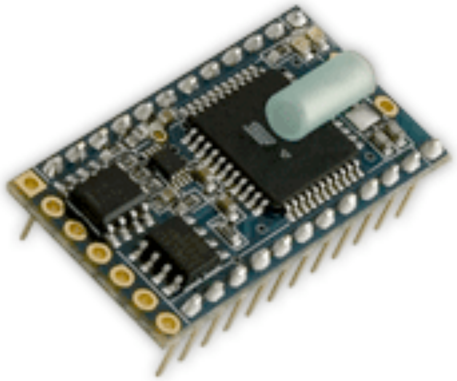
Microcontroller & Arduino

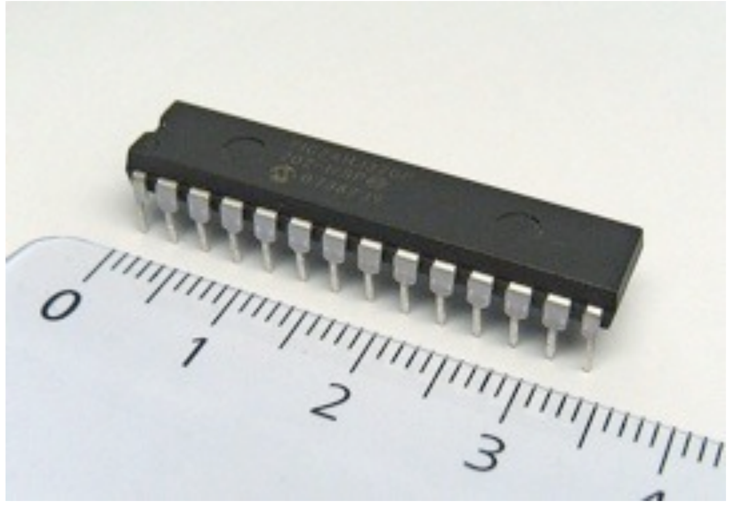
INTRODUCTION

basic stamp



bx 24

basic atom

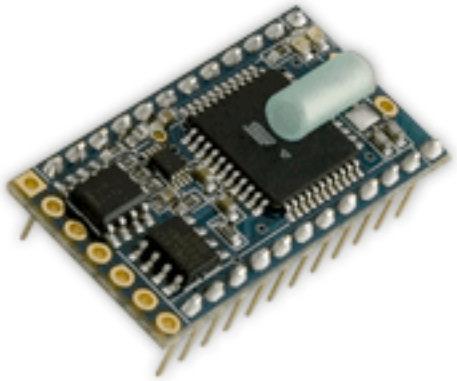


pic

higher level ⟷ **lower level**

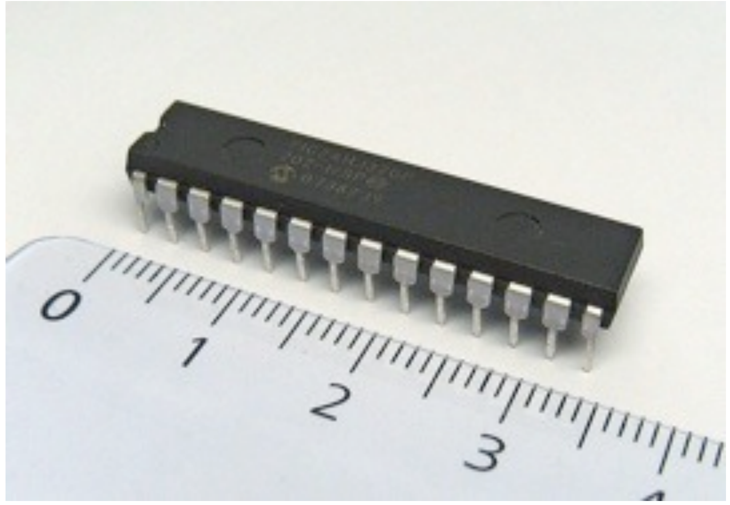
Assembly

basic stamp



bx 24

basic atom



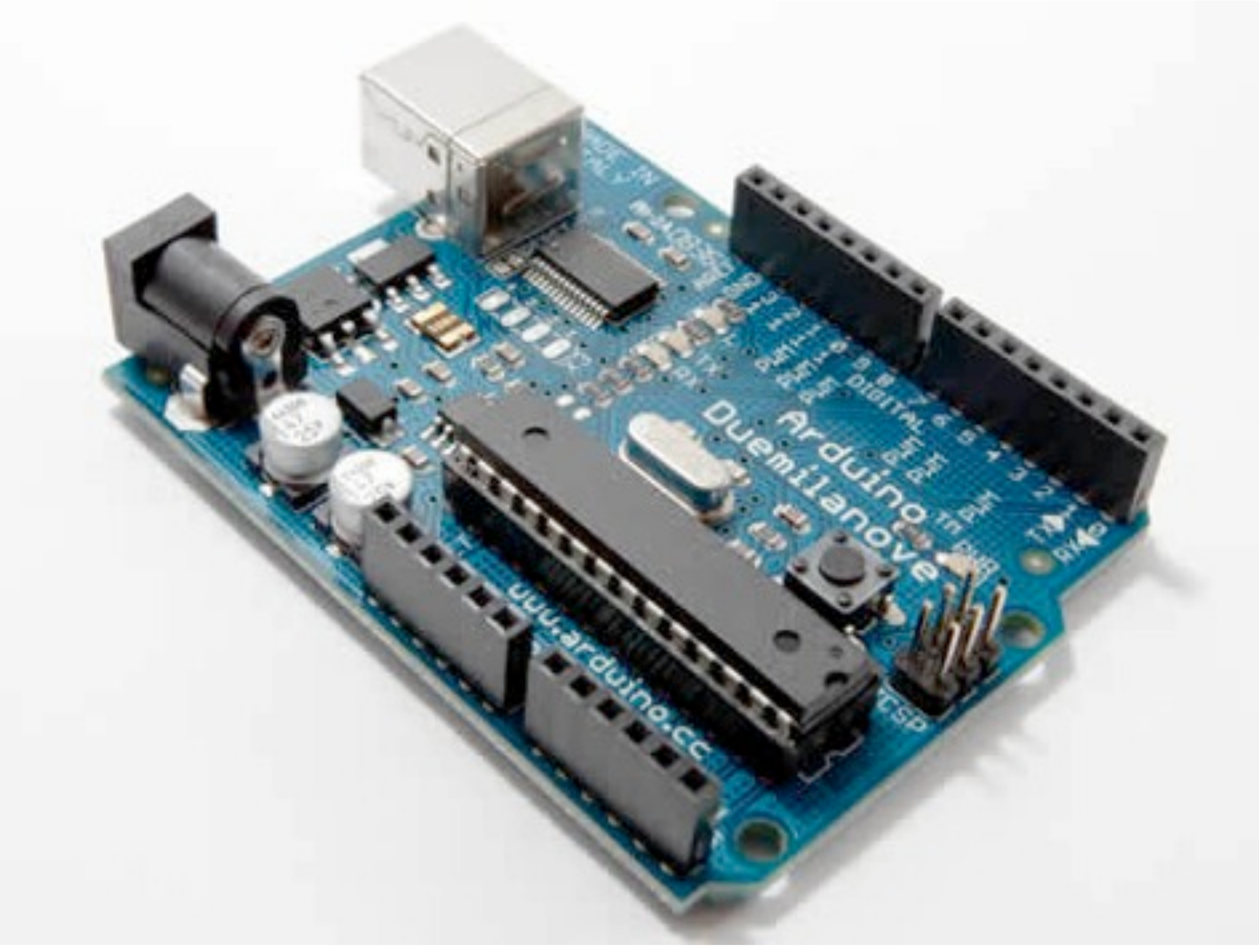
pic

higher level ⟷ **lower level**

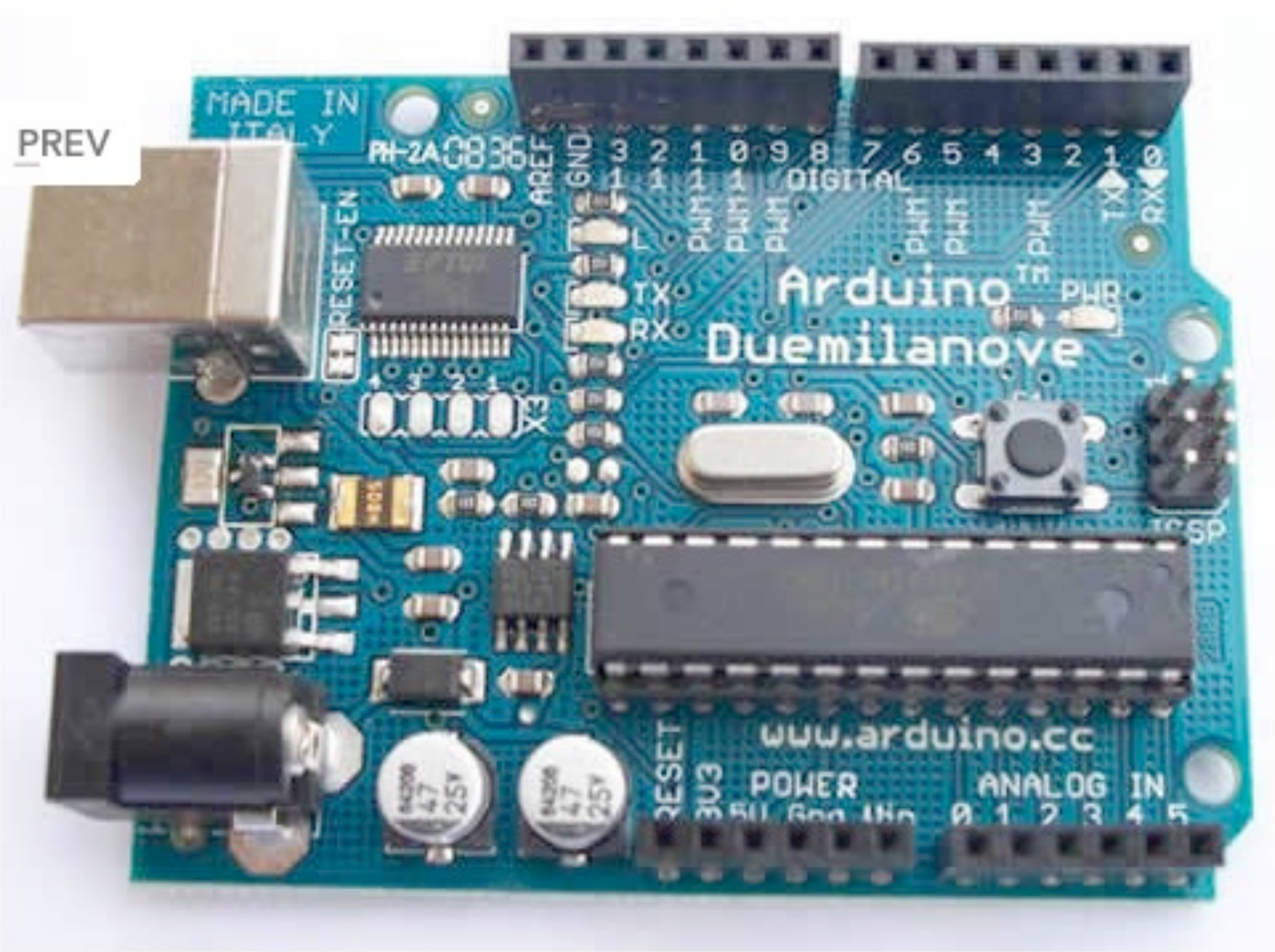
5
MAX MSP
ActionScript
Java
C++

Assembly

Processing



Atmel AT Mega 328



Atmel AT Mega 328

photo credits © arduino.cc

Arduino is an open source physical computing platform based on a simple input/output (I/O) board and a development environment that implements the processing language.
The IDE can be downloaded at **www.arduino.cc**

Main Advantages:

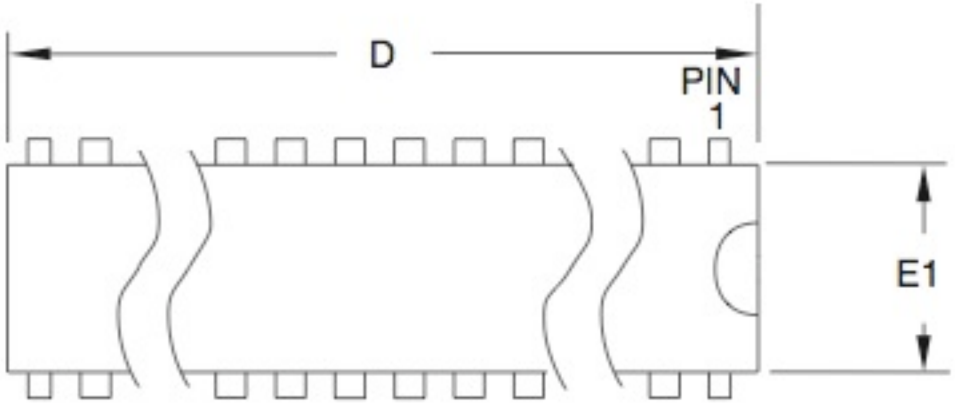
-Multi-platform environment, can run on Windows, Macintosh and Linux

-cheap hardware (around 25 €)

-huge community with tons of libraries

-open source hardware and software

OUR CPU:



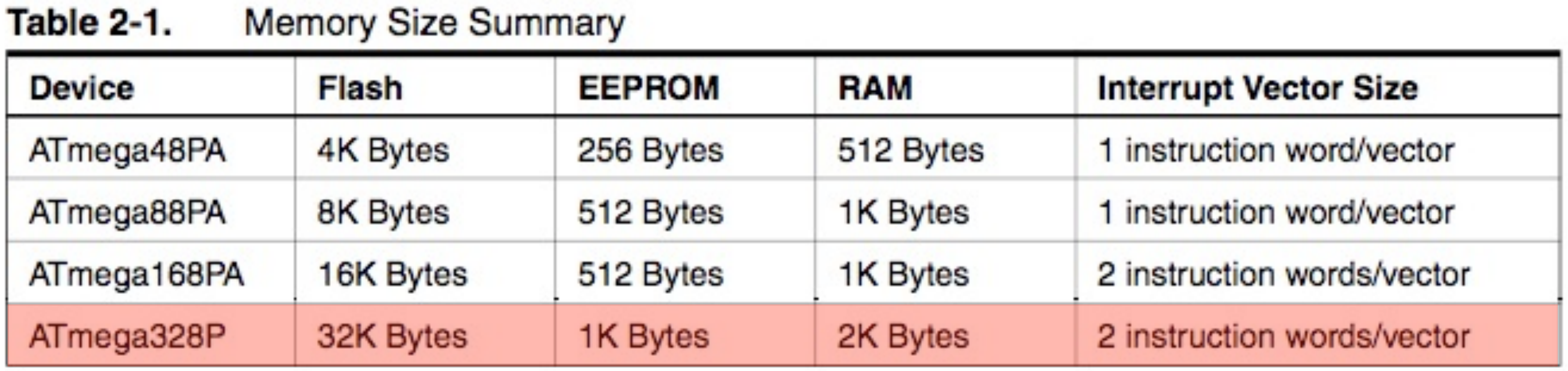
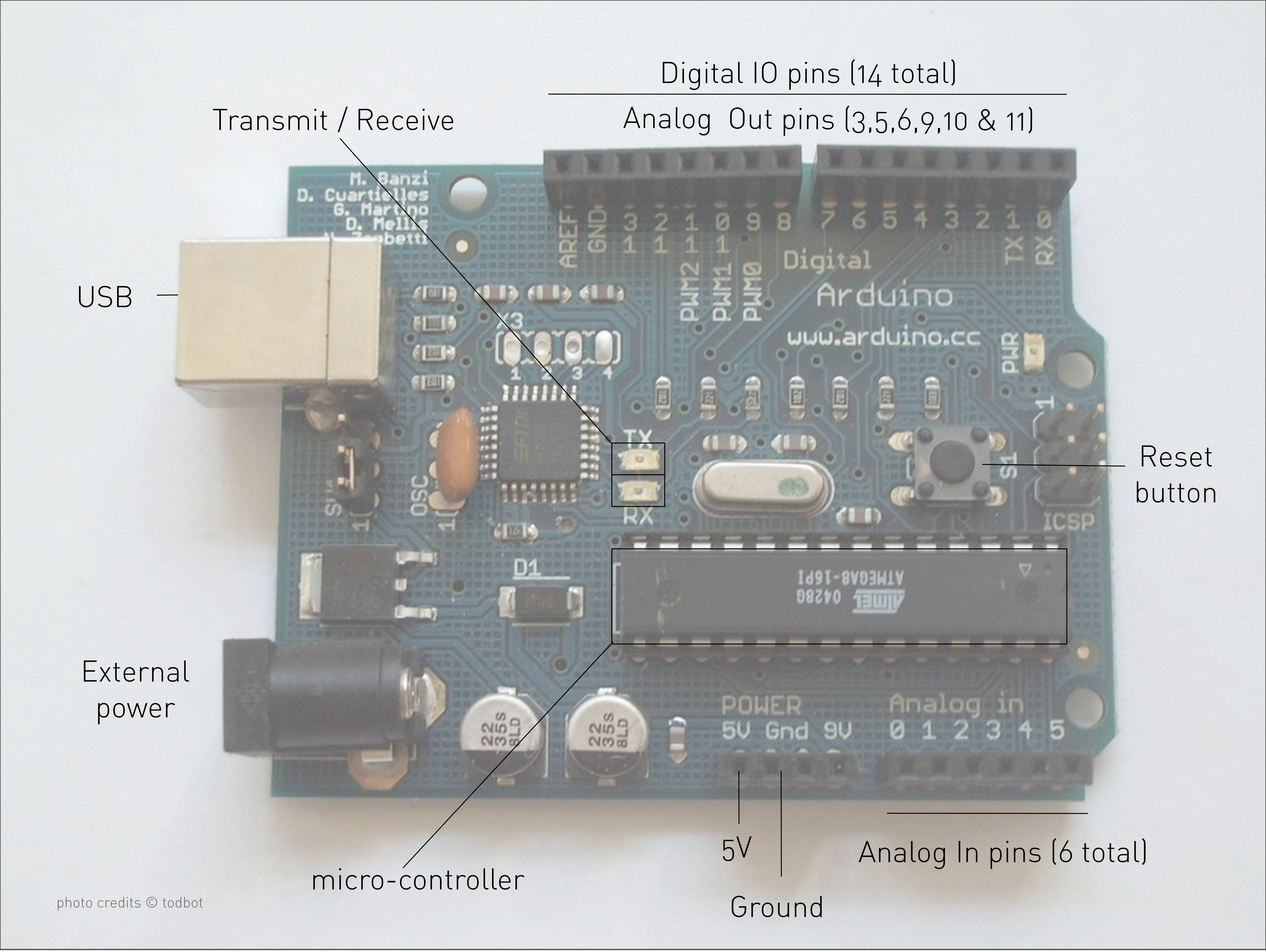


Table 2-1. Memory Size Summary

Device	Flash	EEPROM	RAM	Interrupt Vector Size
ATmega48PA	4K Bytes	256 Bytes	512 Bytes	1 instruction word/vector
ATmega88PA	8K Bytes	512 Bytes	1K Bytes	1 instruction word/vector
ATmega168PA	16K Bytes	512 Bytes	1K Bytes	2 instruction words/vector
ATmega328P	32K Bytes	1K Bytes	2K Bytes	2 instruction words/vector

photo credits © atmel

photo credits © wikimedia



Digital IO pins (14 total)

Analog Out pins (3,5,6,9,10 & 11)

Transmit / Receive

USB

Reset button

External power

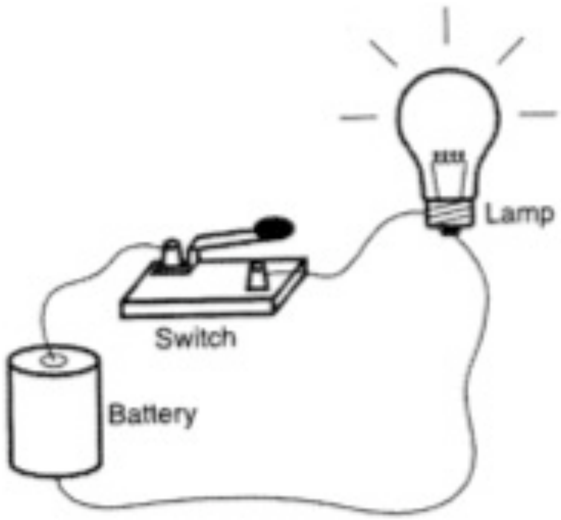
micro-controller

5V

Ground

Analog In pins (6 total)

photo credits © todbot



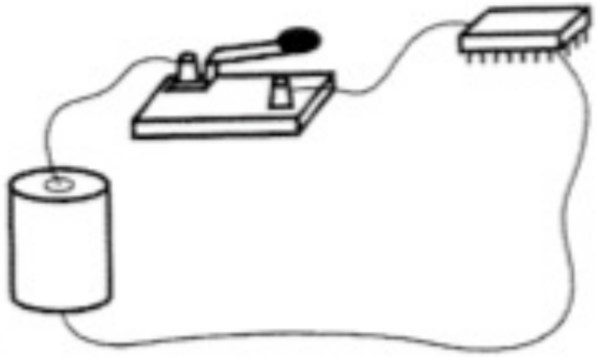
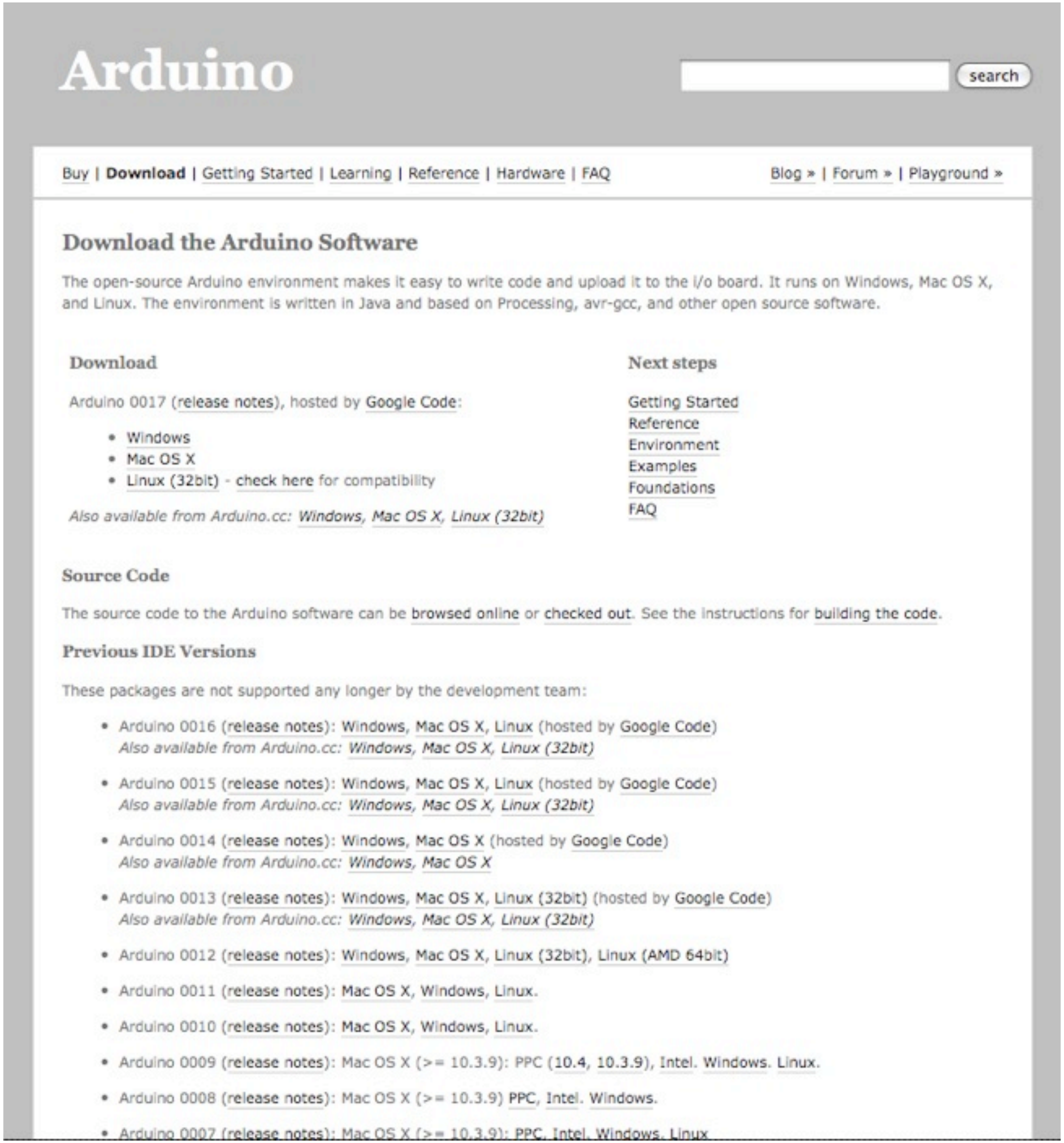


photo credits © tom igoe


Arduino
search
Buy | Download | Getting Started | Learning | Reference | Hardware | FAQ
Blog » | Forum » | Playground »

Download the Arduino Software

The open-source Arduino environment makes it easy to write code and upload it to the i/o board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing, avr-gcc, and other open source software.

Download

Arduino 0017 (release notes), hosted by Google Code:

- Windows
- Mac OS X
- Linux (32bit) - check here for compatibility

Also available from Arduino.cc: Windows, Mac OS X, Linux (32bit)

Next steps

Getting Started
Reference
Environment
Examples
Foundations
FAQ

Source Code

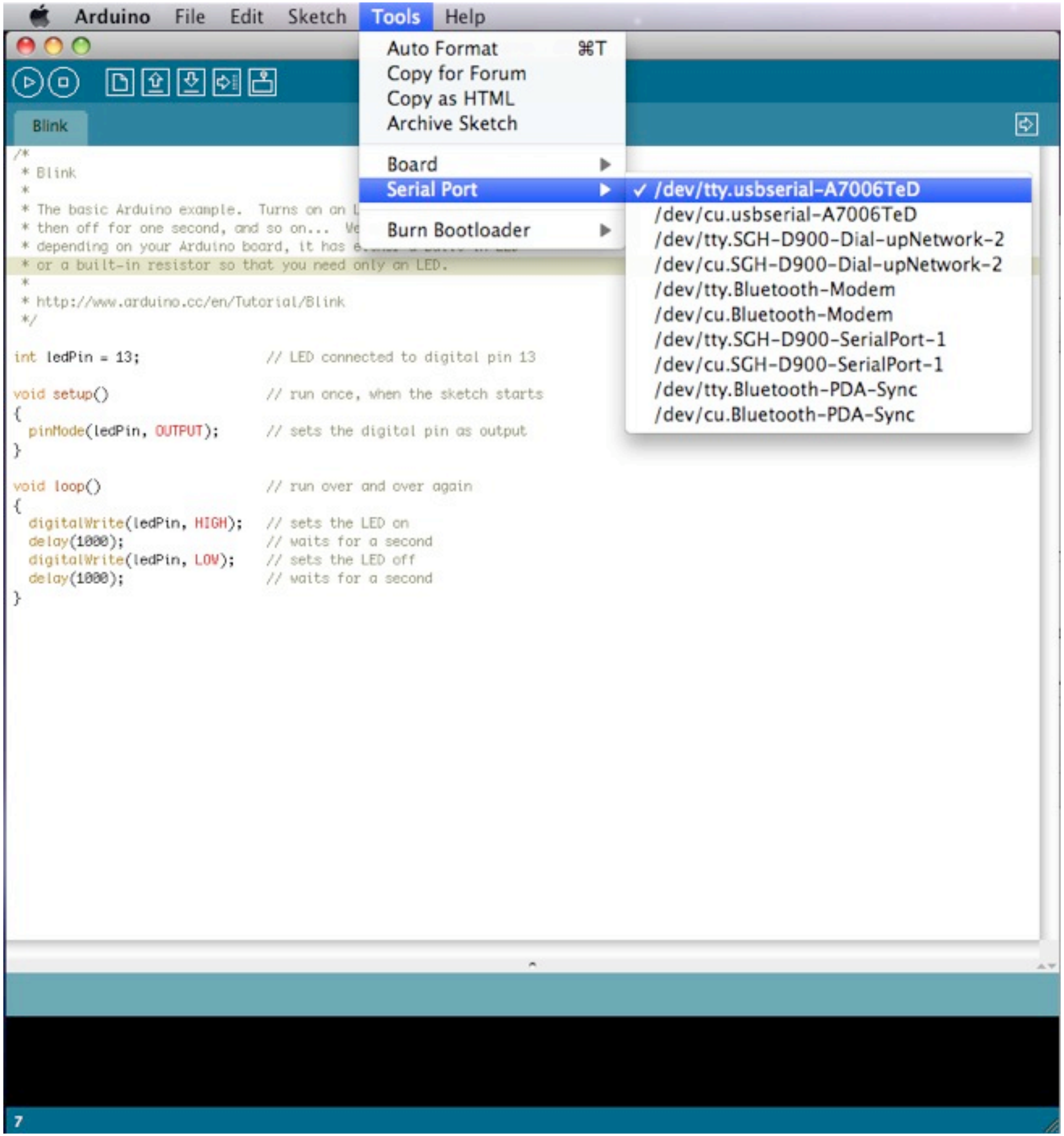
The source code to the Arduino software can be browsed online or checked out. See the instructions for building the code.

Previous IDE Versions

These packages are not supported any longer by the development team:

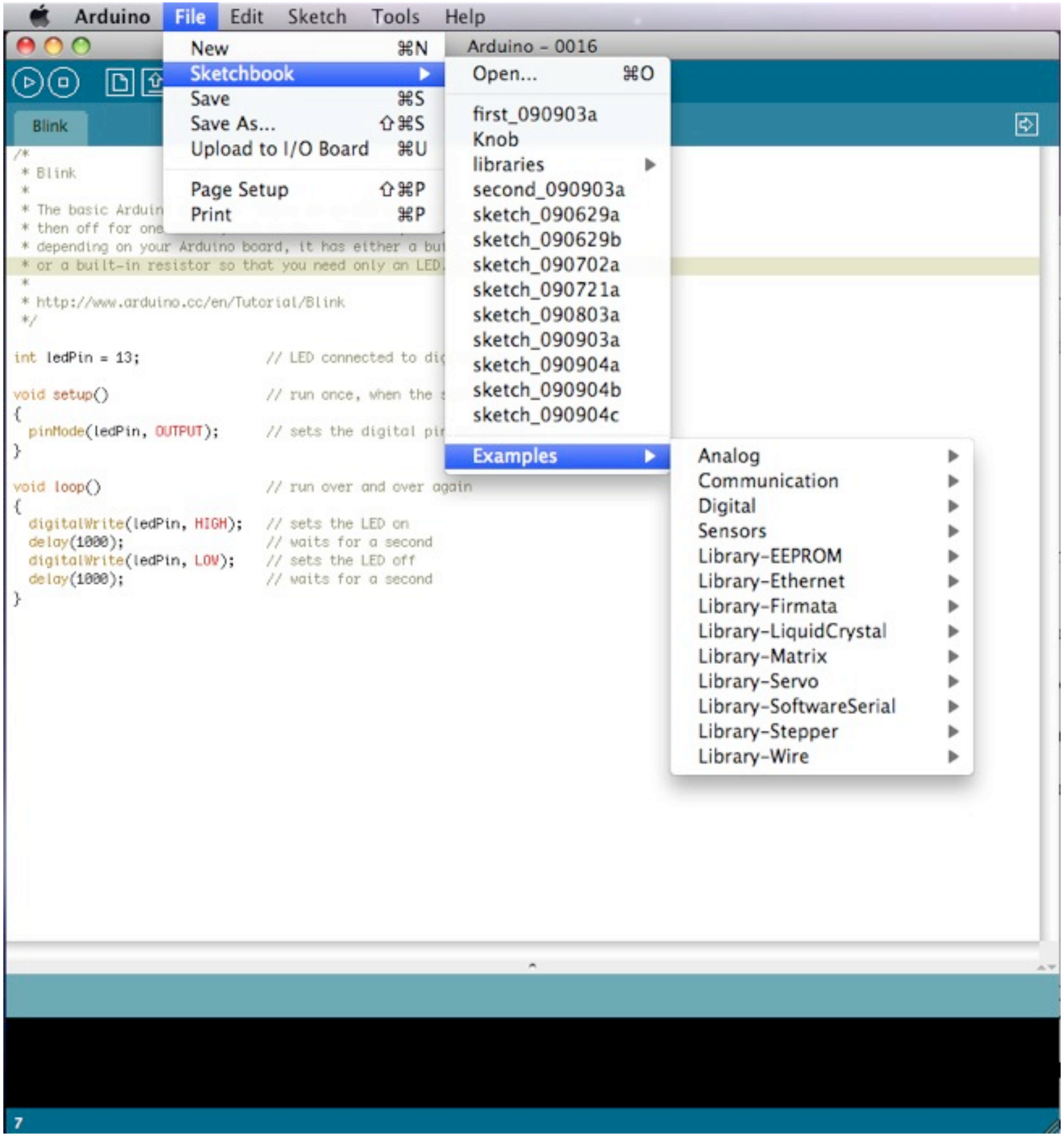
- Arduino 0016 (release notes): Windows, Mac OS X, Linux (hosted by Google Code)
  Also available from Arduino.cc: Windows, Mac OS X, Linux (32bit)
- Arduino 0015 (release notes): Windows, Mac OS X, Linux (hosted by Google Code)
  Also available from Arduino.cc: Windows, Mac OS X, Linux (32bit)
- Arduino 0014 (release notes): Windows, Mac OS X (hosted by Google Code)
  Also available from Arduino.cc: Windows, Mac OS X
- Arduino 0013 (release notes): Windows, Mac OS X, Linux (32bit) (hosted by Google Code)
  Also available from Arduino.cc: Windows, Mac OS X, Linux (32bit)
- Arduino 0012 (release notes): Windows, Mac OS X, Linux (32bit), Linux (AMD 64bit)
- Arduino 0011 (release notes): Mac OS X, Windows, Linux.
- Arduino 0010 (release notes): Mac OS X, Windows, Linux.
- Arduino 0009 (release notes): Mac OS X (>= 10.3.9): PPC (10.4, 10.3.9), Intel. Windows. Linux.
- Arduino 0008 (release notes): Mac OS X (>= 10.3.9) PPC, Intel. Windows.
- Arduino 0007 (release notes): Mac OS X (>= 10.3.9): PPC, Intel, Windows, Linux

- **Download software: http://arduino.cc/**
- Mac OS X PPC or Intel (must pick)
- Windows
- **Install drivers**
- In "drivers" folder, pick appropriate one
- Windows: unzip driver, plug in board, setup
- "macosx-setup-command" for Mac folk
- Reboot

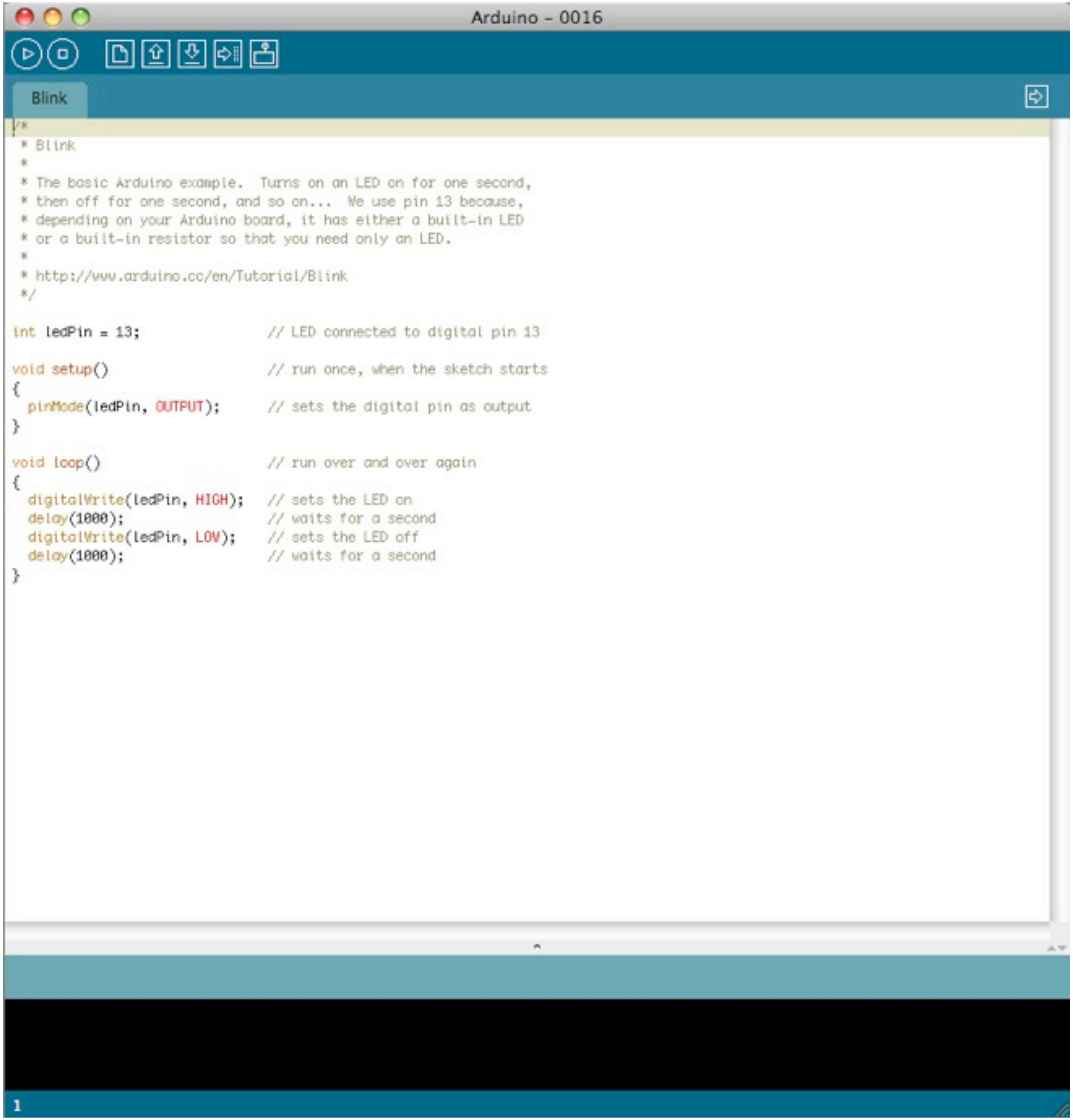


Arduino
File
Edit
Sketch
Tools
Help
Blink
Auto Format ⌘T
Copy for Forum
Copy as HTML
Archive Sketch
Board ▶
Serial Port ▶
Burn Bootloader ▶
✓ /dev/tty.usbserial-A7006TeD
/dev/cu.usbserial-A7006TeD
/dev/tty.SGH-D900-Dial-upNetwork-2
/dev/cu.SGH-D900-Dial-upNetwork-2
/dev/tty.Bluetooth-Modem
/dev/cu.Bluetooth-Modem
/dev/tty.SGH-D900-SerialPort-1
/dev/cu.SGH-D900-SerialPort-1
/dev/tty.Bluetooth-PDA-Sync
/dev/cu.Bluetooth-PDA-Sync
7

```
/*
 * Blink
 *
 * The basic Arduino example.  Turns on an L
 * then off for one second, and so on...  We
 * depending on your Arduino board, it has e
 * 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
}
```



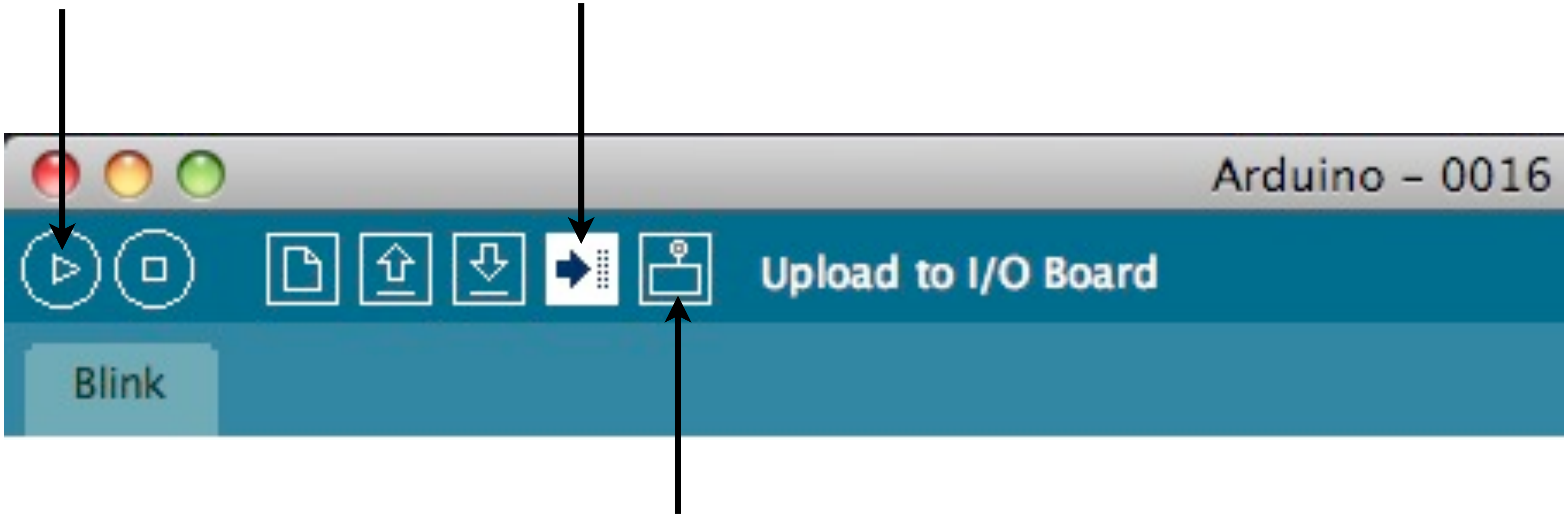
```
/*
 * Blink
 *
 * The basic Arduin
 * then off for one
 * depending on your Arduino board, it has either a bui
 * or a built-in resistor so that you need only an LED.
 *
 * http://www.arduino.cc/en/Tutorial/Blink
 */

int ledPin = 13;                // LED connected to dig

void setup()                    // run once, when the s
{
  pinMode(ledPin, OUTPUT);      // sets the digital pi
}

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
}
```



```
/*
 * Blink
 *
 * 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
}
```

compile button

upload button



serial monitor

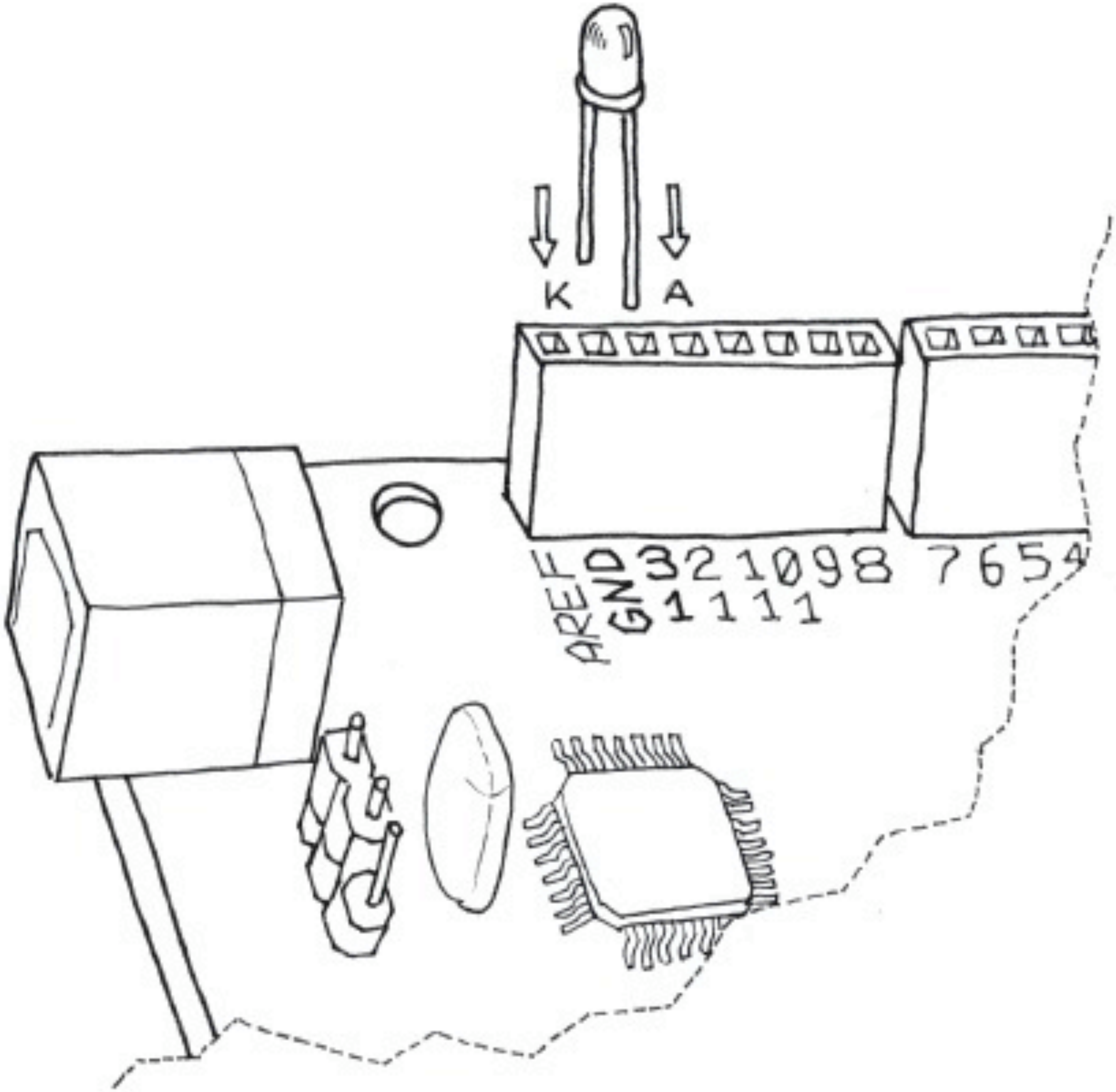
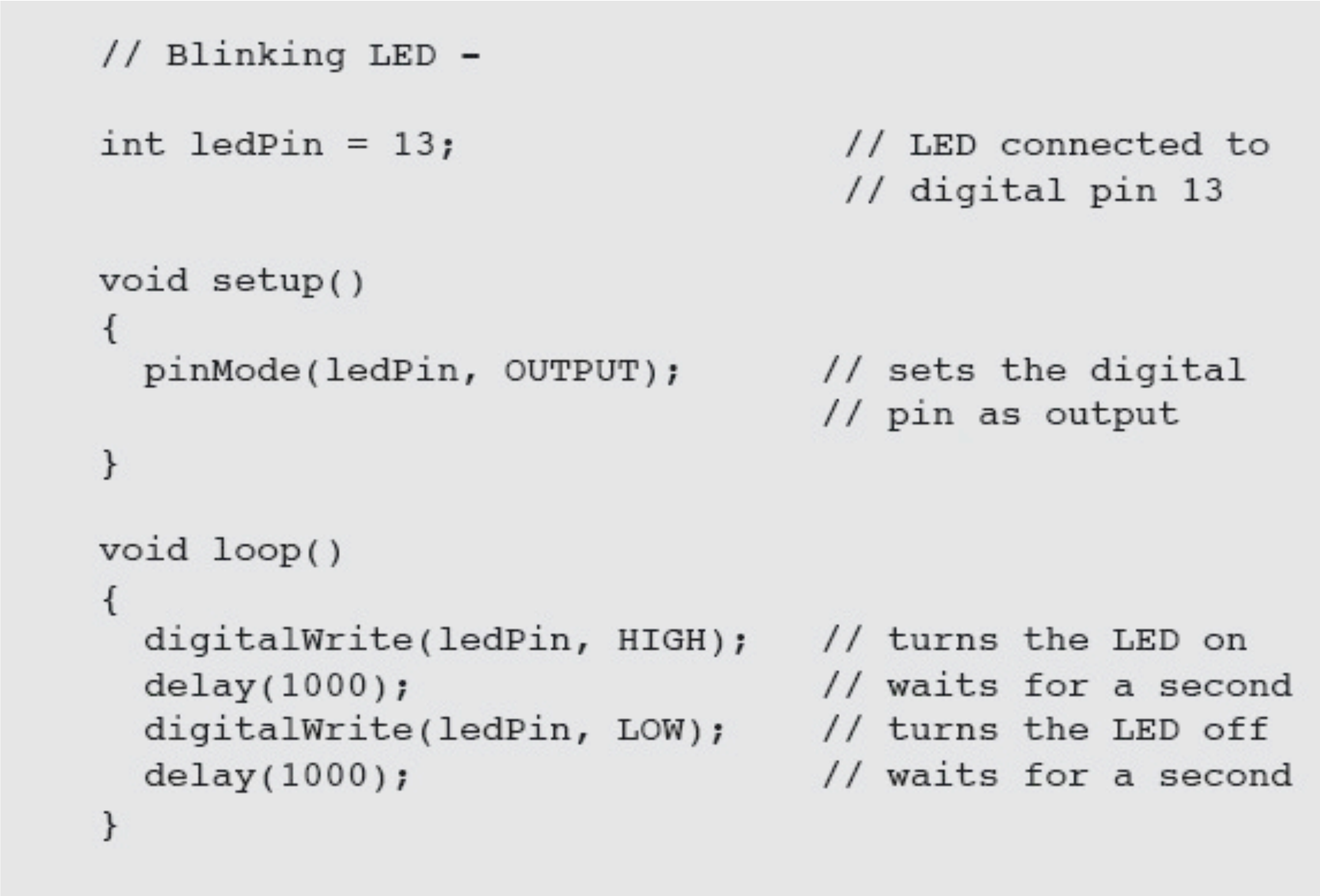
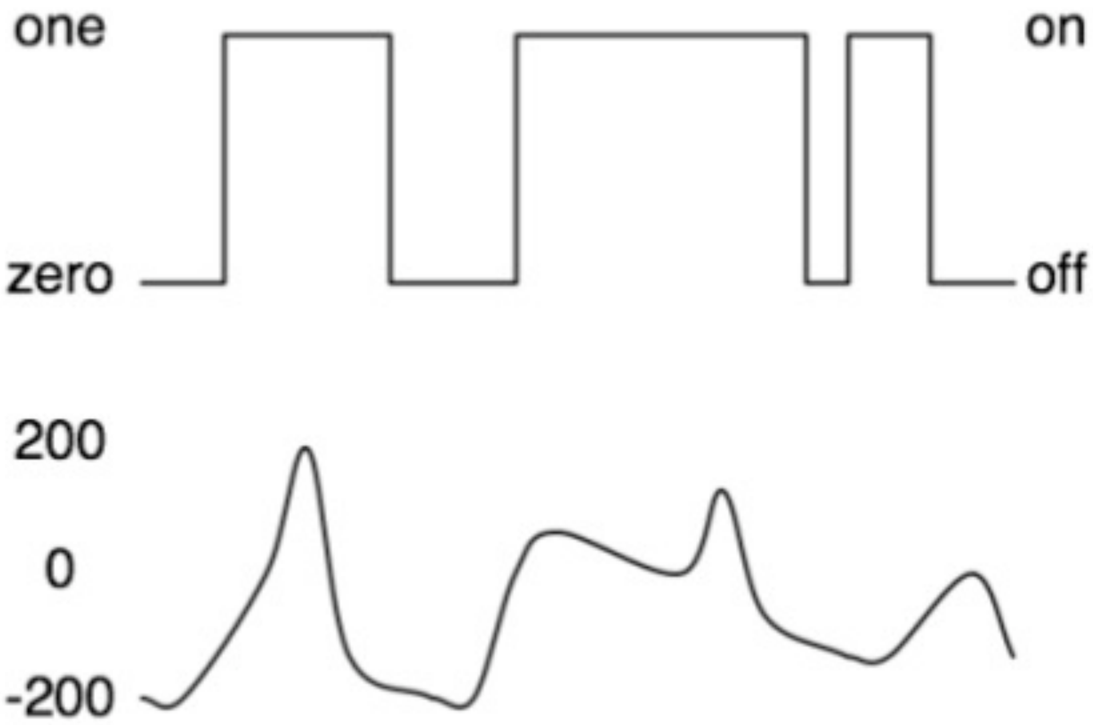


photo credits © elisa canducci



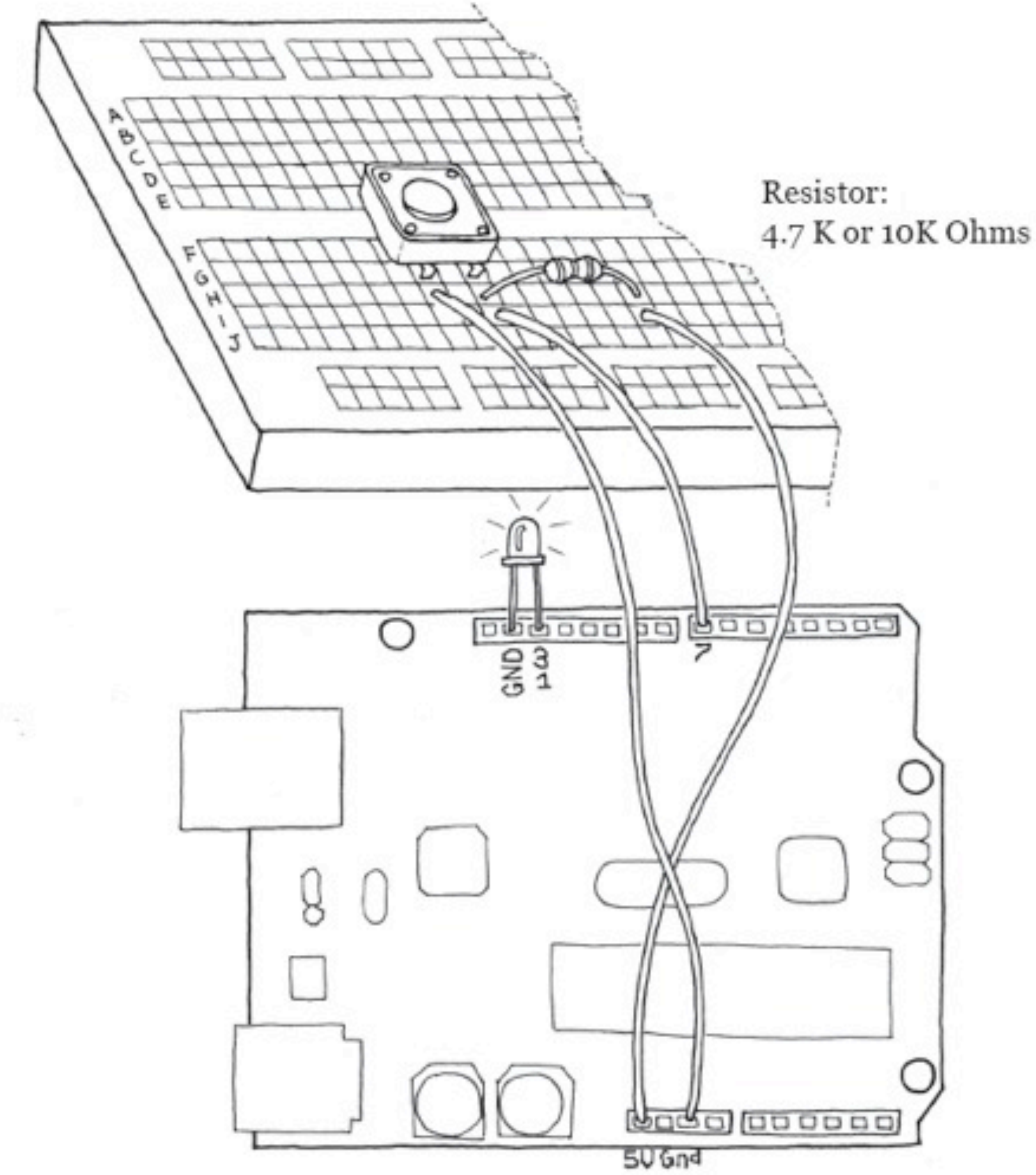
```
// Blinking LED -

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

void setup()
{
  pinMode(ledPin, OUTPUT);      // sets the digital
                                // pin as output
}

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

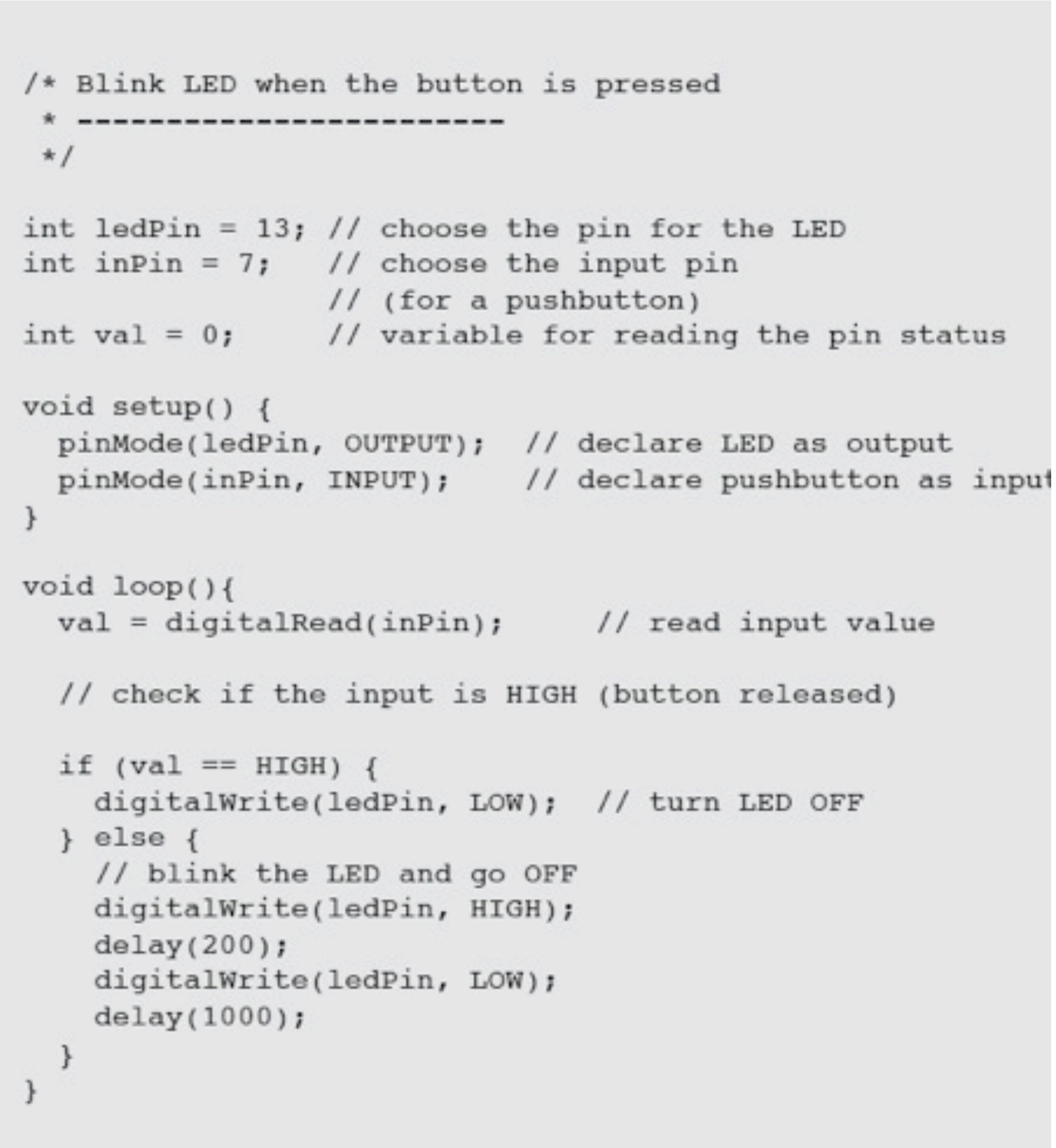
Digital read (listening)



Digital Read vs. Analog Read

photo credits © todbot.com

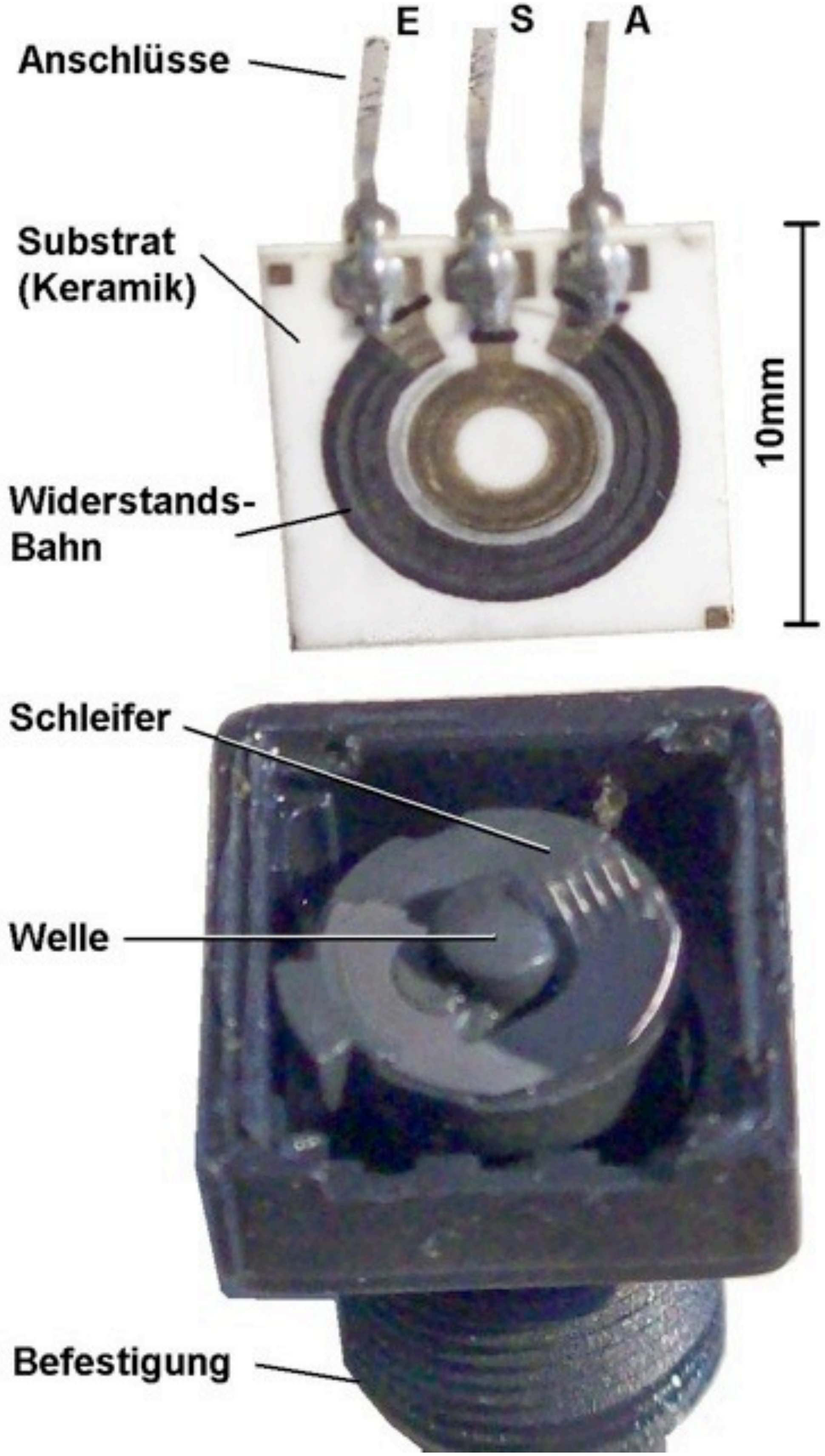

Resistor:
4.7 K or 10K Ohms

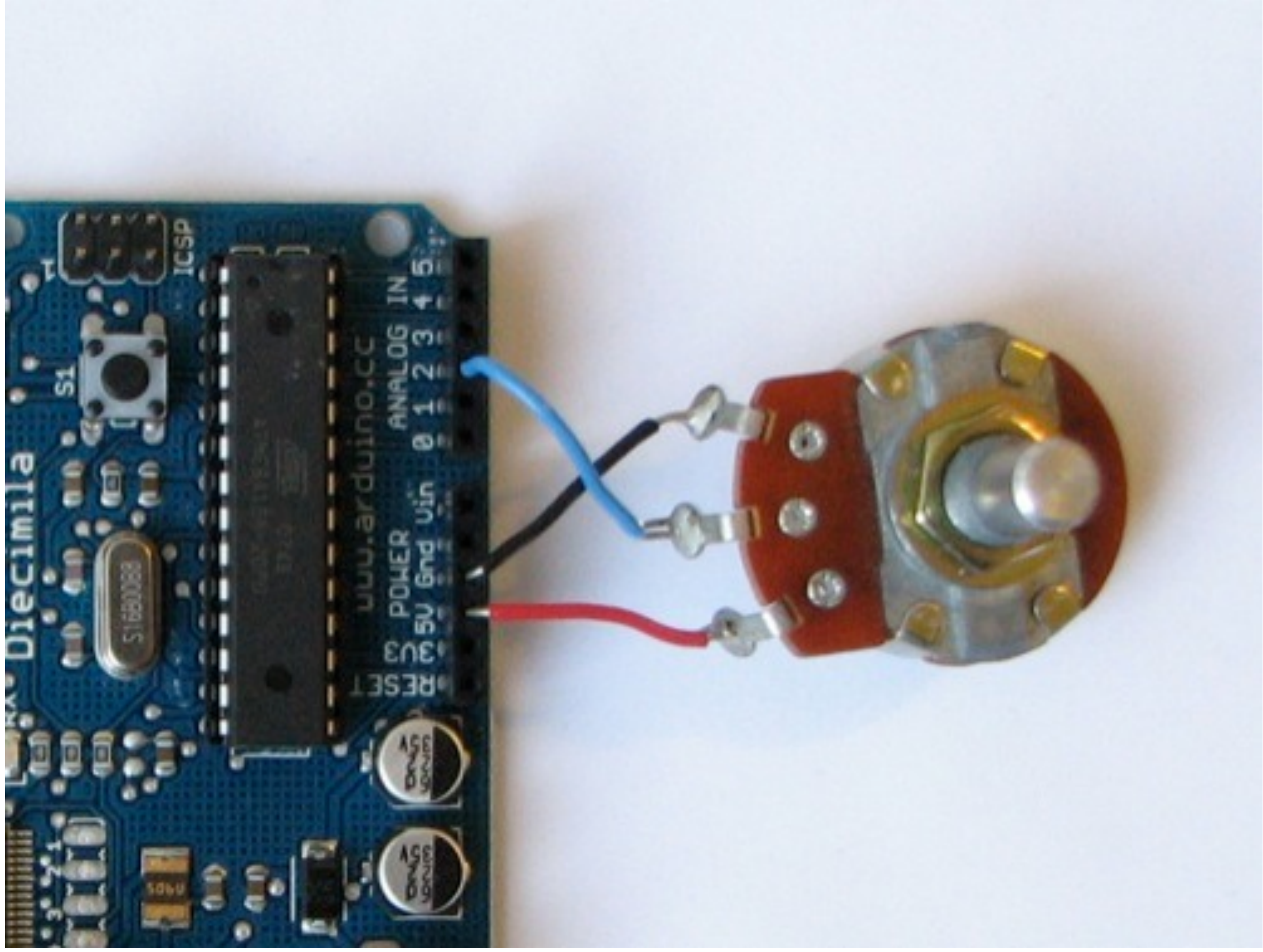


```
/* Blink LED when the button is pressed
 * ------------------------
 */

int ledPin = 13; // choose the pin for the LED
int inPin = 7;   // choose the input pin
                 // (for a pushbutton)
int val = 0;     // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT);  // declare LED as output
  pinMode(inPin, INPUT);    // declare pushbutton as inpu
}

void loop(){
  val = digitalRead(inPin);     // read input value

  // check if the input is HIGH (button released)

  if (val == HIGH) {
    digitalWrite(ledPin, LOW);  // turn LED OFF
  } else {
    // blink the LED and go OFF
    digitalWrite(ledPin, HIGH);
    delay(200);
    digitalWrite(ledPin, LOW);
    delay(1000);
  }
}
```

photo credits © elisa canducci

Analog read


E
S
A
Anschlüsse
Substrat (Keramik)
10mm
Widerstands-Bahn
Schleifer
Welle
Befestigung



```
int potPin = 2;    // select the input pin for the potentiometer
int ledPin = 13;   // select the pin for the LED
int val = 0;       // variable to store the value coming from the sensor

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

void loop() {
  val = analogRead(potPin);    // read the value from the sensor
  digitalWrite(ledPin, HIGH);  // turn the ledPin on
  delay(val);                  // stop the program for some time
  digitalWrite(ledPin, LOW);   // turn the ledPin off
  delay(val);                  // stop the program for some time
}
```

```
int analogValue = 0;    // variable to hold the analog value

void setup() {
  // open the serial port at 9600 bps:
  Serial.begin(9600);
}

void loop() {
  // read the analog input on pin 0:
  analogValue = analogRead(0);

  // print it out in many formats:
  Serial.println(analogValue);       // print as an ASCII-encoded decimal

  // delay 10 milliseconds before the next reading:
  delay(10);
}
```

Analog read

Advanced Sensors:

Thermistor

Bend Sensor

PIR Sensor

photo credits © wikimedia

Force Sensor

Potentiometer

Magnet Switch

photo credits © wikimedia

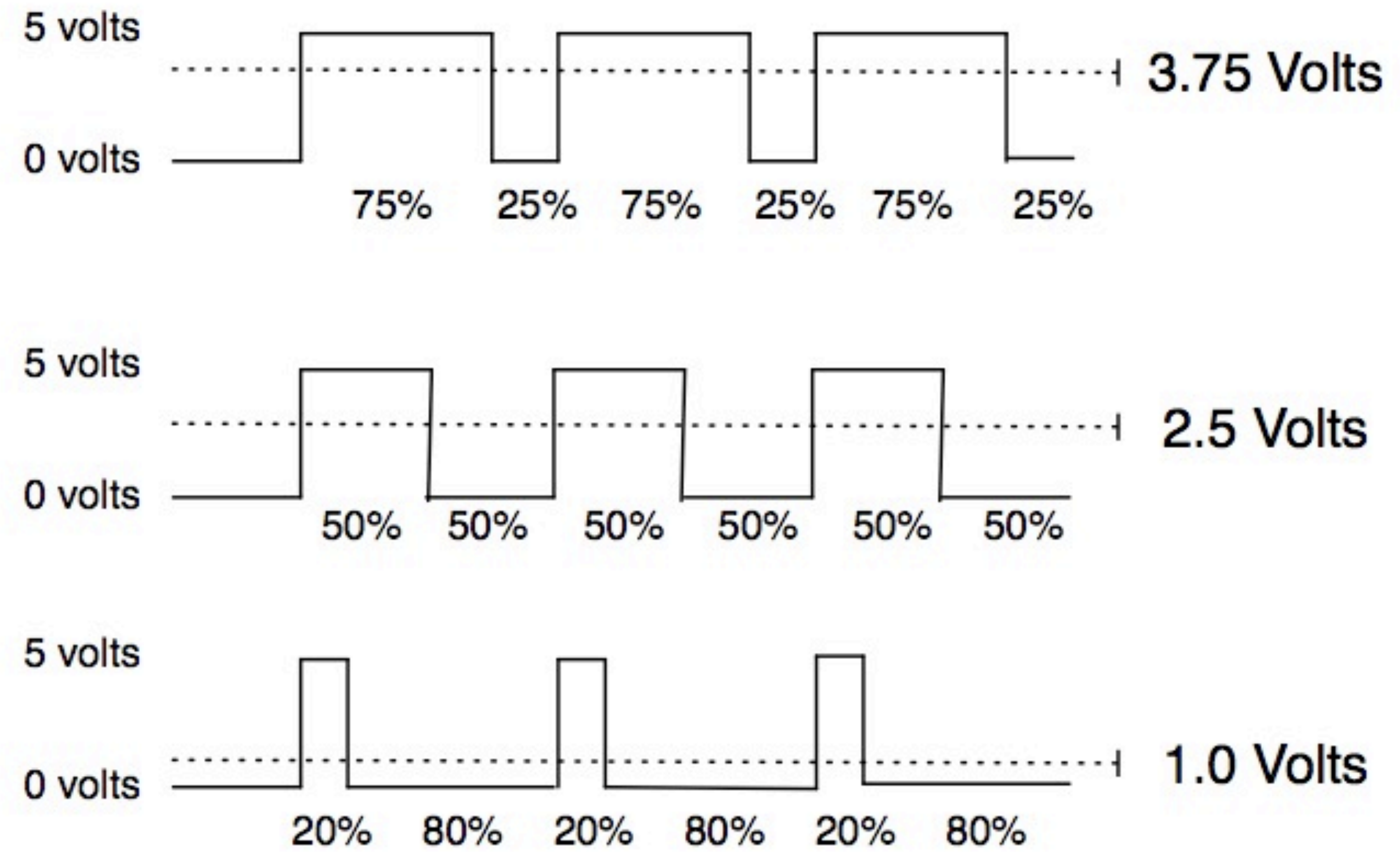
Distance IR Sensor

Touch QT Sensor

Ultrasound Sensor

photo credits © wikimedia

Analog write



PWM

photo credits © todbot.com

Hello World!

Three characteristics of PWM signals

- Pulse width range (min/max)
- Pulse period
- Voltage levels

(0-5V, for instance)width
periodheight

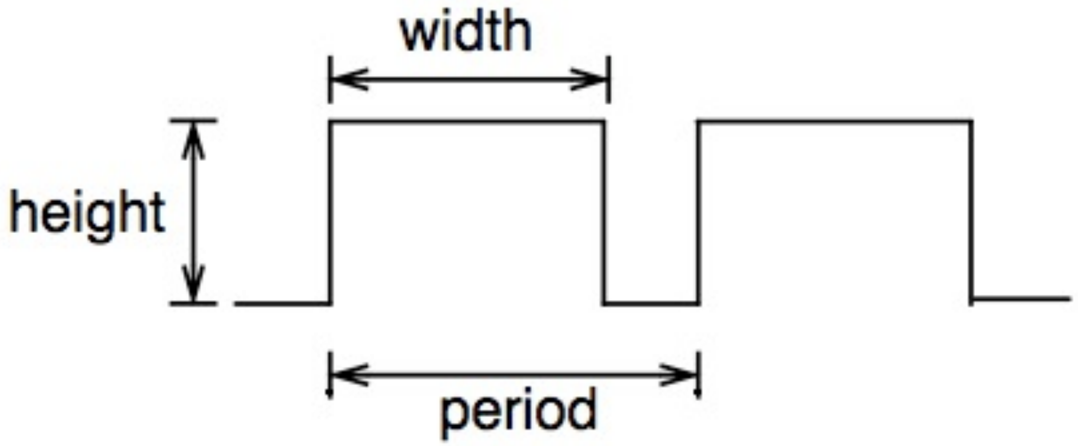
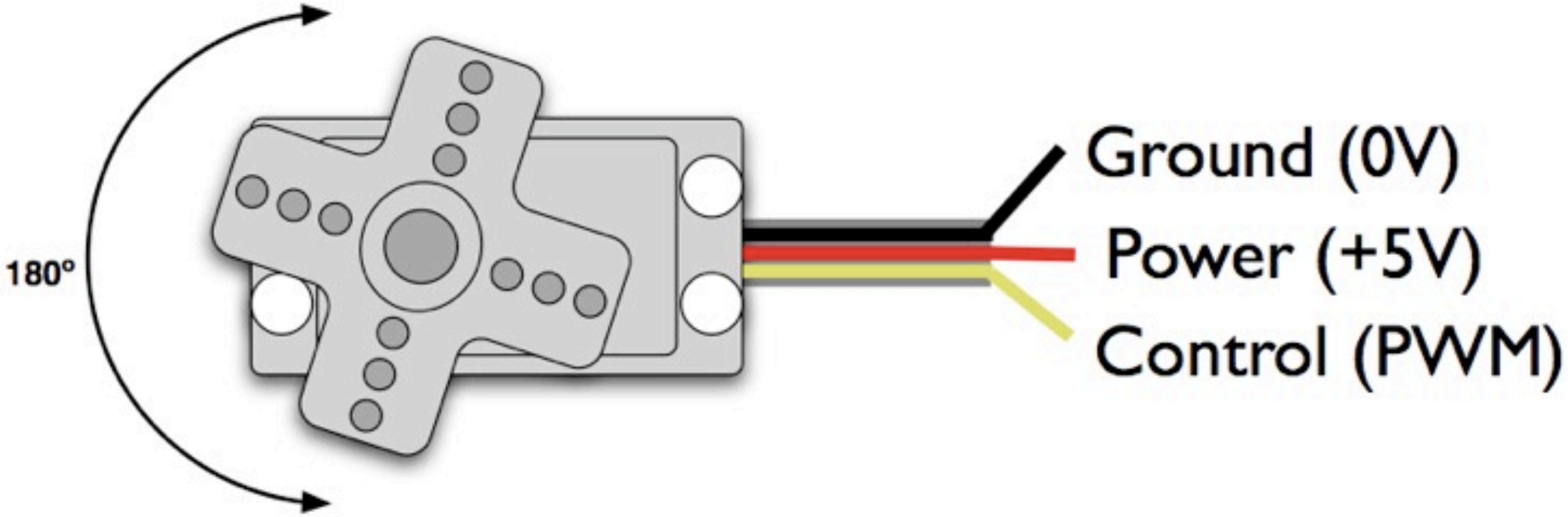


photo credits © todbot.com



- PWM freq is 50 Hz (i.e. every 20 millisecs)
- Pulse width ranges from 1 to 2 millisecs
- 1 millisec = full anti-clockwise position
- 2 millisec = full clockwise position

photo credits © todbot.com

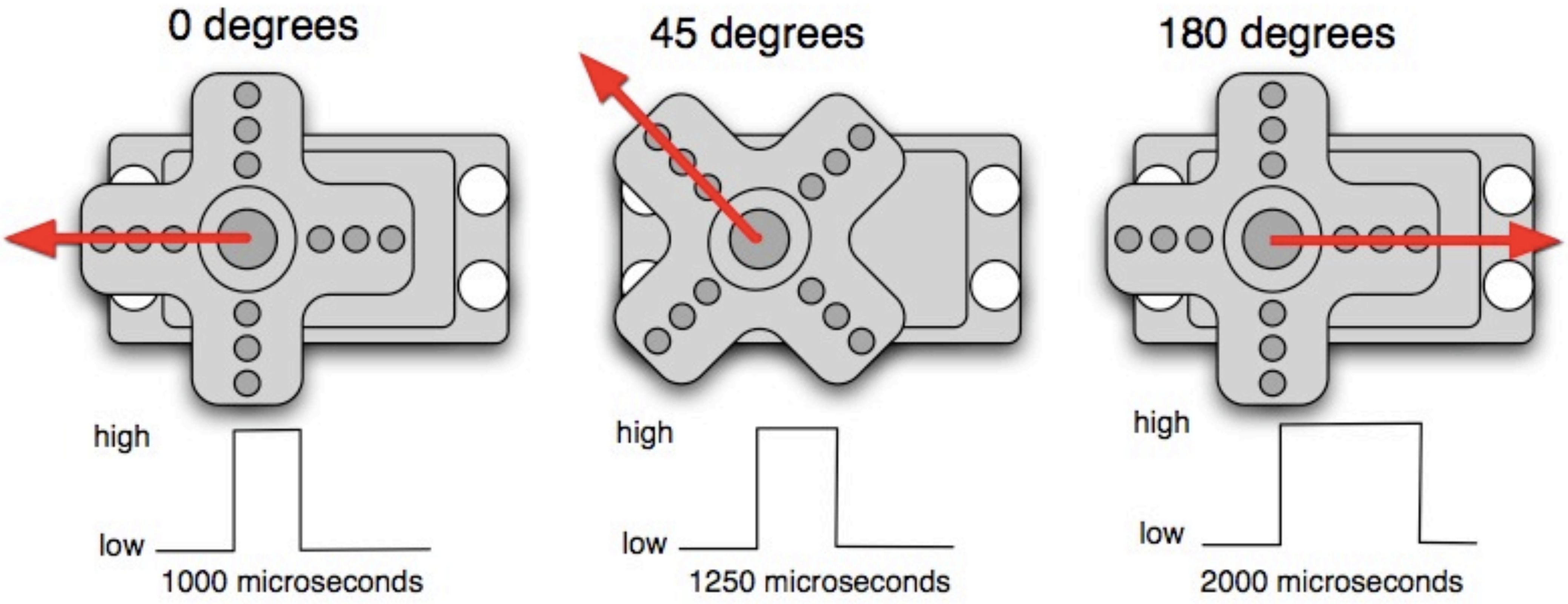
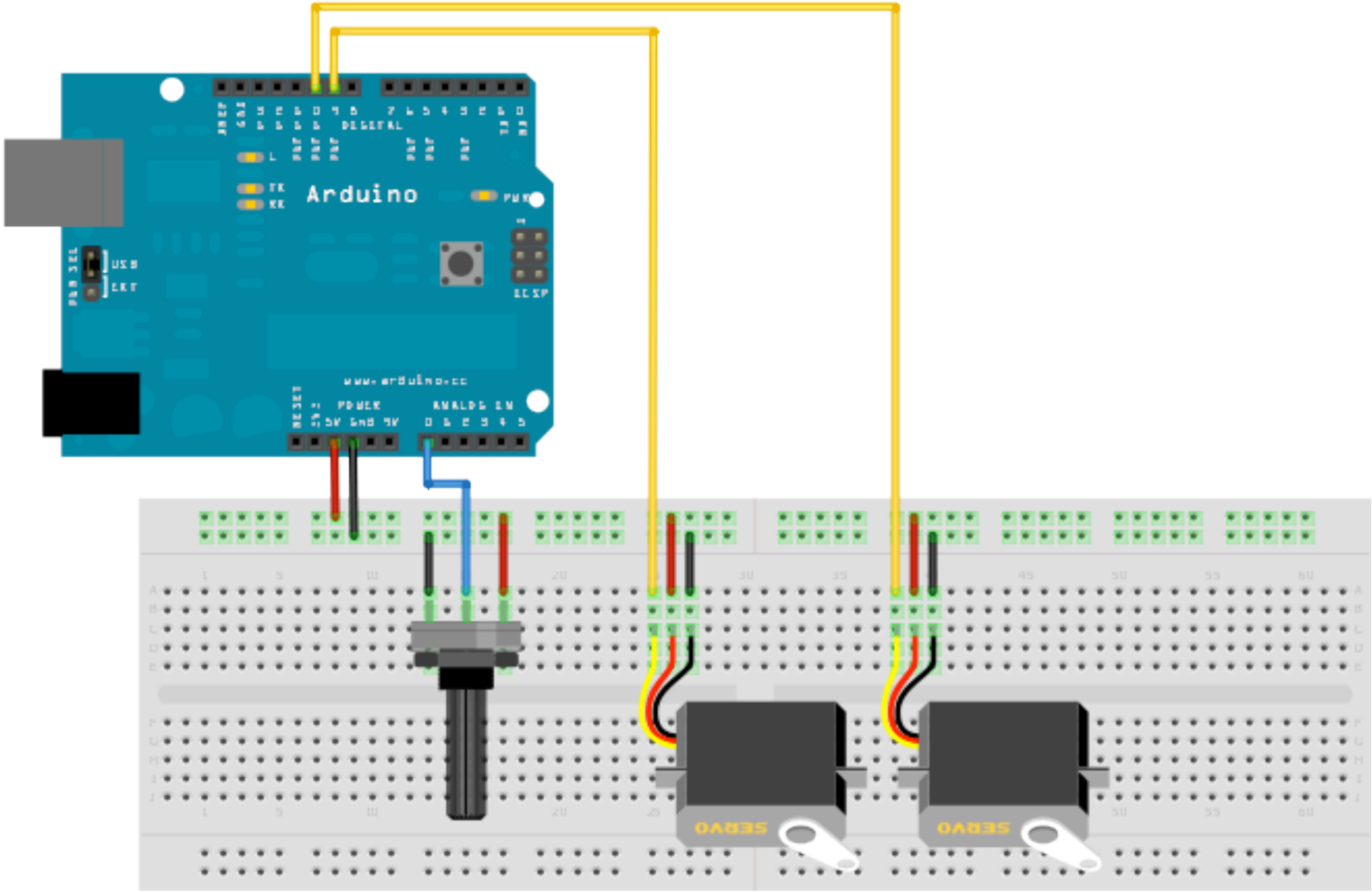
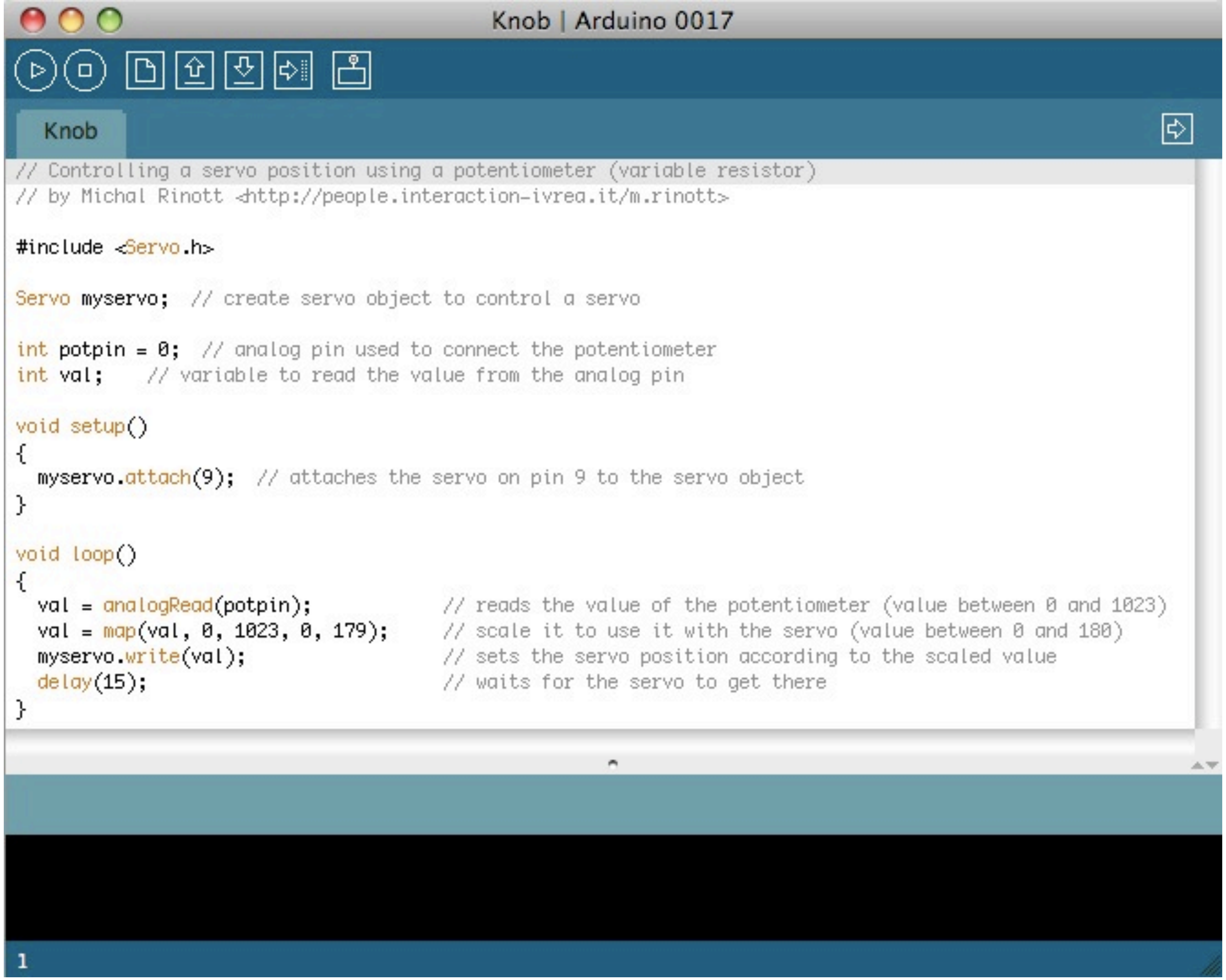


photo credits © todbot.com

Simple Servo Example


Arduino
DIGITAL
PWR
ICSP
www.arduino.cc
POWER
5V Gnd 9V
ANALOG IN
0 1 2 3 4 5
SERVO
SERVO

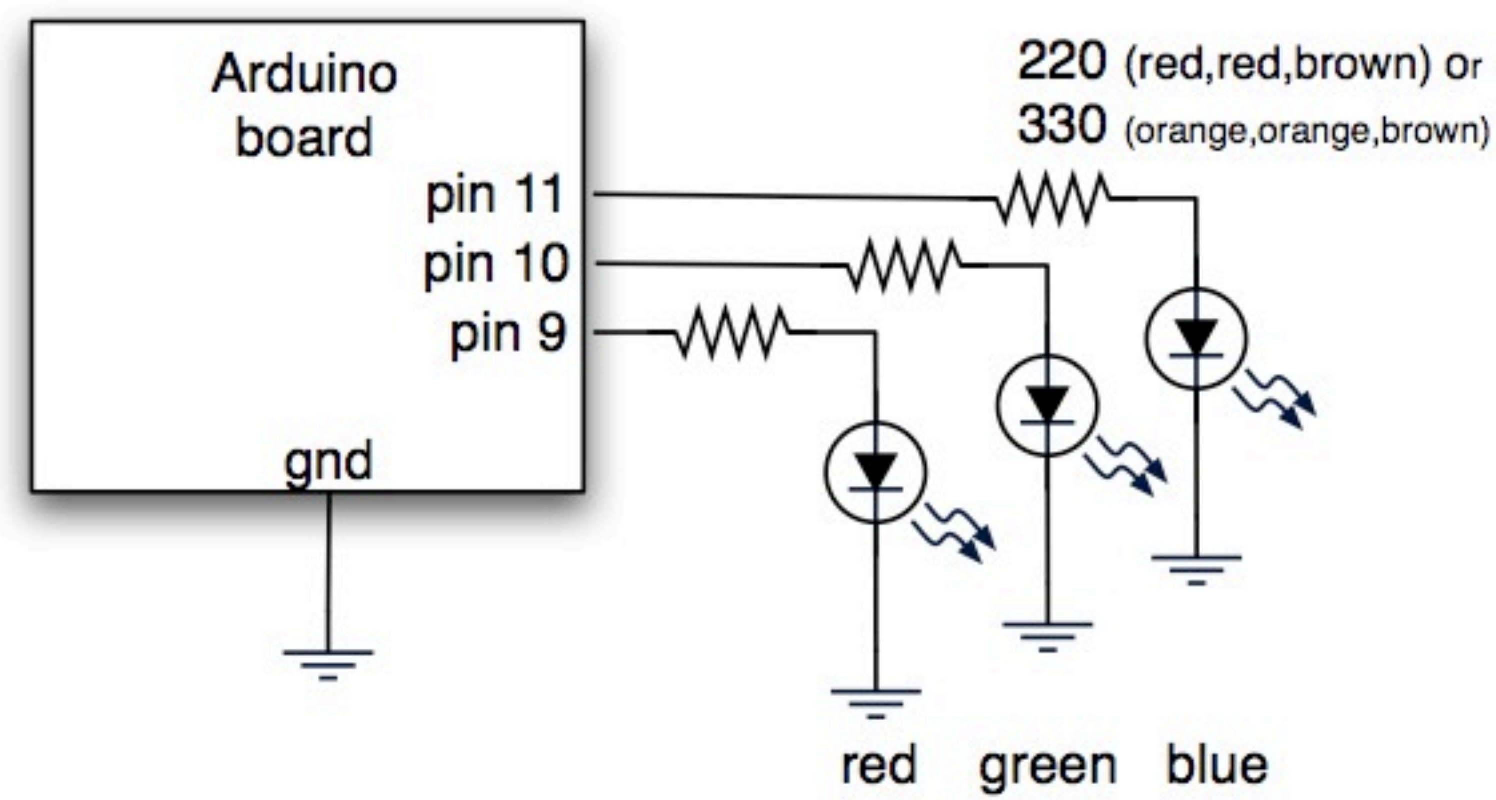


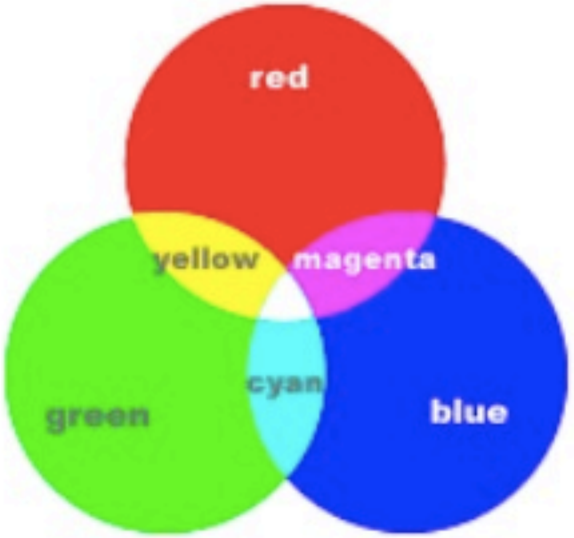
```
// Controlling a servo position using a potentiometer (variable resistor)
// by Michal Rinott <http://people.interaction-ivrea.it/m.rinott>

#include <Servo.h>

Servo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
int val;    // variable to read the value from the analog pin

void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop()
{
  val = analogRead(potpin);            // reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 179);     // scale it to use it with the servo (value between 0 and 180)
  myservo.write(val);                  // sets the servo position according to the scaled value
  delay(15);                           // waits for the servo to get there
}
```

File → Examples → Servo → Knob

RGB LEDs & Interaction with light

photo credits © elisa canducci





With RGB you can make any color

(except black)

photo credits © todbot.com

Debugging:

Keep in mind:

-in electronics nothing ever works right the first time
-when troubleshooting do always **one** modification at a time
-be systematic to solve a problem
-remember to take notes on how you solved the problem

Common sources of error:

-Is the circuit powered ?
-Is the pin mentioned in the software the same in hardware ?
-does the LED work ?
-is the resistor the right value ?
-is the software configured for the right serial port ?
-does another application have control over the serial port ?

End Part 2

hacking:

www.lowtech.propositions.org.uk

http://www.nastypixel.com/instantsoup/website/cover/

www.tinkersoup.de

arduino:

http://itp.nyu.edu/physcomp/Tutorials/Tutorials

http://www.ladyada.net/learn/arduino/index.html

www.arduino.cc

www.freeduino.com

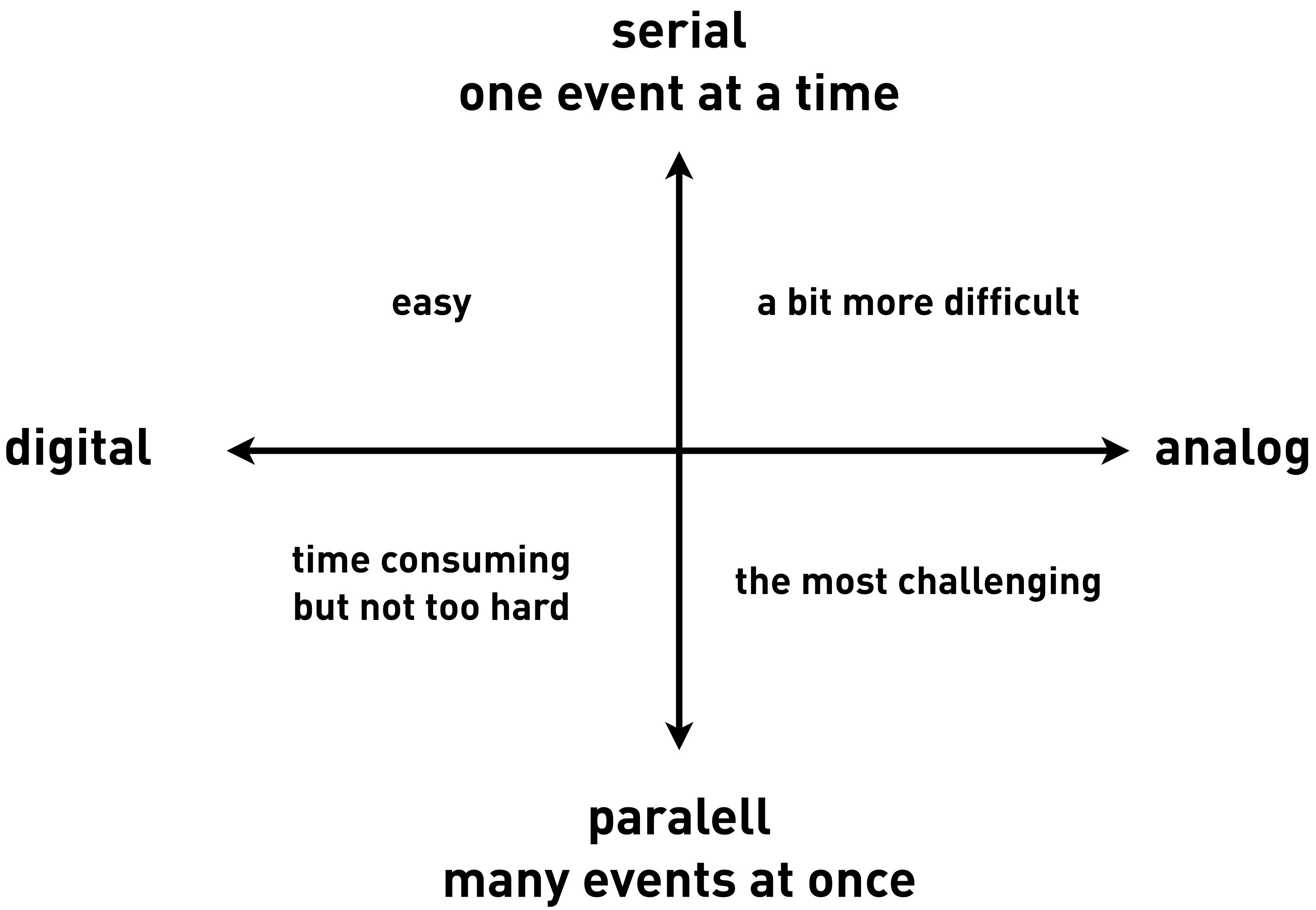
http://www.tigoe.net/pcomp/code/

www.todbot.com

Design your own project:

1.) Brainstorm &
write it down in plain text
from a persons view

example: if a peson walks in the room
the spotlight is switched on and
applause sound is played through the
speakers (scenario)



serial
one event at a time

easy

a bit more difficult

digital

analog

time consuming
but not too hard

the most challenging

paralell
many events at once

2.) categorize your project digital input, analog input, digital output, analog output

3.) Break it down in smaller parts start with pseudo code

Example:

```
If light level is less than ... then
    Turn Light on
    Turn motor on slow
     Loop again
```

4.) Brainstorm on the fastest route to reach your goal (hardware hacking)

5.) use the playground or freeduino.com to find re-usable software elements

6.) make an experimental step by step setup (hardware first)

Lecture: Alexander Wiethoff
Tutorials: Raphael Wimmer