Arduino

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

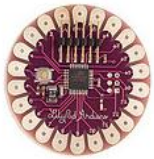
- Arduino Hardware
- Blink an LED
- Digital Input
- Analog Input
- Analog Output
- Serial Communication
- Taking to Processing

Why Arduino?

- **Physical Computing**
  - **uses electronics**
  - **to prototype new materials**
  - **for designers and artists.**
- **Tinkering**
- **Patching**
- **Community**
  - **Blog, Forum, Playground (wiki)**

Hardware

Arduino Uno



Arduino LilyPad

Arduino Ethernet

Arduino Nano

Arduino BT

Arduino Mini

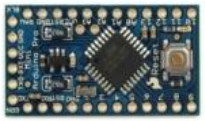
Arduino Mega 2560

Arduino Fio

Arduino BT

Arduino Mini

USB/Serial Light Adapter



Arduino Pro Mini

Arduino Mega ADK

Arduino Pro

USB/Serial Light Adapter

Arduino Pro Mini

UNO

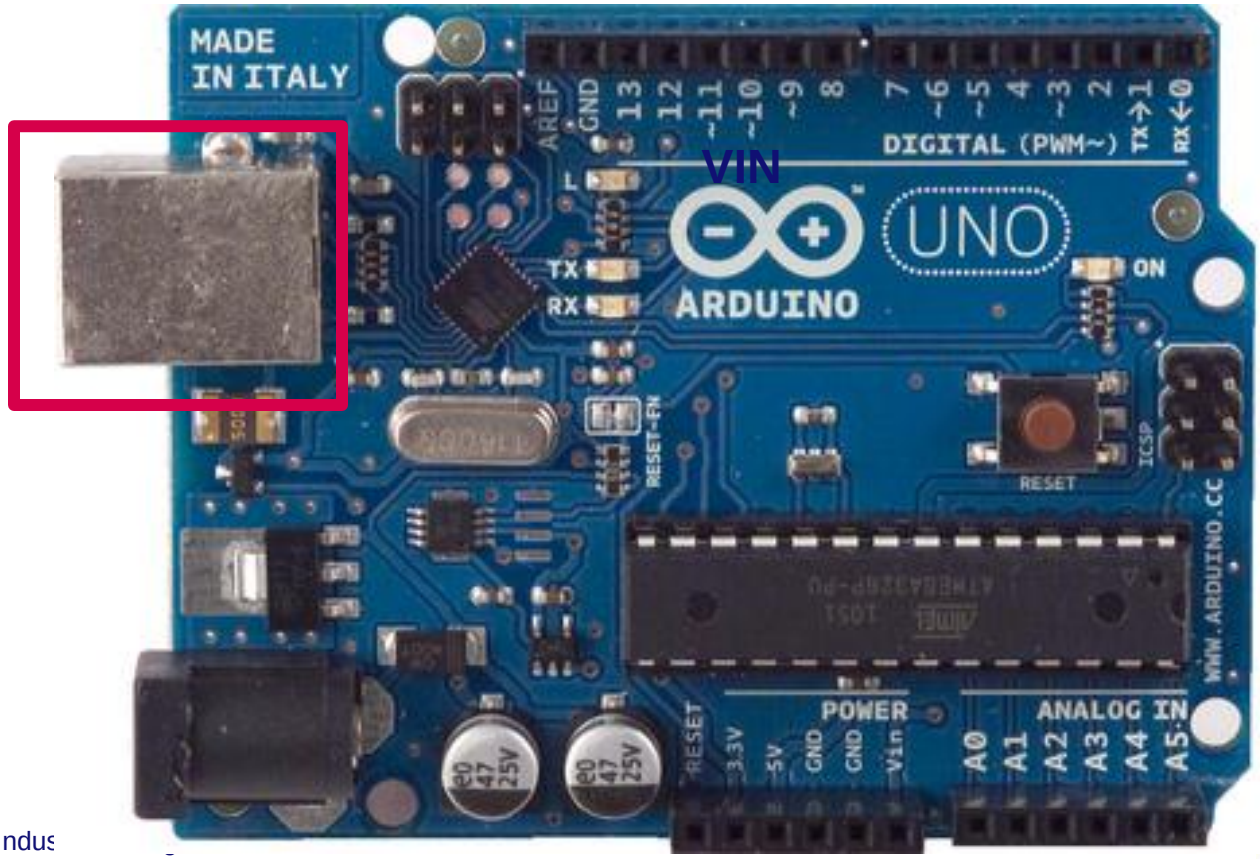

MADE
IN ITALY
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
UNO
ARDUINO
ON
RESET-EN
RESET
ICSP
WWW.ARDUINO.CC
POWER
ANALOG IN
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

UNO

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (VIN) (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 (ATmega328) of which 0.5 KB used by bootloader
SRAM (Static RAM)	2 KB (ATmega328)
EEPROM (Electrically erasable programmable ROM)	1 KB (ATmega328)
Clock Speed	16 MHz

UNO

- **Power: USB Power supply (5V)**



UNO

- **Power: external power supply (7V-12V)**

UNO

- **Power: VIN, input or supply, depending on external power source. (7-12V)**

UNO

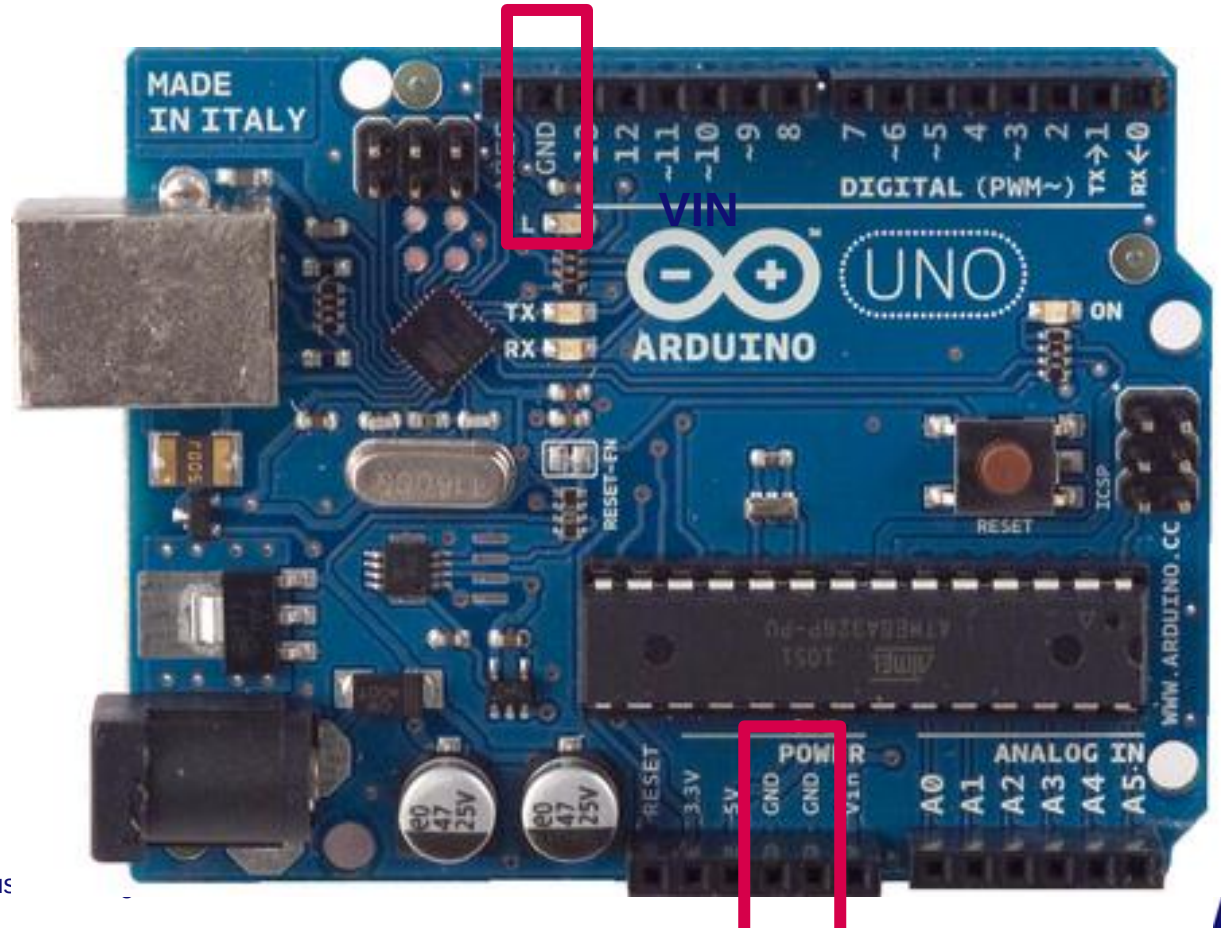
- **Power: 5V supply**

UNO

- **Power: 3.3V supply**

UNO

- **Power: GND pins**

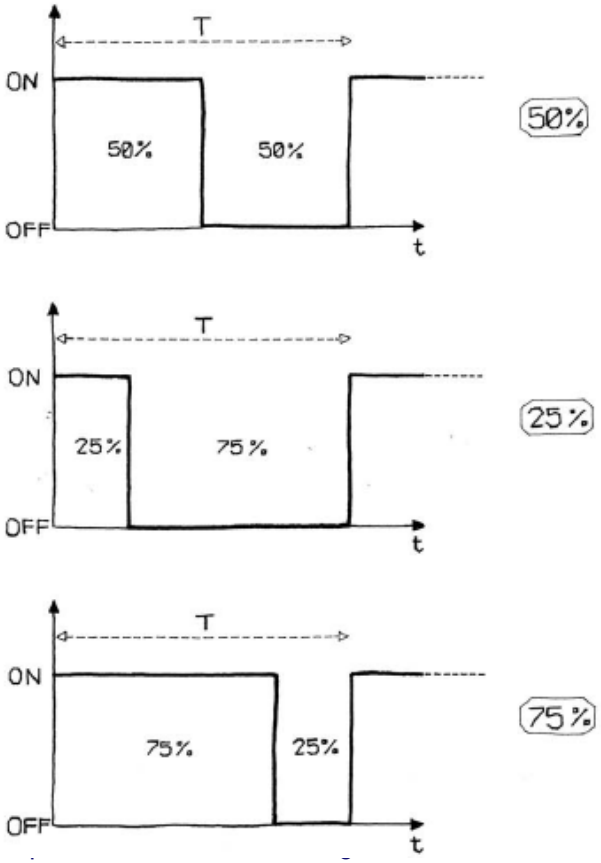


UNO

- **Digital I/O Pins 14 (of which 6 provide PWM output)**

UNO

- **Digital I/O Pins 14 (of which 6 provide PWM output)**
  - **PWM (Pulse-width modulation)**

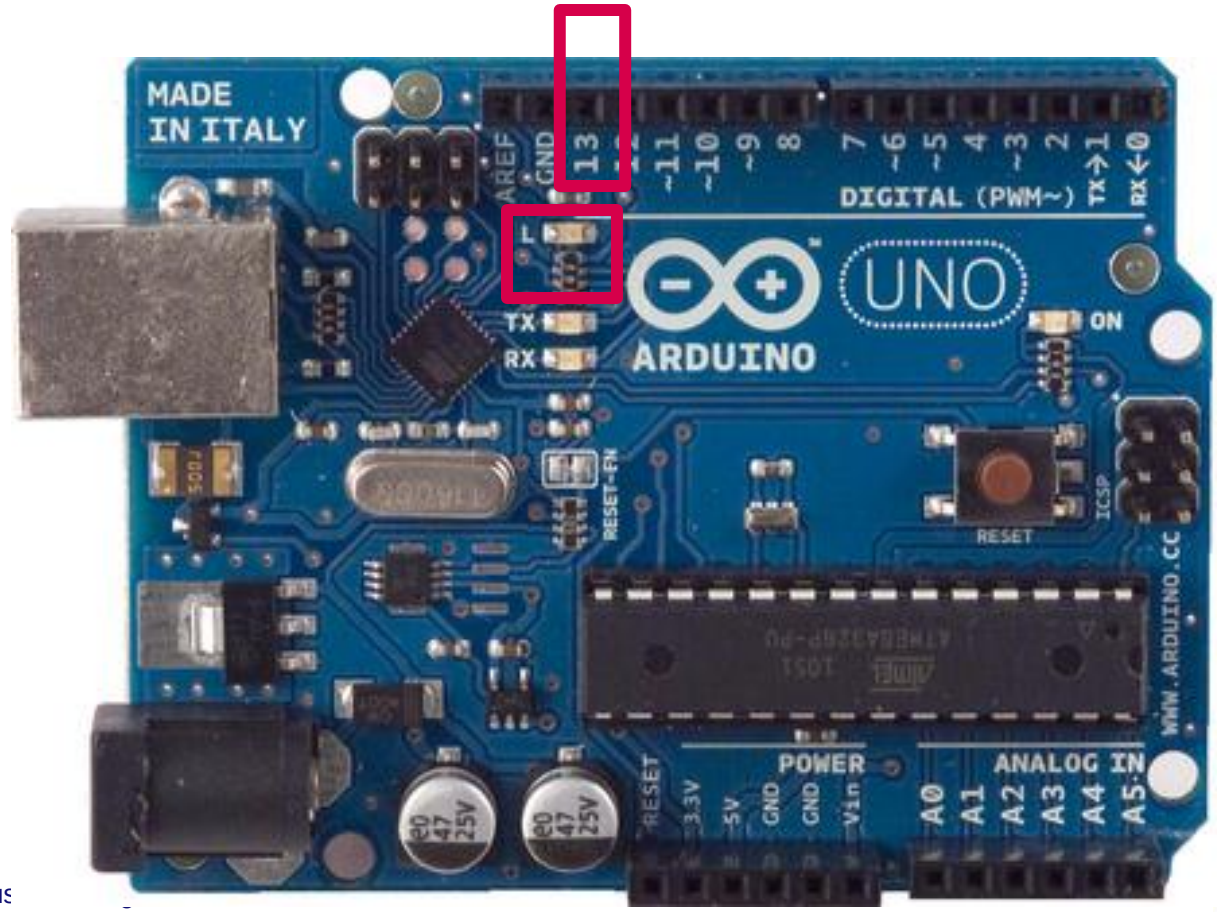


UNO

- **Serial: 0 (RX) and 1 (TX)**

UNO

- **LED: 13**

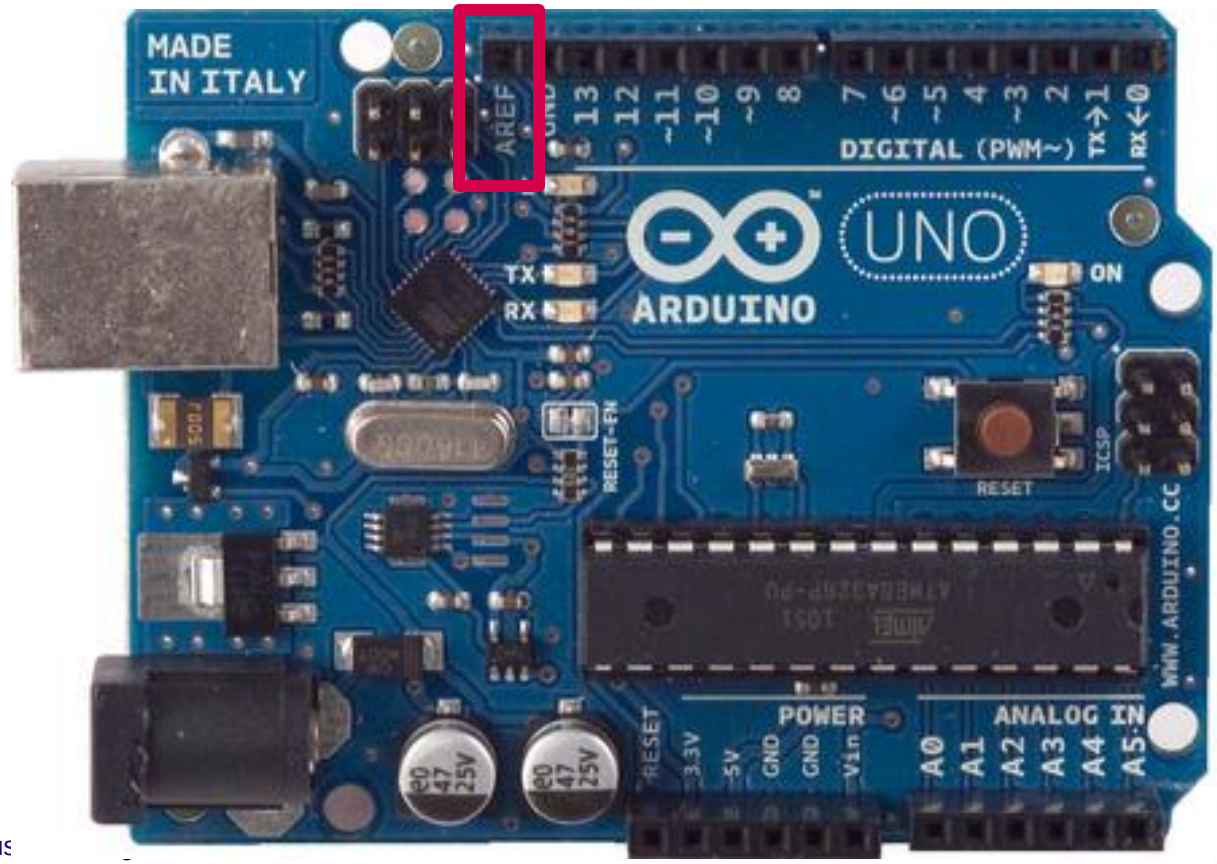

MADE
IN ITALY
AREF
GND
13
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
UNO
ARDUINO
ON
RESET-EN
RESET
ICSP
WWW.ARDUINO.CC
POWER
ANALOG IN
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

UNO

- **6 analog inputs, 10 bits of resolution (i.e. 1024 different values)**

UNO

- **AREF: Reference voltage for the analog inputs**

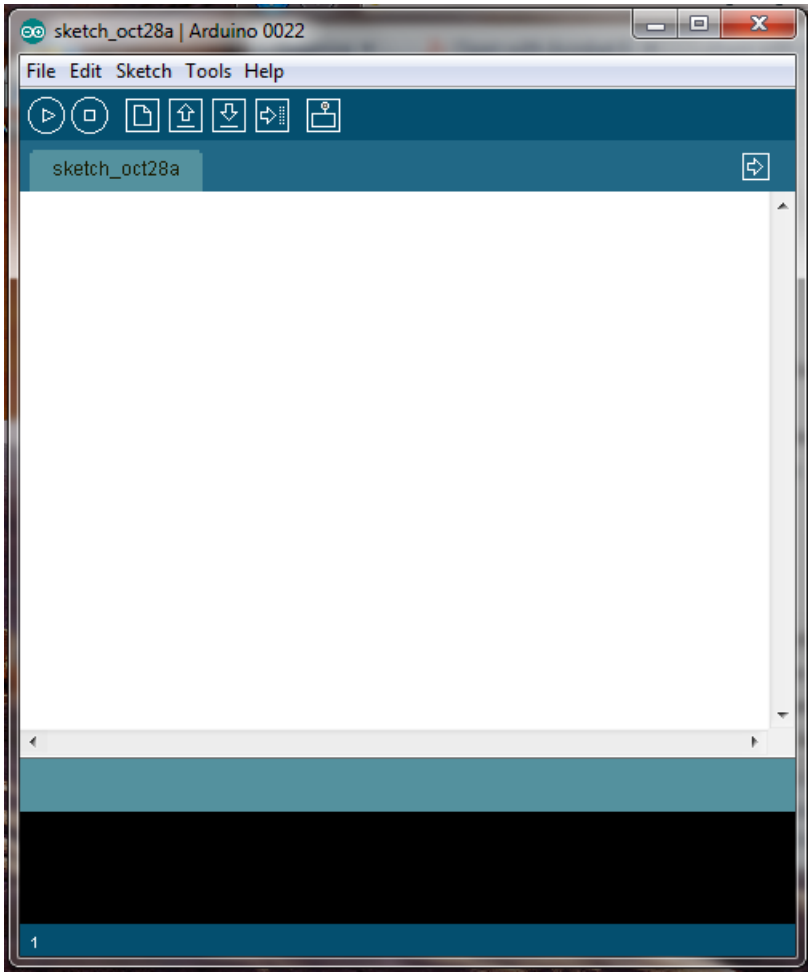


UNO

- **Reset. LOW to reset the microcontroller**

Software: IDE

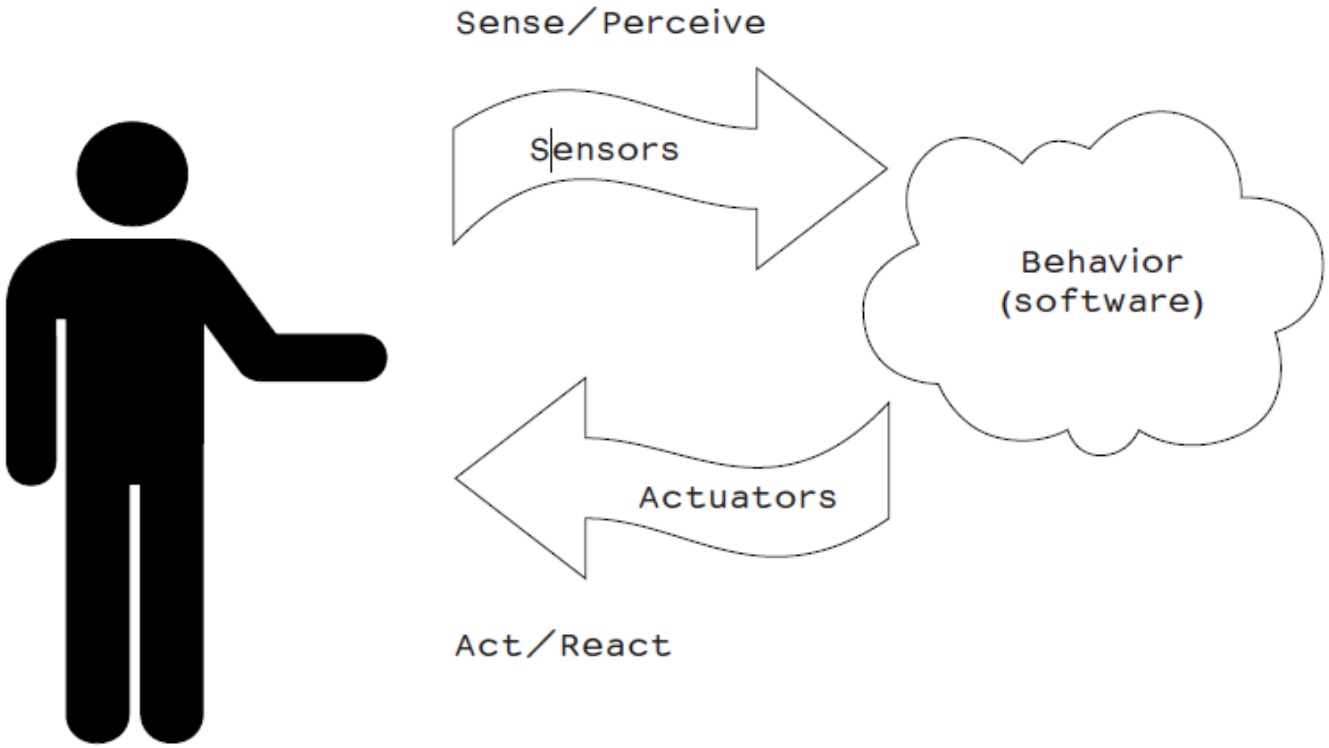
- http://arduino.cc/en/Main/Software



Driver Installation and Port Identification

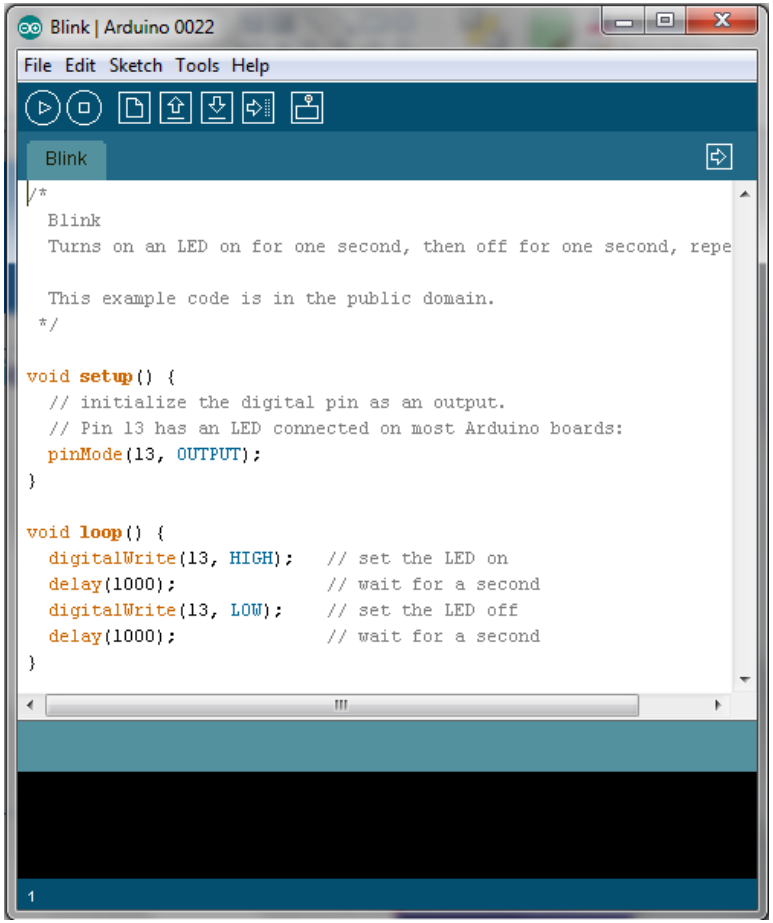
- **Refer to the instructions in**
  - **“Getting Started with Arduino”, page 23-26**

Really getting started

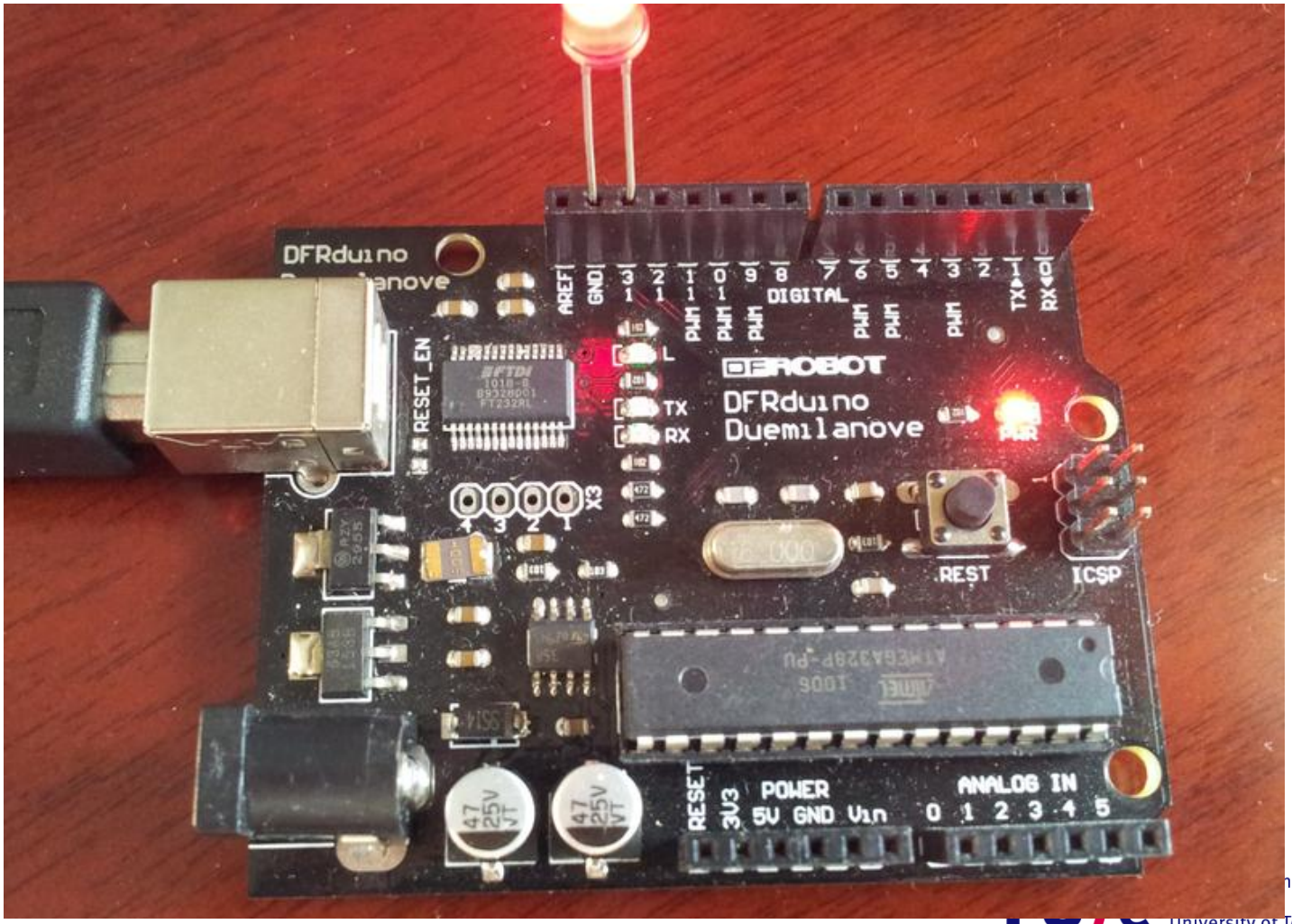


Blinking an LED

- **File>Examples>Basics>Blink**
  - **LED: light-emitting diode**



Blinking an LED



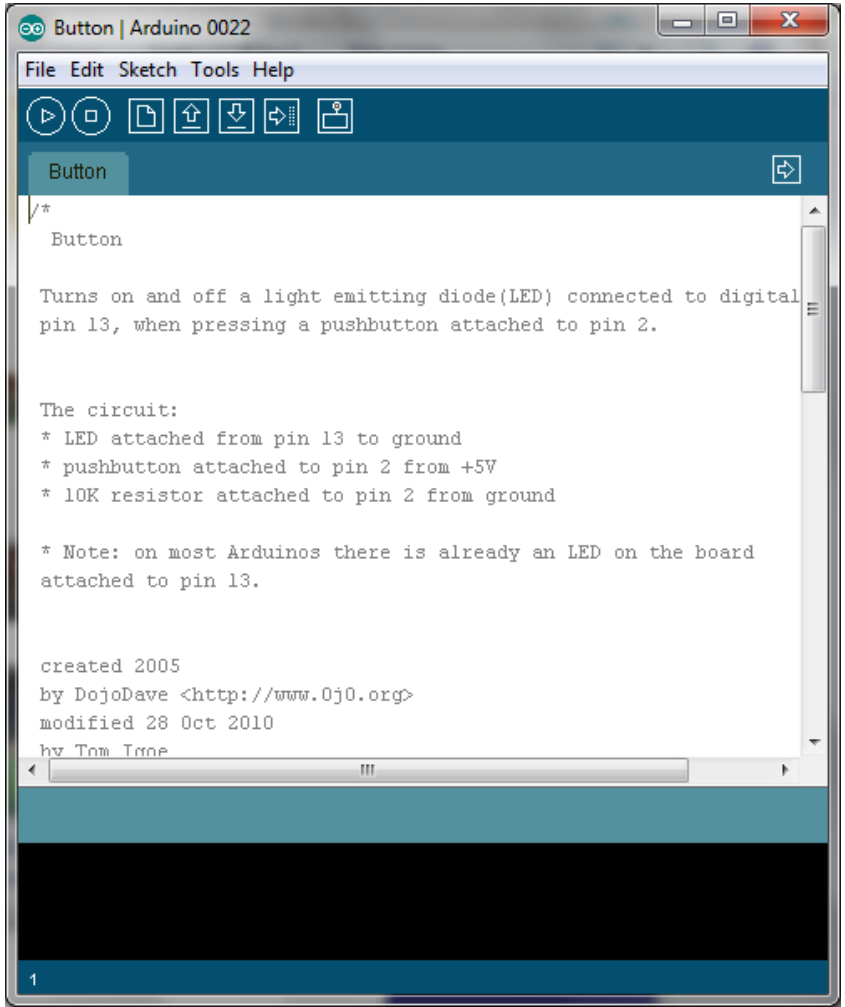
Blink an LED

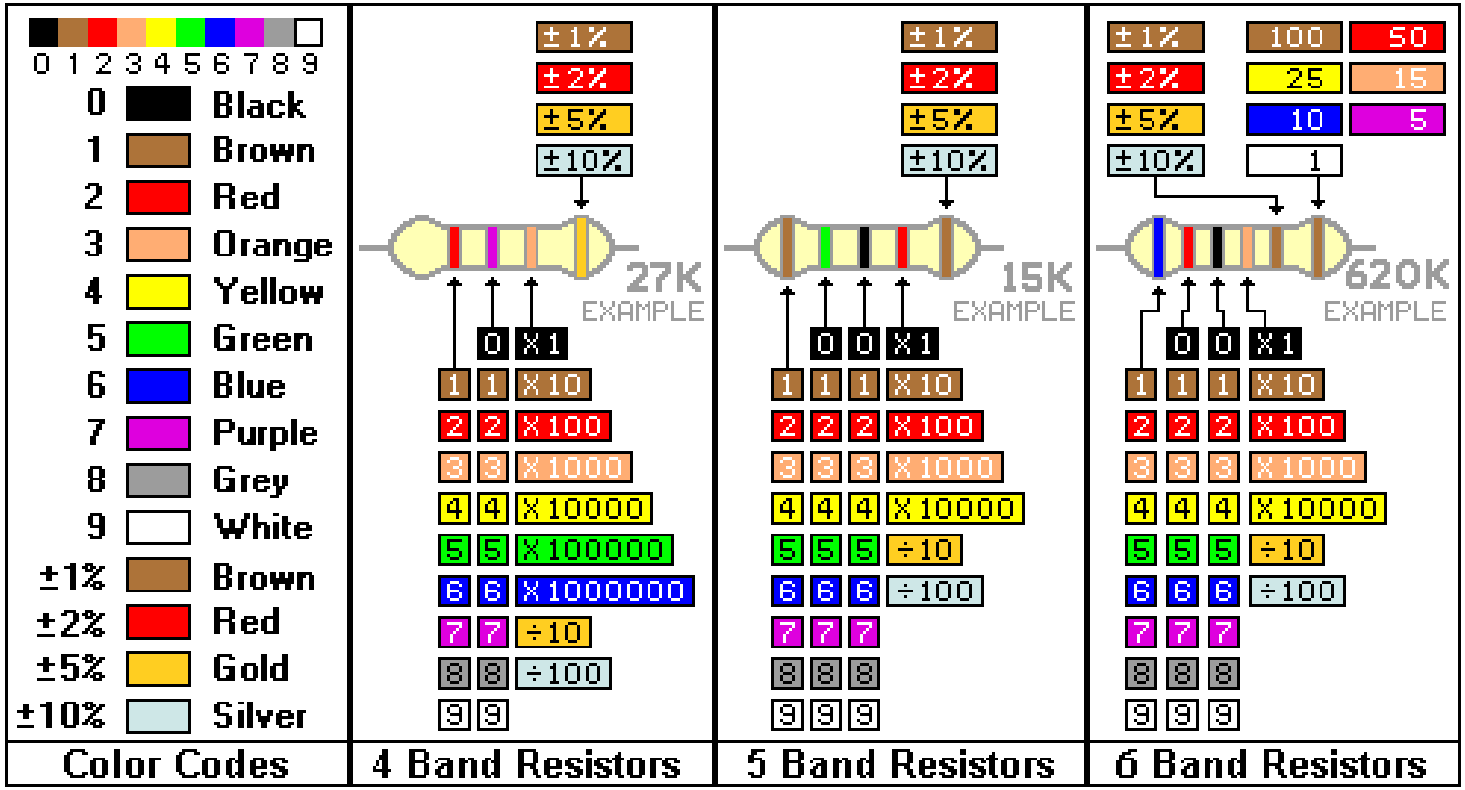
- **#define LED 13**

```
#define LED 13

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

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

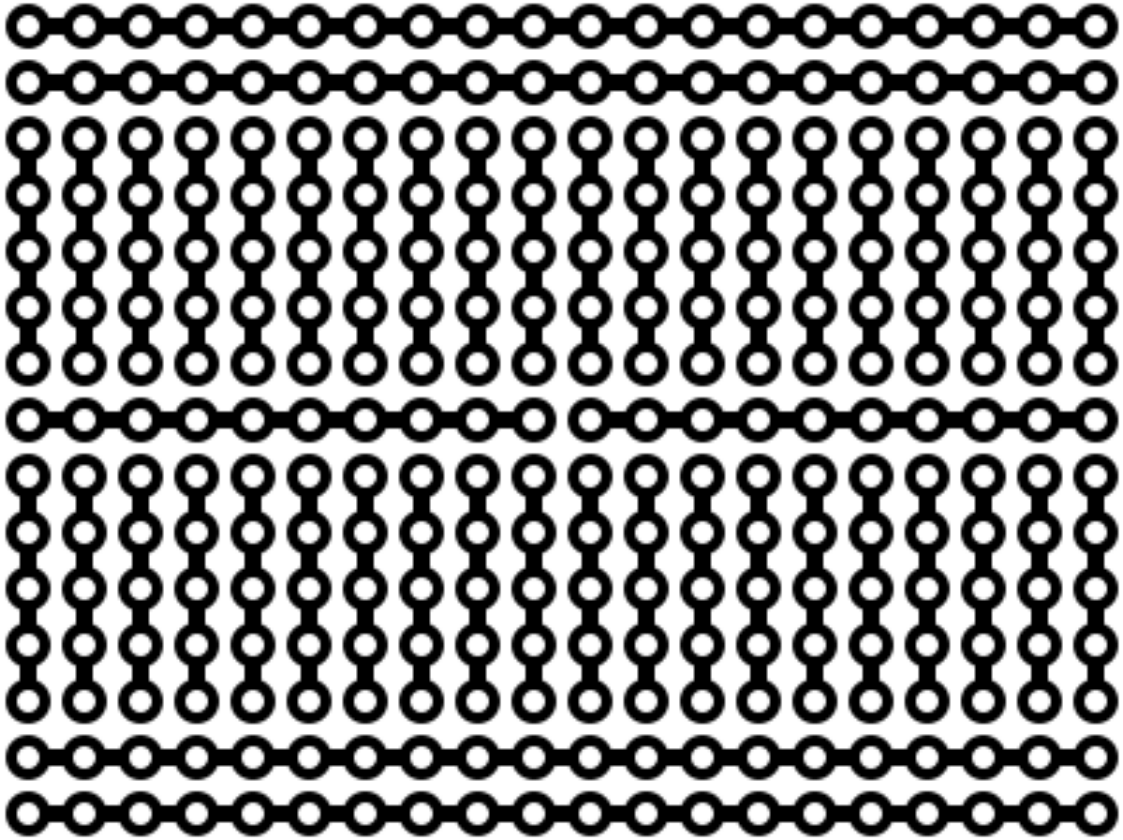
Button to control the LED

- **File>Examples>Digital>Button**

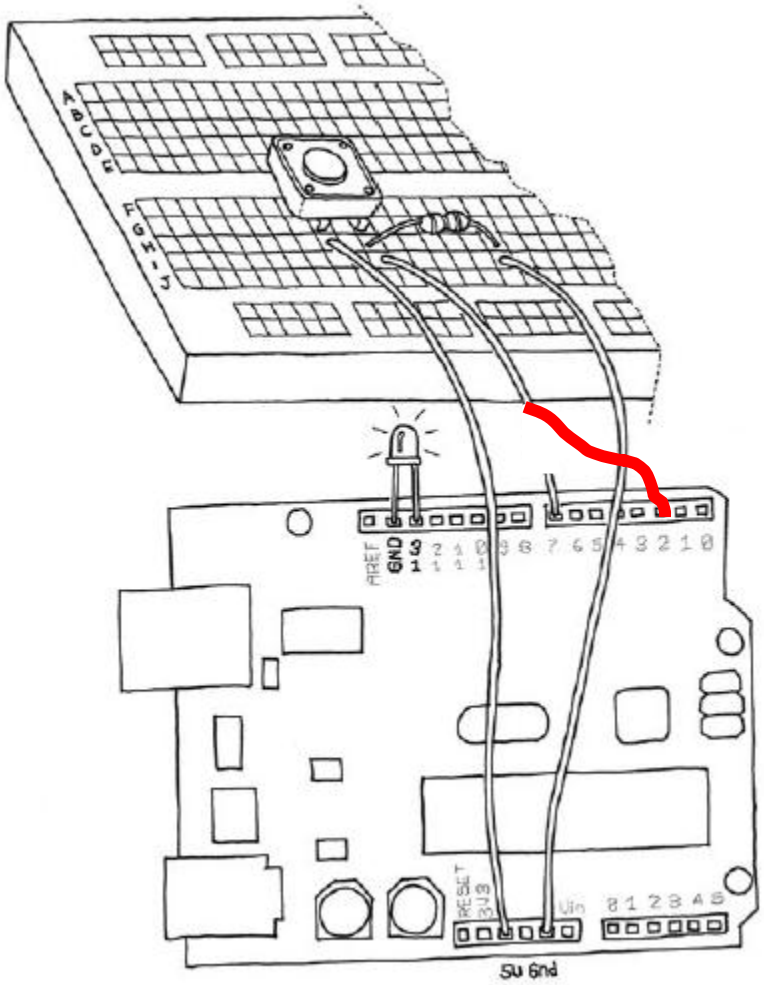



0 1 2 3 4 5 6 7 8 9
0 Black
1 Brown
2 Red
3 Orange
4 Yellow
5 Green
6 Blue
7 Purple
8 Grey
9 White
±1% Brown
±2% Red
±5% Gold
±10% Silver
Color Codes
±1%
±2%
±5%
±10%
27K
EXAMPLE
0 X 1
1 1 X 10
2 2 X 100
3 3 X 1000
4 4 X 10000
5 5 X 100000
6 6 X 1000000
7 7 ÷10
8 8 ÷100
9 9
4 Band Resistors
±1%
±2%
±5%
±10%
15K
EXAMPLE
0 0 X 1
1 1 1 X 10
2 2 2 X 100
3 3 3 X 1000
4 4 4 X 10000
5 5 5 ÷10
6 6 6 ÷100
7 7 7
8 8 8
9 9 9
5 Band Resistors
±1%
±2%
±5%
±10%
100
50
25
15
10
5
1
620K
EXAMPLE
0 0 X 1
1 1 1 X 10
2 2 2 X 100
3 3 3 X 1000
4 4 4 X 10000
5 5 5 ÷10
6 6 6 ÷100
7 7 7
8 8 8
9 9 9
6 Band Resistors

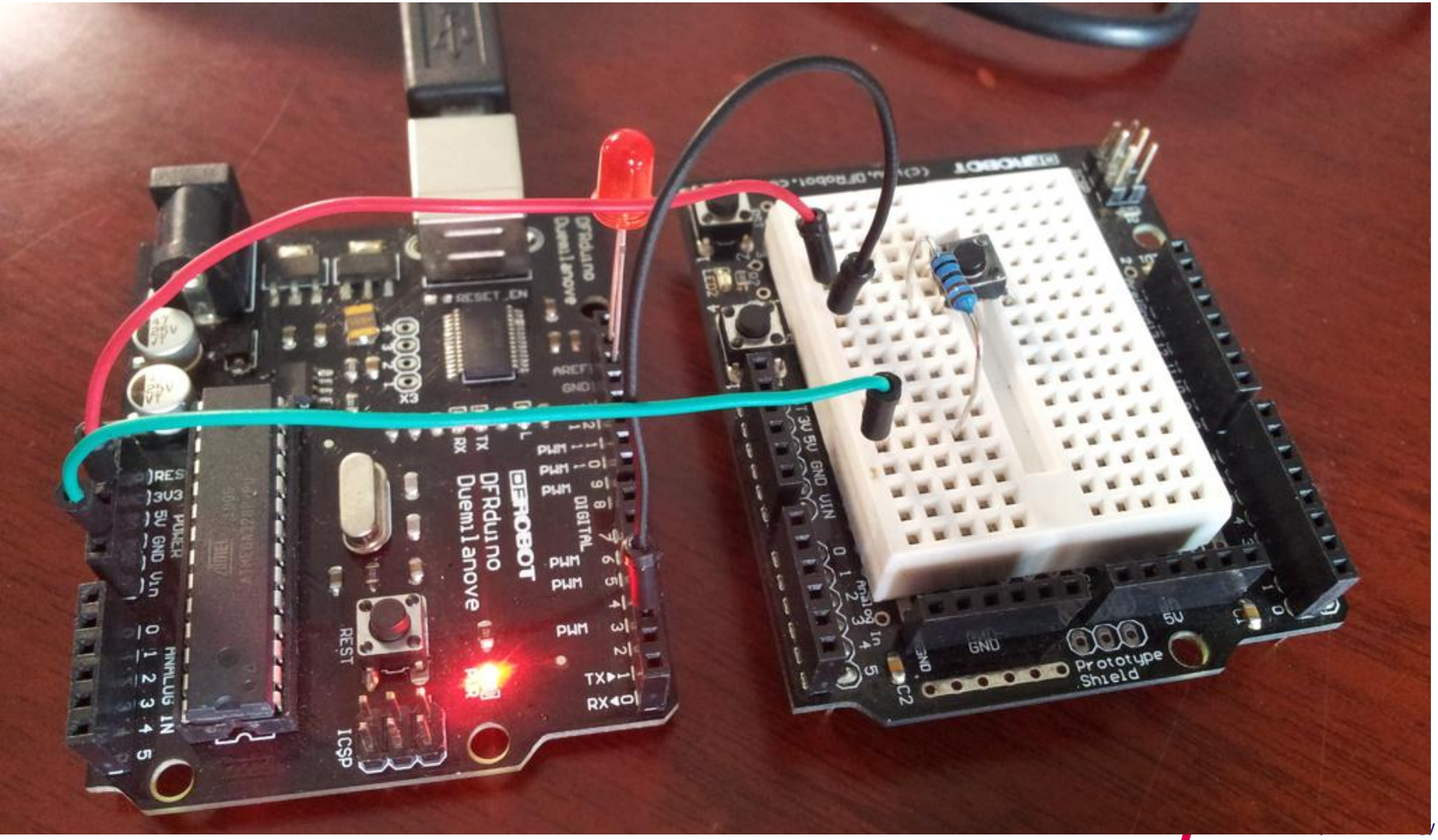
Button to control the LED



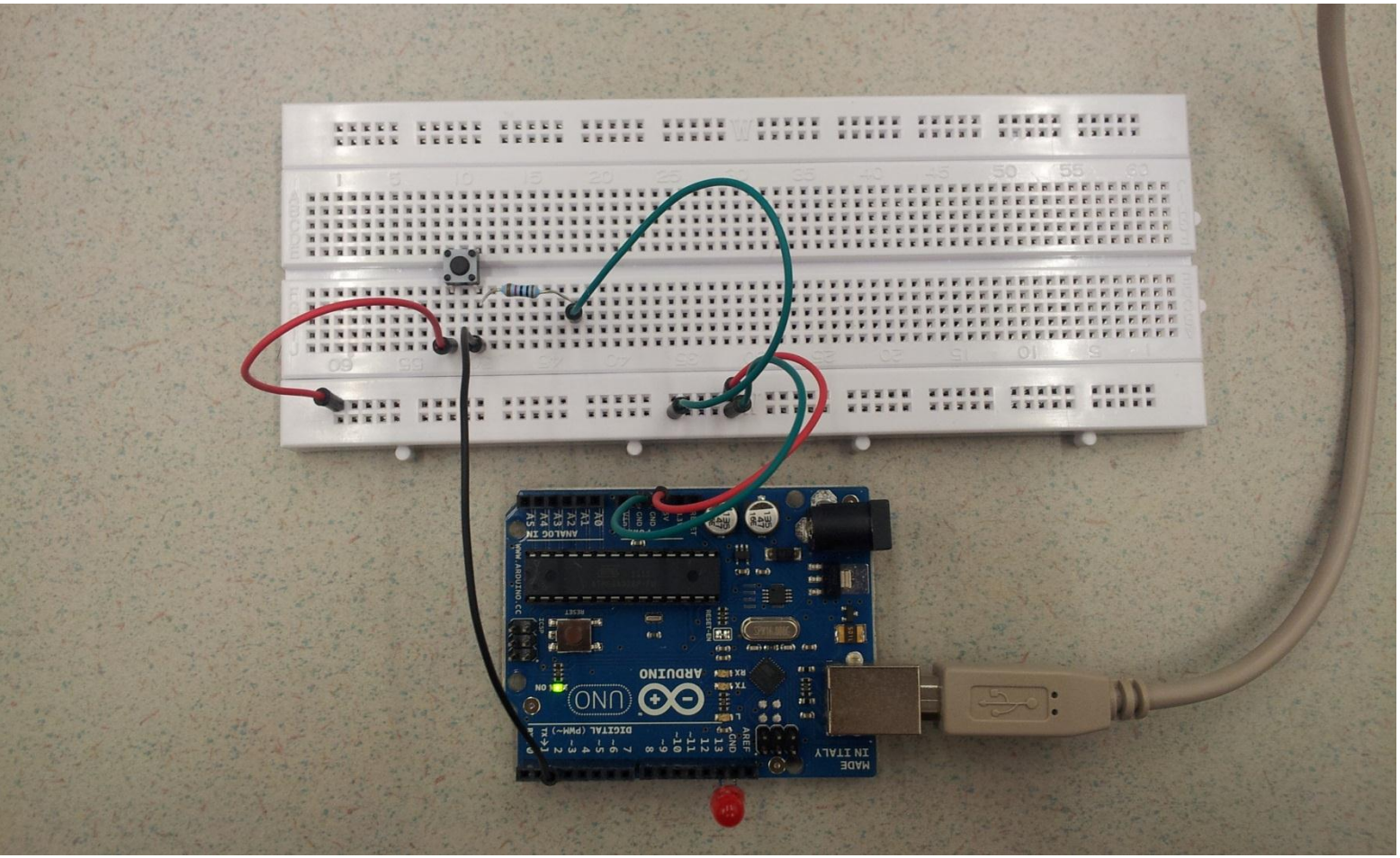
Button to control the LED



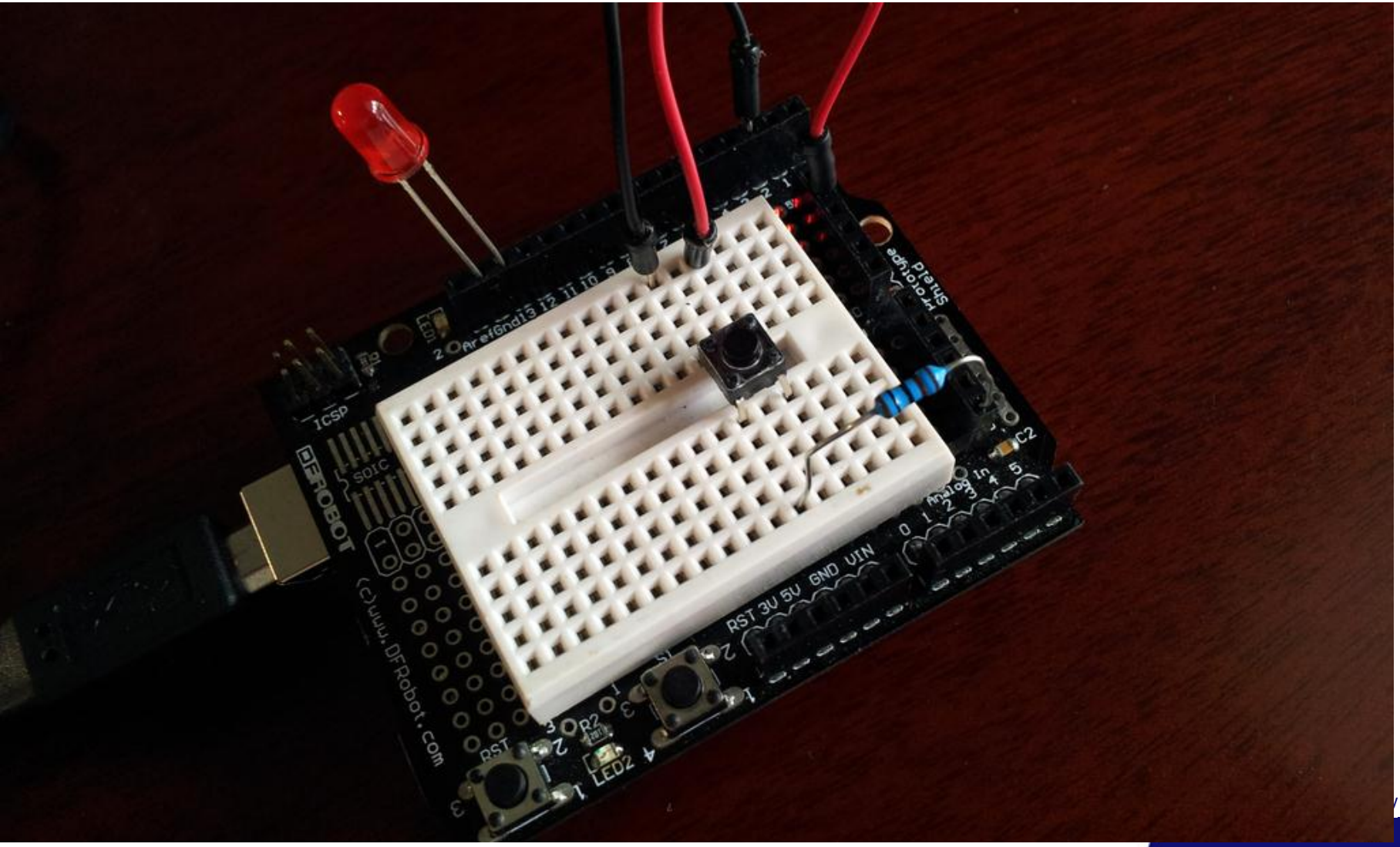
Button to control the LED



Button to control the LED



Button to control the LED



Button to control the LED

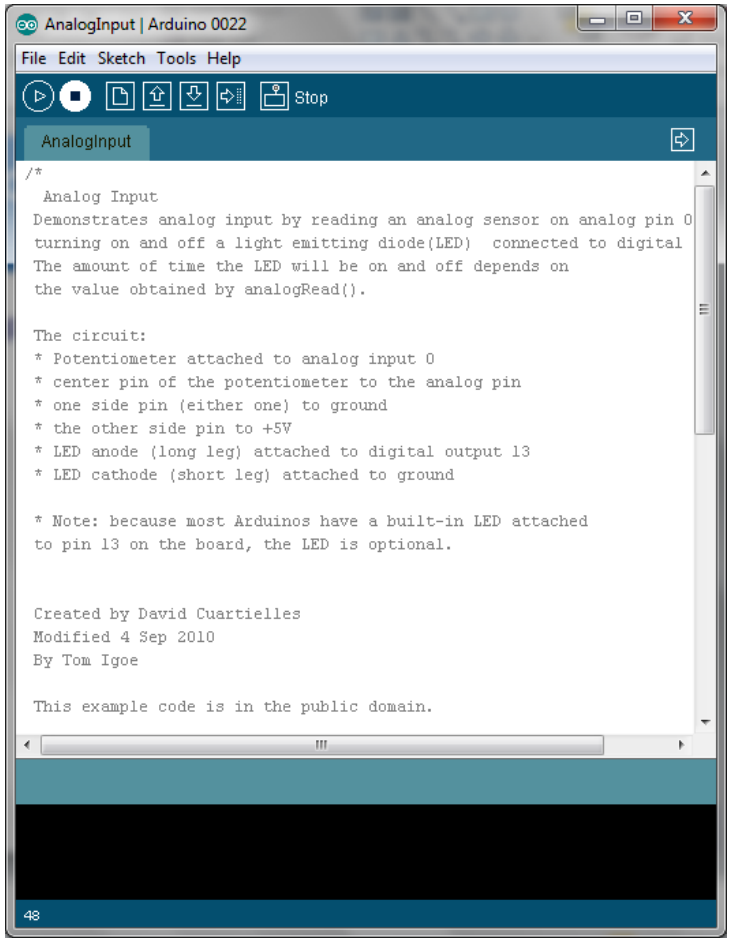
```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// 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);
}
```

Button to control the LED

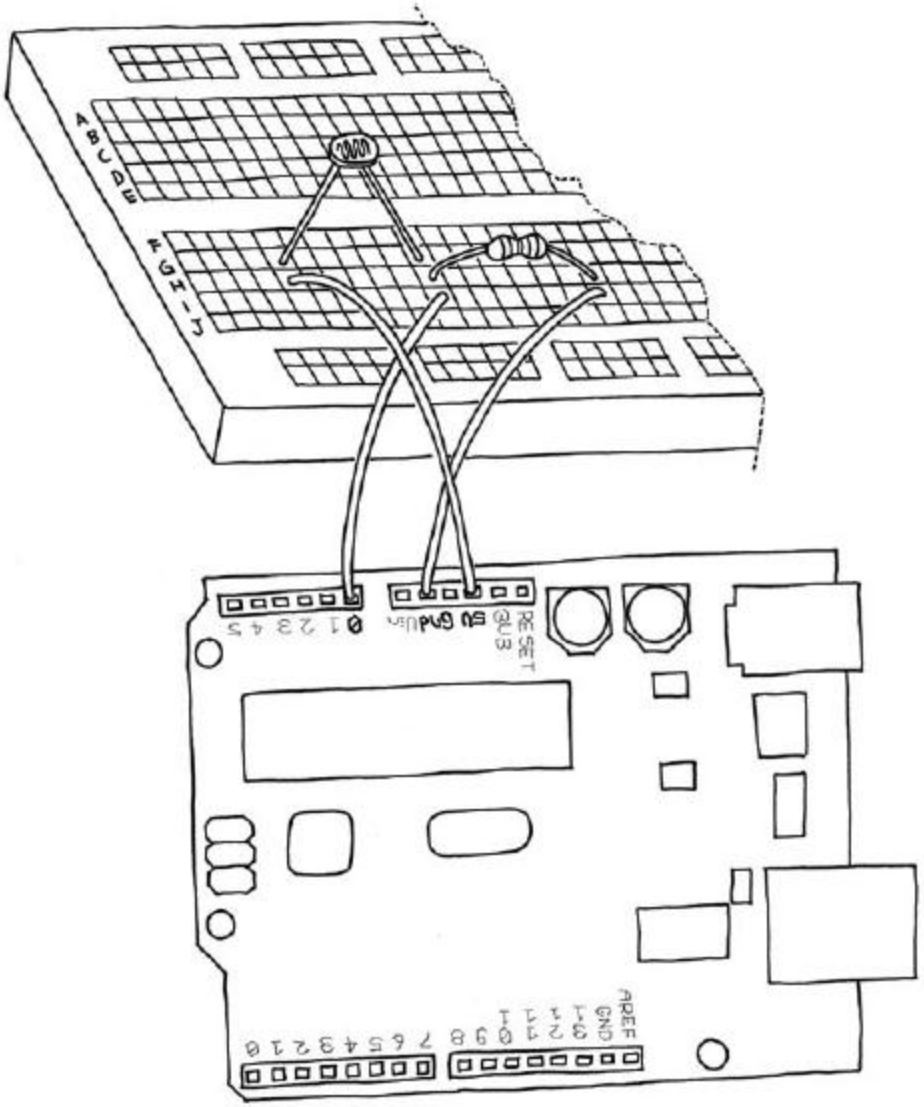
```
void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // 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);
  }
}
```

Analog I/O

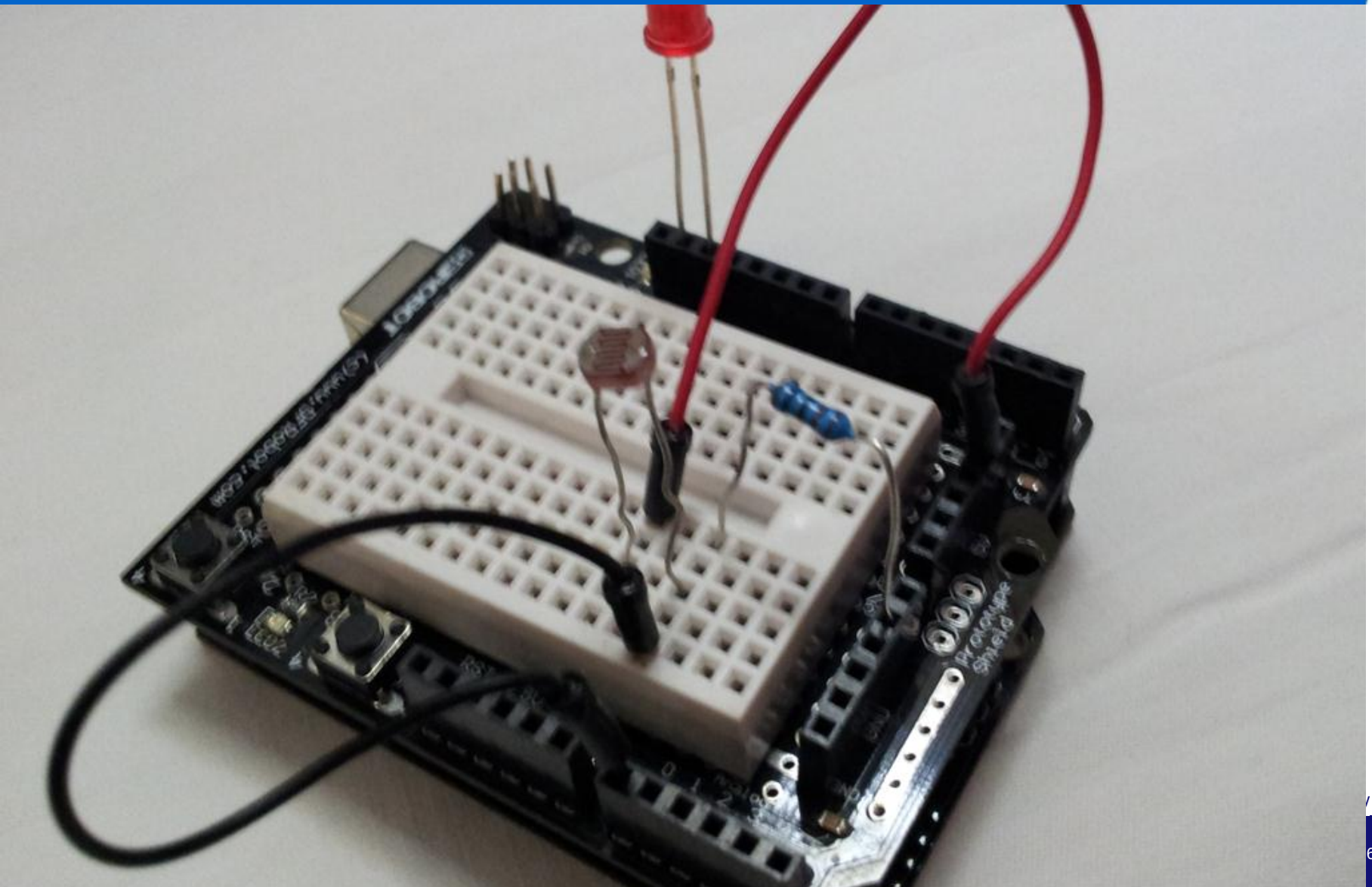
- **File>Examples>Analog>AnalogInput**
  - **Instead of a potentiometer, we use a light sensor**



Analog I/O


AREF
GND
13
12
11
10
9
8
7
6
5
4
3
2
1
0
RESET
GND
3V3
5V
GND
VIN
0
1
2
3
4
5

Analog I/O



Analog I/O

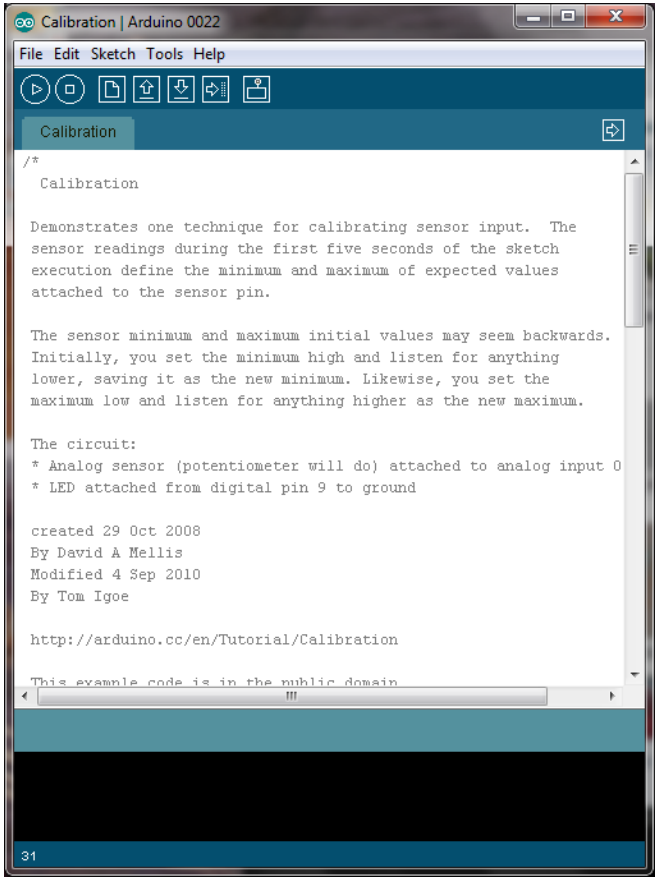
```
int sensorPin = A0;    // select the input pin for the potentiometer
int ledPin = 13;      // select the pin for the LED
int sensorValue = 0;  // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  digitalWrite(ledPin, HIGH);
  // stop the program for <sensorValue> milliseconds:
  delay(sensorValue);
  // turn the ledPin off:
  digitalWrite(ledPin, LOW);
  // stop the program for for <sensorValue> milliseconds:
  delay(sensorValue);
}
```

Analog I/O

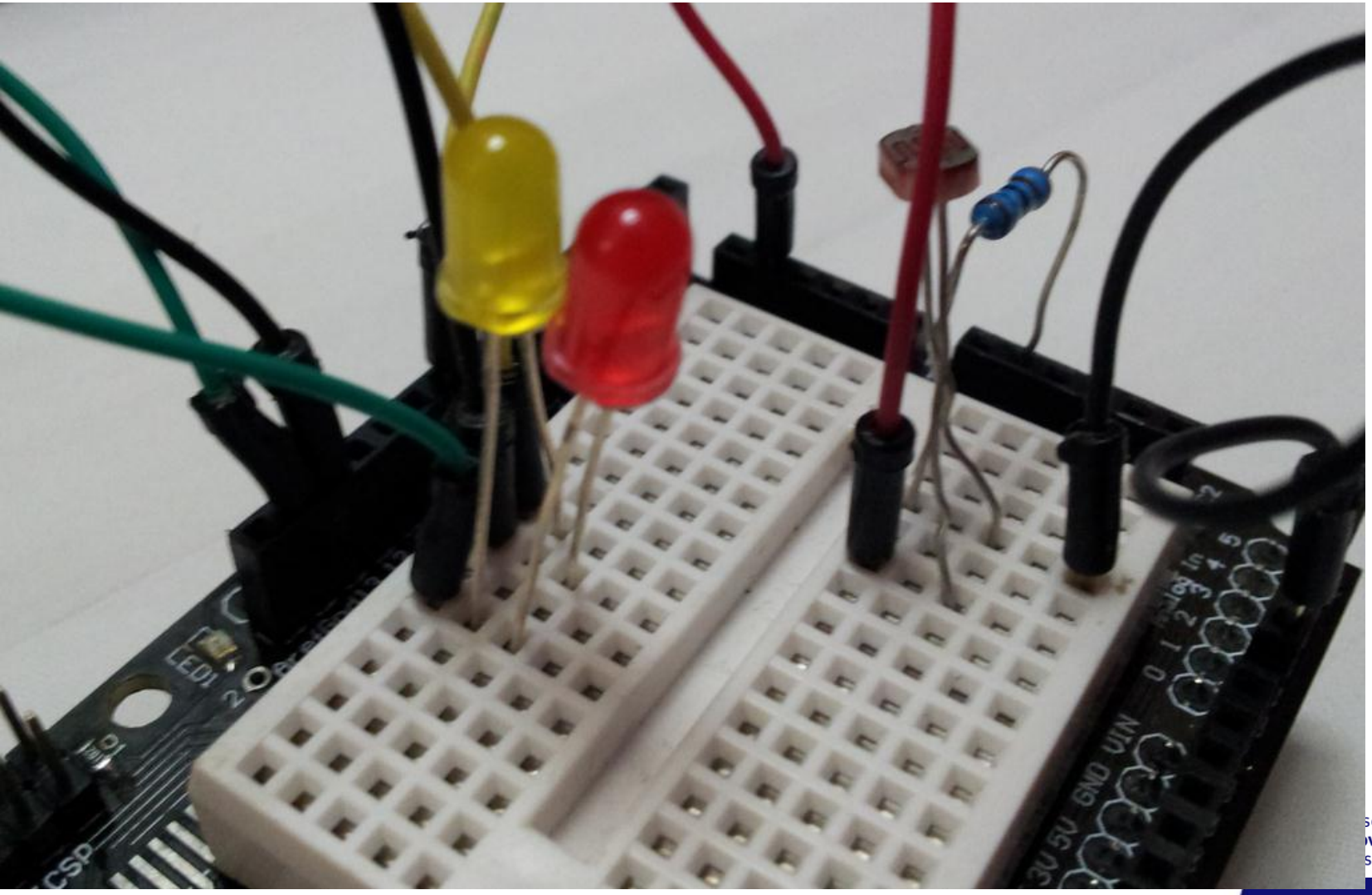
```
int sensorPin = A0;    // select the input pin
int ledPin = 11;      // select the pin for the LED
int sensorValue = 0;  // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  analogWrite(ledPin, sensorValue/4);
}
```

Analog Input Calibration

- **File>Examples>Analog>Calibration**
  - **Connect an LED to pin 9, and the other to pin 13.**



Analog Input Calibration



Analog Input Calibration

```
// These constants won't change:
const int sensorPin = A0;    // pin that the sensor is attached to
const int ledPin = 9;        // pin that the LED is attached to

// variables:
int sensorValue = 0;         // the sensor value
int sensorMin = 1023;        // minimum sensor value
int sensorMax = 0;           // maximum sensor value
```

Analog Input Calibration

```
void setup() {
  // turn on LED to signal the start of the calibration period:
  pinMode(13, OUTPUT);
  digitalWrite(13, HIGH);

  // calibrate during the first five seconds
  while (millis() < 5000) {
    sensorValue = analogRead(sensorPin);

    // record the maximum sensor value
    if (sensorValue > sensorMax) {
      sensorMax = sensorValue;
    }

    // record the minimum sensor value
    if (sensorValue < sensorMin) {
      sensorMin = sensorValue;
    }
  }

  // signal the end of the calibration period
  digitalWrite(13, LOW);
}
```

Analog Input Calibration

```
void loop() {
  // read the sensor:
  sensorValue = analogRead(sensorPin);

  // apply the calibration to the sensor reading
  sensorValue = map(sensorValue, sensorMin, sensorMax, 0, 255);

  // in case the sensor value is outside the range seen during calibration
  sensorValue = constrain(sensorValue, 0, 255);

  // fade the LED using the calibrated value:
  analogWrite(ledPin, sensorValue);
}
```

Serial Communication

- **We use the same example for Calibration**

Serial Communication

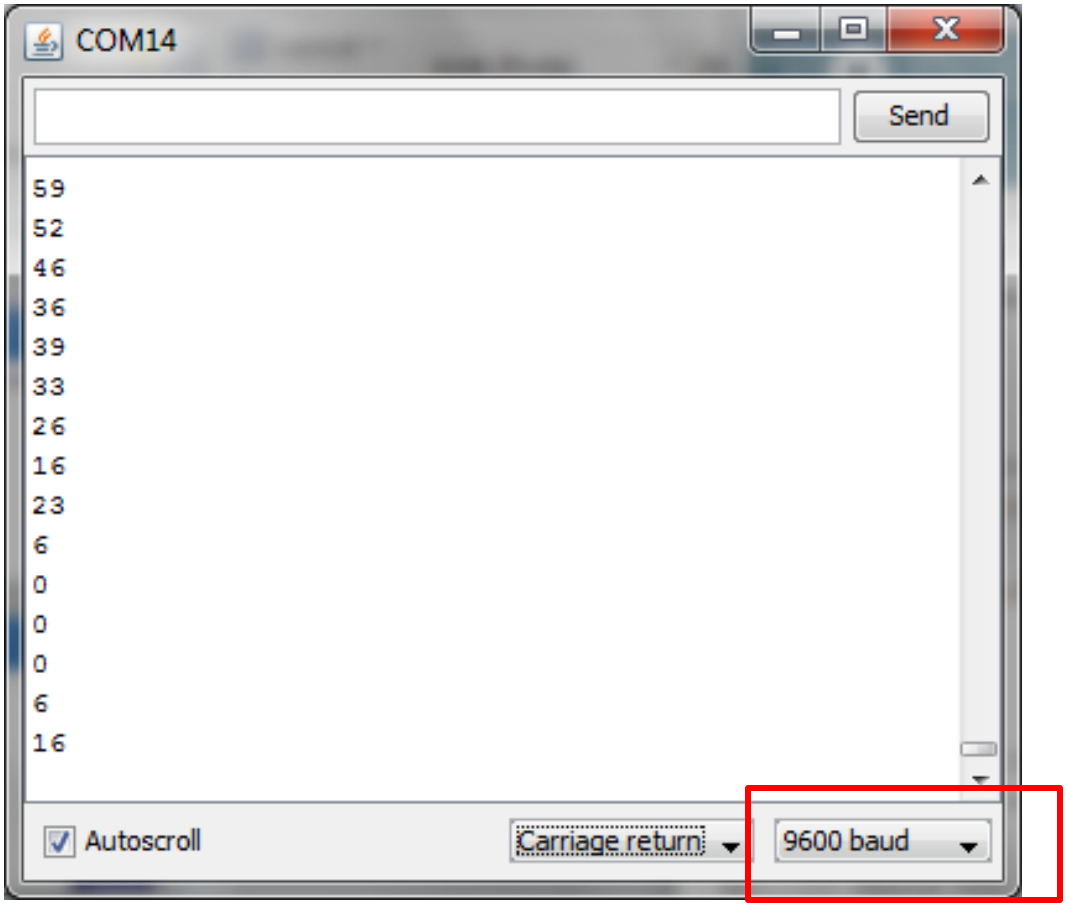
```
void setup() {
  // turn on LED to signal the start of the calibration period:

  pinMode(13, OUTPUT);
  digitalWrite(13, HIGH);

  // calibrate during the first five seconds
  while (millis() < 5000) {
    sensorValue = analogRead(sensorPin);

    // record the maximum sensor value
    if (sensorValue > sensorMax) {
      sensorMax = sensorValue;
    }

    // record the minimum sensor value
    if (sensorValue < sensorMin) {
      sensorMin = sensorValue;
    }
  }

  // signal the end of the calibration period
  digitalWrite(13, LOW);

  Serial.begin(9600);
}
```

Serial Communication

```
void loop() {
  // read the sensor:
  sensorValue = analogRead(sensorPin);

  // apply the calibration to the sensor reading
  sensorValue = map(sensorValue, sensorMin, sensorMax, 0, 255);

  // in case the sensor value is outside the range seen during calibration
  sensorValue = constrain(sensorValue, 0, 255);

  // fade the LED using the calibrated value:
  analogWrite(ledPin, sensorValue);

  Serial.println(sensorValue);
  delay(100);
}
```

Serial Communication

- **Try it out.**



Serial Communication

- **Now change it a bit. Try again the Serial Monitor**

```
void loop() {
  // read the sensor:
  sensorValue = analogRead(sensorPin);

  // apply the calibration to the sensor reading
  sensorValue = map(sensorValue, sensorMin, sensorMax, 0, 255);

  // in case the sensor value is outside the range seen during calibration
  sensorValue = constrain(sensorValue, 0, 255);

  // fade the LED using the calibrated value:
  analogWrite(ledPin, sensorValue);

  Serial.write(sensorValue);
  delay(100);
}
```

Serial Communication

- **Now try to receive the sensor input from Processing**
- **In Processing**
  - **File>Examples>Books>Chapter 11>Ex_11_07**

Serial Communication

```
import processing.serial.*;

Serial port;  // Create object from Serial class
float val;    // Data received from the serial port

void setup() {
  size(440, 220);
  // IMPORTANT NOTE:
  // The first serial port retrieved by Serial.list()
  // should be your Arduino. If not, uncomment the next
  // line by deleting the // before it. Run the sketch
  // again to see a list of serial ports. Then, change
  // the 0 in between [ and ] to the number of the port
  // that your Arduino is connected to.
  println(Serial.list());
  String arduinoPort = Serial.list()[1];
  port = new Serial(this, arduinoPort, 9600);
}
```

Serial Communication

```
void draw() {
  background(255);
  if (port.available() > 0) { // If data is available,
    val = port.read();        // read it and store it in val
    val = map(val, 0, 255, 0, height);  // Convert the value
  }
  rect(40, val-10, 360, 20);
}
```

Serial Communication

- **Now use the same hardware, try out**
- **In Processing**
  - **File>Examples>Books>Chapter 11>Ex_11_08**

- **That was Arduino.**