

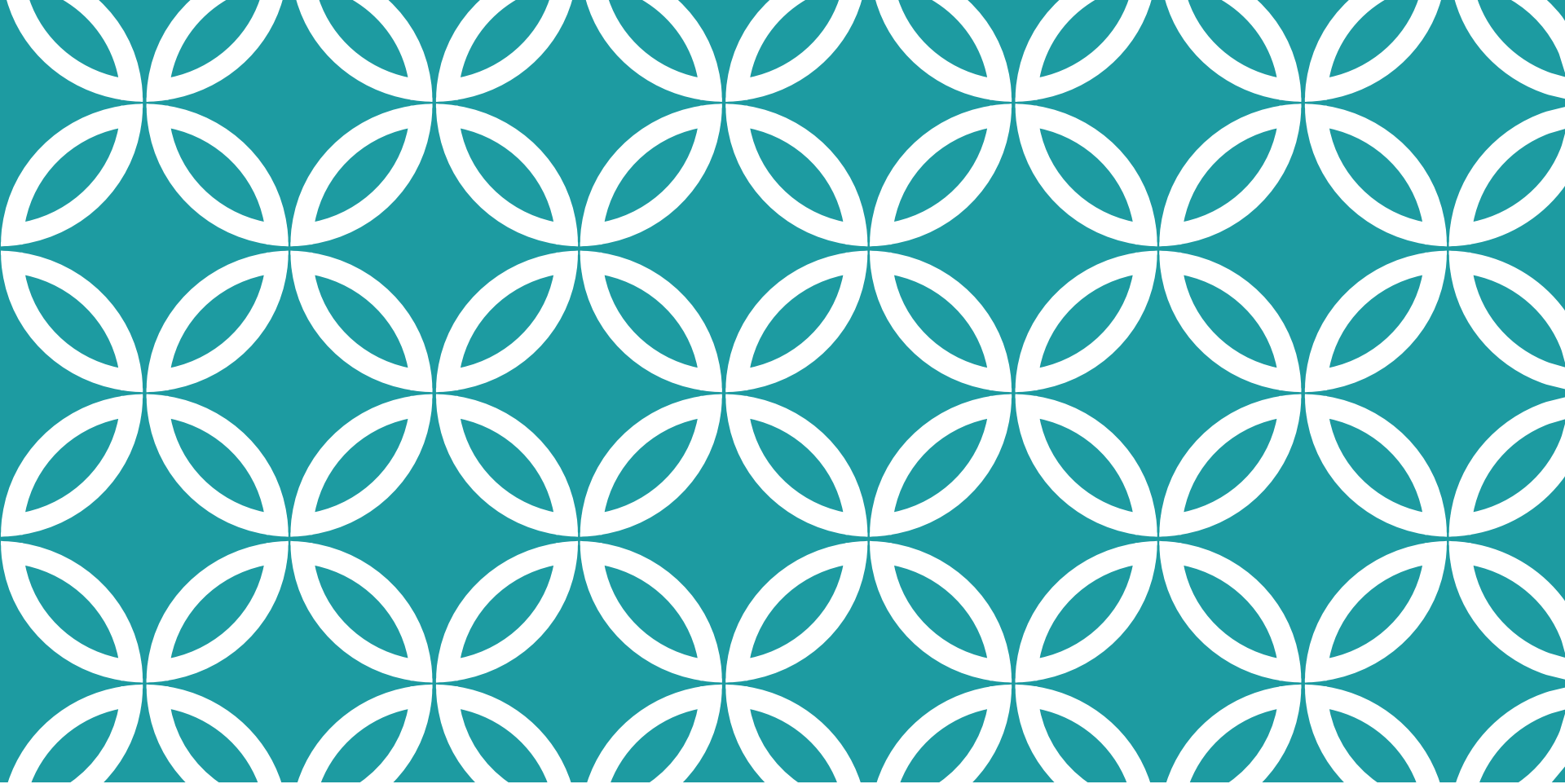
AVVISO

Per la prossima lezione (domani), portate (se ne avete):

- PC portatile
- cavi USB del tipo per stampanti
- Arduini vari.

Installate l'ambiente Arduino nel vostro PC

Non usate la web app!!!



TECNOLOGIE ASSISTIVE PER LA DIDATTICA

Stefano Chessa

Stefano.chessa@unipi.it

INTERNET DELLE COSE (IOT)

In questo spazio nero (1 m²)
ci sono
655.571 miliardi di miliardi
di «things»

INTERNET DELLE COSE

- Molti dei dispositivi connessi non sono utilizzati direttamente da esseri umani
- Sensori e attuatori autonomi con un loro programma di funzionamento
- Osservare, analizzare, capire il mondo circostante e modificare (nel caso di attuatori)

SENSORI INDOSSABILI E AMBIENTALI



frequenza
cardiaca



Sensori per
smart cities



smartwatch



Sensori in
smartphone



Rilevatore di
cadute



Braccialetti
per fitness

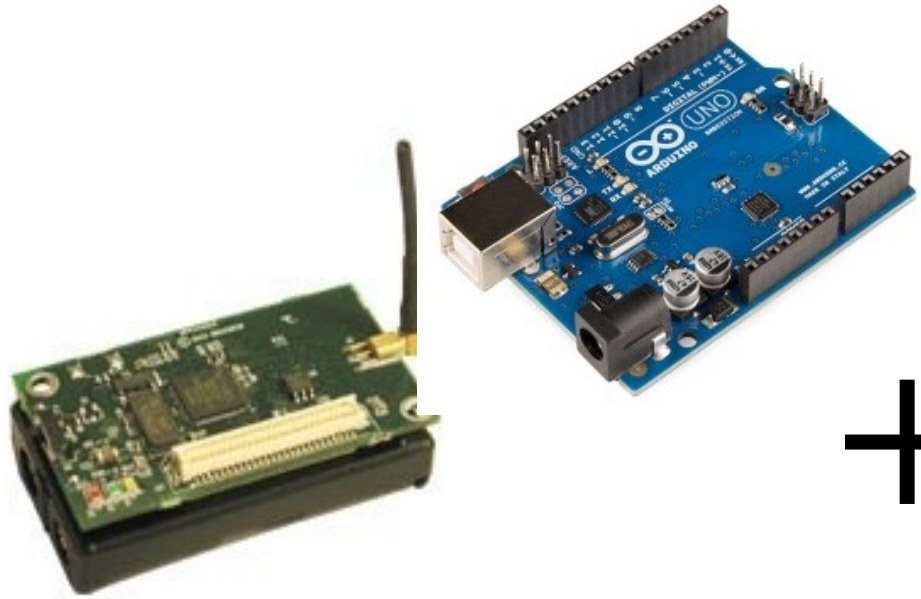


localizzazione

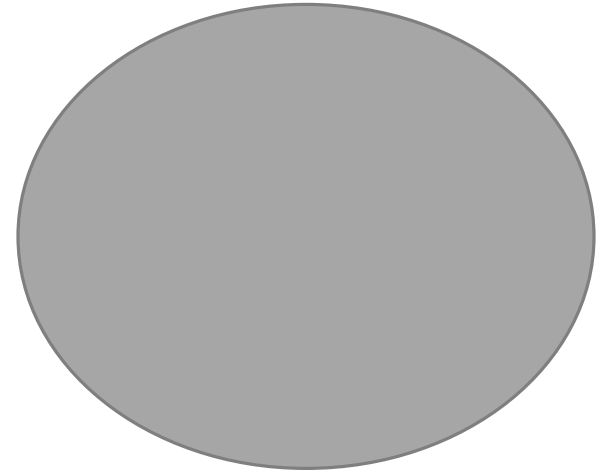


Sensori
medicali

COSA C'È DENTRO?



+



QUINDI I SENSORI...

Sono in grado di capire cosa avviene
intorno a loro,

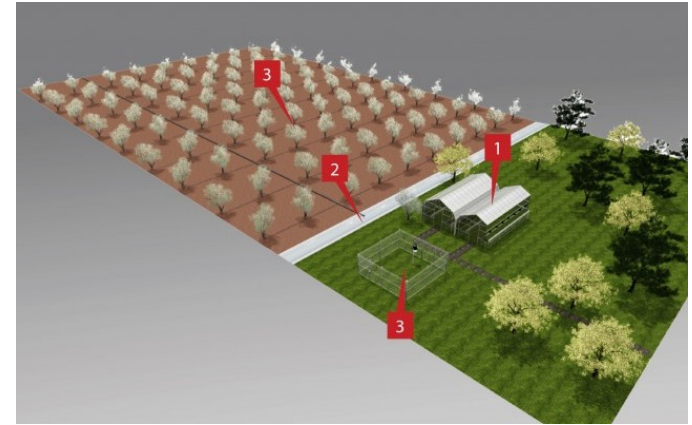
sulla base di informazioni parziali, imperfette...

LA SFIDA...



APPLICAZIONI

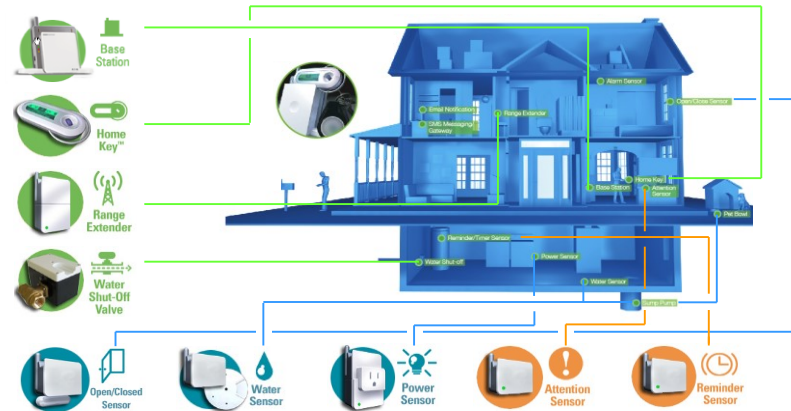
Smart agriculture



Smart cities



E-health



Ambient assisted living
& Smart homes

UN ESEMPIO: TORTOISE@



Riconosce quando
la tartaruga
costruisce il suo
nido



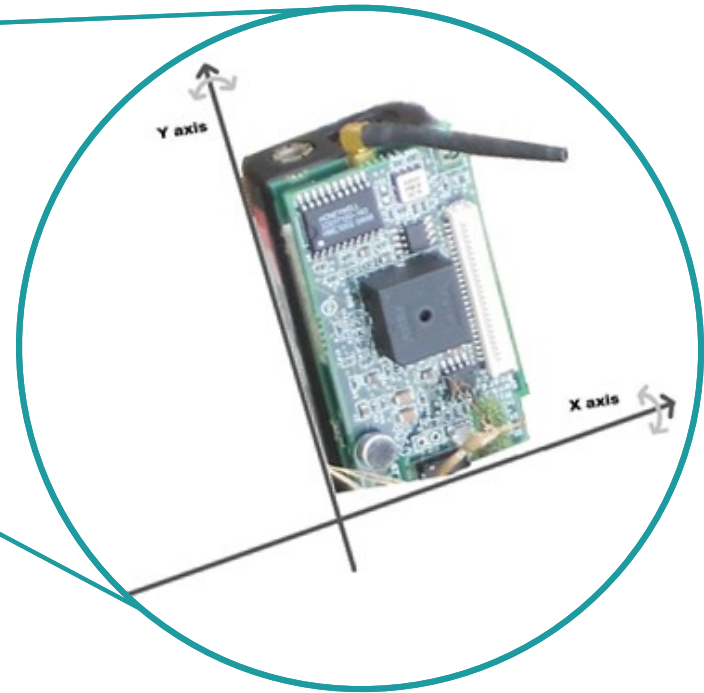
Comunica agli
addetti la
posizione del
nido



Recupero uova e
schiusa in un luogo
protetto

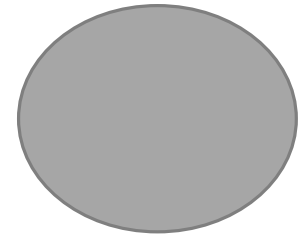
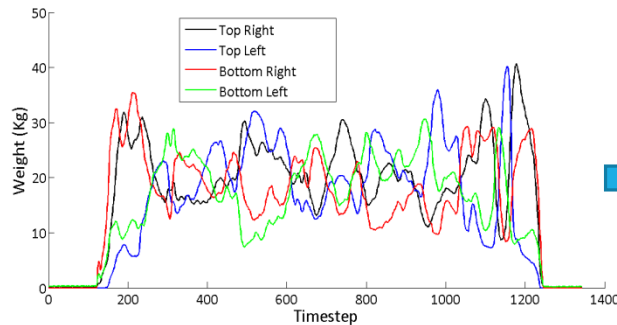
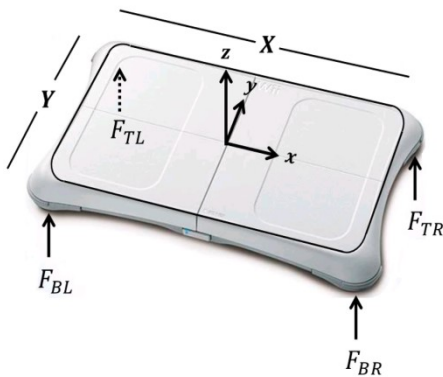


UN ESEMPIO: TORTOISE@



UN ALTRO ESEMPIO...

Bilancia per valutare l'equilibrio delle persone (DOREMI)



Bracciale per consumo calorico, attività e localizzazione

... SPERIMENTAZIONI ...



Magnetic sensors



Pressure sensor



Light sensors

Magnetic sensors

PIR sensor



Doorbell sensor

PIR sensor

Light sensor



Pressure sensor

TV sensor
(On / Off)



Bluetooth pulsometer
in its box

... SPERIMENTAZIONI ...



... SPERIMENTAZIONI ...



RIASSUMENDO...

Con IoT abbiamo la possibilità di rendere «intelligenti» i più svariati oggetti e ambienti

- Dotandoli di sensori
- Dotandoli di attuatori
- Connettendoli tra loro

Arduino è un esempio di tecnologia che possiamo usare per fare in modo semplice questo lavoro

- Vedremo tra poco come Arduino funziona

Ma che relazione ha con problemi specifici quali l'autismo, il declino cognitivo, la disabilità in generale e altre malattie?

RIASSUMENDO...

In altri termini:

OK, l'loT è molto interessante,

ma ...

a che relazione concreta ha con le
tecnologie assistive?

UN ESEMPIO LEGATO ALL'AUTISMO

Un'informazione importante per lo studio dell'autismo nei bambini (e non solo) riguarda lo studio delle abitudini in condizioni «libere»

Ad esempio:

- sapere come usa gli spazi in casa
- come interagisce con gli oggetti che ha intorno
- quale è la sua «routine»
- ... e mettere tutto questo anche in relazione con il suo stato (agitazione, ansia, calma etc..) e con altri parametri (ad esempio qualità del sonno etc.)

UN ESEMPIO LEGATO ALL'AUTISMO

- queste informazioni possono essere usate per vari scopi:
 - diagnosi
 - progettare gli spazi domestici
 - Migliorare le condizioni ambientali

UN ESEMPIO LEGATO ALL'AUTISMO

- I disturbi dello spettro autistico comportano diverse differenze nel comportamento
- Ci concentriamo sul gioco nei bambini:
 - Le modalità di gioco nei bimbi con disturbi dello spettro autistico sono spesso differenti dagli altri bambini
 - compiono azioni quali portare giocattoli in giro senza giocarci mai
- Analizzare il gioco è un utile metodo di diagnosi

UN ESEMPIO LEGATO ALL'AUTISMO

Per questi studi è utile osservare il gioco del bambino nel suo ambiente. D'altra parte:

Non si può spostare uno specialista in casa del bambino, la sua sola presenza altera l'ambiente sotto osservazione

Alcuni ospedali hanno stanze attrezzate per osservare i bambini con relativa discrezione, ma si tratta comunque di soluzioni parziali:

- Brevi periodi di osservazione
- Si tratta comunque di un territorio alieno, l'osservazione in casa è differente

In casa si potrebbero installare telecamere e microfoni per poter fare l'analisi da remoto

- Difficilmente (per nulla) accettabile per una famiglia

UN ESEMPIO LEGATO ALL'AUTISMO

Con IoT cambia la prospettiva, si possono usare sensori che:

- siano installati nella casa, negli spazi e negli oggetti che il bambino usa
- misurino parametri tollerabili per la famiglia che li installa (no video o audio)
- Siano combinati con algoritmi che integrano questi dati e individuano le attività e stati del bambino (gioco, riposo, rabbia, calma etc..) e che li associno agli spazi e agli oggetti nel tempo.

UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLE

Un lavoro sviluppato con una Tesi di Laurea Magistrale in informatica Umanistica

Idea:

- sensorizzare alcuni giocattoli per rilevare il loro movimento
- Dal movimento rilevare automaticamente il tipo di azioni effettuate dal bambino
- Rende possibile «osservare» da remoto il gioco, senza interferire nell'ambiente del bambino con elementi invasivi

UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLI

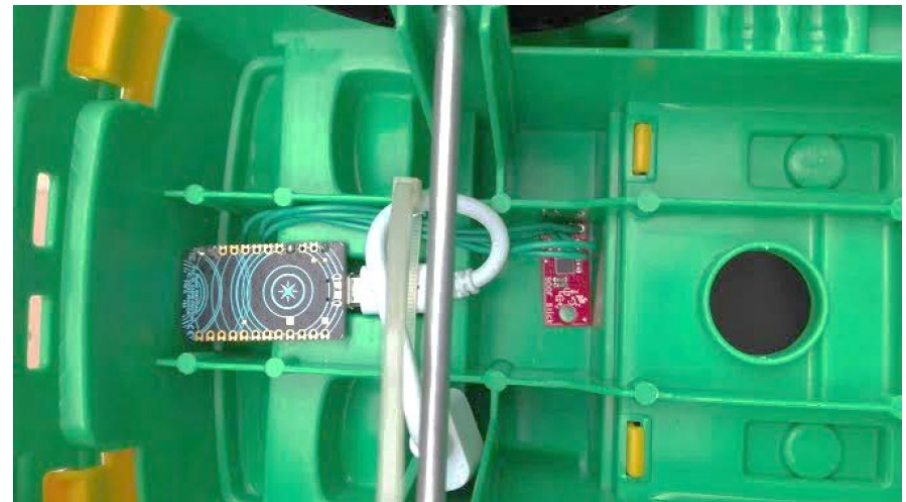
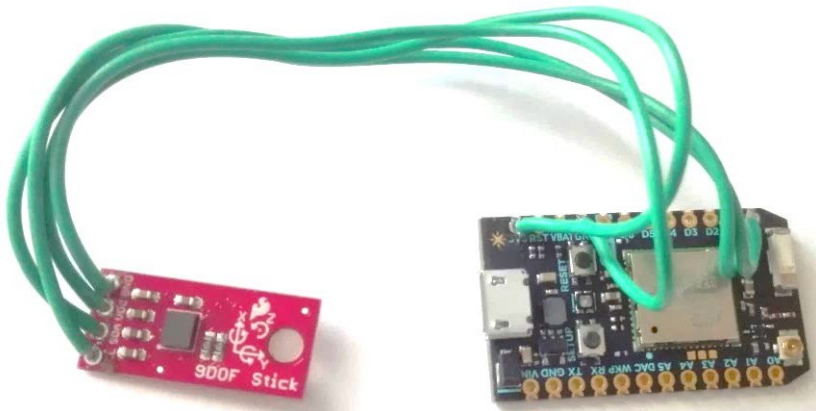
Due giocattoli:



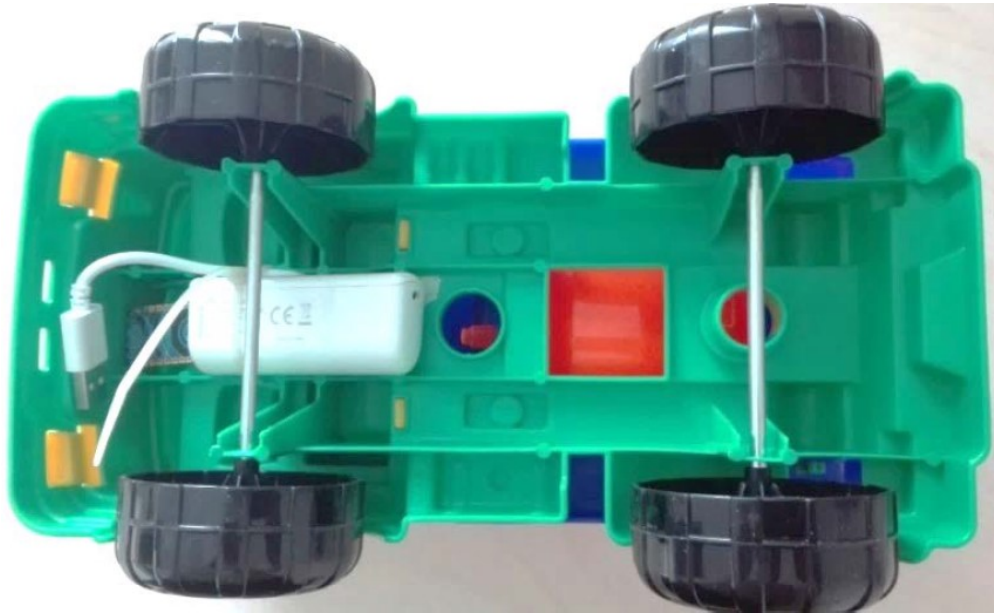
UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLI

Giocattoli sensorizzati con un particle photon:

- programmato con piattaforma Arduino
- Incapsula accelerometri, giroscopio e bussola (magnetometro)

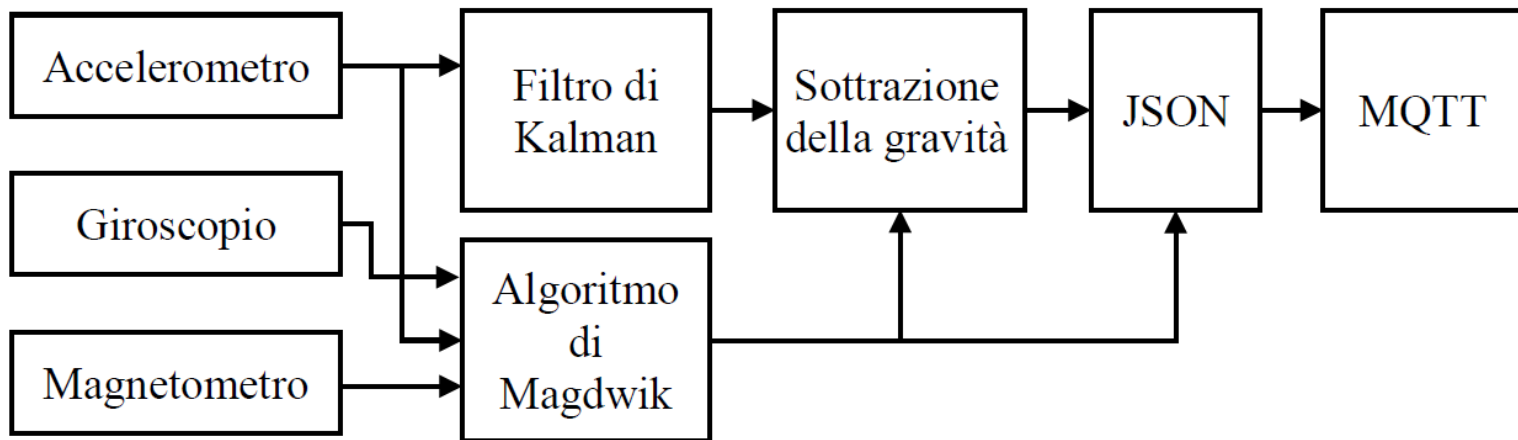


UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLI



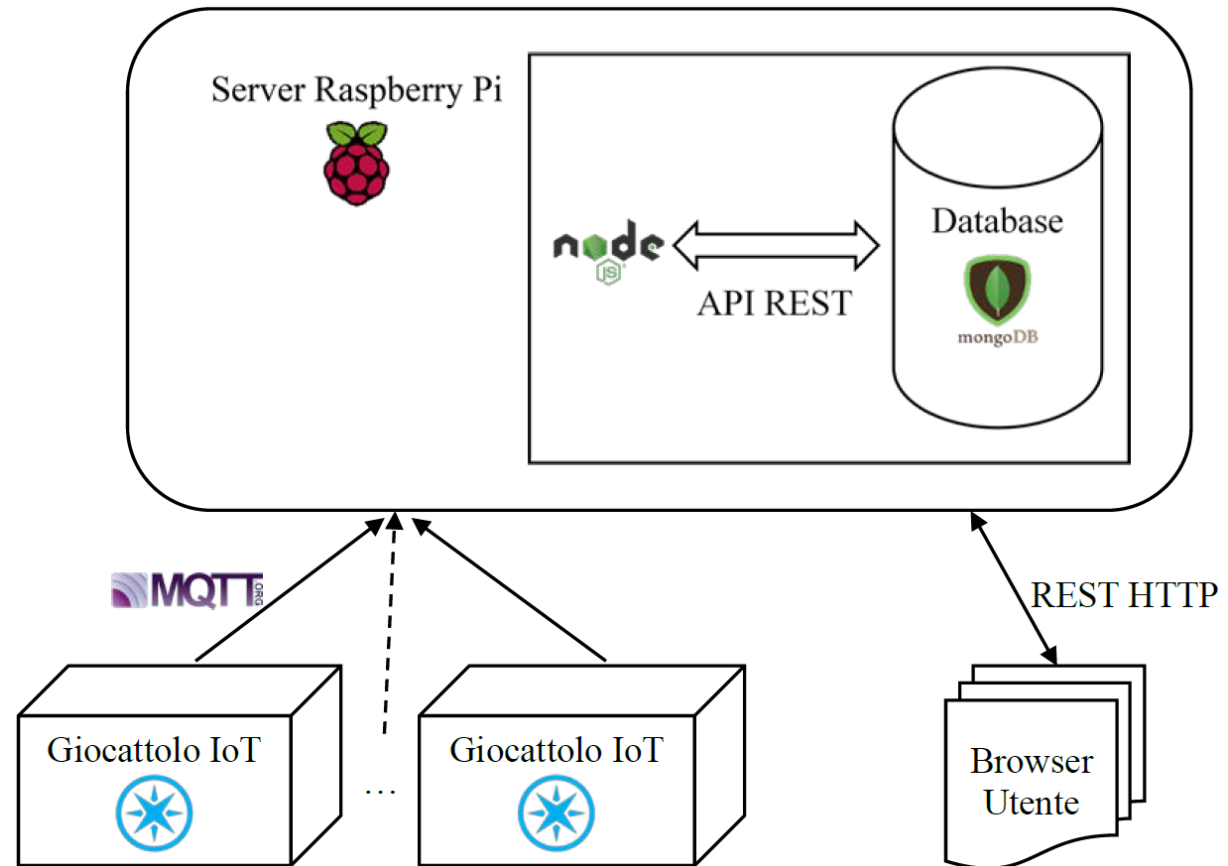
UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLE

- Un gateway raspberry PI e un server con storage (MongoDB)
- E un sistema di elaborazione del segnale (piuttosto sofisticato) per determinare, ad ogni istante, l'orientamento del giocattolo



UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLI

Architettura software

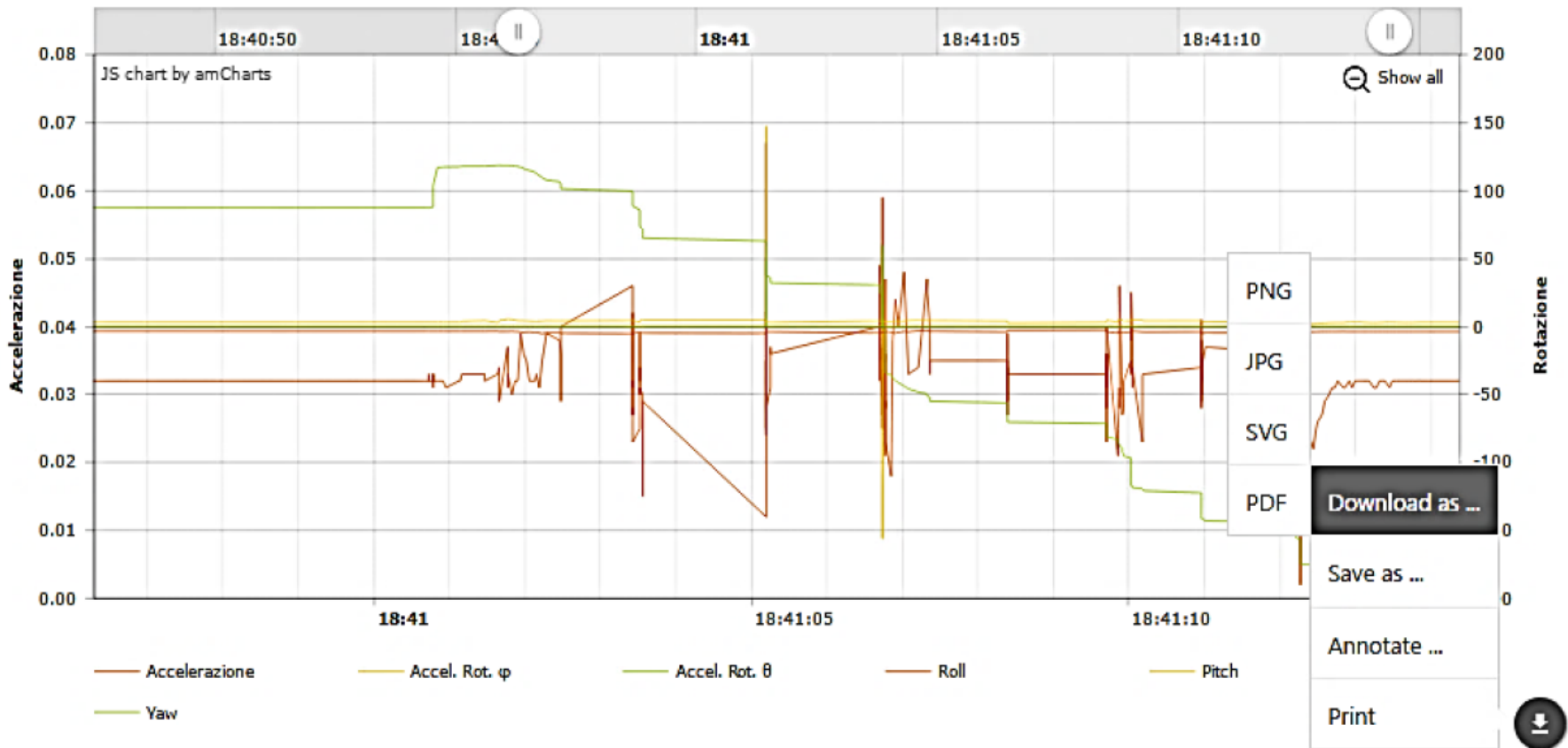


RISULTATI

Database

Rimuovi

Autocarro



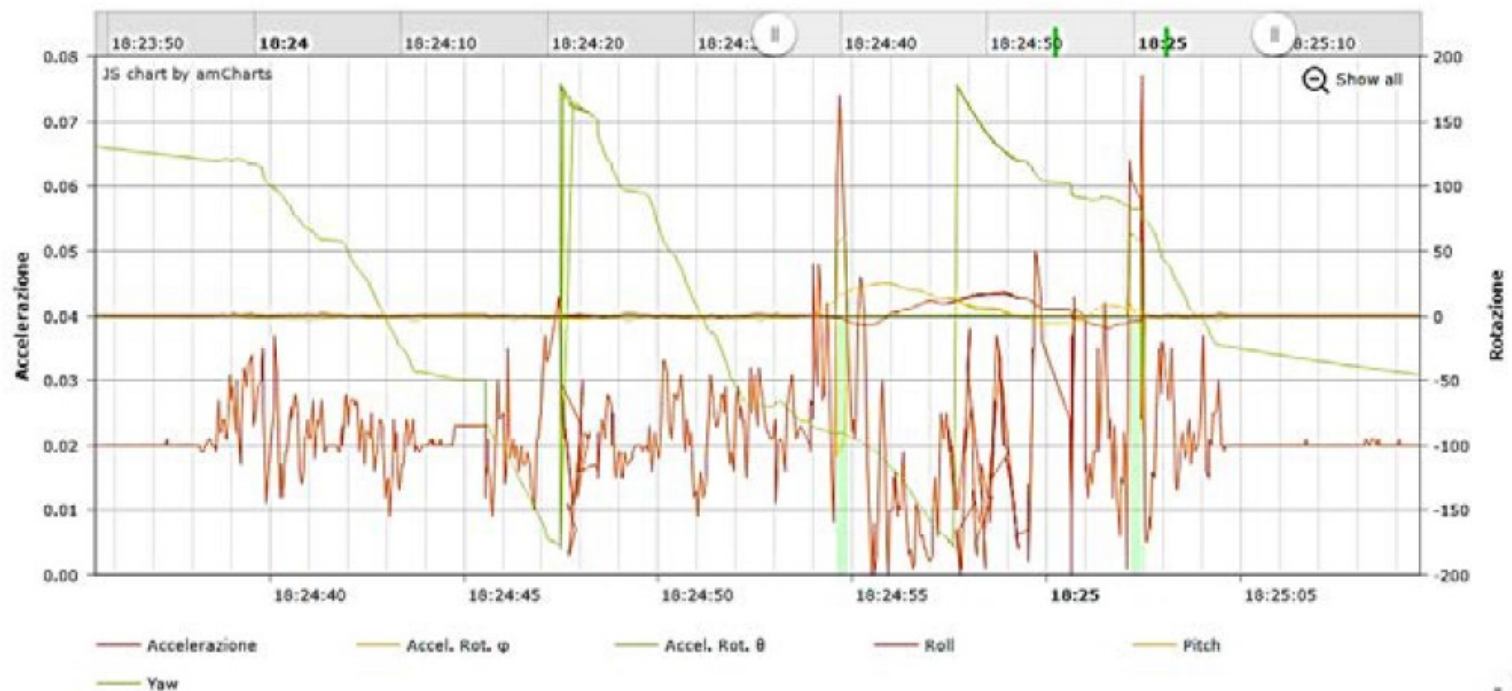
RISULTATI: RULLAGGIO SULLA PISTA

192.168.1.11:3000/view?data=2017-06-24T16%3A23%3A49.056Z

Università di Pisa Home Pazienti Sessione Database

Database Rimuovi

Super Wings



UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLE

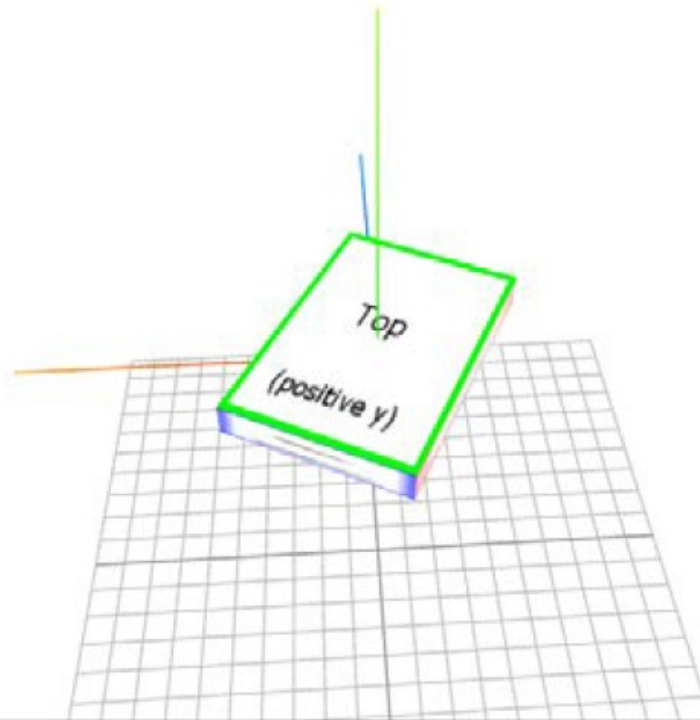
Autocarro



UN ESEMPIO CONCRETO: SENSORIZZAZIONE GIOCATTOLE

Autocarro

Segnale -33dBm



ALCUNE CONSIDERAZIONI

Tutto questo si può fare ora, ma non esistono oggetti già pronti sul mercato.

Sono tecnologie alla portata di tutti ma che non hanno avuto ancora questo impiego, per due motivi:

1. Domanda e offerta non si sono incontrate:
 - I medici spesso non sanno cosa la tecnologia può offrire
 - Gli informatici non sanno a quali problemi possono applicare le loro conoscenze
2. Si tratta spesso di soluzioni che vanno progettate ad hoc, sulla base di indicazioni precise
 - Non è un mercato di massa

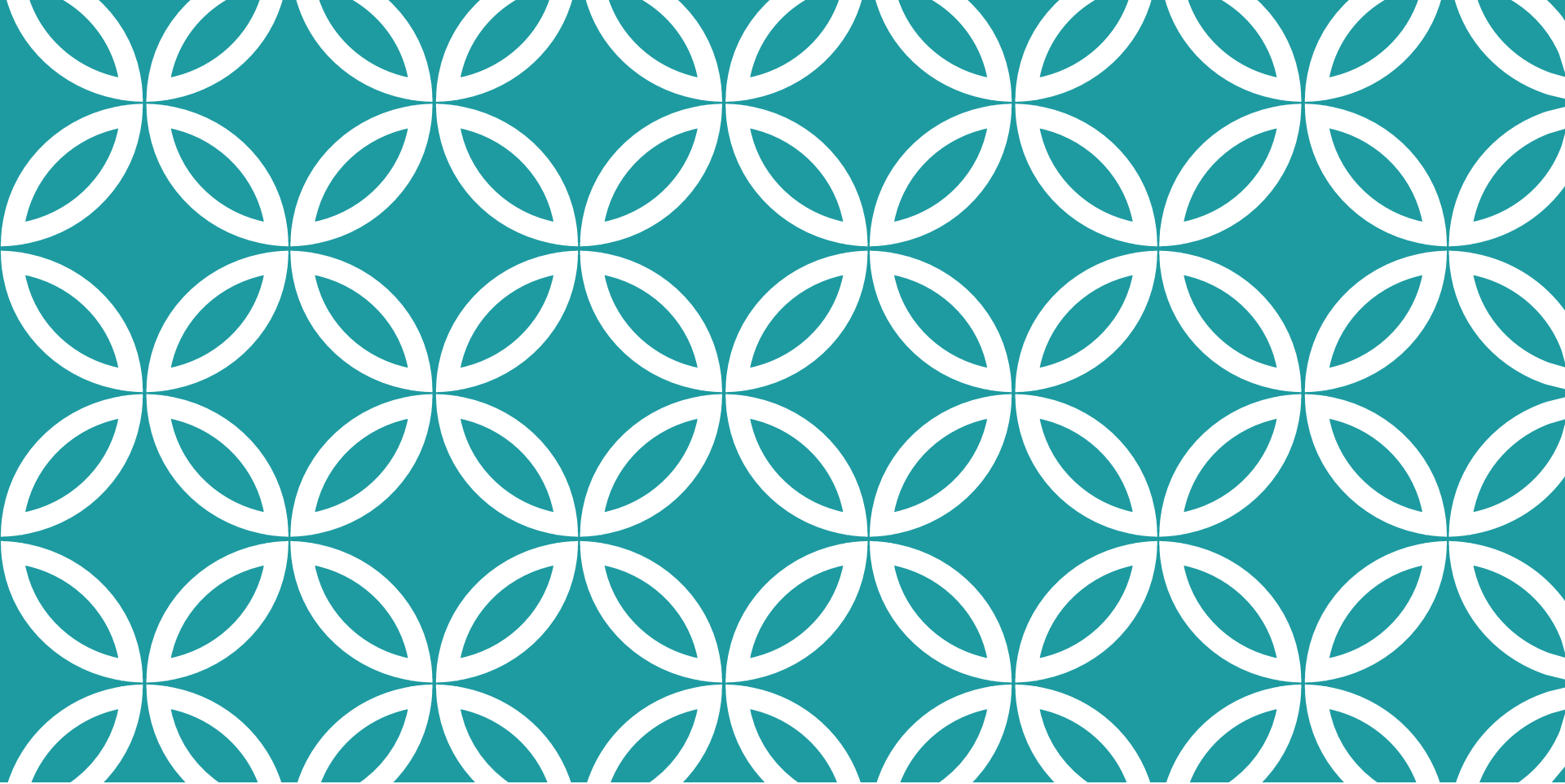
ALCUNE CONSIDERAZIONI

Per questo motivo una figura di un «informatico umanista», vicino ai problemi e vicino alle tecnologie è molto utile

Quindi la risposta alla domanda:

«a che serve l'IoT per l'autismo o per altre patologie, disabilità etc.?»

in futuro la dovrete dare voi.



ARDUINO |

THE IDEA OF ARDUINO

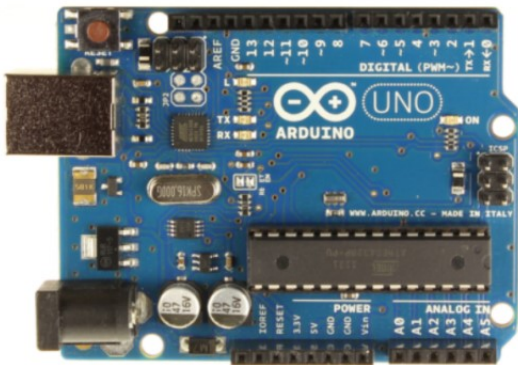
Arduino is an **open-source electronics prototyping platform** based on flexible, **easy-to-use** hardware and software.

It's intended for artists, designers, hobbyists and anyone interested in **creating interactive objects or environments.**

“ARDUINO”

“Arduino” concept comprises:

- Device



- IDE

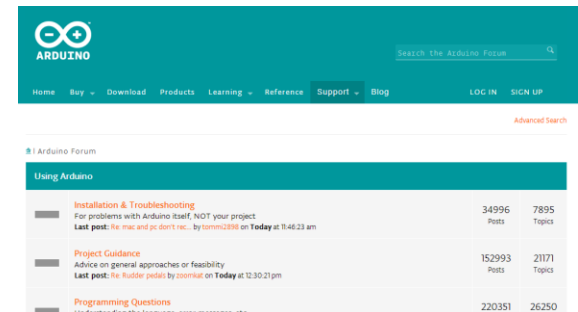
A screenshot of the Arduino IDE (version 1.5.5-r2) showing a sketch named "sketch_feb22a". The code is as follows:

```
sketch_feb22a
void setup() {
  // put your setup code here, to run once:
}

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

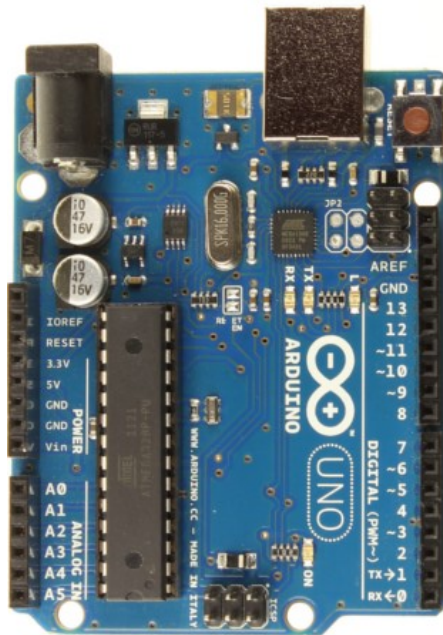
The IDE interface includes a menu bar (File, Edit, Sketch, Tools, Help) and a status bar at the bottom indicating "Arduino Uno on COM12".

- Forum



ARDUINO HARDWARE

UNO

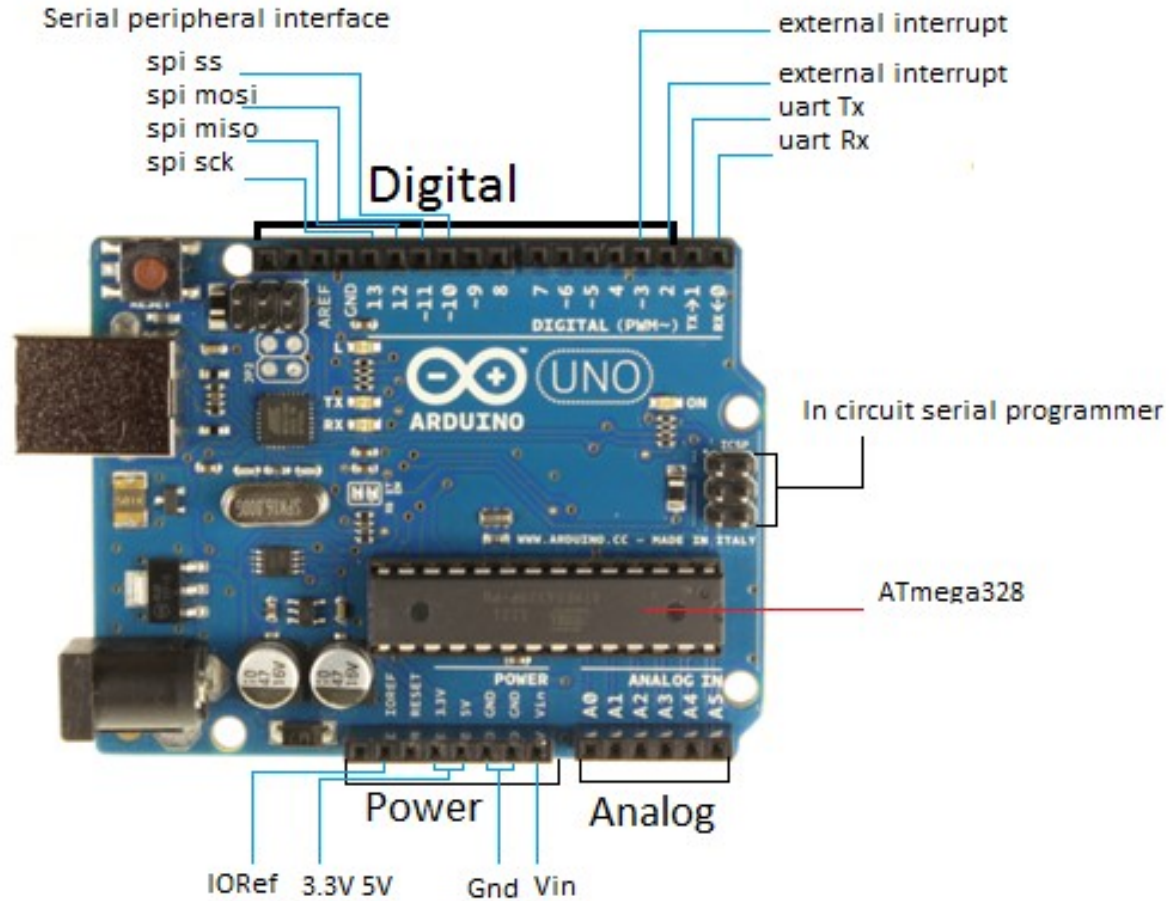


YÚN



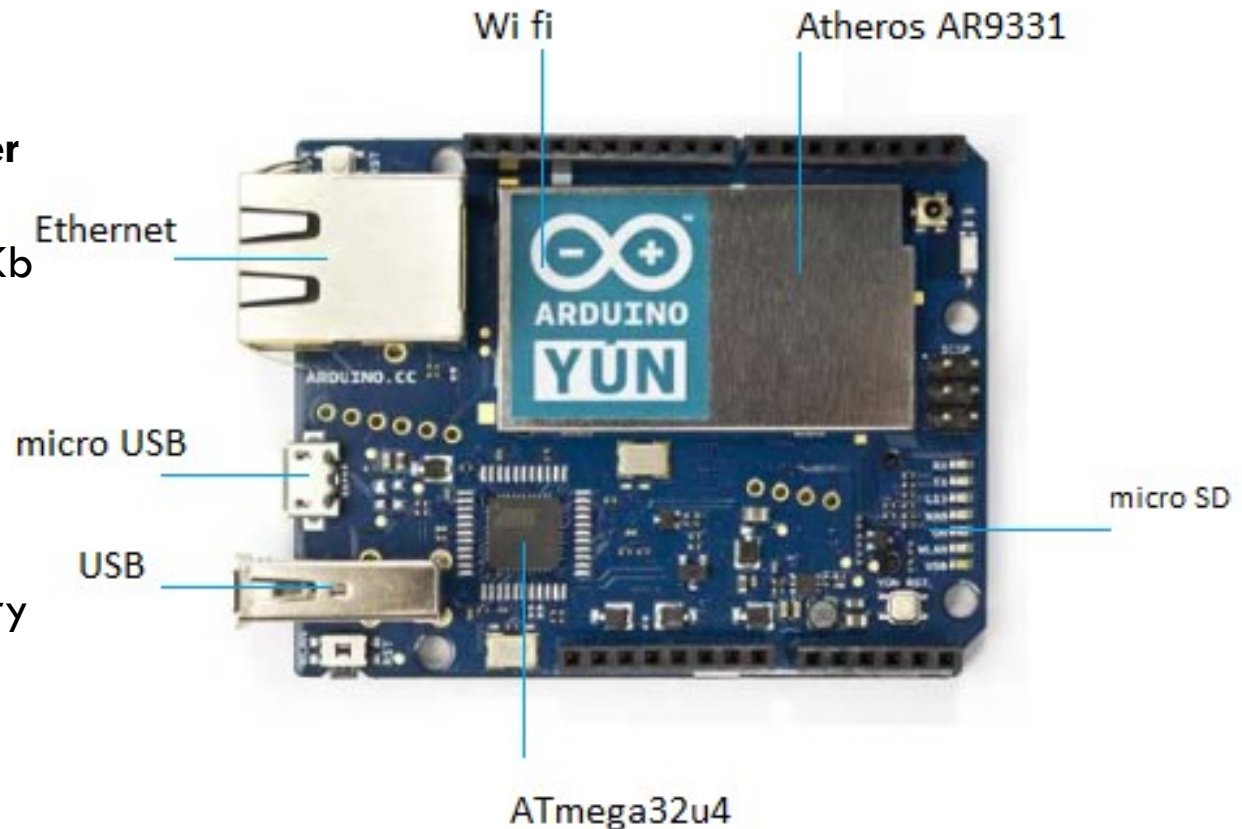
ARDUINO UNO

- **AVR Arduino microcontroller**
 - **Atmega328**
 - SRAM 2KB
 - EEPROM da 1KB
 - Flash memory 32 KB



ARDUINO YÚN

- **AVR Arduino microcontroller**
 - **Atmega32u4**
 - Flash memory 32 Kb
 - SRAM 2.5KB
 - EEPROM 1KB
- **Linux microprocessor**
 - **Atheros AR9331**
 - RAM 64 MB DDR2
 - 16MB Flash memory



SENSORS, ACTUATORS AND SHIELDS

Sensors

- Accelerometer module
- Tilt module
- Button module
- Linear potentiometer
- Rotatory potentiometer
- Joystick module
- Hall sensor module
- LDR sensor module
- Temperature sensor module
- Touch sensor module
- Humidity sensor
- GPS module
- Piezo

Actuators

- Led (red, blue, green, yellow)
- Power Led module
- Servo motors
- Stepper motors
- Paper panel

For high power

- Mosfet module
- Relay module

Shields

- Bluetooth
- GSM
- Zigbee

BLUETOOTH AND XBEE MODULES

- Bluetooth® version 2.1 module
- It supports the EDR (Enhanced Data Rate)
- Delivers up to a 3 Mbps data rate for distances up to 20 meters



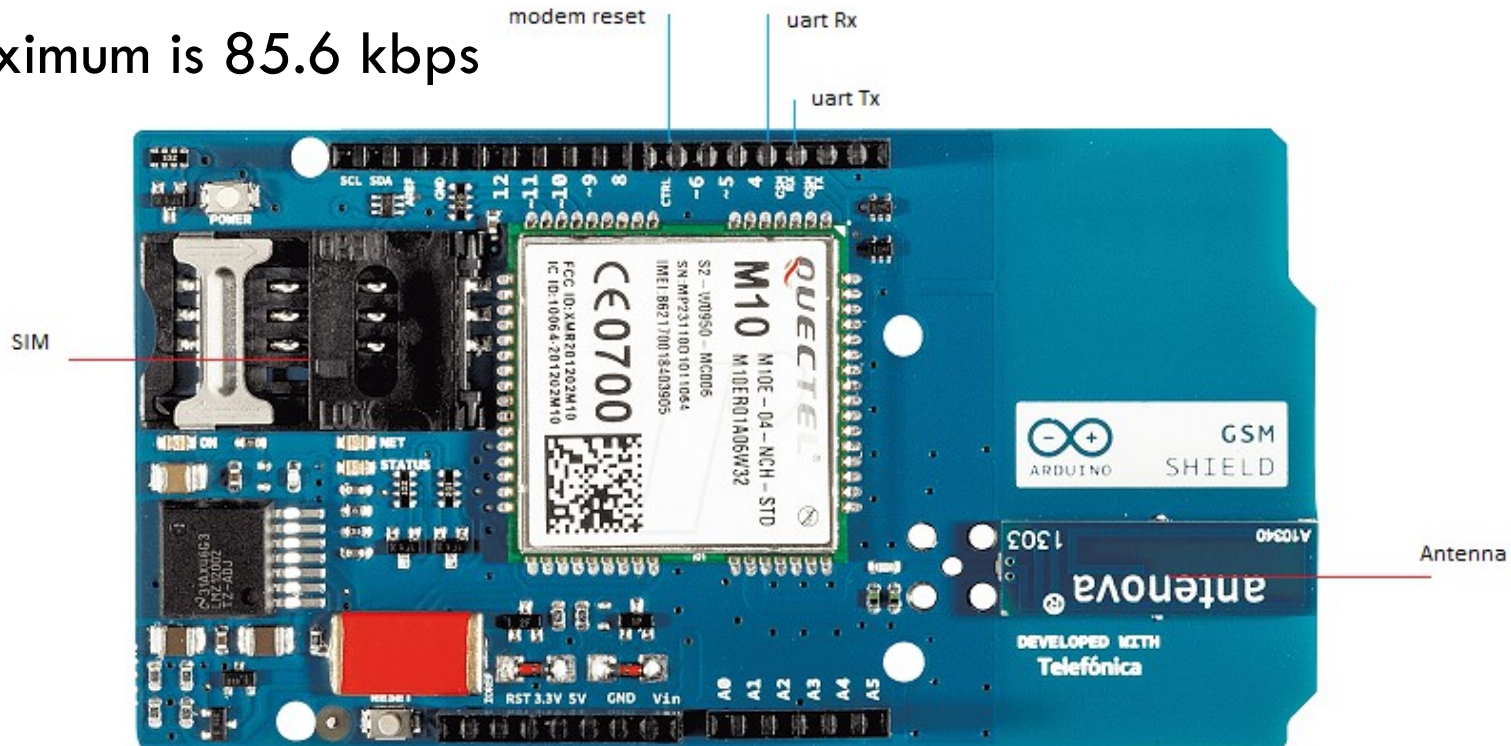
- Xbee module series 1
- Standard 802.15.4
- Set as coordinator, router, end node
- 250kbps Max data rate
- 100m range

GSM SHIELD

Quad-band GSM/GPRS modem

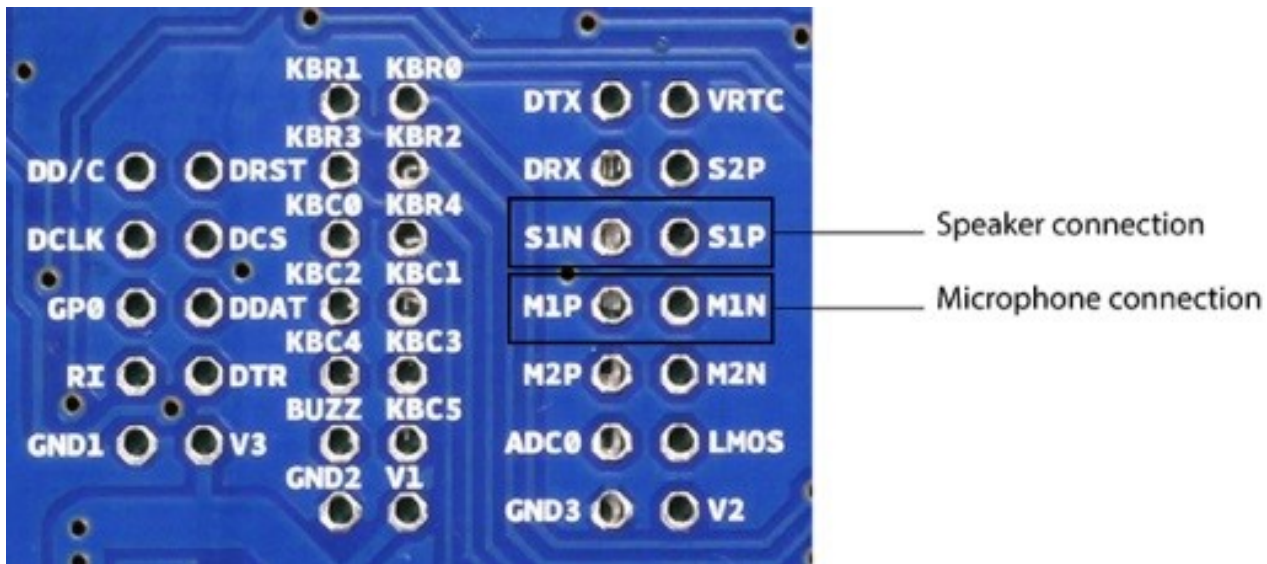
Supports TCP/UDP and HTTP

Speed maximum is 85.6 kbps



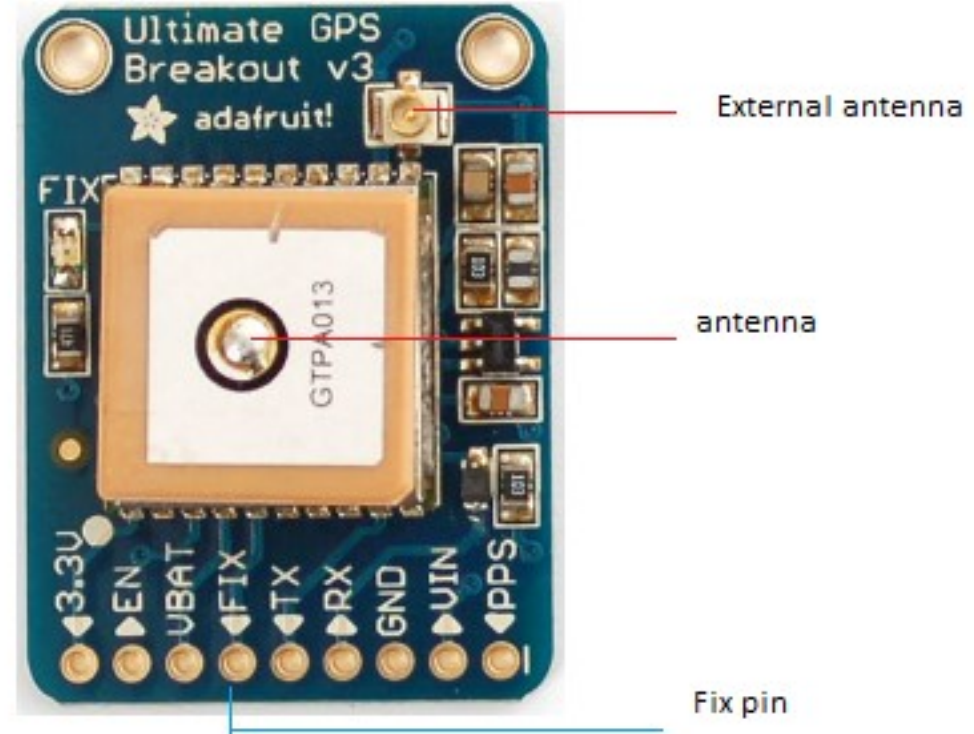
GSM SHIELD

Through the modem, it is possible to make voice calls.



GPS MODULE

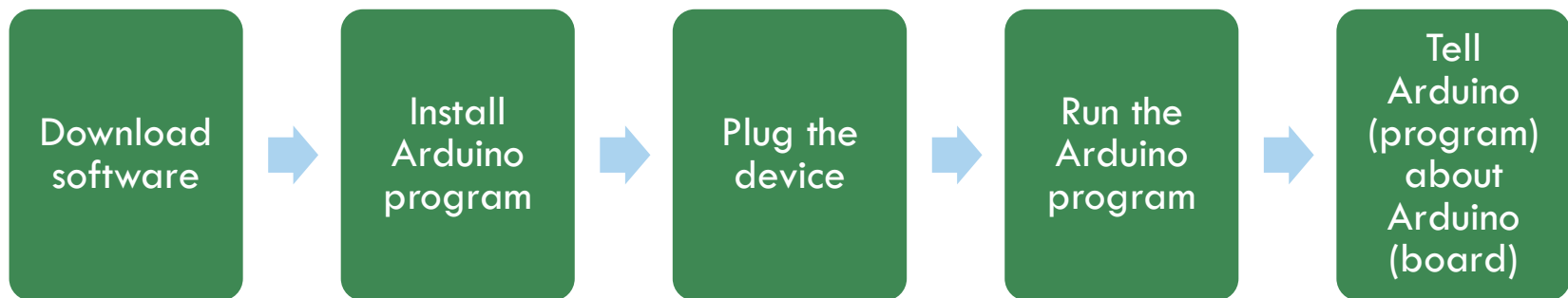
- Power usage is incredibly low
- Ultra-low dropout 3.3V regulator so you can power it with 3.3-5VDC in, 5V level safe inputs
- Position accuracy of 1.8 meters
- Velocity accuracy of .1 meters per second



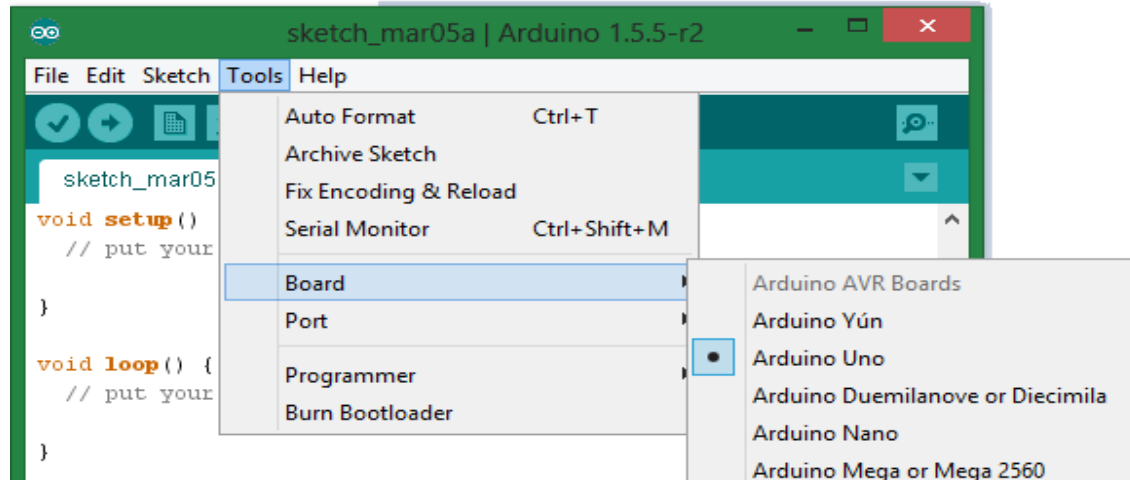
Software: preparing the environment

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.

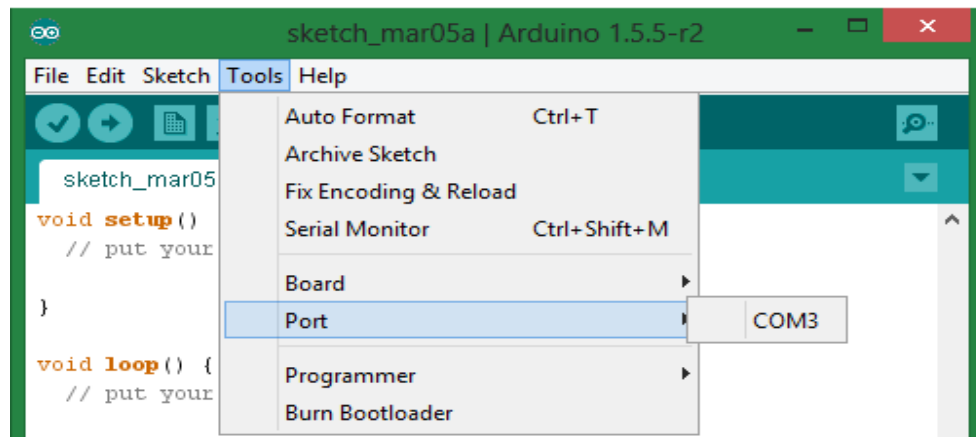
Arduino IDE can be downloaded at www.arduino.cc



SELECT LOCATION AND TYPE



Select your arduino



Select the location of device

TERMINOLOGY

“*sketch*” – a program you write to run on an Arduino board

“*pin*” – an input or output connected to something.

- e.g. output to an LED, input from a knob.

“*digital*” – value is either HIGH or LOW.

- (aka on/off, one/zero) e.g. switch state

“*analog*” – value ranges, usually from 0-1023.

- e.g. LED brightness, motor speed, etc.

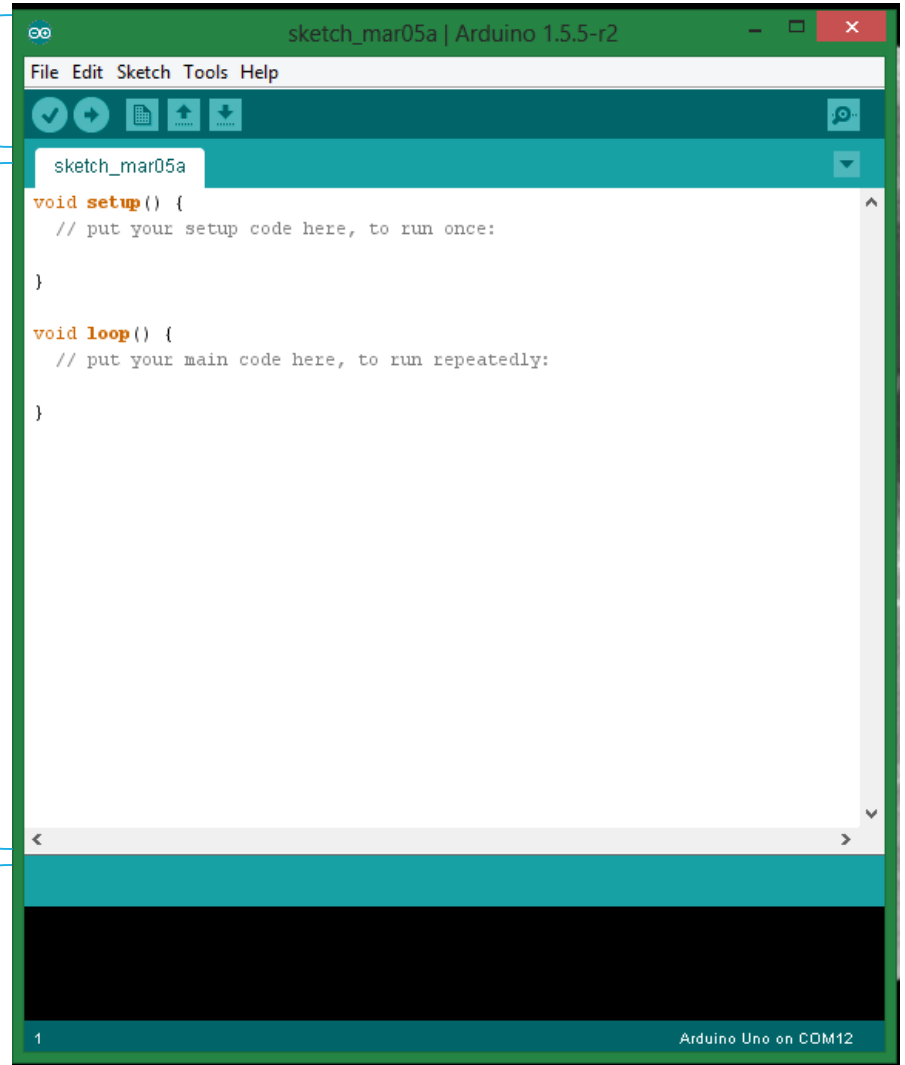
IDE

Toolbar buttons

Sketch editor

Console display

- ✓ Verify
- ➔ Upload
- 📄 New
- ⬆️ Open
- ⬇️ Save
- 🔍 Serial monitor



LANGUAGE

The Arduino environment is based on Atmel Atmega microcontrollers. The AVR language is a "C" environment for programming Atmel chips.

The programs can be divided in three main parts:

Sketch Structure

Variables

Functions

SKETCH AND ITS STRUCTURE

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

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

Called when a sketch starts.
The setup function will only run once.

Does precisely what its name suggests,
and loops consecutively.

OTHER FUNCTIONS & STRUCTURES

Control Structures: `if then else, for, switch, while, continue, return, goto ...;`

Further Syntax: `;, {}, //, /**/, #include, #define;`

Arithmetic Operators: `+, -, =, /, *, %;`

Comparison Operators: `==, !=, <, >, <=, >=;`

Boolean Operators: `&&, ||, !;`

Pointer Access Operators: `*, &;`

Bitwise Operators: `&, |, ^, >>, <<, ~;`

Compound Operators: `++, --, ==, +=, -=, *=, /=, &=, |=;`

VARIABLES

Constants: level of energy (HIGH; LOW); mode of pin(INPUT; OUTPUT; INPUT_PULLUP); led13(LED_BUILTIN);...

Types: word; String;...

Conversions: word();...

Variable scope and qualifiers: Volatile;...

Usefulness: sizeof();

FUNCTIONS

Functions are distinguished according to the pin:

Digitals: `pinMode()`; `digitalRead()`; `digitalWrite()`;

Analogs: `analogReference()`; `analogRead()`; `analogWrite()`;

Advanced I/O: `tone()`; `noTone()`; `shiftOut()`; `shiftIn()`; `pulseIn()`;

Time: `millis()`; `micros()`; `delay()`; `delayMicroseconds()`;

Math: `min()`; `max()`; `abs()`; ...;

Trigonometry: `sin()`; `cos()`; `tan()`;

Random Numbers: `randomSeed()`; `random()`;

Bits and Bytes: `lowByte()`; `highByte()`; `bitRead()`; `bitWrite()`; `bitSet()`; `bitClear()`; `bit()`;

External Interrupts: `attachInterrupt()` `detachInterrupt()`

Interrupts: `interrupts()`; `noInterrupts()`;

Communication: `Serial`; `Stream`;

LIBRARIES

All Libraries for all Arduino shields and components are on:

<http://arduino.cc/en/Reference/Libraries>

AN EXAMPLE: ANALOG & DIGITAL READ

```
// Reads an analog input on pin 0 and converts it to voltage;  
// Reads a digital input on pin 2;  
// When input on pin 2 is high prints voltage in the serial and switches the led.
```

```
const int buttonPin = 2;    // the number of the pushbutton pin  
const int ledPin = 13;     // the number of the LED pin  
int buttonState = 0;       // variable for reading the pushbutton status
```

```
void setup() {              // runs once when you press reset  
  Serial.begin(9600);      // initialize serial communication at 9600 bit per second  
  pinMode(ledPin, OUTPUT); // initialize the LED pin as an output  
  pinMode(buttonPin, INPUT); // initialize the pushbutton pin as an input  
}
```


AN EXAMPLE: ANALOG & DIGITAL READ

```
void loop() {  
  buttonState = digitalRead(buttonPin); // read the state of the pushbutton value:  
  
  if (buttonState == HIGH) {           // if the pushbutton is pressed  
    digitalWrite(ledPin, HIGH);        // turns LED on  
    int sensorValue = analogRead(A0);  // read the input (in [0,1023] on analog pin 0:  
    float voltage = sensorValue * (5.0 / 1023.0); // Convert analog reading  
                                           to a voltage (0-5V):  
    Serial.println(voltage);           // print out the voltage  
  }  
  else digitalWrite(ledPin, LOW);      // turns LED off  
}
```

AN EXAMPLE: BLINK WITH THINKERKIT

```
#include <TinkerKit.h>
```

TKLed led(00); // create the object
'led' that belongs to the 'TKLed' class
and give the value to the desired output
pin

```
void setup() {  
  Serial.begin(9600);  
  Serial.print("iniziamo: ");  
}
```

```
void loop()  
{  
  led.on(); // set the LED on  
  Serial.print(" on ");  
  delay(1000); // wait for a second  
  Serial.print(" off ");  
  led.off(); // set the LED off  
  delay(1000); // wait for a second  
}
```

FORUM & SUPPORT

Support for arduino programmer: <http://forum.arduino.cc>

Tutorial of Arduino Owner:

[Arduino Tutorial](#)

Starter projects with Arduino:

[Starter Projects](#)

Tutorial for AdaFruit component:

- [GSM and GPS](#)
- [Adafruit products](#)

SOME SAMPLE PROJECTS

[Bare Conductive](#)

[Smart citizen kit](#)

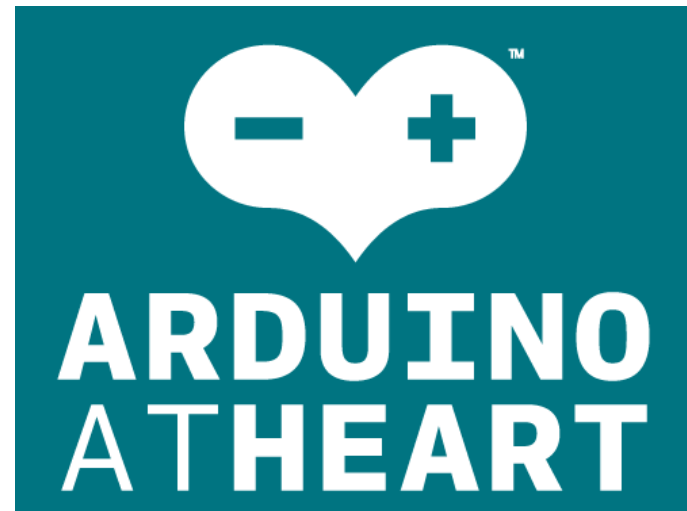
[Little Robot Friends](#)

[Little Bits](#)

[Primo](#)

[Earth Make](#)

[Annikken Andee](#)



TOMORROW'S CLASS

Two assignments:

1. Install Arduino SW on your laptop
2. Bring your laptop!

We will play with Arduino

INSTALLARE TINKERKIT

Copiare la directory **TinkerKitmaster** nella directory:

`documenti/arduino/libraries`

Iniziare lo sketch con la dichiarazione:

```
#include <TinkerKit.h>
```

ESERCIZIO 1

Caricare lo sketch Joystick per arduino e thinkerkit che usa il joystick per comandare due led.

Configurare Arduino e il thinkerkit collegando il joystick e i sensori

Testare il funzionamento

Provare a applicare alcune modifiche per alterarne il comportamento

ESERCIZIO 2

Scrivere e testare uno sketch per arduino e thinkerkit che usa il sensore touch, un interruttore e un led

Alla pressione sul sensore touch il led si accende, alla pressione del pulsante il led si spegne

Note:

- dichiarare `TKTouchSensor touch(I0)` per un sensore touch connesso all'ingresso I0
 - usare `touch.readSwitch()` per leggere dal sensore
- dichiarare `TKButton button(I1)` per usare un pulsante connesso all'ingresso I1
 - usare `button.readSwitch()` per leggere la pressione del pulsante