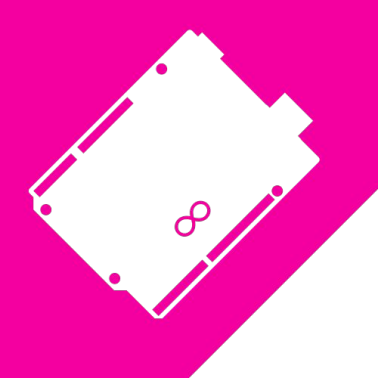
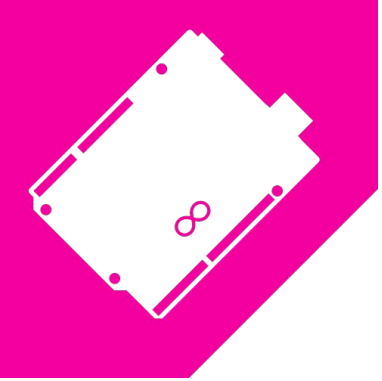


Lezione 2

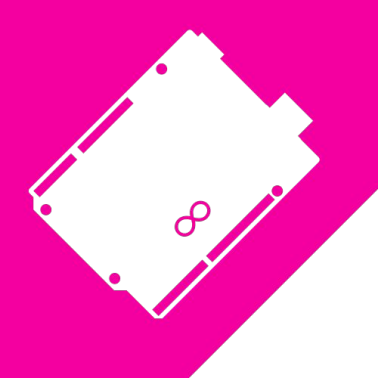
Un corso gentilemente offerto con il sudore e le lacrime di MugRomaTre e Roma Tre

e Magliana



Chiedo Scusa

L'ultima volta mi sono chiesto di chiedere chi fosse da Ingengeria Meccanica.
Sono molto dispiaciuto di questa mancanza, per cui lo chiedo adesso: chi è di Meccanica?



Alcuni numeri di oggi

- 75 prenotati
- 24 hanno la loro arduino (daje)
- Una persona ogni 12.5 si chiama Andrea
- Un ragazzo fa di cognome Peroni *-*

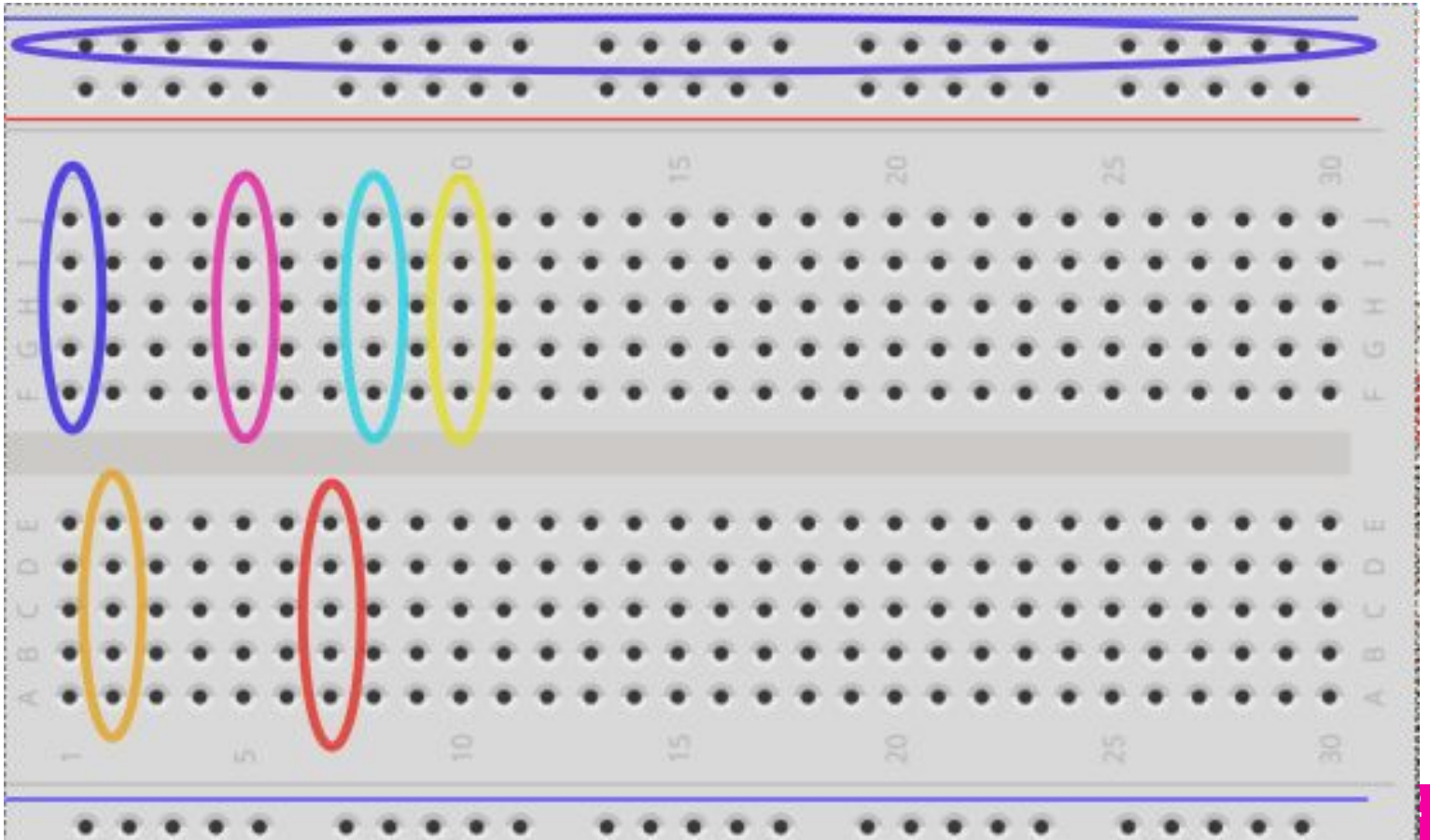
Ospiti di Oggi

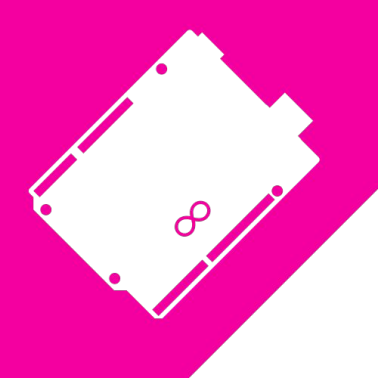
- Andrea Rosati
- Valerio Marta
- La loro bomba (giocattolo)

King africa - Bomba

Breadboard

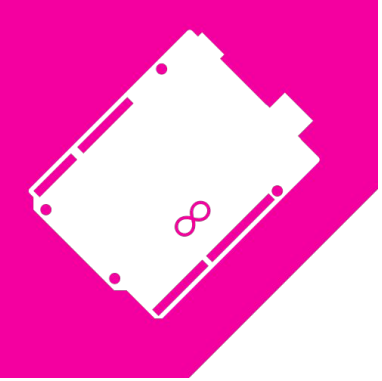
- Ogni riga di 5 socket è connessa elettricamente
- Alcune hanno delle righe dedicate per l'alimentazione (sono quelle blu o rosse)





Sensori eletronici

- Strumenti che misurano una quantità e la ripropongono
  - Analogamente: Generando un voltaggio o una corrente (per es: Potenziometro)
  - Temporalmente: Generando un impulso la cui lunghezza corrisponde alla misura (per es: Ping Sensor)
  - Digitalmente: convertono la misura in numeri binari (per es: accelerometri)

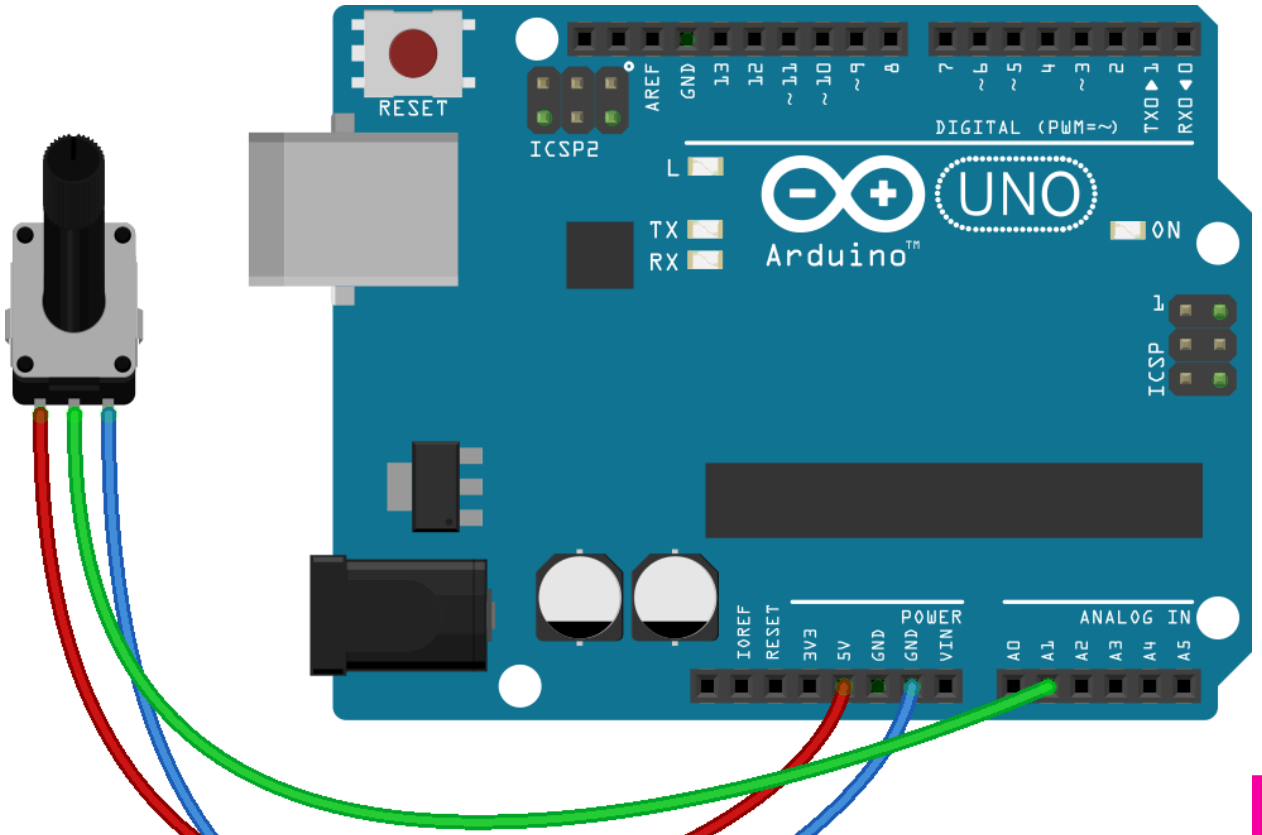


Come leggerli

- Sensori Analogici:
  - int val = analogRead([pin]) → per letture 0v~5v
  - int val = pulseIn([pin], [HIGH | LOW]) → per misurare impulsi
- Sensori Digitali:
  - i2c, spi, usart. Questi protocolli li vedremo più avanti

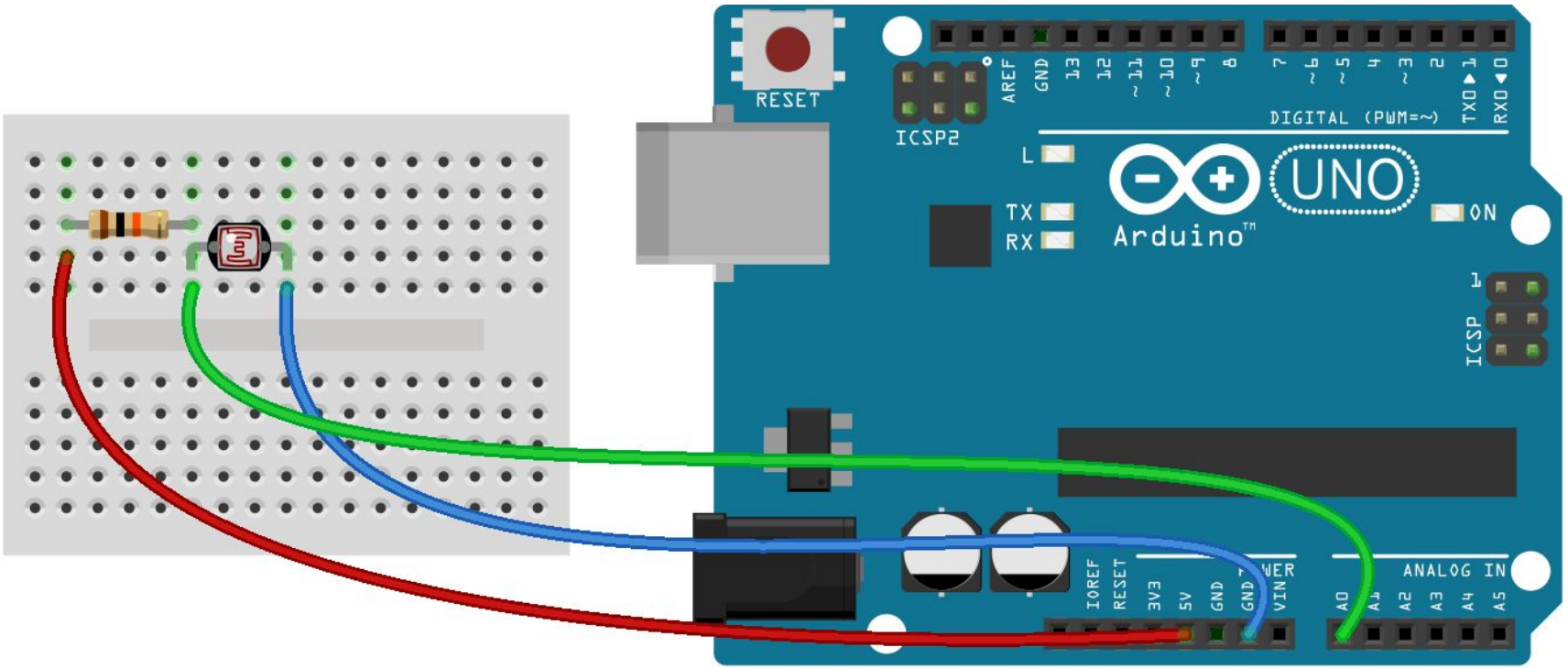
Trimmer

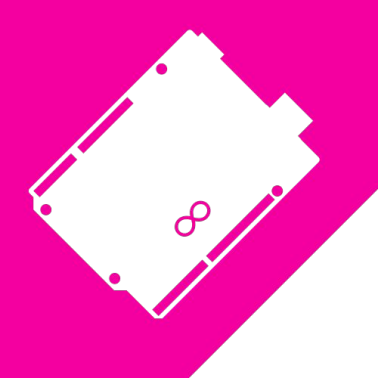
- Una resistenza variabile con 3 pin
- Quando c'è una tensione ai due piedini esterni, sul piedino centrale si legge una tensione intermedia tramite analogRead



Fotoresistenza

- Una resistenza dipendente dalla luce, quando è illuminata 1kΩ, quando è al buio 15kΩ
- Misurabile con un partitore di tensione e analogRead



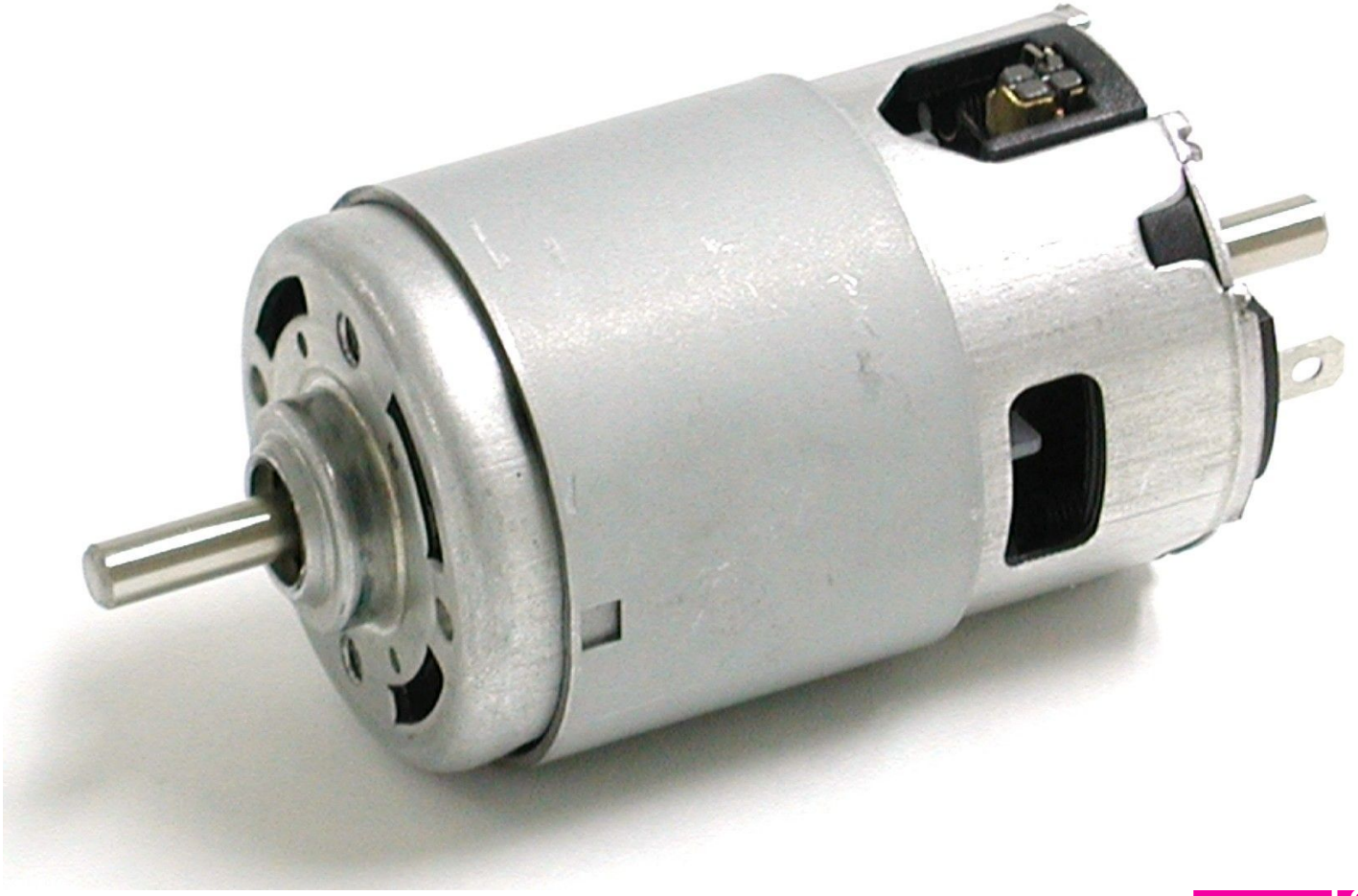


Attuatori

- Traduce un segnale in un movimento
  - Rotazionale, lineare, vibrante
- Tantissimi tipi, con scopi differente e caratteristiche differenti
- Alto consumo elettrico
  - Raramente possono essere collegati direttamente con arduino

Motori DC

- I più semplici, ma non offrono nessun controllo fra segnale → azione
- Si alimentano applicando un voltaggio

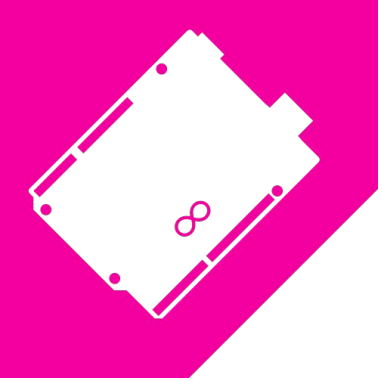


ServoMotori

- Input: PWM Output: angolo o velocità di rotazione
- Controllabili da arduino! (circa)
  - Vanno alimentati
- Usati in robotica hobbistica
- 3pin: Vcc, Gnd, Sig

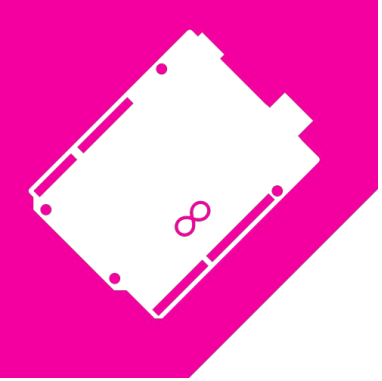
Codice servo

```
#include <Servo.h>
     //Libreria per controllare un servo
Servo myservo;
     //Crea l'oggetto Servo per comandare il Servo

void setup() {
  myservo.attach(9);
     //attacca il servo al pin 9
  myservo.write(45);
     //muovi il braccetto a 45°
  //delay(100);
     //a volte serve aspettare che il braccetto arrivi in posizione
}
```



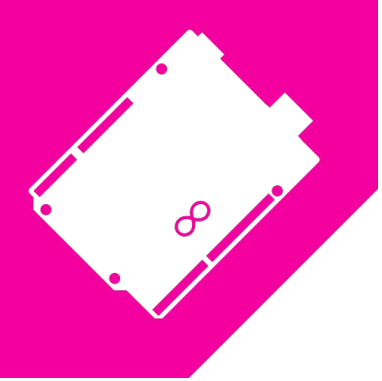
Stepper

- Sono motori capaci di muoversi in “passi”
- Si controllano con una sequenza impulsi per far avanzare di un passo in avanti o indietro
  - Servirà una libreria e un circuito per pilotarli
- Possono sviluppare molta forza
- Non li vedremo oggi :D



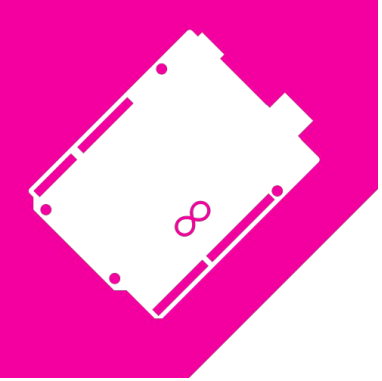
“Non conosco c++ / arduino / elettronica!”

- Studia la teoria
  - Chi non ha mai visto C o C++ → https://learnxinyminutes.com/docs/c/
  - Chi si vuole rinfrescare C++ → https://learnxinyminutes.com/docs/c++/
  - Tutti → http://www.arduino.cc/en/Reference/HomePage



“Non conosco c++/arduino/elettronica!” - 2

- Studia gli esempi
  - https://www.arduino.cc/en/Tutorial/HomePage

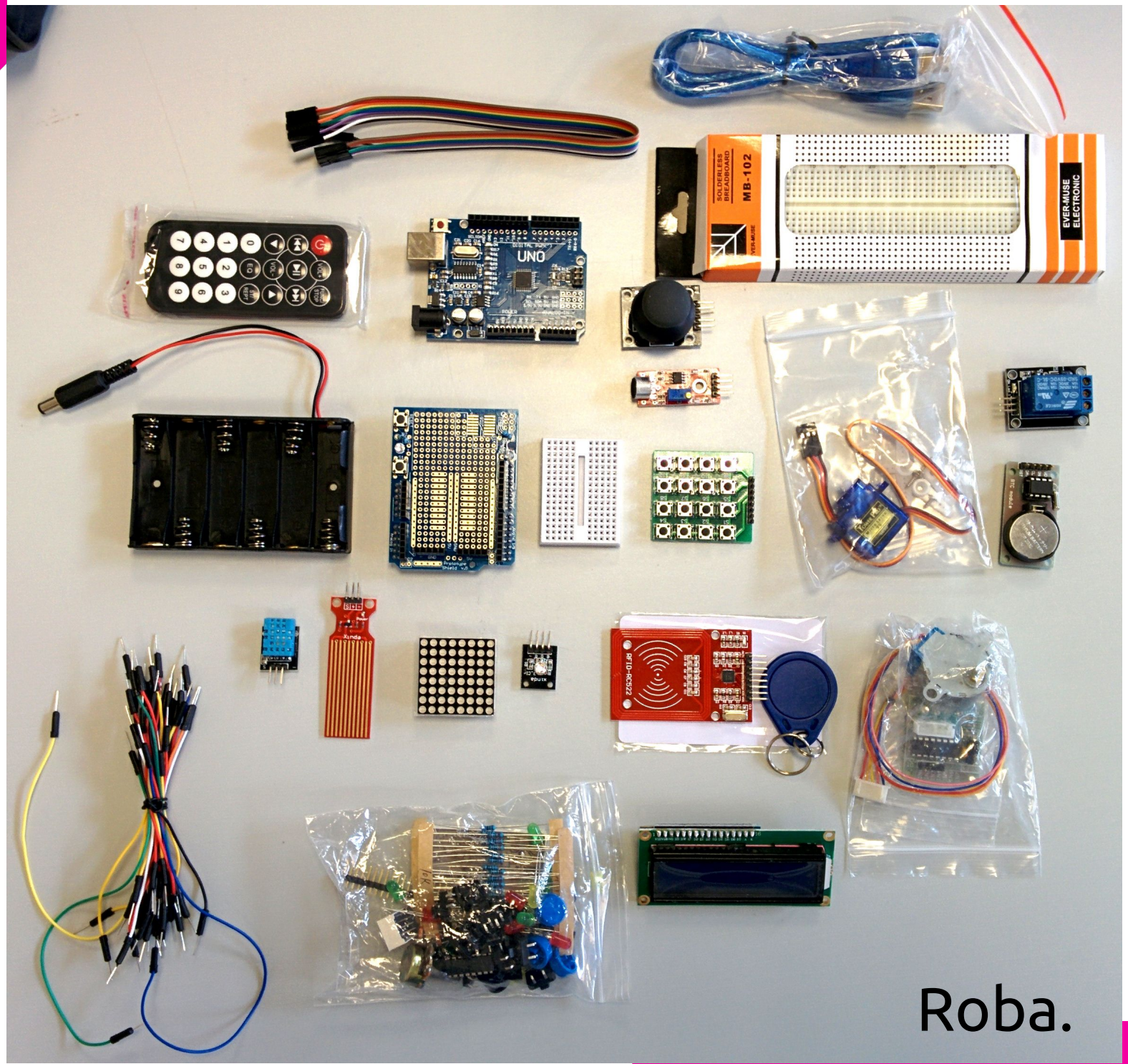


“Non conosco c++/arduino/elettronica!” - 3

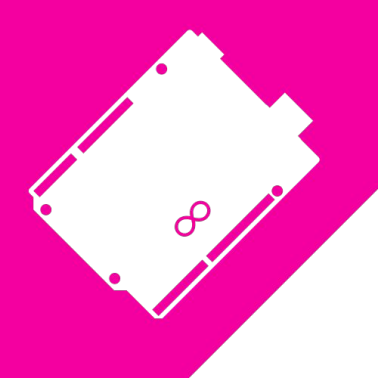
WARNING per chi è più pratico di programmazione:

- Il compilatore accetta fino al C++11 MA
  - Non esiste una stdlibrary
    - un possibile rimpiazzo
  - Libc non è conforme allo standard
    - per cui il manuale ogni tanto va visto
  - L'architettura avr non è coperta perfettamente dal linguaggio, per cui alcune estensioni non standard sono usate per fornire gli interrupt e leggere la memoria
    - Vedi manuale di prima
- L'elettronica è sempre più complicata di quanto ci si aspetti

Cosa c'è nel nostro kit?

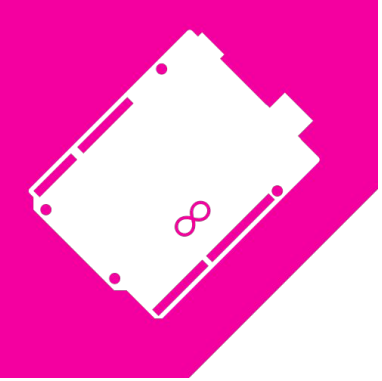


Roba.



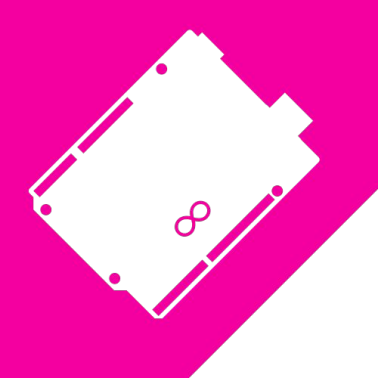
Cosa proviamo oggi?

- Blink
- SerialAnalogRead
- Knob
- Wave
- PhotoServo
- PhotoServoClock



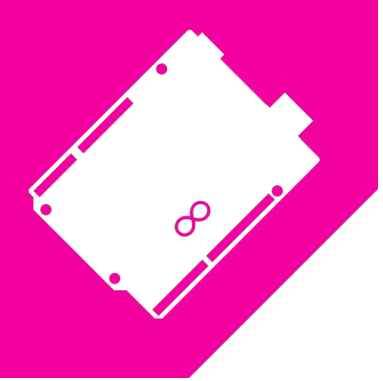
Blink

- Voglio accendere e spegnere il led sul pin 13, acceso per 700ms e spento per 350ms
- Hint: delay(ms) aspetta ms millisecondi



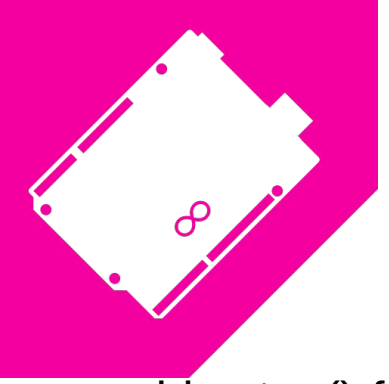
Blink Soluzione

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(700);               // wait for 700 ms
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(350);               // wait for 350 ms
}
```



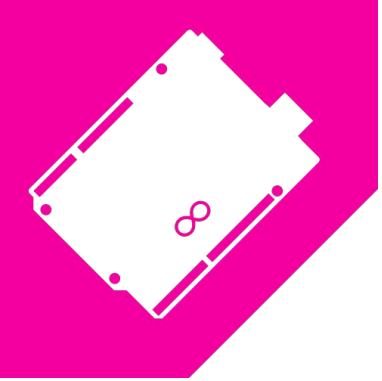
SerialAnalogRead

- Voglio stampare sul monitor seriale il voltaggio letto sul pin A0
- Ricordati di aprire il monitor seriale!
- Sul pin A0 potrei mettere il trimmer o il la fotoresistenza
- Hint: analogRead(pin) tutta la vita
- Hint: Serial.println(val)



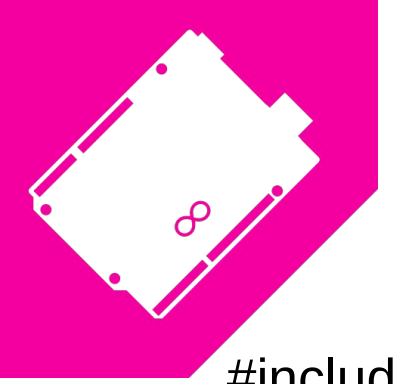
SerialAnalogRead Soluzione

```
void setup() {
  // initialize the serial communication:
  Serial.begin(9600);
  // remember to select this same velocity on the serial monitor
}

void loop() {
  // send the value of analog input 0:
  Serial.println(analogRead(A0));
  // wait a bit for the analog-to-digital converter
  // to stabilize after the last reading:
  delay(2);
}
```



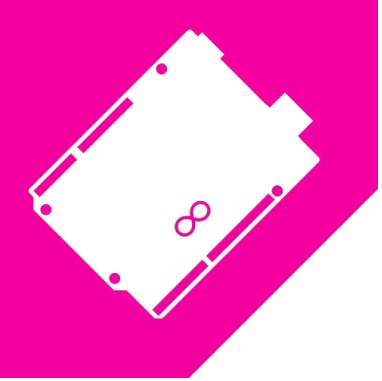
Knob

- Voglio controllare la posizione del servo con un potenziometro
- Servo: marrone → gnd, rosso→ 5V, arancione → pin 9
- Hint: int res= map(value, fromLow, fromHigh, toLow, toHigh) scala value da un range ad un altro.
  - y = map(x, 0, 1023, 0, 180); scala x da 1023 a 180



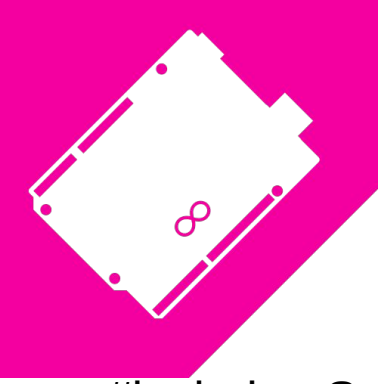
Knob Soluzione

```
#include <Servo.h>

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

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

void loop() {
  int val = analogRead(A0);
      //reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 180);
      // 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
}
```



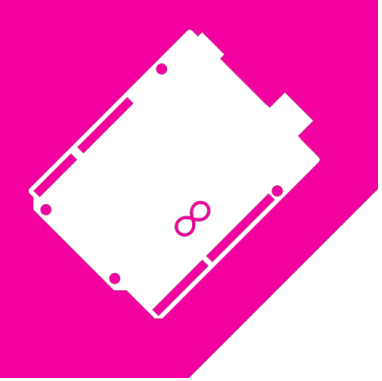
Wave

- Voglio essere salutato dal servo
- Il servo dovrebbe muoversi a destra e sinistra, e poi aspettare 10 secondi prima di salutare di nuovo
- Pro: posso utilizzare la fotoresistenza per farmi salutare solo quando sono davanti alla arduino?
  - Hint: si



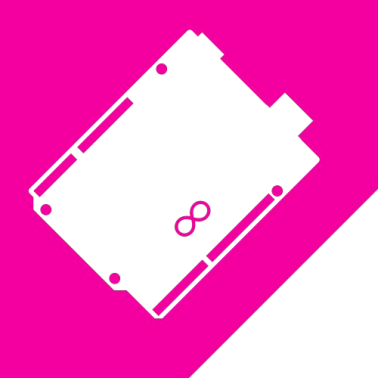
Wave Soluzione

```
#include <Servo.h>

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

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
  myservo.write(90);     //go to middle position
  delay(15);
}

void loop() {
  for(int i=0; i<3; i++){
    myservo.write(0);
    delay(15);
    myservo.write(180);
    delay(15);
  }
  delay(10000);
}
```



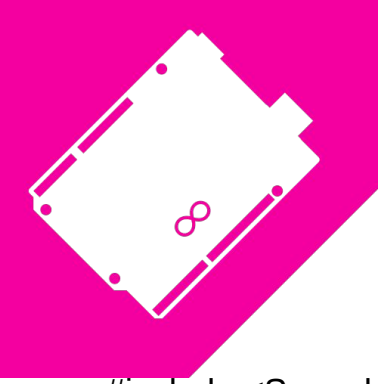
PhotoServo

- Voglio controllare la posizione del servo con la fotoresistenza
- Praticamente uguale all'esempio Knob
  - La fotoresistenza non ha una risposta lineare, ci si può sbizzarrire con map



PhotoServoClock

- Come l'esempio di prima, ma voglio che ogni 3 secondi il braccetto vada a 90 gradi
- Hint: long val = millis() ritorna il numero di millisecondi passati dall'inizio dello sketch
  - Se chiamo millis() più volte, posso sottrarre i risultati per sapere quanti millisecondi sono passati fra due chiamate
  - Posso controllare che siamo passati 3000 millisecondi per fare una azione, e aggiornare un contatore



PhotoServoClock Soluzione

```
#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

long timePoint;
//variable to save a time
void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
  timePoint = millis();
  //record current time
}

void loop() {
  if(millis() - timePoint > 3000){
//have 3000 ms passed? if yes execute this action
     myservo.write(90);
     delay(100);
     timePoint = millis();
     //update timePoint, ready for next tick
  }
  val = analogRead(potpin);            // reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 180);     // 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
}
```