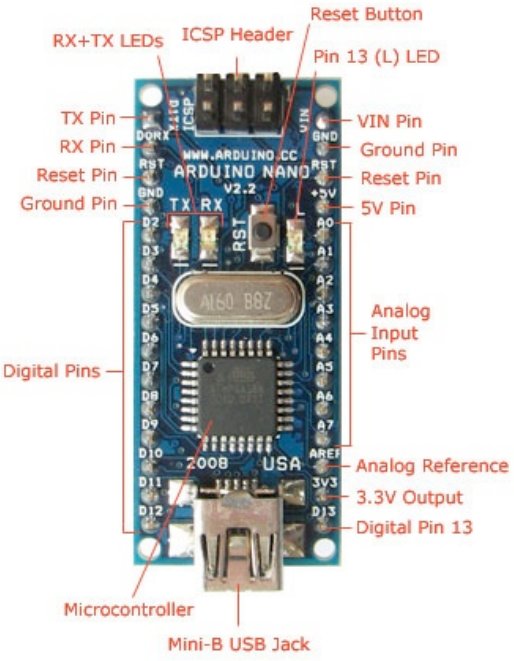
Lab: Arduino Basics and Exercises

Introduction to Arduino

- The Arduino is an open-source computing platform that is comprised of a microcontroller development platform paired with a development environment for writing software in a manner that the board can execute.

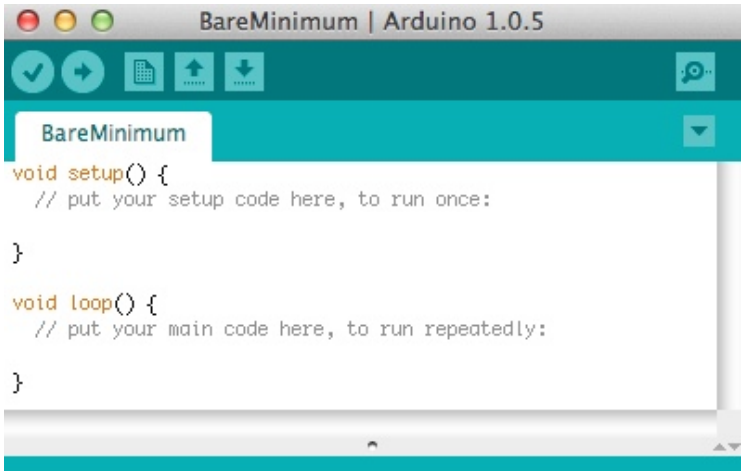
Arduino Nano: Schematic and Quick Background



- **Microcontroller** holds code and executes commands that you specify
- **Mini-B USB Jack** enables you to upload your program to the Arduino via a USB connection
- **Pins** can serve as digital inputs and outputs, and you can address them in the programs you'll write
- **Power** of 5V is provided over the USB cable and is generally sufficient power for the board
- **Reset Pin** can be used to restart the execution of your program

Code Structure: BareMinimum

You must include two functions in *every* Arduino program: void **setup**() and void **loop**()



Useful Code

Structure

void setup() code inside runs once you press reset
void loop() the loops runs over and over again forever

Syntax

// single line comment
/* (multi-line comment) */

Data Types

void
int integers, e.g. 32

Time

delay(milliseconds)
e.g. delay(1000) means delay of 1000 ms = 1 s

Digital I/O (Input/Output)

pinMode(pin, [INPUT, OUTPUT])
e.g. pinMode(pin1, INPUT) initializes pin1 as an input
digitalWrite(pin, value)
e.g. digitalWrite(pin1, 0)
int **digitalRead**(pin)
*e.g. int **digitalRead**(pin1) reads the value of pin1*

Libraries

Serial.begin(speed)
e.g. Serial.begin(9600) initializes serial communication at 9600 bits per second
Serial.println("text")
e.g. Serial.println("Hello world!") starts a new line and prints Hello world!
Serial.print(number)
e.g. Serial.print(8) prints "8"

Arduino Exercises

1. Get the LED light blinking on and off, 1 time per second.
2. Set up the serial monitor, so that a fixed string is communicated to the serial monitor.
3. Get the LED blinking at a frequency of 1 blink per second, and print a 1 when the light is on and a 0 when the light is off.
4. Get the serial terminal to echo a letter.
5. Add in a function called 'echo' than you can call in the main loop, for all future exercises.
6. Control the state of the LED through the serial terminal. If you type a 1, the light should turn on. If you type a 0, the light should turn off. If you type anything else, you should get a message indicating that the user should press a 1 or a 0.
7. Connect the Arduino to Bluetooth, so you can type a 1 or 0 on a cell phone and have the Arduino led light turn on and off.

8. Create a simple calculator: type a number with 3 digits (e.g. “102” or “005”) and then an operator (e.g. “+” “-” “*” or “/”) and then a second three-digit number. On a new line, the Arduino should give you the answer.