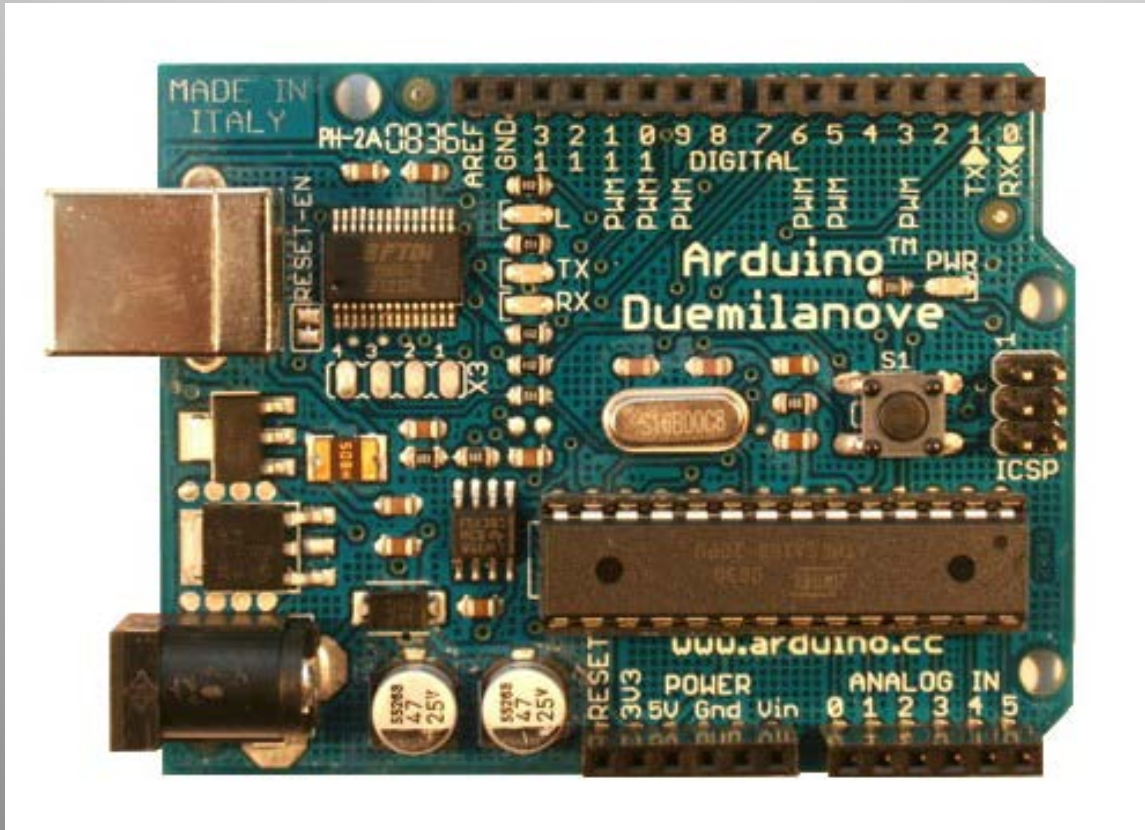


# **ARDUINO**

**Um tutorial inicial**

# Arduino

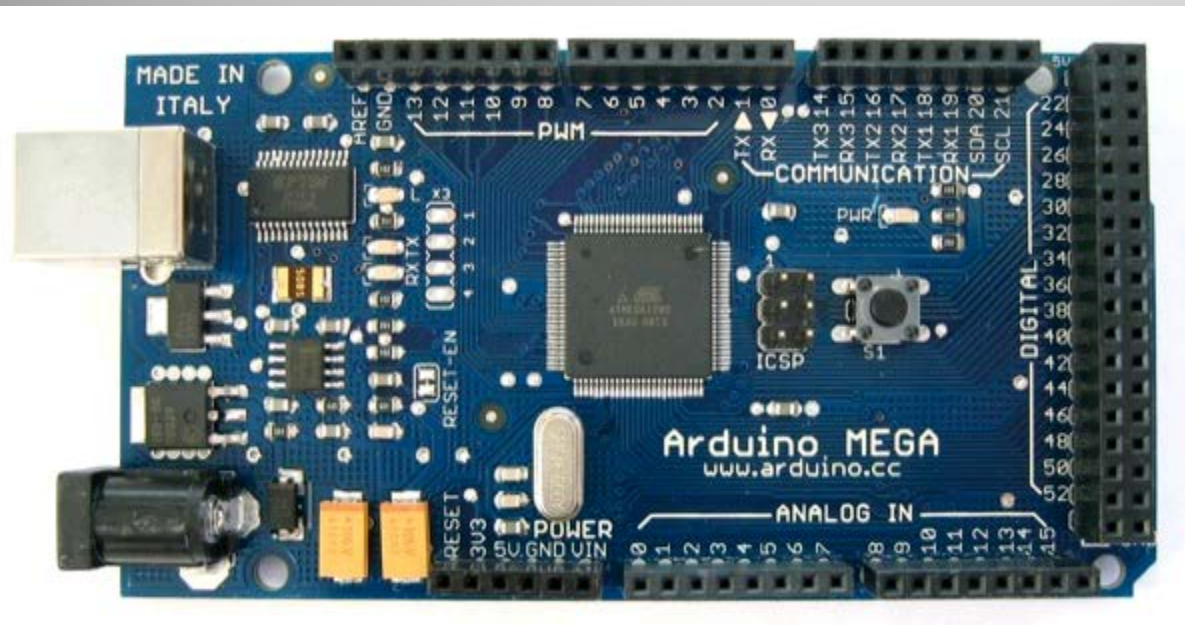
- ▶ Placa desenvolvida na Itália em 2005 (open source);
- ▶ Facilitar o desenvolvimento através de Shields;



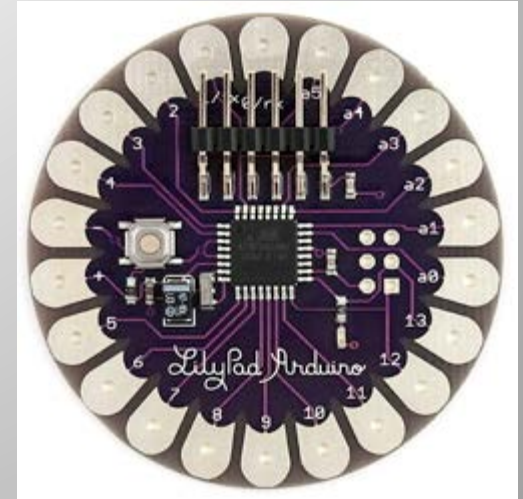
# Mas é só essa plaquinha ?

Não!

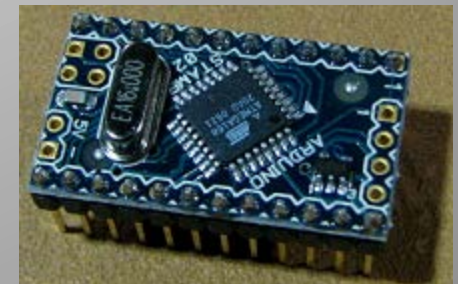
Temos varios modelos para aplicações diversas.



- Arduino Mega- 128k (Flash Memory)
- 54 Pinos de I/O



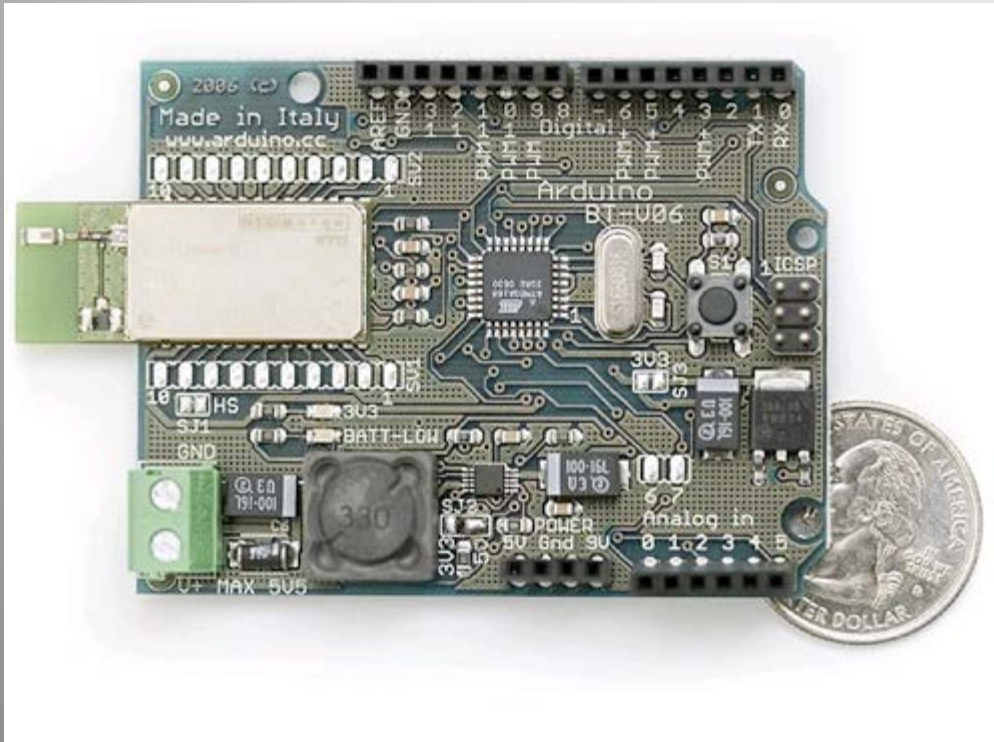
➤ LilyPad Arduino



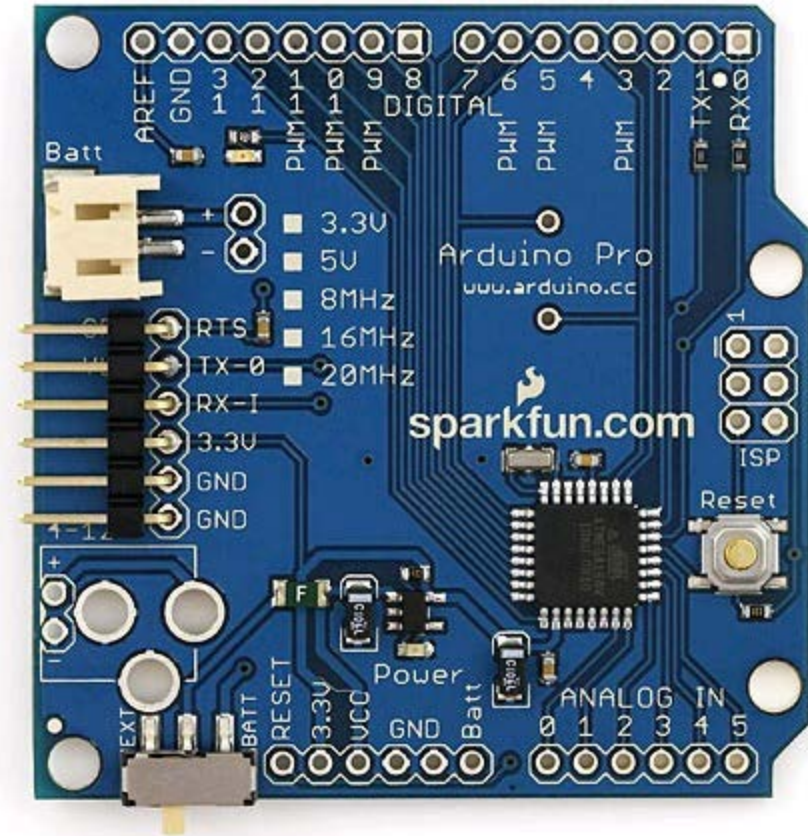
➤ Arduino Mini



# Mais ...



➤ Arduino BT (Bluetooth)

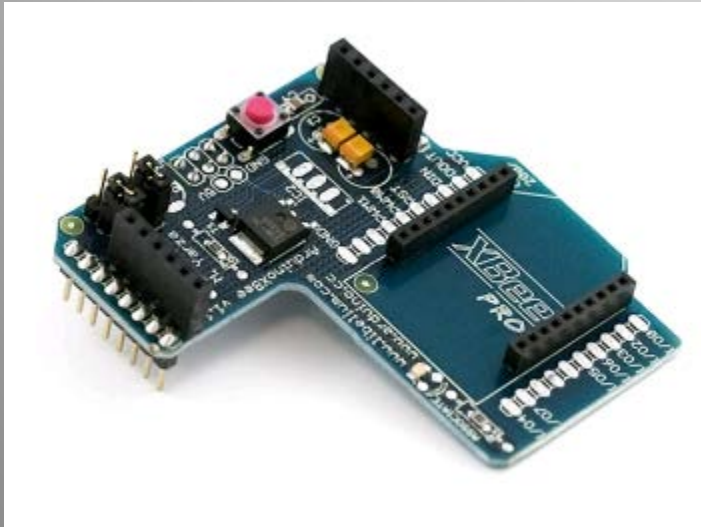


➤ Arduino Pro

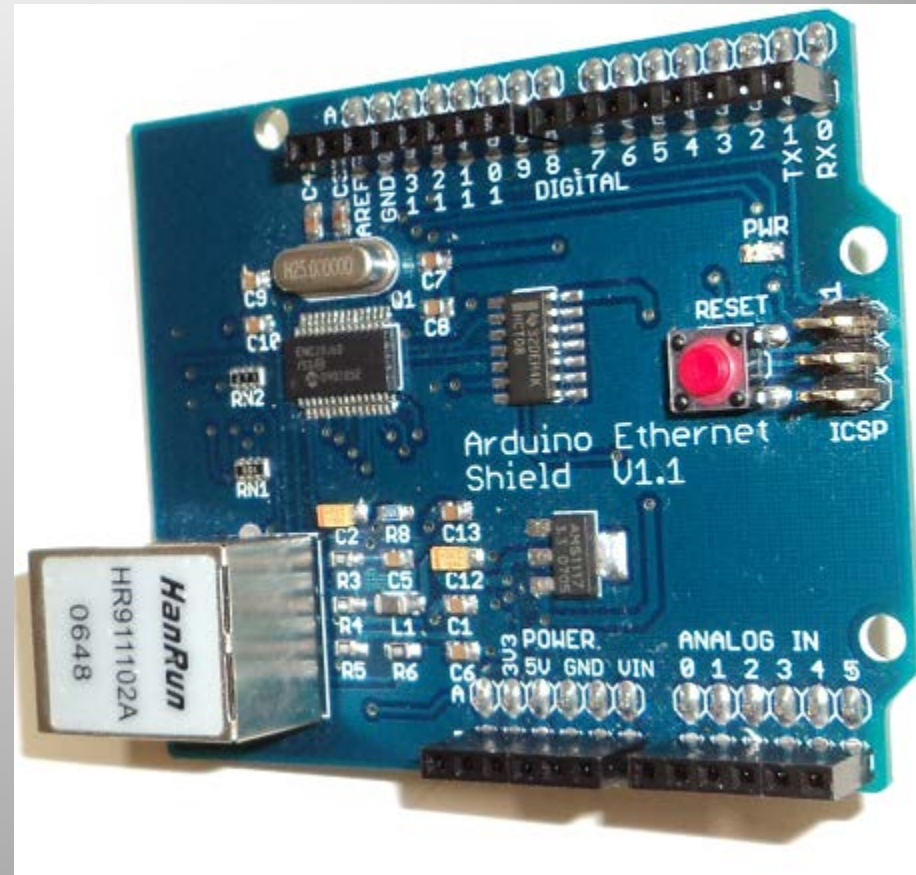
# E ainda tem os SHIELDS ...

## O que são Shields?

“Escudos”. Placas adicionais com conexões ao arduino e que permitem interagir com tecnologias diversas e com facilidade.



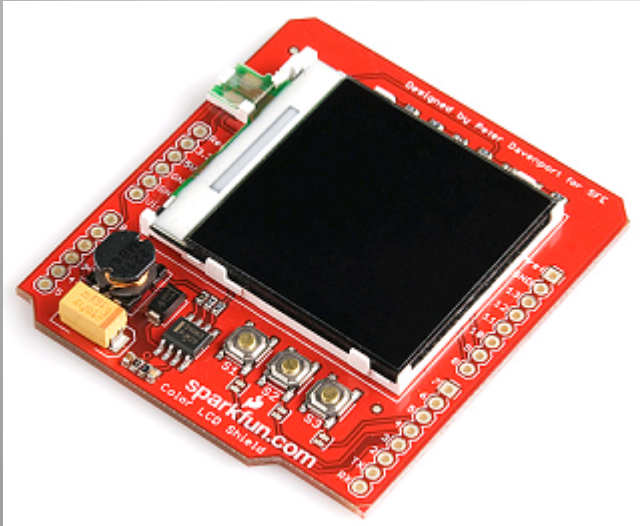
➤ Xbee shield



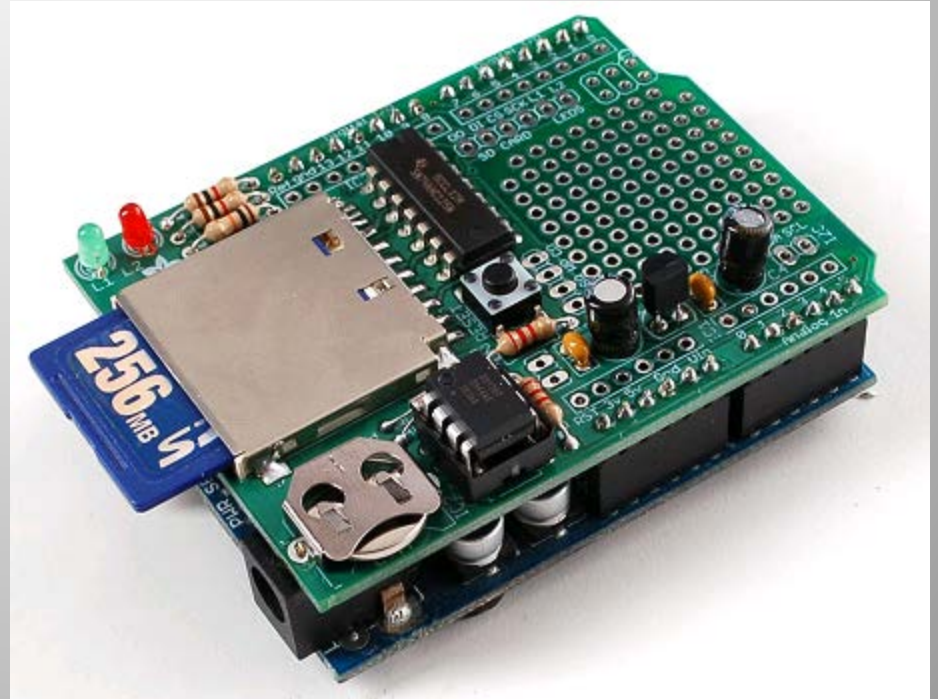
➤ Ethernet shield



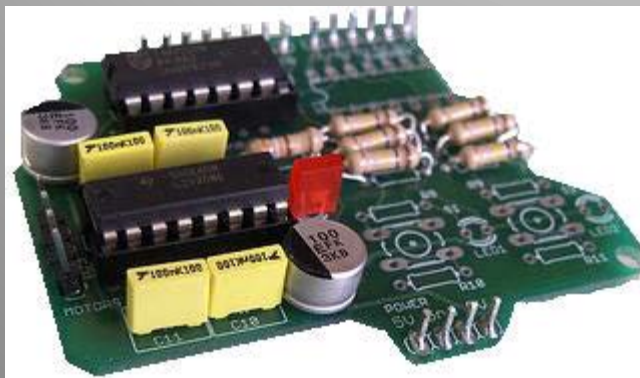
# Mais ...



➤ LCD shield



➤ SD shield

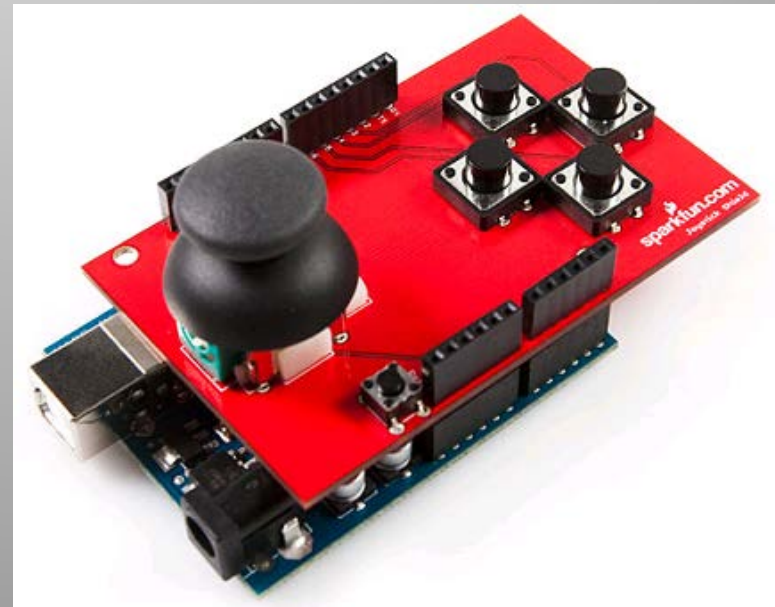


➤ Motor shield

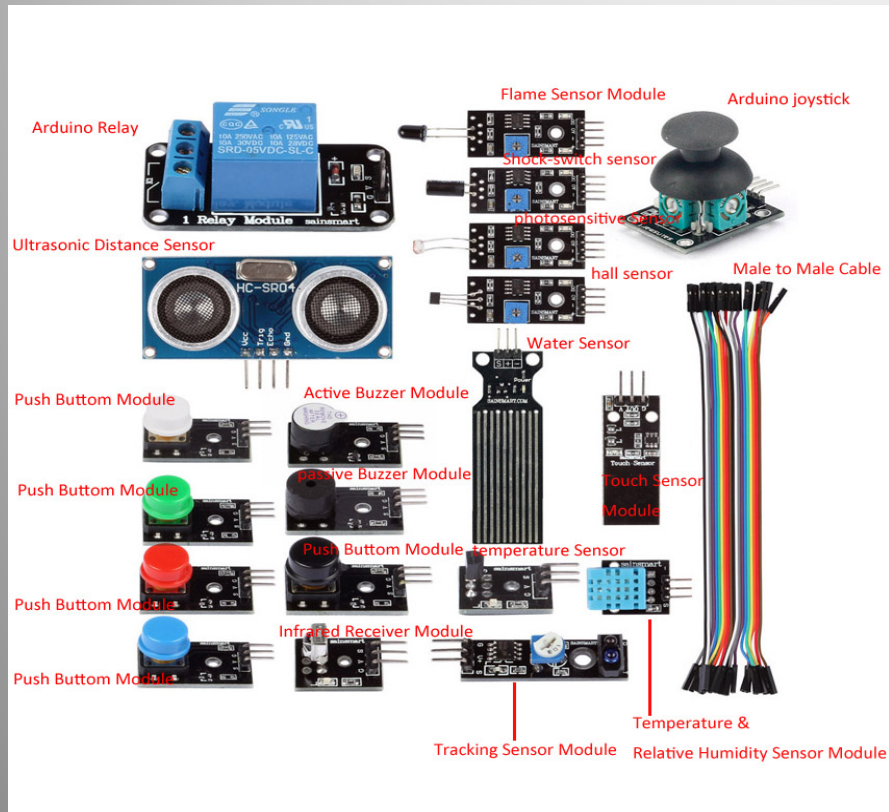
# Como liga ?

## Como usar os Shields?

Conecta-se sobre o Arduino .

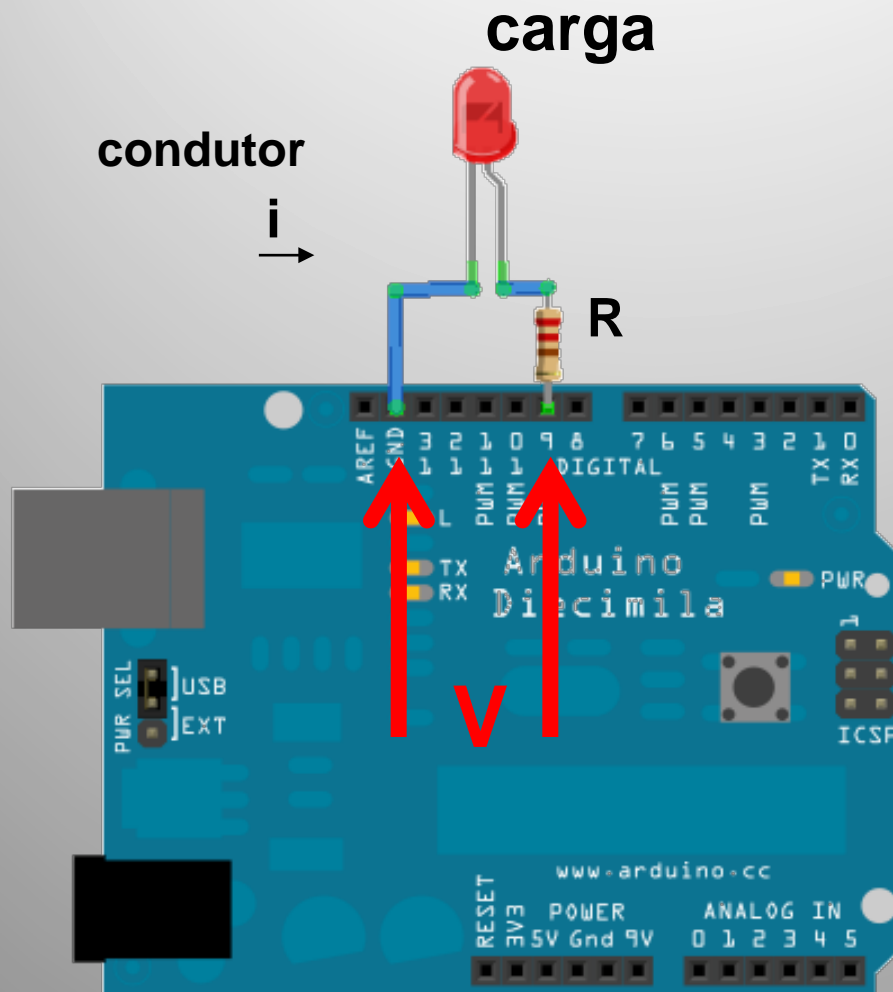


# E ainda os sensores e atuadores

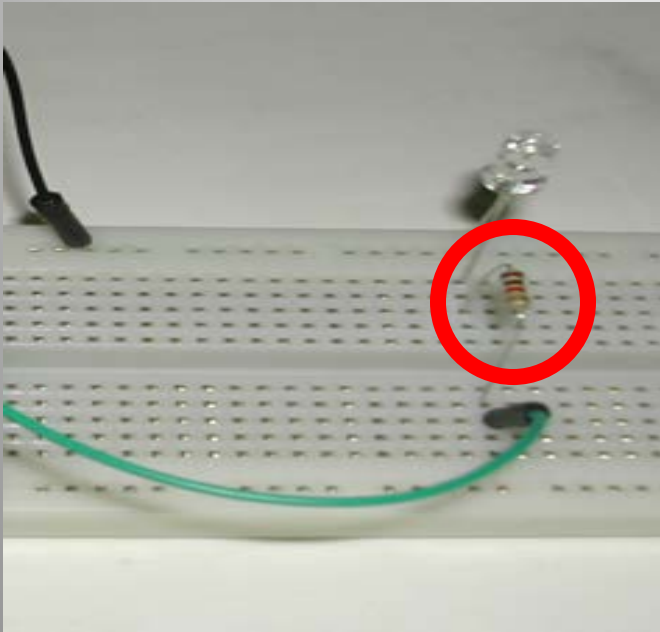




# Qual será o nosso circuito?



# ATENÇÃO

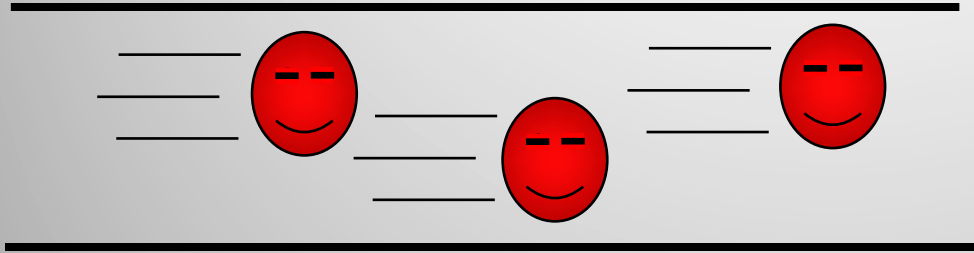


**Porque é necessário um resistor junto com o led ?**

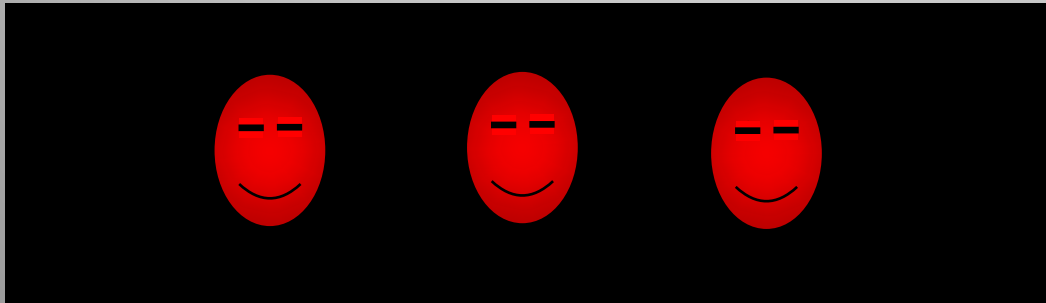
Porque deveremos proteger a saída do Arduino limitando a quantidade de corrente que irá passar.

**Não ligue nada (motores, lâmpadas, leds, etc) diretamente no Arduino sem pesquisar corrente e tensão.**

# Um pouco de eletricidade...



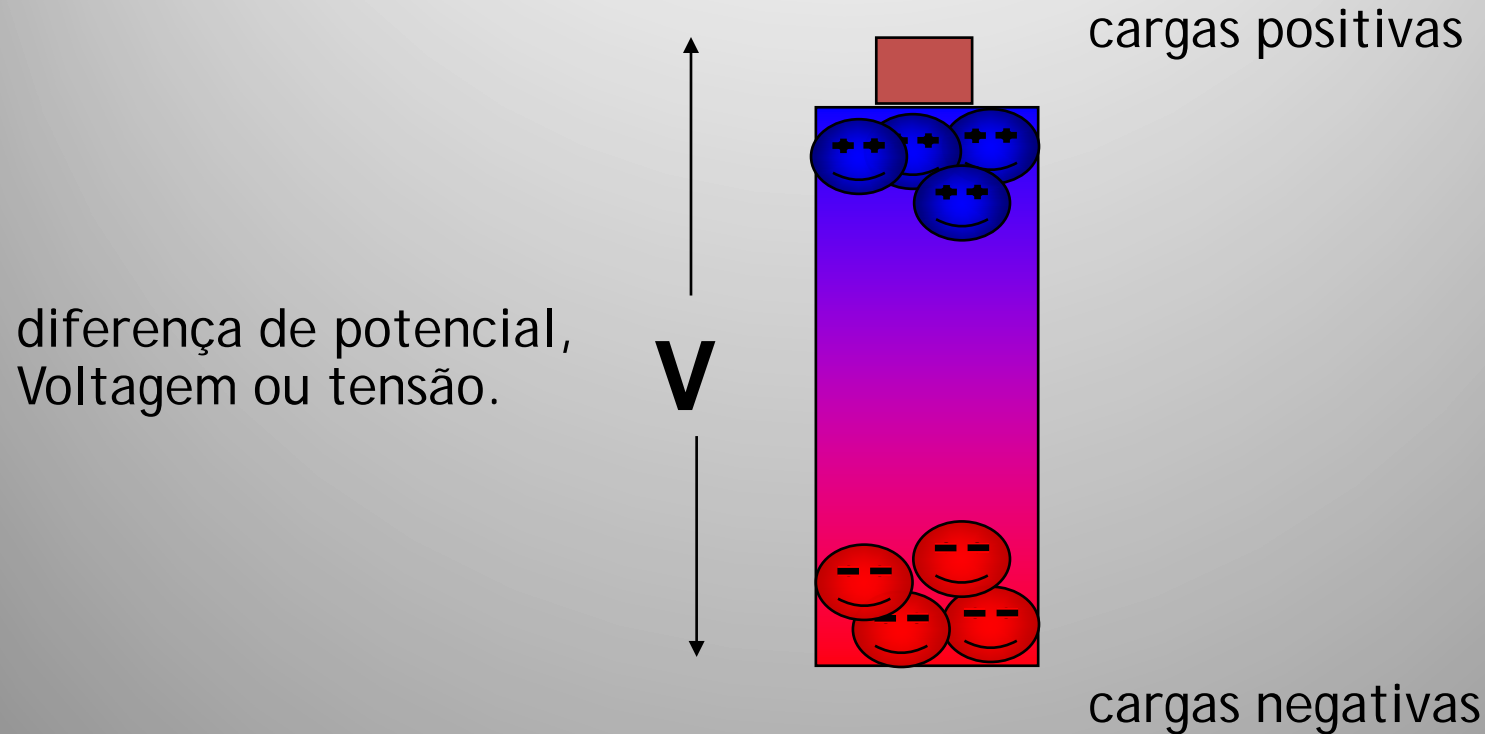
Um condutor permite o fluxo de elétrons



Um isolante evita a passagem de elétrons

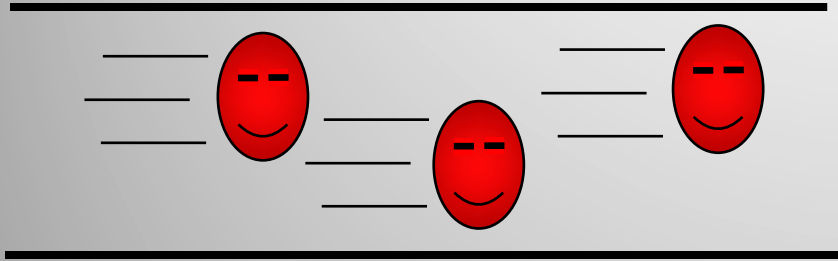


# Voltagem ou Tensão elétrica

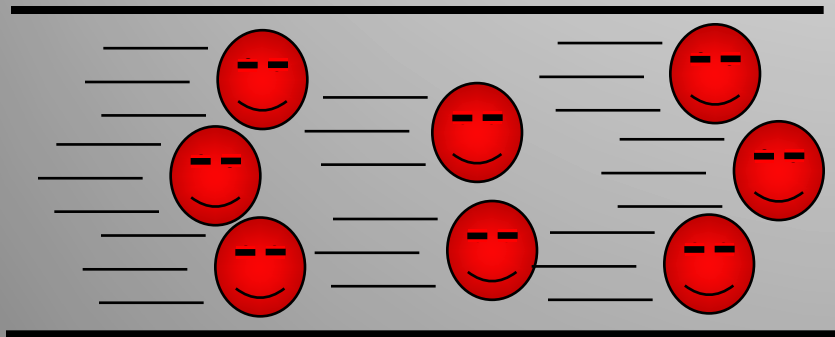


quanto maior a tensão, mais "força" possuem os elétrons

# Corrente elétrica

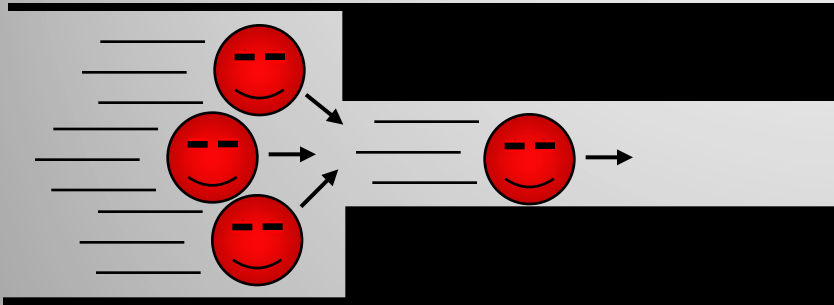


fluxo de elétrons em um  
condutor

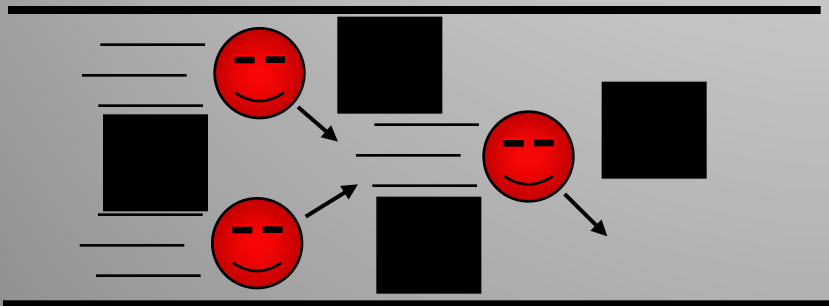


quanto maior a corrente,  
maior a “quantidade” de  
elétrons

# Resistência elétrica



propriedade do material  
condutor em reduzir a  
passagem dos elétrons



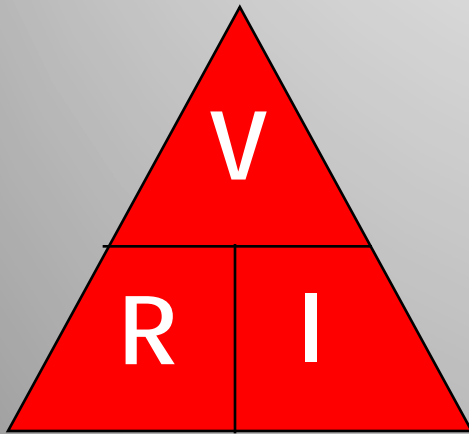
elétrons "se acumulam e  
batem" no condutor,  
"dissipando" sua energia  
(gerando calor)



# Lei de OHM

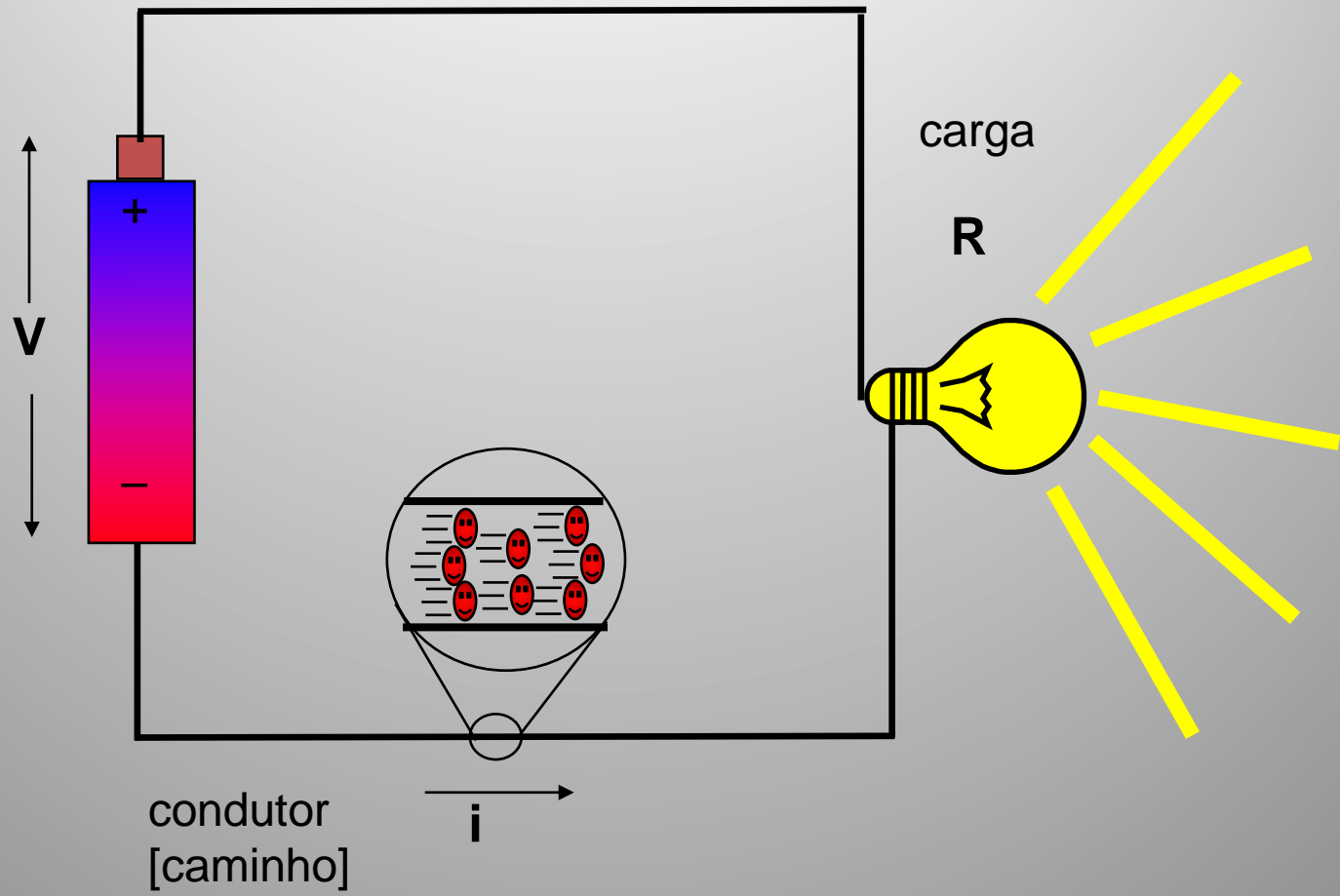
$$V = R \times I \text{ ou}$$

$$I = V / R$$

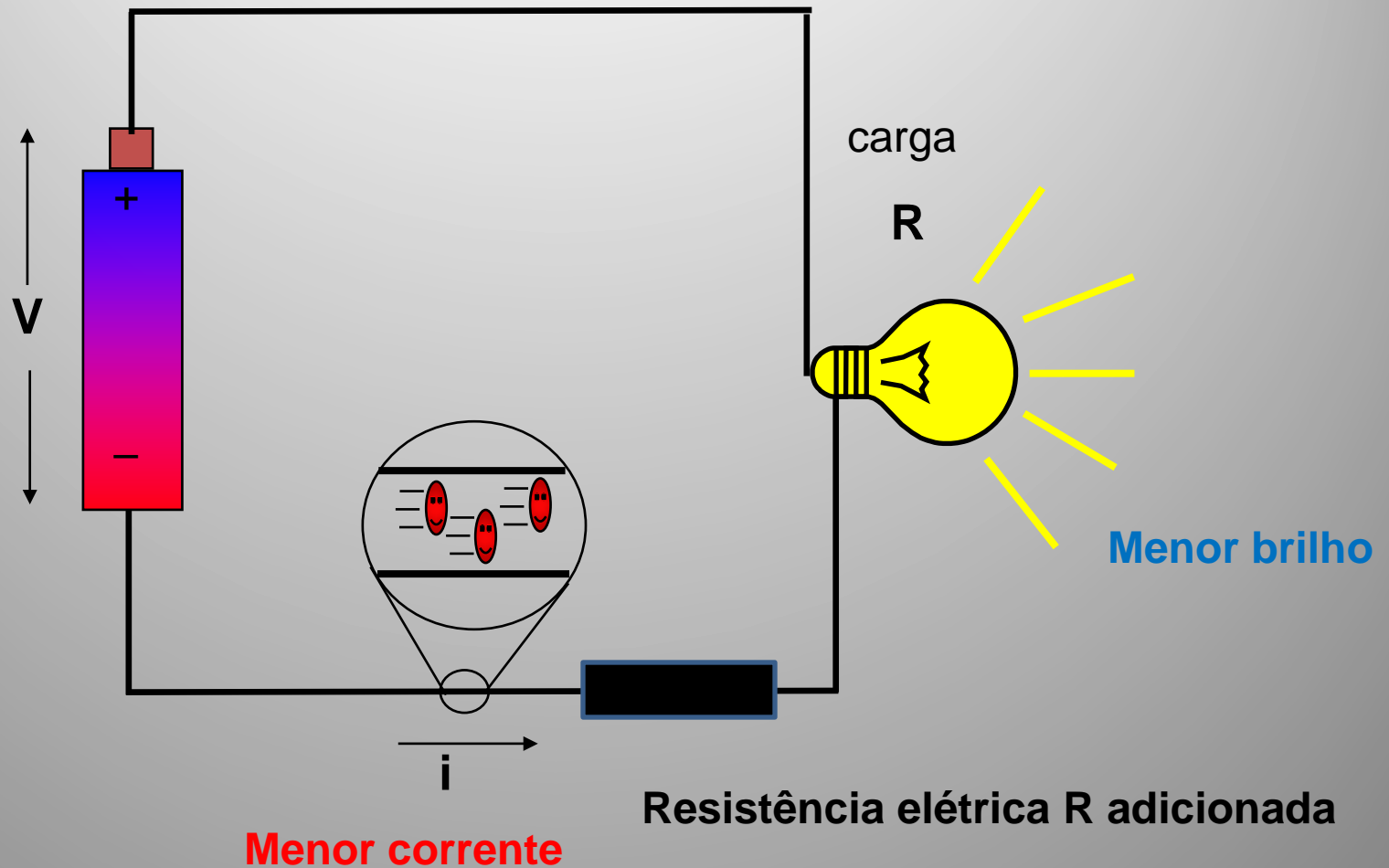


Se uma tensão  $V$  for aplicada a um circuito, um elemento  $R$  poderá diminuir a corrente que circulará no mesmo

# Um circuito elétrico

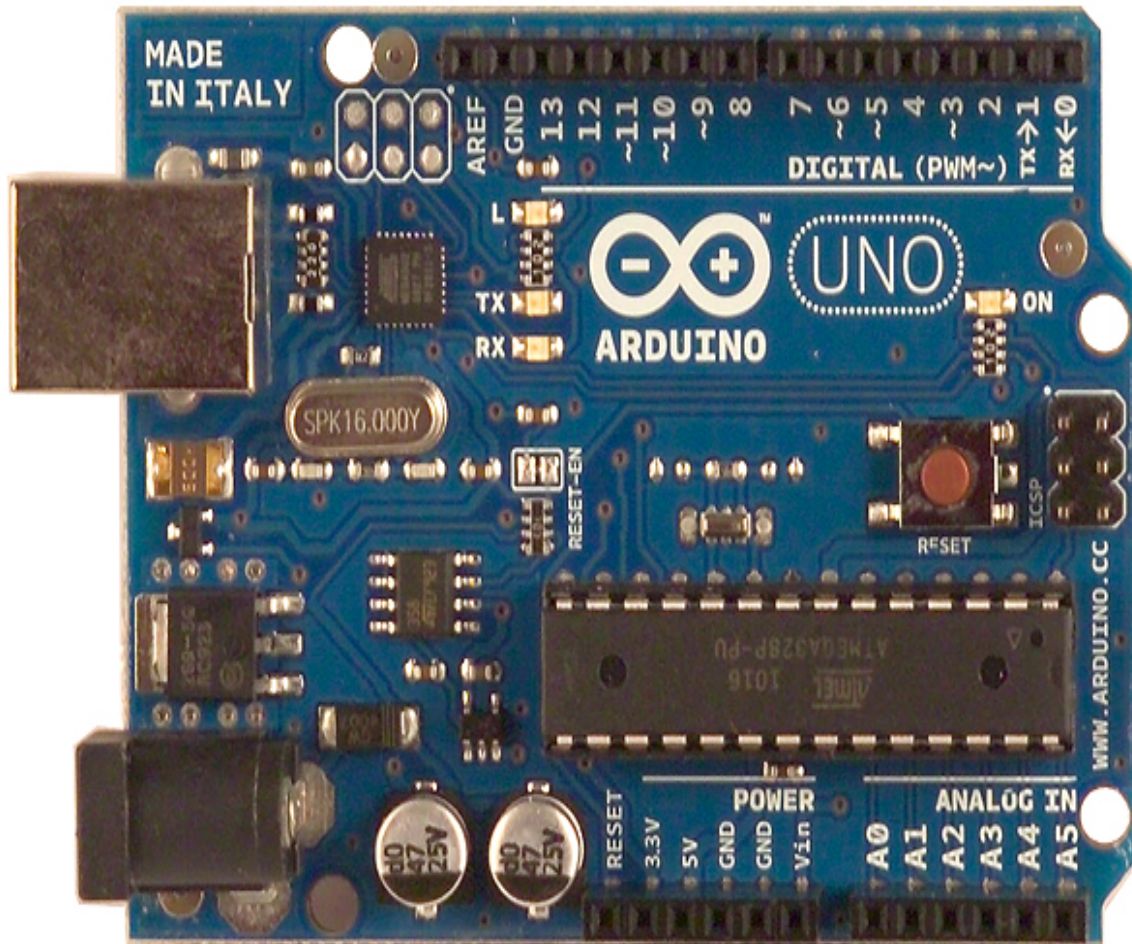


# Um circuito elétrico





# Os pinos do Arduino



Observe a identificação dos pinos.

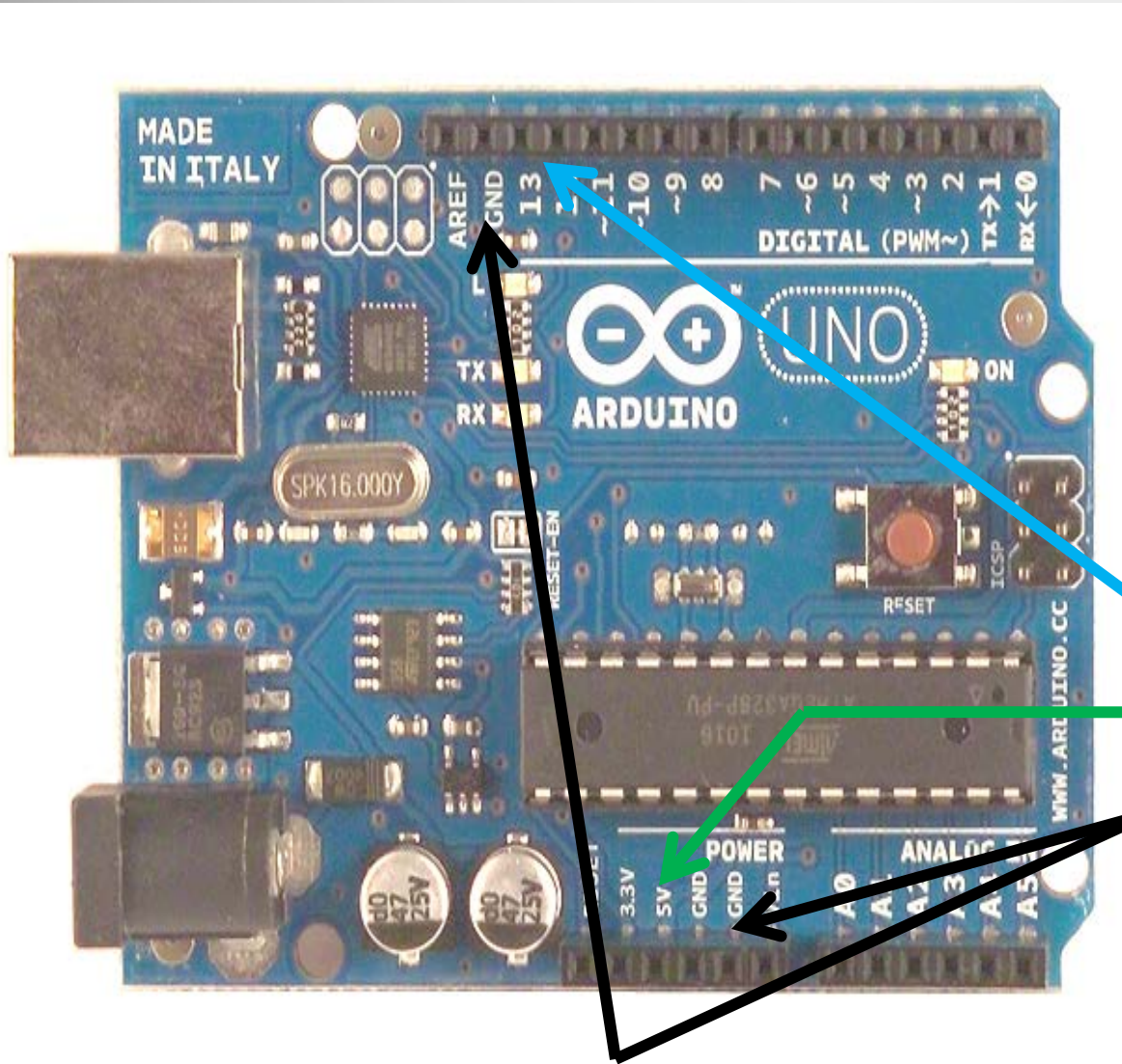
Encontre os seguintes:

13

5 V

GND

# Ligar o Arduino no PC



Observe a identificação dos pinos.

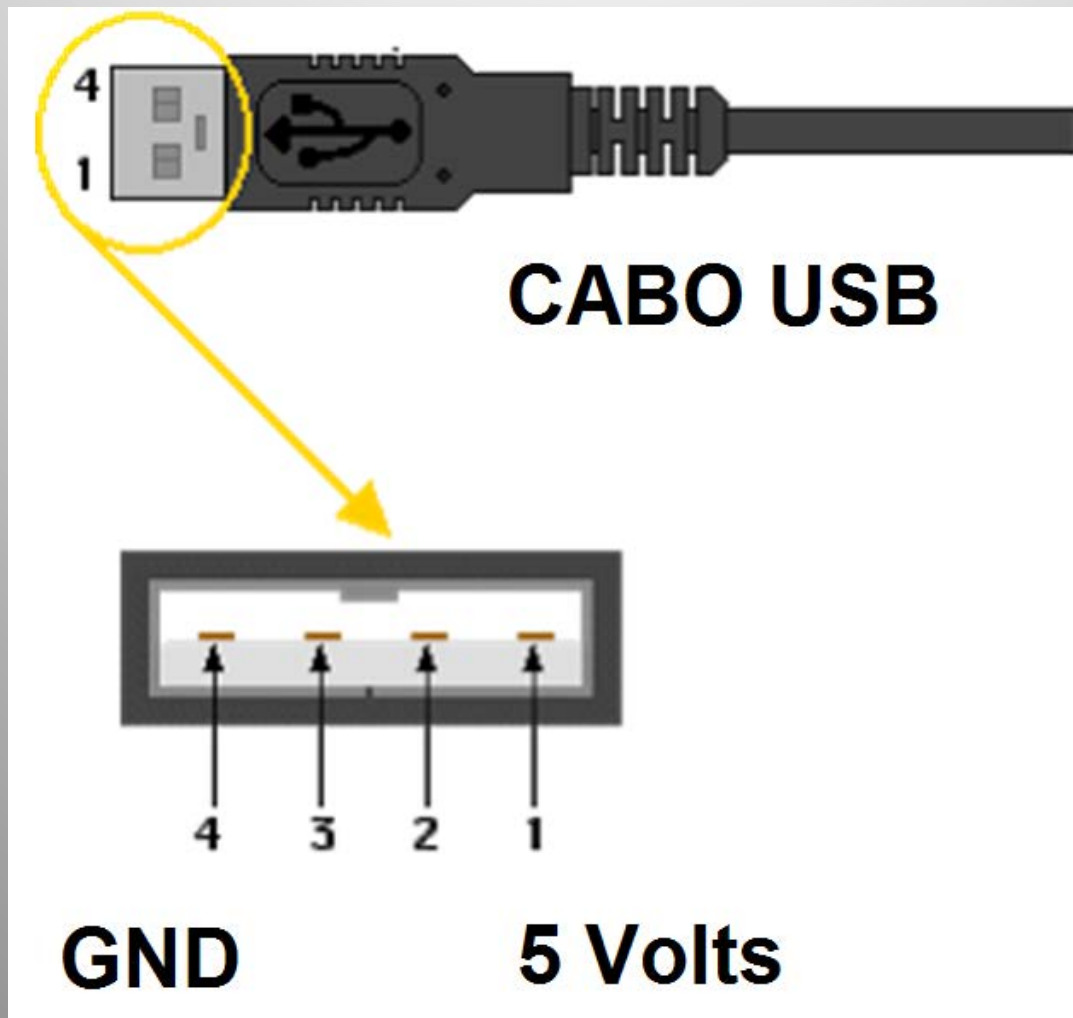
Encontre os seguintes:

13

5 V

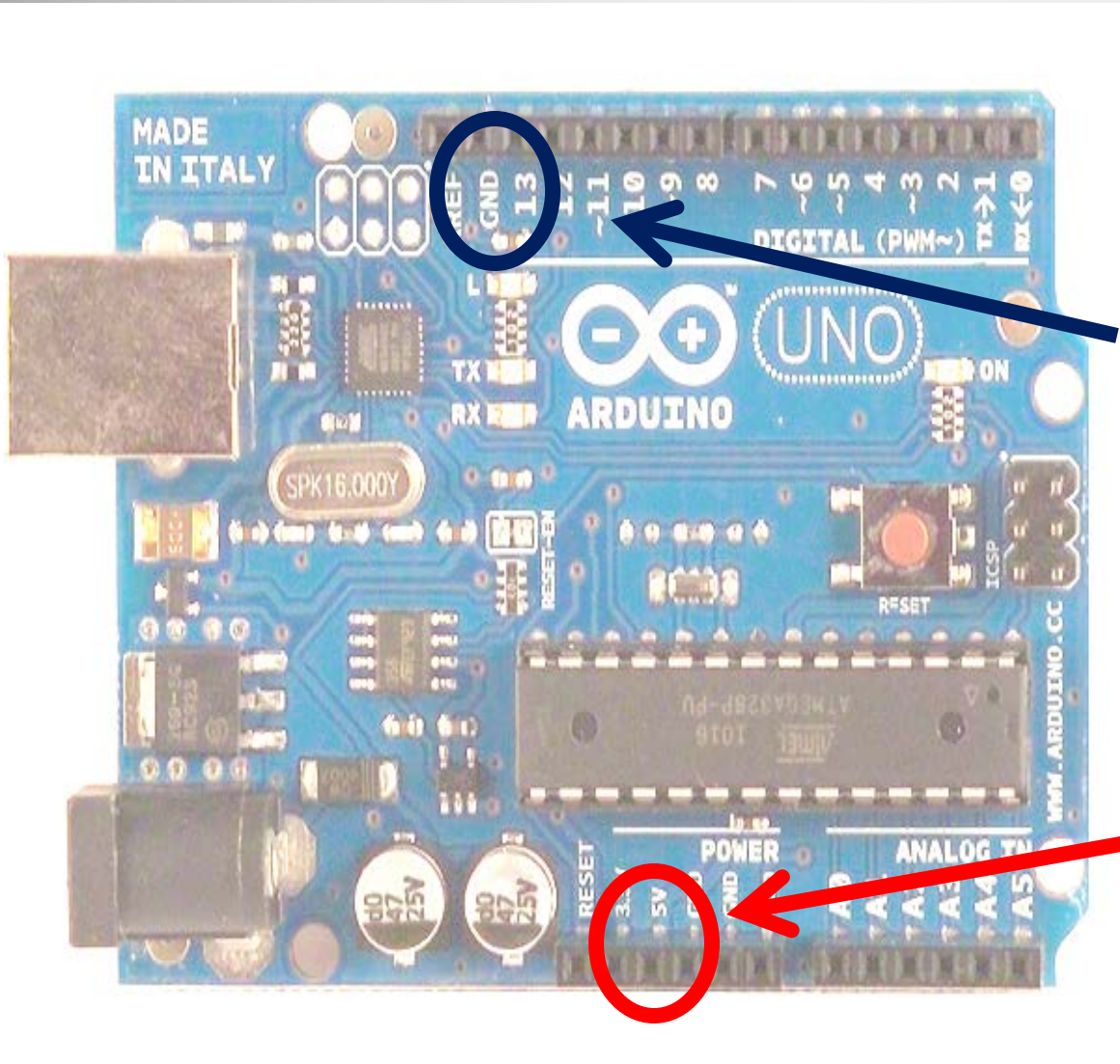
GND

De onde vem a tensão, ou V?





De onde vem a tensão, ou V?



Da Porta USB,

**GND**

e

**5 volts**

# Primeira placa de testes

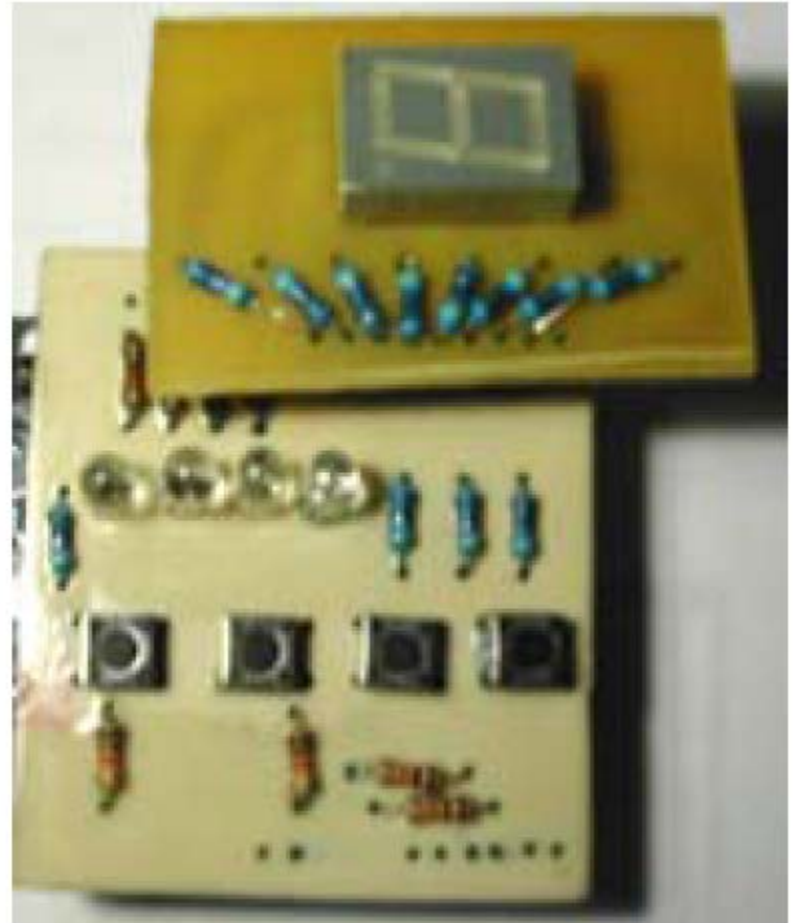


Esse Shield será utilizado na maioria dos testes do Lab.

Possui:

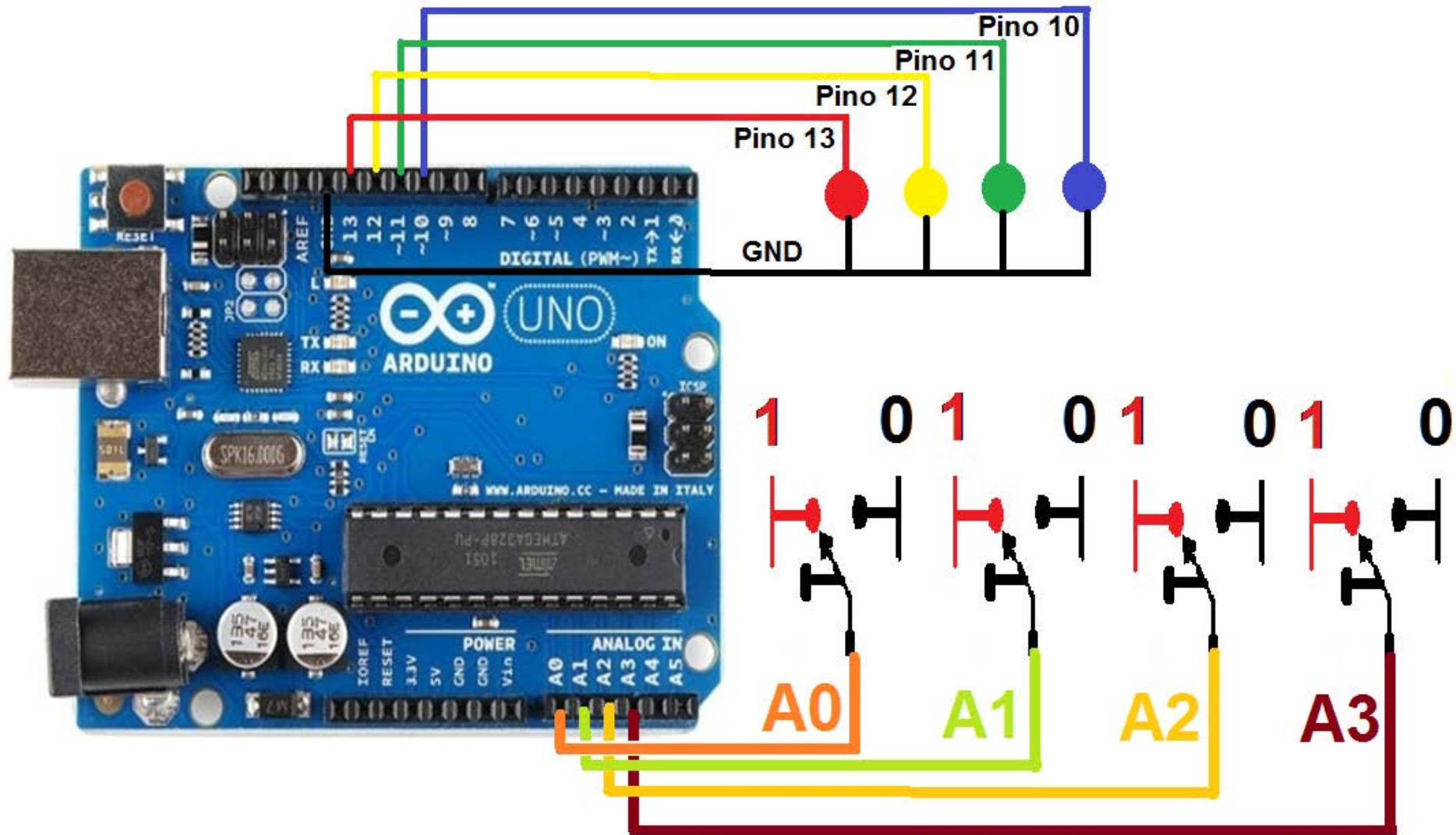
- de 4 a 12 Leds
- 4 chaves
- display de 7 segmentos

# Primeira placa de testes

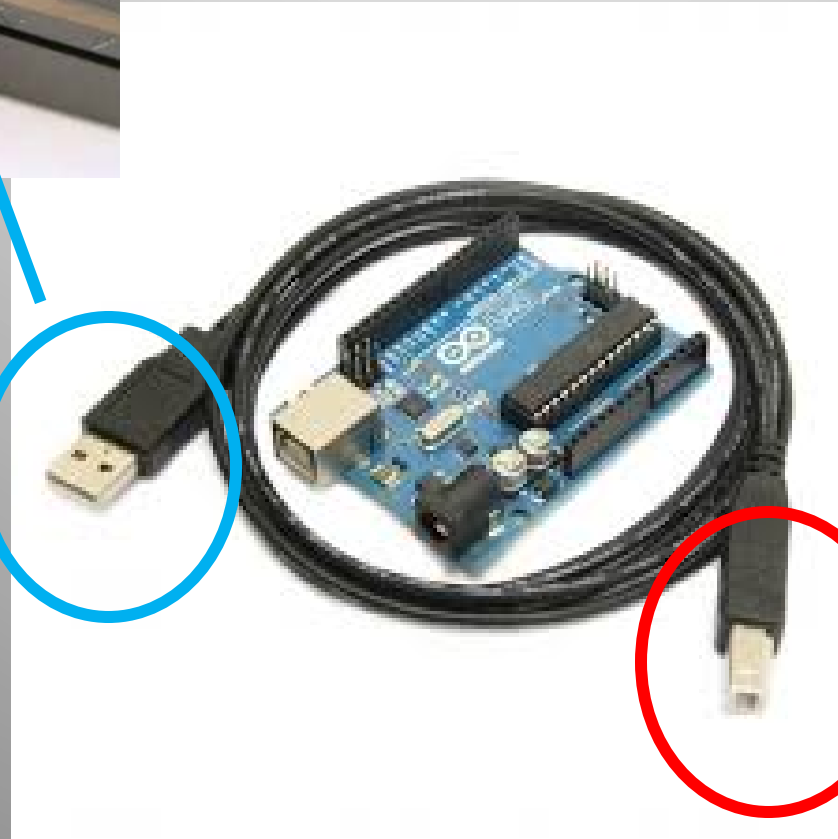
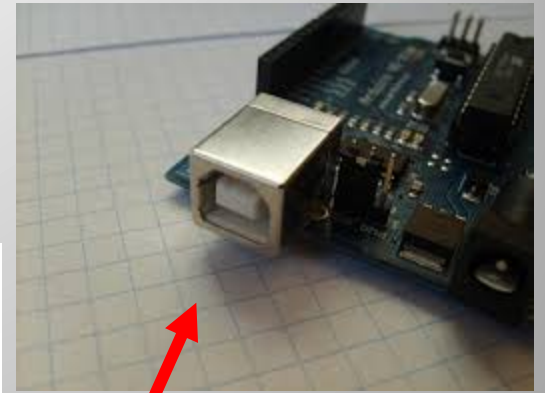




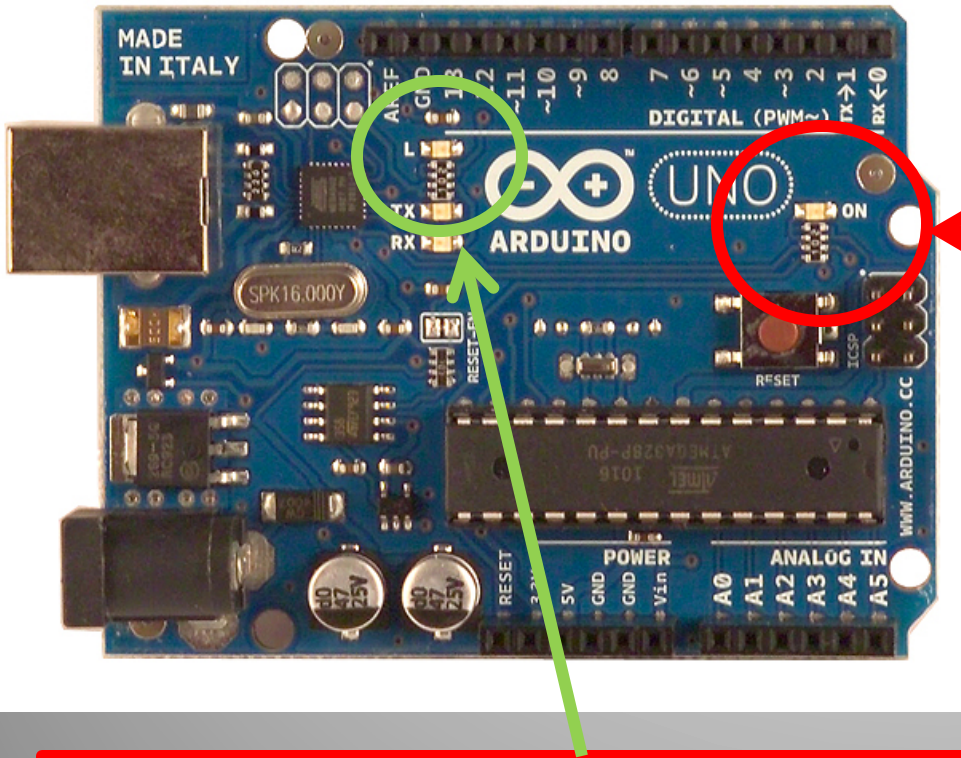
# Esquema das conexões



# Ligar o Arduino no PC



# Ligar o Arduino no PC



Assim que a conexão for feita, um led na placa do arduino deverá acender.

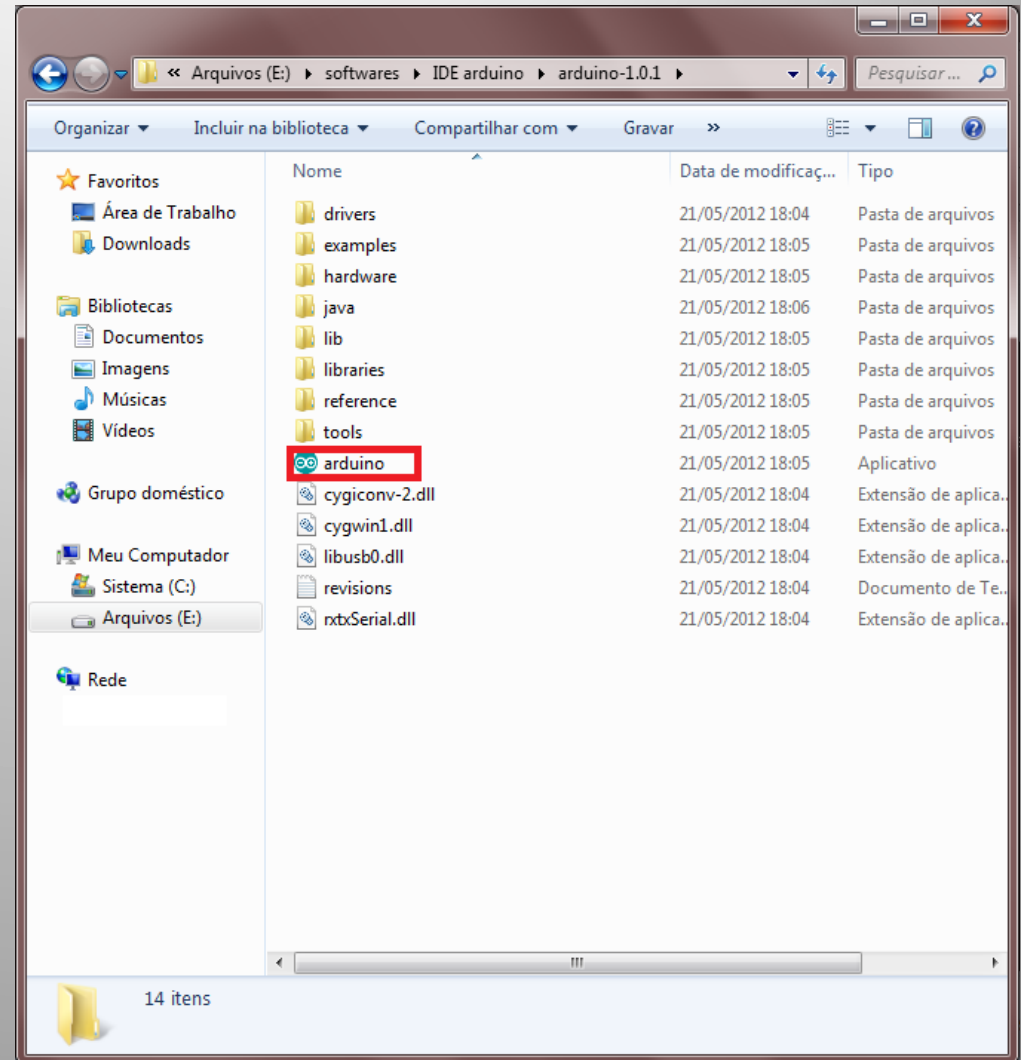
Outro led poderá acender ou piscar, possivelmente algum programa armazenado, **isso não é um problema.**

# Iniciar o programa

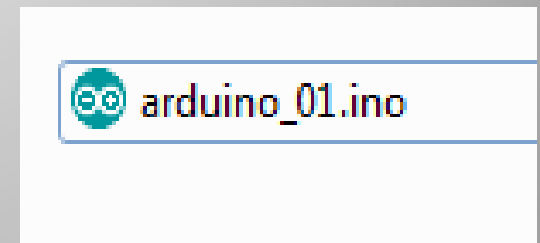
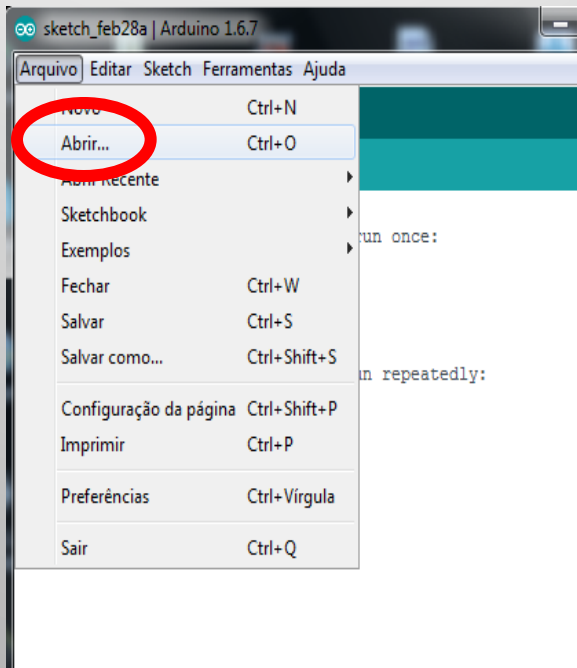
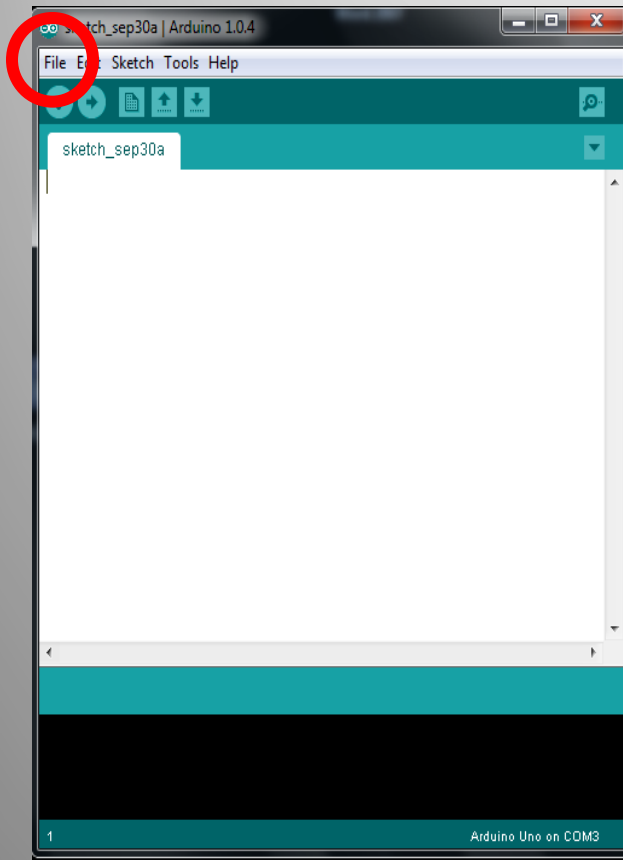
Clicar no ícone do arduino



ou



# Primeiro Programa





# Primeiro Programa

```
Arquivo  Editar  Sketch  Ferramentas  Ajuda

arduino_01 $

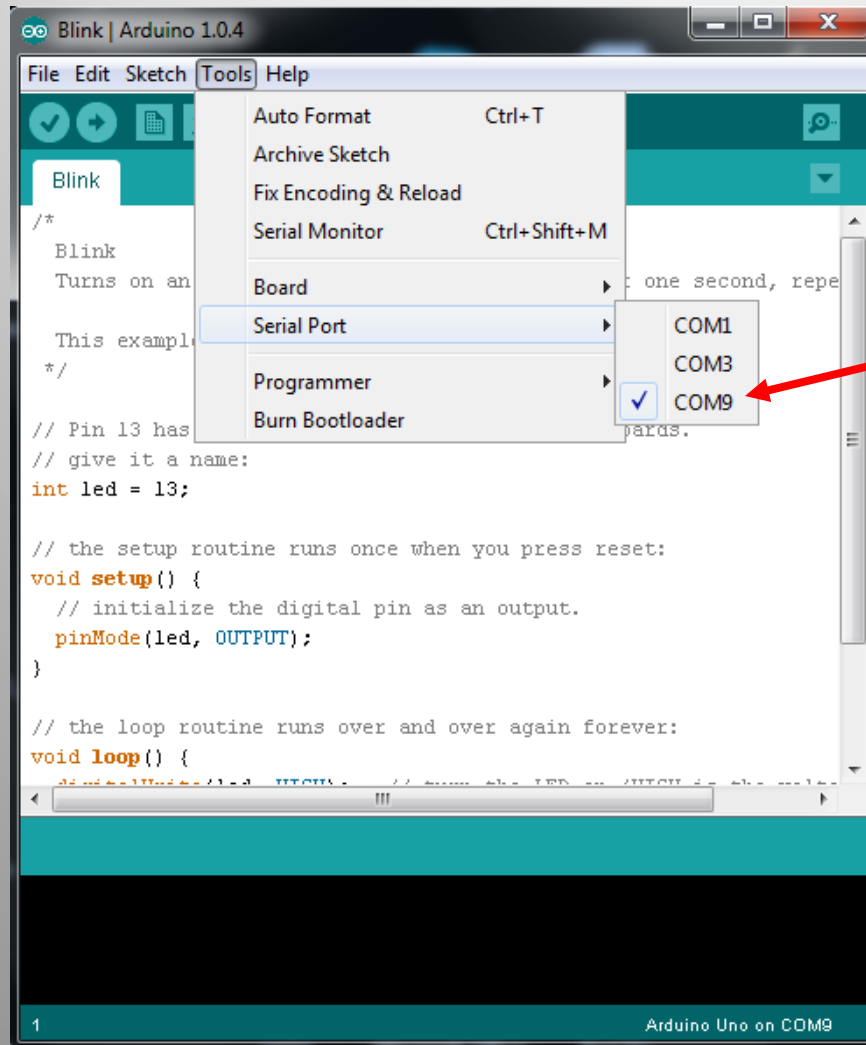
/*
  Programa 01
  Liga todos os Leds
*/

// Definição de valores para variáveis
int led10 = 10;
int led11 = 11;
int led12 = 12;
int led13 = 13;

// Rotina executada 1 vez e que em geral configura
void setup() {
  // configura os pinos como saídas DIGITAIS.
  pinMode(led10, OUTPUT);
  pinMode(led11, OUTPUT);
  pinMode(led12, OUTPUT);
  pinMode(led13, OUTPUT);
}
```

```
// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led10, HIGH); // Faz a saída do respectivo Led ser alta ou High)
  delay(100);                // espera por 100 ms
  digitalWrite(led11, HIGH); // Faz a saída do respectivo Led ser alta ou High)
  delay(100);                // espera por 100 ms
  digitalWrite(led12, HIGH); // Faz a saída do respectivo Led ser alta ou High)
  delay(100);                // espera por 100 ms
  digitalWrite(led13, HIGH); // Faz a saída do respectivo Led ser alta ou High)
  delay(100);                // espera por 100 ms
  digitalWrite(led10, LOW);  // Faz a saída do respectivo Led ser baixa ou Low)
  delay(100);                // espera por 100 ms
  digitalWrite(led11, LOW);  // Faz a saída do respectivo Led ser baixa ou Low)
  delay(100);                // espera por 100 ms
  digitalWrite(led12, LOW);  // Faz a saída do respectivo Led ser baixa ou Low)
  delay(100);                // espera por 100 ms
  digitalWrite(led13, LOW);  // Faz a saída do respectivo Led ser baixa ou Low)
  delay(100);                // espera por 100 ms
}
```

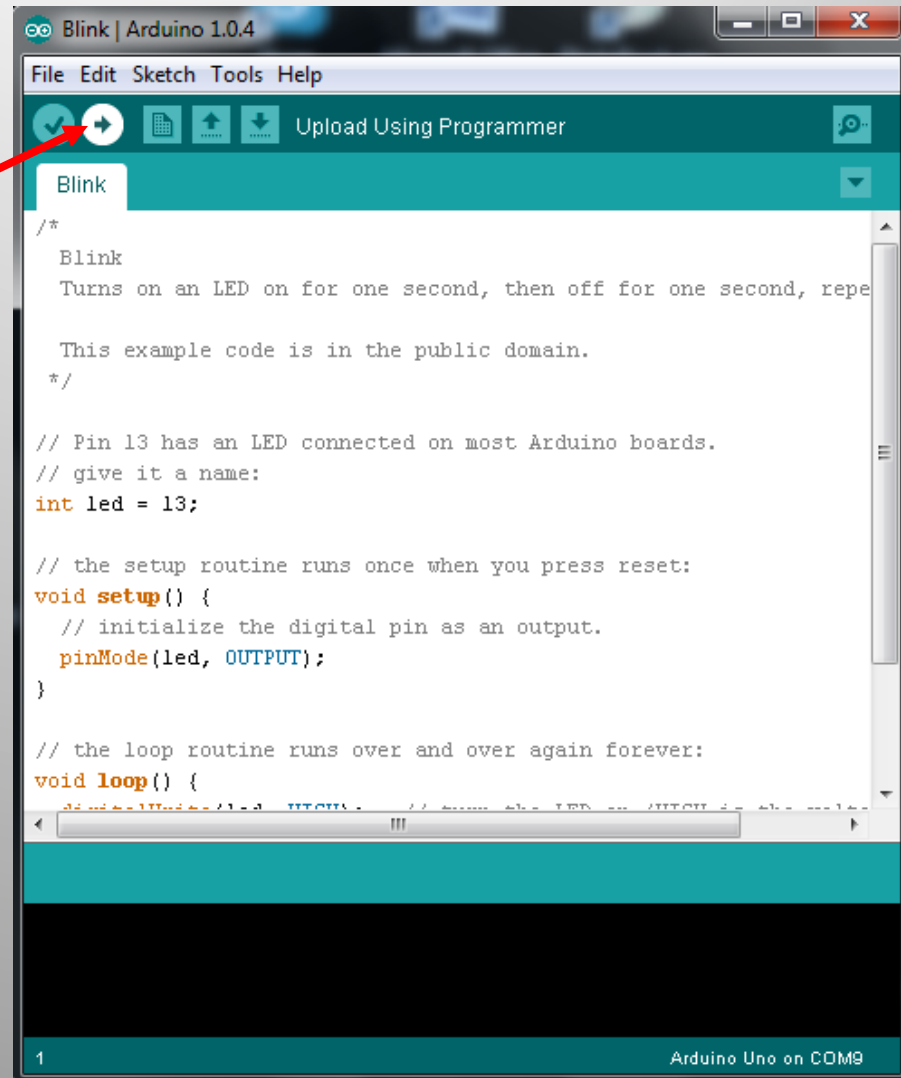
# Primeiro Programa



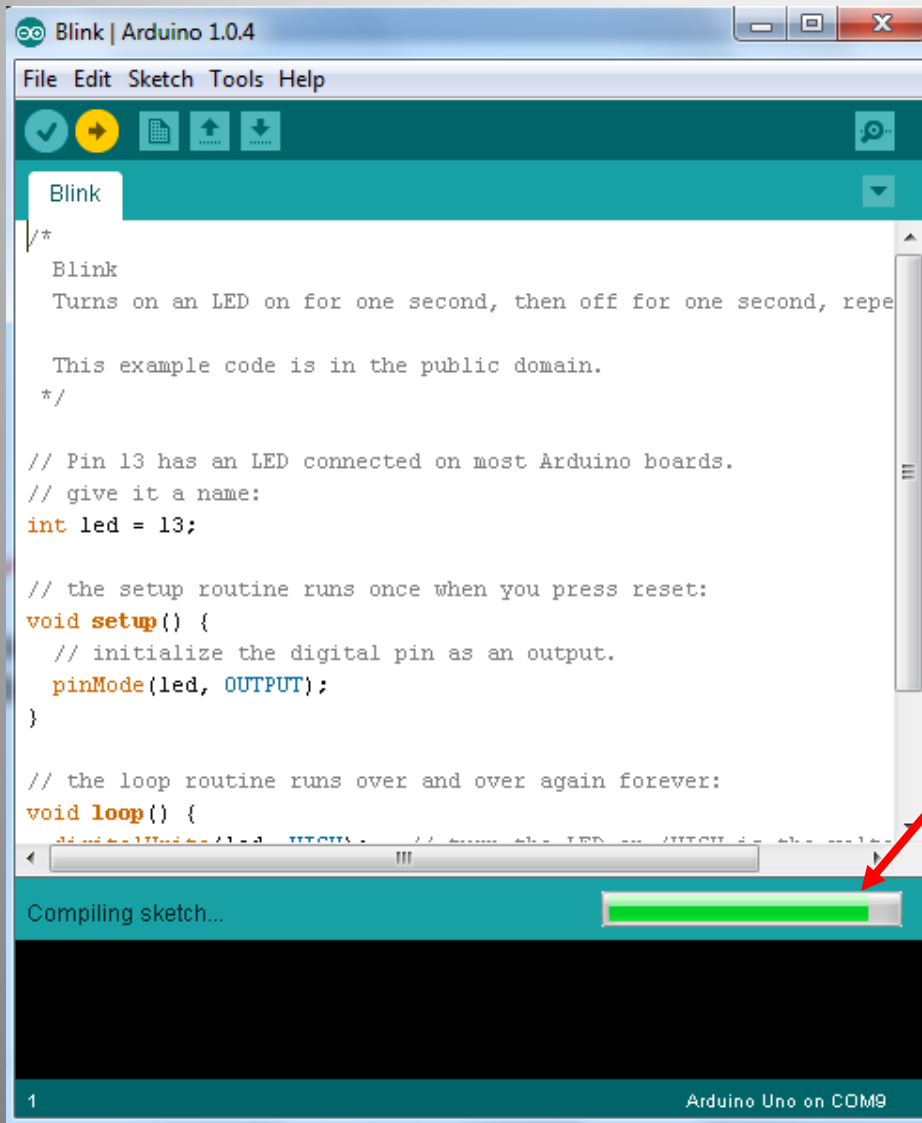
Onde o arduino  
está conectado

# Primeiro Programa

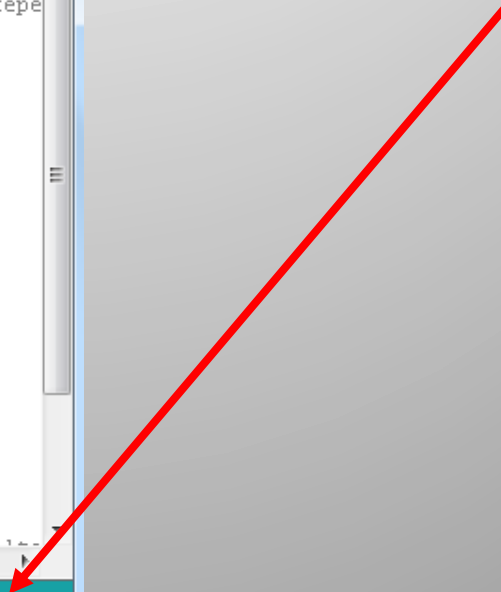
Clicar nesse botão para fazer o upload do programa no arduino



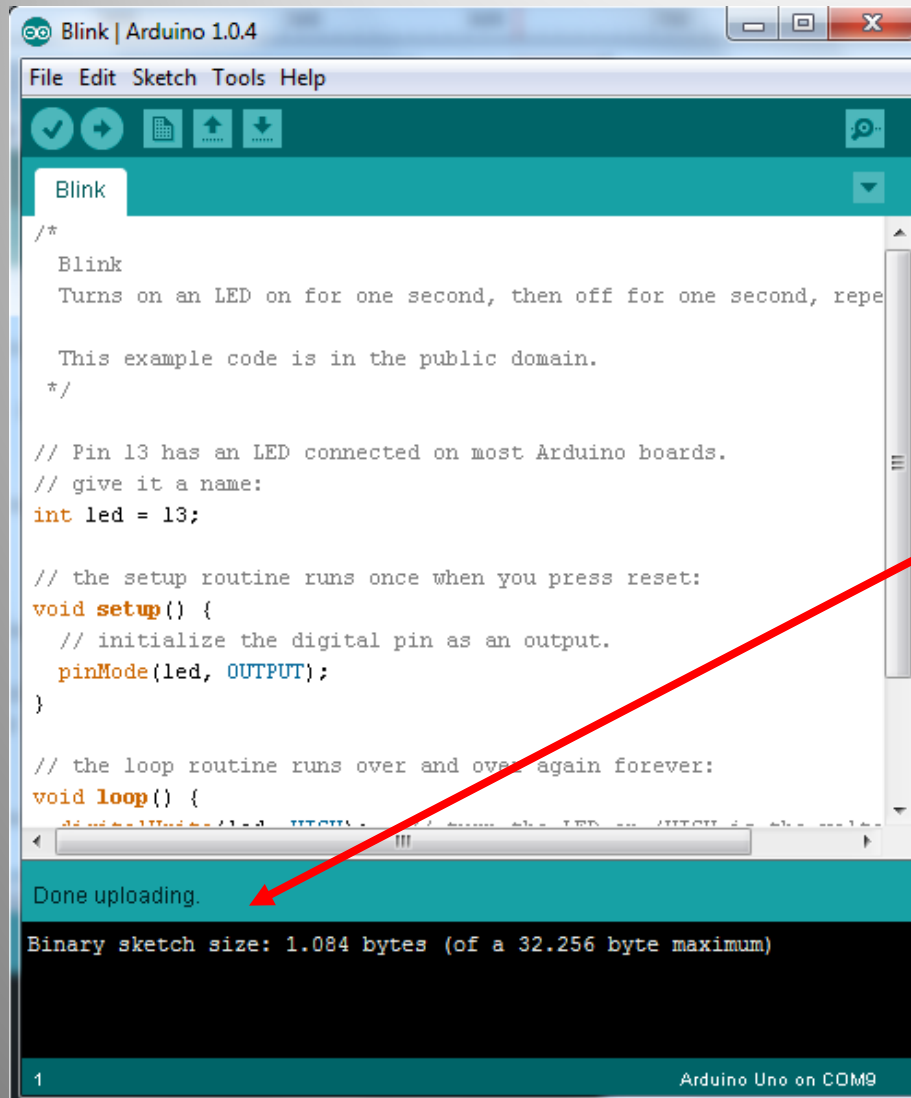
# Primeiro Programa



Programa sendo carregado  
no arduino



# Primeiro Programa



The screenshot shows the Arduino IDE interface with the 'Blink' sketch loaded. The code is as follows:

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeating.

  This example code is in the public domain.
  */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(led, LOW);   // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

At the bottom of the window, a teal status bar displays the message: "Done uploading." A red arrow points from the text on the right towards this message. Below the status bar, a black console area shows the text: "Binary sketch size: 1.084 bytes (of a 32.256 byte maximum)". The bottom-most bar indicates the board and port: "1" and "Arduino Uno on COM9".

Mensagem indicando que o programa foi carregado sem erros



# Entendendo o Programa

- comandos básicos da linguagem
  - `pinMode ( pino, modo )`

Esta define um pino com entrada ou saída. O arduino possui 20 pinos disponíveis, 14 digitais e 6 analógicos (0 a 13 e 14 a 19). Esse comando deverá estar preferencialmente na função `setup ()`.

Exemplo:

```
pinMode (13, OUTPUT);  
pinMode (10, INPUT);
```

# Entendendo o Programa

- **digitalWrite ( pino, valor )**

Liga ou desliga uma saída digital, apenas caso o pino seja definido como OUTPUT.

Os valores podem ser HIGH ou LOW (1 ou 0).

Exemplo:

```
digitalWrite (13, HIGH);  
digitalWrite (13, LOW);
```

# Entendendo o Programa

- `delay( ms)`

Aguarda o tempo passado como argumento em ms.

Exemplo:

```
delay (1000) ;
```

# Entendendo o Programa

```
int led = 13;
void setup() {
  pinMode(led, OUTPUT);
}
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

Definimos uma variável **led**,  
no caso será o nosso pino 13

Avisamos que será uma  
saída, pois iremos ligar  
um led.

Saída digital led recebe 1

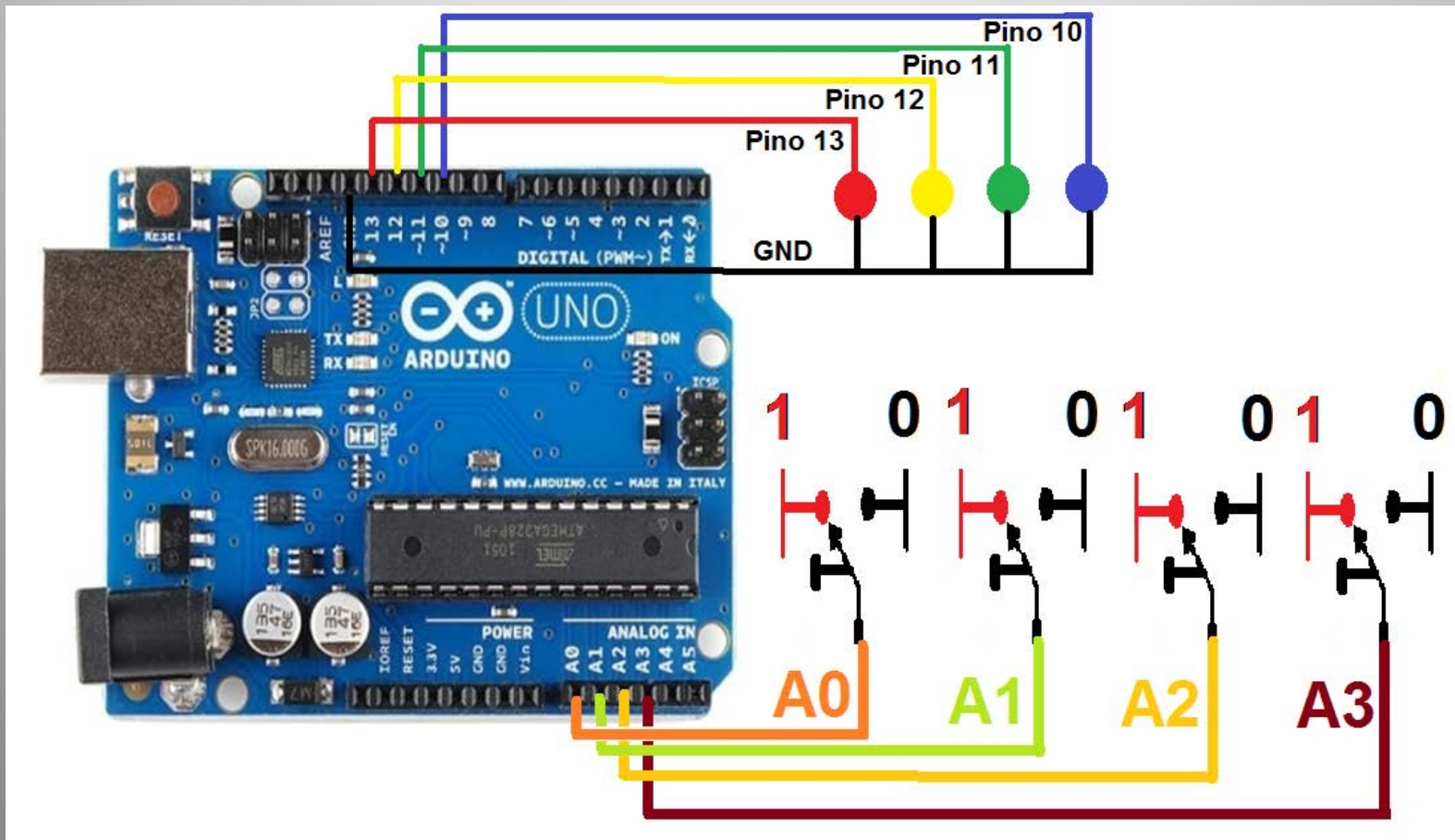
Saída digital led recebe 0

Aguardamos 1 segundo



# Entrada de Dados

Para esta montagem iremos precisar saber onde as chaves estarão conectadas.



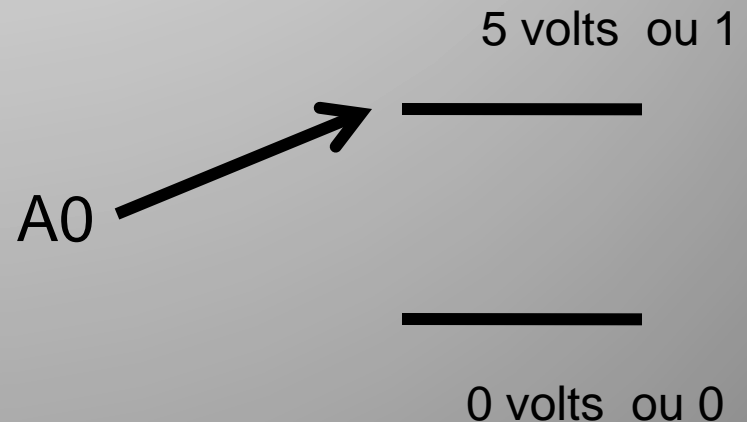
# Entrada de Dados

A idéia é simularmos um interruptor.

Atenção que você ainda não sabe de que forma a chave está ligada. Isso depende do Hardware.

Como as chaves estão ligadas no módulo:

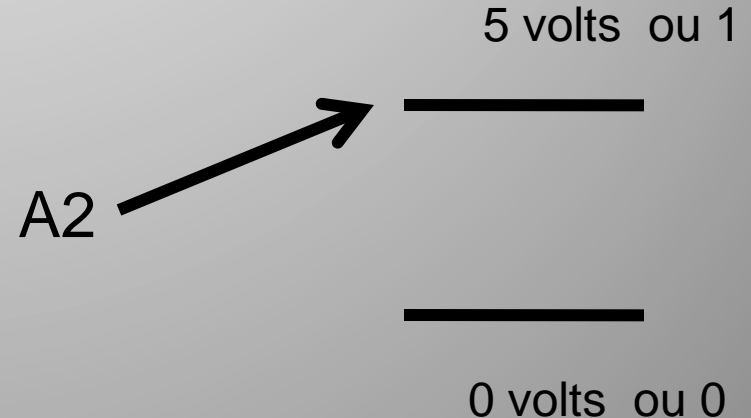
Normalmente, sem apertá-la, temos o valor 1 na entrada A0. Apertando teremos o valor zero.



# Entrada de Dados

```
int chave = A2;  
int led = 13;  
int estadochave = 0;  
void setup() {  
    pinMode(led, OUTPUT);  
    pinMode(chave, INPUT);  
    digitalWrite(chave, HIGH);  
}  
void loop(){  
    estadochave = digitalRead(chave);  
    if (estadochave == HIGH) {  
        digitalWrite(led, HIGH);  
    }  
    else {  
        digitalWrite(led, LOW);  
    }  
}
```

Este programa verifica o estado da chave, se ela for apertada teremos uma entrada zero, e assim, o led se apaga.



# arduino\_02.ino

arduino\_02 §

```
// Chaves
int botao1 = A0;    // define primeiro botao
int botao2 = A1;    // define segundo botao
int botao3 = A2;    // define terceiro botao
int botao4 = A3;    // define quarto botao
int led1 = 13;      // pino do led
int led2 = 12;
int led3 = 11;
int led4 = 10;

// variaveis de controle dos botoes
int pos_bot1 = 0;
int pos_bot2 = 0;
int pos_bot3 = 0;
int pos_bot4 = 0;

void setup() {
// Define as entradas e saídas
pinMode(led1, OUTPUT);
pinMode(led2, OUTPUT);
pinMode(led3, OUTPUT);
pinMode(led4, OUTPUT);

pinMode(botao1, INPUT);
pinMode(botao2, INPUT);
pinMode(botao3, INPUT);
pinMode(botao4, INPUT);
}
```

```
void loop() {
// lê estado dos botoes:
pos_bot1 = digitalRead(botao1);
pos_bot2 = digitalRead(botao2);
pos_bot3 = digitalRead(botao3);
pos_bot4 = digitalRead(botao4);
if (pos_bot1 == HIGH) {
    digitalWrite(led1, HIGH);
}
else {
    digitalWrite(led1, LOW);
}
if (pos_bot2 == HIGH) {
    digitalWrite(led2, HIGH);
}
else {
    digitalWrite(led2, LOW);
}
if (pos_bot3 == HIGH) {
    digitalWrite(led3, HIGH);
}
else {
    digitalWrite(led3, LOW);
}
if (pos_bot4 == HIGH) {
    digitalWrite(led4, HIGH);
}
}
```



# Entendendo o Programa

- **digitalRead ( pino )**

Lê o estado lógico de um pino. Uma variável deverá ser definida para receber o valor (1 ou 0).

Exemplo:

```
int botão;  
botão = digitalRead (10);
```

Obs.: O pino 10 deverá ser definido anteriormente como entrada.

```
char entrada = 'c';
int led = 13;
int i;
void setup() {
    Serial.begin(9600);    // abre a porta serial a 9600 bps
    pinMode(led, OUTPUT);
}

void loop() {

    // verifica se existe dados a ser lido
    if (Serial.available() > 0)
    {
        // lê o dado
        entrada = Serial.read();
        if (entrada != '\n')
        {
            if (entrada == 'a')
                digitalWrite(led, HIGH);
            if (entrada == 'b')
                digitalWrite(led, LOW);
            Serial.println(entrada);
        }
    }
}
```

# Entendendo o Programa

- **Serial.begin ( velocidade )**

Essa função abre uma comunicação serial na velocidade passada como argumento. A função usa os pinos 0 e 1 para recepção e transmissão.

Exemplo:

```
Serial.begin ( 9600 ) ;
```

# Entendendo o Programa

- **Serial.println ( valor )**

Essa função transmite os caracteres ascii do arduino para o computador.

Exemplo:

```
Serial.println ( " Alo, tudo bem? " ) ;
```

# Entendendo o Programa

- `Serial.read ( )`

Essa função lê um byte recebido pelo arduino.

Exemplo:

```
int recebido;  
recebido = Serial.read ( ) ;
```



# Entendendo o Programa

- **Serial.available ( )**

Essa função retorna o número de bytes disponíveis para leitura na porta serial.

Exemplo:

```
int recebido;  
if (Serial.available () > 0 )  
    recebido = Serial.read ( ) ;
```

# Comunicação Serial

```
char entrada = '0';  
void setup()  
{  
  Serial.begin(9600);  
}
```

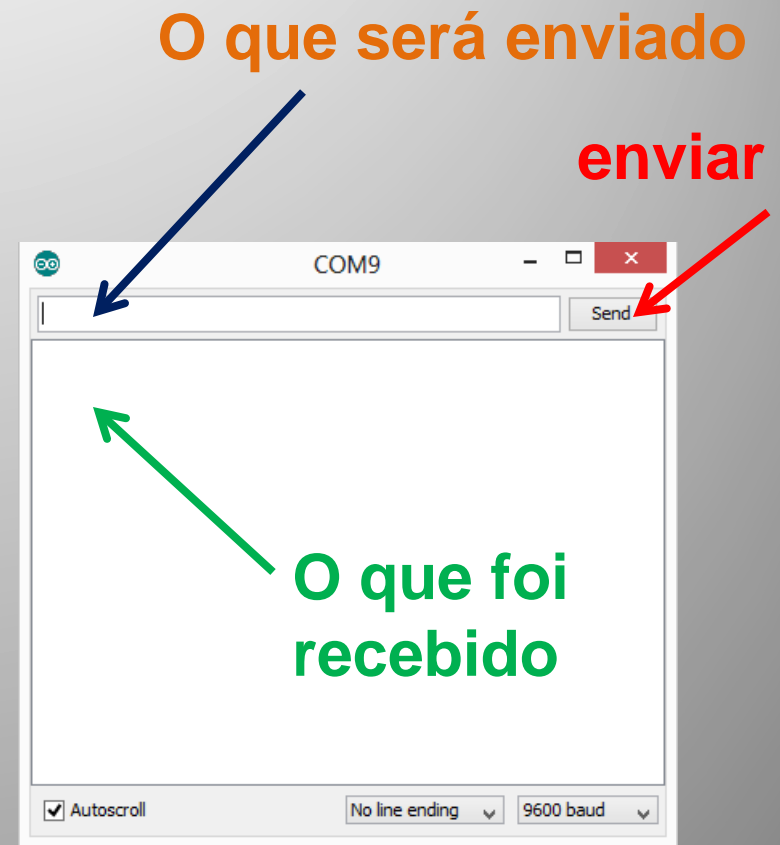
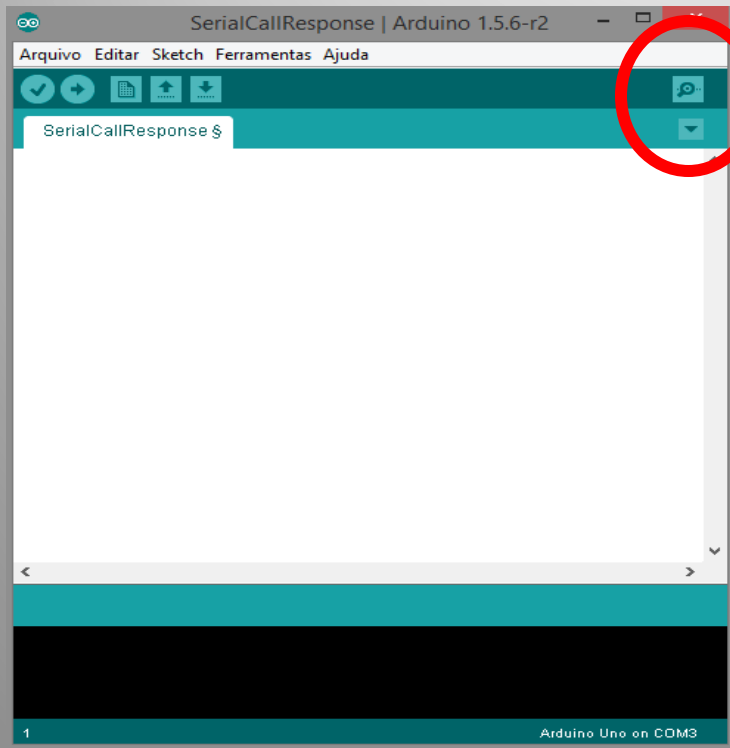
```
void loop()  
{  
  if (Serial.available() > 0) {  
    entrada = Serial.read();  
    Serial.print(entrada);  
  }  
}
```

**Lê da porta serial**

**Escreve na porta serial**

# Comunicação Serial

- ▶ Carregar o programa no arduino
- ▶ Executar a comunicação serial



# Comunicação Serial - Resumo

- `available()`    Obtém o número de bytes disponíveis para leitura na porta serial.
- `begin()`    Configura a taxa de dados em bauds para transmissão serial de dados
- `end()`    Desabilita a comunicação serial, permitindo que os pinos TX e RX (0 e 1) sejam usados para entrada e saída geral de sinais.
- `print()`    Imprime dados na porta serial em formato legível por humanos (texto ASCII)
- `println()`    Idem a `print()`, porém adicionando um caractere de retorno de carro (ASCII 13) e um de nova linha (ASCII 10) no final da string.
- `read()`    Lê dados a partir da porta serial.
- `readBytes()`    Lê caracteres a partir da porta serial em um buffer até que uma determinada quantidade de caracteres tenha sido lida.
- `write()`    Escreve dados binários na porta serial

```
int entrada = 0;
int led = 13;

void setup() {
    Serial.begin(9600);
    pinMode(led, OUTPUT);
}

void loop() {

    if (Serial.available() > 0) {
        entrada = Serial.parseInt();
        if (entrada != '\n')
        {
            if (entrada == 1)
                digitalWrite(led, HIGH);
            if (entrada == 2)
                digitalWrite(led, LOW);

            Serial.print(entrada);
        }
    }
}
```



```
int entrada1 = 0;
int entrada2 = 0;
int saida;
int led = 13;
void setup() {
    Serial.begin(9600);
    pinMode(led, OUTPUT);
}
void loop() {
    if (Serial.available() > 0) {
        entrada1 = Serial.parseInt();
        entrada2 = Serial.parseInt();
        if (Serial.read() == '\n')
        {
            saida = soma(entrada1, entrada2);

            Serial.print(saida);

            if (saida == 1)
                digitalWrite(led, HIGH);
            if (saida == 2)
                digitalWrite(led, LOW);
        }
    }
}
int soma(int a, int b)
{
    return(a+b);
}
```

```
int entrada1 = 0;
int entrada2 = 0;
int entrada3 = 0;
int saida;
int led = 13;
void setup() {
    Serial.begin(9600);
    pinMode(led, OUTPUT);
}
void loop() {
    if (Serial.available() > 0) {
        entrada1 = Serial.parseInt();
        entrada2 = Serial.parseInt();
        entrada3 = Serial.parseInt();

        if (Serial.read()=='\n')
        {
            saida = soma(entrada1, entrada2, entrada3);
            Serial.print(saida);
            if (saida == 1)
                digitalWrite(led, HIGH);
            if (saida == 2)
                digitalWrite(led, LOW);
        }
    }
}
```

```
int soma(int a, int b, int c)
{
    return (a+b+c);
}
```

## Exercícios:

1) Se o arduino receber a letra 'a' um Led pisca em uma velocidade, se receber uma letra 'b' ele irá piscar em outra velocidade.

## Solução 2:

```
int led = 13;

char letra;

char recebido = 'a';

void setup() {

    pinMode(led, OUTPUT);

    Serial.begin(9600);
}
```

```
void loop() {
    if (Serial.available() > 0) {
        letra = Serial.read();
        if (letra=='a')
            recebido = 'a';
        if (letra=='b')
            recebido = 'b';
    }

    if (recebido == 'a') {
        digitalWrite(led, HIGH);
        delay(100);
        digitalWrite(led, LOW);
        delay(100);
    }

    if (recebido == 'b') {
        digitalWrite(led, HIGH);
        delay(50);
        digitalWrite(led, LOW);
        delay(50);
    }
}
```

```
/* Use File->Load Prog to
   load a different Program
*/

int count;

void setup()
{
  count=0;
}

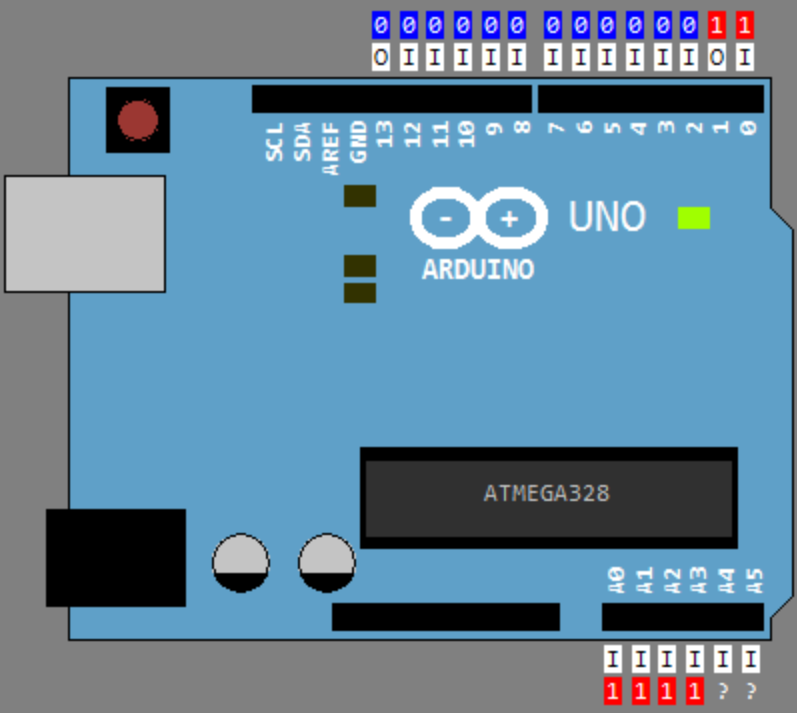
void loop()
{
  count=count+1;
  delay(100);
}

//the "int main()" below is IMPLICIT in Arduino
//but is shown here EXPLICITLY by UnoArduSim
int main()
{
  setup();
  while(true)
  {
    loop();
    serialEventRun();
  }
}
```

count= 0

Hardware control panel with buttons for digital pins (A0-A3), LEDs (10-13), and switches.

Serial monitor interface showing TX and RX character counts and a Baud rate of 9600.



Right-side hardware control panel with buttons for digital pins (0-9), LEDs (10-13), and switches.