshop Agenda

- What Exactly is the Internet of Things
- Arduino Introduction
- Microcontroller Hardware Overview
- Electronic Circuits
- Microcontroller Programming
- Hands on Lab Projects
- Where to go from here

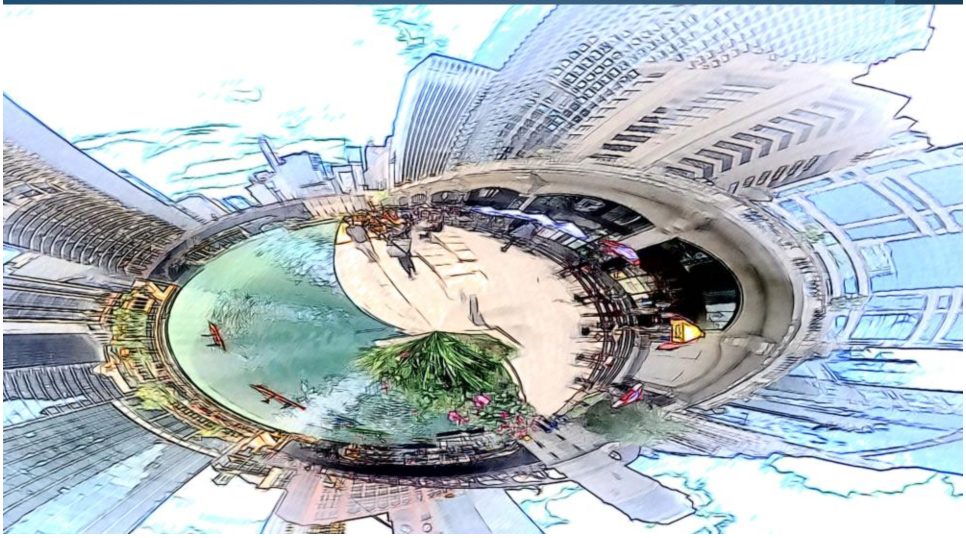
Rasberry-Pi
Prototyping
Distributed
Batteries
HTML-5
Javascript
IDE
Ideas
IoT
Wi-Fi
Lipo
Arduino
Web
Bluetooth
Microprocessors
Intelligent
Linux
Low-Cost
Python
Technologies
Sensors

Learning Objectives

- Understand how Arduino relates to IoT
- Understand what a microcontroller is and it’s general architecture
- Explain the concepts of microcontroller pins as inputs and outputs
- Know what an Arduino is and how it can be used for building and programming projects
- Understand basic Electronic components and Sensors
- Know how to setup and program the Arduino using a breadboard and components

The Internet of Things

We're building a world-sized robot, and we don't even realize it. - Bruce Schneier 2016

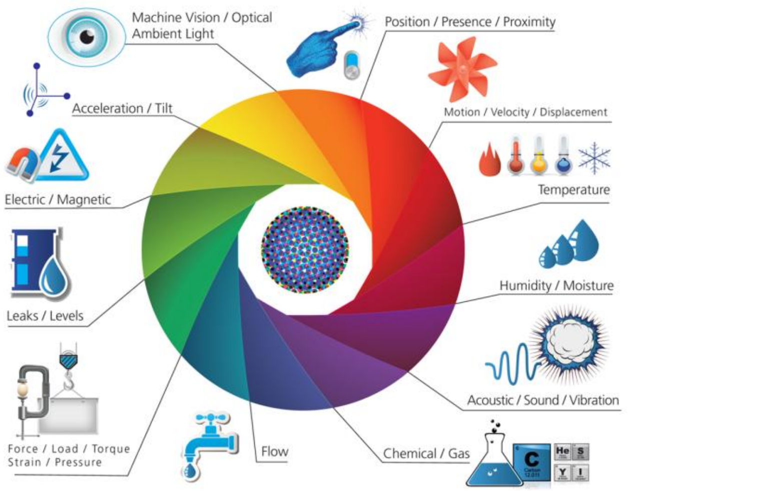


Internet of Things IoT

- Devices enabled with Machine-to-machine communications (Sensors and Actuators)
- The future will see an Internet of Things where billions of devices are connected to each other, all sharing data via the Internet
- As sensors become a low-cost technology, it is cheaper and cheaper to add them to any device
- IoT capabilities can be added to just about any physical object including clothing, jewelry, thermostats, medical devices, household appliances, home automation, industrial controls, wearable computing, even light bulbs.

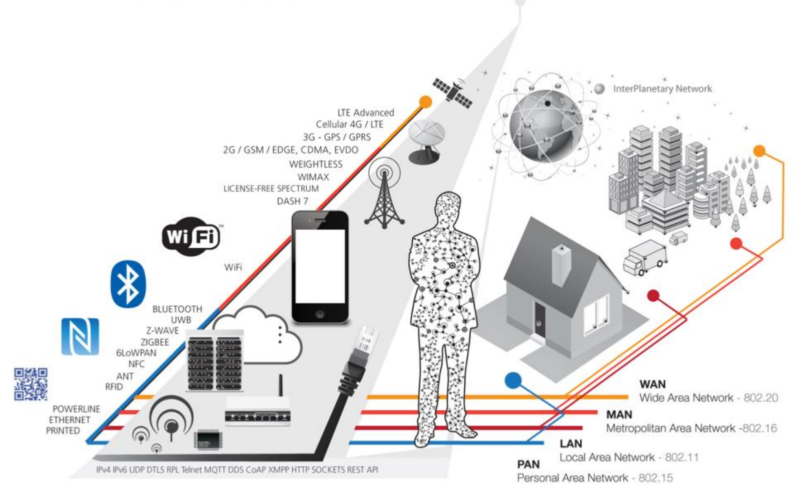
Sensors & Actuators

We are giving our world a digital nervous system. Location data using GPS sensors. Eyes and ears using cameras and microphones, along with sensory organs that can measure everything from temperature to pressure changes.



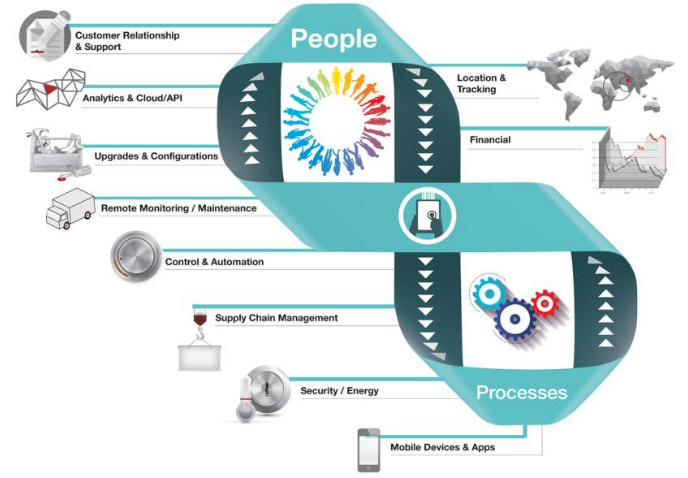
Connectivity

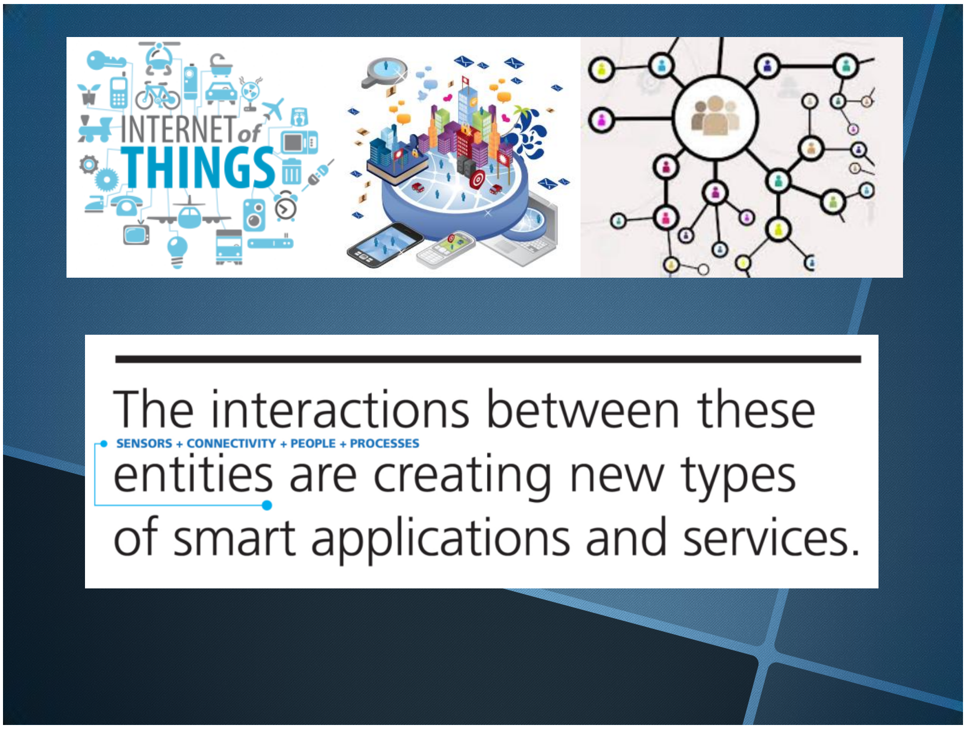
These inputs are digitized and placed onto networks.


InterPlanetary Network
LTE Advanced
Cellular 4G / LTE
3G - GPS / GPRS
2G / GSM / EDGE, CDMA, EVDO
WEIGHTLESS
WIMAX
LICENSE-FREE SPECTRUM
DASH 7
Wi Fi
WiFi
BLUETOOTH
UWB
Z-WAVE
ZIGBEE
6LoWPAN
NFC
ANT
RFID
POWERLINE
ETHERNET
PRINTED
IPv4 IPv6 UDP DTLS RPL Telnet MQTT DDS CoAP XMPP HTTP SOCKETS REST API
WAN
Wide Area Network - 802.20
MAN
Metropolitan Area Network -802.16
LAN
Local Area Network - 802.11
PAN
Personal Area Network - 802.15

People & Processes

These networked inputs can then be combined into bi-directional systems that integrate data, people, processes and systems for better decision making.




INTERNET of THINGS

SENSORS + CONNECTIVITY + PEOPLE + PROCESSES

The interactions between these entities are creating new types of smart applications and services.

By 2020 the Business Opportunity is bigger than $1 Trillion

The **Internet** gave us the opportunity to connect in ways we could never have dreamed possible.
The **Internet of Things** will take us beyond connection to become part of a living, moving, **global nervous system**.

What is a Microcontroller

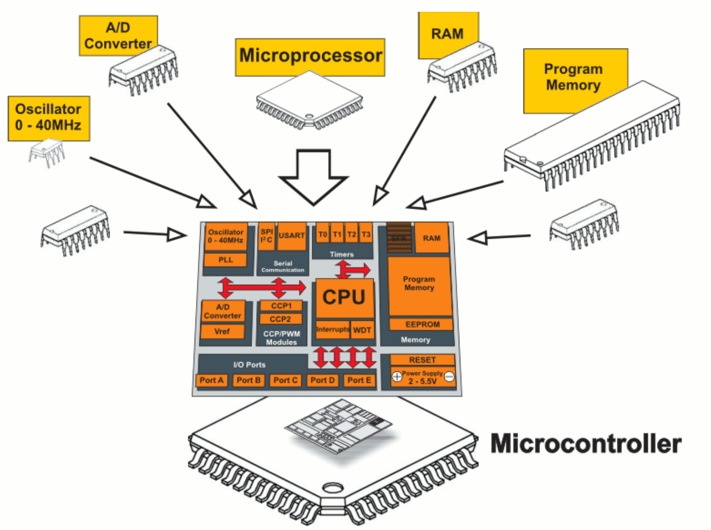
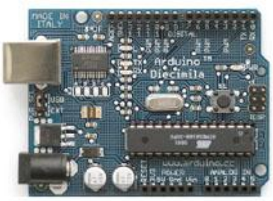


Fig. 0-1 Microcontroller versus Microprocessor

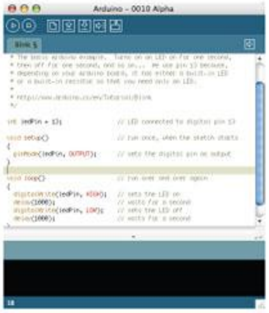
What is the Arduino

The word "Arduino" can mean 3 things

A physical piece of hardware



A programming environment



```
 * The basic Arduino example.  Turns on an LED on for one second,
 * then off for one second, and so on...  We use pin 13 because,
 * depending on your Arduino board, it has either a built-in LED
 * or a built-in resistor so that you need only an LED.
 *
 * http://www.arduino.cc/en/Tutorial/Blink
 */

int ledPin = 13;                // LED connected to digital pin 13

void setup()                    // run once, when the sketch starts
{
  pinMode(ledPin, OUTPUT);      // sets the digital pin as output
}

void loop()                     // run over and over again
{
  digitalWrite(ledPin, HIGH);   // sets the LED on
  delay(1000);                  // waits for a second
  digitalWrite(ledPin, LOW);    // sets the LED off
  delay(1000);                  // waits for a second
```

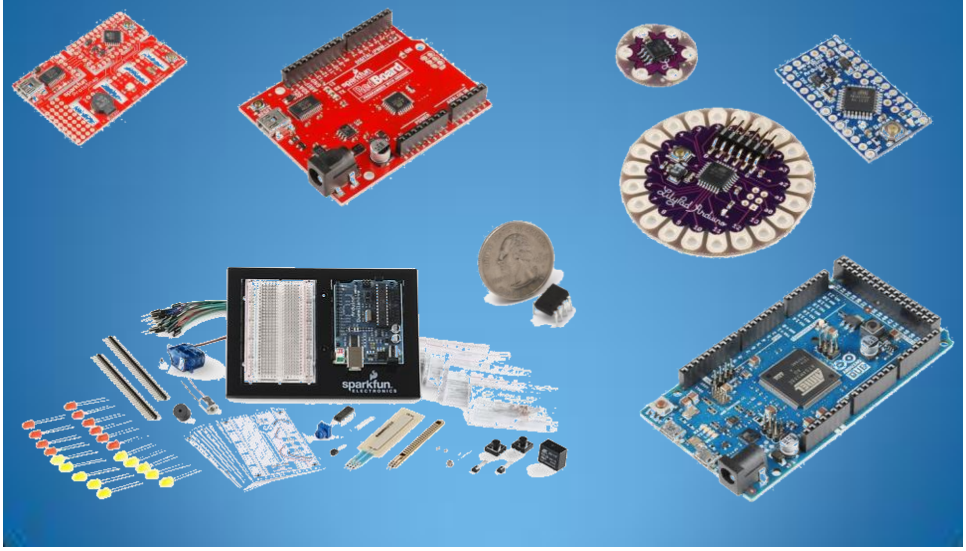
A community & philosophy

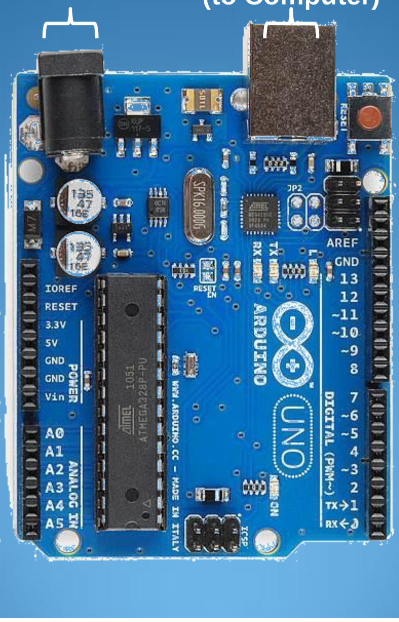
Arduino playground

:: About the Arduino Playground ::

:: RoadMap: What Needs to be Done? ::

Arduino & Arduino Compatible Boards





PWR IN

USB
(to Computer)

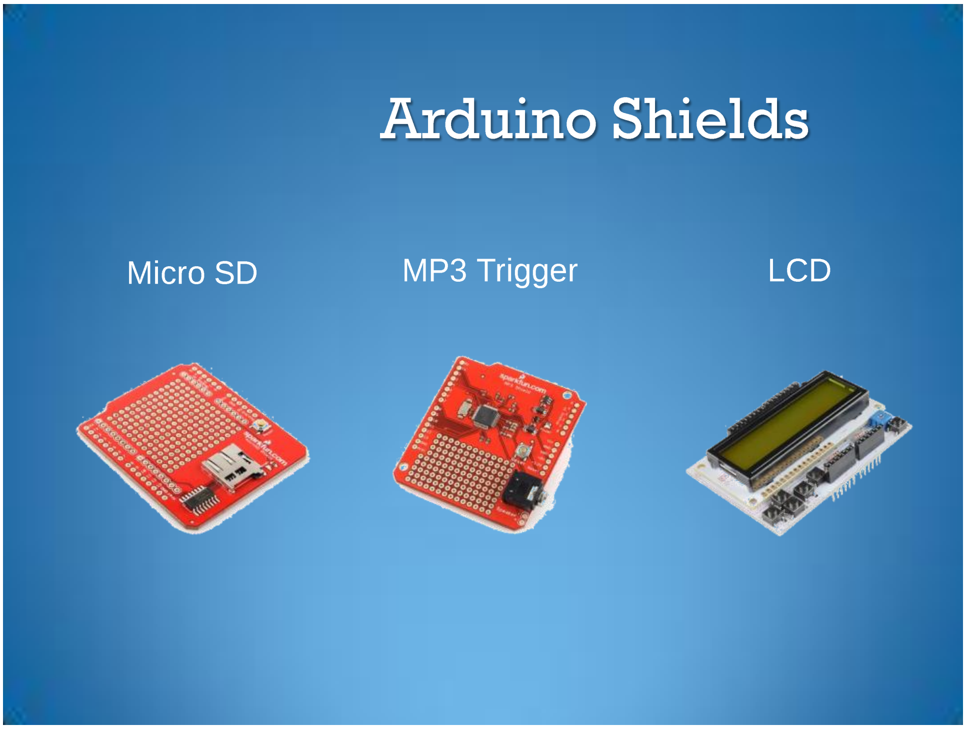
RESET

SCL\SDA
(I2C Bus)

POWER
5V / 3.3V / GND

Digital I\O
PWM(3, 5, 6, 9, 10, 11)

Analog
INPUTS

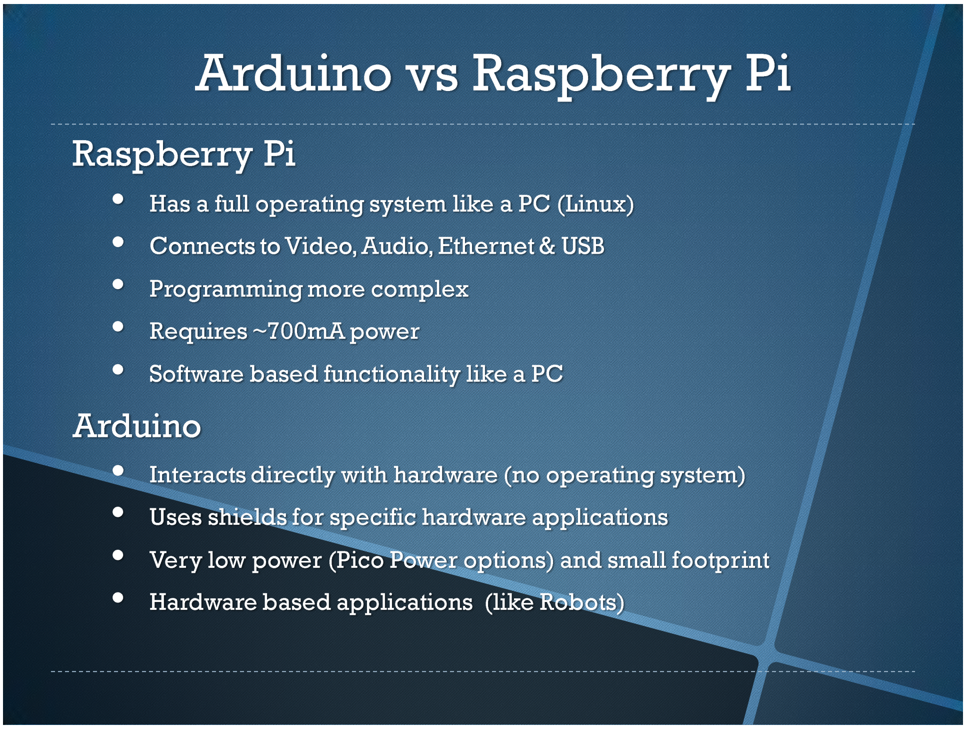


Arduino Shields

Micro SD

MP3 Trigger

LCD



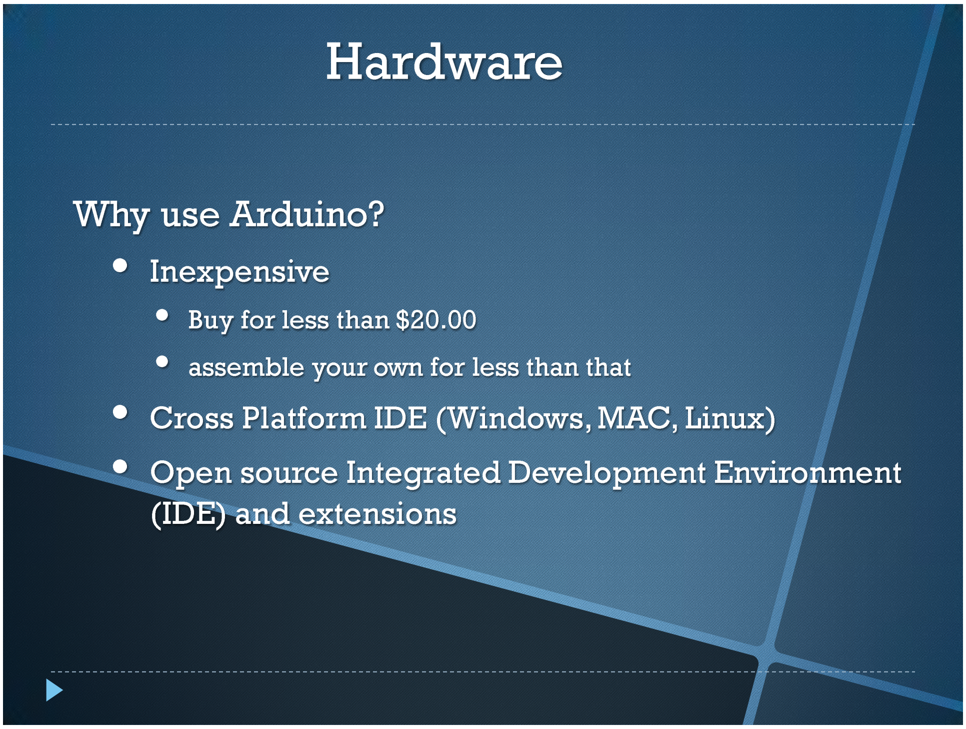
Arduino vs Raspberry Pi

Raspberry Pi

- Has a full operating system like a PC (Linux)
- Connects to Video, Audio, Ethernet & USB
- Programming more complex
- Requires ~700mA power
- Software based functionality like a PC

Arduino

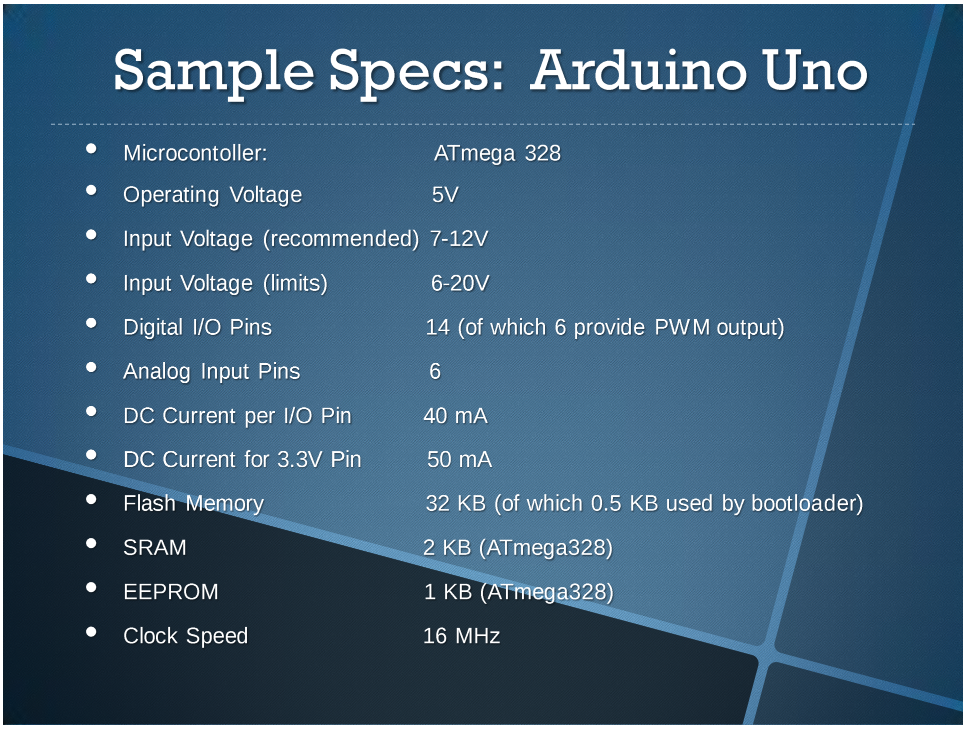
- Interacts directly with hardware (no operating system)
- Uses shields for specific hardware applications
- Very low power (Pico Power options) and small footprint
- Hardware based applications (like Robots)



Hardware

Why use Arduino?

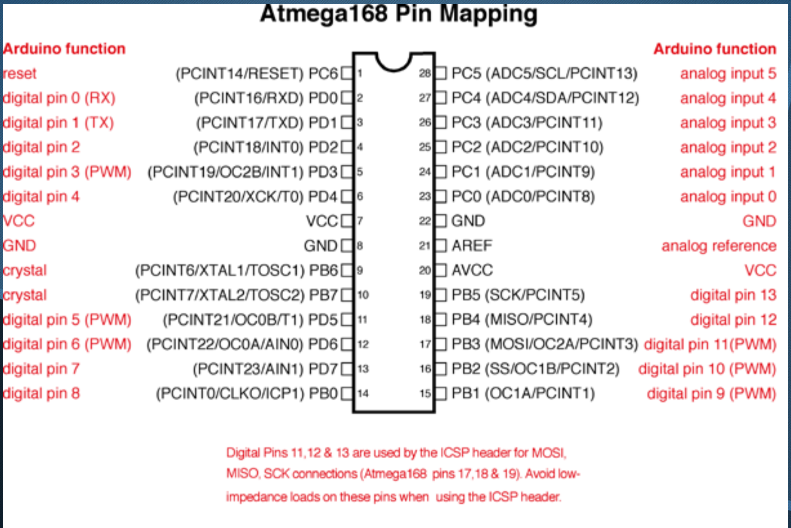
- Inexpensive
  - Buy for less than $20.00
  - assemble your own for less than that
- Cross Platform IDE (Windows, MAC, Linux)
- Open source Integrated Development Environment (IDE) and extensions



Sample Specs: Arduino Uno

Microcontoller:	ATmega 328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (of which 0.5 KB used by bootloader)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

The Arduino Microcontroller: Atmel Atmega 168/328

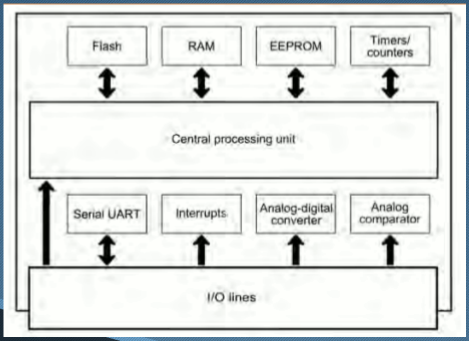


Atmega168 Pin Mapping

Arduino function					Arduino function
reset	(PCINT14/RESET) PC6	1	28	PC5 (ADC5/SCL/PCINT13)	analog input 5
digital pin 0 (RX)	(PCINT16/RXD) PD0	2	27	PC4 (ADC4/SDA/PCINT12)	analog input 4
digital pin 1 (TX)	(PCINT17/TXD) PD1	3	26	PC3 (ADC3/PCINT11)	analog input 3
digital pin 2	(PCINT18/INT0) PD2	4	25	PC2 (ADC2/PCINT10)	analog input 2
digital pin 3 (PWM)	(PCINT19/OC2B/INT1) PD3	5	24	PC1 (ADC1/PCINT9)	analog input 1
digital pin 4	(PCINT20/XCK/T0) PD4	6	23	PC0 (ADC0/PCINT8)	analog input 0
VCC	VCC	7	22	GND	GND
GND	GND	8	21	AREF	analog reference
crystal	(PCINT6/XTAL1/TOSC1) PB6	9	20	AVCC	VCC
crystal	(PCINT7/XTAL2/TOSC2) PB7	10	19	PB5 (SCK/PCINT5)	digital pin 13
digital pin 5 (PWM)	(PCINT21/OC0B/T1) PD5	11	18	PB4 (MISO/PCINT4)	digital pin 12
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0) PD6	12	17	PB3 (MOSI/OC2A/PCINT3)	digital pin 11(PWM)
digital pin 7	(PCINT23/AIN1) PD7	13	16	PB2 (SS/OC1B/PCINT2)	digital pin 10 (PWM)
digital pin 8	(PCINT0/CLKO/ICP1) PB0	14	15	PB1 (OC1A/PCINT1)	digital pin 9 (PWM)

Digital Pins 11,12 & 13 are used by the ICSP header for MOSI, MISO, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

The Arduino Microcontroller: Architecture



Concepts: INPUT vs. OUTPUT

Referenced from the perspective of the microcontroller

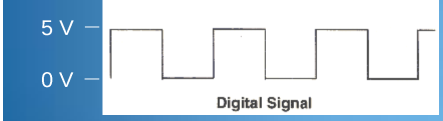
Inputs is a signal / information going into the board.

Output is any signal exiting the board.

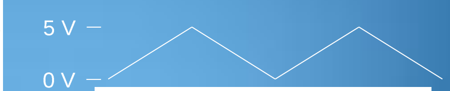
Examples: Buttons Switches, Light Sensors, Flex Sensors, Humidity Sensors, Temperature Sensors…	Examples: LEDs, DC motor, servo motor, a piezo buzzer, relay, an RGB LED

Concepts: Analog vs. Digital

- Microcontrollers are **digital** devices – ON or OFF.
- **analog** signals are anything that can be a full range of values. What are some examples?



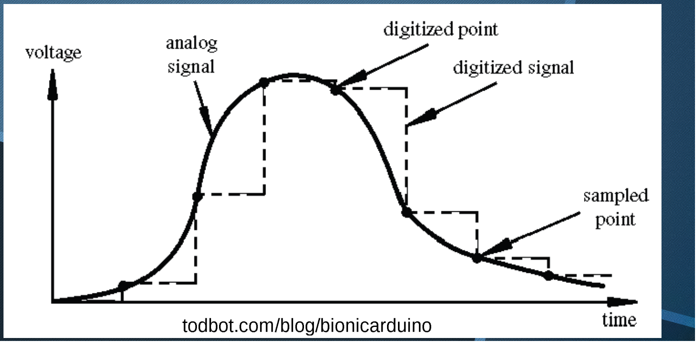
Digital Signal



Analog Signal

Digital? Analog?

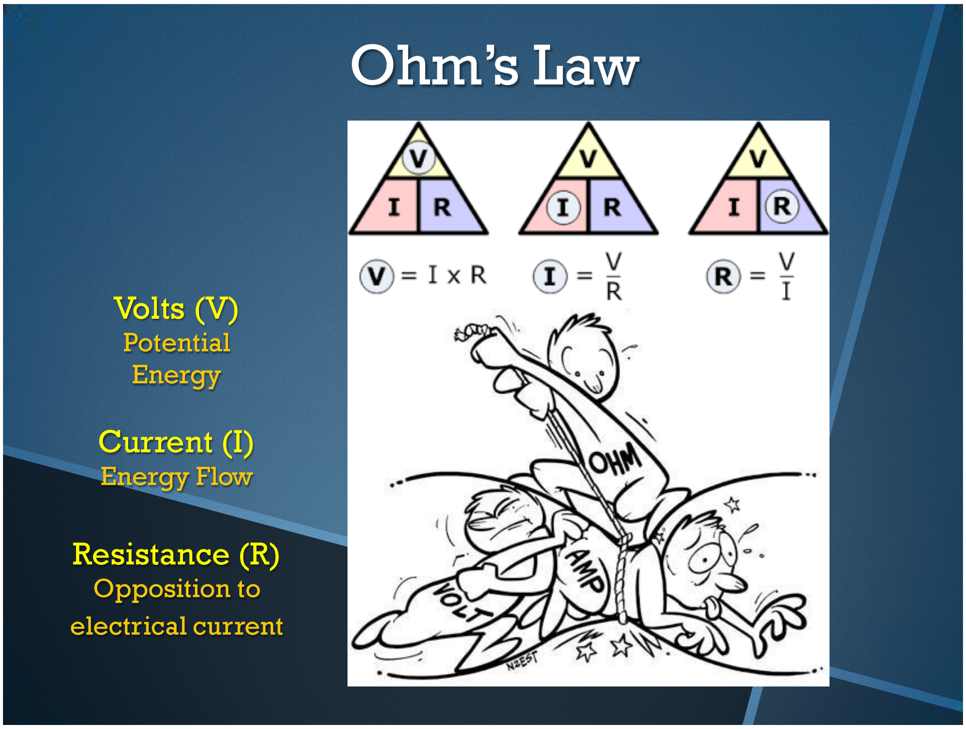
- Digital has two values: **on** and **off**
- Analog has many (infinite) values
- Computers don’t really do analog, they ***quantize***
- Remember the 6 analog input pins---here’s how they work



todbot.com/blog/bionicarduino

Electricity \ Electronics Basic Concepts

- Ohms Law
- Voltage
- Current
- Resistance
- Power (Watts)
- Using a Multi-meter



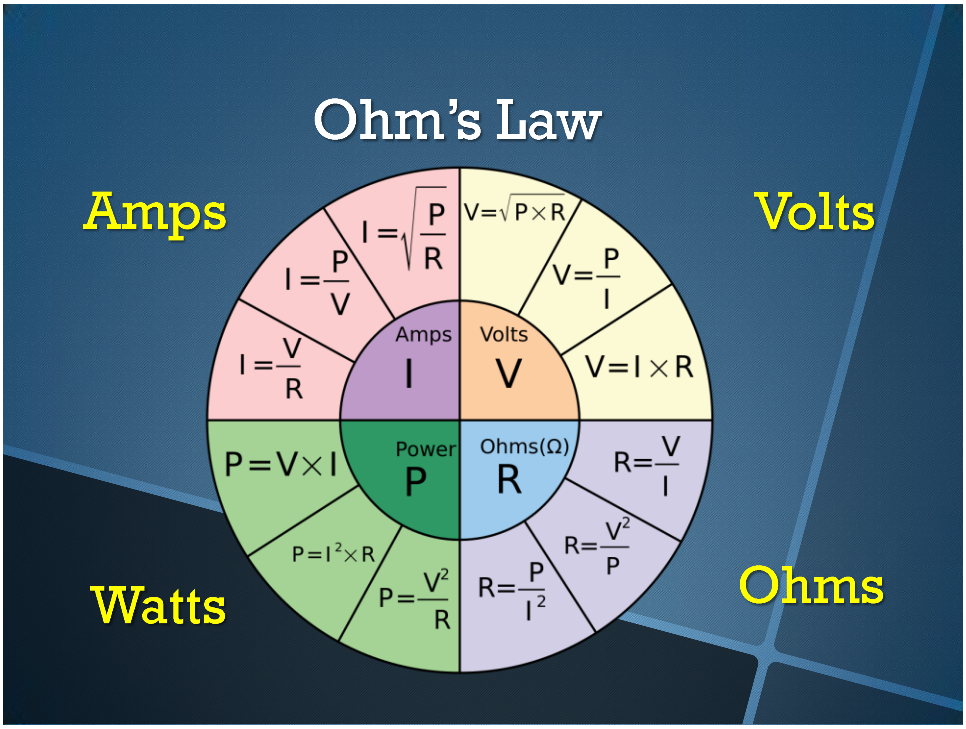
Ohm's Law

Volts (V)
Potential Energy

Current (I)
Energy Flow

Resistance (R)
Opposition to electrical current

$V = I \times R$ $I = \frac{V}{R}$ $R = \frac{V}{I}$



Ohm's Law

Amps

Volts

Watts

Ohms

Amps
I

Volts
V

Power
P

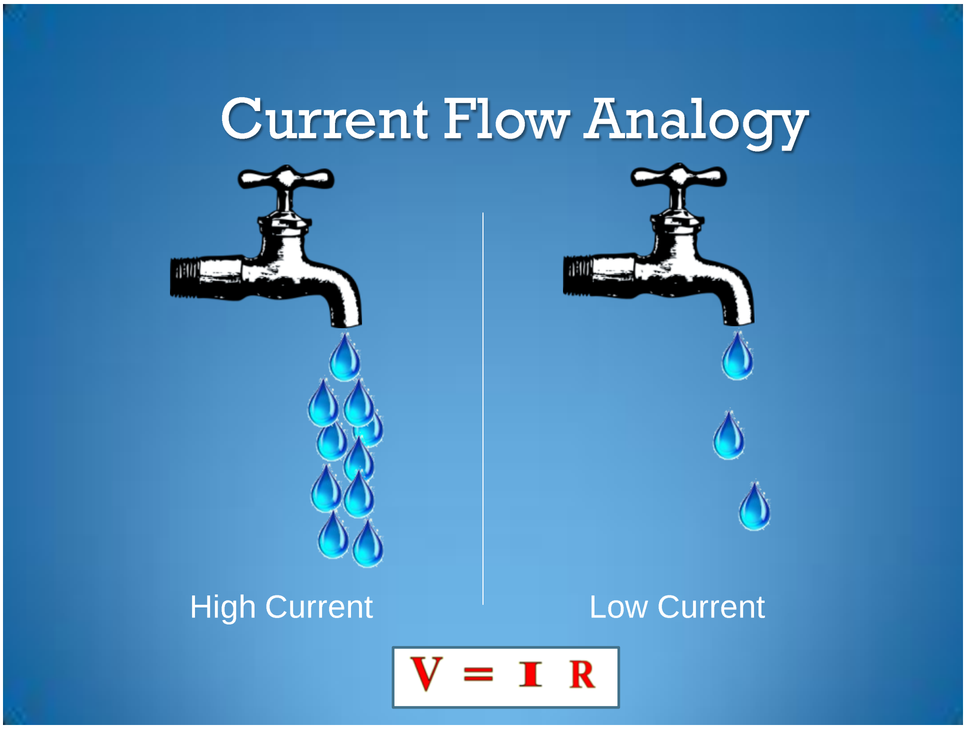
Ohms(Ω)
R

$I = \sqrt{\frac{P}{R}}$ $I = \frac{P}{V}$ $I = \frac{V}{R}$

$V = \sqrt{P \times R}$ $V = \frac{P}{I}$ $V = I \times R$

$R = \frac{V}{I}$ $R = \frac{V^2}{P}$ $R = \frac{P}{I^2}$

$P = V \times I$ $P = I^2 \times R$ $P = \frac{V^2}{R}$

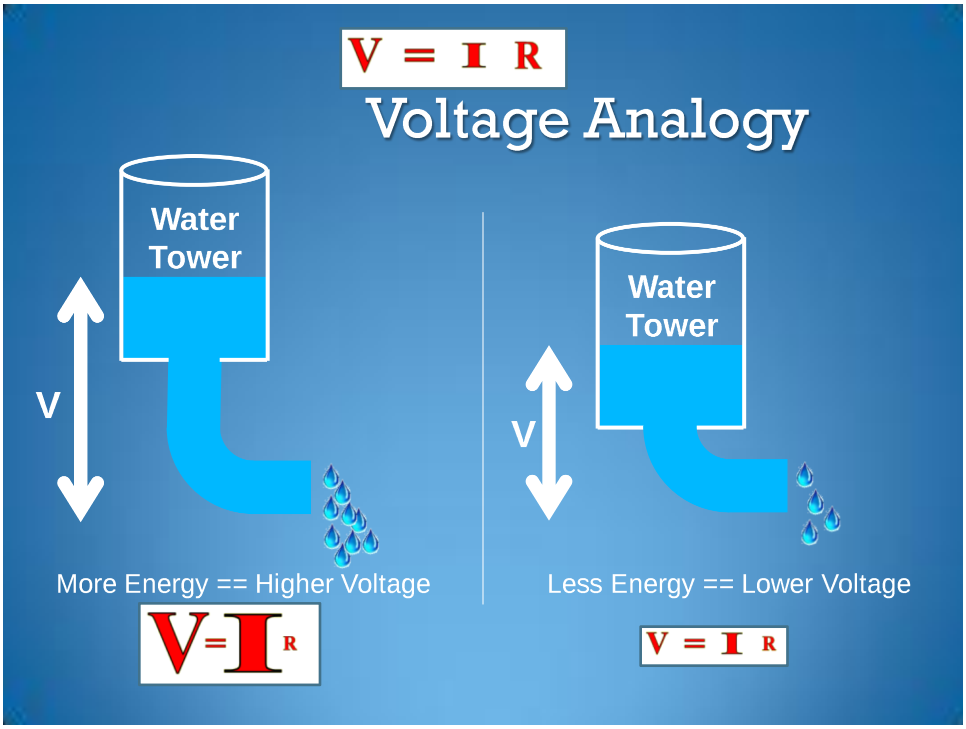


Current Flow Analogy

High Current

Low Current

V = I R



V = I R

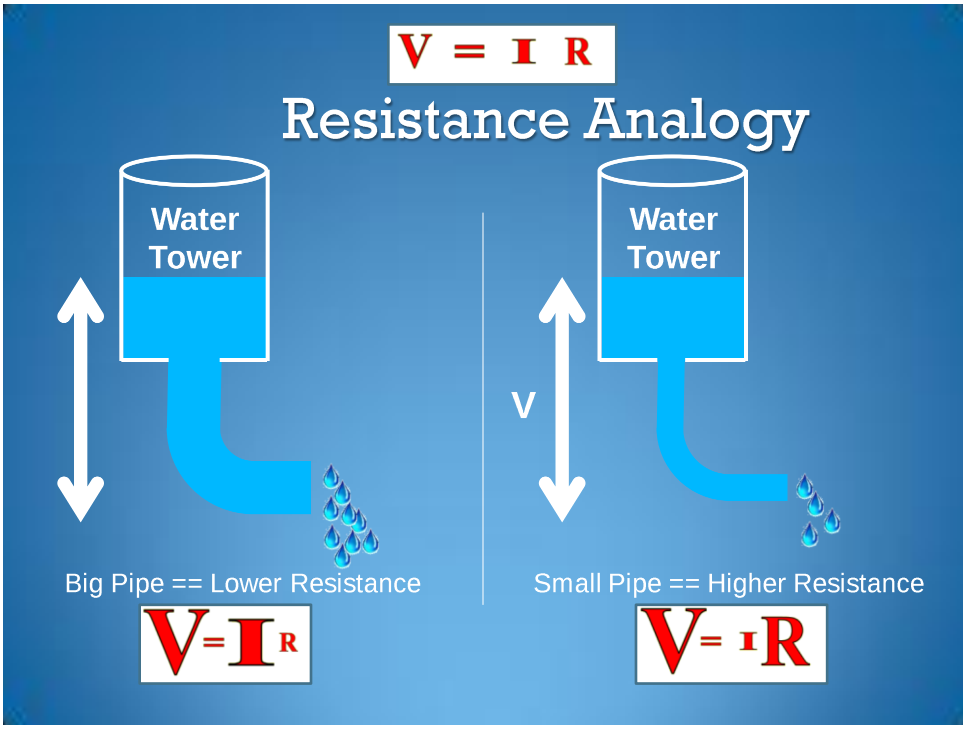
Voltage Analogy

More Energy == Higher Voltage

V=IR

Less Energy == Lower Voltage

V = I R

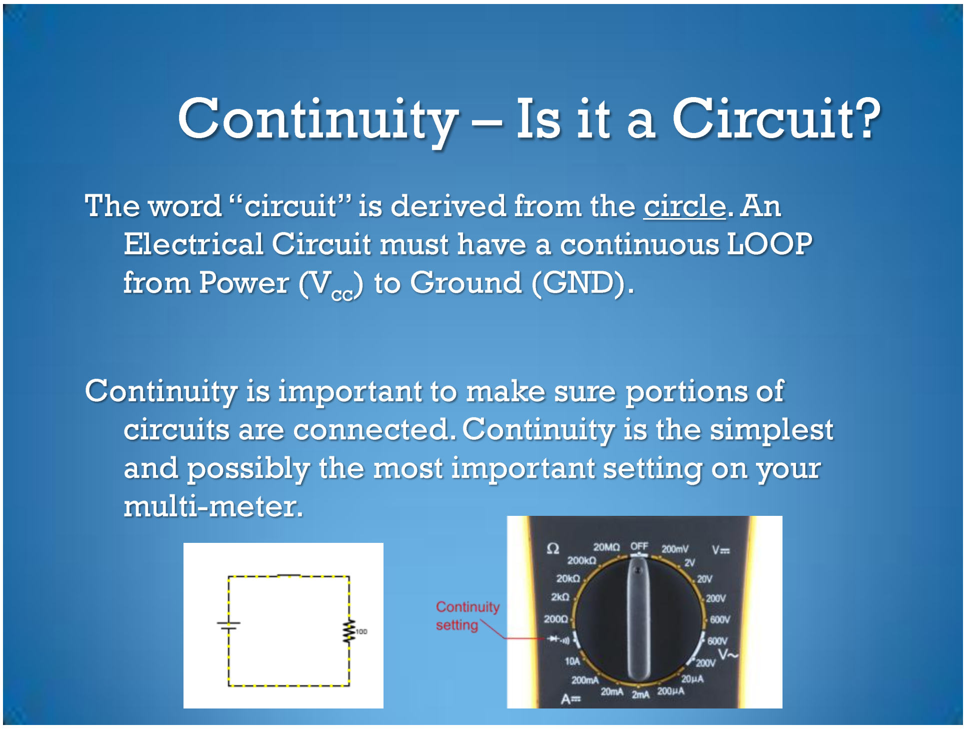


V = I R

Resistance Analogy

Big Pipe == Lower Resistance

Small Pipe == Higher Resistance



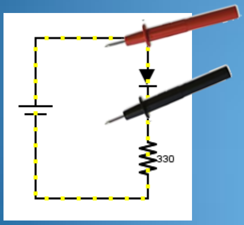
Continuity – Is it a Circuit?

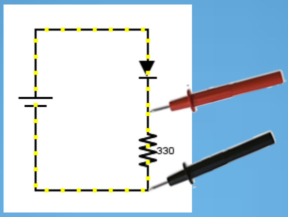
The word "circuit" is derived from the circle. An Electrical Circuit must have a continuous LOOP from Power (V_{CC}) to Ground (GND).

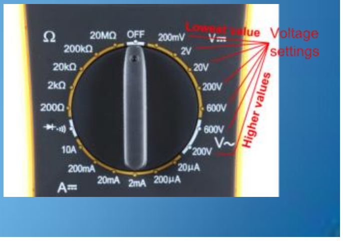
Continuity is important to make sure portions of circuits are connected. Continuity is the simplest and possibly the most important setting on your multi-meter.

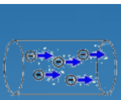
Measuring Electricity – Voltage

Voltage is a measure of potential electrical energy. A voltage is also called a potential difference – it is measured between two points in a circuit – across a device.





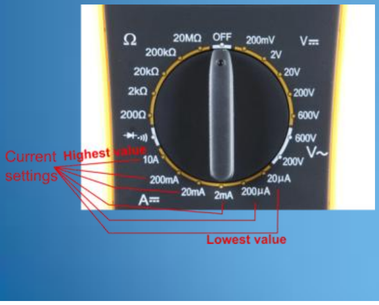


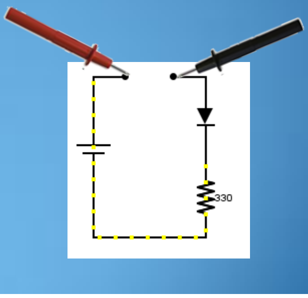


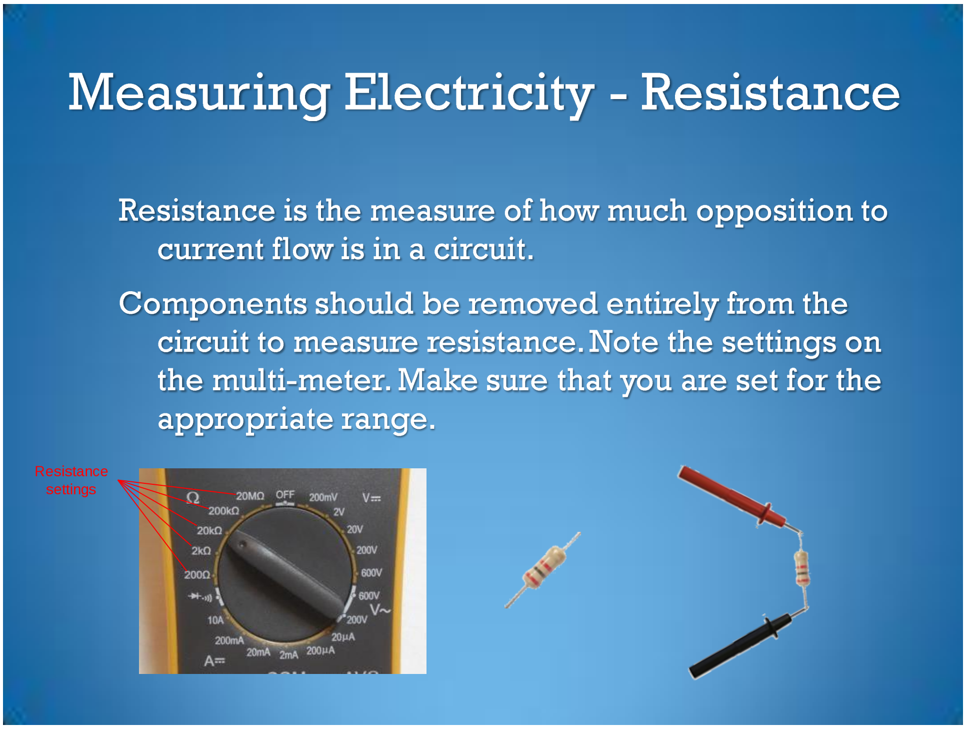
Measuring Electricity -- Current

Current is the measure of the rate of charge flow. For Electrical Engineers – we consider this to be the movement of electrons.

In order to measure this – you must break the circuit or insert the meter in-line (series).





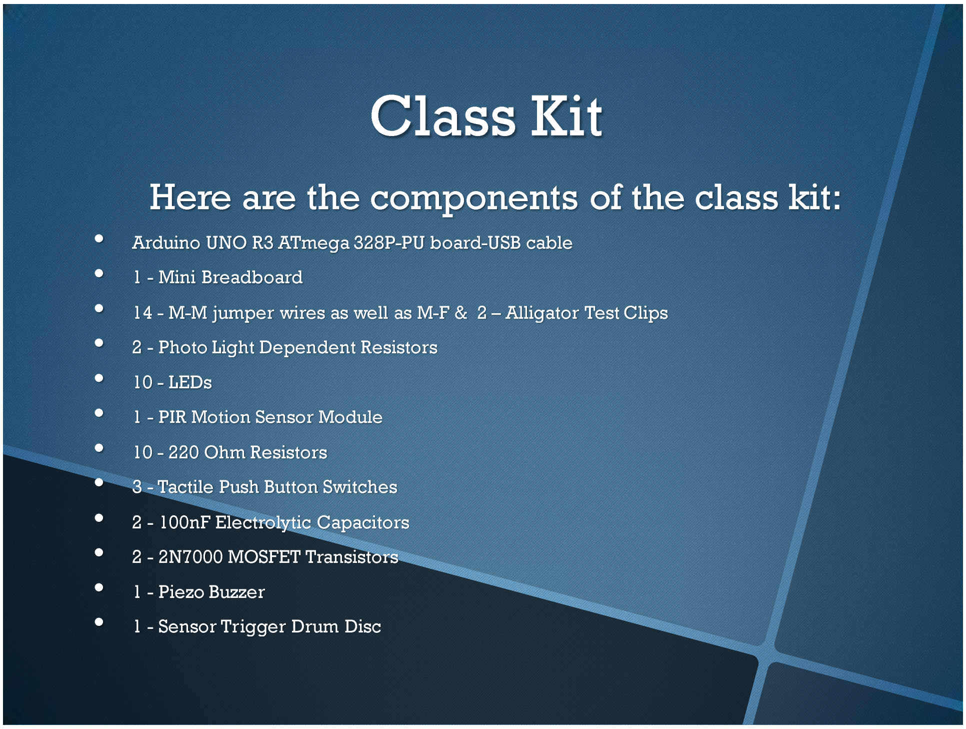


Measuring Electricity - Resistance

Resistance is the measure of how much opposition to current flow is in a circuit.

Components should be removed entirely from the circuit to measure resistance. Note the settings on the multi-meter. Make sure that you are set for the appropriate range.

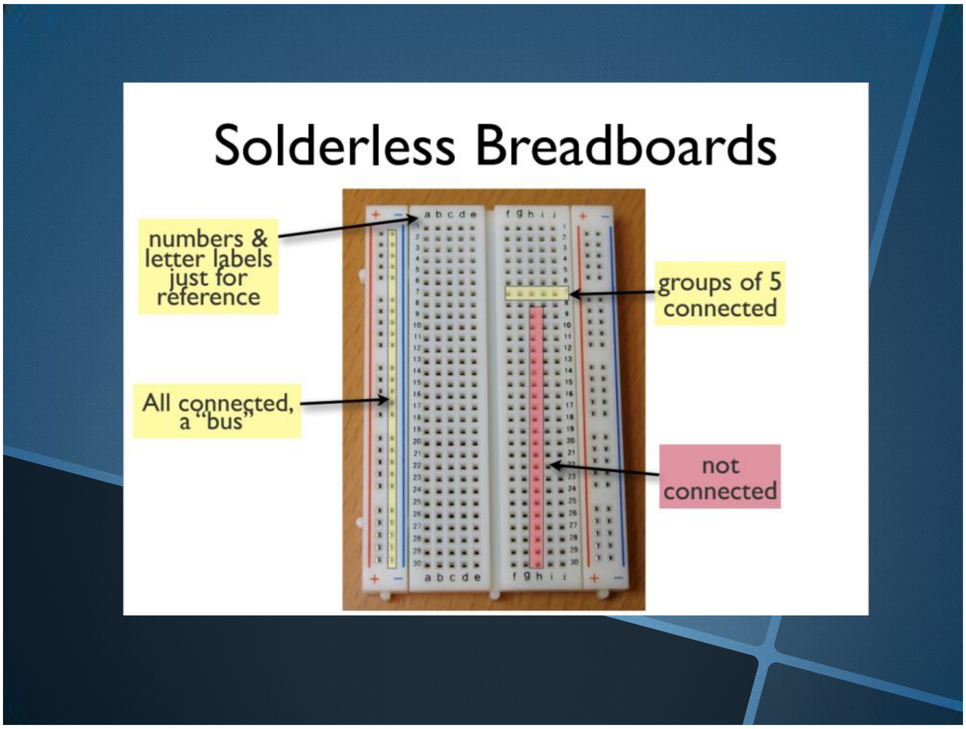
Resistance settings



Class Kit

Here are the components of the class kit:

- Arduino UNO R3 ATmega 328P-PU board-USB cable
- 1 - Mini Breadboard
- 14 - M-M jumper wires as well as M-F & 2 – Alligator Test Clips
- 2 - Photo Light Dependent Resistors
- 10 - LEDs
- 1 - PIR Motion Sensor Module
- 10 - 220 Ohm Resistors
- 3 - Tactile Push Button Switches
- 2 - 100nF Electrolytic Capacitors
- 2 - 2N7000 MOSFET Transistors
- 1 - Piezo Buzzer
- 1 - Sensor Trigger Drum Disc



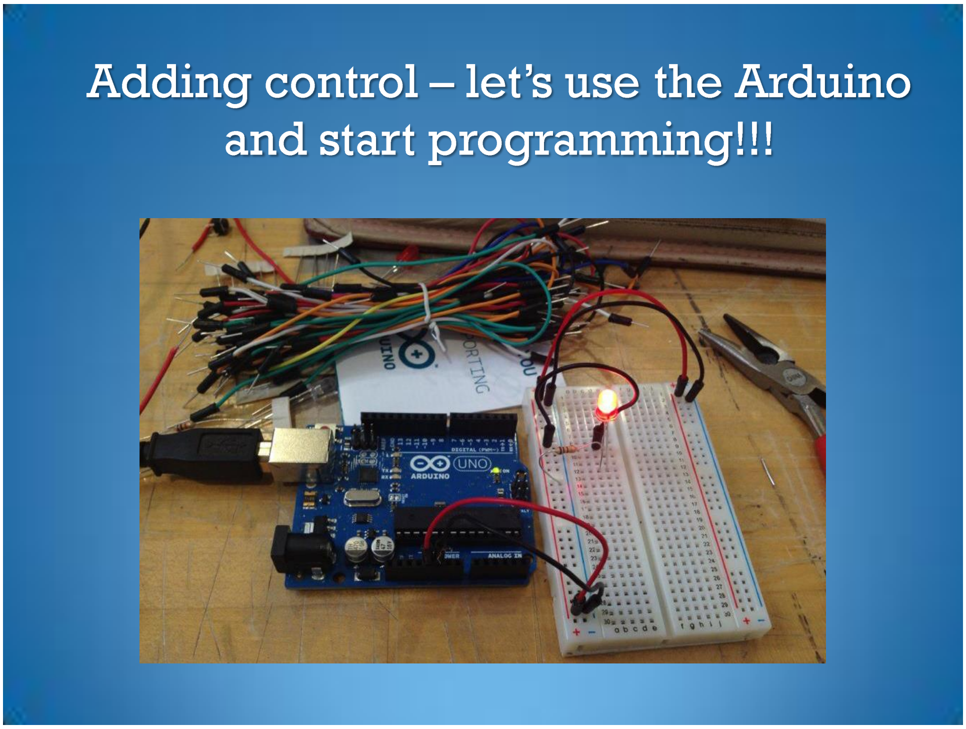
Solderless Breadboards

numbers & letter labels just for reference

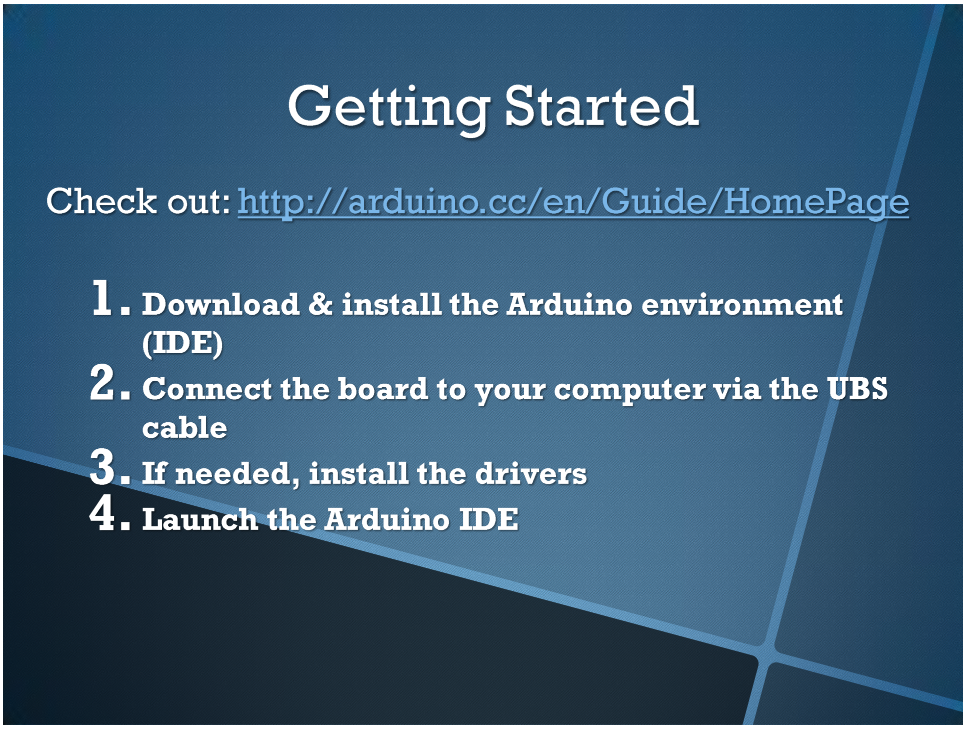
All connected, a "bus"

groups of 5 connected

not connected



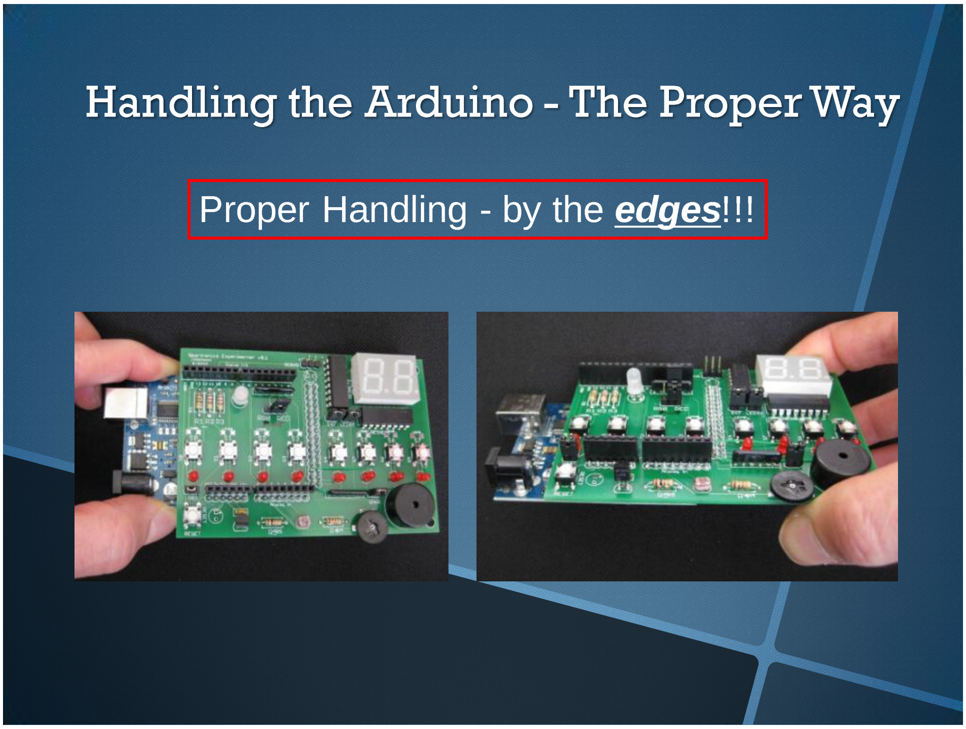
Adding control – let's use the Arduino and start programming!!!



Getting Started

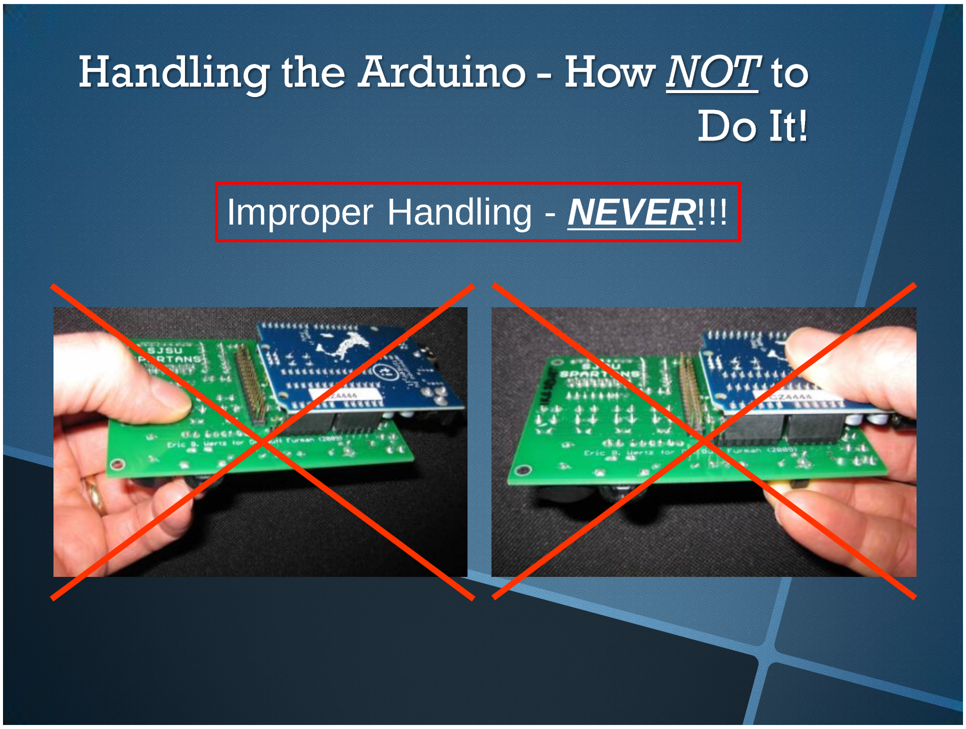
Check out: http://arduino.cc/en/Guide/HomePage

1. **Download & install the Arduino environment (IDE)**
2. **Connect the board to your computer via the UBS cable**
3. **If needed, install the drivers**
4. **Launch the Arduino IDE**



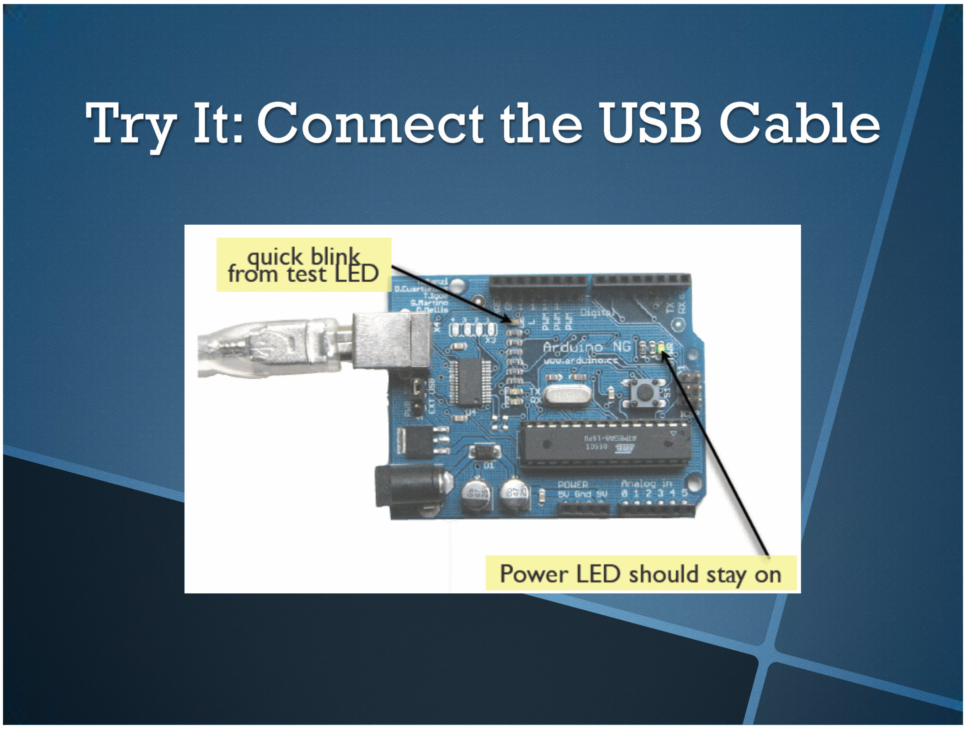
Handling the Arduino - The Proper Way

Proper Handling - by the ***edges***!!!



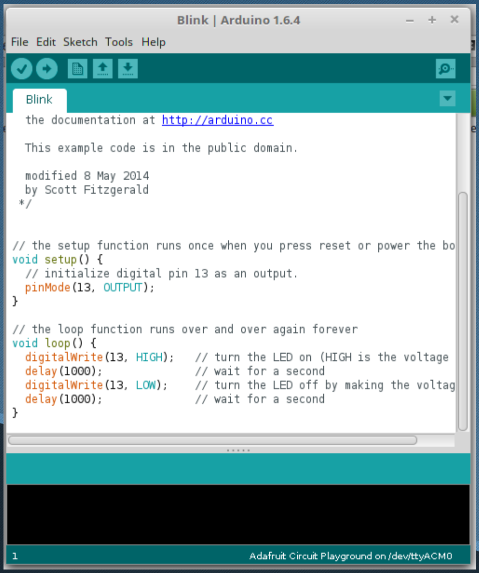
Handling the Arduino - How *NOT* to Do It!

Improper Handling - ***NEVER***!!!



Try It: Connect the USB Cable

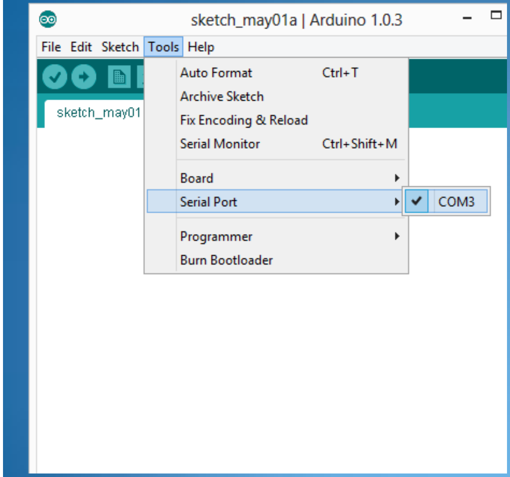
Arduino IDE


Blink | Arduino 1.6.4
File Edit Sketch Tools Help
Blink
Adafruit Circuit Playground on /dev/ttyACM0

```
  the documentation at http://arduino.cc

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald
 */


// the setup function runs once when you press reset or power the bo
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltag
  delay(1000);              // wait for a second
}
```

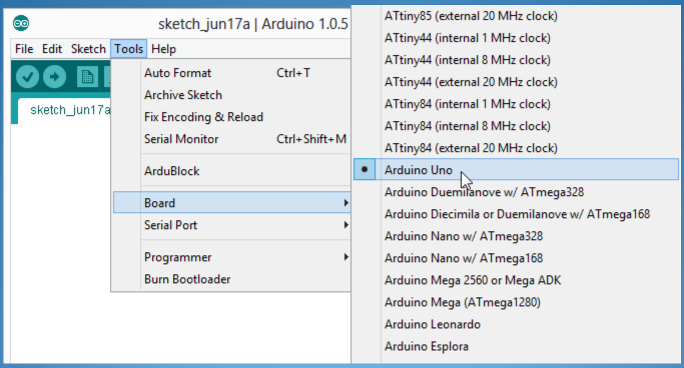
See: http://arduino.cc/en/Guide/Environment for more information

Settings: Tools → Serial Port



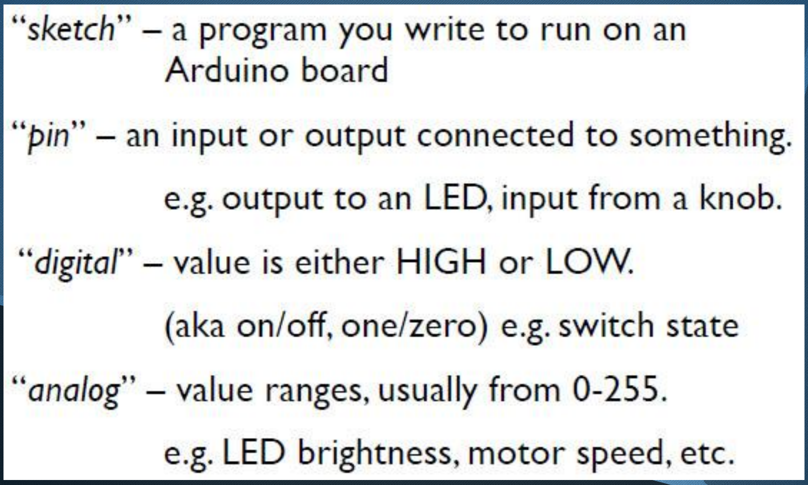
- Your computer communicates to the Arduino microcontroller via a serial port → through a USB-Serial adapter.
- Check to make sure that the drivers are properly installed.

Settings: Tools → Board


sketch_jun17a | Arduino 1.0.5
File
Edit
Sketch
Tools
Help
sketch_jun17a
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
ArduBlock
Board
Serial Port
Programmer
Burn Bootloader
ATtiny85 (external 20 MHz clock)
ATtiny44 (internal 1 MHz clock)
ATtiny44 (internal 8 MHz clock)
ATtiny44 (external 20 MHz clock)
ATtiny84 (internal 1 MHz clock)
ATtiny84 (internal 8 MHz clock)
ATtiny84 (external 20 MHz clock)
● Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
Arduino Leonardo
Arduino Esplora

Next, double-check that the proper board is selected under the Tools→Board menu.

Terminology



"*sketch*" – a program you write to run on an Arduino board

"*pin*" – an input or output connected to something.

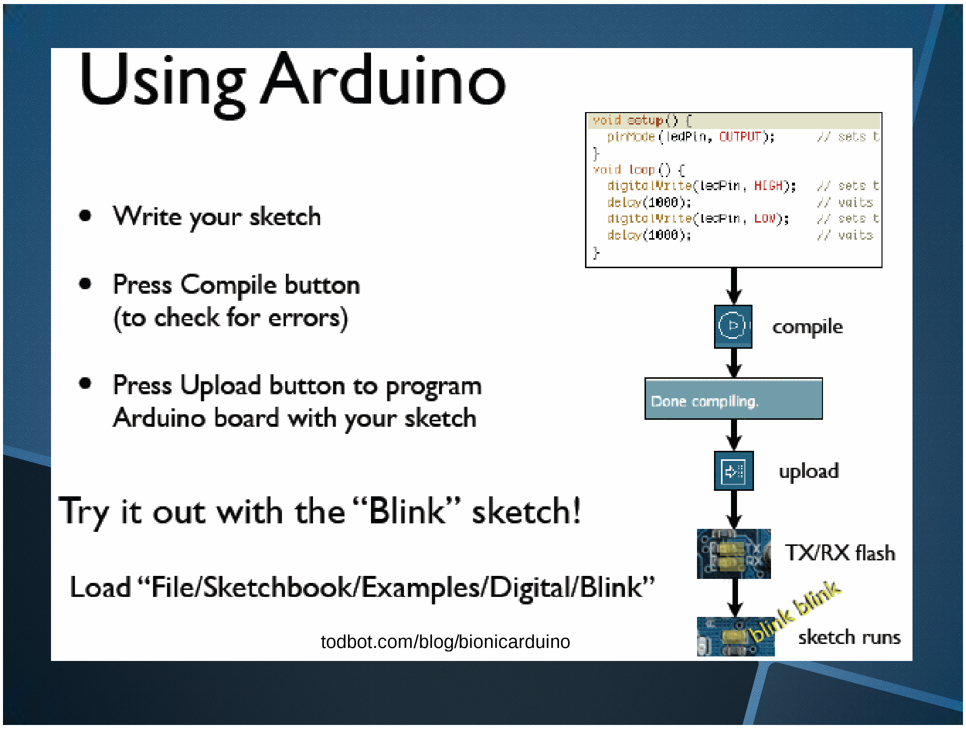
e.g. output to an LED, input from a knob.

"*digital*" – value is either HIGH or LOW.

(aka on/off, one/zero) e.g. switch state

"*analog*" – value ranges, usually from 0-255.

e.g. LED brightness, motor speed, etc.



Using Arduino

- Write your sketch
- Press Compile button (to check for errors)
- Press Upload button to program Arduino board with your sketch

```
void setup() {
  pinMode(ledPin, OUTPUT);        // sets t
}
void loop() {
  digitalWrite(ledPin, HIGH);     // sets t
  delay(1000);                    // waits
  digitalWrite(ledPin, LOW);      // sets t
  delay(1000);                    // waits
}
```

Try it out with the "Blink" sketch!

Load "File/Sketchbook/Examples/Digital/Blink"

todbot.com/blog/bionicarduino

Structure of an Arduino Program

An arduino program == '**sketch**'

Must have:

setup()

loop()

setup()

configures pin modes and registers

loop()

runs the main body of the program forever

like while(1) {…}

Where is main() ?

Arduino simplifies things

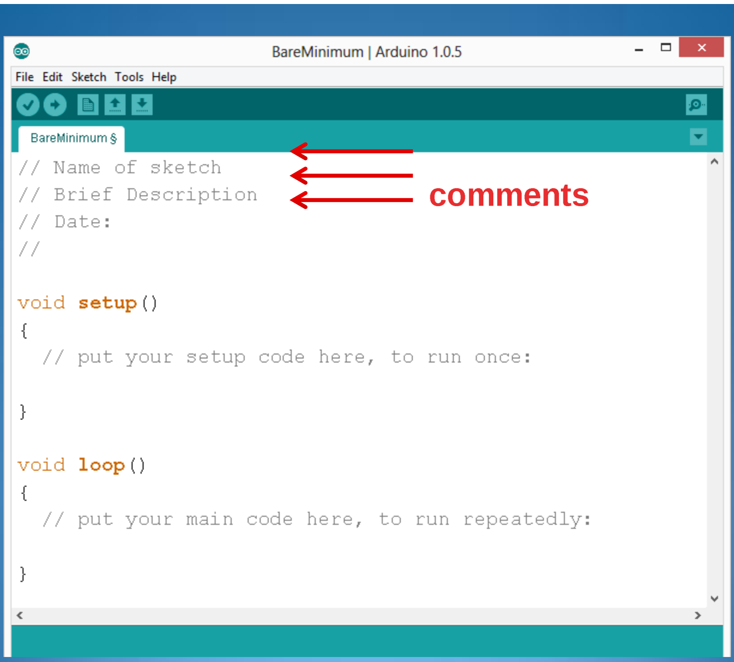
Does things for you

```
/* Blink  -  turns on an LED for DELAY_ON msec,
then off for DELAY_OFF msec, and repeats
BJ Furman rev. 1.1   Last rev: 22JAN2011
*/
#define LED_PIN   13    // LED on digital pin 13
#define DELAY_ON  1000
#define DELAY_OFF 1000

void setup()
{
  // initialize the digital pin as an output:
  pinMode(LED_PIN, OUTPUT);
}

// loop() method runs forever,
// as long as the Arduino has power

void loop()
{
  digitalWrite(LED_PIN, HIGH);    // set the LED on
  delay(DELAY_ON);  // wait for DELAY_ON msec
  digitalWrite(LED_PIN, LOW);  // set the LED off
  delay(DELAY_OFF);  // wait for DELAY_OFF msec
}
```

Programming Basics

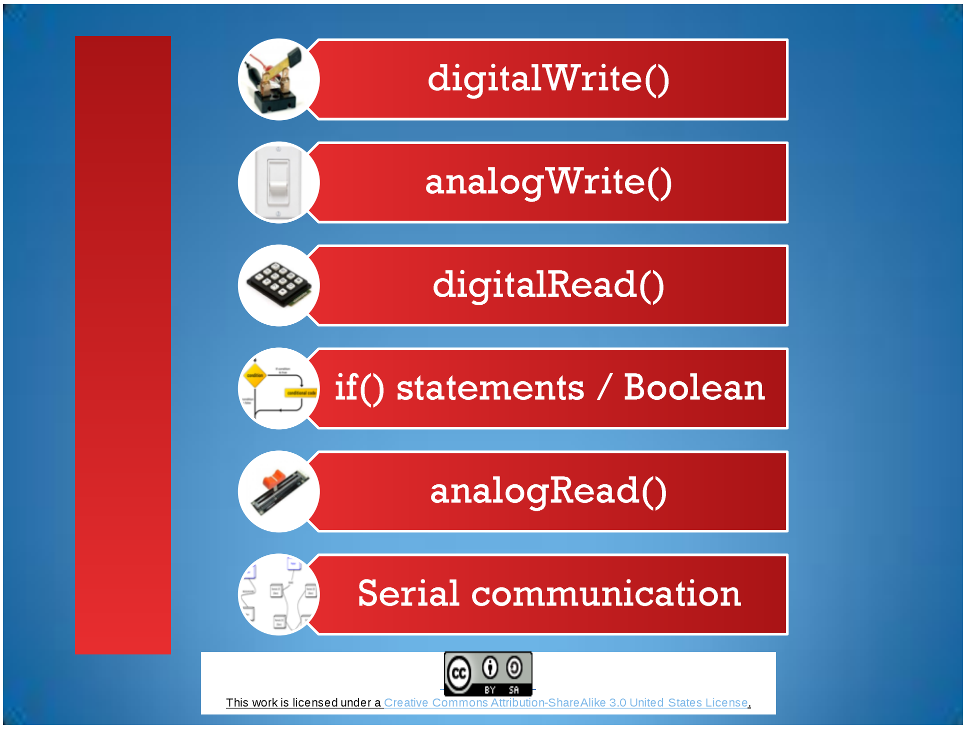
- Code is Case Sensitive
- Comments are for you – the programmer and your friends…or anyone else human that might read your code.

```
// this is for single line comments
// it’s good to put a description at the top and
before anything ‘tricky’

/* this is for multi-line comments
   Like this…
   And this….
*/
```

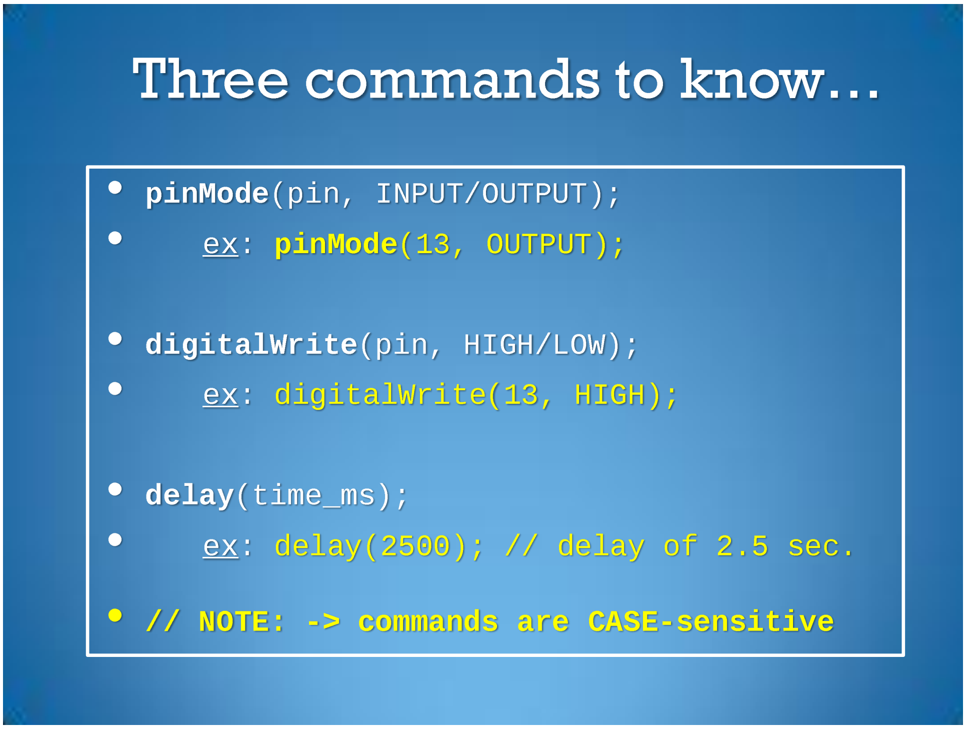


```
// Name of sketch
// Brief Description
// Date:
//

void setup()
{
  // put your setup code here, to run once:

}

void loop()
{
  // put your main code here, to run repeatedly:

}
```

comments

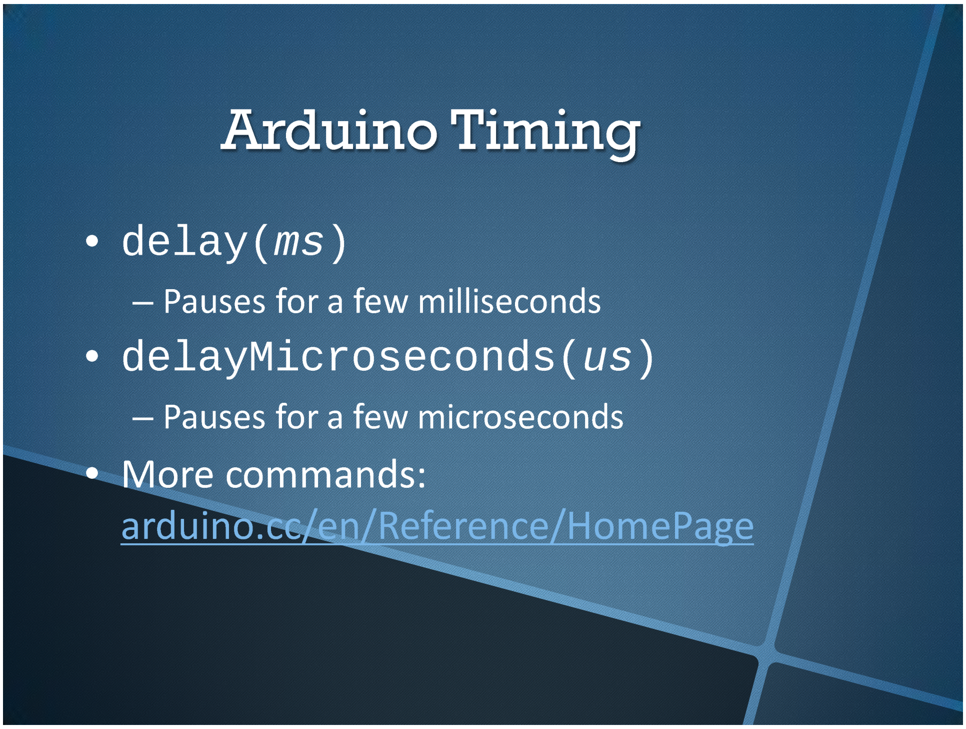


- digitalWrite()
- analogWrite()
- digitalRead()
- if() statements / Boolean
- analogRead()
- Serial communication



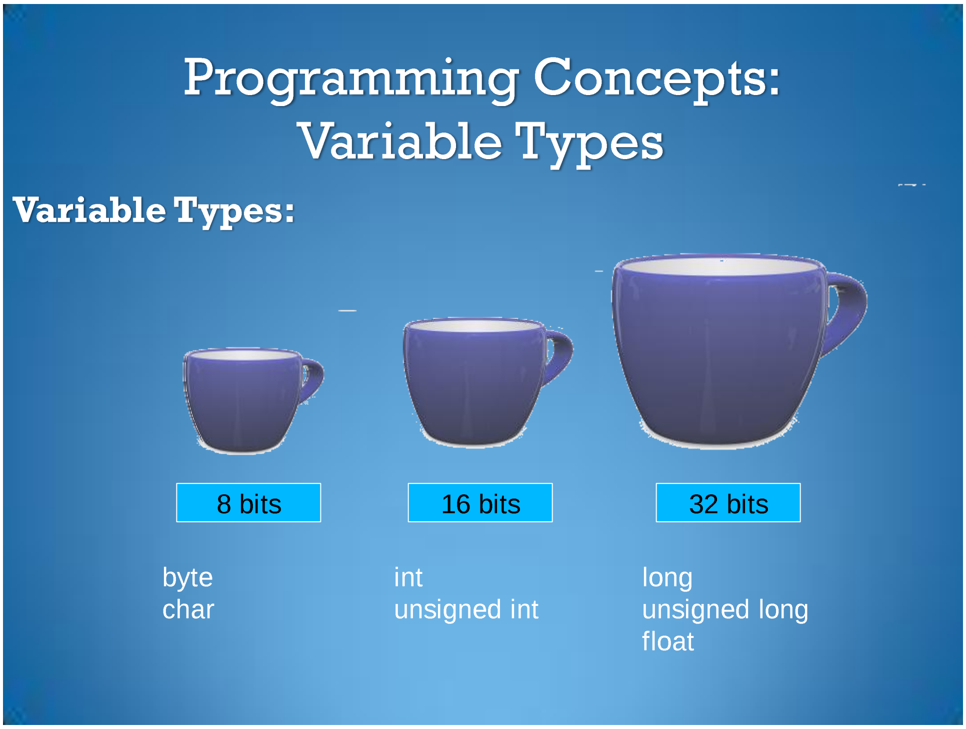
Three commands to know…

- **pinMode**(pin, INPUT/OUTPUT);
- <u>ex</u>: **pinMode**(13, OUTPUT);
- **digitalWrite**(pin, HIGH/LOW);
- <u>ex</u>: digitalWrite(13, HIGH);
- **delay**(time_ms);
- <u>ex</u>: delay(2500); // delay of 2.5 sec.
- **// NOTE: -> commands are CASE-sensitive**



Arduino Timing

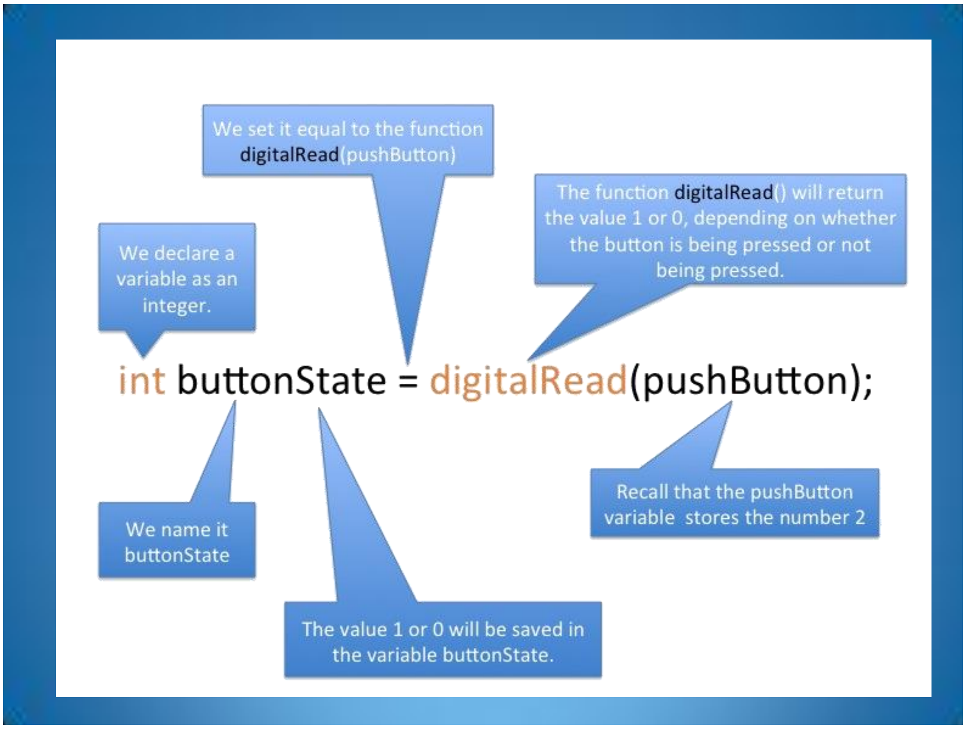
- `delay(ms)`
  - Pauses for a few milliseconds
- `delayMicroseconds(us)`
  - Pauses for a few microseconds
- More commands: arduino.cc/en/Reference/HomePage



Programming Concepts: Variable Types

Variable Types:

8 bits	16 bits	32 bits
byte	int	long
char	unsigned int	unsigned long
		float



We set it equal to the function digitalRead(pushButton)

The function digitalRead() will return the value 1 or 0, depending on whether the button is being pressed or not being pressed.

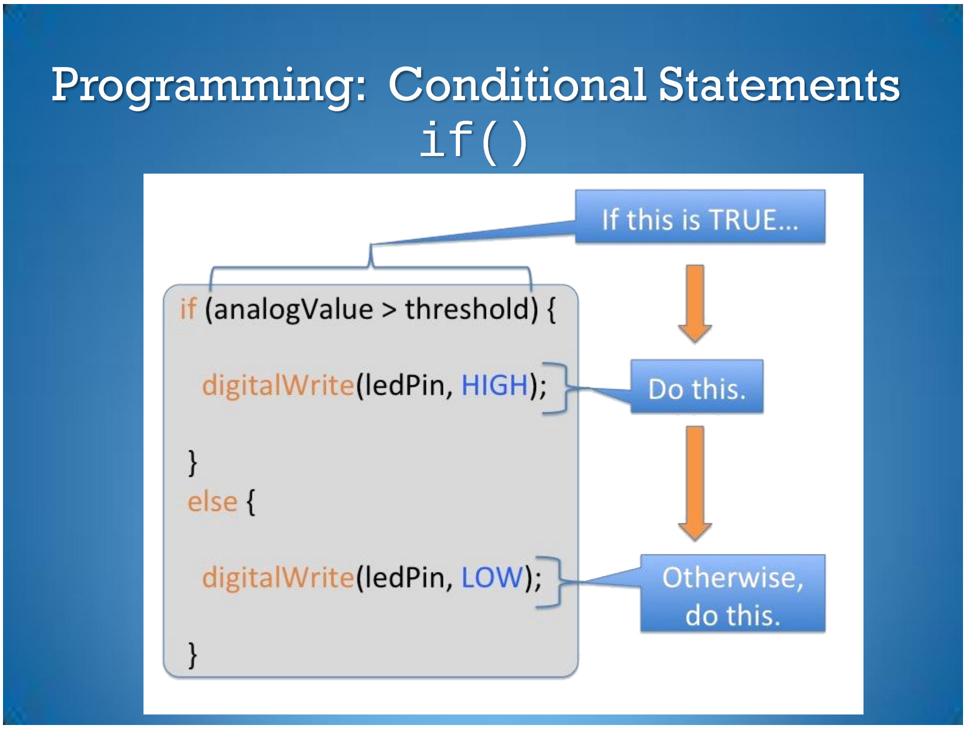
We declare a variable as an integer.

```
int buttonState = digitalRead(pushButton);
```

We name it buttonState

The value 1 or 0 will be saved in the variable buttonState.

Recall that the pushButton variable stores the number 2



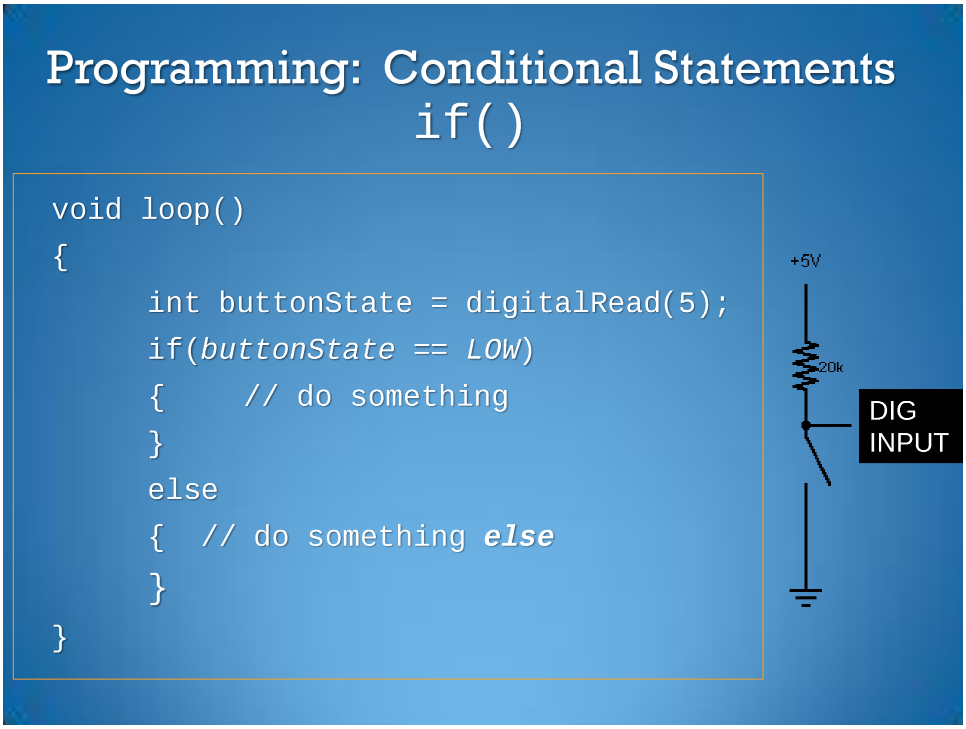
Programming: Conditional Statements `if()`

If this is TRUE…

```
if (analogValue > threshold) {

  digitalWrite(ledPin, HIGH);

}
else {

  digitalWrite(ledPin, LOW);

}
```

Do this.

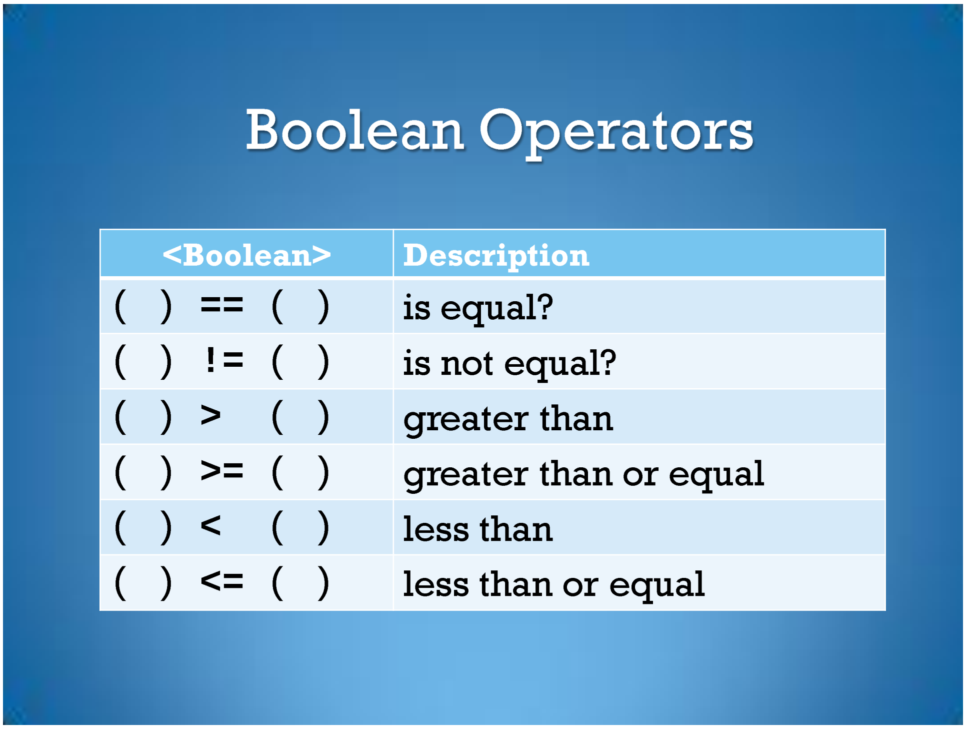
Otherwise, do this.



Programming: Conditional Statements if()

```
void loop()
{
        int buttonState = digitalRead(5);
        if(buttonState == LOW)
        {       // do something
        }
        else
        {   // do something else
        }
}
```

+5V
20k
DIG INPUT



Boolean Operators

<Boolean>	Description
( ) == ( )	is equal?
( ) != ( )	is not equal?
( ) > ( )	greater than
( ) >= ( )	greater than or equal
( ) < ( )	less than
( ) <= ( )	less than or equal

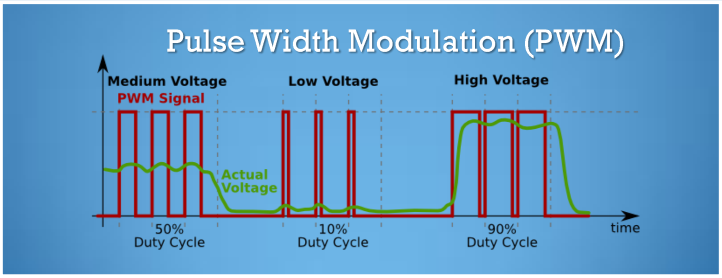
Fading in and Fading Out (Analog or Digital?)

- A few pins on the Arduino allow for us to modify the output to mimic an analog signal.

- This is done by a technique called:
- Pulse Width Modulation (PWM)

Concepts: Analog vs. Digital

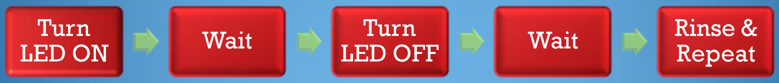
To create an analog signal, the microcontroller uses a technique called PWM. By varying the duty cycle, we can mimic an "average" analog voltage.



Let's get to coding…

- Project #1 – Turn LED on and off
  - "Hello World" of Physical Computing

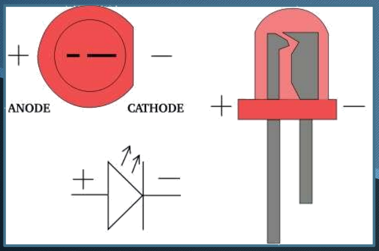
- *Psuedo-code – how should this work?*



Turn LED ON → Wait → Turn LED OFF → Wait → Rinse & Repeat

Add an External LED to pin 13

- **File > Examples > Digital > Blink**
- LED's have polarity
  - – Negative indicated by flat side of the housing and a short leg



www.instructables.com

Project #1: Wiring Diagram

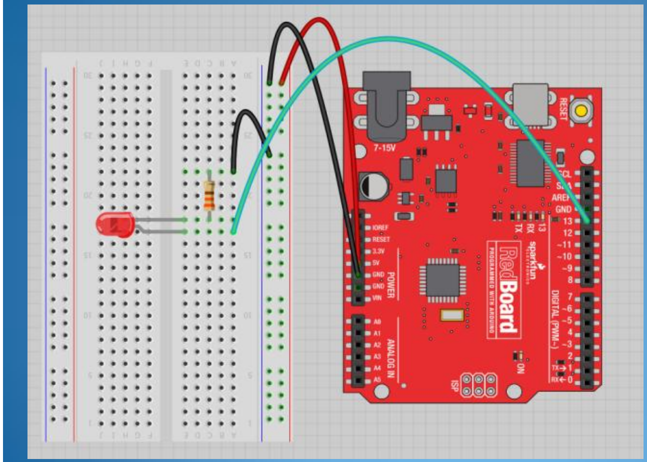
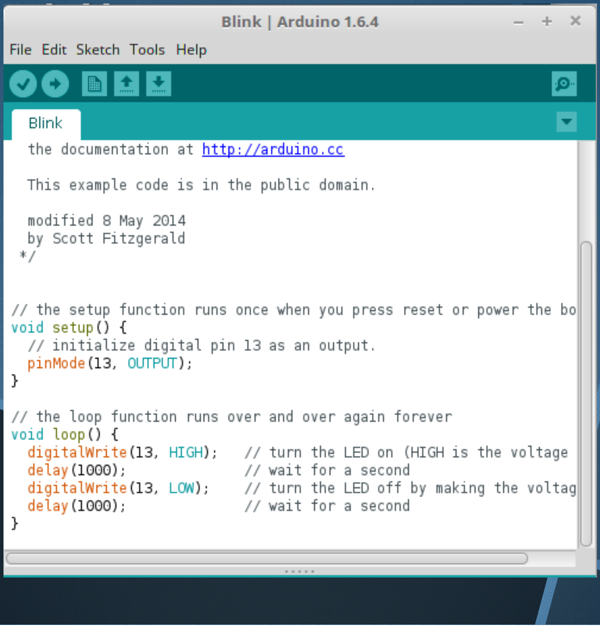


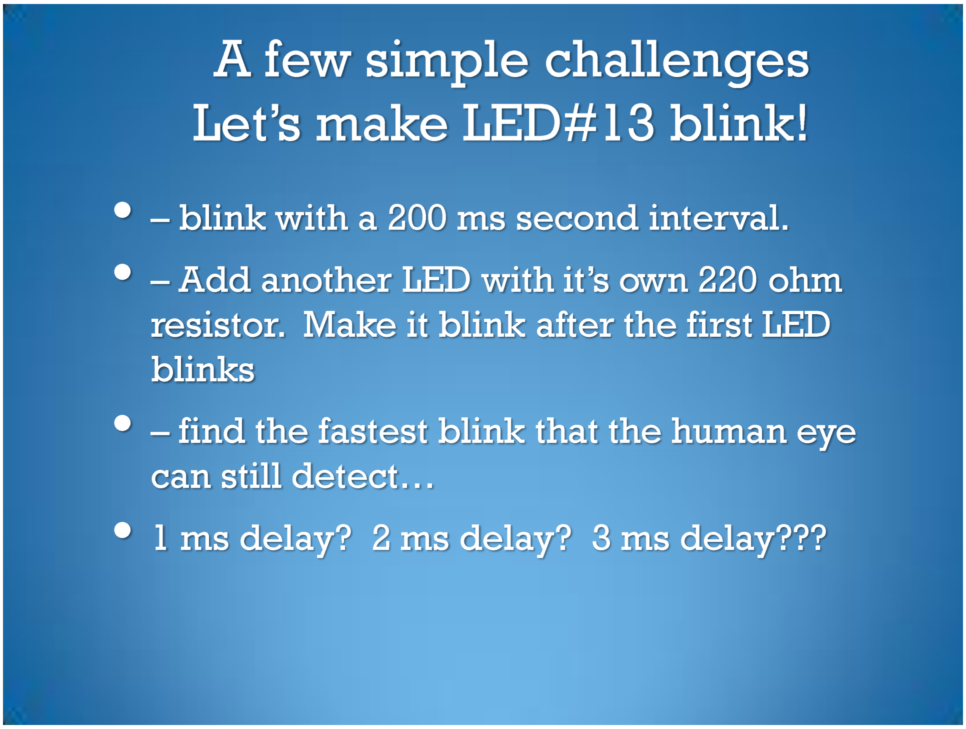
Image created in Fritzing

Move the green wire from the power bus to pin 13 (or any other Digital I/O pin on the Arduino board.

Our First Program

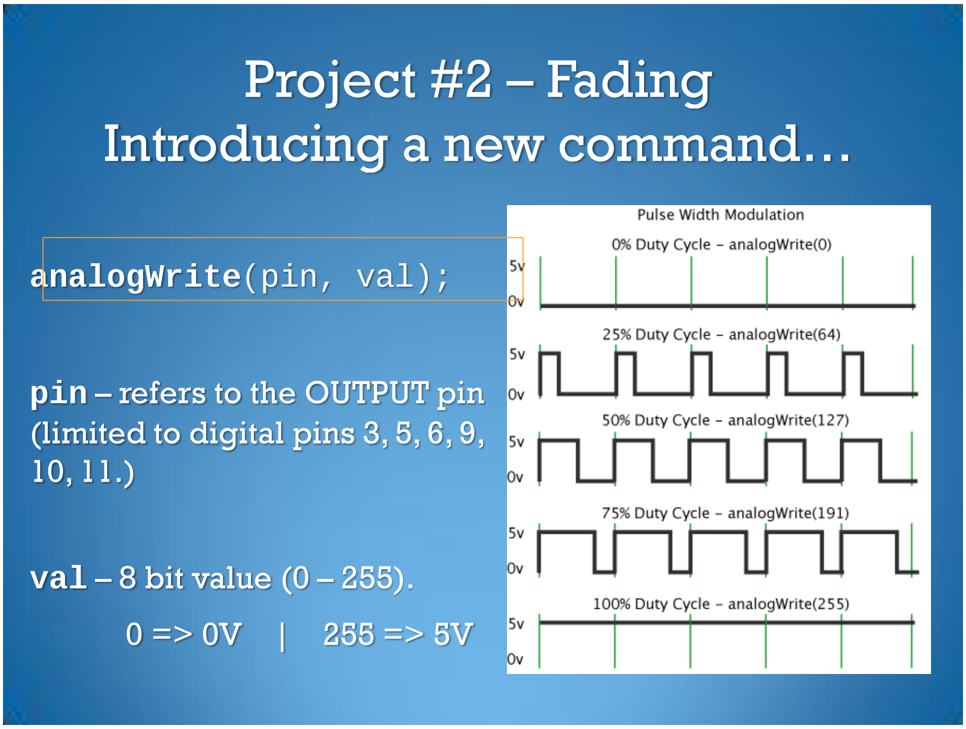


```
  the documentation at http://arduino.cc

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald
 */


// the setup function runs once when you press reset or power the bo
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltag
  delay(1000);              // wait for a second
}
```



A few simple challenges
Let’s make LED#13 blink!

- – blink with a 200 ms second interval.
- – Add another LED with it’s own 220 ohm resistor. Make it blink after the first LED blinks
- – find the fastest blink that the human eye can still detect…
- 1 ms delay? 2 ms delay? 3 ms delay???



Project #2 – Fading
Introducing a new command…

```
analogWrite(pin, val);
```

`pin` – refers to the OUTPUT pin (limited to digital pins 3, 5, 6, 9, 10, 11.)

`val` – 8 bit value (0 – 255).

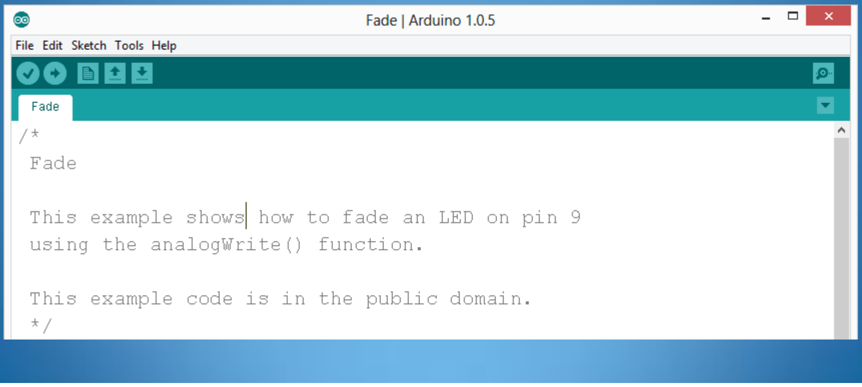
0 => 0V | 255 => 5V

Pulse Width Modulation
0% Duty Cycle – analogWrite(0)
5v
0v
25% Duty Cycle – analogWrite(64)
5v
0v
50% Duty Cycle – analogWrite(127)
5v
0v
75% Duty Cycle – analogWrite(191)
5v
0v
100% Duty Cycle – analogWrite(255)
5v
0v

Move one of your LED pins over to Pin 9

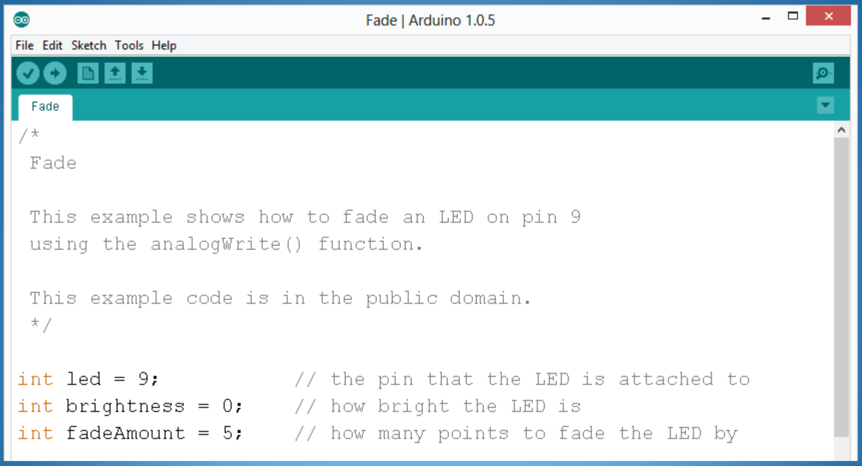
In Arduino, open up:

File → Examples → 01.Basics → Fade



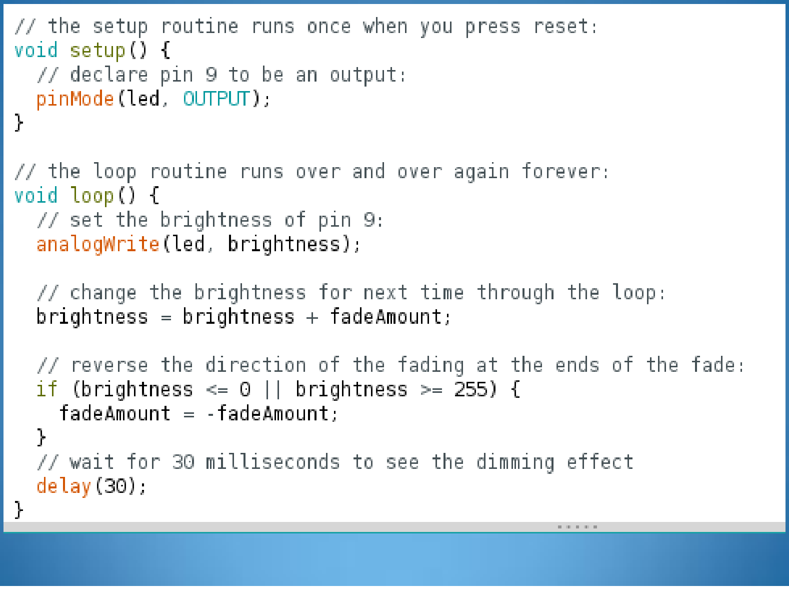
```
/*
 Fade

 This example shows how to fade an LED on pin 9
 using the analogWrite() function.

 This example code is in the public domain.
 */
```

Fade - Code Review



```
/*
 Fade

 This example shows how to fade an LED on pin 9
 using the analogWrite() function.

 This example code is in the public domain.
 */

int led = 9;           // the pin that the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by
```

Fade - Code Review



```
// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

Project# 2 -- Fading

- **Challenge 2a** – Change the rate of the fading in and out. There are at least two different ways to do this – can you figure them out?

- **Challenge 2b** – Use 2 (or more) LEDs – so that one fades in as the other one fades out.

Piezo Buzzer

If a voltage is applied across a piece of piezoelectric material it will flex, or change dimension in one direction.

In a piezo 'speaker, a piece of piezoelectric material is attached to a diaphragm.

When the signal voltage from an amplifer is applied across the piezo element, its flexure or dimensional movement is transferred to the diaphragm.

Example

- Write a program that plays two notes (440 & 660 Hz) for 200ms and delays 1 second
- Load the PiezoBuzzer Sketch to your Arduino
- Note the short lead on the buzzer goes to ground. If you have a Piezo Drum Disk, it is not polarity sensitive.
- Connect the long signal lead to Digital Pin 3

```
tone(speakerPin, frequency, duration); // play the tone

delay(duration); //wait for the tone to finish
```

Try the Piezo Tone Melody Mario Sketch

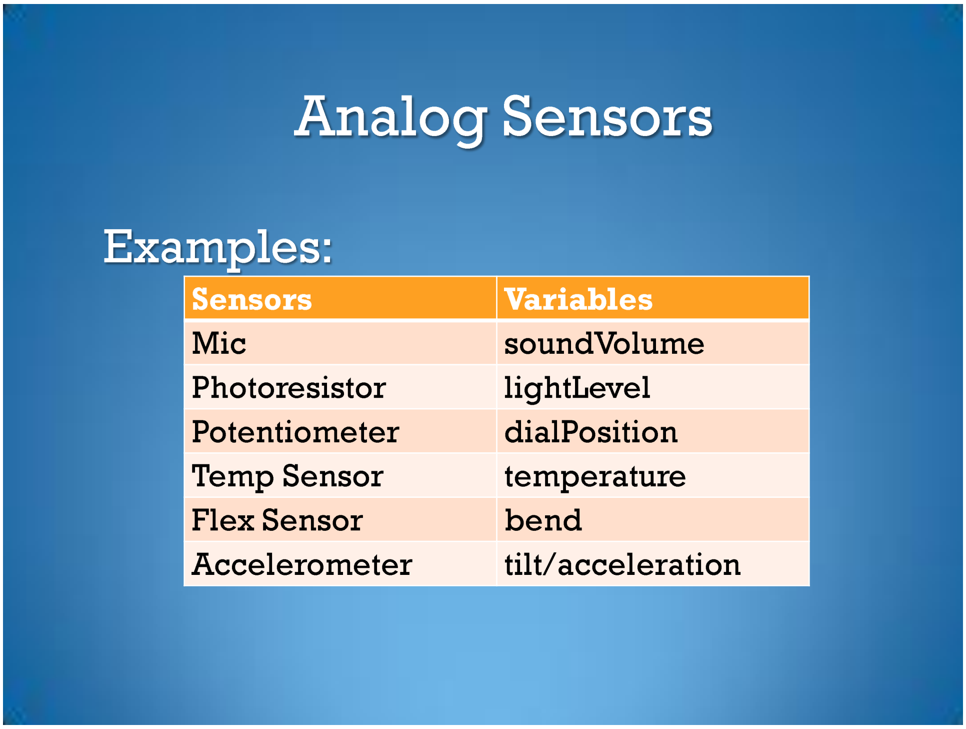
Input

Input is any signal entering an electrical system .

- Both digital and analog sensors are forms of input
- Input can also take many other forms: Keyboards, a mouse, infrared sensors, biometric sensors, or just plain voltage from a circuit

Digital Sensors

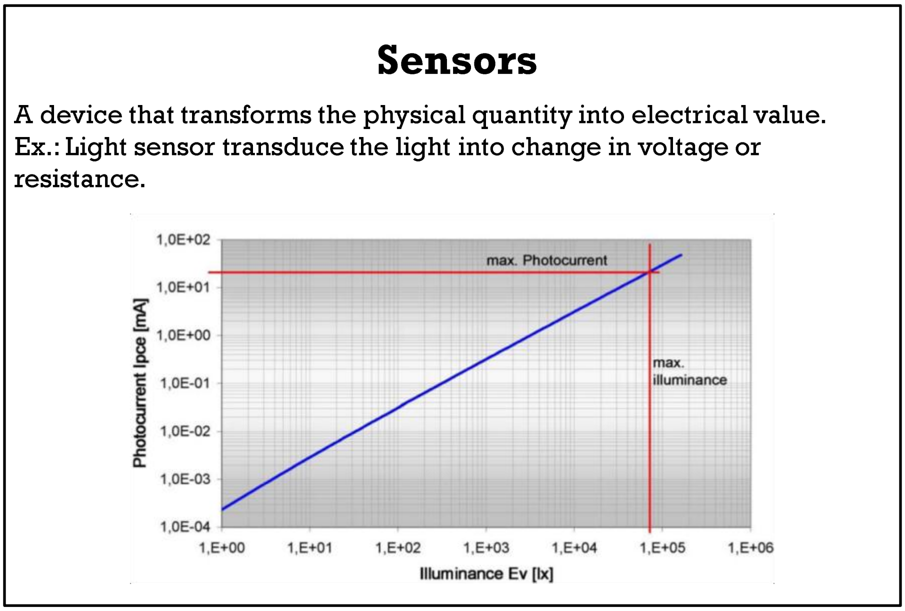
- Digital sensors are more straight forward than Analog
- No matter what the sensor there are only two settings: On and Off
- Signal is always either HIGH (On) or LOW (Off)
- Voltage signal for HIGH will be a little less than 5V on your Uno
- Voltage signal for LOW will be 0V on most systems



Analog Sensors

Examples:

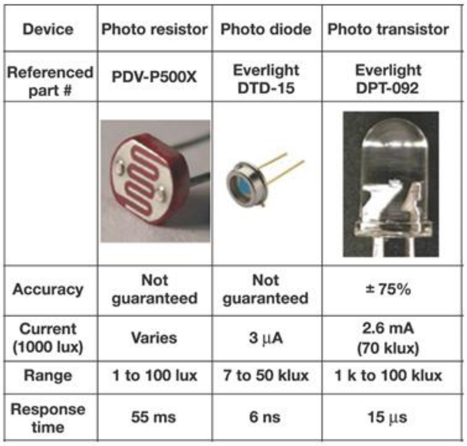
Sensors	Variables
Mic	soundVolume
Photoresistor	lightLevel
Potentiometer	dialPosition
Temp Sensor	temperature
Flex Sensor	bend
Accelerometer	tilt/acceleration



Sensors

A device that transforms the physical quantity into electrical value.
Ex.: Light sensor transduce the light into change in voltage or resistance.

Light sensors:



Device	Photo resistor	Photo diode	Photo transistor
Referenced part #	PDV-P500X	Everlight DTD-15	Everlight DPT-092
Accuracy	Not guaranteed	Not guaranteed	± 75%
Current (1000 lux)	Varies	3 μA	2.6 mA (70 klux)
Range	1 to 100 lux	7 to 50 klux	1 k to 100 klux
Response time	55 ms	6 ns	15 μs

analogRead()

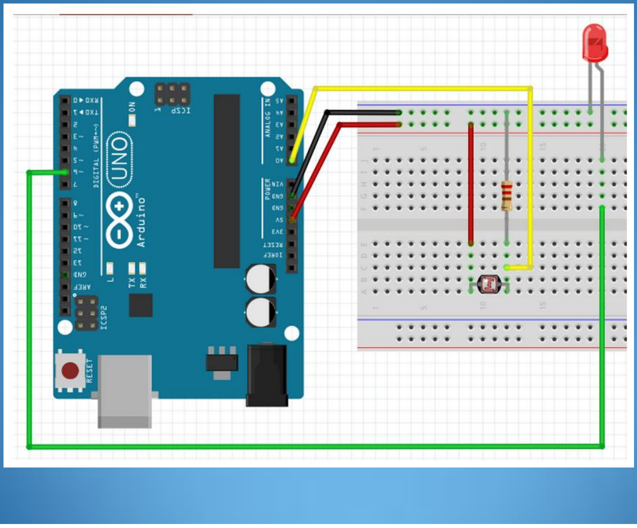
- Arduino uses a 10-bit A/D Converter:
- this means that you get input values from 0 to 1023
  - 0 V → 0
  - 5 V → 1023
- Ex:
- `int sensorValue = analogRead(A0);`

Photo Resistor:

The value of the resistance depends on the incident light density.

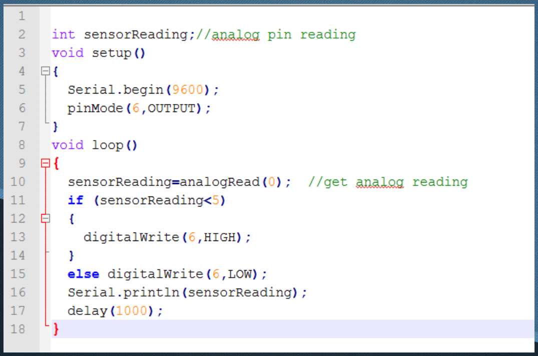
1 K-Ohm at light, 10 K-Ohm at darkness.

Project #3 – Analog Input



Code for Light Sensitive Resistor

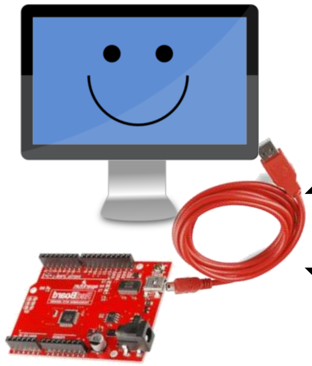
Open LightResistor Sketch
Build circuit as in previous diagram
Open serial monitor to see Analog readings



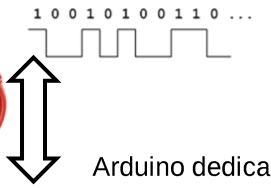
```

int sensorReading;//analog pin reading
void setup()
{
  Serial.begin(9600);
  pinMode(6,OUTPUT);
}
void loop()
{
  sensorReading=analogRead(0);  //get analog reading
  if (sensorReading<5)
  {
    digitalWrite(6,HIGH);
  }
  else digitalWrite(6,LOW);
  Serial.println(sensorReading);
  delay(1000);
}
```

Using Serial Communication

Method used to transfer data between two devices.

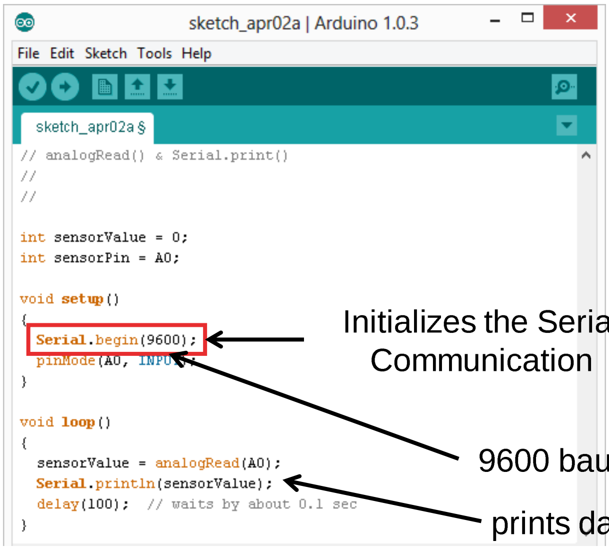


Data passes between the computer and Arduino through the USB cable. Data is transmitted as zeros ('0') and ones ('1') sequentially.



Arduino dedicates Digital I/O pin # 0 to receiving and Digital I/O pin #1 to transmit.

Serial Monitor & analogRead()



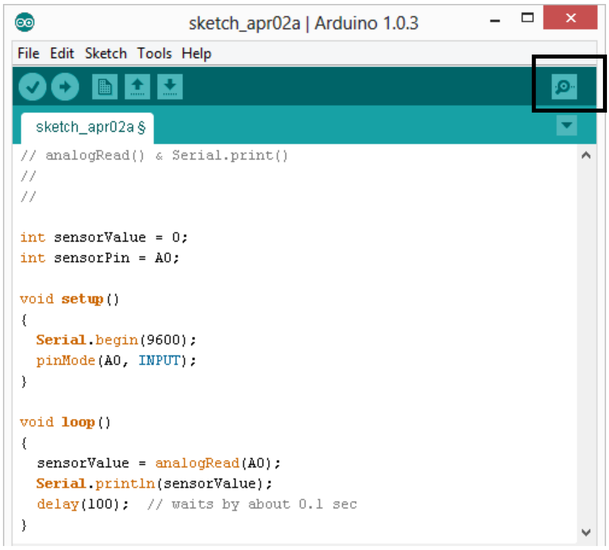
```
// analogRead() & Serial.print()
//
//

int sensorValue = 0;
int sensorPin = A0;

void setup()
{
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop()
{
  sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  delay(100);  // waits by about 0.1 sec
}
```

Initializes the Serial Communication

9600 baud data rate

prints data to serial bus

Serial Monitor & analogRead()



```
// analogRead() & Serial.print()
//
//

int sensorValue = 0;
int sensorPin = A0;

void setup()
{
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop()
{
  sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  delay(100);  // waits by about 0.1 sec
}
```

Opens up a Serial Terminal Window

Additional Serial Communication
Sending a Message

```
void loop ( )
{
  Serial.print(“Hands on “) ;
  Serial.print(“Learning ”) ;
  Serial.println(“is Fun!!!”) ;

}
```

PIR Motion Sensor

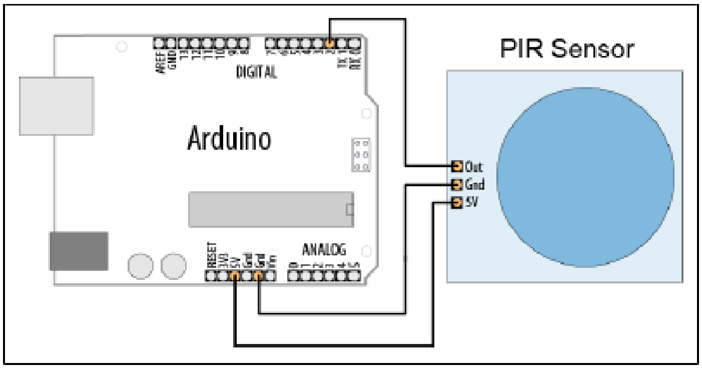
Detects levels of infrared radiation. (Everything emits radiation)

The sensor is split in two halves and detects change in motion

The two halves are wired up so that they cancel each other out.

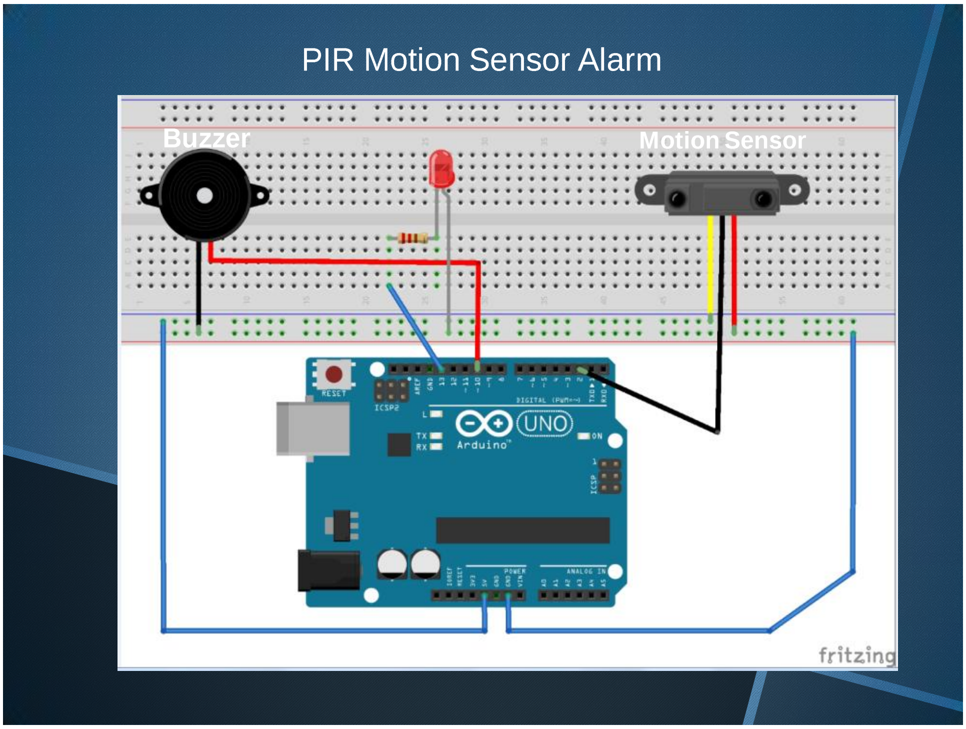
If one half sees more or less IR radiation than the other, the output will swing high or low.

Using PIR motion sensors



PIR Motion Sensor Alarm Example

```
int ledPin = 13;            // choose the pin for the LED
int inputPin = 2;           // choose the input pin (for PIR
sensor)
int pirState = LOW;         // we start, assuming no motion
detected
int val = 0;                // variable for reading the pin status
int pinSpeaker = 10;        //Set up a speaker on a PWM pin
                                        (digital 9, 10, or 11)
```

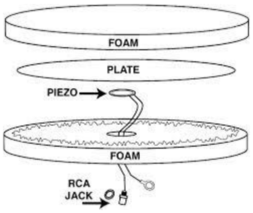


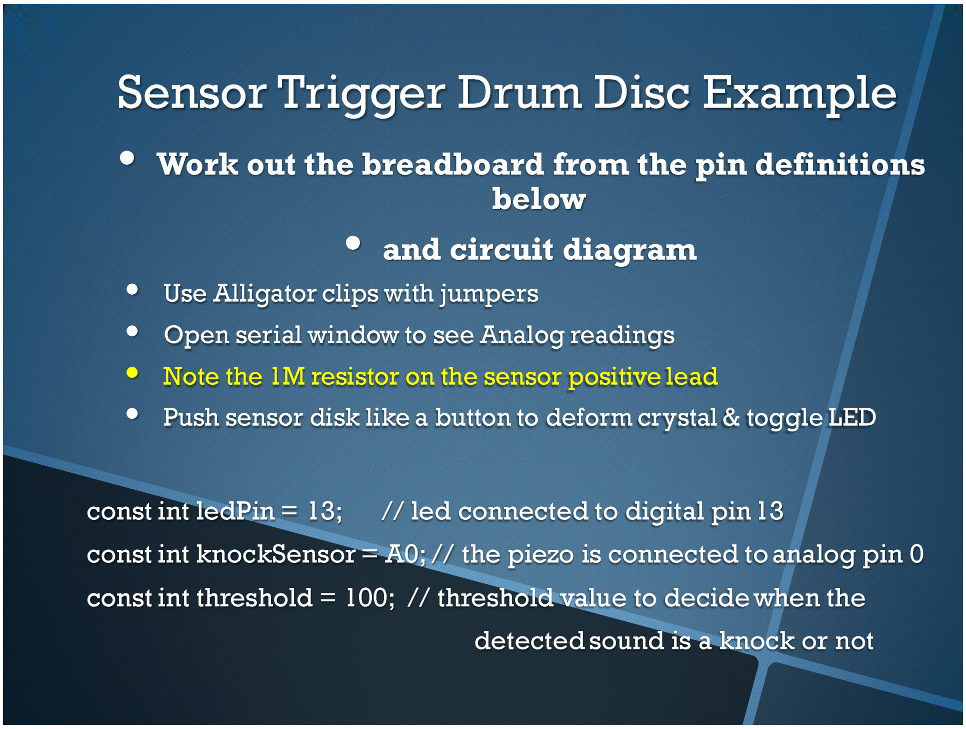
PIR Motion Sensor Alarm

Sensor Trigger Drum Disc

The Piezo Sensor is Pressure Sensitive

- A piezo is a thin metal disc with a hard, ceramic material in the center.
- As a high-impedance, passive transducer, it converts mechanical energy to electrical energy.
- Crystals and certain ceramic materials generate an electrical signal when stress is applied.

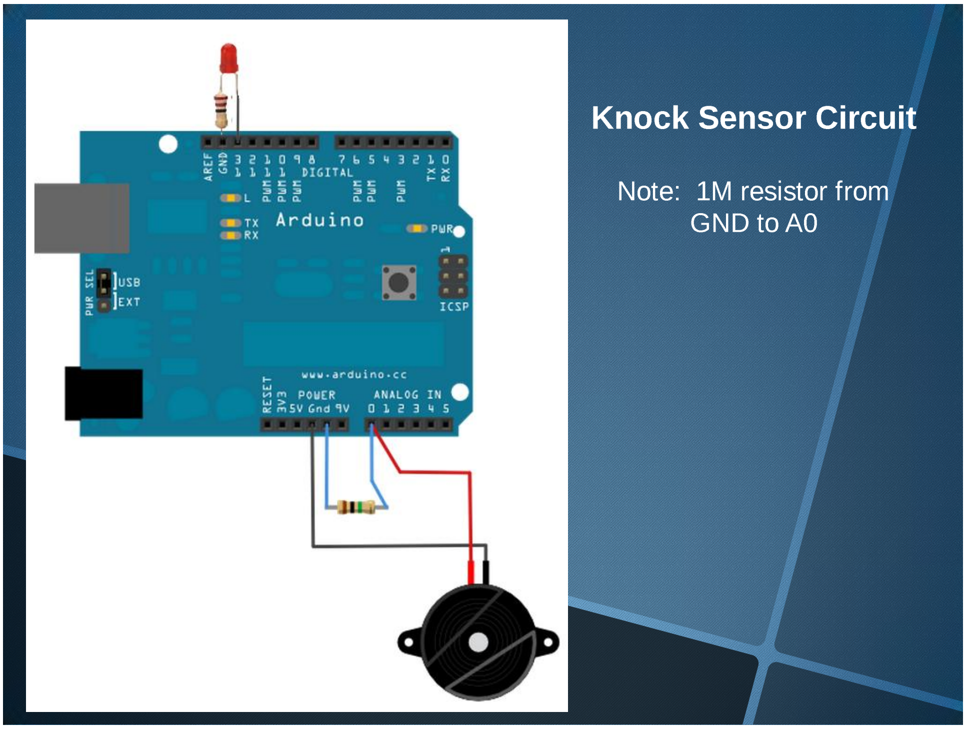




Sensor Trigger Drum Disc Example

- **Work out the breadboard from the pin definitions below**
  - **and circuit diagram**
- Use Alligator clips with jumpers
- Open serial window to see Analog readings
- Note the 1M resistor on the sensor positive lead
- Push sensor disk like a button to deform crystal & toggle LED

```
const int ledPin = 13;      // led connected to digital pin 13
const int knockSensor = A0; // the piezo is connected to analog pin 0
const int threshold = 100;  // threshold value to decide when the
                                    detected sound is a knock or not
```



Knock Sensor Circuit

Note: 1M resistor from GND to A0

Recommended Books

- Make AVR Programming (Elliot Williams) ***Great Book!!***
- Getting Started with Arduino 2nd Edition
- Make an Arduino Controlled Robot
- TinyAVR Microsontroller Projects for the Evil Genius
- Learn Electronics with Arduino
- Programming Your Home (Mike Riley)
- Arduino Cookbook
- Arduino Robotics
- Arduino Bots and Gadgets

Recommended Books

- Make AVR Programming (Elliot Williams) ***Great Book!!***
- Getting Started with Arduino 2nd Edition
- Make an Arduino Controlled Robot
- TinyAVR Microsontroller Projects for the Evil Genius
- Learn Electronics with Arduino
- Programming Your Home (Mike Riley)
- Arduino Cookbook
- Arduino Robotics
- Arduino Bots and Gadgets

Questions?

Arduino Workshop

Robin C. Moseley
President & CEO
Great Lakes IT, Inc

OPPORTUNITY
ADVANCEMENT
INNOVATION
in WORKFORCE DEVELOPMENT

OAI Chicago Southland (OCS)
214 Forest Blvd Park Forest, IL 60466
(708)-283-5020
Oaiinc.org

Great Lak
Inc.
Computer Consulting
www.greatlakesitinc.com
1.847.867.2280