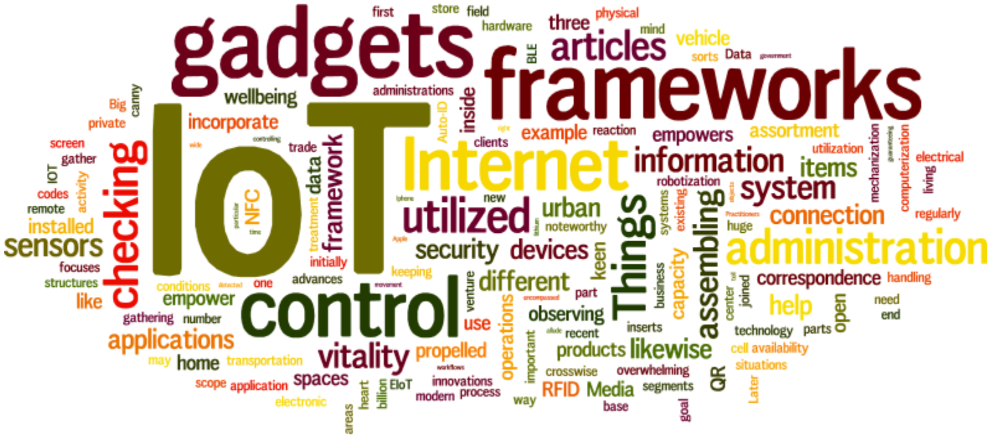
Internet of Things

Introduction to Arduino Programming

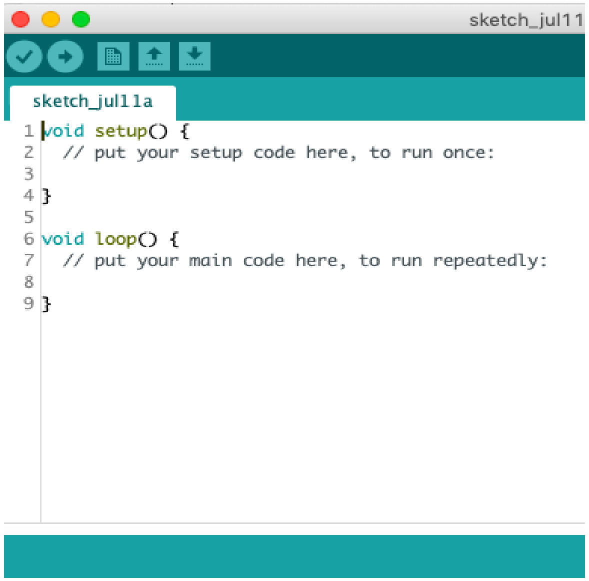


Thanks to Dr.. Manas Khatu

Introduction

- The Arduino Software (IDE) allows you to write programs (i.e. **sketches**) and upload them to your board.

- A sketch is consists of two **mandatory** functions:
  - ✓ **Setup( )** -- it is executed **once**
  - ✓ **Loop( )** -- it is executed **repeatedly**



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

}

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

}
```

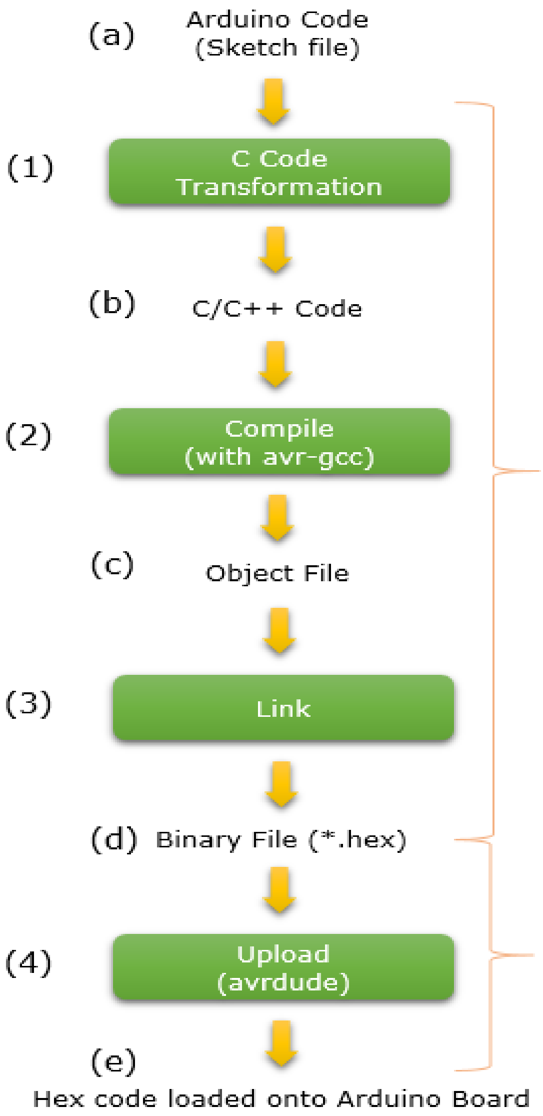
- Setup( ) is used for
  - ✓ initialization of serial communication
  - ✓ defining pinMode
  - ✓ declaring variables

- Loop( ) is used for
  - ✓ writing the main code which has to execute continuously.
  - ✓ e.g. reading inputs from the sensors, triggering outputs to the external device, etc.

Cont…

- **Sketches** are **compiled** by avr-gcc / avr-g++
  - It is based on C/C++ programming language

- So, the program syntax is almost similar to C/C++
  - Supported data types
  - Variables
  - Constants
  - Control structure
  - Looping structure
  - Arrays
  - Strings
  - Function



(a) Arduino Code (Sketch file)

(1) C Code Transformation

(b) C/C++ Code

(2) Compile (with avr-gcc)

(c) Object File

(3) Link

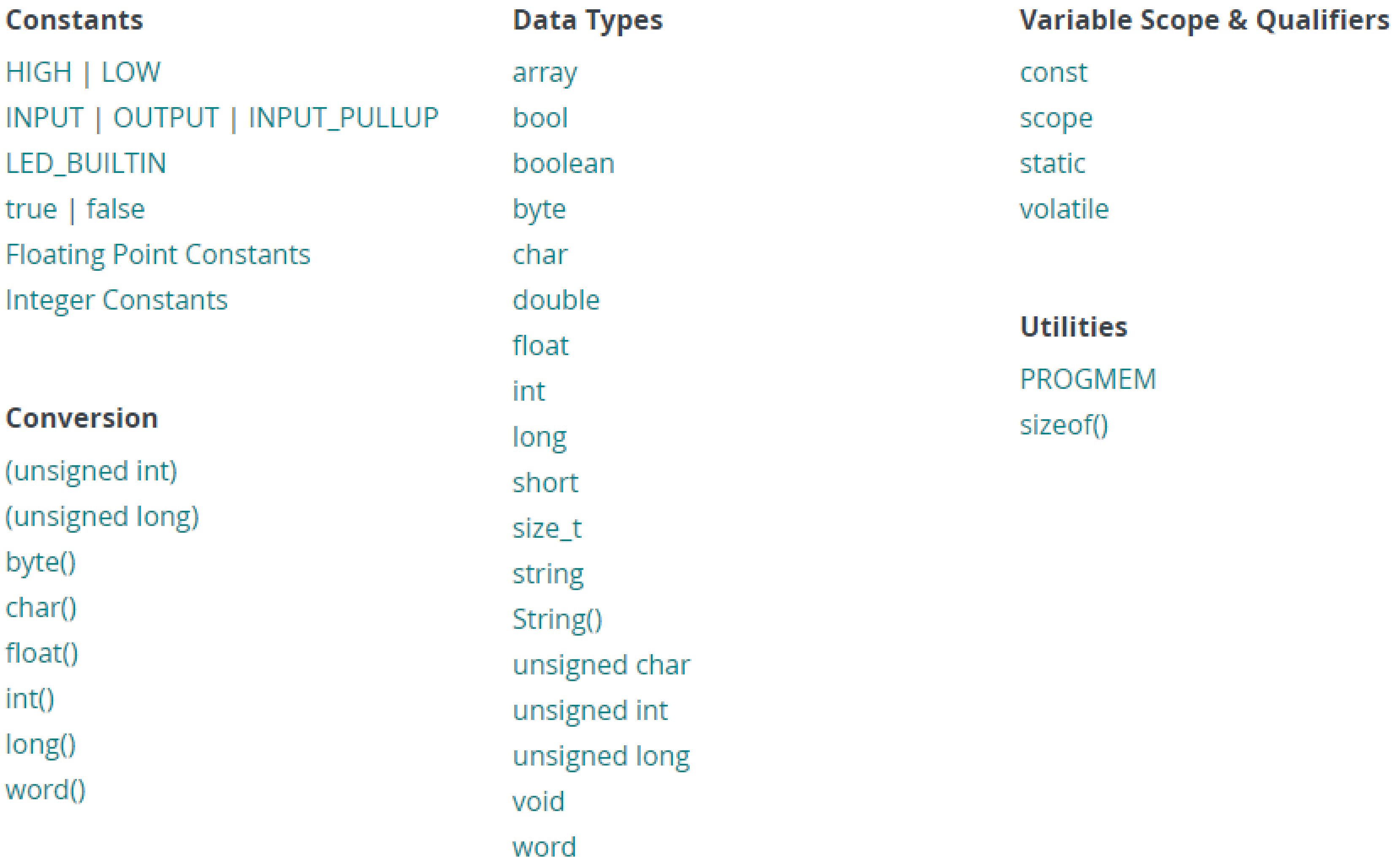
(d) Binary File (*.hex)

(4) Upload (avrdude)

(e) Hex code loaded onto Arduino Board

- One important extension is : **Arduino Libraries**
  - Libraries are a collection of code that makes it easy for you to connect to a sensor, display, module, etc.

Variables



Constants

HIGH | LOW

INPUT | OUTPUT | INPUT_PULLUP

LED_BUILTIN

true | false

Floating Point Constants

Integer Constants

Conversion

(unsigned int)

(unsigned long)

byte()

char()

float()

int()

long()

word()

Data Types

array

bool

boolean

byte

char

double

float

int

long

short

size_t

string

String()

unsigned char

unsigned int

unsigned long

void

word

Variable Scope & Qualifiers

const

scope

static

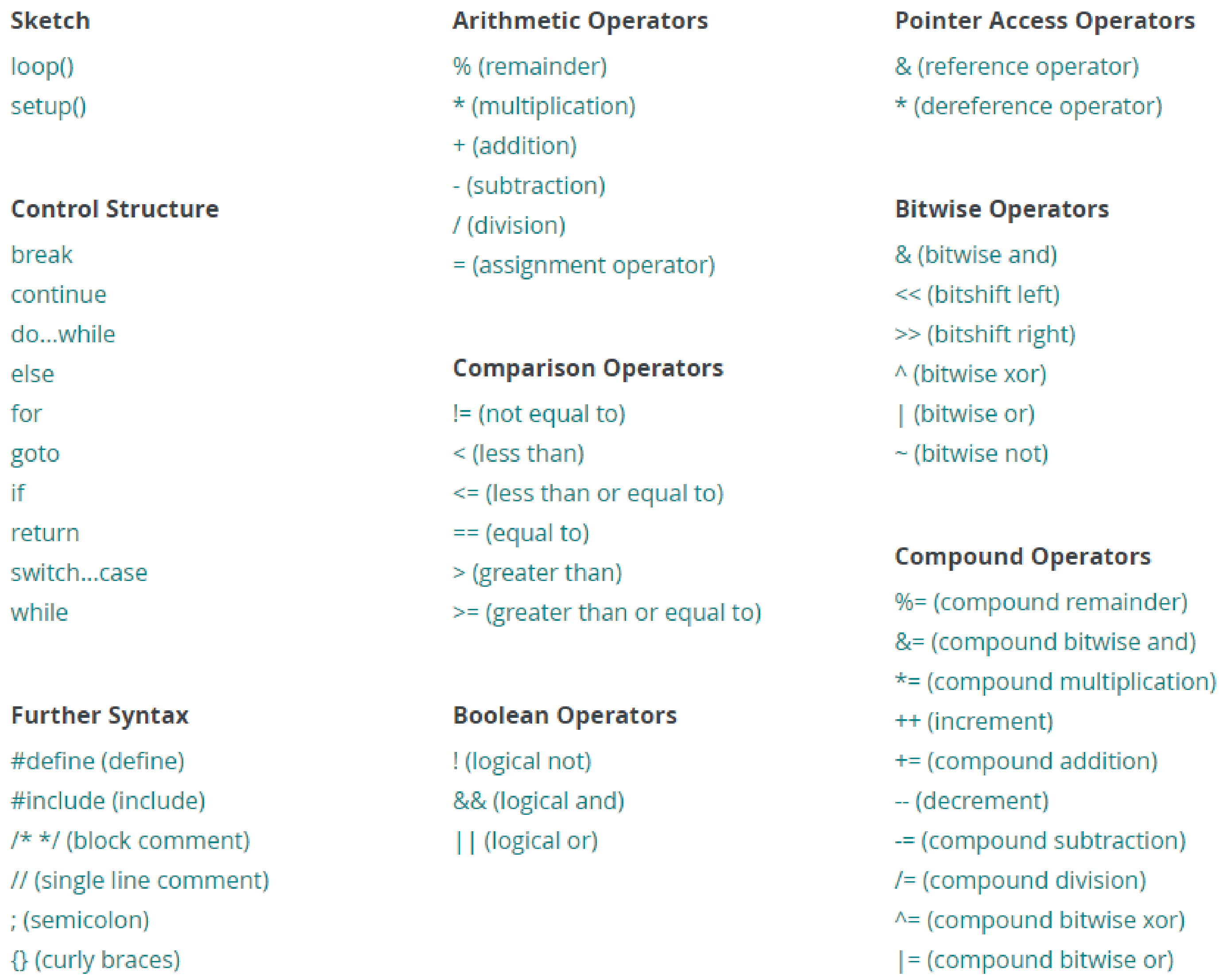
volatile

Utilities

PROGMEM

sizeof()

Operators & Structures



Sketch

loop()
setup()

Control Structure

break
continue
do...while
else
for
goto
if
return
switch...case
while

Further Syntax

#define (define)
#include (include)
/* */ (block comment)
// (single line comment)
; (semicolon)
{} (curly braces)

Arithmetic Operators

% (remainder)
* (multiplication)
+ (addition)
- (subtraction)
/ (division)
= (assignment operator)

Comparison Operators

!= (not equal to)
< (less than)
<= (less than or equal to)
== (equal to)
> (greater than)
>= (greater than or equal to)

Boolean Operators

! (logical not)
&& (logical and)
| | (logical or)

Pointer Access Operators

& (reference operator)
* (dereference operator)

Bitwise Operators

& (bitwise and)
<< (bitshift left)
>> (bitshift right)
^ (bitwise xor)
| (bitwise or)
~ (bitwise not)

Compound Operators

%= (compound remainder)
&= (compound bitwise and)
*= (compound multiplication)
++ (increment)
+= (compound addition)
-- (decrement)
-= (compound subtraction)
/= (compound division)
^= (compound bitwise xor)
|= (compound bitwise or)

Few Built-in Functions

https://www.arduino.cc/reference/en/

- **pinMode (pin, mode)**
  - It configures the specified pin to behave either as input or as output
  - By default the digital pins in Arduino function as input.
  - **pin**: is the number of the pin whose mode needs to be set
  - **mode**: can be INPUT, OUTPUT, INPUT_PULLUP.

```
pinMode(9,OUTPUT);
```

- **digitalReadPin(pin)**
  - Reads the value from a specified digital pin, either HIGH or LOW.

```
val = digitalRead(inPin);
```

- **digitalWrite(pin, value)**
  - Used for output by using the LOW/HIGH logic level (i.e. 0V / 5V)
  - **value**: LOW / HIGH

```
digitalWrite(10,HIGH);
```

- **analogRead(pin)**
  - Access and gets value from a particular Analog pin having 10-bit resolution (i.e. 10-bit ADC)
  - Returns: 0-1023 (integer)
  - Arduino UNO yields a resolution between readings of: 5 volts / 1024 units. It will map input voltages between 0 and the operating voltage(5V or 3.3V) into integer values between 0 and 1023.
  - The input range can be changed using analogReference()

```
val = analogRead(A3);
```

- **analogWrite(pin, value)**
  - Write the analog value (PWM wave) to a pin
  - **value:** it is the duty cycle value between 0 and 255 (as 6 pins).
  - Note: analogRead values go from 0 to 1023, analogWrite values from 0 to 255

```
analogWrite(9, val / 4);
```

Cont…

- **delay(ms)**
  - Pause the program for the amount of time (in millisecond) specified by **ms**

delay(1000); // wait for a second

- **Serial.begin(speed)**
  - It sets the **speed** in bps (baud rate) for serial data transmission from computer to Arduino board

Serial.begin(9600);

- **Serial.available ()**
  - Returns: the number of bytes (characters) available to read

if (Serial.available() > 0) { }

- **Serial.print(value)**
  - Print data to the serial port as human-readable ASCII text
  - Numbers are printed using ASCII character for each digit
  - Floats are printed as ASCII digits (upto 2 decimal places)
  - Bytes are send as a single character
  - Characters and Strings are sent as is.

Serial.print("I received: ");

- **Serial.print(value, format)**
  - The optional 2nd argument specifies the base (format) to use
  - **format:** BIN / OCT / DEC / HEX

Serial.print(i,DEC);
// Print Decimal value of number i

- **Serial.println(value) , Serial.println(value, format)**
  - Additionally it returns the number of bytes written

Cont…

- **Serial.read()**
  - Reads incoming serial data.
- **Serial.write(val) or .write(str) or .write(buf, len)**
  - Writes binary data to the serial port.
  - This data is sent as a byte or series of bytes; to send the characters representing the digits of a number use the print() function instead.
- **Trigonometry:**
  - **cos()**
  - **sin()**
  - **tan()**
- **Math:**
  - **abs()**
  - **max()**
  - **min()**
  - **pow()**
  - **sq()**
  - **sqrt()**
  - **random()**
  - **randomSeed()**

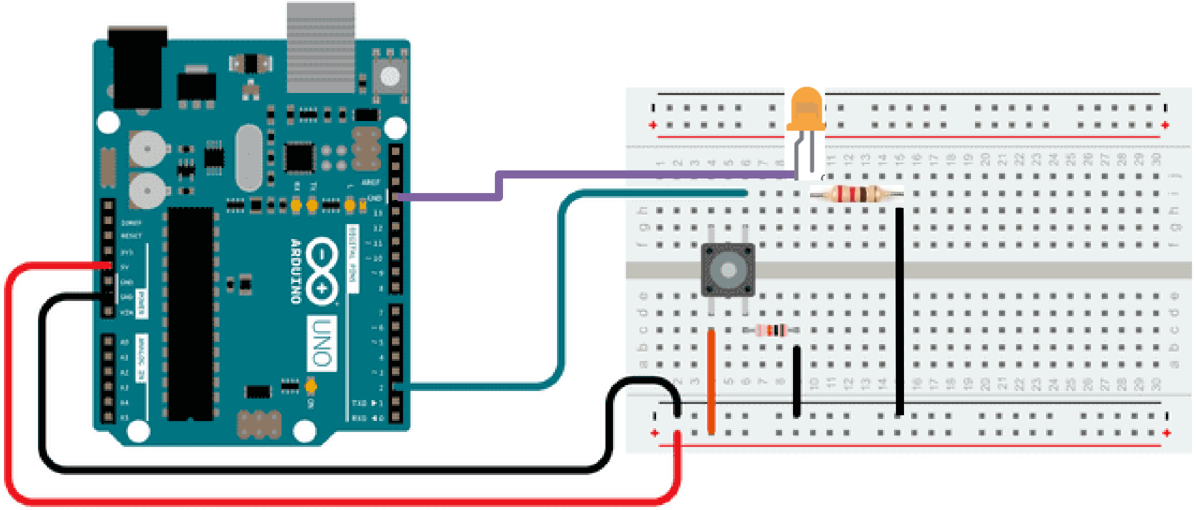
```
incomingByte = Serial.read();
```

```
Serial.write(45); // send a byte with the value 45

int bytesSent = Serial.write("hello");  //send the string "hello" and return the length of the string.
```

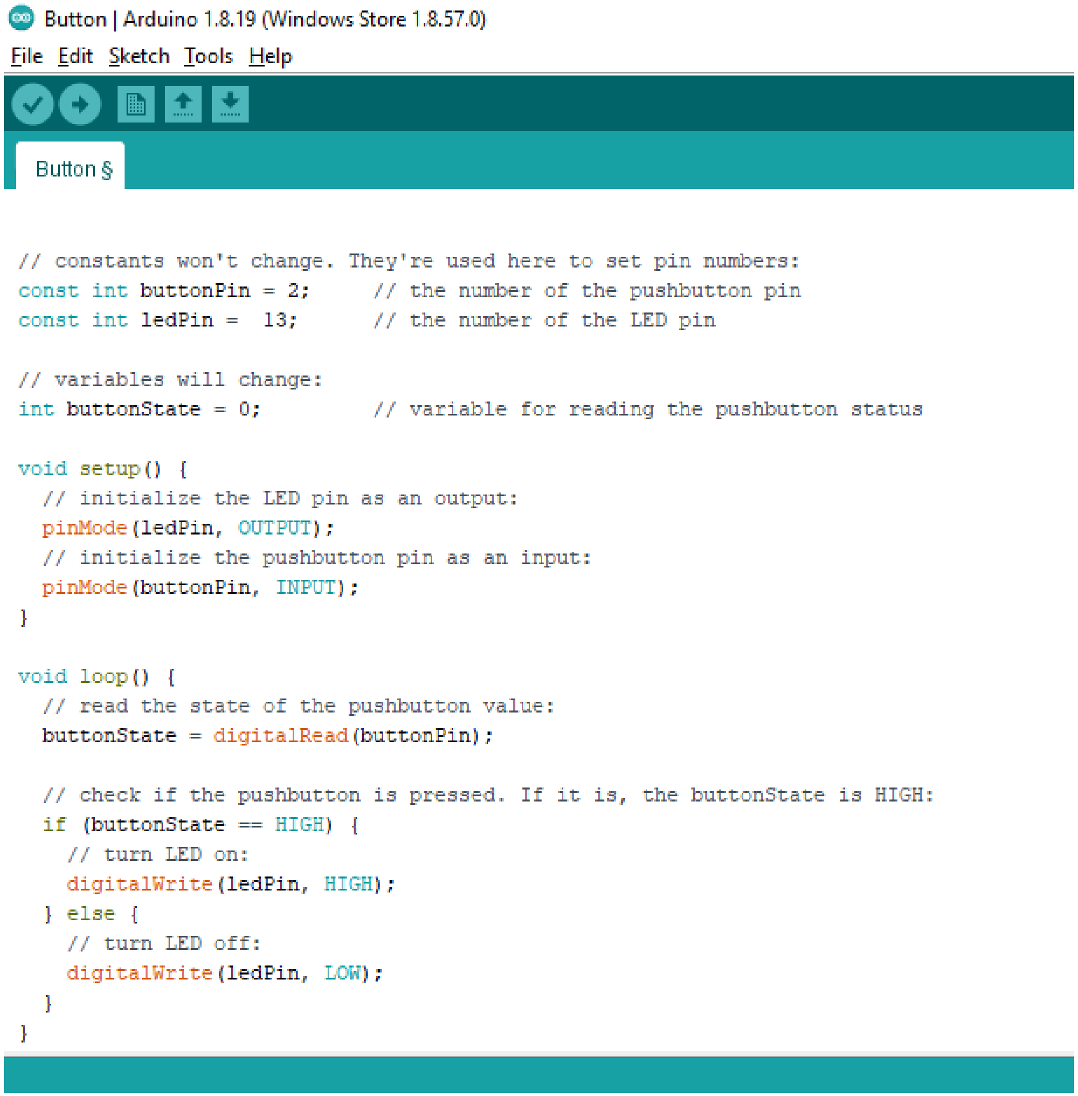
Example 1: Digital Read-Write

- Objective:
  - Turns on and off a LED connected to digital pin 13, when pressing a pushbutton attached to pin 2.



- The circuit:
  - LED attached from pin 13 to ground through 220 ohm resistor
  - One leg of the Pushbutton attached to pin 2
  - That same leg of the button connects through a pull-down resistor (here 10K ohm) to ground.
  - The other leg of the button connects to the 5 volt supply.

Cont…


Button | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Button §

```
// constants won't change. They're used here to set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

- When the pushbutton is open (**unpressed**)
  - there is no connection between the two legs of the pushbutton, so the pin is connected to ground (through the pull-down resistor) and we read a LOW.

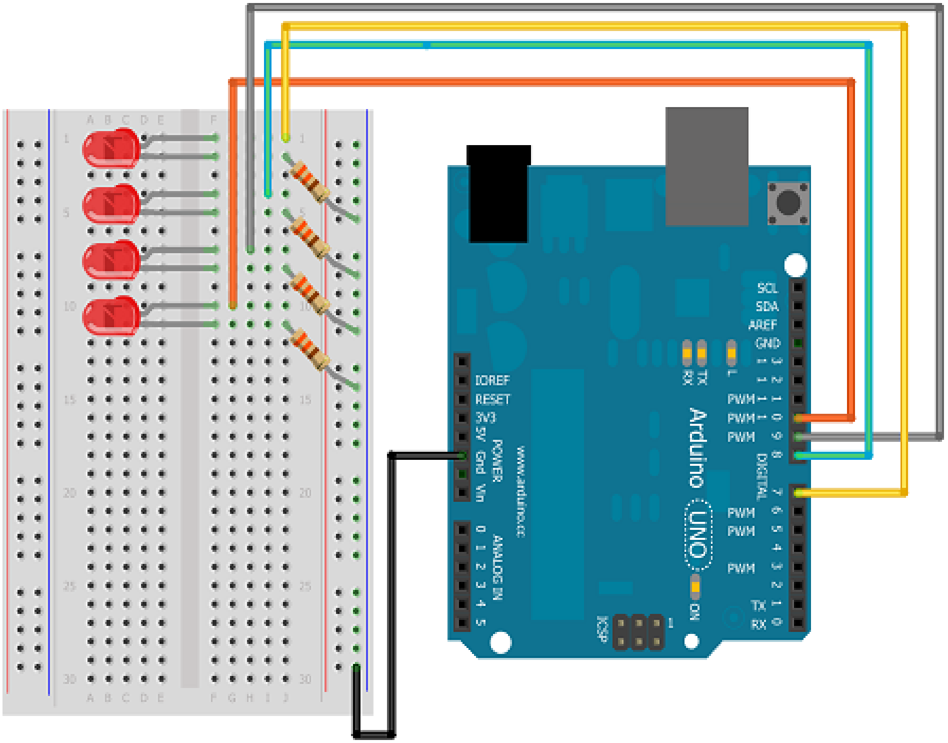
- When the button is closed (**pressed**)
  - it makes a connection between its two legs, connecting the pin to 5 volts, so that we read a HIGH.

Example 2: Binary Counter in LED

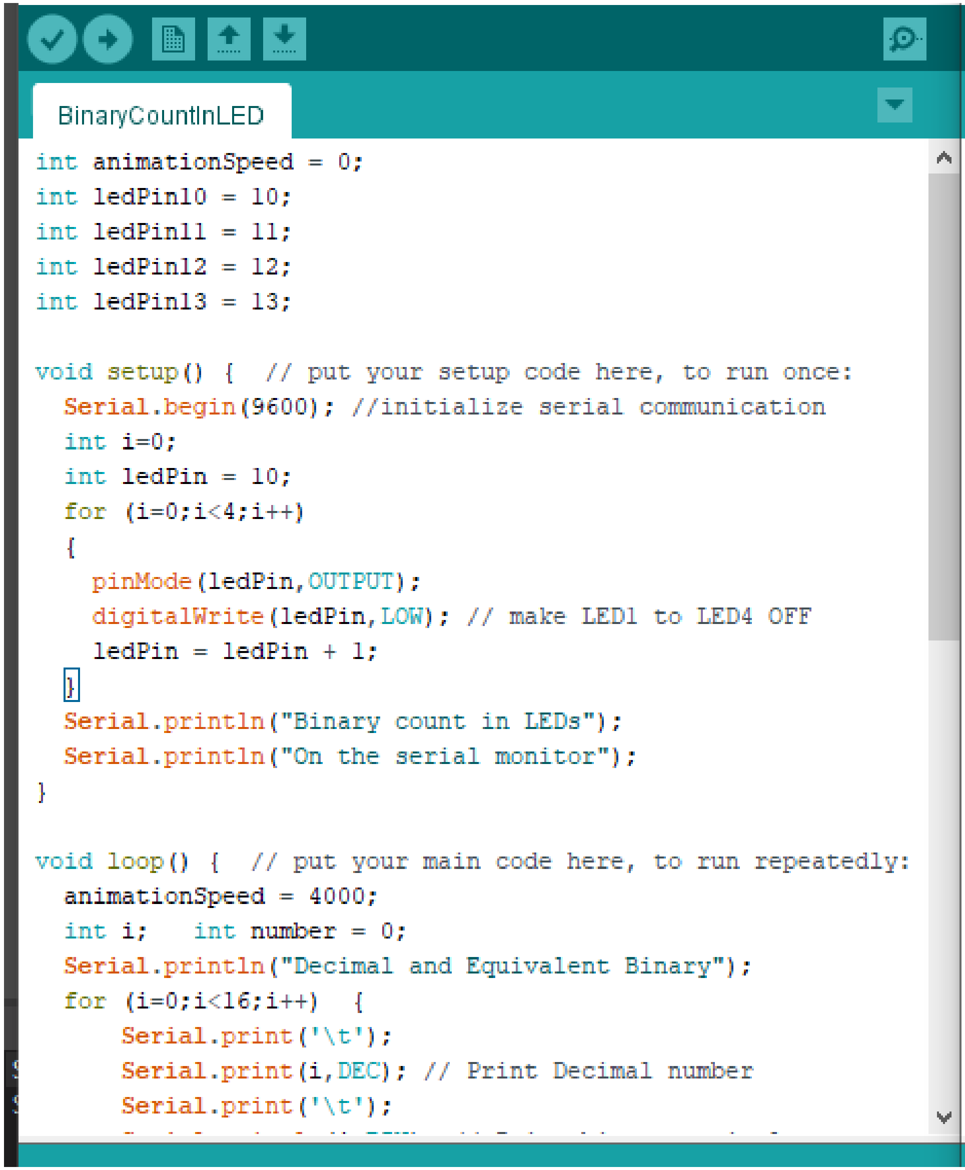
- Requirements:
  - Arduino UNO
  - USB connector
  - Breadboard
  - 4 piece LEDs
  - 4 piece 1K ohm resistor
  - Arduino IDE

- Connection:
  - Place the LED and resistor on breadboard
  - Connect the bradboard power with Arduino
  - Connect the LED with Arduino
  - Connect the Arduino board with PC/Laptop

- Arduino Programming
  - Install IDE in PC/Laptop
  - Run the IDE
  - Select the Arduino board in IDE
  - Select the connected COM port
  - Start writing new sketch



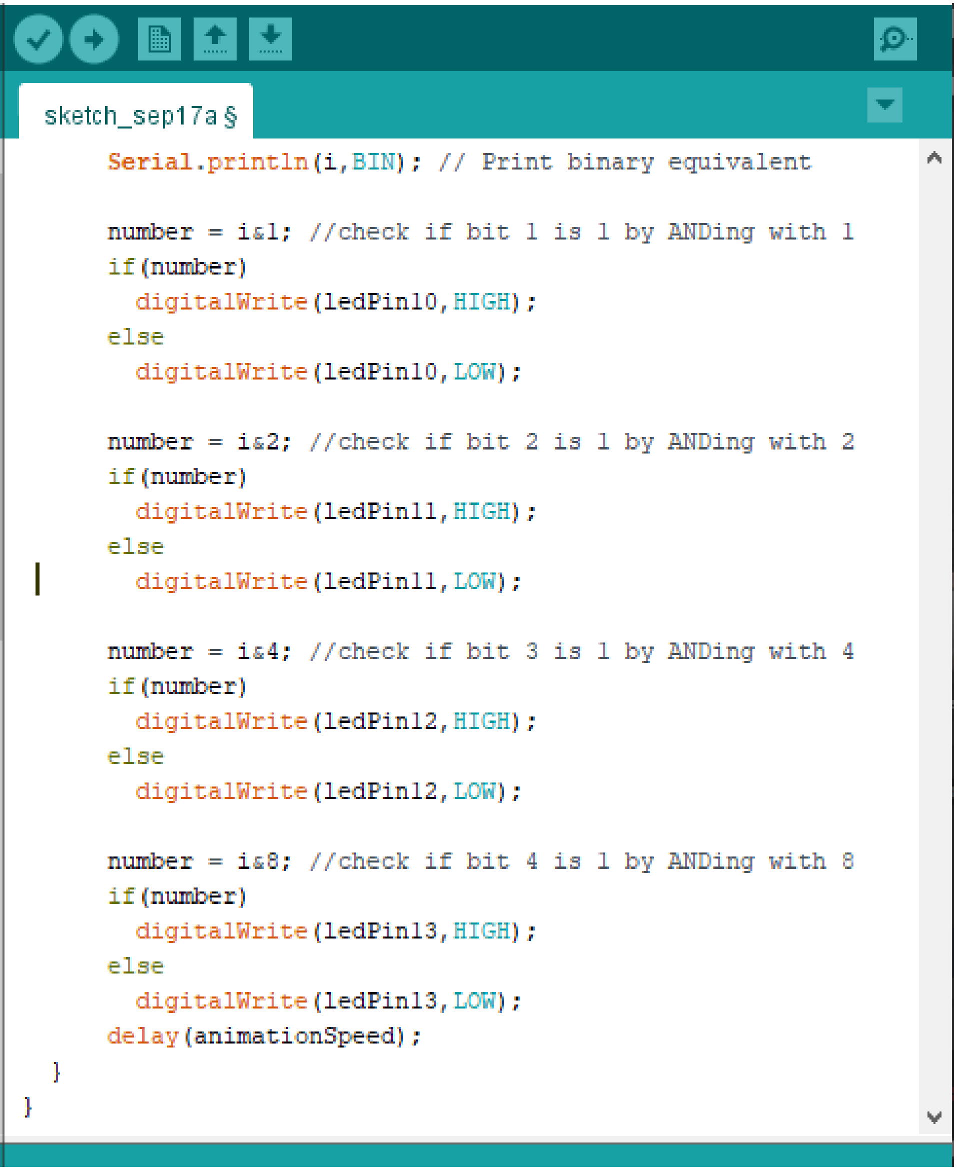
Sketch of Binary Counter



```
int animationSpeed = 0;
int ledPin10 = 10;
int ledPin11 = 11;
int ledPin12 = 12;
int ledPin13 = 13;

void setup() {  // put your setup code here, to run once:
  Serial.begin(9600); //initialize serial communication
  int i=0;
  int ledPin = 10;
  for (i=0;i<4;i++)
  {
    pinMode(ledPin,OUTPUT);
    digitalWrite(ledPin,LOW); // make LED1 to LED4 OFF
    ledPin = ledPin + 1;
  }
  Serial.println("Binary count in LEDs");
  Serial.println("On the serial monitor");
}

void loop() {  // put your main code here, to run repeatedly:
  animationSpeed = 4000;
  int i;   int number = 0;
  Serial.println("Decimal and Equivalent Binary");
  for (i=0;i<16;i++)  {
      Serial.print('\t');
      Serial.print(i,DEC); // Print Decimal number
      Serial.print('\t');
```



```
      Serial.println(i,BIN); // Print binary equivalent

      number = i&1; //check if bit 1 is 1 by ANDing with 1
      if(number)
        digitalWrite(ledPin10,HIGH);
      else
        digitalWrite(ledPin10,LOW);

      number = i&2; //check if bit 2 is 1 by ANDing with 2
      if(number)
        digitalWrite(ledPin11,HIGH);
      else
        digitalWrite(ledPin11,LOW);

      number = i&4; //check if bit 3 is 1 by ANDing with 4
      if(number)
        digitalWrite(ledPin12,HIGH);
      else
        digitalWrite(ledPin12,LOW);

      number = i&8; //check if bit 4 is 1 by ANDing with 8
      if(number)
        digitalWrite(ledPin13,HIGH);
      else
        digitalWrite(ledPin13,LOW);
      delay(animationSpeed);
  }
}
```

Demo on Binary Counter in LED

Live Demo

- See the live demo on
  - Connecting 4 LEDs with Arduino
  - Sketch writing, compiling, uploading and execution

18-09-2020 Dr. Manas Khatua

Read Analog Voltage

- ADC provide digital output which is proportional to analog value.

- To know what is input analog value, we need to convert the received digital value back to analog value through program.

Aout = digital value * (Vref/2^n – 1)

- Example:
  - digital value = 512 and ADC is 10-bit with 5V Vref.
  - What analog voltage is giving the respective digital value?

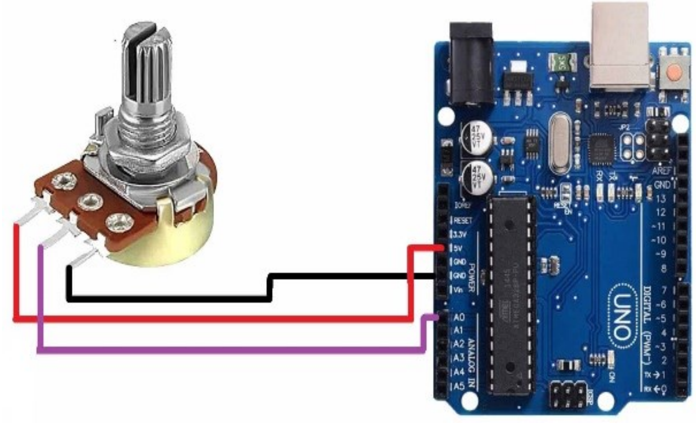
Aout = 512 * (5 V / 1023) = 2.5 V

digitalValue = analogRead (pin)

pin **-** number of analog pin which we want to read
digitalValue: 0 – 1023

Example: Read Analog Voltage

```
// select the input pin for the potentiometer
int sensorPin = A0;
// variable to store the value coming from the sensor
int digitalValue = 0;
float analogVoltage = 0.00;
 void setup() {
          Serial.begin(9600);
}
 void loop() {
        // read the value from the analog channel
        digitalValue = analogRead(sensorPin);
        Serial.print("digital value = ");
         //print digital value on serial monitor
        Serial.print(digitalValue);
        //convert digital value to analog voltage
        analogVoltage = (digitalValue * 5.00)/1023.00;
        Serial.print("  analog voltage = ");
        Serial.println(analogVoltage);
        delay(1000);
}
```

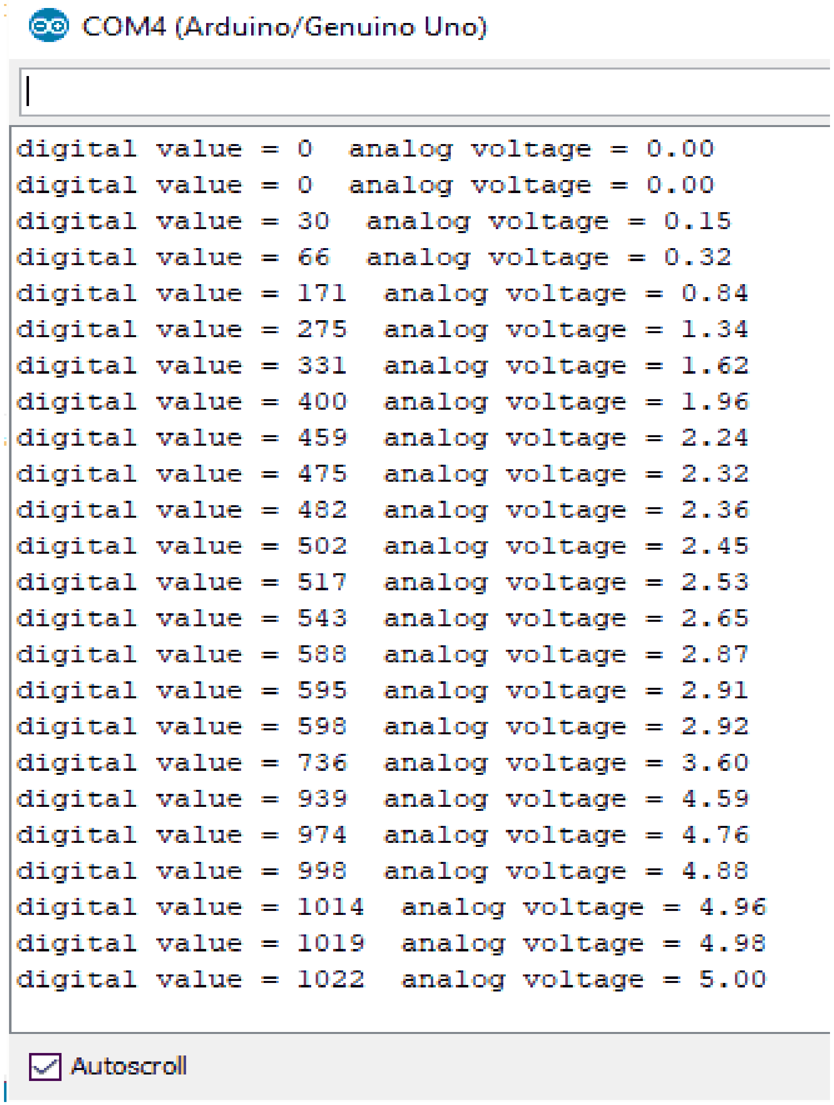


Pin 1 & 3 of Potentiometer: connect them to Vcc and GND of Arduino

Pin 2 of Potentiometer: Connect with A0 pin of Arduino

Example: Read Analog Voltage

```
// select the input pin for the potentiometer
int sensorPin = A0;
// variable to store the value coming from the sensor
int digitalValue = 0;
float analogVoltage = 0.00;
 void setup() {
        Serial.begin(9600);
}
 void loop() {
       // read the value from the analog channel
       digitalValue = analogRead(sensorPin);
       Serial.print("digital value = ");
        //print digital value on serial monitor
       Serial.print(digitalValue);
       //convert digital value to analog voltage
       analogVoltage = (digitalValue * 5.00)/1023.00;
       Serial.print("  analog voltage = ");
       Serial.println(analogVoltage);
       delay(1000);
}
```



COM4 (Arduino/Genuino Uno)

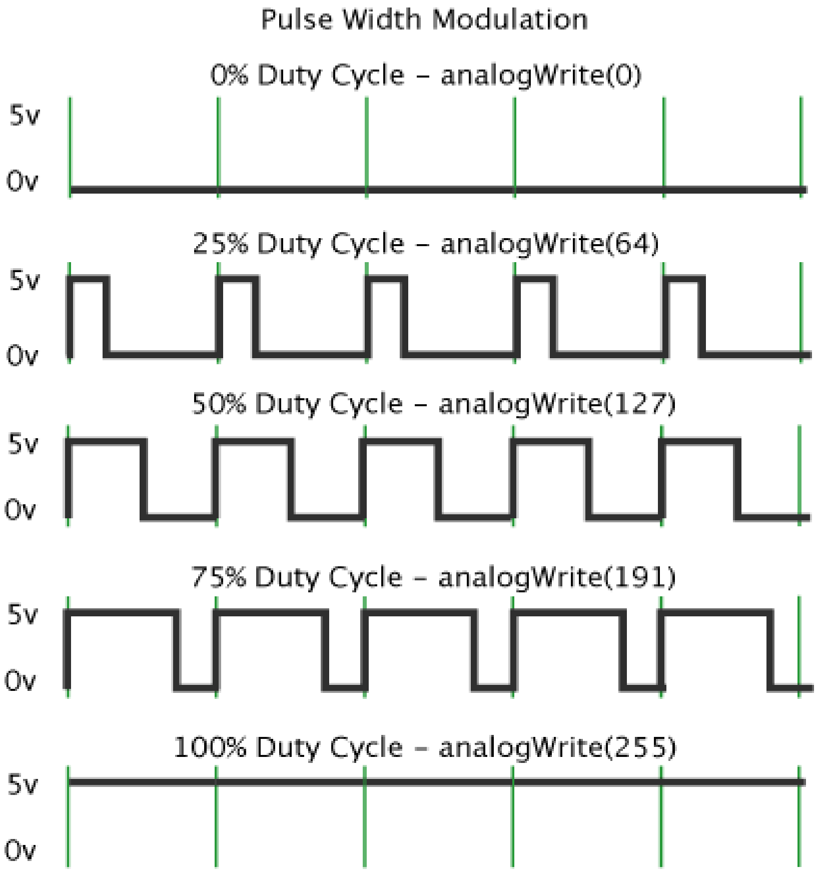
```
digital value = 0  analog voltage = 0.00
digital value = 0  analog voltage = 0.00
digital value = 30  analog voltage = 0.15
digital value = 66  analog voltage = 0.32
digital value = 171  analog voltage = 0.84
digital value = 275  analog voltage = 1.34
digital value = 331  analog voltage = 1.62
digital value = 400  analog voltage = 1.96
digital value = 459  analog voltage = 2.24
digital value = 475  analog voltage = 2.32
digital value = 482  analog voltage = 2.36
digital value = 502  analog voltage = 2.45
digital value = 517  analog voltage = 2.53
digital value = 543  analog voltage = 2.65
digital value = 588  analog voltage = 2.87
digital value = 595  analog voltage = 2.91
digital value = 598  analog voltage = 2.92
digital value = 736  analog voltage = 3.60
digital value = 939  analog voltage = 4.59
digital value = 974  analog voltage = 4.76
digital value = 998  analog voltage = 4.88
digital value = 1014  analog voltage = 4.96
digital value = 1019  analog voltage = 4.98
digital value = 1022  analog voltage = 5.00
```

Write Analog Value

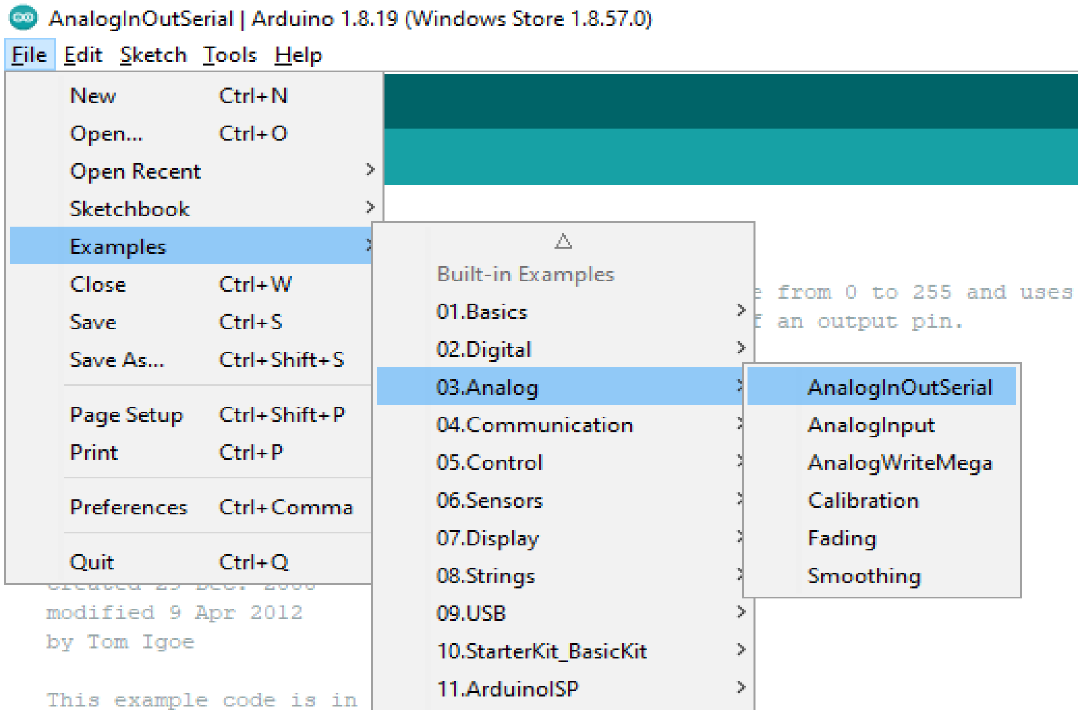
- Digital control is used to create a square wave, a signal switched between ON and OFF.

- This on-off pattern can simulate voltages in between Vcc and GND.
  - ✓ by changing the portion of the time the signal spends ON versus the time that the signal spends OFF

- The analogWrite(value) is on a scale of 0 – 255.
  - ✓ Zero value means 0% duty cycle, 255 value means 100% duty cycle.


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

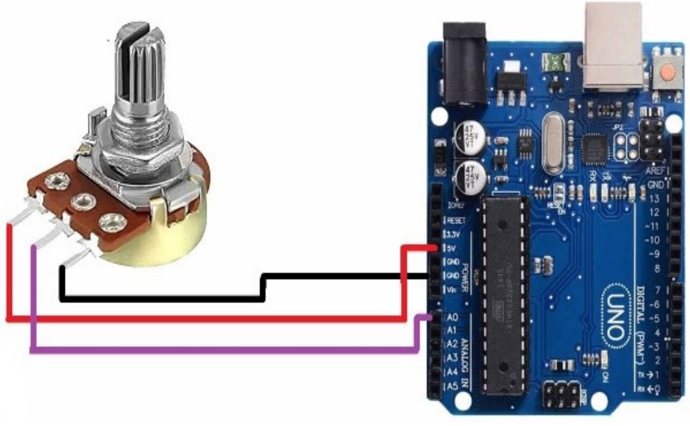
Example: Write Analog Value



```
e from 0 to 255 and uses
f an output pin.

modified 9 Apr 2012
by Tom Igoe

This example code is in
```



Pin 1 & 3 of Potentiometer: connect them to Vcc and GND of Arduino

Pin 2 of Potentiometer: Connect with A0 pin of Arduino

One **LED** connected with digital pin 9 and grounded through 220 ohm or 1 Kohm resistor

OUTPUT: LED Dimming by Potentiometer

Lessons Learned

- ✓ What is Arduino Programming
- ✓ Syntax of Arduino Programming
- ✓ Supporting variable, structures, operators
- ✓ In-Built Arduino Function Library
- ✓ Programming example - LED blink
- ✓ Program and Demo on binary counter in LED
- ✓ Analog Read and Write

Thanks!