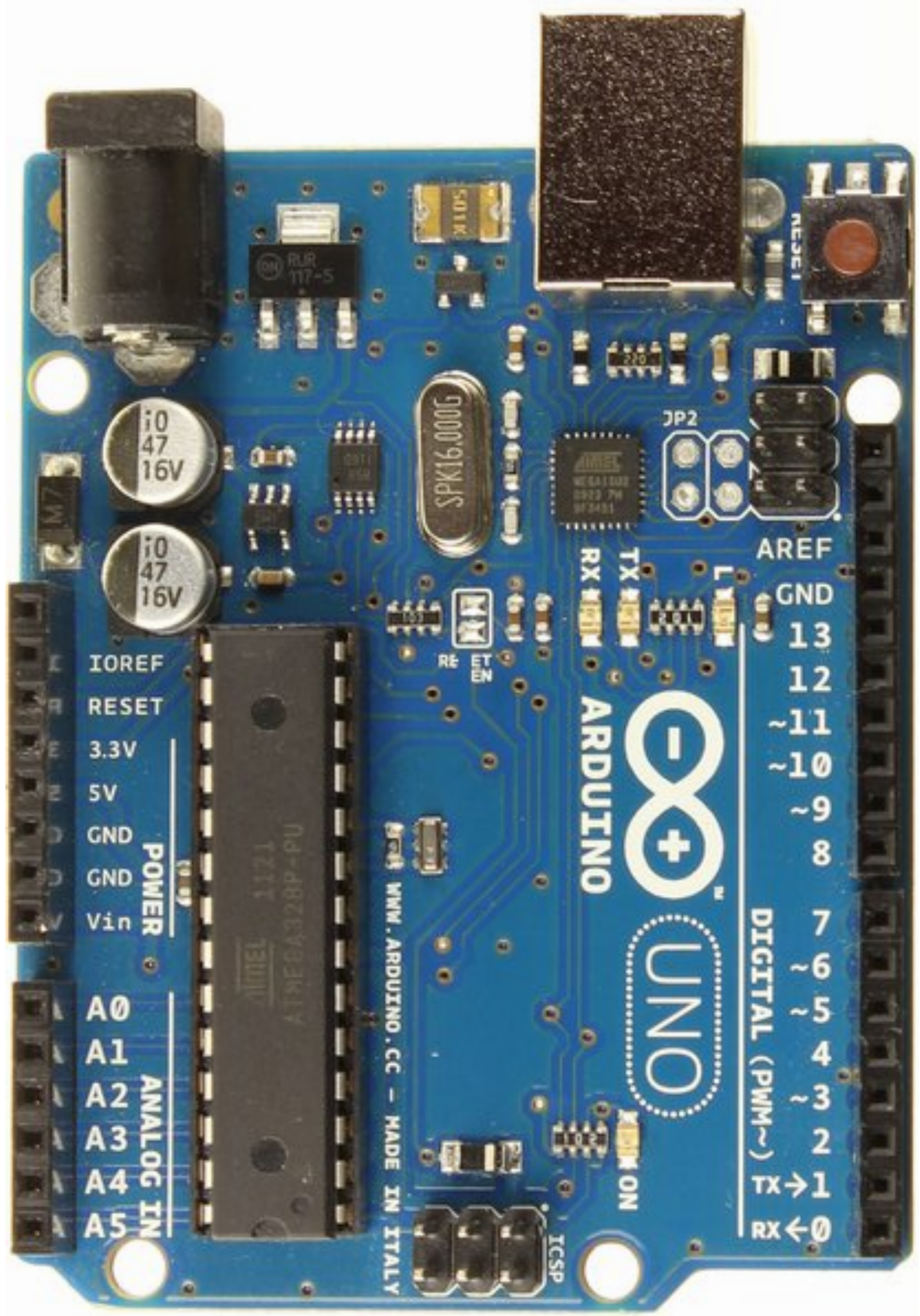
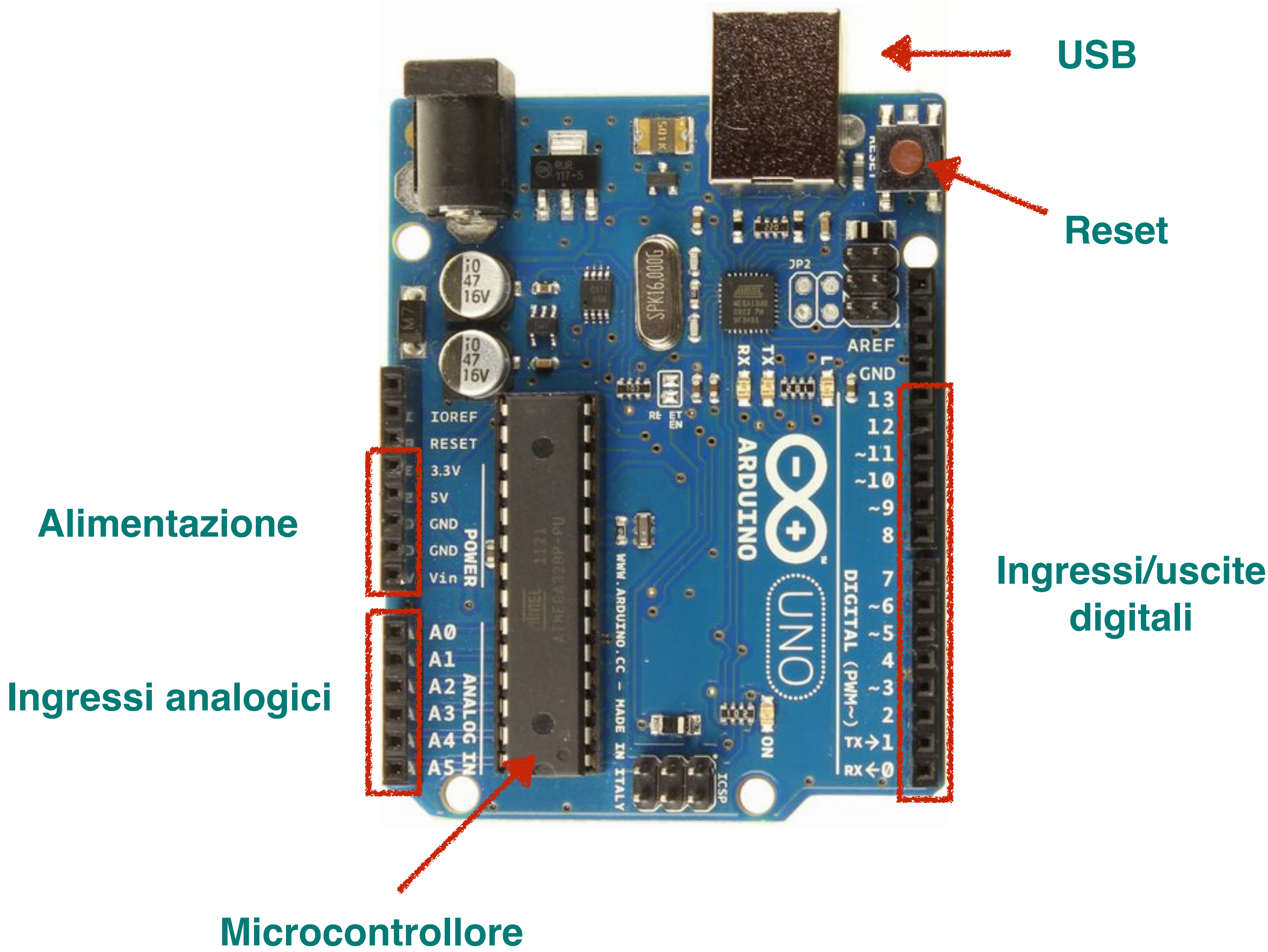
ARDUINO
DAY 2021

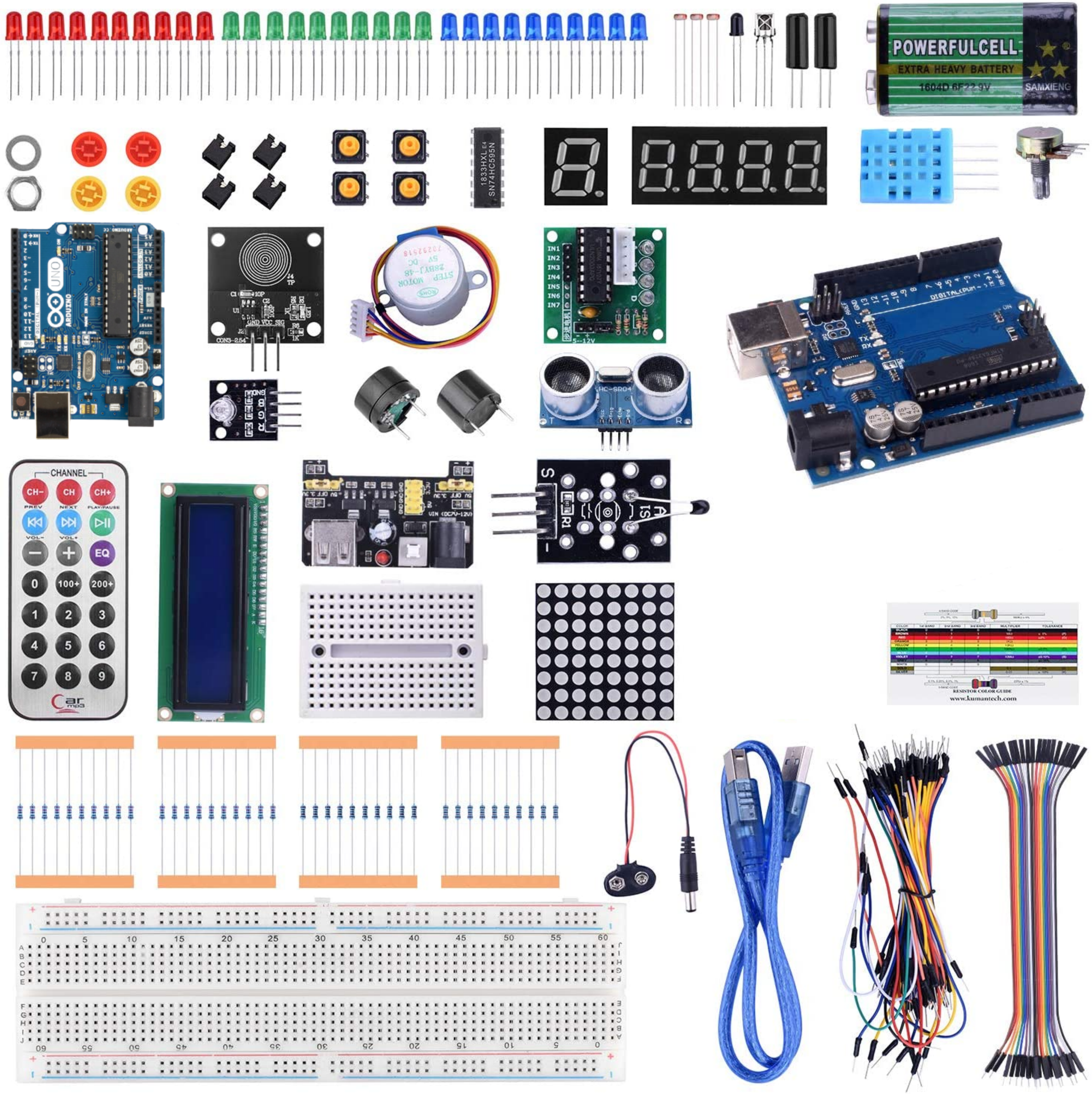
Programmazione con kit Arduino: introduzione per genitori con figli a casa

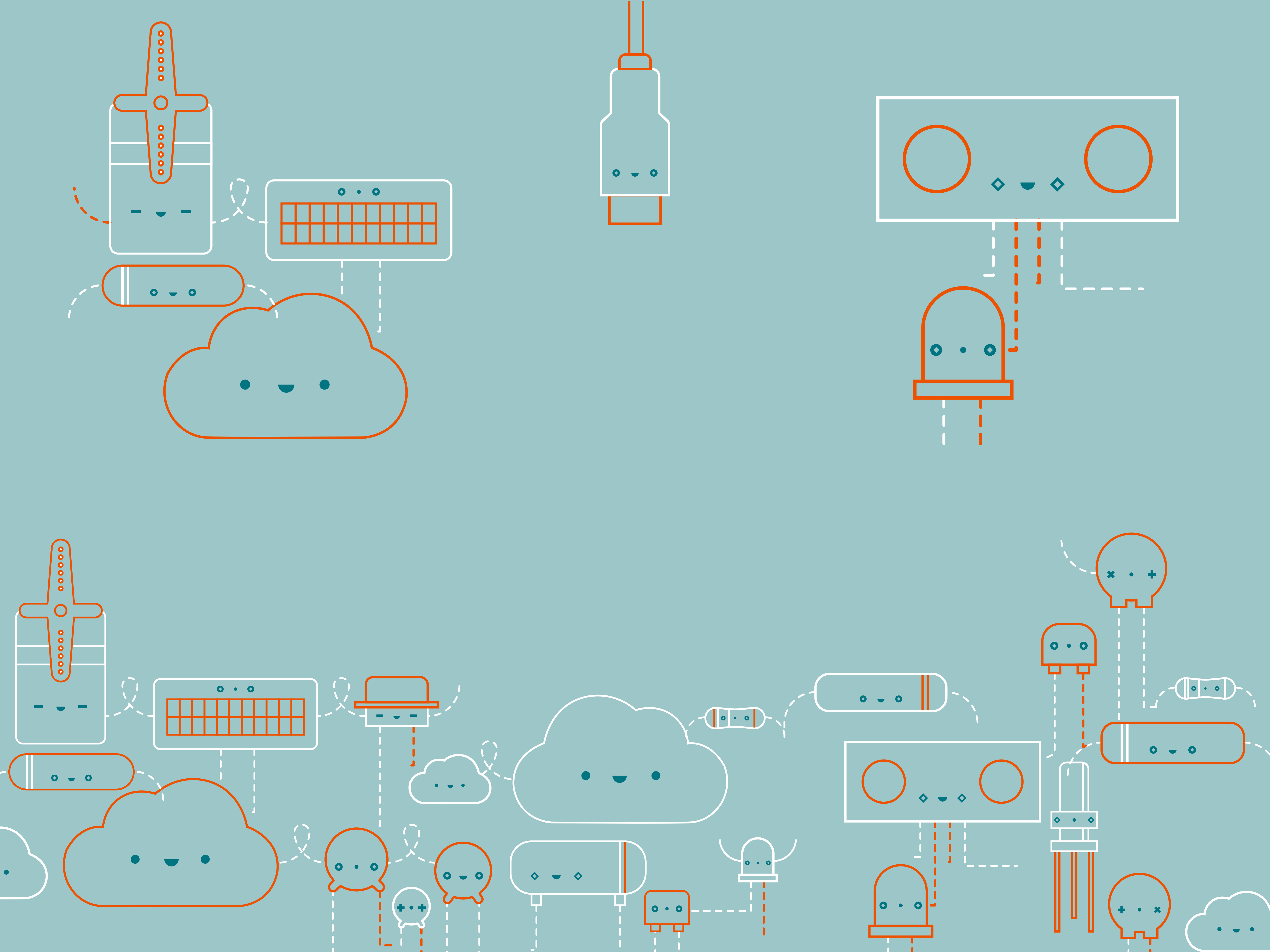
SciFabLab

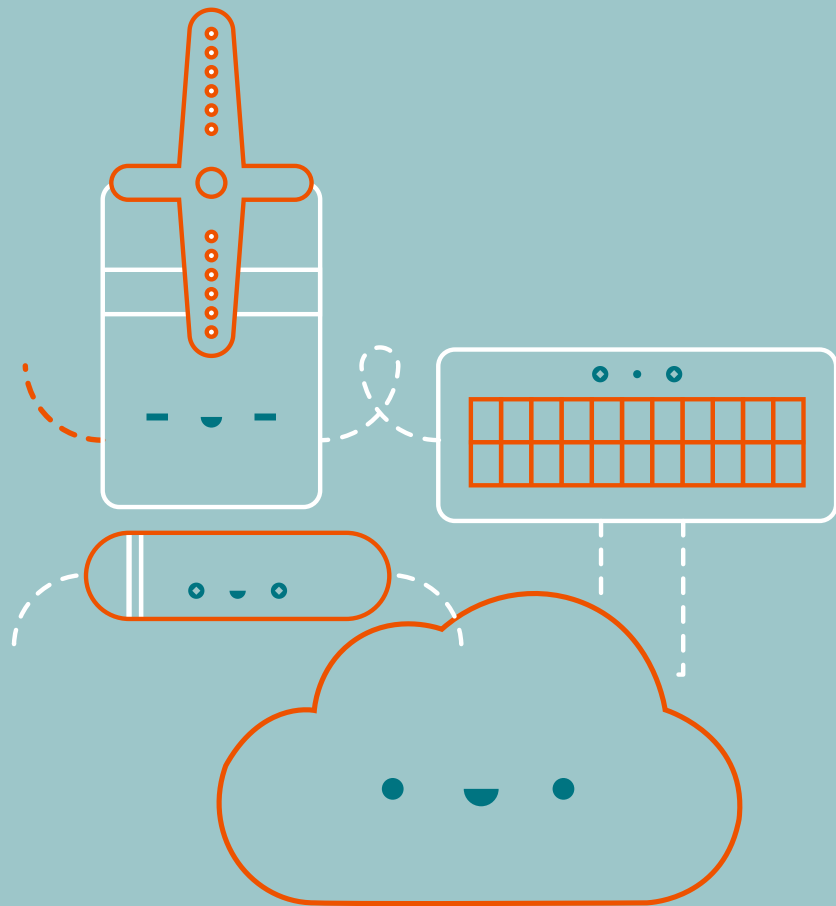
CHE È ?

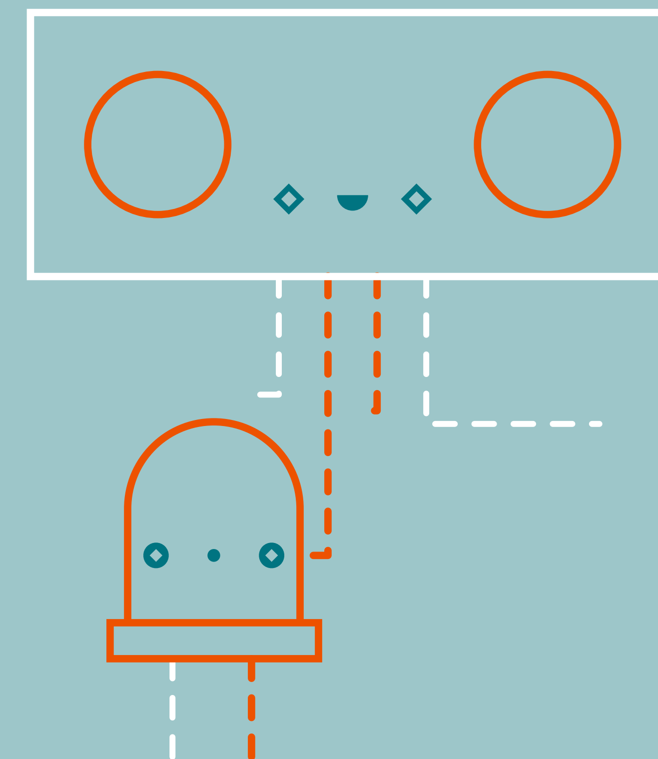

IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
ARDUINO
UNO
WWW.ARDUINO.CC - MADE IN ITALY
ICSP
ON
RX
TX
L
JP2
SPK16.000G


USB
Reset
Alimentazione
Ingressi analogici
Ingressi/uscite
digitali
Microcontrollore
ARDUINO
UNO

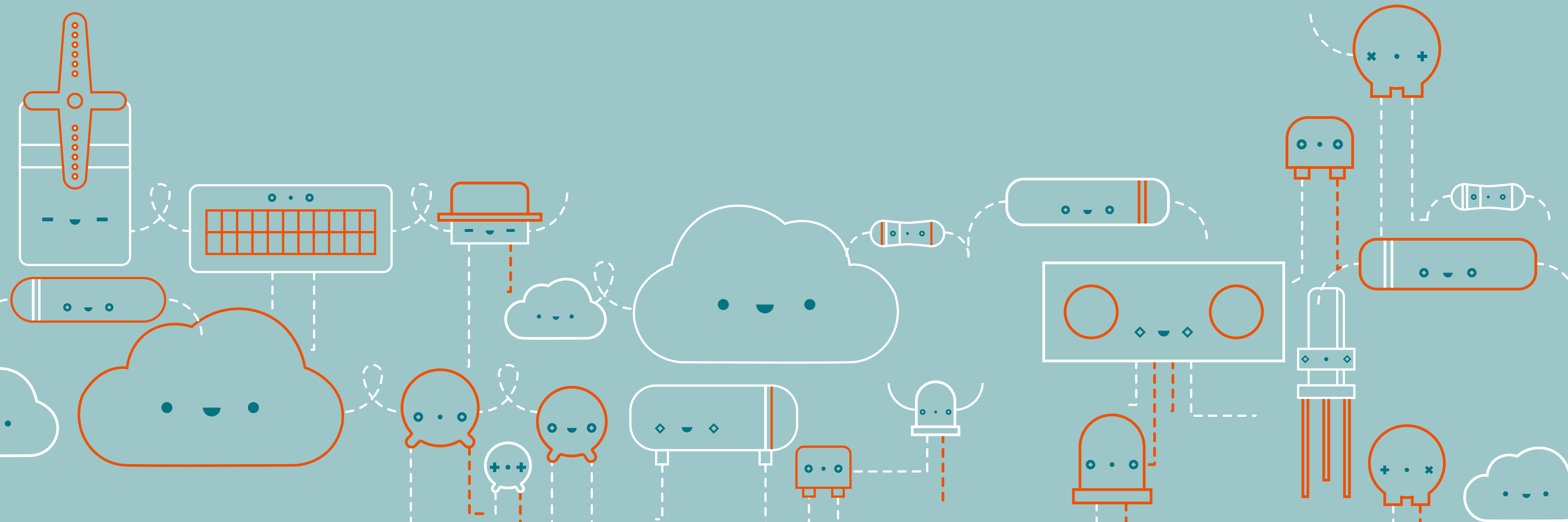

POWERFULCELL
EXTRA HEAVY BATTERY
SAMXIENG
CHANNEL
CH-
CH
CH+
EQ
100+
200+
HC-SR04
www.kumantech.com







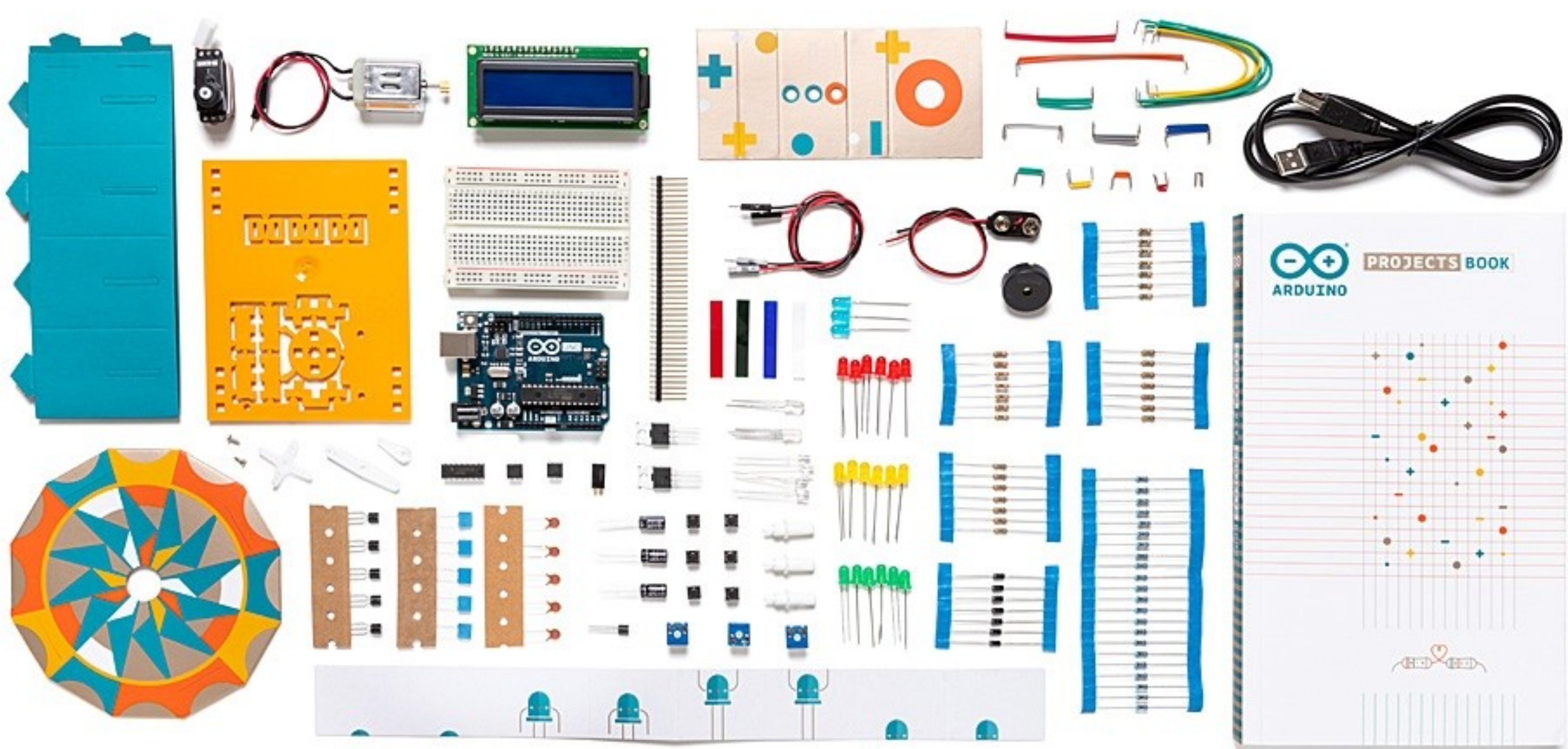
KIT



Arduino

https://store.arduino.cc/

- **80€** (Arduino store)
- **Manuale in italiano** (inglese, spagnolo, francese, tedesco, cinese, coreano, arabo)
- Non disponibile per il download, ma sono presenti i video tutorial
  e il codice dei singoli progetti
  https://www.youtube.com/playlist?list=PLT6rF_I5kknPf2qlVFlvH47qHvqvzkknd



BUILD THE CIRCUIT

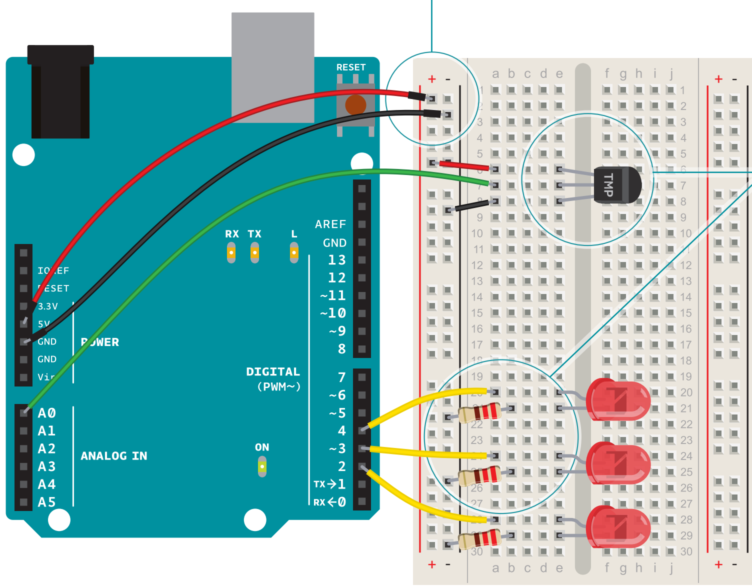


Fig. 2

1. Just as you've been doing in the earlier projects, wire up your breadboard so you have power and ground.

2. Attach the cathode (short leg) of each of the LEDs you're using to ground through a 220-ohm resistor. Connect the anodes of the LEDs to pins 2 through 4. These will be the indicators for the project.

3. Place the TMP36 on the breadboard with the rounded part facing away from the Arduino (the order of the pins is important!) as shown in Fig. 2. Connect the left pin of the flat facing side to power, and the right pin to ground. Connect the center pin to pin A0 on your Arduino. This is analog input pin 0.

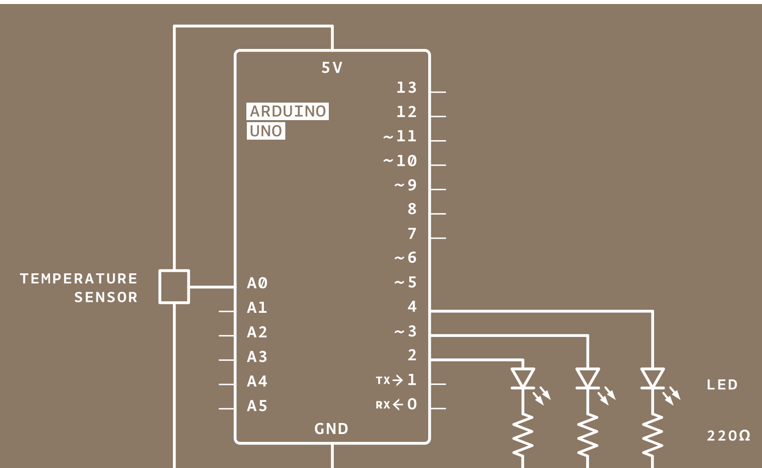
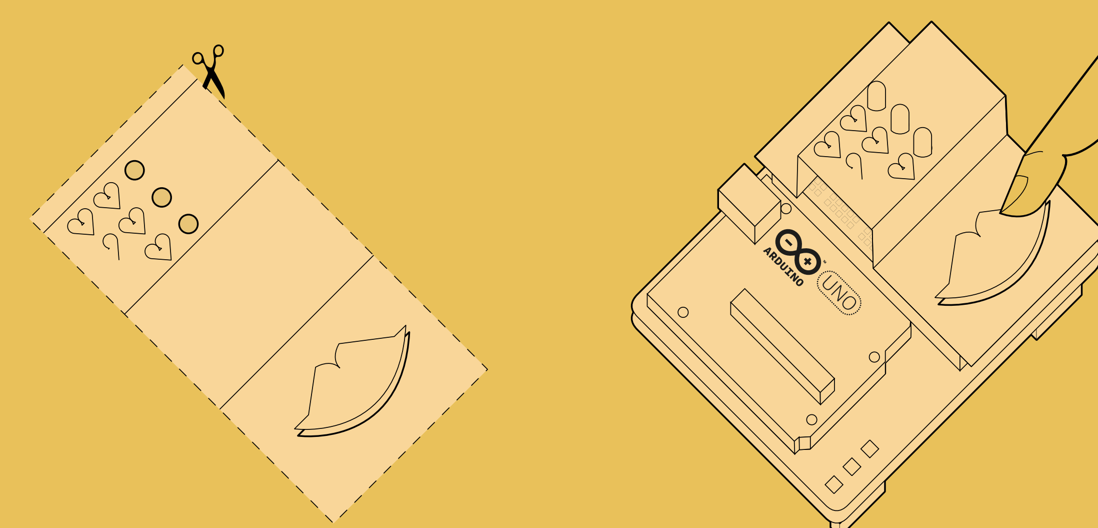


Fig. 3

Create an interface for your sensor for people interact with. A paper cutout in the shape of a hand is a good indicator. If you're feeling lucky, create a set of lips for someone to kiss, see how well that lights things up! You might also want to label the LEDs to give them some meaning. Maybe one LED means you're a cold fish, two LEDs means you're warm and friendly, and three LEDs means you're too hot to handle!

In this project, you need to check the ambient temperature of the room before proceeding. You're checking things manually right now, but this can also be accomplished through calibration. It's possible to use a button to set the baseline temperature, or to have the Arduino take a sample before starting the `loop()` and use that as the reference point. Project 6 gets into details about this, or you can look at the Calibration example that comes bundled with the Arduino software: *arduino.cc/calibration*



1. Cut out a piece of paper that will fit over the breadboard. Draw a set of lips where the sensor will be, and cut some circles for the LEDs to pass through.

2. Place the cutout over the breadboard so that the lips cover the sensor and the LEDs fit into the holes. Press the lips to see how hot you are!

THE CODE

A pair of useful constants

Constants are similar to variables in that they allow you to uniquely name things in the program, but unlike variables they cannot change. Name the analog input for easy reference, and create another named constant to hold the baseline temperature. For every 2 degrees above this baseline, an LED will turn on.
You've already seen the int datatype, used here to identify which pin the sensor is on. The temperature is being stored as a *float*, or floating-point number. This type of number has a decimal point, and is used for numbers that can be expressed as fractions.

```
const int sensorPin = A0;
const float baselineTemp = 20.0;
```

Initialize the serial port to the desired speed

In the setup you're going to use a new command, `Serial.begin()`. This opens up a connection between the Arduino and the computer, so you can see the values from the analog input on your computer screen.
The argument `9600` is the speed at which the Arduino will communicate, 9600 bits per second. You will use the Arduino IDE's serial monitor to view the information you choose to send from your microcontroller. When you open the IDE's serial monitor verify that the baud rate is 9600.

```
void setup(){
  Serial.begin(9600); // open a serial port
```

Initialize the digital pin directions and turn off

Next up is a `for()` loop to set some pins as outputs. These are the pins that you attached LEDs to earlier. Instead of giving them unique names and typing out the `pinMode()` function for each one, you can use a `for()` loop to go through them all quickly. This is a handy trick if you have a large number of similar things you wish to iterate through in a program. Tell the `for()` loop to run through pins 2 to 4 sequentially.

```
  for(int pinNumber = 2; pinNumber<5; pinNumber++){
    pinMode(pinNumber,OUTPUT);
    digitalWrite(pinNumber, LOW);
  }
}
```

`for()` loop tutorial
arduino.cc/for

Read the temperature sensor

In the `loop()`, you'll use a local variable named `sensorVal` to store the reading from your sensor. To get the value from the sensor, you call `analogRead()` that takes one argument: what pin it should take a voltage reading on. The value, which is between 0 and 1023, is a representation of the voltage on the pin.

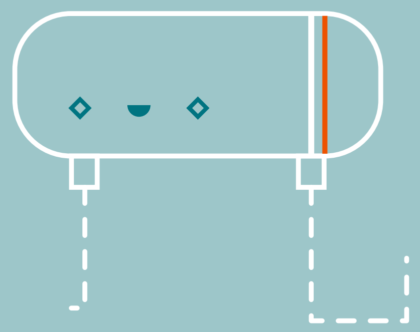
```
void loop(){
  int sensorVal = analogRead(sensorPin);
```

Send the temperature sensor values to the computer

The function `Serial.print()` sends information from the Arduino to a connected computer. You can see this information in your serial monitor. If you give `Serial.print()` an argument in quotation marks, it will print out the text you typed. If you give it a variable as an argument, it will print out the value of that variable.

```
  Serial.print("Sensor Value: ");
  Serial.print(sensorVal);
```

- Applicazioni pratiche dei singoli progetti
- Materiale aggiuntivo (cartoncini) per rendere i progetti più interattivi
- Componenti di qualità
- Manuale stampato (seppur di difficile lettura)
- Codice ampiamente commentato
- Libro resta difficilmente aperto, rende difficoltoso montaggio del progetto



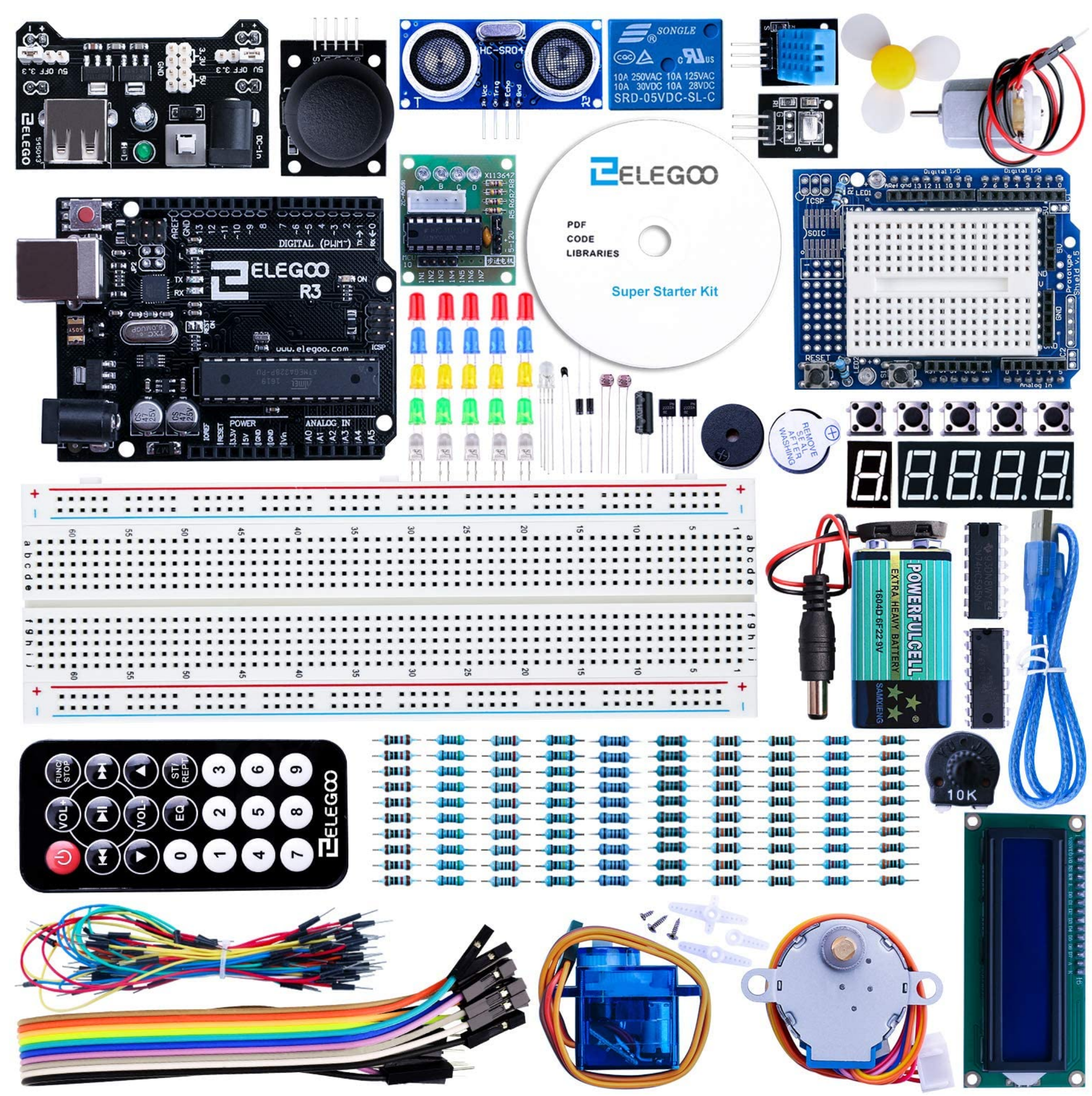
Elegoo

www.elegoo.com

37€ (Amazon)

Manuale in italiano (inglese, spagnolo, francese, tedesco, giapponese)

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial



Introduzione

In questa lezione imparerai come misurare l' intensità della luce utilizzando un input analogico. Nella lezione costruirai un circuito in grado di usare il livello della luce per controllare la quantità di LED che si accenderanno. La fotocellula sarà situata nella parte inferiore della breadboard ad ha un funzionamento simile ad un potenziometro.

Componenti Richiesti:

(1) x Elegoo UNO R3

(1) x Breadboard con 830 punti di collegamento

(8) x LED

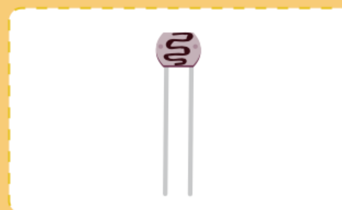
(8) x Resistenze da 220 ohm

(1) x Resistenze da 1k ohm

(1) x Circuito integrato 74hc595

(1) x Fotoresistore (Fotocellula)

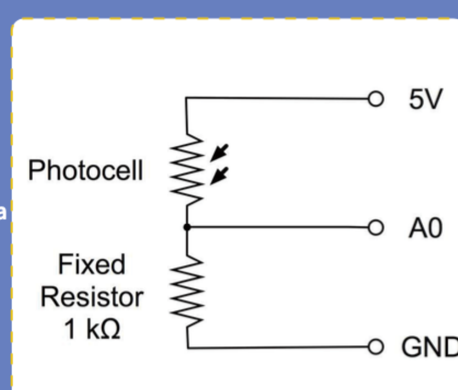
(16) x M-M Cavetti (Cavetti di collegamento Maschio - Maschio)

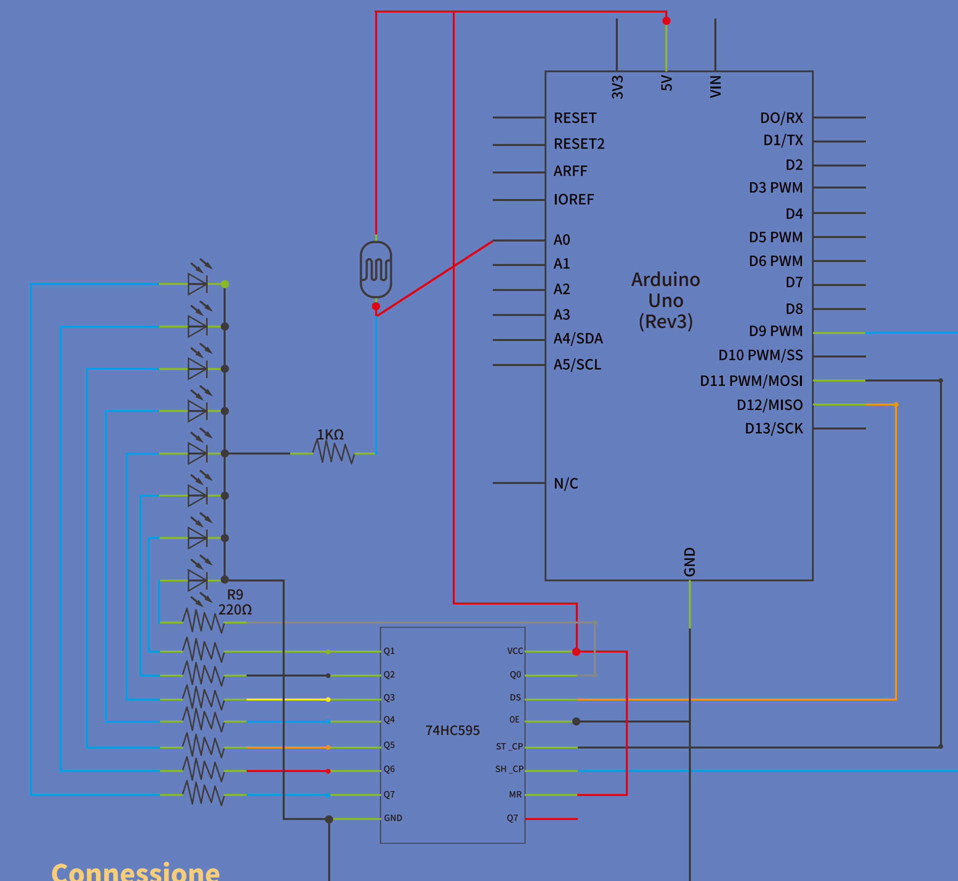


Introduzione al componente

Fotocellula:

- La fotocellula utilizzata è un tipo di resistore dipendente dalla luce, spesso abbreviato con LDR. Come suggerisce il nome questi componenti agiscono similmente a dei resistori, fatta eccezione per il fatto che la resistenza cambia in base alla quantità di luce che la illumina.
- La fotocellula che utilizzeremo ha una resistenza di circa 50 kΩ al buio e circa 500 Ω quando sottoposta ad alta luminosità.
- Per convertire questo valore di resistenza variabile in qualcosa che possiamo misurare con un input analogico nella nostra scheda UNO R3 misureremo il voltaggio.
- Il modo più semplice di fare questa cosa è combinare il fotoresitore con un resistore fisso.
- Il resistore e la fotocellula insieme agiscono come un potenziometro. Quando la luce è intensa la resistenza del fotoresistore è molto bassa se comparata con il valore della resistenza fissa, in questo caso è come se il potenziometro fosse girato verso il massimo.
- Quando la fotocellula è sottoposta ad una luce molto bassa o nulla, la resistenza sarà molto più alta rispetto alla seconda resistenza fissa da 1 kΩ, così sarà come se il potenziometro sia girato verso la messa a terra GND.
- Carica il codice che ti abbiamo fornito e prova a coprire la fotocellula con il dito, dopodiché avvicinala ad una sorgente di luce.





Connessione Schema

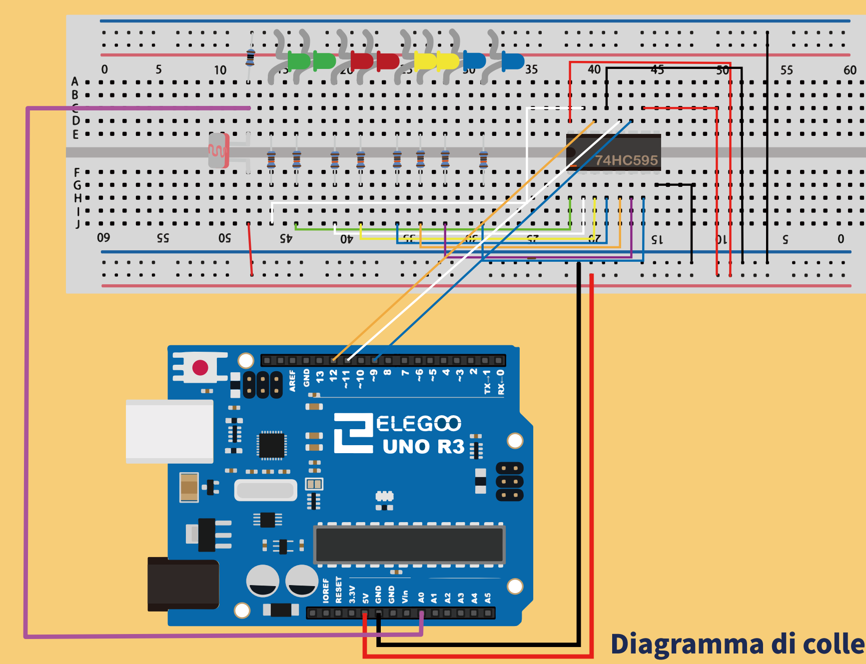


Diagramma di collegamento

```
//www.elegoo.com
//2016.12.9

int lightPin = 0;
int latchPin = 11;
int clockPin = 9;
int dataPin = 12;

int leds = 0;

void setup()
{
  pinMode(latchPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
}
/*   The most common method of using 74CH595
 *   lctchPin->LOW : Begin transmitting signals.
 *   shiftOut(dataPin, clockPin, bitOrder, value)
 *   dataPin: the pin on which to output each bit. Allowed data types: int.
 *   clockPin: the pin to toggle once the dataPin has been set to the correct
value. Allowed data types: int.
 *   bitOrder: which order to shift out the bits; either MSBFIRST or LSBFIRST.
(Most Significant Bit First, or, Least Significant Bit First).
 *   value: the data to shift out. Allowed data types: byte.
 *   lctchPin->HIch : The end of the transmission signal.
*/
void updateShiftRegister()
{
   digitalWrite(latchPin, LOW);
   shiftOut(dataPin, clockPin, LSBFIRST, leds);
   digitalWrite(latchPin, HIGH);
}
void loop()
{
  int reading  = analogRead(lightPin);
  int numLEDSLit = reading / 57;  //1023 / 9 / 2
  if (numLEDSLit > 8) numLEDSLit = 8;
  leds = 0;   // no LEDs lit to start
  for (int i = 0; i < numLEDSLit; i++)
  {
    leds = leds + (1 << i);  // sets the i'th bit
  }
  updateShiftRegister();
}
```

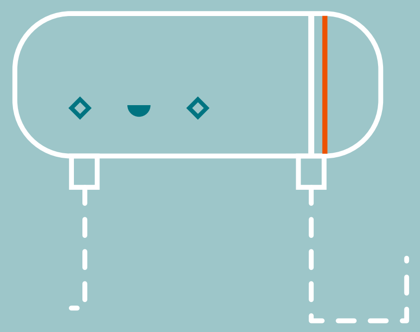
Codice

- programma.Se hai dubbio riguardo all' uploading del programma o se riscontri qualche errore durante il caricamento torna a vedere la lezione 5 nella parte 1
- La prima cosa da notare è che abbiamo cambiato il nome del pin analogico chiamandolo 'lightPin' al posto di 'potPin' dato che non avremo un potenziometro collegato.
- L' unico cambiamento sostanziale al codice è la linea che calcola quanti led vengono accesi.

```
int numLEDSLit = reading / 57;  // all LEDs lit at 1k
```

- Questa volta, dividiamo il valore grezzo letto per 57 invece che per 114. In altre parole lo dividiamo per la metà rispetto a quanto abbiamo fatto con il potenziometro, vogliamo infatti dividere per nove zone il range di valori, il primo corrisponderà a tutti i led spenti e l' ultimo corrisponderà a tutti i led illuminati. Questo fattore è dovuto alla resistenza fissa da 1 kΩ. Questo perché quando la fotocellula ha una resistenza pari a 1 kΩ (la stessa del resistore fisso), la lettura grezza sarà 1023 / 2 = 511. Questo significa che tutti i led vengono accesi e quindi numLEDSlit sarà pari a 8.

- Miglioramento istruzioni rispetto all’altra versione
- Codice commentato molto bene, con la spiegazione dei comandi
- Non ha un riscontro visivo dei componenti, più scomodo per un pubblico non esperto
- Non fornisce esempi di applicazioni pratiche o varianti di progetto



DF Robot

https://www.dfrobot.com

58€ (RS components)

Manuale in inglese

https://github.com/
DFRobot/Beginner-Kit-for-Arduino

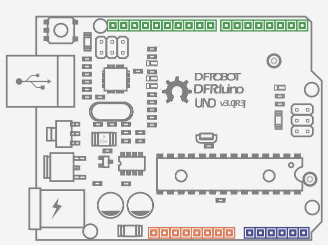
DFROBOT
DRIVE THE FUTURE

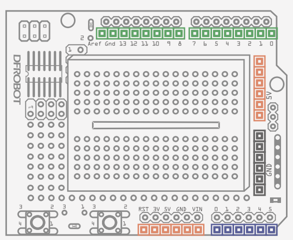
Light Sensitive LED

Let's introduce a new sensor component: the photo diode. In simple terms, when the sensor detects light, its resistance changes. The stronger light in the surrounding environment, the lower the resistance value the photo diode will read. By reading the photo diode's resistance value, we can work out the ambient lighting in an environment. The photo diode provided in the starter kit is a typical light sensor.

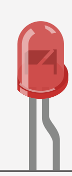
In this project, we will make an automatic light that can adjust itself according to the ambient lighting around it. When it is dark, the photo diode detects the change and triggers the light, and vice versa.

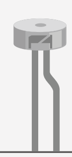
Components


DFRduino UNO R3 x1


Prototype Shield x1


Jumper Cables M/M x5


5MM LED x1


Ambient Light Sensor x1


Resistor 220R x1


Resistor 10K x1

Be aware that photo diodes are polarized, just like LEDs, so they will only work if connected the correct way around.

The photo diode has to be connected with a 10k resistor rather than a 220Ω resistor

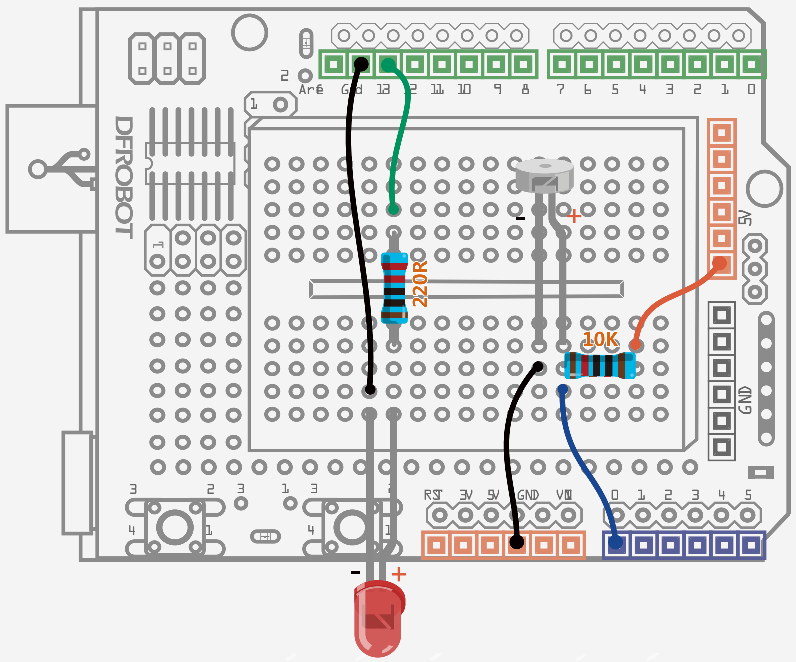


Diagram of the photo diode circuit

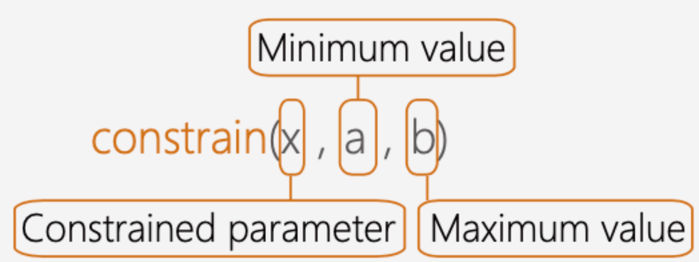
Sample code 9-1 :

```
// Project 9– Light the lamp
int LED = 13;                     //define LED digital pin 13
int val = 0;                      //define the voltage value of photo diode in
digital pin 0
void setup(){
   pinMode(LED,OUTPUT);           //  Configure LED as output mode
   Serial.begin(9600);            //Configure baud rate 9600
}

void loop(){
   val = analogRead(0);   // Read voltage value ranging from 0 -1023
   Serial.println(val);   // read voltage value from serial monitor
   if(val<1000){          // If lower than 1000, turn off LED
      digitalWrite(LED,LOW);
   }else{                         // Otherwise turn on LED
      digitalWrite(LED,HIGH);
   }
   delay(10);                     // delay for 1(
}
```

After uploading the code, you can shine a flashlight on the photodiode to alter the light levels in the environment. When it is dark, the should light up. When it is bright, the LED should turn off.

Here we will introduce "constrain()" and "random()". Do try to look them up with websites we mentioned in the last homework first and see if you can understand them.

The format of the constrain function is as follows:



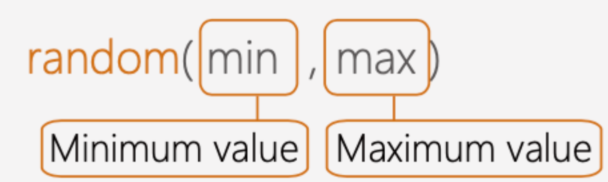
The "constrain()" function requires three parameters: x, a and b.

"x" is a constraint number here, "a" is the minimum, and "b" is the maximum.

If the value is less than "a", it will return to "a". If it is greater than "b", it will return to "b".

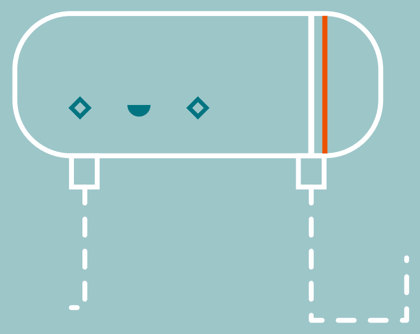
Red, green and blue are our constrained parameters. They are constrained between 0 and 255 (which falls into the range of PWM values). Values are generated at random using the "random()" function.

The format of "random()" is as below:



The first variable of this function is the minimum value and the second is the maximum. So we configure as "random(0,255)" in this program.

- Se si possiede il kit con lo shield originale, si seguono più facilmente gli schemi
- La mini breadboard rende difficoltoso il montaggio
- Elenco dei componenti chiaro, con riferimento visivo
- Codice commentato bene
- Spiegazione schematica del funzionamento dei componenti
- Proposte di modifiche di progetto (compiti per casa)



Sparkfun

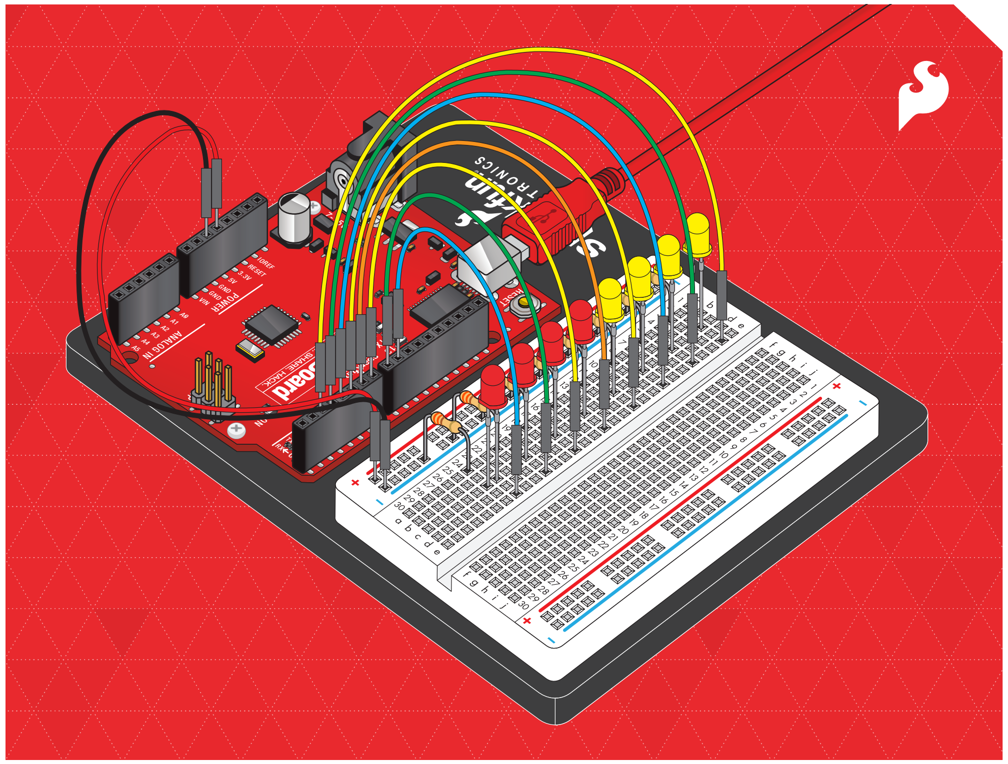
https://www.sparkfun.com

83€ (amazon)

Manuale in inglese

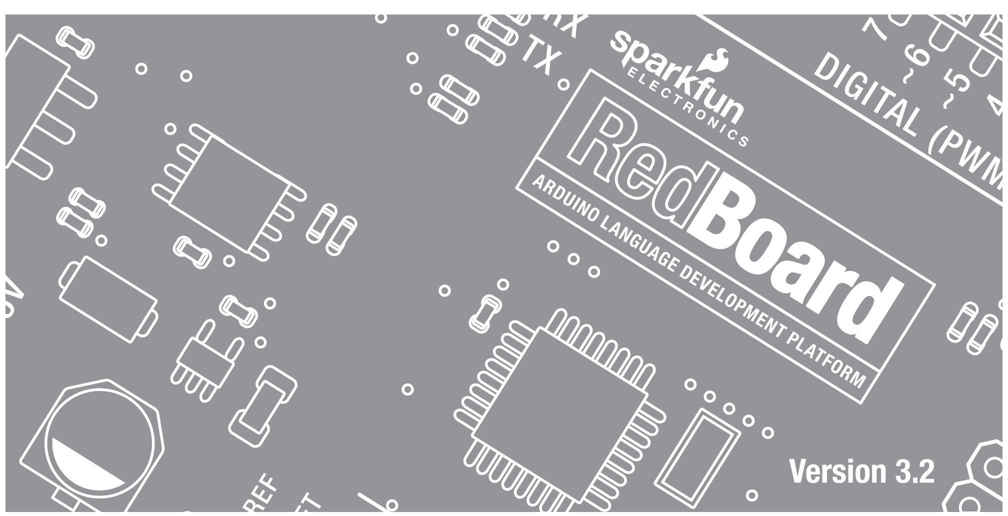
https://cdn.sparkfun.com/datasheets/Kits/RedBoard_SIK_3.2.pdf *(old)*

https://learn.sparkfun.com/tutorials/sparkfun-inventors-kit-experiment-guide---v41 *(new)*



SIK GUIDE

Your guide to the SparkFun Inventor's Kit for the SparkFun RedBoard

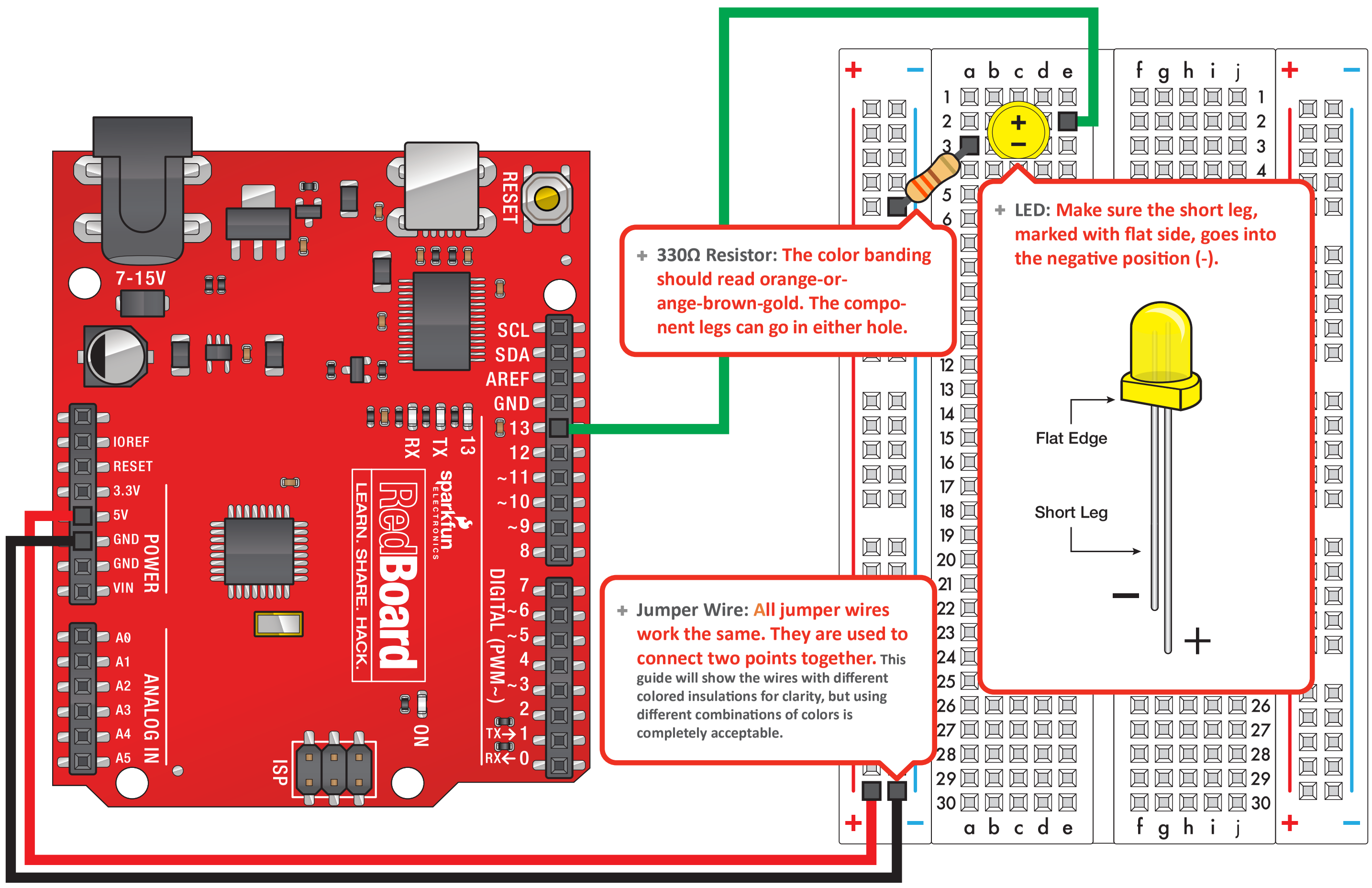


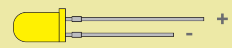
SparkFun Inventor's Kit

VERSION 4.1

Your Guide to the SIK
for the SparkFun RedBoard

sparkfun
START SOMETHING


330Ω Resistor: The color banding should read orange-orange-brown-gold. The component legs can go in either hole.
LED: Make sure the short leg, marked with flat side, goes into the negative position (-).
Flat Edge
Short Leg
Jumper Wire: All jumper wires work the same. They are used to connect two points together. This guide will show the wires with different colored insulations for clarity, but using different combinations of colors is completely acceptable.
RESET
7-15V
SCL
SDA
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
RX
TX
sparkfun
ELECTRONICS
RedBoard
LEARN. SHARE. HACK.
ON
ISP
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN

Component:	Image Reference:	RedBoard	Breadboard	
LED (5mm)			c2 + / c3 -	+ Components like LEDs are inserted into the breadboard sockets c2(long leg) c3(short leg). Steps highlighted with a yellow warning triangle represent a polarized component. Pay special attention to the component's markings indicating how to place it on the breadboard.
330Ω Resistor			a3 - -	+ Resistors are placed in breadboard sockets only. The "-" symbol represents any socket in its vertical column on the Power bus.
Jumper Wire		GND	-	+ "GND" on the RedBoard should be connected to the row marked "-" on the breadboard.
Jumper Wire		5V	+	+ "5V" on the RedBoard connects to the row marked "+" on the breadboard.
Jumper Wire		Pin 13	e2	+ "Pin 13" on the RedBoard connects to socket "e2" on the breadboard.

+ **RedBoard:** The red background represents a connection to one of the RedBoard header pins.

+ **Breadboard:** The white background represents a connection to a breadboard socket specified by a letter-number coordinate such as e2. These coordinates are merely suggestions that align with the graphic image.

Component:	Image Reference:	RedBoard	Breadboard	
LED (5mm) ⚠	+ -		c2 (+) c3 (-)	+ Components like LEDs are inserted into the breadboard sockets c2(long leg) c3(short leg). Steps highlighted with a yellow warning triangle represent a polarized component. Pay special attention to the component's markings indicating how to place it on the breadboard.
330Ω Resistor			a3 -	+ Resistors are placed in breadboard sockets only. The "-" symbol represents any socket in its vertical column on the Power bus.
Jumper Wire		GND	-	+ "GND" on the RedBoard should be connected to the row marked "-" on the breadboard.
Jumper Wire		5V	+	+ "5V" on the RedBoard connects to the row marked "+" on the breadboard.
Jumper Wire		Pin 13	e2	+ "Pin 13" on the RedBoard connects to socket "e2" on the breadboard.

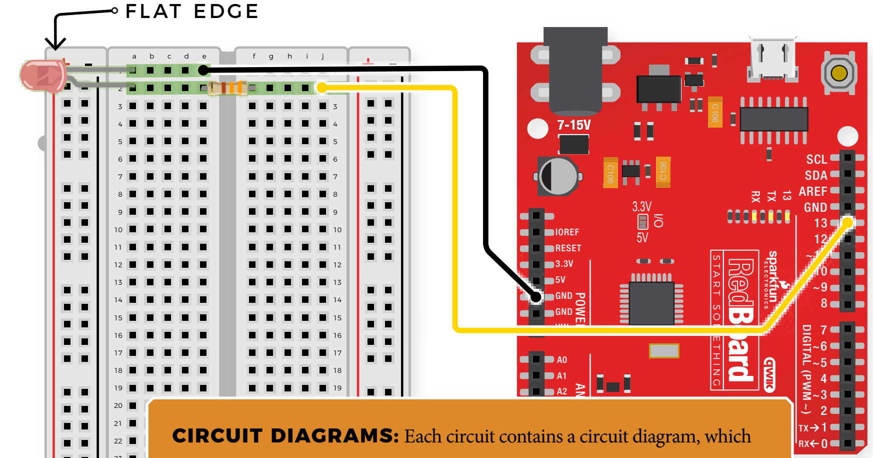
+ **RedBoard:** The red background represents a connection to one of the RedBoard header pins.

+ **Breadboard:** The white background represents a connection to a breadboard socket specified by a letter-number coordinate such as e2. These coordinates are merely suggestions that align with the graphic image.

HOOKUP GUIDE

READY TO START HOOKING EVERYTHING UP? Check out the circuit diagram and hookup table below to see how everything is connected.

FLAT EDGE



CIRCUIT DIAGRAMS: Each circuit contains a circuit diagram, which acts as a visual aid designed to make it easier for you to see how your circuit should be built. Each colored line represents a jumper wire connection in the circuit. All wires should have two connection points, which you also see in the hookup table below.

COLORS: Please note that while traditionally red is used for power and black is used for ground, all wires, no matter their color, function the same.

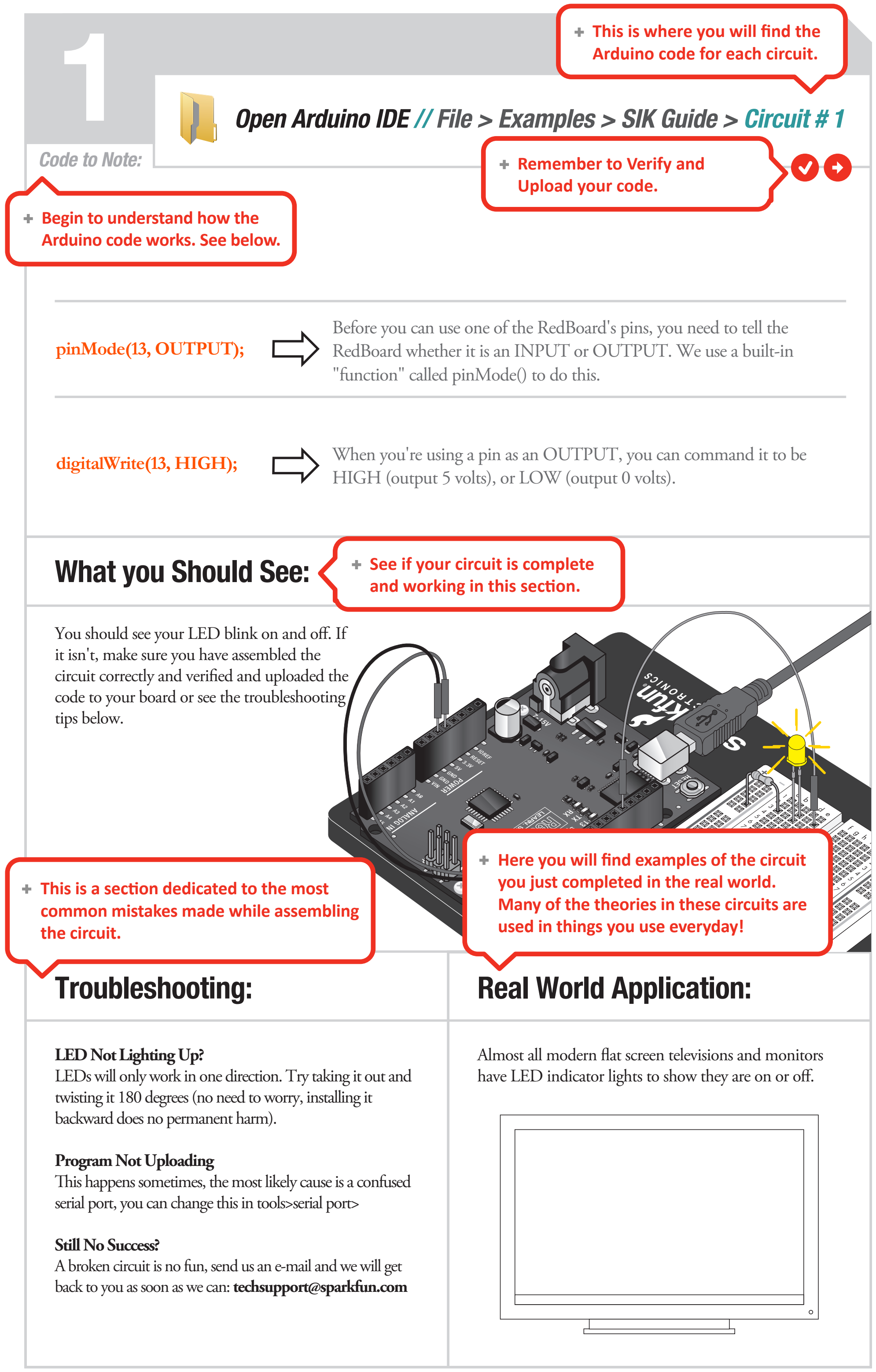
HOOKUP TABLES: Many electronics beginners find it helpful to have a coordinate system when building their circuits. For each circuit, you'll find a hookup table that lists the coordinates of each component or wire and where it connects to the RedBoard, the breadboard, or both. The breadboard has a letter/number coordinate system, just like the game Battleship.

◆ D13 to ■ J2

...means one end of a component connects to digital pin 13 on your RedBoard and the other connects to J2 on the breadboard

CONNECTION TYPES	◆ REDBOARD CONNECTION ■ BREADBOARD CONNECTION	
JUMPER WIRES	◆ D13 to ■ J2	◆ GND to ■ E1
LED	■ A1(-) to ■ A2(+)	
330Ω RESISTOR (ORANGE, ORANGE, BROWN)	■ E2 to ■ F2	

In this table, a yellow highlight indicates that a component has polarity and will only function if properly oriented.



+ This is where you will find the Arduino code for each circuit.

1

Open Arduino IDE // File > Examples > SIK Guide > Circuit # 1

Code to Note:

+ Remember to Verify and Upload your code.

+ Begin to understand how the Arduino code works. See below.

pinMode(13, OUTPUT);	Before you can use one of the RedBoard's pins, you need to tell the RedBoard whether it is an INPUT or OUTPUT. We use a built-in "function" called pinMode() to do this.
digitalWrite(13, HIGH);	When you're using a pin as an OUTPUT, you can command it to be HIGH (output 5 volts), or LOW (output 0 volts).

What you Should See:

+ See if your circuit is complete and working in this section.

You should see your LED blink on and off. If it isn't, make sure you have assembled the circuit correctly and verified and uploaded the code to your board or see the troubleshooting tips below.

+ This is a section dedicated to the most common mistakes made while assembling the circuit.

+ Here you will find examples of the circuit you just completed in the real world. Many of the theories in these circuits are used in things you use everyday!

Troubleshooting:

LED Not Lighting Up?
LEDs will only work in one direction. Try taking it out and twisting it 180 degrees (no need to worry, installing it backward does no permanent harm).

Program Not Uploading
This happens sometimes, the most likely cause is a confused serial port, you can change this in tools>serial port>

Still No Success?
A broken circuit is no fun, send us an e-mail and we will get back to you as soon as we can: techsupport@sparkfun.com

Real World Application:

Almost all modern flat screen televisions and monitors have LED indicator lights to show they are on or off.

Page 24

PROGRAM OVERVIEW

1	Turn the LED on by sending power (5V) to digital pin 13.
2	Wait 2 seconds (2000 milliseconds).
3	Turn the LED off by cutting power (0V) to digital pin 13.
4	Wait 2 seconds (2000 milliseconds).
5	Repeat.

ONBOARD LED PIN 13: You may have noticed a second, smaller LED blinking in unison with the LED in your breadboard circuit. This is known as the onboard LED, and you can find one on almost any Arduino or Arduino-compatible board. In most cases, this LED is connected to **digital pin 13 (D13)**, the same pin used in this circuit.

NEW IDEAS

CODE TO NOTE: The sketches that accompany each circuit introduce new programming techniques and concepts as you progress through the guide. The Code to Note section highlights specific lines of code from the sketch and explains them in greater detail.

CODE TO NOTE

SETUP AND LOOP:

`void setup(){}` & `void loop(){}`

Every Arduino program needs these two functions. Code that goes in between the curly brackets `{}` of `setup()` runs once. The code in between the `loop()` curly brackets `{}` runs over and over until the RedBoard is reset or powered off.

INPUT OR OUTPUT?:

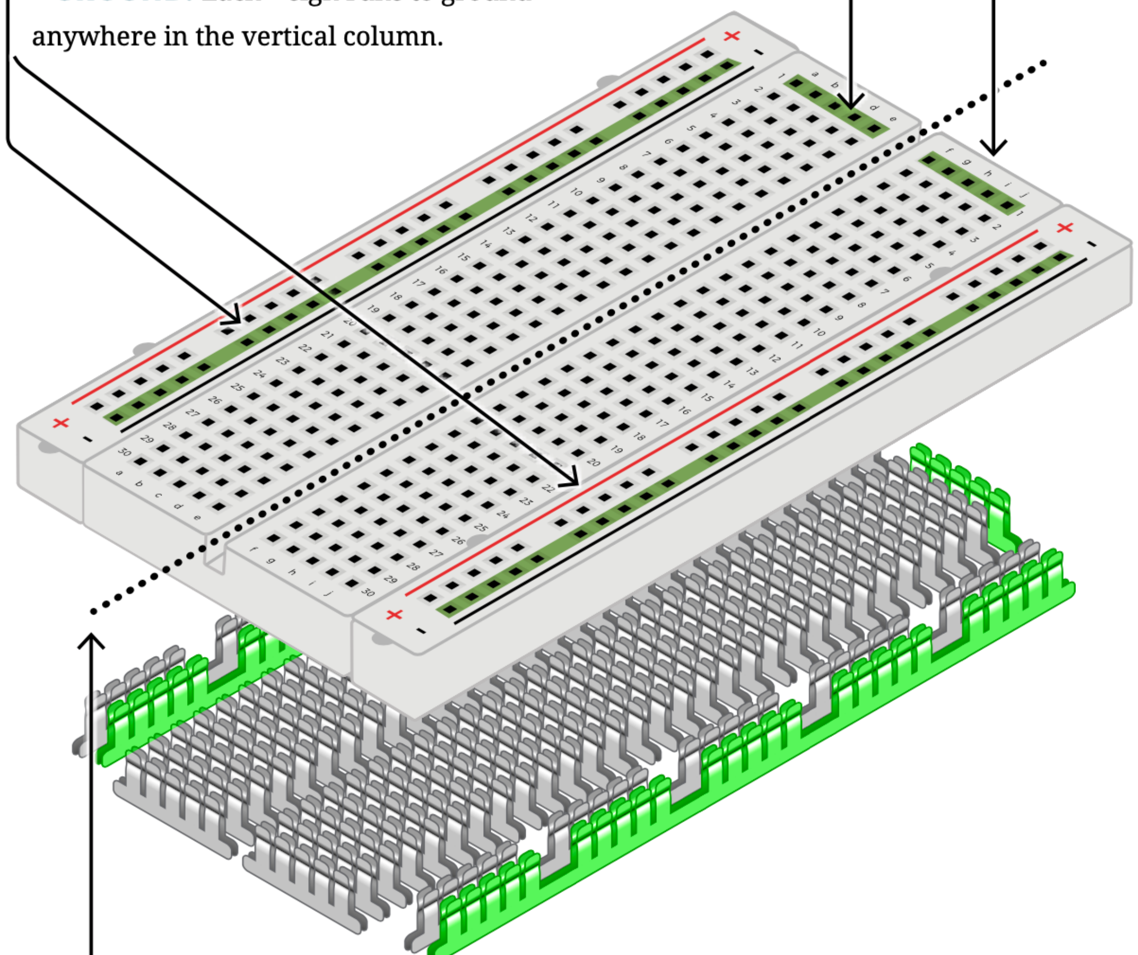
`pinMode(13, OUTPUT);`

Before you can use one of the digital pins, you need to tell the RedBoard whether it is an `INPUT` or `OUTPUT`. We use a built-in "function" called `pinMode()` to make pin 13 a digital output. You'll learn more about digital inputs in Project 2.

Components connected to a row will be connected to any other part inserted in the same row.

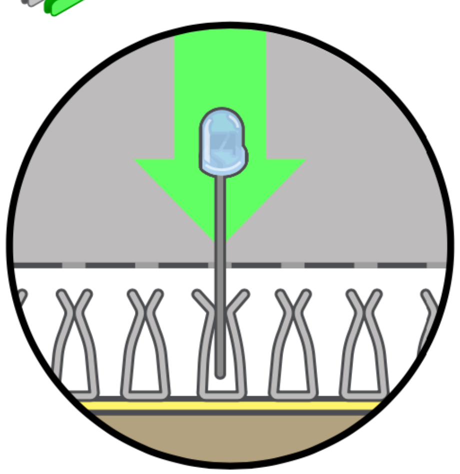
+ POWER: Each + sign runs power anywhere in the vertical column.

– GROUND: Each – sign runs to ground anywhere in the vertical column.



CENTERLINE

This line divides the breadboard in half, restricting electricity to one half or the other.

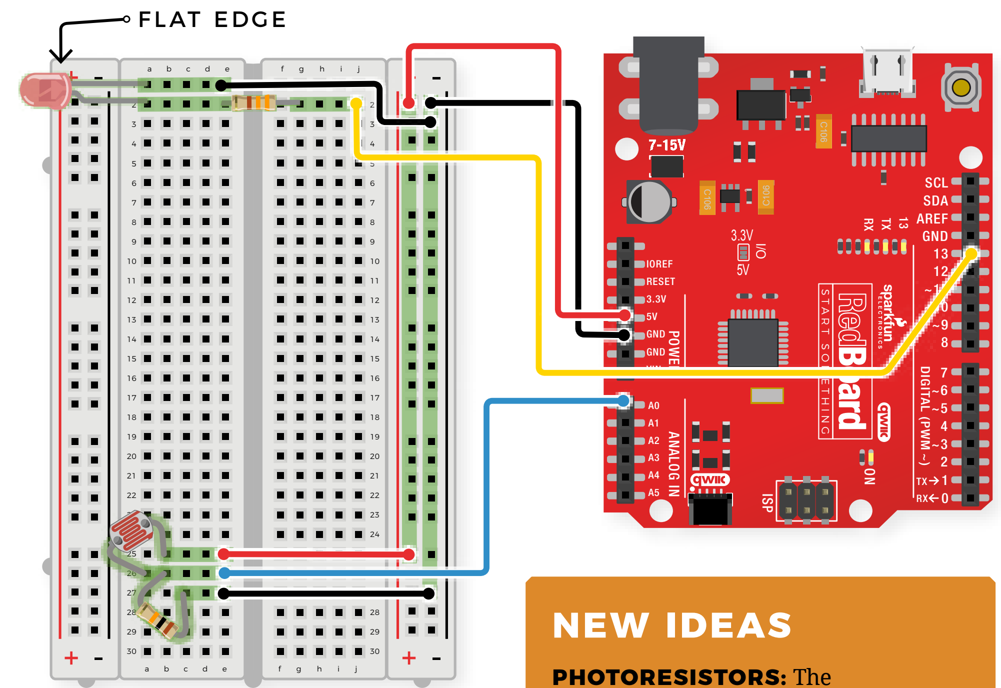


MAKING A CONNECTION

Most of the components in this kit are breadboard-friendly and can be installed and removed.

HOOKUP GUIDE

READY TO START HOOKING EVERYTHING UP? Check out the circuit diagram and hookup table below to see how everything is connected.

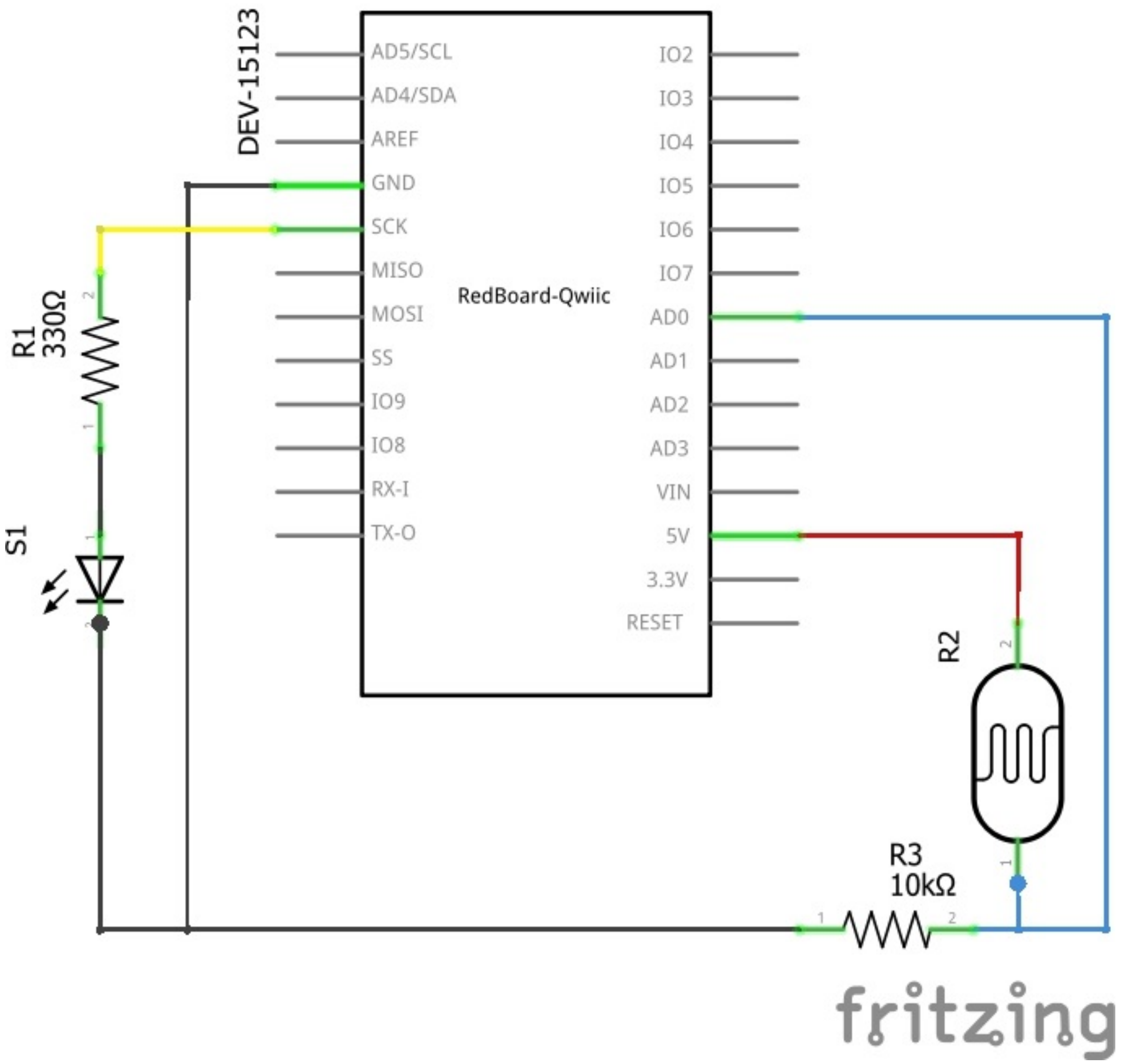


NEW IDEAS

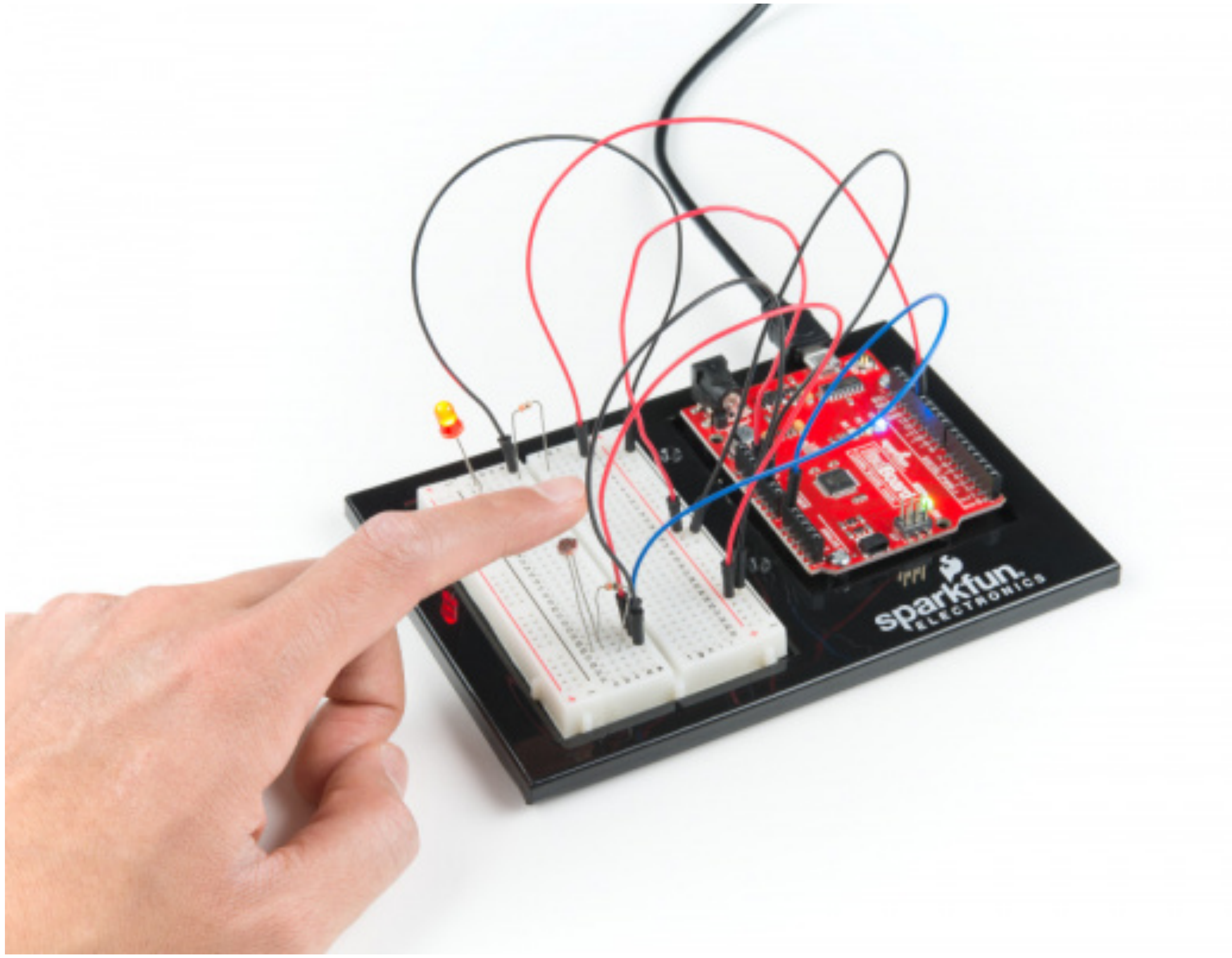
PHOTORESISTORS: The photoresistors, like regular resistors, are not polarized and can be installed in either direction.

CONNECTION TYPES ◆ REDBOARD CONNECTION ▪ BREADBOARD CONNECTION

JUMPER WIRES	◆ 5V to ▪ 5V(+); ◆ GND to ▪ GND (-); ◆ D13 to ▪ J2; ◆ A0 to ▪ E26; ▪ E1 to ▪ GND(-); ▪ E25 to ▪ 5V(+); ▪ E27 to ▪ GND(-)
LED	▪ A1(-) to ▪ A2(+)
330Ω RESISTOR (ORANGE, ORANGE, BROWN)	▪ E2 to ▪ G2
10KΩ RESISTOR (BROWN, BLACK, ORANGE)	▪ B26 to ▪ C27
PHOTORESISTOR	▪ A26 to ▪ B25



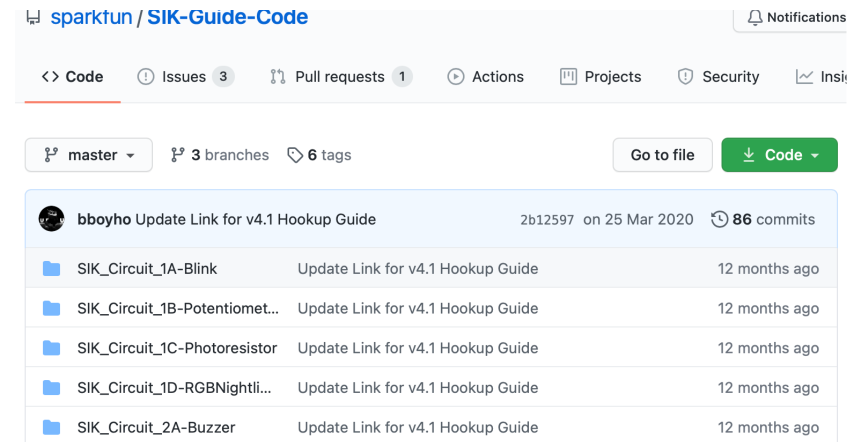
DEV-15123
RedBoard-Qwiic
AD5/SCL
AD4/SDA
AREF
GND
SCK
MISO
MOSI
SS
IO9
IO8
RX-I
TX-O
IO2
IO3
IO4
IO5
IO6
IO7
AD0
AD1
AD2
AD3
VIN
5V
3.3V
RESET
R1
330Ω
S1
R2
R3
10kΩ
fritzing



sparkfun
ELECTRONICS

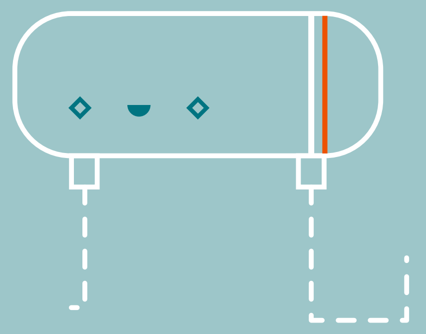
Code

https://github.com/sparkfun/SIK-Guide-Code




```
/*
  SparkFun Inventor’s Kit
  Circuit 1C-Photoresistor

  Use a photoresistor to monitor how bright a room is, and turn an LED on when it gets dark.

  This sketch was written by SparkFun Electronics, with lots of help from the Arduino community.
  This code is completely free for any use.

  View circuit diagram and instructions at: https://learn.sparkfun.com/tutorials/sparkfun-inventors-kit-experiment-guide---v41
  Download drawings and code at: https://github.com/sparkfun/SIK-Guide-Code
*/

int photoresistor = 0;              //this variable will hold a value based on the brightness of the ambient light
int threshold = 750;                //if the photoresistor reading is below this value the the light will turn on

void setup()
{
  Serial.begin(9600);               //start a serial connection with the computer

  pinMode(13, OUTPUT);              //set pin 13 as an output that can be set to HIGH or LOW
}

void loop()
{
  //read the brightness of the ambient light
  photoresistor = analogRead(A0);   //set photoresistor to a number between 0 and 1023 based on how bright the ambient light is
  Serial.println(photoresistor);    //print the value of photoresistor in the serial monitor on the computer

  //if the photoresistor value is below the threshold turn the light on, otherwise turn it off
  if (photoresistor < threshold) {
    digitalWrite(13, HIGH);         // Turn on the LED
  } else {
    digitalWrite(13, LOW);          // Turn off the LED
  }

  delay(100);                       //short delay to make the printout easier to read
}
```

- Troubleshooting
- Applicazioni dei progetti nella vita reale
- Molto chiaro e meticoloso
- Codice richiamabile già all’interno dell’IDE di Arduino tramite libreria
- Elenco dei componenti visivo e testuale
- Le schede sono impaginate in modo da essere seguite facilmente durante il montaggio



Oomlout
http://oomlout.com

Non più disponibile il kit

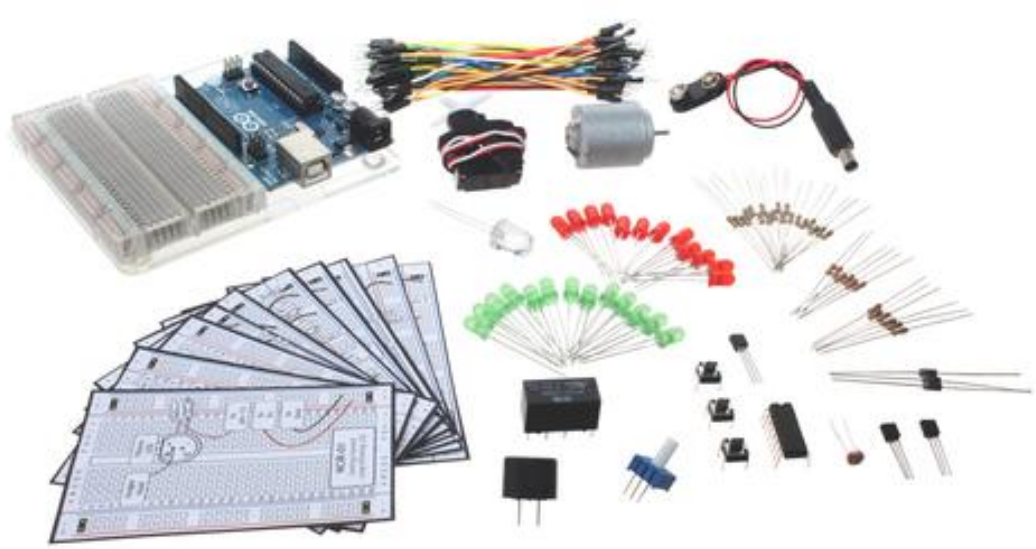
Versione ripresa ed estesa della Adafruit 85$ (**MetroX**)

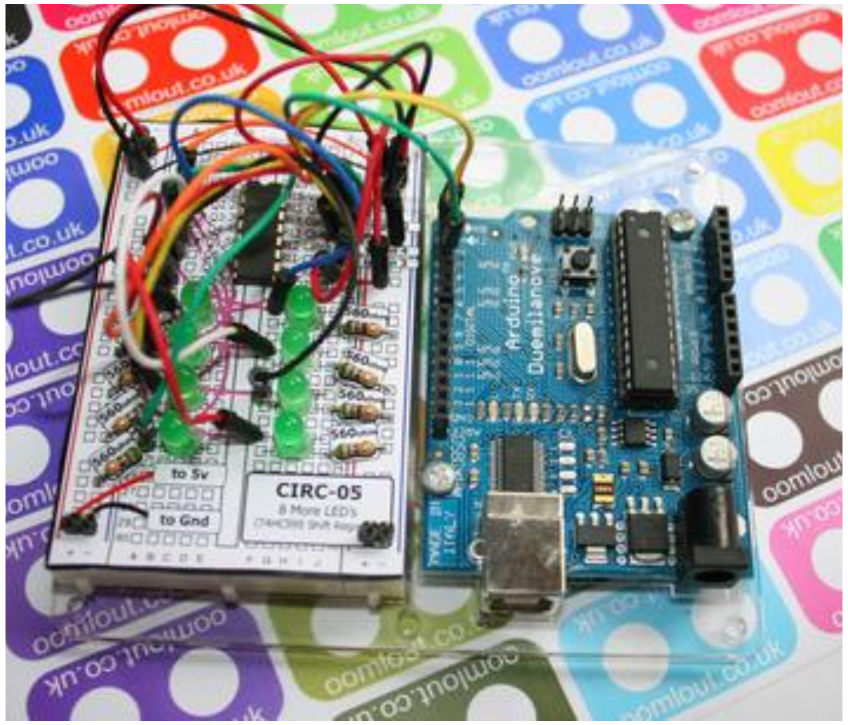
https://learn.adafruit.com/experimenters-guide-for-metro

Manuale in inglese

http://ardx.org/src/guide/2/ARDX-EG-OOML-WEB.pdf

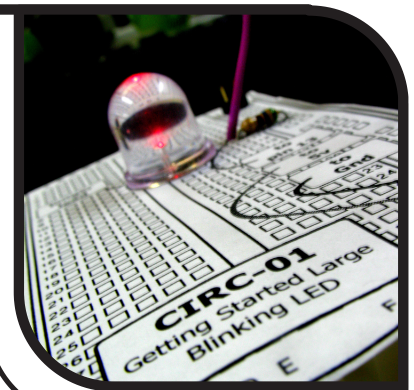
http://ardx.org/src/





CIRC-01

.:Getting Started:.
.:(Blinking LED):.



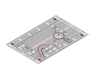
What We're Doing:

LEDs (light emitting diodes) are used in all sorts of clever things which is why we have included them in this kit. We will start off with something very simple, turning one on and off, repeatedly, producing a pleasant blinking effect. To get started, grab the parts listed below, pin the layout sheet to your breadboard and then plug everything in. Once the circuit is assembled you'll need to upload the program. To do this plug the Arduino board into your USB port. Then select the proper port in **Tools > Serial Port > (the comm port of your Arduino)**. Next upload the program by going to **File > Upload to I/O Board (ctrl+U)**. Finally, bask in the glory and possibility that controlling lights offers.

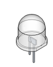
If you are having trouble uploading, a full trouble shooting guide can be found here: **http://ardx.org/TRBL**

The Circuit:

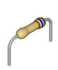
Parts:


CIRC-01 Breadboard Sheet x1

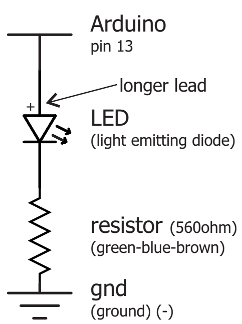
2 Pin Header x4


10mm LED x1


Wire

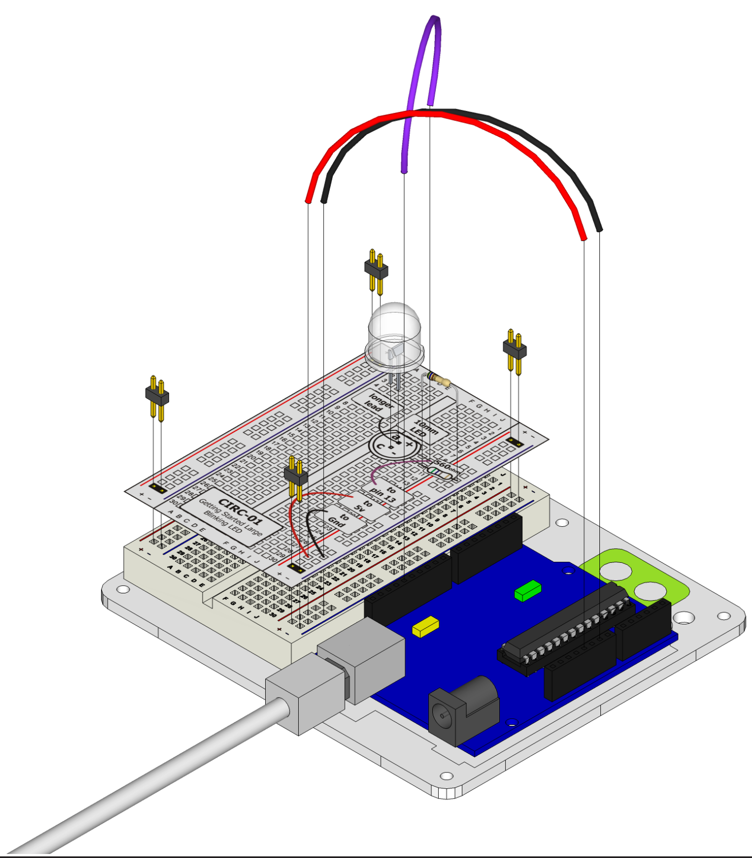

560 Ohm Resistor Green-Blue-Brown x1

Schematic

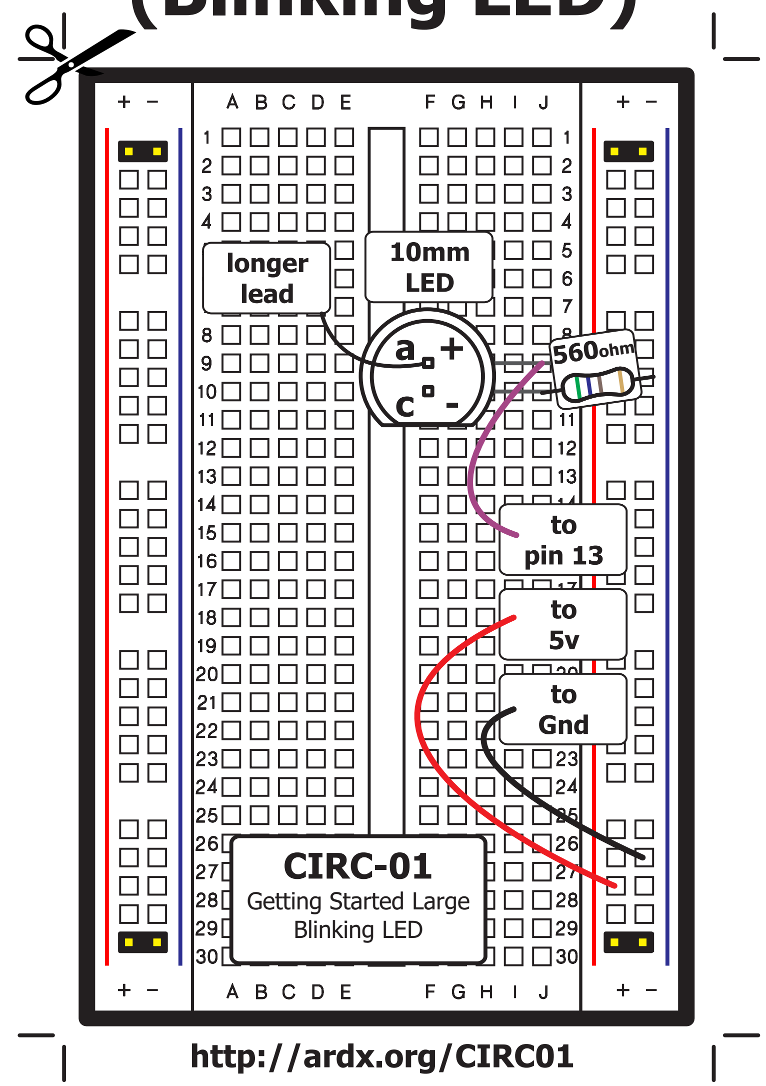


The Internet

.:download:.
breadboard layout sheet
http://ardx.org/BBLS01
.:view:.
assembly video
http://ardx.org/VIDE01



CIRC-01
Getting Started
(Blinking LED)



http://ardx.org/CIRC01

oomlout.com

Note: Breadboard dimensions vary between manufacturers so some adjustments may be necessary.

.:For a full programming reference visit:.
http://ardx.org/PROG

03 PROG
programming primer

Maths Operators

Operators used for manipulating numbers. (they work like simple maths).

- **=** (assignment) makes something equal to something else (eg. x = 10 * 2 (x now equals 20))
- **%** (modulo) gives the remainder when one number is divided by another (ex. 12 % 10 (gives 2))
- **+** (addition)
- **-** (subtraction)
- ***** (multiplication)
- **/** (division)

Comparison Operators

Operators used for logical comparison.

- **==** (equal to) (eg. 12 == 10 is FALSE or 12 == 12 is TRUE)
- **!=** (not equal to) (eg. 12 != 10 is TRUE or 12 != 12 is FALSE)
- **<** (less than) (eg. 12 < 10 is FALSE or 12 < 12 is FALSE or 12 < 14 is TRUE)
- **>** (greater than) (eg. 12 > 10 is TRUE or 12 > 12 is FALSE or 12 > 14 is FALSE)

Control Structure

Programs are reliant on controlling what runs next, here are the basic control elements (there are many more online).

```
if(condition){ }
else if( condition ){ }
else { }
```

This will execute the code between the curly brackets if the condition is true, and if not it will test the `else if` condition if that is also false the `else` code will execute.

```
for(int i = 0; i <
#repeats; i++){   }
```

Used when you would like to repeat a chunk of code a number of times (can count up i++ or down i-- or use any variable)

Digital

`pinMode(pin, mode);`

Used to set a pin's mode, `pin` is the pin number you would like to address 0-19 (analog 0-5 are 14-19). The mode can either be `INPUT` or `OUTPUT`.

`digitalWrite(pin, value);`

Once a pin is set as an `OUTPUT`, it can be set either `HIGH` (pulled to +5 volts) or `LOW` (pulled to ground).

`int digitalRead(pin);`

Once a pin is set as an INPUT you can use this to return whether it is HIGH (pulled to +5 volts) or LOW (pulled to ground).

Analog

The Arduino is a digital machine but it has the ability to operate in the analog realm (through tricks). Here's how to deal with things that aren't digital.

`int analogWrite(pin, value);`

Some of the Arduino's pins support pulse width modulation (3, 5, 6, 9, 10, 11). This turns the pin on and off very quickly making it act like an analog output. The value is any number between 0 (0% duty cycle ~0v) and 255 (100% duty cycle ~5 volts).

`int analogRead(pin);`

When the analog input pins are set to input you can read their voltage. A value between 0 (for 0 volts) and 1024 (for 5 volts) will be returned.

CIRC-01

Code (no need to type everything in just click)

File > Examples > 1.Basic > Blink
(example from the great arduino.cc site, check it out for other ideas)

```
/* Blink
 * Turns on an LED on for one second, then off for one second,
 * repeatedly.
 * Created 1 June 2005 By David Cuartielles
 * http://arduino.cc/en/Tutorial/Blink
 * based on an orginal by H. Barragan for the Wiring i/o board
 */

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


// The setup() method runs once, when the sketch starts
void setup()   {                 // initialize the digital pin as an output:
  pinMode(ledPin, OUTPUT);     }

// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()                     {
  digitalWrite(ledPin, HIGH);   // set the LED on
  delay(1000);                  // wait for a second
  digitalWrite(ledPin, LOW);    // set the LED off
  delay(1000);                  // wait for a second
}
```

Not Working? (3 things to try)

LED Not Lighting Up?
LEDs will only work in one direction. Try taking it out and twisting it 180 degrees. (no need to worry, installing it backwards does no permanent harm).

Program Not Uploading
This happens sometimes, the most likely cause is a confused serial port, you can change this in **tools>serial port>**

Still No Success?
A broken circuit is no fun, send us an e-mail and we will get back to you as soon as we can.
help@oomlout.com

Making it Better

Changing the pin:
The LED is connected to pin 13 but we can use any of the Arduino's pins. To change it take the wire plugged into pin 13 and move it to a pin of your choice (from 0-13) (you can also use analog 0-5, analog 0 is 14...)

Then in the code change the line:
`int ledPin = 13; -> int ledPin = newpin;`

Then upload the sketch: (ctrl-u)

Change the blink time:
Unhappy with one second on one second off?

In the code change the lines:
```
digitalWrite(ledPin, HIGH);
delay(time on);  //(seconds * 1000)
digitalWrite(ledPin, LOW);
delay(time off); //(seconds * 1000)
```

Control the brightness:
Along with digital (on/off) control the Arduino can control some pins in an analog (brightness) fashion. (more details on this in later circuits). To play around with it.

Change the LED to pin 9: (also change the wire)
`ledPin = 13; -> int ledPin = 9;`

Replace the code inside the `{ }`'s of `loop()` with this:
`analogWrite(ledPin, new number);`

(new number) = any number between 0 and 255.
0 = off, 255 = on, in between = different brightness

Fading:
We will use another included example program. To open go to

File > Examples > 3.Analog > Fading

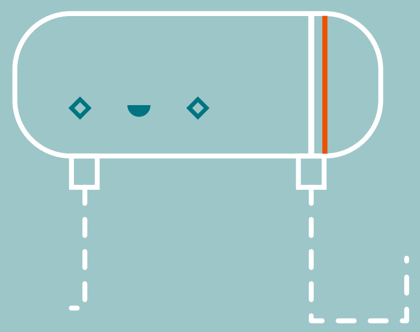
Then upload to your board and watch as the LED fades in and then out.

More, More, More:

More details, where to buy more parts, where to ask more questions:

http://ardx.org/CIRC01

- Introduzione al progetto
- Troubleshooting
- Proposte di modifiche sia al codice che al circuito (con spiegazione)
- Schema del circuito con elenco visivo componenti
- Pochissimo commento al codice
- Alcuni codici sono quelli del kit di Arduino

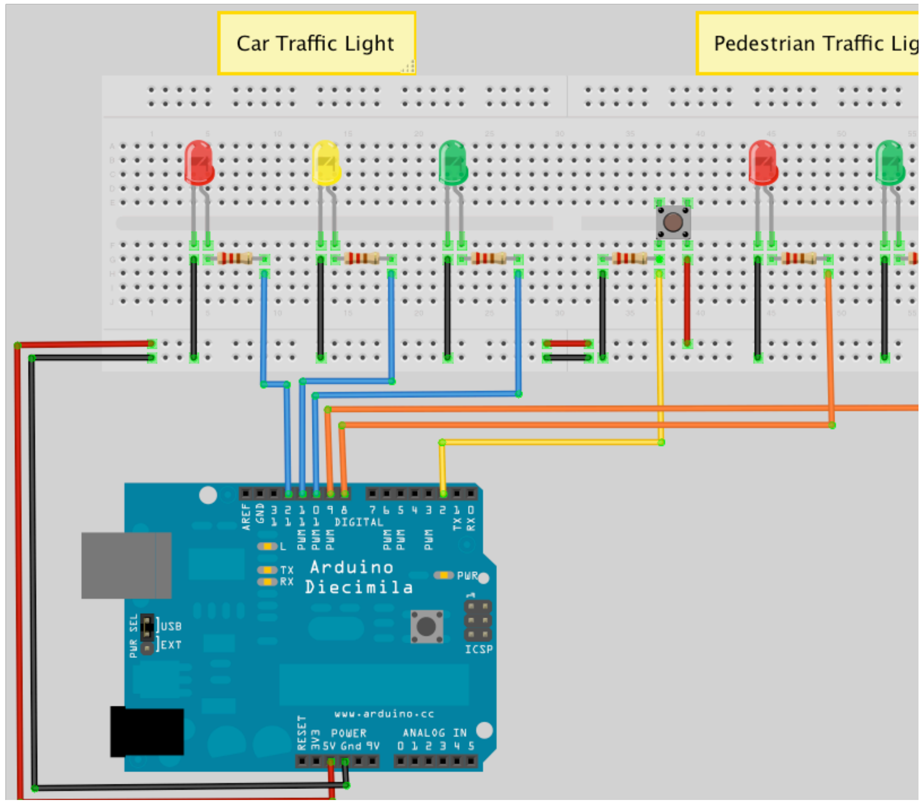


Earthshine Design

Non più disponibile

Manuale in inglese

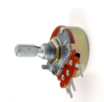
https://ia800603.us.archive.org/15/items/ArduinoStarterKitManual/Arduino_Beginners_Manual.pdf



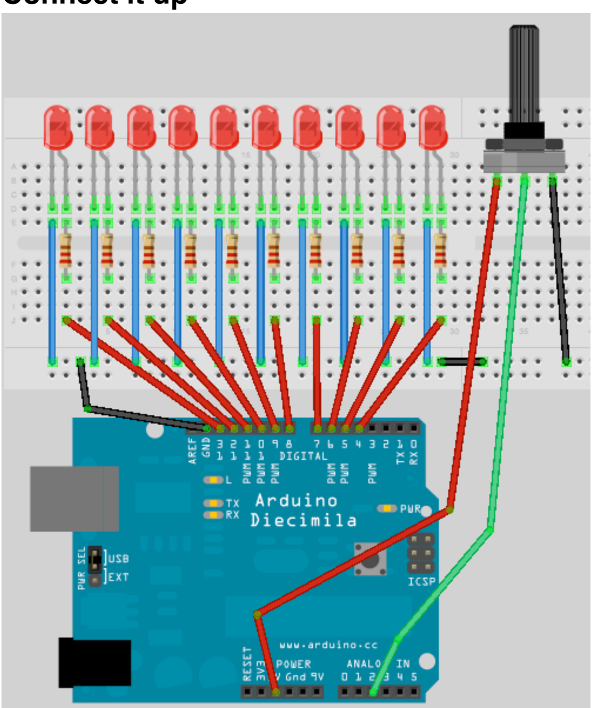
Project 6 - Interactive LED Chase Effect

We are now going to use a string of LED's (10 in total) to make an LED chase effect, similar to that used on the car KITT in the Knightrider TV Series and on the way introduce the concept of arrays.

What you will need

Parts from previous project plus....	
4K7 Potentiometer	

Connect it up



This is the same circuit as in Project 5, but we have simply added the potentiometer and connected it to 5v, Ground and Analog Pin 5.

Enter the code

```
// Create array for LED pins
byte ledPin[] = {4, 5, 6, 7, 8, 9, 10,
11, 12, 13};
int ledDelay; // delay between changes
int direction = 1;
int currentLED = 0;
unsigned long changeTime;
int potPin = 2;    // select the input
pin for the potentiometer

void setup() {
  // set all pins to output
  for (int x=0; x<10; x++) {
    pinMode(ledPin[x], OUTPUT); }
    changeTime = millis();
}

void loop() {
// read the value from the pot
ledDelay = analogRead(potPin);
  // if it has been ledDelay ms since
last change
  if ((millis() - changeTime) >
ledDelay) {
    changeLED();
    changeTime = millis();
  }
}

void changeLED() {
  // turn off all LED's
  for (int x=0; x<10; x++) {
    digitalWrite(ledPin[x], LOW);
  }
  // turn on the current LED
  digitalWrite(ledPin[currentLED],
HIGH);
  // increment by the direction value
  currentLED += direction;
  // change direction if we reach the
end
  if (currentLED == 9) {direction =
-1;}
  if (currentLED == 0) {direction = 1;}
}
```

This time when verify and upload your code, you should now see the lit LED appear to bounce back and forth between each end of the string of lights as before. But, by turning the knob of the potentiometer, you will change the value of `ledDelay` and speed up or slow down the effect.

Let's take a look at how this works and find our what a potentiometer is.

Project 6 - Code Overview

```
// Create array for LED pins
byte ledPin[] = {4, 5, 6, 7, 8, 9, 10, 11,
12, 13};
int ledDelay; // delay between changes
int direction = 1;
int currentLED = 0;
unsigned long changeTime;
int potPin = 2;    // select the input pin
for the potentiometer

void setup() {
  // set all pins to output
  for (int x=0; x<10; x++) {
    pinMode(ledPin[x], OUTPUT); }
    changeTime = millis();
}

void loop() {
// read the value from the pot
ledDelay = analogRead(potPin);
  // if it has been ledDelay ms since last
change
  if ((millis() - changeTime) > ledDelay) {
    changeLED();
    changeTime = millis();
  }
}

void changeLED() {
  // turn off all LED's
  for (int x=0; x<10; x++) {
    digitalWrite(ledPin[x], LOW);
  }
  // turn on the current LED
  digitalWrite(ledPin[currentLED], HIGH);
  // increment by the direction value
  currentLED += direction;
  // change direction if we reach the end
  if (currentLED == 9) {direction = -1;}
  if (currentLED == 0) {direction = 1;}
}
```

The code for this Project is almost identical to the previous project. We have simply added a potentiometer to our hardware and the code has additions to enable us to read the values from the potentiometer and use them to adjust the speed of the LED chase effect.

We first declare a variable for the potentiometer pin

```
int potPin = 2;
```

as our potentiometer is connected to analog pin 2. To read the value from an analog pin we use the `analogRead` command. The Arduino has 6 analog input/outputs with a 10-bit analog to digital convertor (we will discuss bits later on). This means the analog pin can read in voltages between 0 to 5 volts in integer values between 0 (0 volts) and 1023 (5 volts). This gives a resolution of 5 volts / 1024 units or 0.0049 volts (4.9mV) per unit.

We need set our delay using the potentiometer so we will simply use the direct values read in from the pin to adjust the delay between 0 and 1023 milliseconds. We do this be directly reading the value of the potentiometer pin into ledDelay. Notice that we do not need to set an analog pin to be an input or output like we need to with a digital pin.

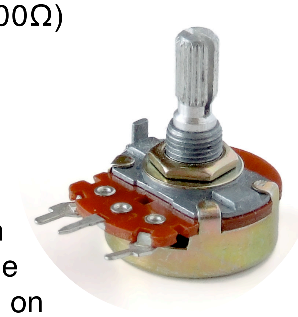
```
ledDelay = analogRead(potPin);
```

This is done during our main loop and therefore it is constantly being read and adjusted. By turning the knob you can adjust the delay value between 0 and 1023 milliseconds (or just over a second) and therefore have full control over the speed of the effect.

OK let's find out what a potentiometer is and how it works.

Project 6 - Hardware Overview

The only additional piece of hardware used in this project was the 4K7 (4700Ω) potentiometer.



You have already come across a resistor and know how they work. The potentiometer is simply an adjustable resistor with a range from 0 to a set value (written on the side of the pot). In the kit you have been given a 4K7 or 4,700Ω potentiometer which means it's range is from 0 to 4700 Ohms.

The potentiometer has 3 legs. By connecting up just two legs the potentiometer becomes a variable resistor. By connecting all 3 legs and applying a voltage across it, the pot becomes a voltage divider. This is how we have used it in our circuit. One side is connected to ground, the other to 5v and the centre pin to our analog pin. By adjusting the knob, a voltage between 0 and 5v will be leaked from the centre pin and we can read the value of that voltage on Analog Pin 2 and use it's value to change the delay rate of the light effect.

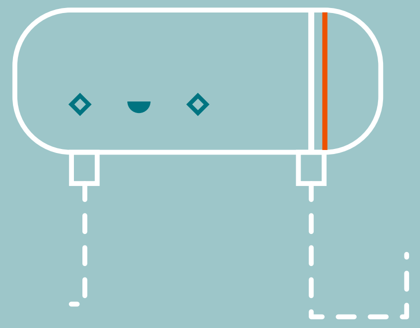
The potentiometer can be very useful in providing a means of adjusting a value from 0 to a set amount, e.g. the volume of a radio or the brightness of a lamp. In fact, dimmer switches for your home lamps are a kind of potentiometer.

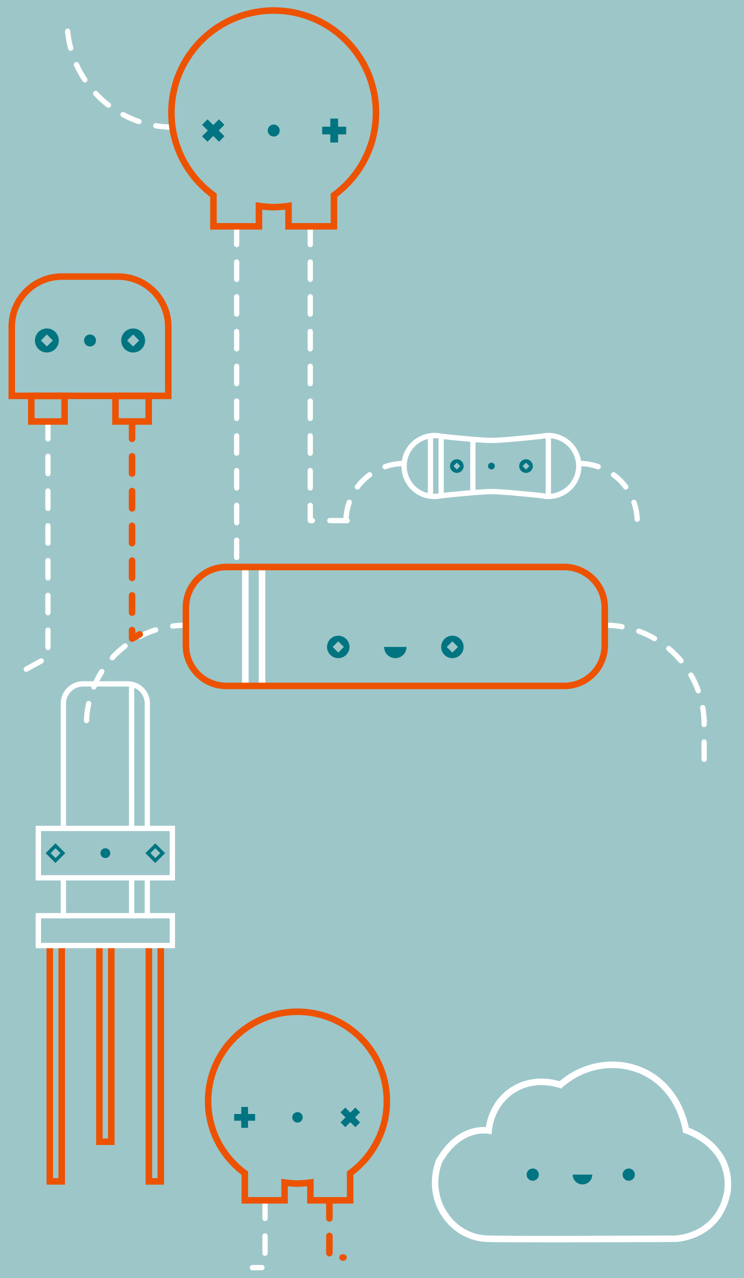
Exercises

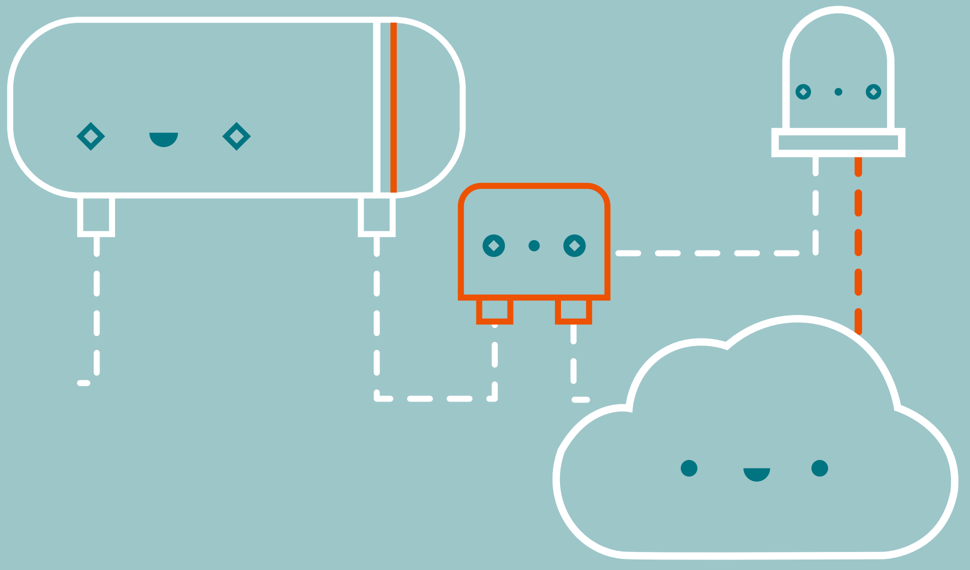
1. Get the LED's at BOTH ends of the strip to start as on, then to both move towards each other, appear to bounce off each other and then move back to the end.

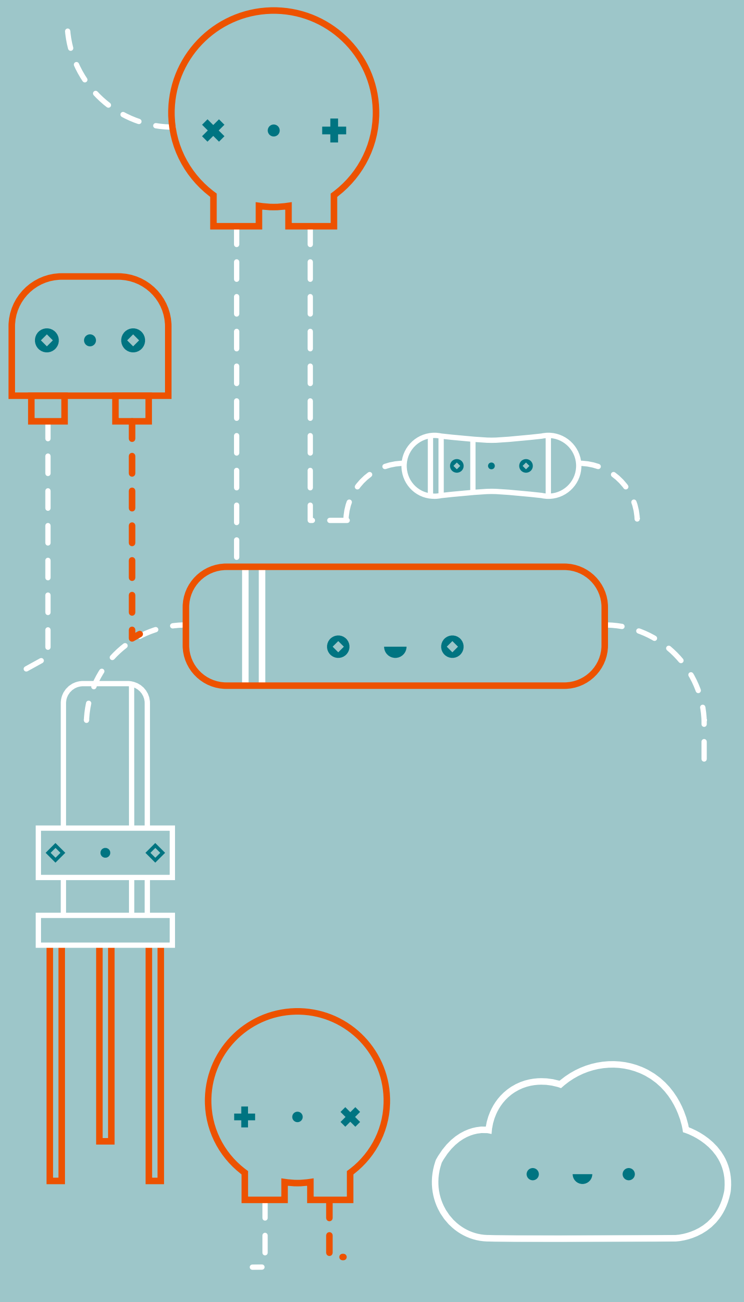
2. Make a bouncing ball effect by making the LED start at one end, 'drop' toward the other end, bounce back up, but to only go up 9 spaces, bounce, go up 8 spaces, then 7, then 6, etc. To give the effect it is a bouncing ball, getting bouncing up to a lower height on each bounce.

- Elenco visivo componenti con overview contenuti e descrizione in dettaglio hardware
- Descrizione molto approfondita del codice
- Il codice si può copiare direttamente dal pdf
- Non contiene lo schema del circuito
- Esercizi e spunti per modificare il codice

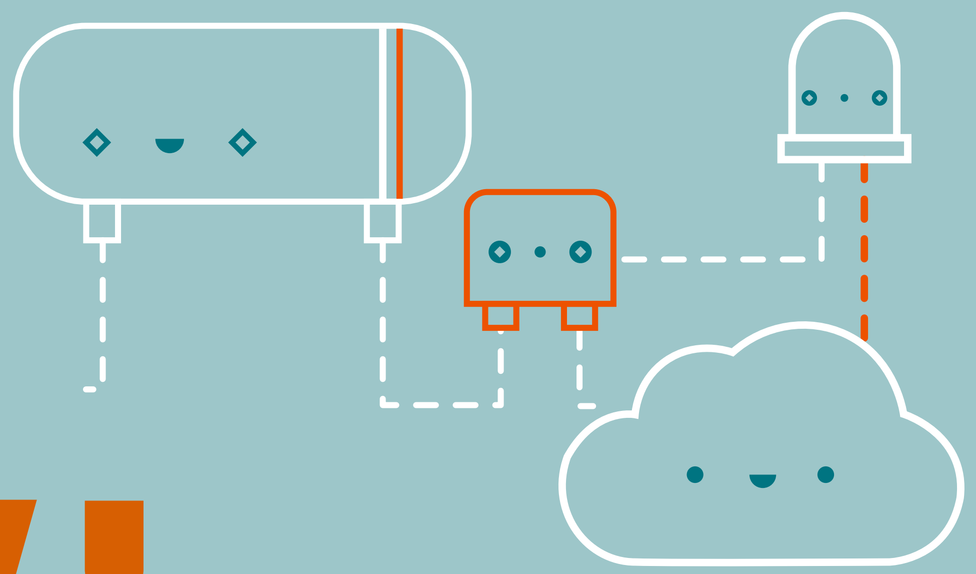


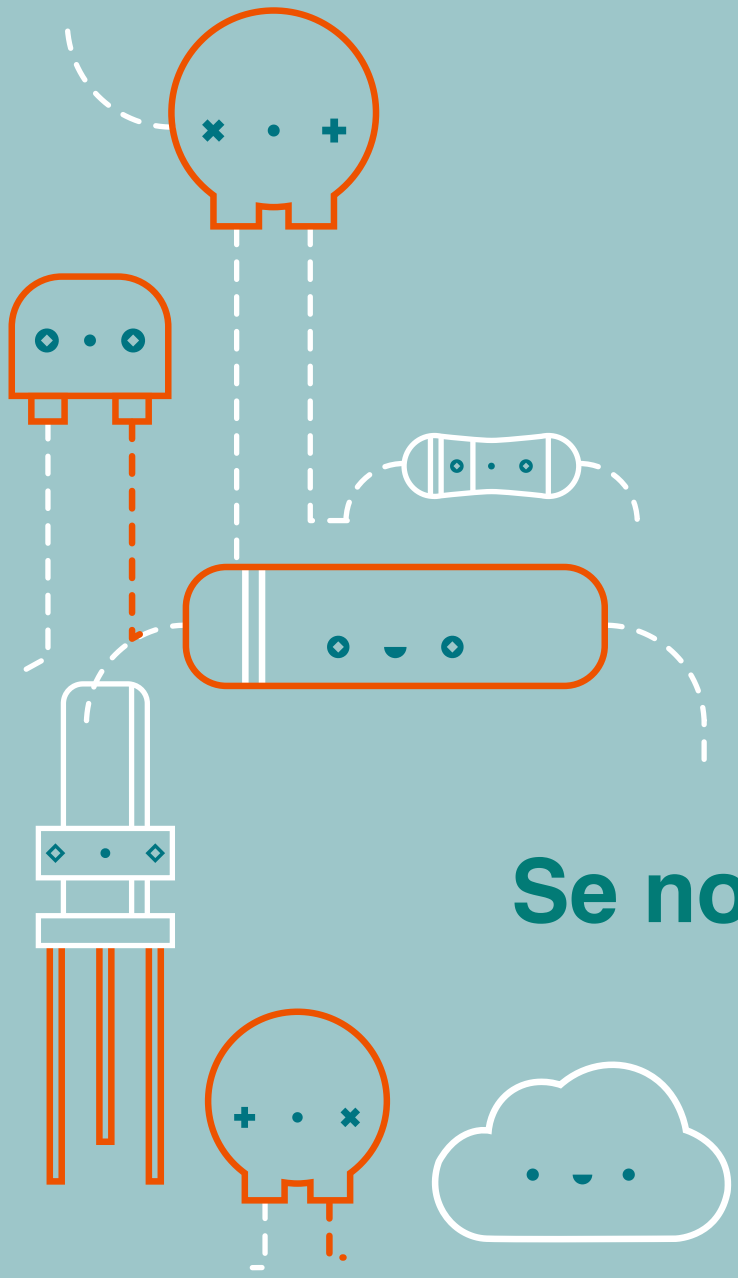


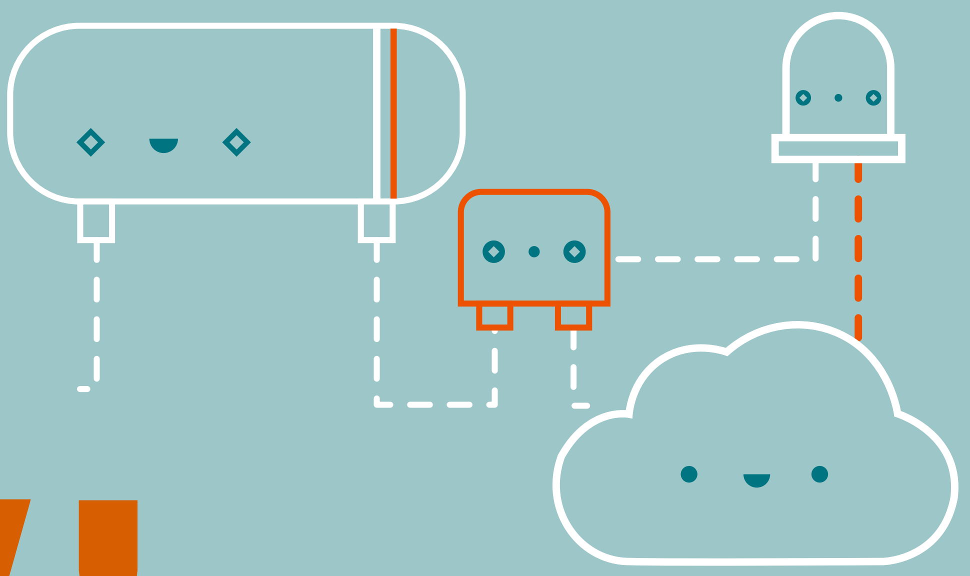




NOW!





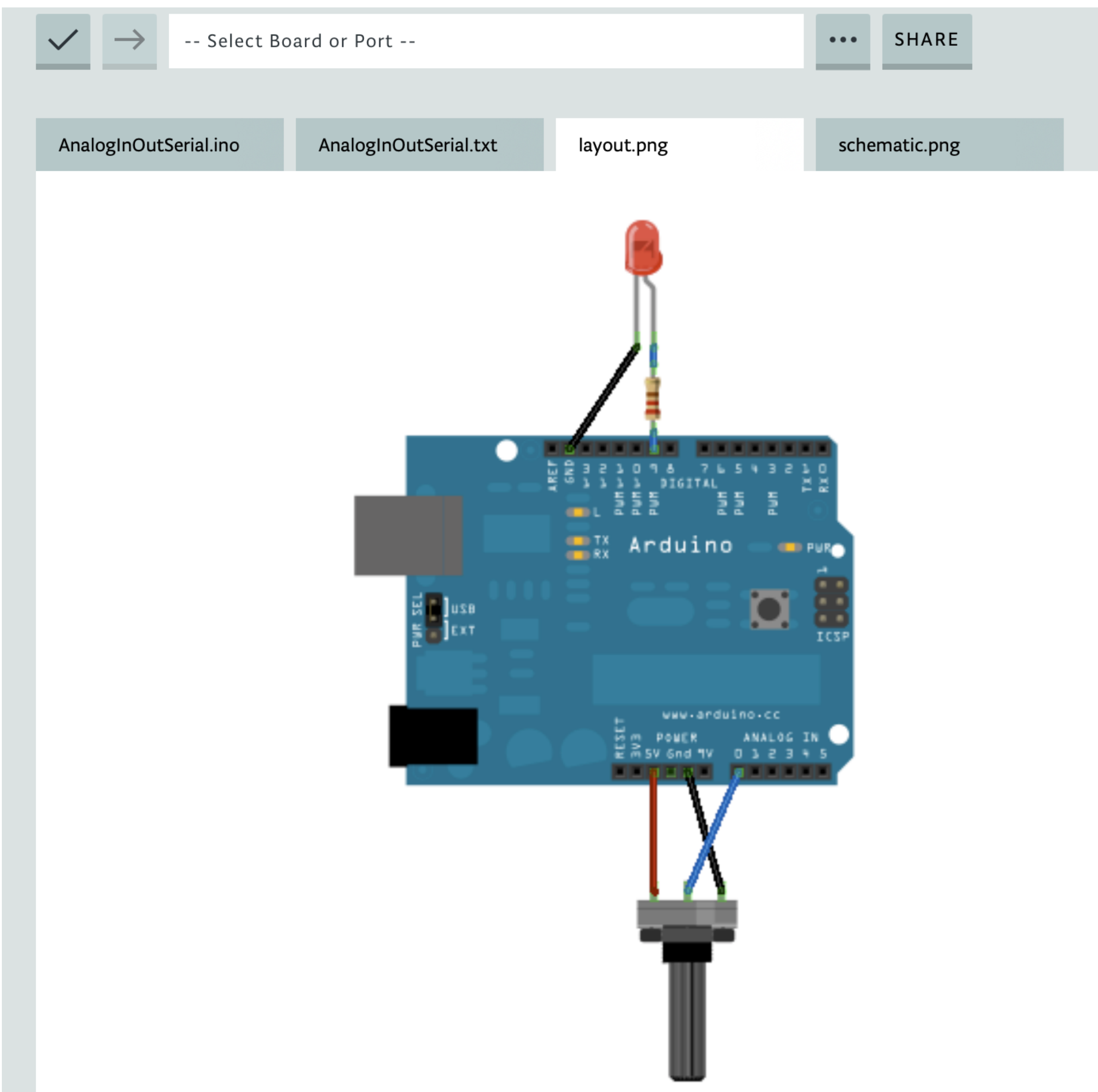


NOW!

Se non ho voglia di aspettare il corriere

Tool online

https://create.arduino.cc/

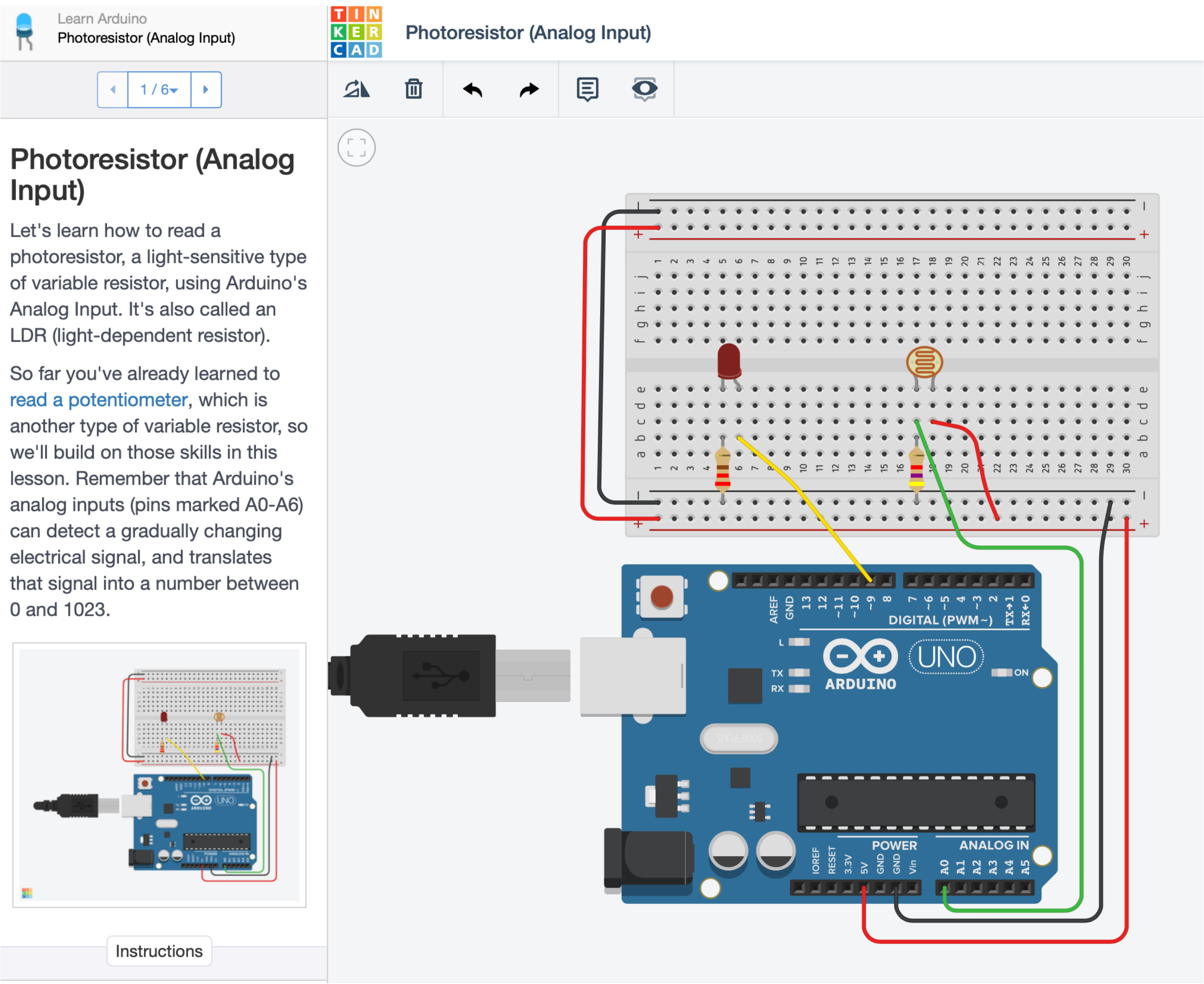


AnalogInOutSerial.ino | AnalogInOutSerial.txt | layout.png | schematic.png

```
/*
  Analog input, analog output, serial output

  Reads an analog input pin, maps the result to a range from 0 to 255 and uses
  the result to set the pulse width modulation (PWM) of an output pin.
  Also prints the results to the Serial Monitor.

  The circuit:
  - potentiometer connected to analog pin 0.
    Center pin of the potentiometer goes to the analog pin.
    side pins of the potentiometer go to +5V and ground
  - LED connected from digital pin 9 to ground

  created 29 Dec. 2008
  modified 9 Apr 2012
  by Tom Igoe

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/AnalogInOutSerial
*/

// These constants won't change. They're used to give names to the pins used:
const int analogInPin = A0;  // Analog input pin that the potentiometer is attached to
const int analogOutPin = 9; // Analog output pin that the LED is attached to

int sensorValue = 0;        // value read from the pot
int outputValue = 0;        // value output to the PWM (analog out)

void setup() {
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
}
```

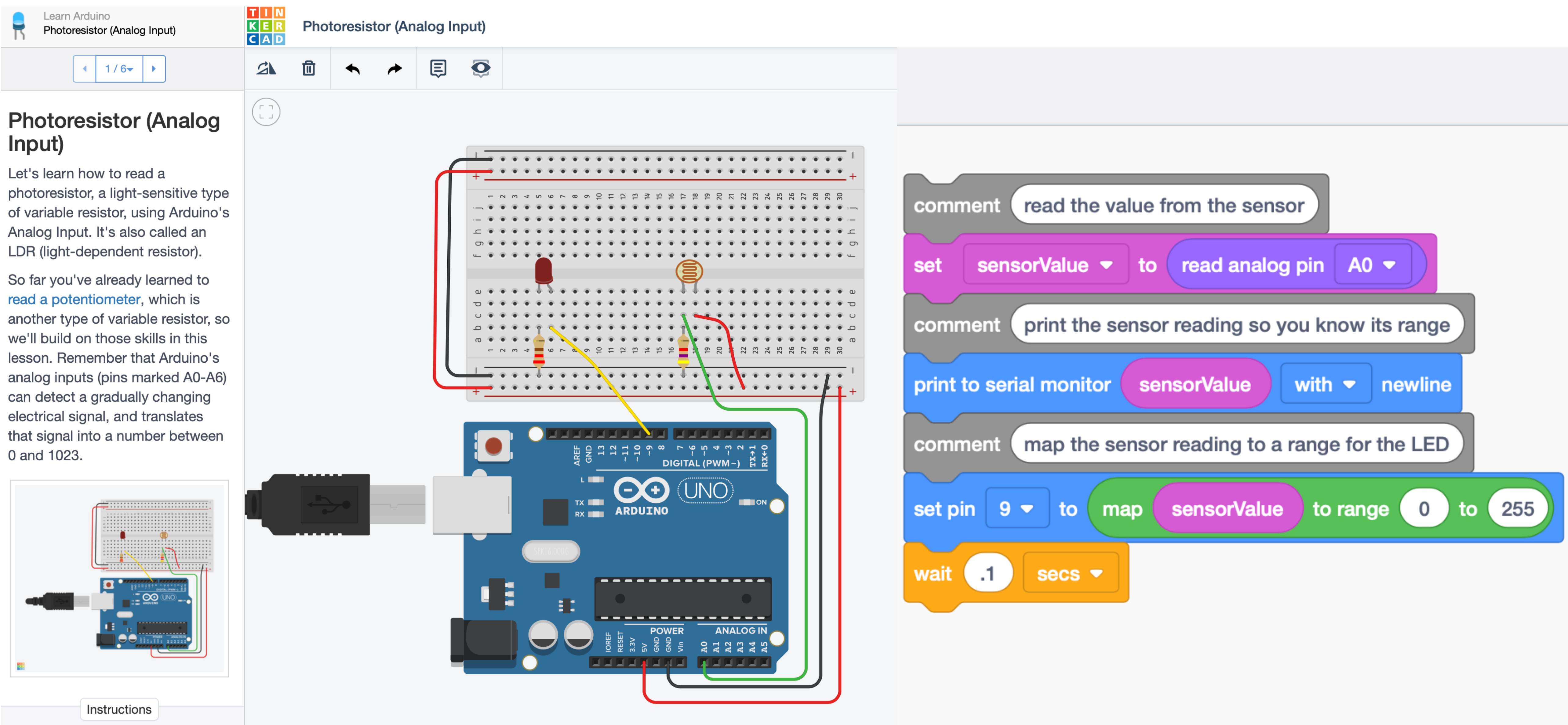
Tool online

https://www.tinkercad.com



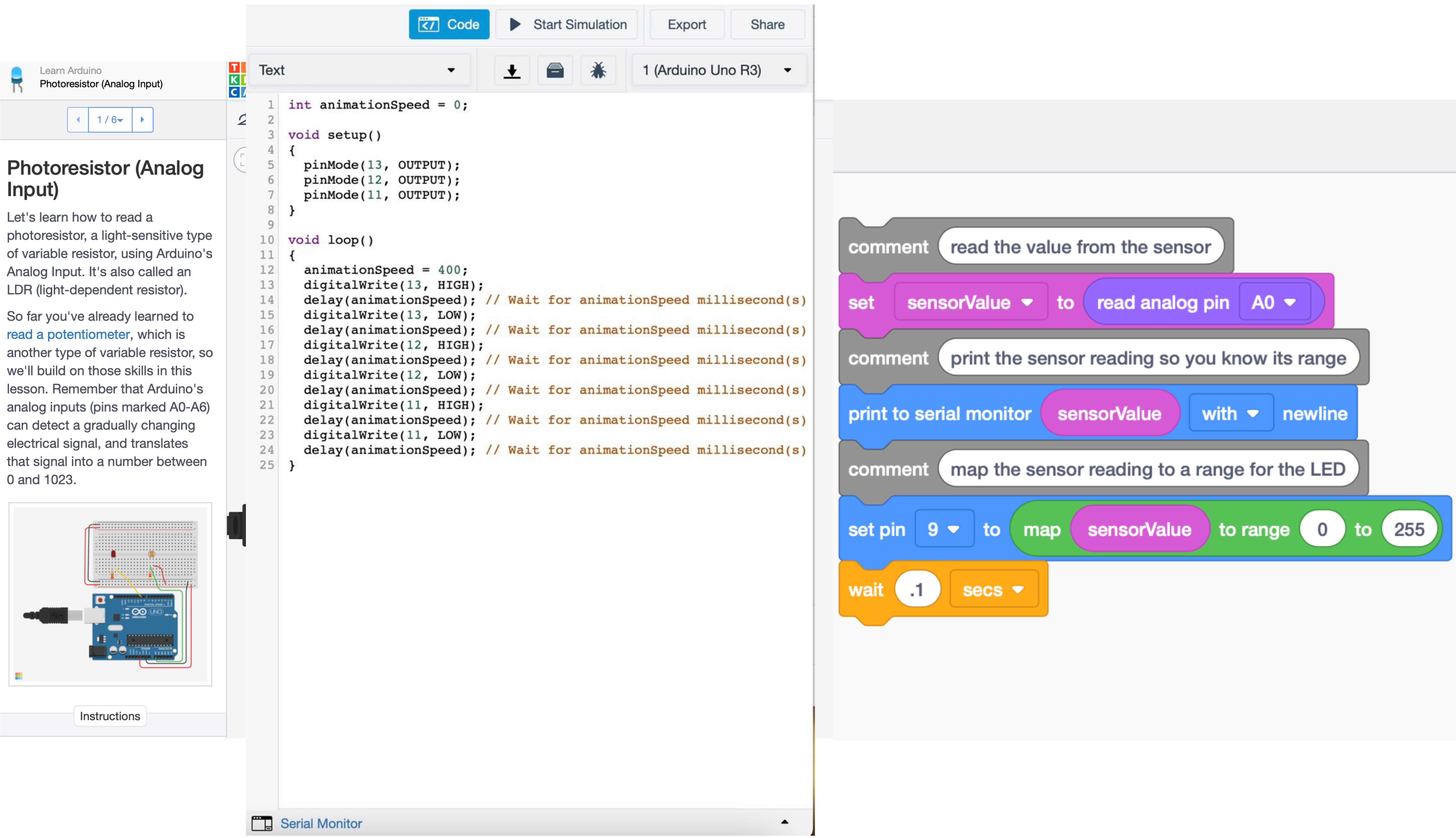
Tool online

https://www.tinkercad.com



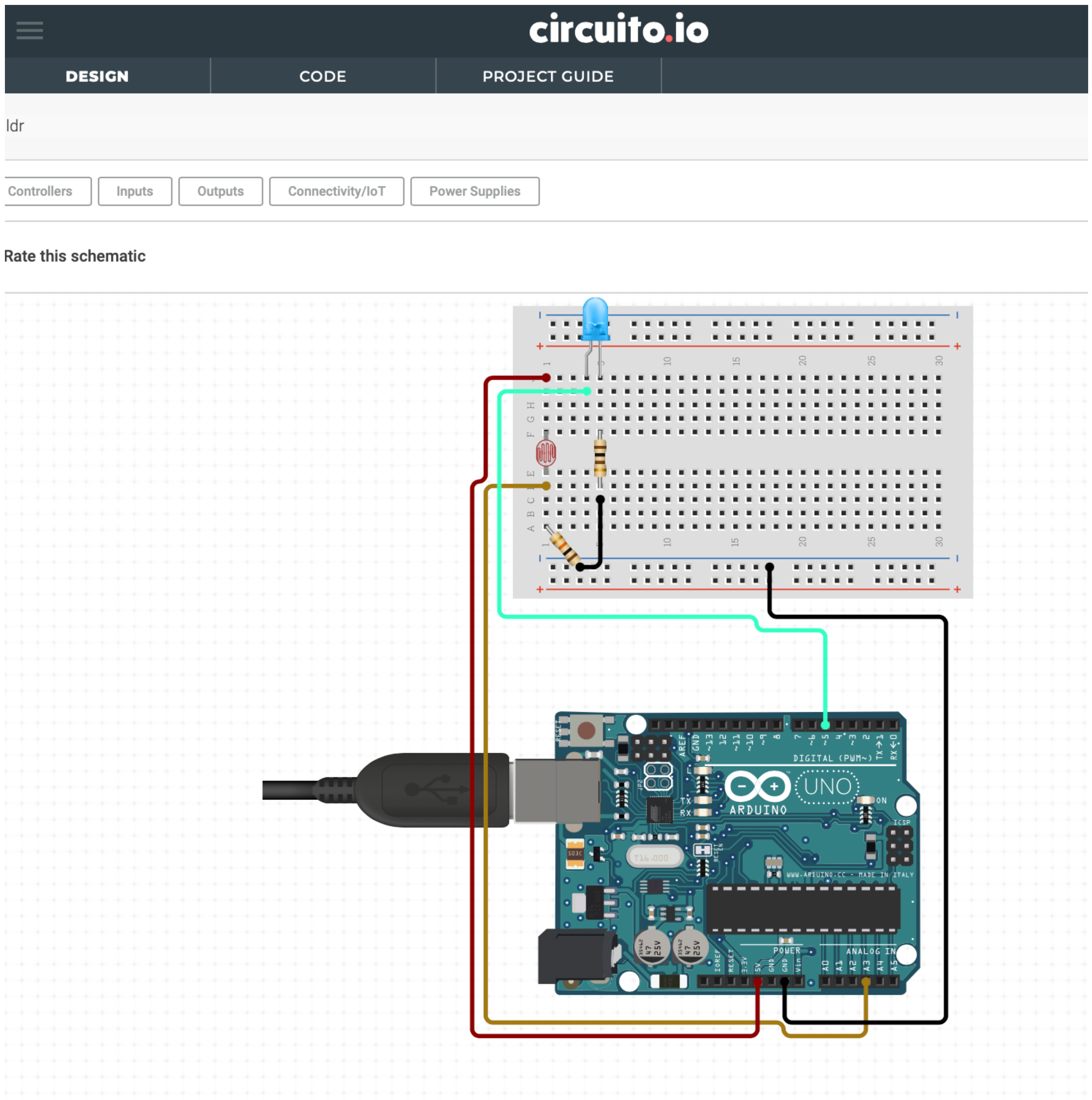
Tool online

https://www.tinkercad.com



Tool online

https://www.circuito.io/



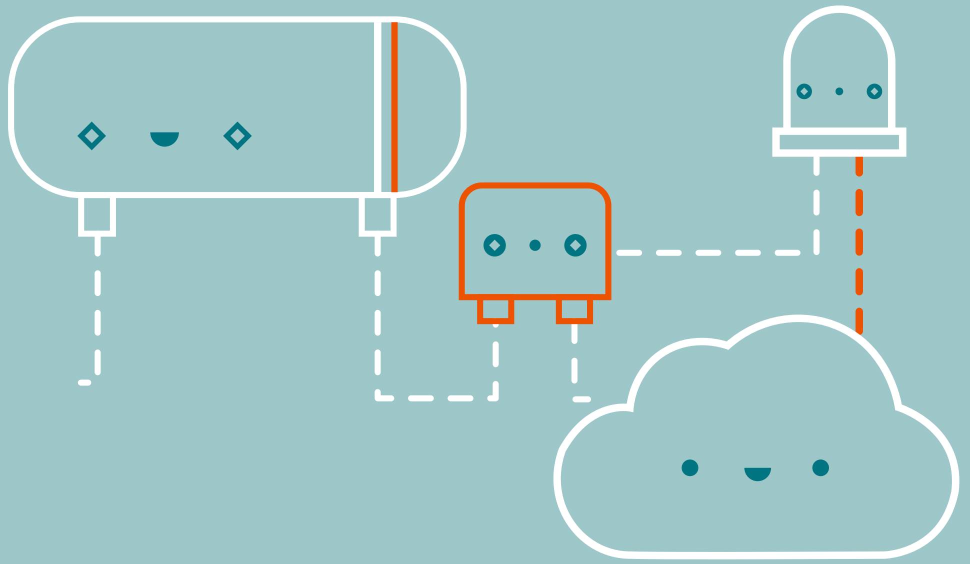
circuito.io

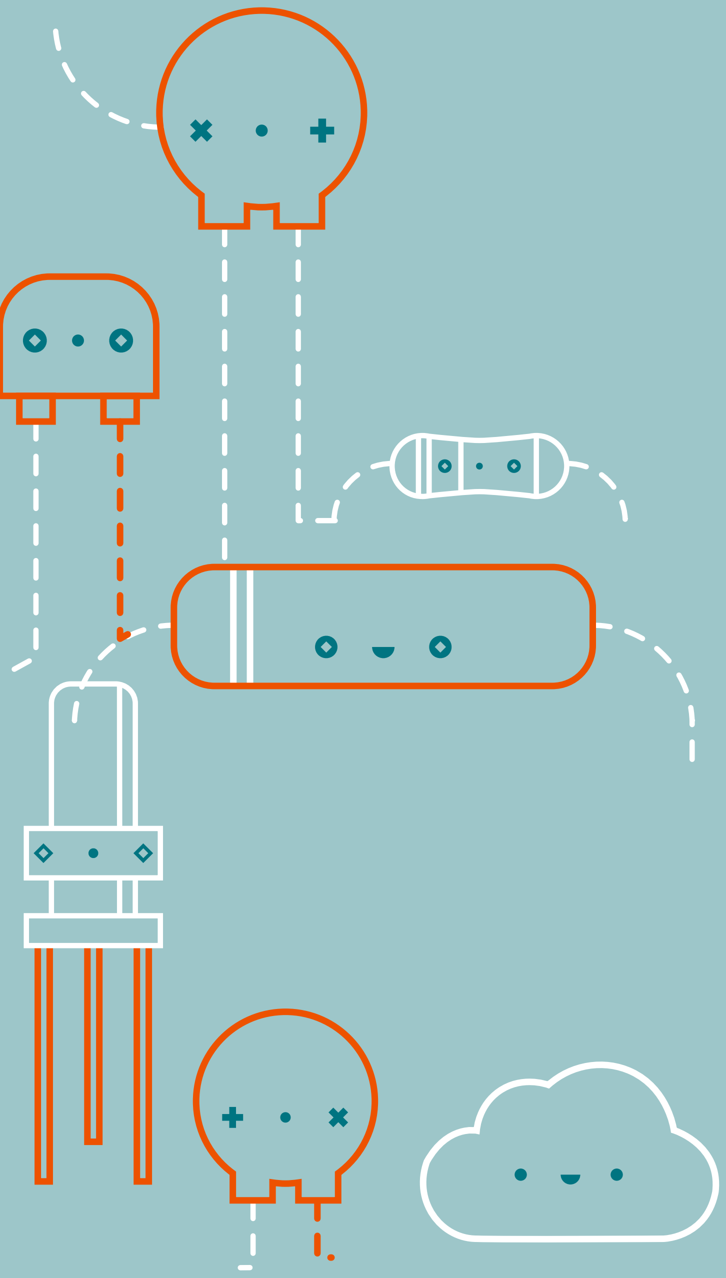
DESIGN | CODE | PROJECT GUIDE

Files:

- Firmware.zip
  - **Firmware.ino**
  - LDR.cpp
  - LDR.h
  - LDR_license.txt
  - LED.cpp
  - LED.h
  - LED_license.txt
  - Switchable.h
  - Switchable.cpp

```

// Include Libraries
#include "Arduino.h"
#include "LDR.h"
#include "LED.h"


// Pin Definitions
#define LDR_PIN_SIG A3
#define LEDB_PIN_VIN    5



// Global variables and defines
#define THRESHOLD_ldr   100
int ldrAverageLight;
// object initialization
LDR ldr(LDR_PIN_SIG);
LED ledB(LEDB_PIN_VIN);


// define vars for testing menu
const int timeout = 10000;       //define timeout of 10 sec
char menuOption = 0;
long time0;

// Setup the essentials for your circuit to work. It runs first every time your circuit is powe
void setup()
{
    // Setup Serial which is useful for debugging
    // Use the Serial Monitor to view printed messages
    Serial.begin(9600);
    while (!Serial) ; // wait for serial port to connect. Needed for native USB
    Serial.println("start");

    ldrAverageLight = ldr.readAverage();
```





E POI ?

Forum

PROFESSIONAL | EDUCATION | STORE | Search on Arduino.cc | SIGN IN

HARDWARE | SOFTWARE | CLOUD | DOCUMENTATION | COMMUNITY | BLOG | ABOUT

Arduino Forum > International > Italiano

Sub-Boards

Generale **Moderators:** UweFederer, gpb01 **Last post:** **Today** at 09:59 am Re: Traslazione di scala by Datman	214,227 Posts	16,819 Topics
Hardware **Moderators:** UweFederer, gpb01 **Last post:** **Today** at 09:57 am Re: Display touch per ES... by alanmasutti	116,855 Posts	10,559 Topics
Software **Moderators:** UweFederer, gpb01 **Last post:** **Today** at 10:25 am Re: Problemi con multipl... by lo_sty	124,073 Posts	11,917 Topics
Megatopic Topic con progetti particolari (v. quadricotteri) o quelli che hanno riscosso particolare successo in termini di post/visite **Moderators:** UweFederer, gpb01 **Last post:** Mar 23, 2021, 09:59 pm Re: Modulo WiFi+Bt ESP32 by gpb01	24,117 Posts	61 Topics

Adafruit IO: IoT for Everyone

Learn More

Start Learning Electronics

Tutorials and series to get you on your way

Fun Electronic Projects

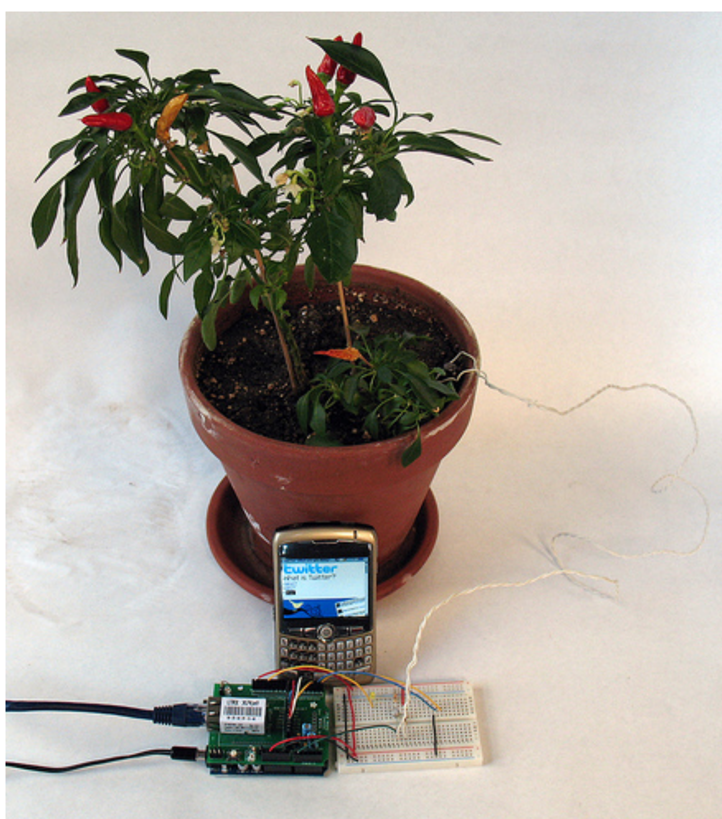
Find inspiration here

Electronic Skills Tutorials

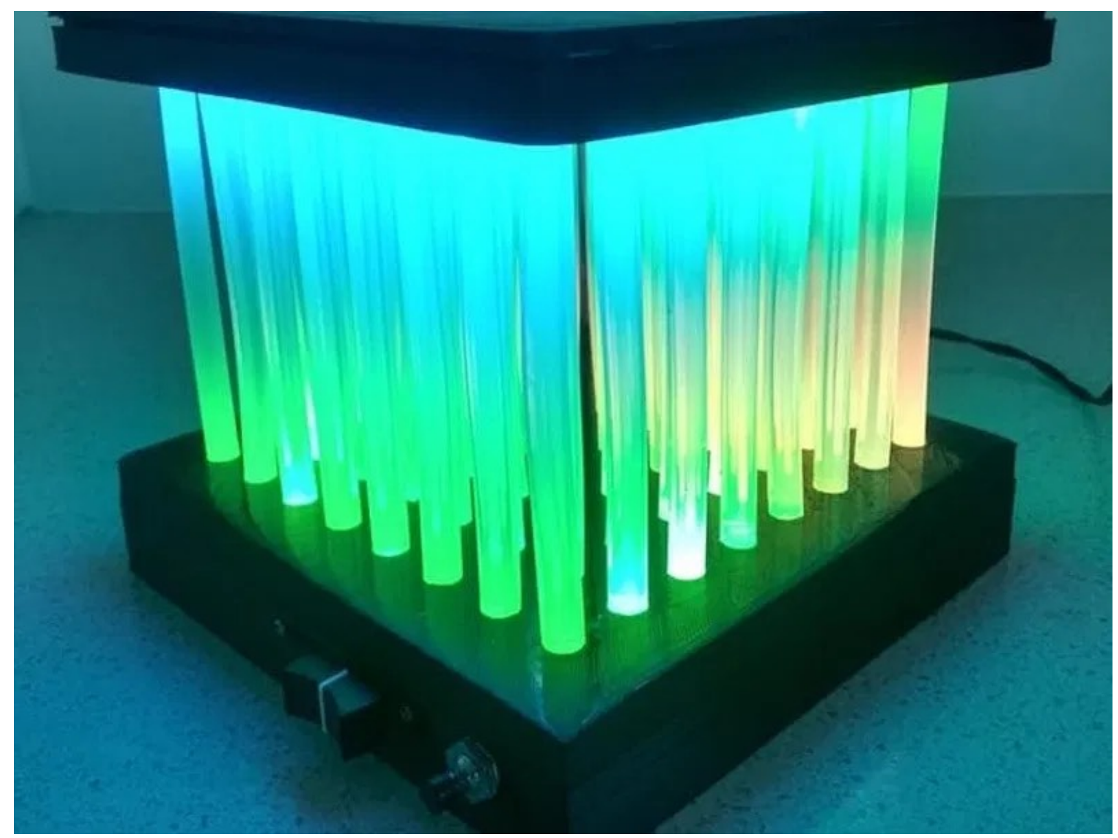
Tools for every project

Tweet-a-Pot
http://www.instructables.com/

Cat feeder
http://www.damonkohler.com

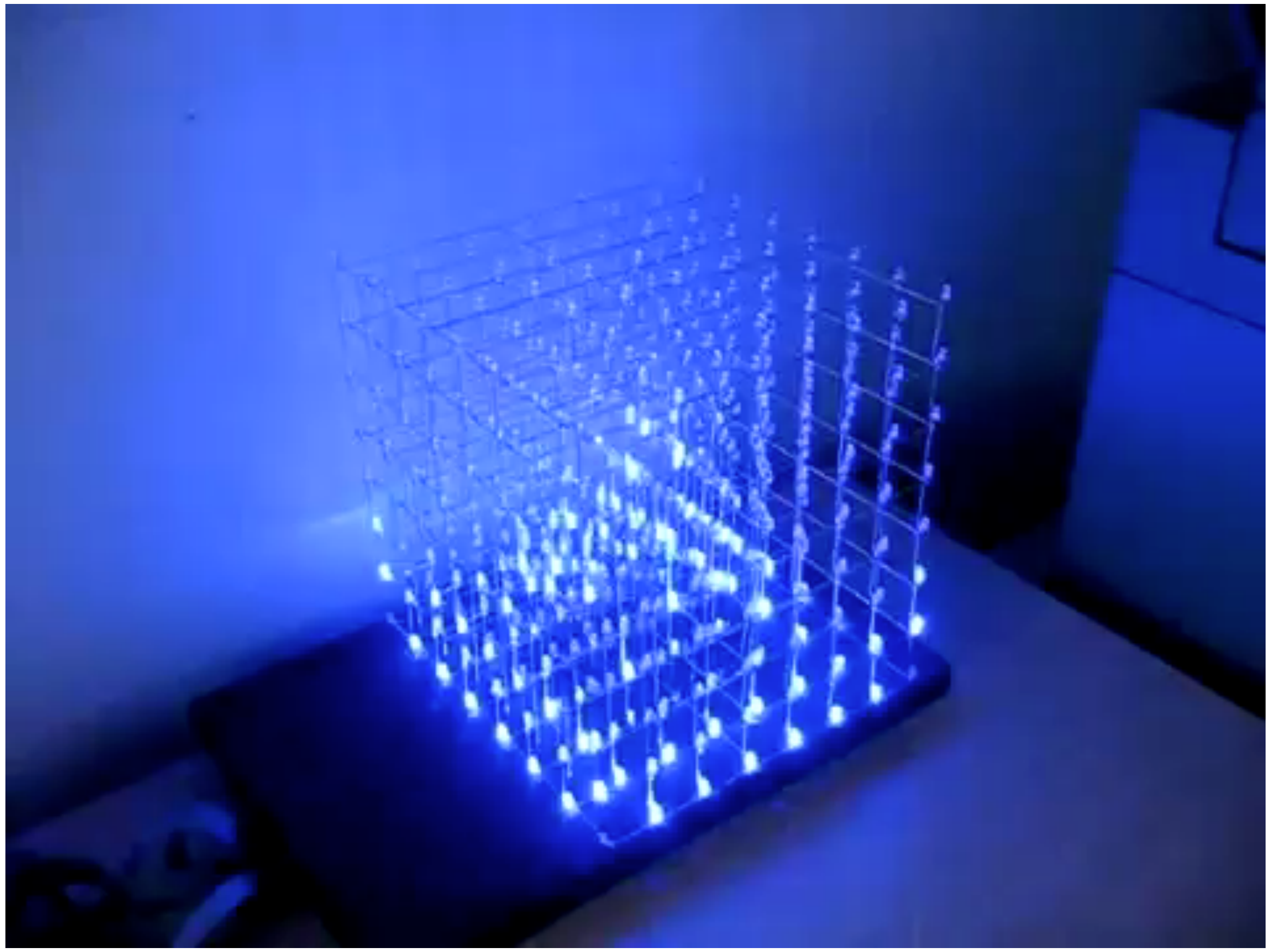


Botanicalls
http://www.botanicalls.com

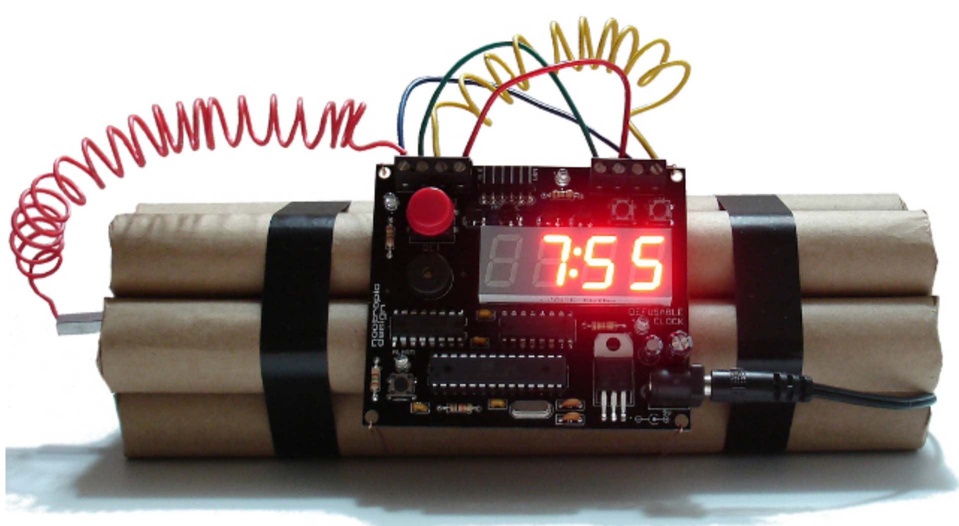


LED Matrix lamp with hot glue
https://www.thingiverse.com/thing:3403301

http://www.instructables.com/id/Led-Cube-8x8x8



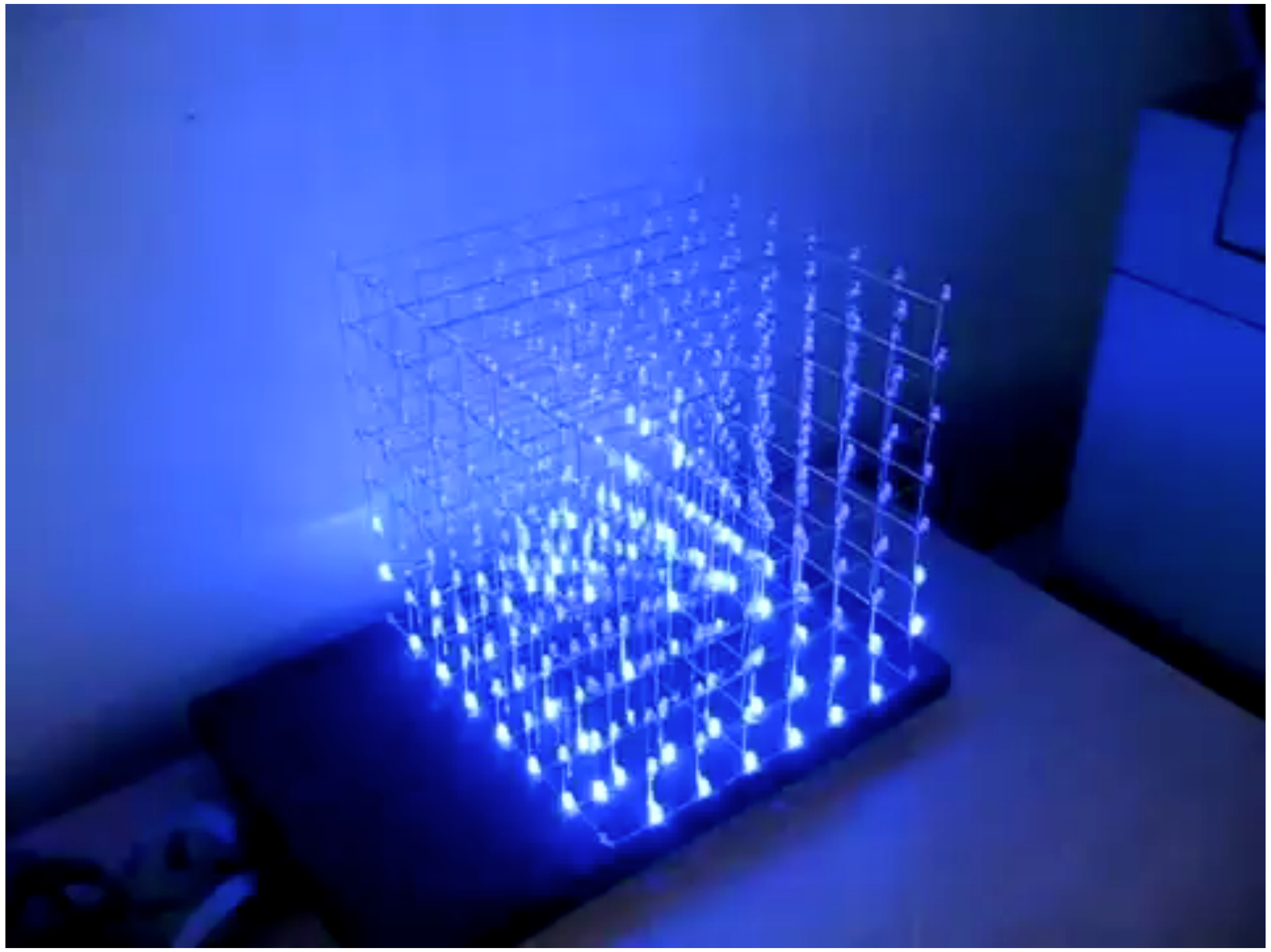
www.graysonearle.com/frogutopia/



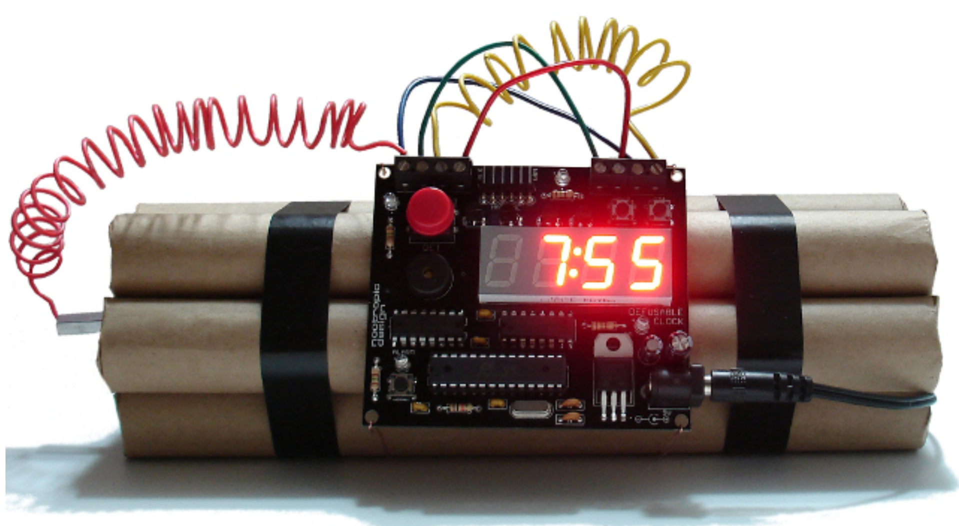
nootropicdesign.com/projectlab/2011/
09/05/defusable-clock/

http://www.instructables.com/id/
Pressure-Activated-Light-Up-Umbrella/

http://www.instructables.com/id/Led-Cube-8x8x8



www.graysonearle.com/frogutopia/

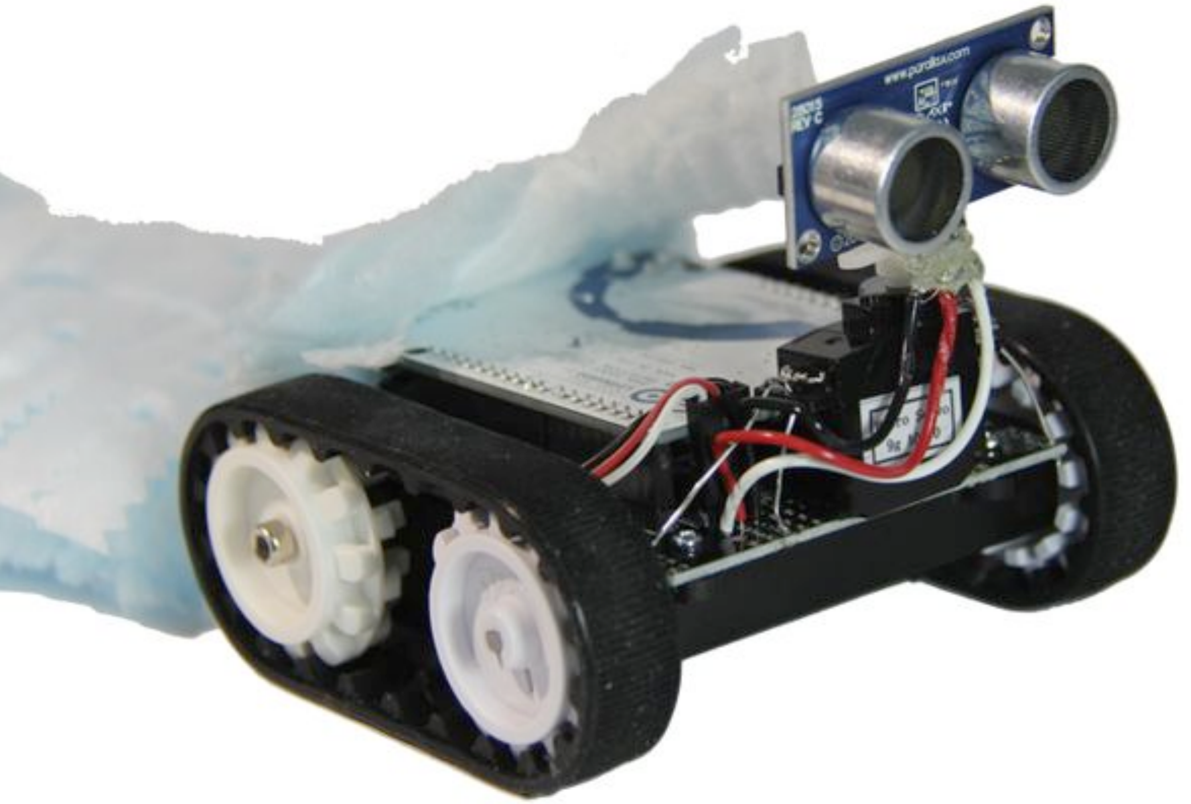


nootropicdesign.com/projectlab/2011/
09/05/defusable-clock/

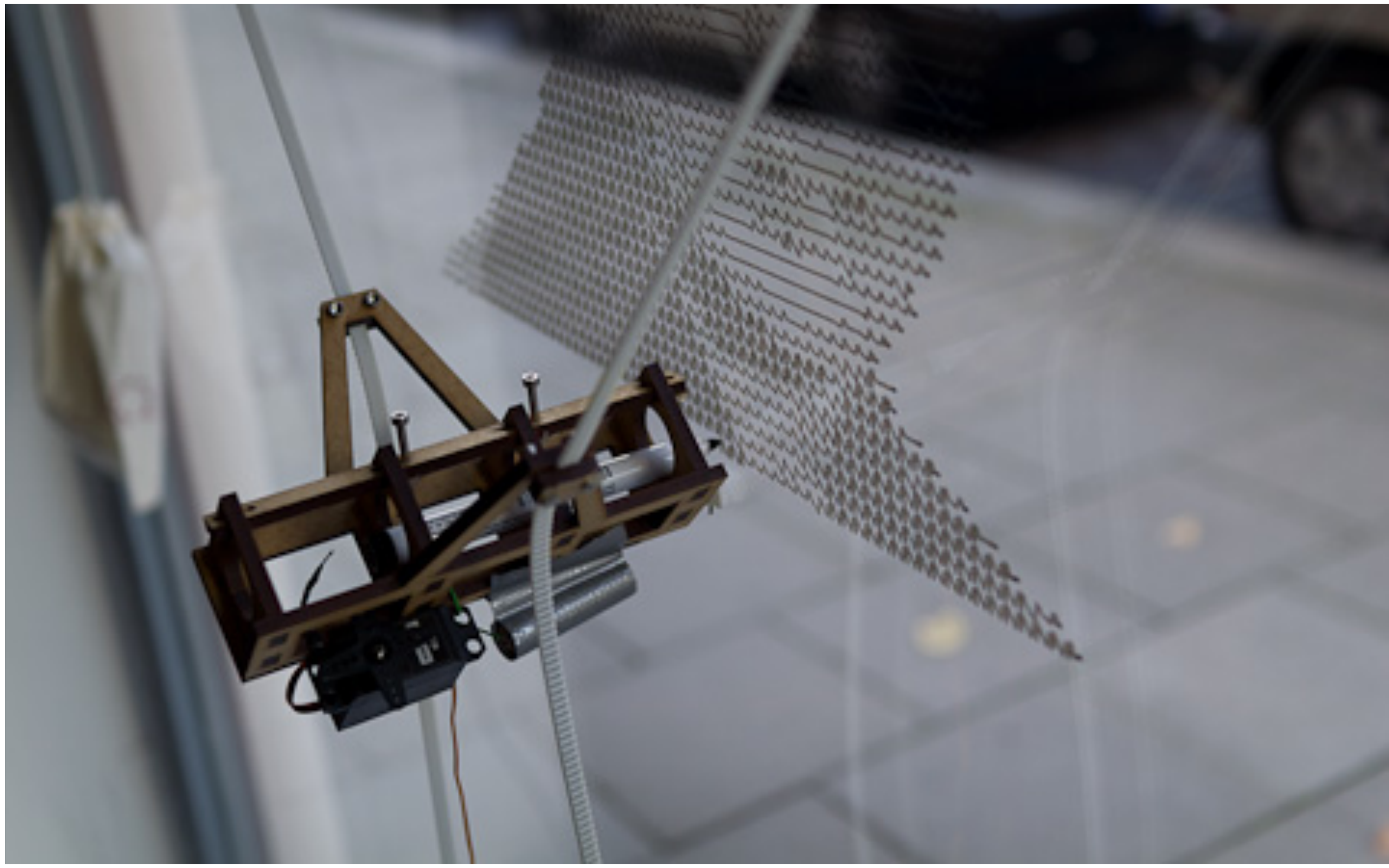
http://www.instructables.com/id/
Pressure-Activated-Light-Up-Umbrella/

www.instructables.com/id/3D-Printed-Quadcopter

http://www.thingiverse.com/thing:248009



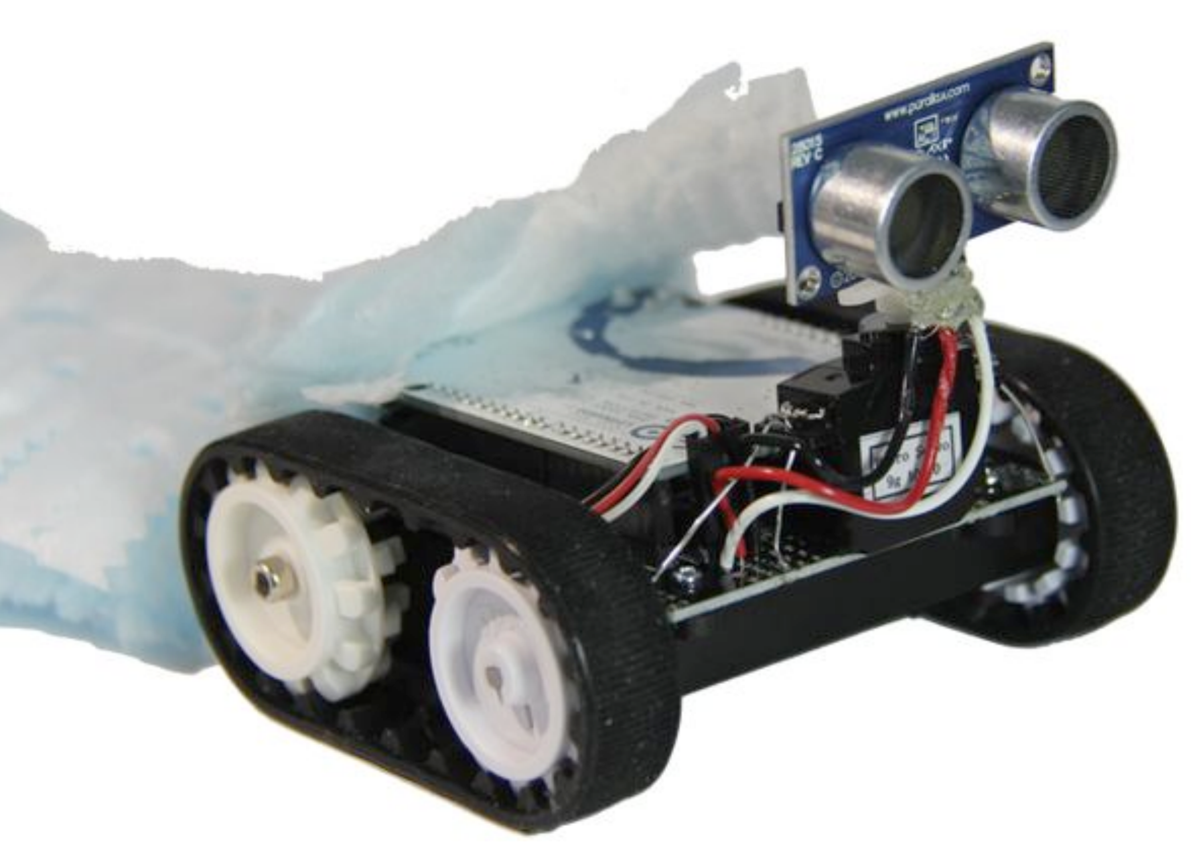
http://www.instructables.com/id/FuzzBot/

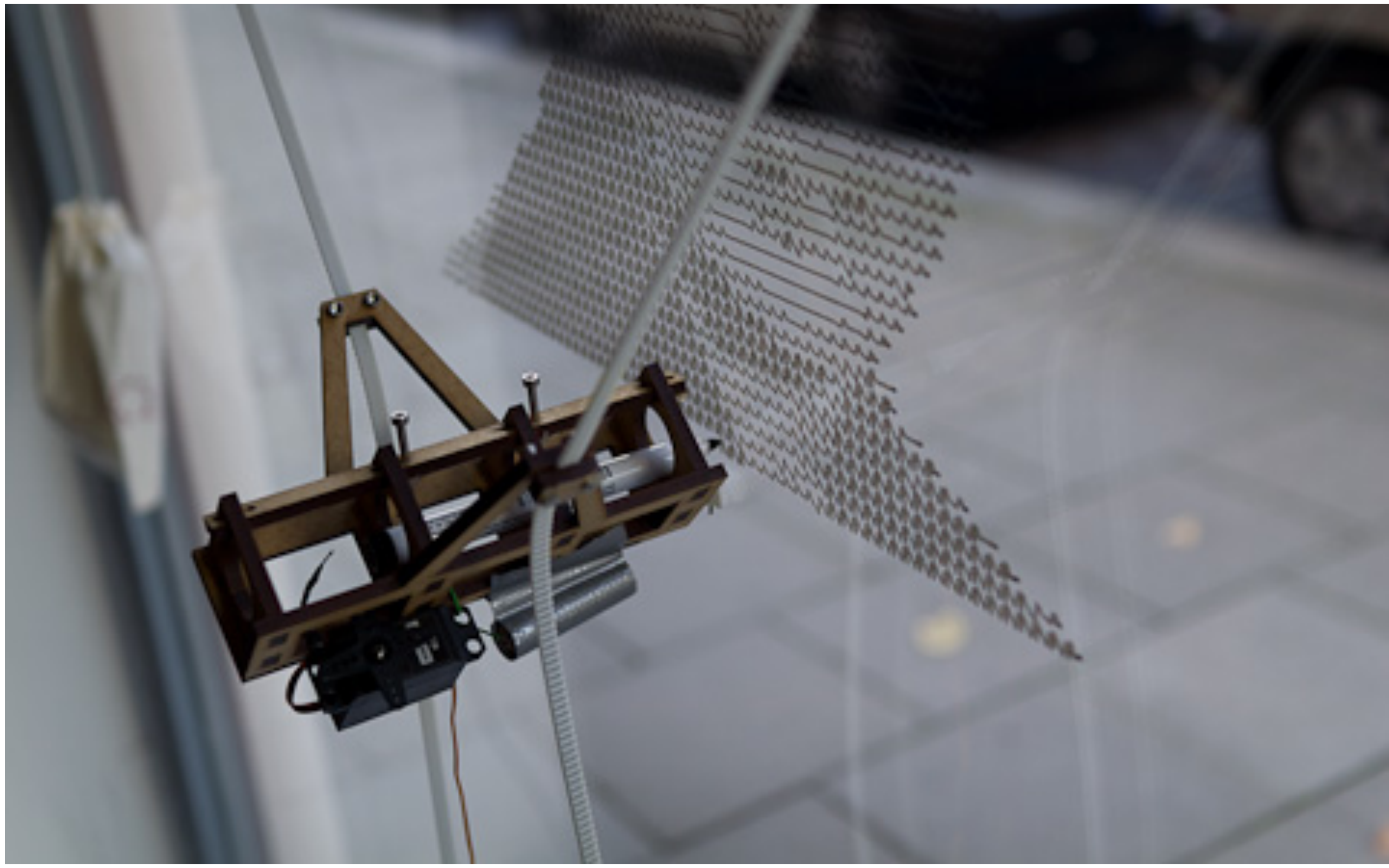


http://tinkerlog.com/2011/09/02/der-kritzler/

www.instructables.com/id/3D-Printed-Quadcopter

http://www.thingiverse.com/thing:248009





http://www.instructables.com/id/FuzzBot/

http://tinkerlog.com/2011/09/02/der-kritzler/

A Trieste

FABLAB

ICTP SCI FAB LAB
scifablab.ictp.it

URBAN
Research
Business
Accelerator
Network
CENTER

Hackerspace

mittelab