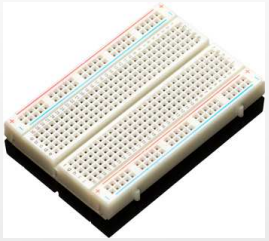
Verifichiamo se abbiamo tutto il necessario ed identifichiamo i componenti:

Resistenze

Connettori

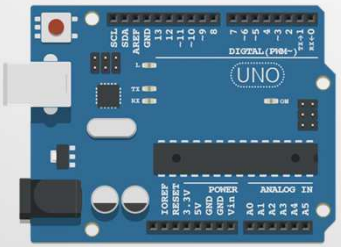
Cavo USB

Led



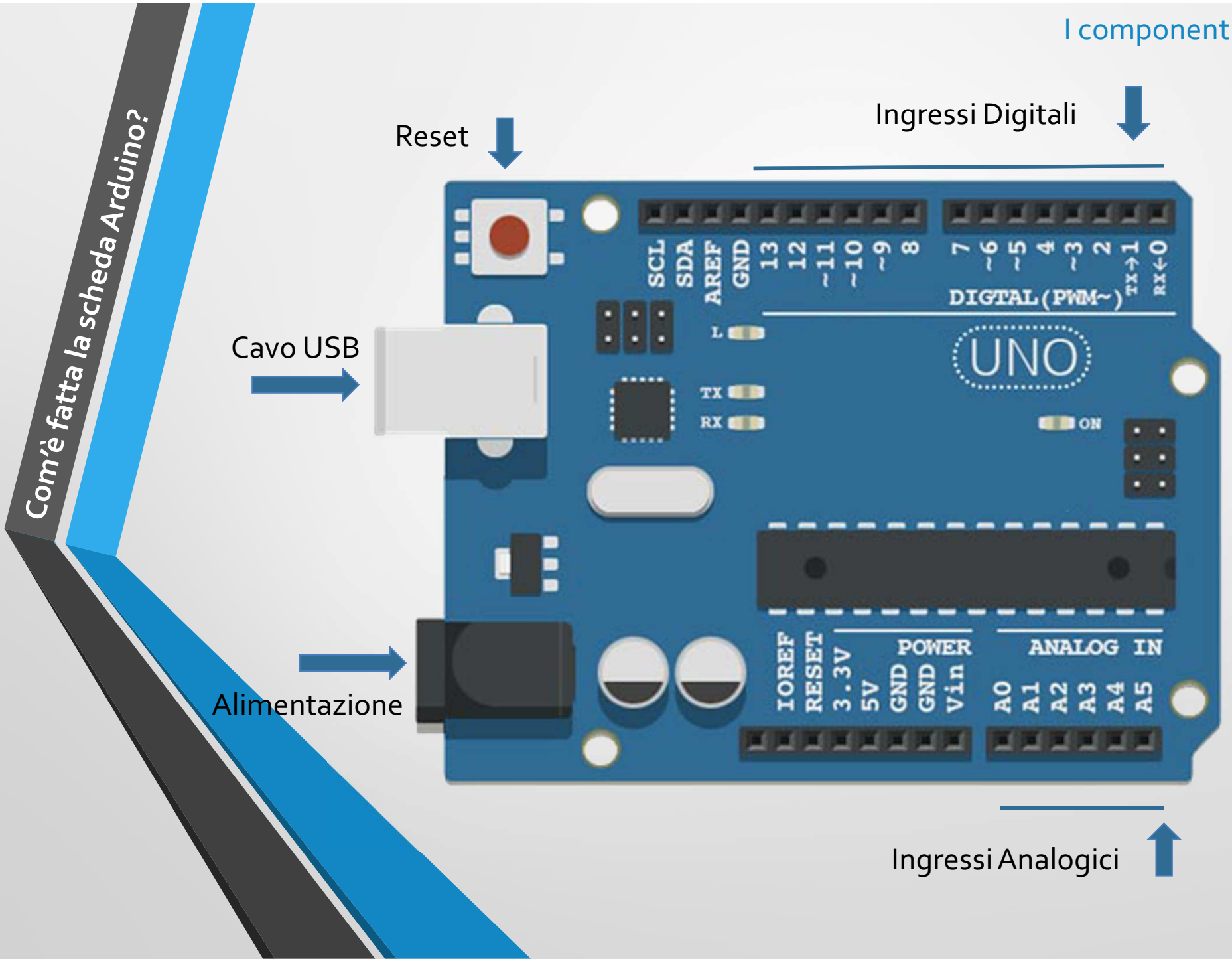
Breadboard

Arduino

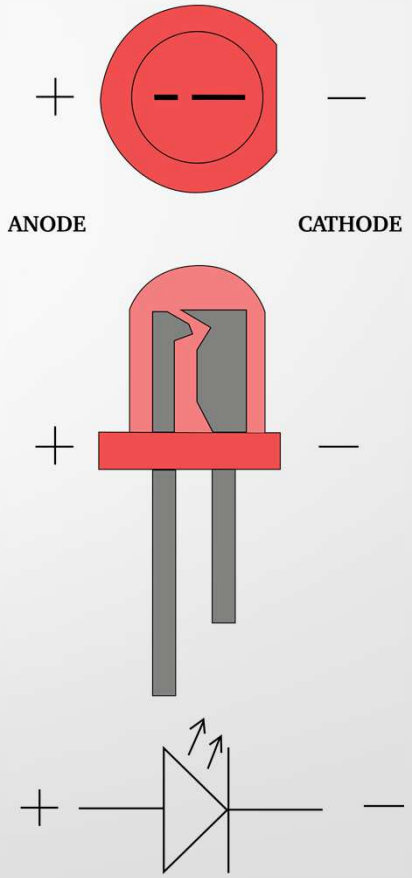


Cavo alimentazione

Com'è fatta la scheda Arduino?

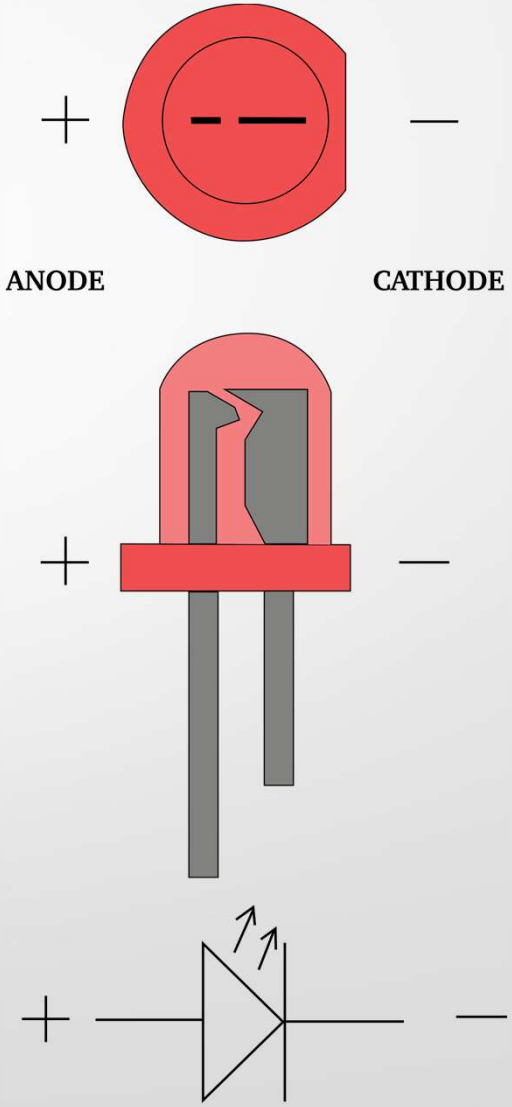


Concentriamo la nostra attenzione sul Led (ne troverete nella confezione uno libero ed uno con saldati ai piedini due connettori di diverso colore. Il nostro led ha due piedini, uno più lungo ed uno più corto. Cosa significa?

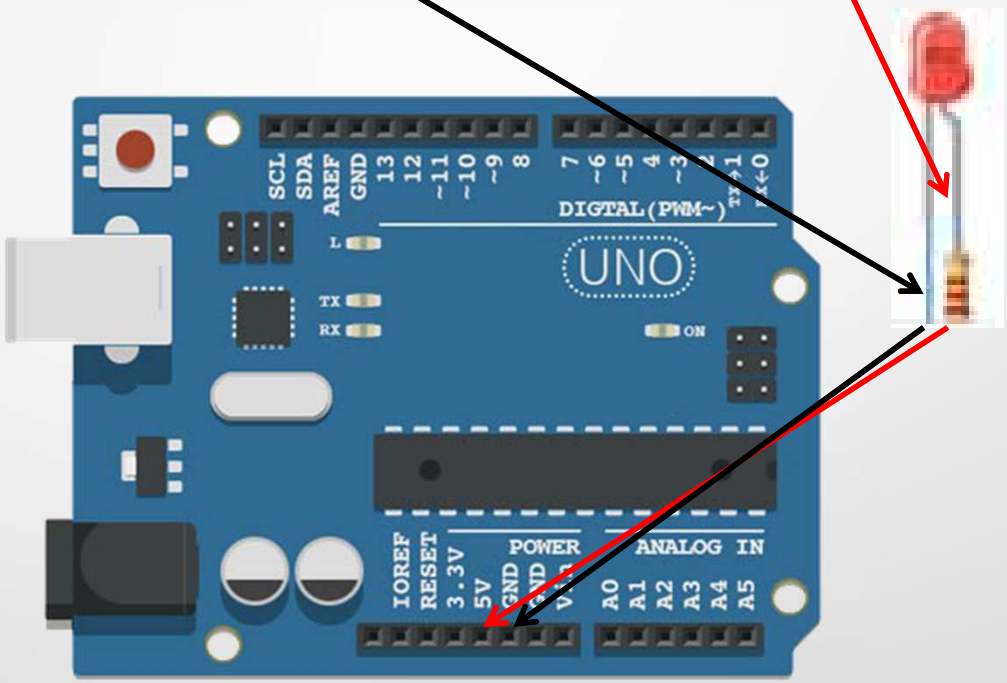


Semplice!
Il nostro led ha un polo positivo ed un polo negativo (come una batteria)

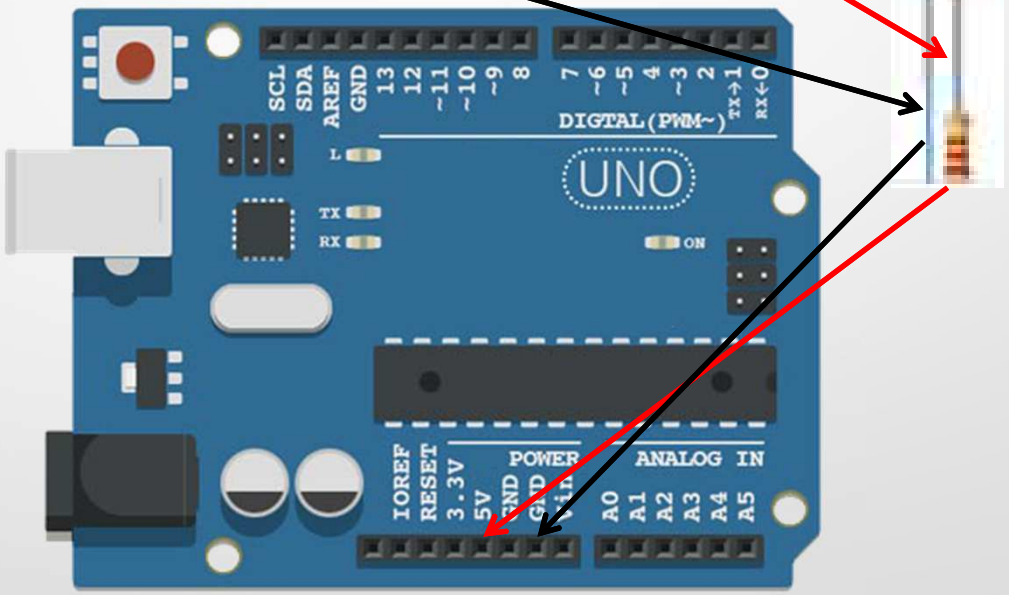
Sì, ma in che verso va collegato il LED?
dovremo fare in modo che il **catodo,** cioè il polo **negativo,** sia collegato a massa (GND), mentre l' **anodo** (polo **positivo**) al pin di controllo.



Il primo esperimento che faremo sarà quello di accendere un led con Arduino. Il nostro led ha un polo positivo ed un polo negativo (come una batteria)



Utilizziamo il Led con saldati alle estremità i due connettori di diverso colore.
Inseriamo il connettore saldato al terminale positivo del Led (quello più lungo, in corrispondenza della scritta 5V presente su Arduino.
L'altro connettore va inserito in corrispondenza di u
delle scritte GND)



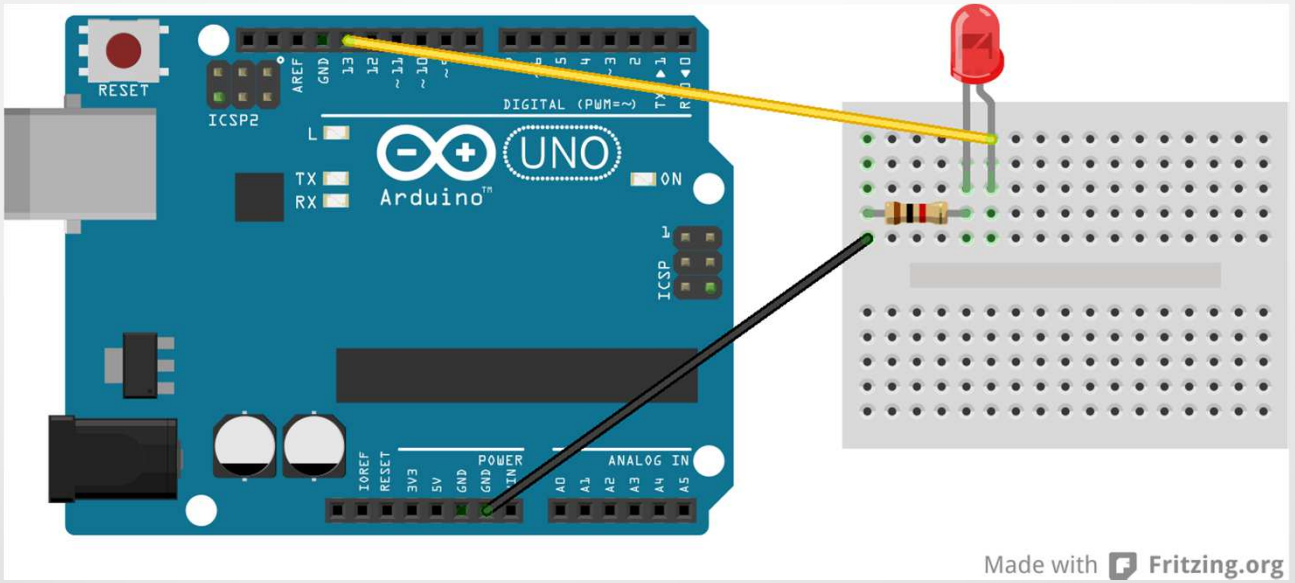
Operiamo…

Stiamo però utilizzando Arduino come una semplice fonte di energia elettrica per accendere il nostro LED.
Sappiamo che Arduino può accendere il nostro LED se lo istruiamo in maniera opportuna con un programma detto Sketch.

Il secondo esperimento che faremo, quindi, sarà quello di accendere un led con Arduino utilizzando uno Sketch.

Ricordiamo di togliere il Led utilizzato precedentemente da Arduino.

Vediamo come collegare i fili ed i componenti utilizzando la Breadboard.



Programmare Arduino…

Adesso bisogna dire ad Arduino cosa deve fare.
Arduino capisce solo quello che noi saremo in grado di dirgli nella sua lingua. Questa è composta da istruzioni espresse in lingua inglese. Non si può scrivere una poesia in questa lingua ma è molto efficace per dare ordini precisi.
In realtà il linguaggio utilizzato per parlare con Arduino è una versione semplificata di un linguaggio di programmazione molto importante, il C++

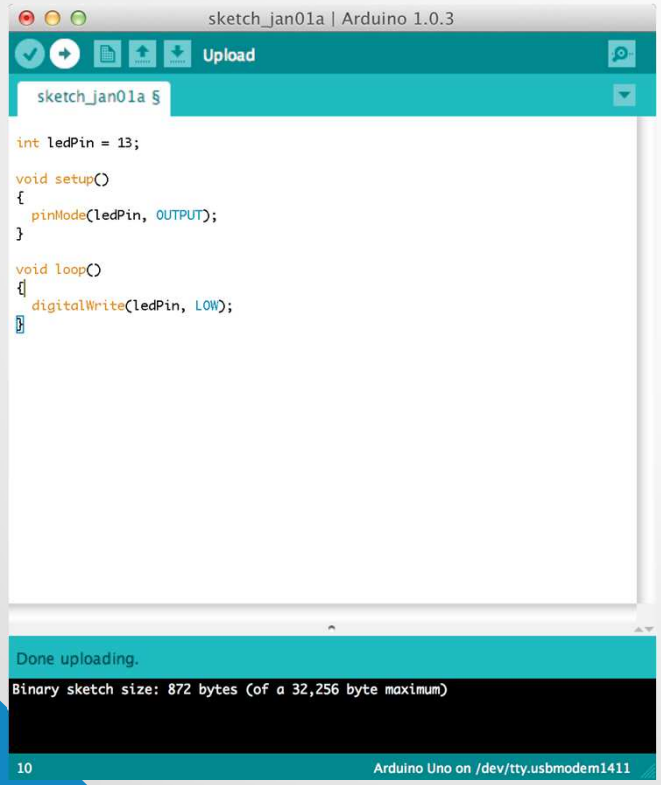
Alcune parole in *Arduinese*.

digitalWrite

delay

pinMode

Programmare Arduino…



Il programma che ci permette di comunicare con Arduino è l'IDE. Si tratta di un foglio bianco sul quale scriviamo le parole seguendo le regole corrette. Il programma ci aiuta: se le parole sono esatte e scritte nel giusto modo si colorano, altrimenti abbiamo sbagliato in qualche cosa.

Accendiamo adesso il computer individuiamo e facciamo Click sull'icona:

Iniziamo allora a scriverle ed a comprenderne il significato:

```
void setup() {

  pinMode(10, OUTPUT);
}

void loop() {
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);
}
```

Commentiamo insieme il codice:

```
void setup() {

  pinMode(10, OUTPUT);
}

void loop() {
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);
}
```

La funzione setup( ) è la prima ad essere chiamata quando parte uno sketch. Viene utilizzata per inizializzare variabili, per impostare lo stato dei pinLa funzione di setup() sarà la prima ad essere eseguita dopo ogni accensione o reset di Arduino.

Commentiamo insieme il codice:

```
void setup() {

  pinMode(10, OUTPUT);
}

void loop() {
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);
}
```

pinMode(pin, mode);
pin definisce il numero del pin di cui si vuole impostare la modalità ingresso o uscita, mode sempre INPUT o OUTPUT

Commentiamo insieme il codice:

```
void setup() {

  pinMode(10, OUTPUT);
}

void loop() {
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);
}
```

Dopo la creazione della funzione setup(), che inizializza e imposta i valori iniziali, la funzione loop() fa proprio quanto suggerisce il proprio nome eseguendo ciclicamente il programma definito al suo interno.

Commentiamo insieme il codice:

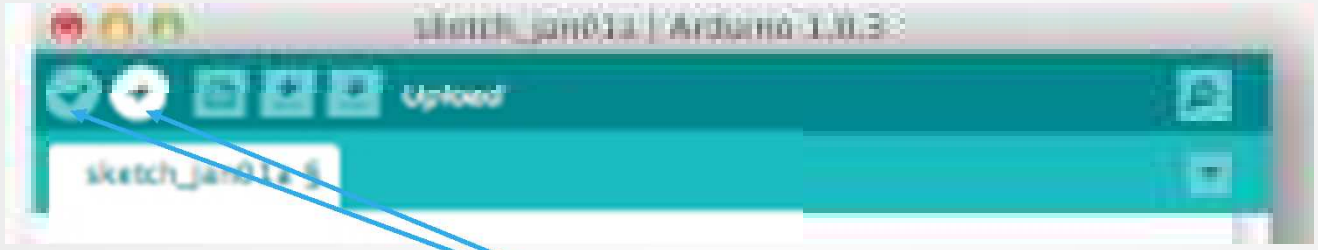
```
void setup() {

  pinMode(10, OUTPUT);
}

void loop() {
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);
}
```

Scrive un valore HIGH o LOW su un pin impostato come digitale. Se il pin è stato impostato come OUTPUT con il pinMode(), la sua tensione sarà impostata al corrispondente valore di 5V per HIGH e 0V per LOW.

Commentiamo insieme il codice:

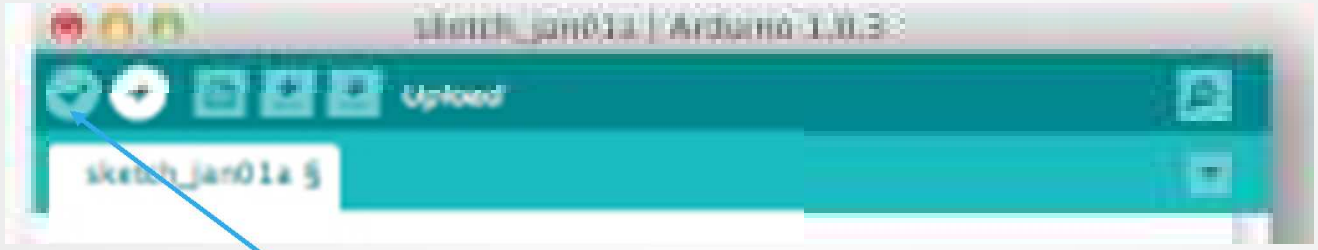
```
void setup() {

 pinMode(10, OUTPUT);
}

void loop() {
 digitalWrite(10, HIGH);
 delay(1000);
 digitalWrite(10, LOW);
 delay(1000);
}
```

Mette in pausa il programma per un certo intervallo di tempo in millisecondi specificato da un parametro.



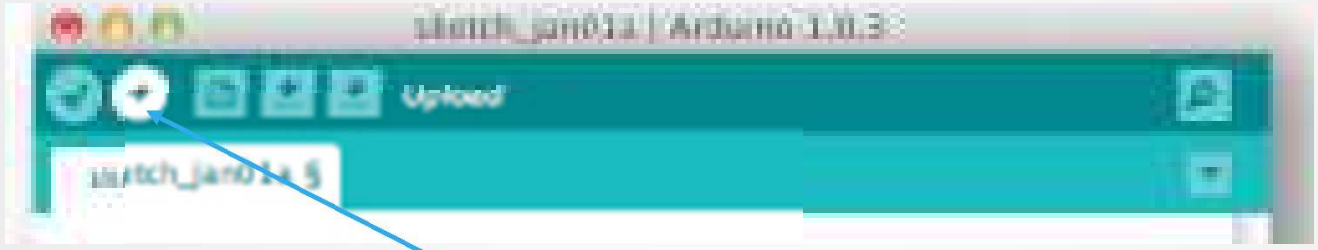
Osserviamo adesso i due pulsanti presenti nella barra in alto sull'IDE

- ✔ Serve per controllare il codice scritto
- ➡ Serve per trasferire nella scheda il codice scritto



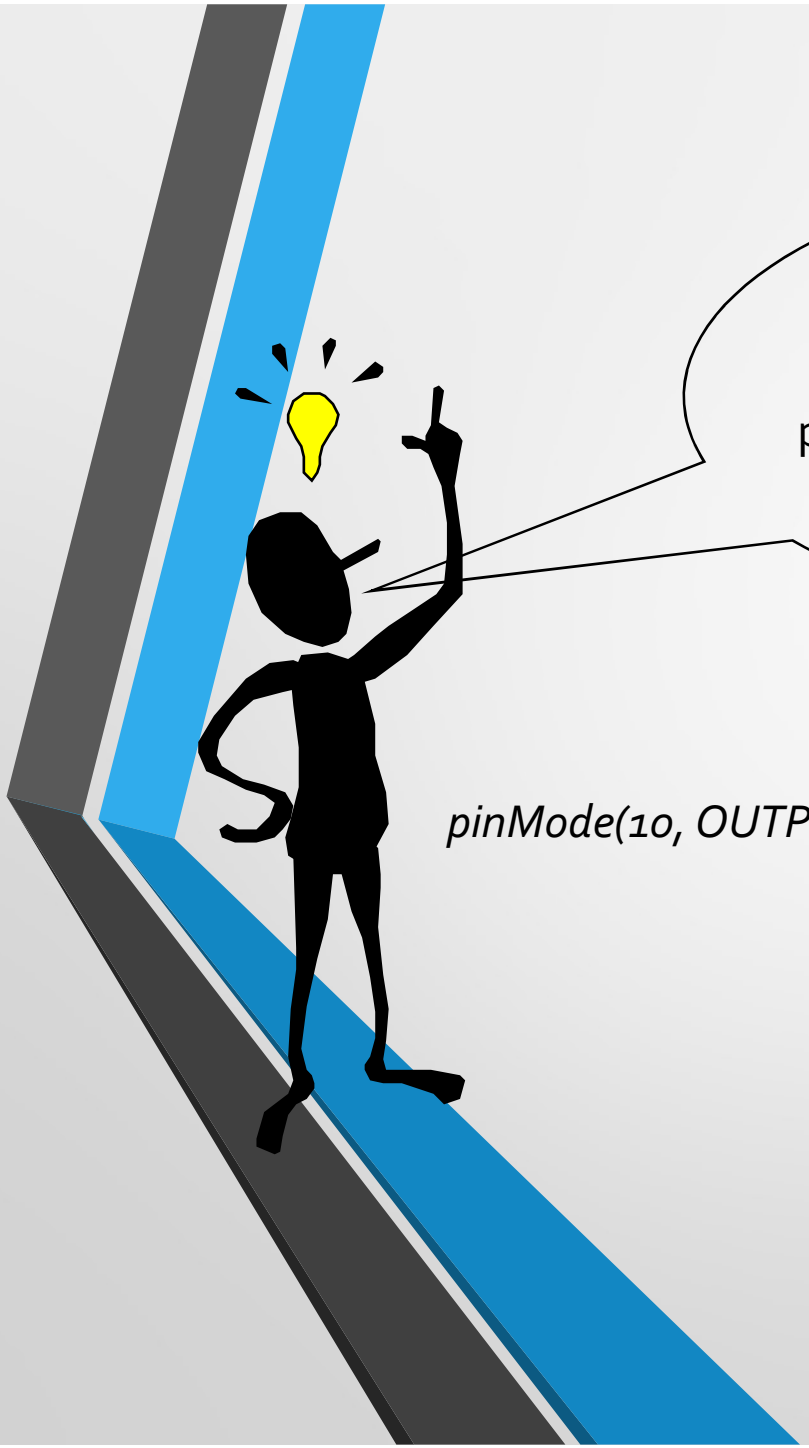
- Serve per controllare il codice scritto

Premendo il primo pulsante IDE controlla che non ci siano errori nella scrittura del programma. Se questi sono presenti saranno segnalati in una finestra sistemata in basso sull'IDE, altrimenti verrà restituito il valore della memoria occupata sulla scheda



Se è tutto ok allora possiamo premere il secondo pulsante.
Il codice verrà adesso trasferito nella scheda

- Serve per trasferire nella scheda il codice scritto



Se il led non si accende bisogna ricontrollare il circuito verificando se i piedini sono stati inseriti nel pin Arduino indicato nel programma

pinMode(10, OUTPUT);

Viene individuata l'uscita per alimentare il Led e comunicato il valore ad Arduino