

**INSTITUTO
FEDERAL**
Santa Catarina

Câmpus
Chapecó

IFSC - TDS Programação IV (C + Arduino)



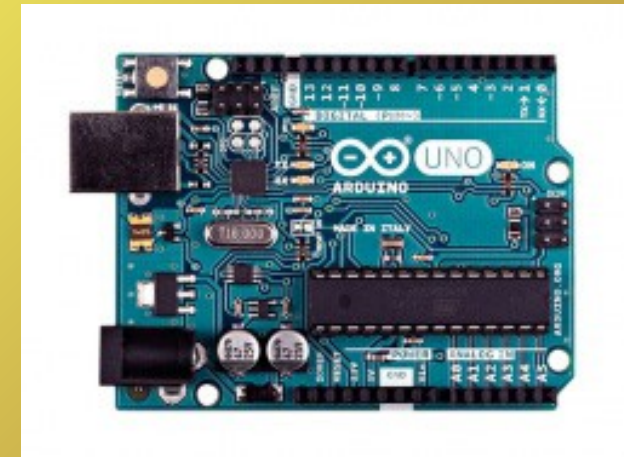
O que é Arduino ?

- O Arduino foi criado em 2005 por um grupo de 5 pesquisadores : Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis. O objetivo era elaborar um dispositivo que fosse ao mesmo tempo barato, funcional e fácil de programar, sendo dessa forma acessível a estudantes e projetistas amadores. Além disso, foi adotado o conceito de hardware livre, o que significa que qualquer um pode montar, modificar, melhorar e personalizar o Arduino, partindo do mesmo hardware básico.
- <https://www.filipeflop.com/blog/o-que-e-arduino/>

Arduino é uma ferramenta de código aberto usado na construção de projetos eletrônicos(*protótipos*).

Arduino é composto por uma placa física programável, um circuito e um ambiente de desenvolvimento, ou IDE, que é executado em seu computador, é utilizado para escrever(linguagem c) e fazer upload de código do computador para a placa.

Depois de programado, o microcontrolador pode ser usado de forma independente, ou seja, você pode colocá-lo para controlar um robô, uma lixeira, um ventilador, as luzes da sua casa, a temperatura do ar condicionado, pode utilizá-lo como um aparelho de medição ou qualquer outro projeto que vier à cabeça.



Modelos de placas Arduino

O Arduino Uno é uma placa de microcontrolador baseado no ATmega328 (datasheet). Ele tem 14 pinos de entrada/saída digital (dos quais 6 podem ser usados como saídas PWM), 6 entradas analógicas, um cristal oscilador de 16MHz, uma conexão USB, uma entrada de alimentação uma conexão ICSP e um botão de reset. Ele contém todos os componentes necessários para suportar o microcontrolador, simplesmente conecte a um computador pela porta USB ou alimentar com uma fonte ou com uma bateria e tudo pronto para começar.

Dados técnicos:

Microcontrolador	ATmega328
Tensão de operação	5V
Tensão de alimentação (recomendada)	7-12V
Tensão de alimentação (limite)	6-20V
Entradas e saídas digitais	14 das quais 6 podem ser PWM
Entradas analógicas	6
Corrente contínua por pino de I/O	40 mA
Corrente contínua para o pino 3.3V	50 mA
Memória Flash	32 KB (ATmega328) dos quais 0.5 KB são usados pelo bootloader
Memória SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Velocidade do Clock	16 MHz
Dimensões	68,58mm x 53,34mm
Peso	150g

Arduino Mega

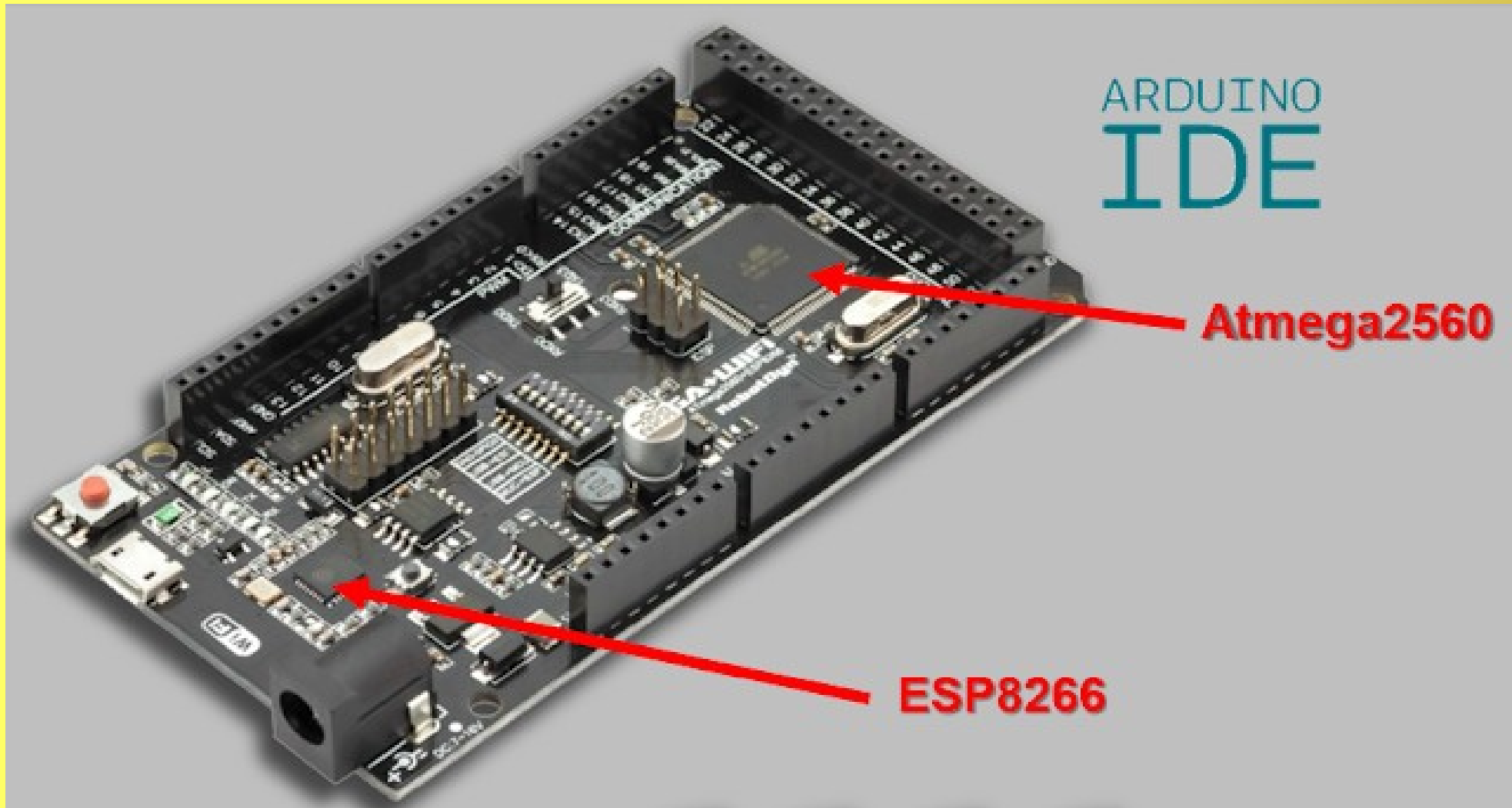
A Placa Arduino Mega 2560 é uma placa de microcontrolador baseada no ATmega2560 (datasheet). Ele possui 54 pinos de entradas/saídas digitais, 16 entradas analógicas, 4 UARTs (portas seriais de hardware), um oscilador de cristal de 16 MHz, uma conexão USB, uma entrada de alimentação, uma conexão ICSP e um botão de reset. Ele contém tudo o que é necessário para dar suporte ao microcontrolador; basta conectar a um computador com um cabo USB ou a uma fonte de alimentação e já está pronto para começar. O mega é compatível com a maioria dos shields desenhados para os Arduino Uno, Duemilanove e para o Diecimila. Possui ainda o dobro de memória do antigo Arduino Mega.

Dados técnicos:

Microcontrolador	ATmega2560
Tensão de operação	5V
Tensão de alimentação (recomendada)	7-12V
Tensão de alimentação (limite)	6-20V
Entradas e saídas digitais	54 das quais 15 podem ser PWM
Entradas analógicas	16
Corrente contínua por pino de I/O	40 mA
Corrente contínua para o pino 3.3	50 mA
Memória Flash	256 KB dos quais 8KB são usados pelo bootloader
Memória SRAM	8 KB
EEPROM	4 KB
Velocidade do Clock	16 MHz
Dimensões	101,6mm x 53,4mm
Peso	150g



Arduino Mega com ESP8266



Acessórios de proteção

Este case é um acessório fundamental para proteger a sua placa Arduino Uno. Ele disponibiliza acesso aos pinos e terminais da placa, além é claro de deixar o seu projeto com uma aparência muito mais profissional e um design único.

Acompanha parafusos para fixação da placa no interior do case, e este por sua vez também pode ser fixado a uma parede ou móvel, conforme sua furação na parte de trás do case. Sua montagem é bastante simples e rápida.



O que você pode fazer com o Arduino ?

A lista de possibilidades é praticamente infinita. Você pode automatizar sua casa, seu carro, seu escritório, criar um novo brinquedo, um novo equipamento ou melhorar um já existente. Tudo vai depender da sua criatividade.

Para isso, o Arduino possui uma quantidade enorme de *sensores e componentes* que você pode utilizar nos seus projetos. Grande parte do material utilizado está disponível em módulos, que são pequenas placas que contém os sensores e outros componentes auxiliares como resistores, capacitores e leds.

Sensores



Sensor de chuva

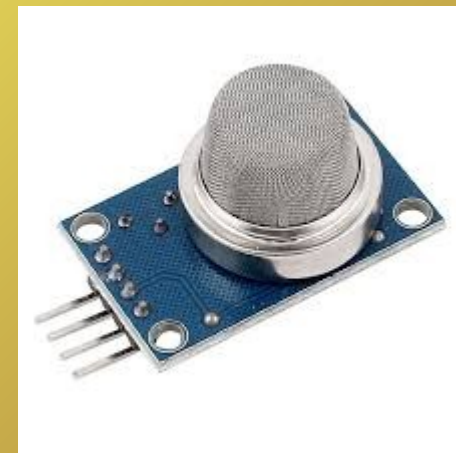


Sensor Fotoelétrico
(luz)

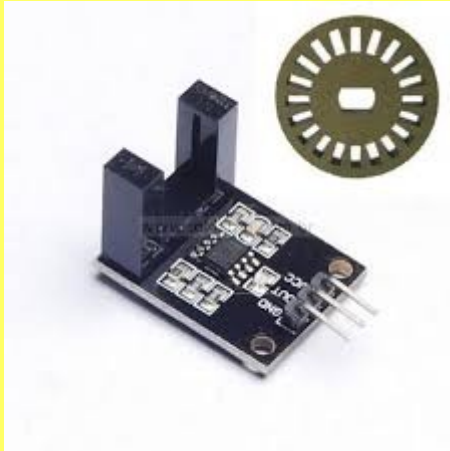


Sensor de distância
(ultrasom)

Sensor de detecção
de gás



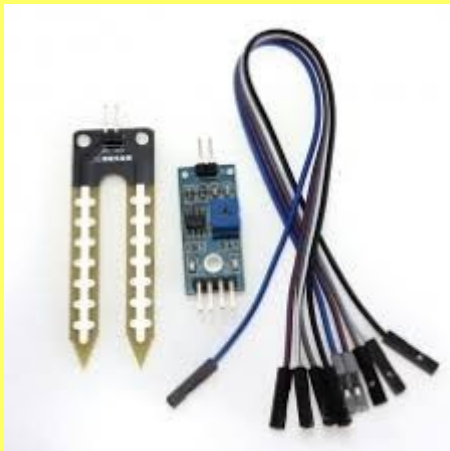
Sensores



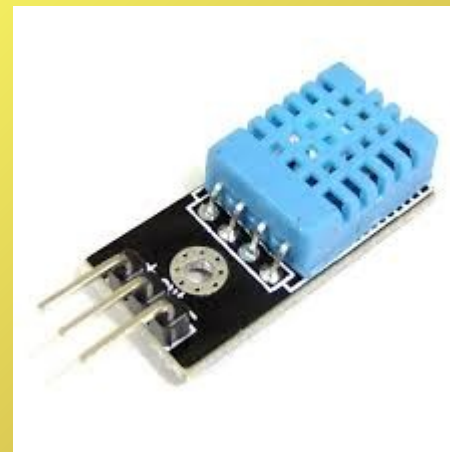
Sensor de
velocidade



Sensor de
presença



Sensor de
umidade

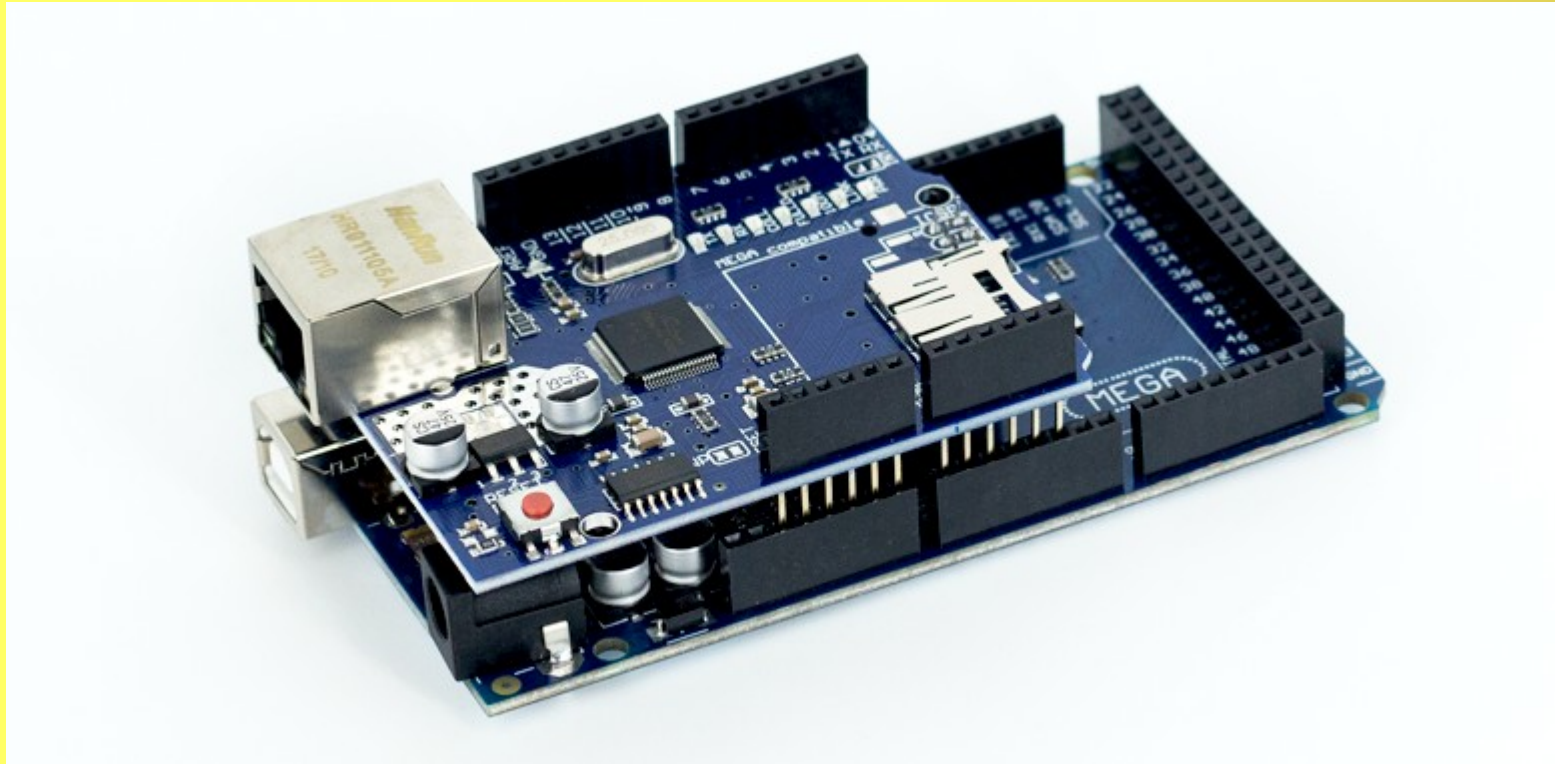


Sensor de umidade
e temperatura

Arduino Shield

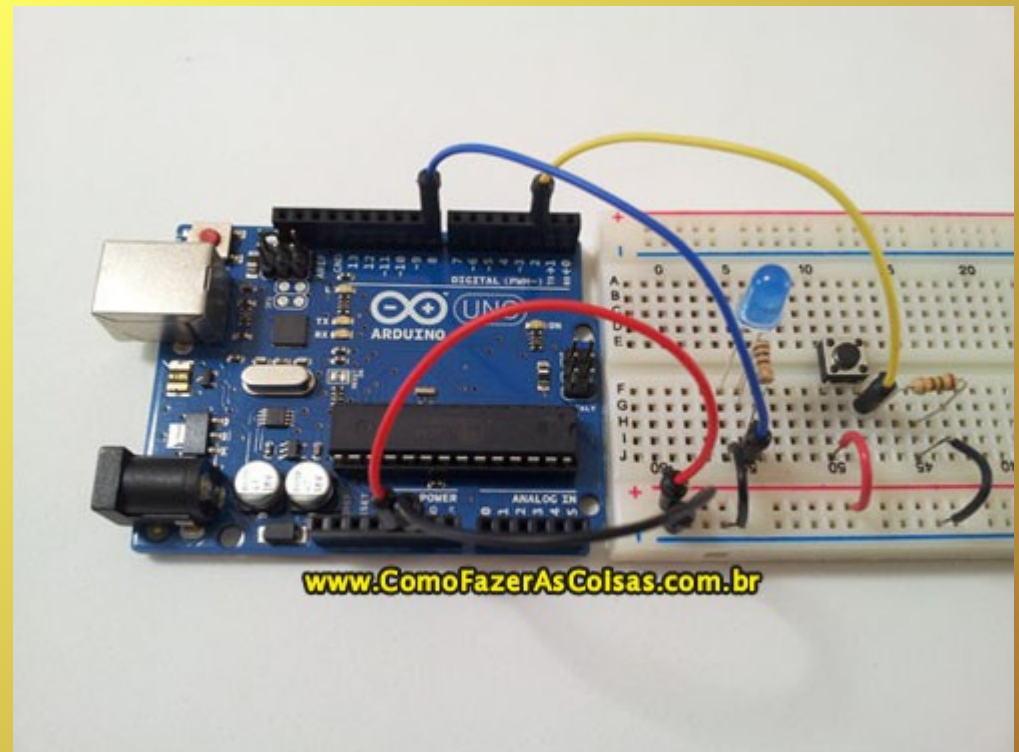
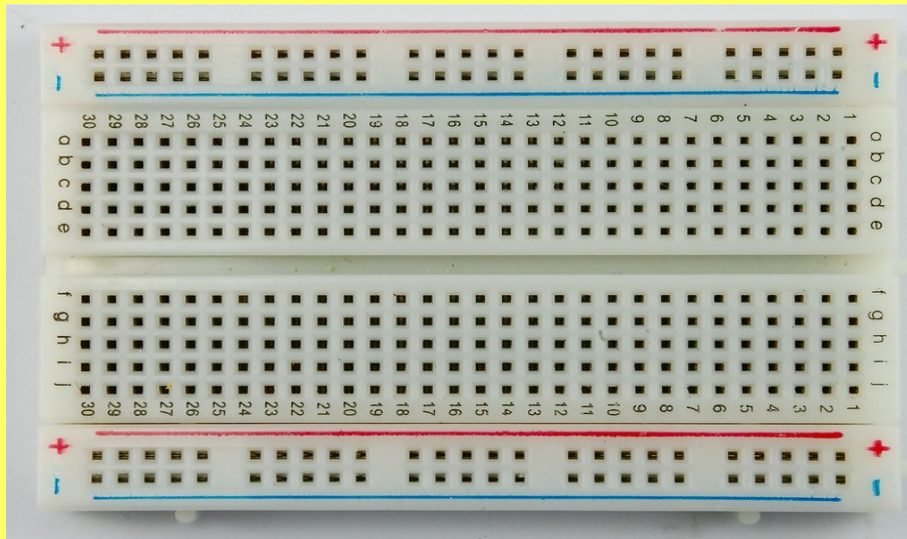
- Existem também os chamados Shields, que são placas que você encaixa no Arduino para expandir suas funcionalidades. A imagem a seguir mostra um *Arduino Ethernet Shield* encaixado no Arduino Mega 2560. Ao mesmo tempo que permite o acesso à uma rede ou até mesmo à internet, mantém os demais pinos disponíveis para utilização, assim você consegue, por exemplo, utilizar os pinos para receber dados de temperatura e umidade de um ambiente, e consultar esses dados de qualquer lugar do planeta:

Ethernet Shield

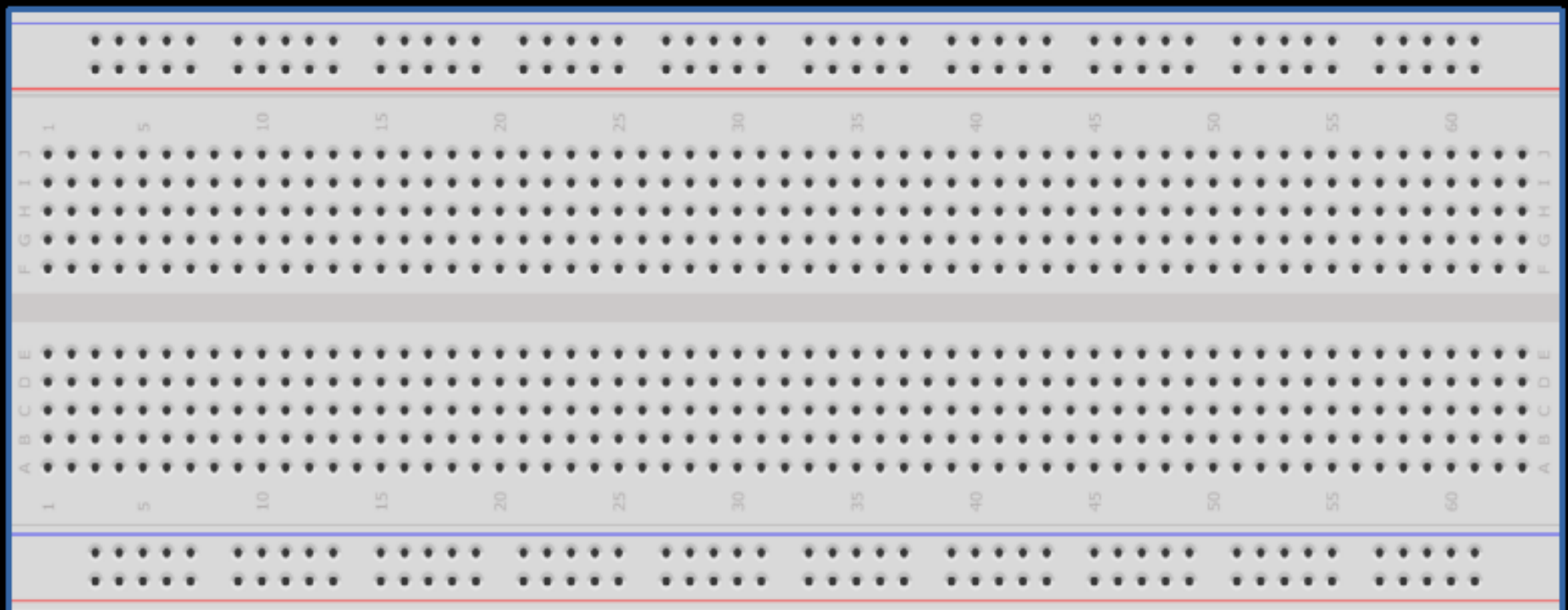


A protoboard

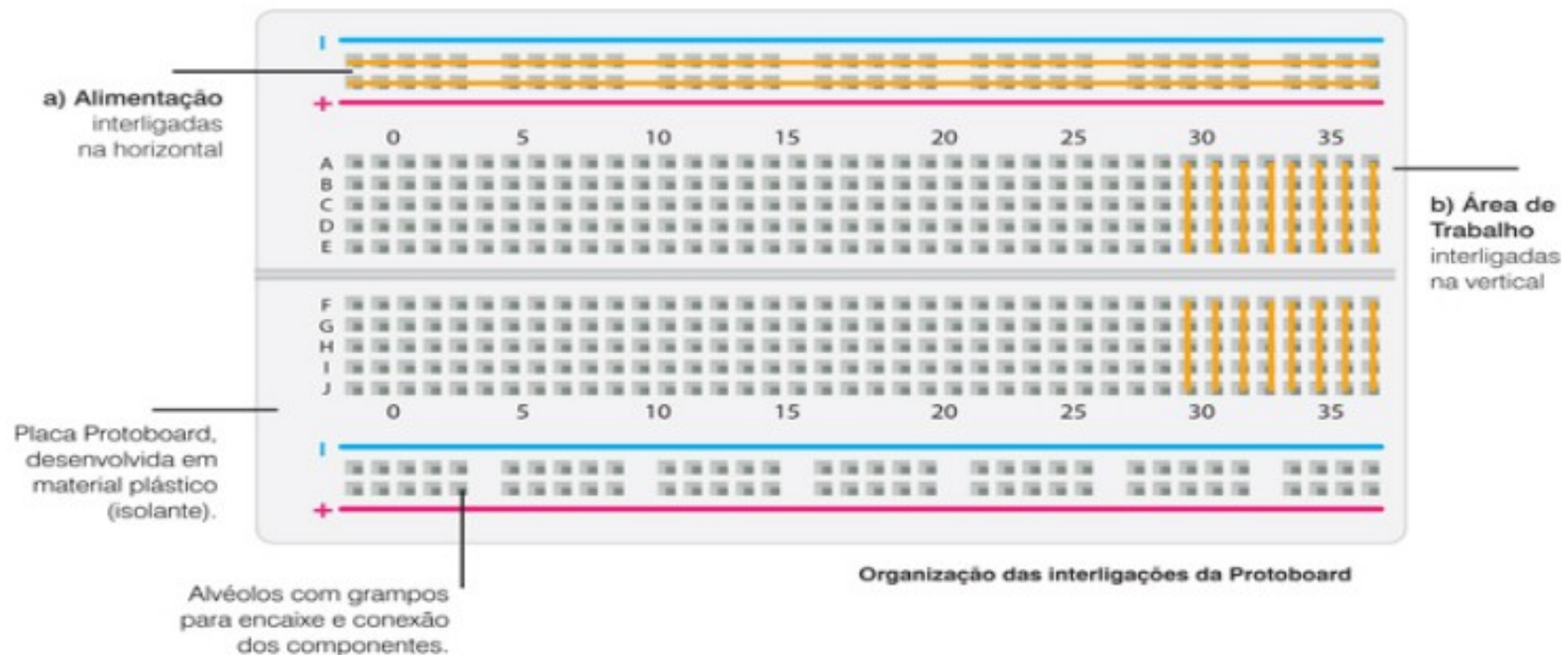
Para ligação dos componentes eletrônicos como led, resistor, sensor de umidade, sensor de temperatura entre vários outros, usa-se a *protoboard*.



A “**protoboard**” ou “**Matriz de contatos**” é utilizada para fazer montagens provisórias e/ou teste de projetos. É constituída por uma base plástica, contendo inúmeros orifícios destinados à inserção de terminais de componentes eletrônicos. Internamente existem ligações determinadas que interconectam os orifícios, permitindo a montagem de circuitos eletrônicos sem a utilização de solda.



As linhas são eletricamente independentes, isto é, não há conexão elétrica entre os furos de uma linha e de outra.



Portas Analógicas X Portas Digitais

Portas Analógicas (input)

Portas analógicas são aquelas que podem assumir valores com intervalo indefinido; Ex: 1 – 1,2 – 1,3 – 2 – 2,99... Como exemplo a temperatura, pressão e umidade são grandezas que variam dessa forma.

No nosso projeto a umidade do solo será uma grandeza de valor variável e não múltiplo.

O valor de retorno do sensor será interpretado pela porta analógica **A0**.



Portas Digitais(input/output)

Portas digitais são aquelas que podem assumir apenas dois níveis lógicos bem definidos, nível alto(HIGH) e baixo(LOW). Normalmente, o nível lógico alto é a tensão de alimentação do Arduino (5V) e nível lógico baixo é 0V (pino conectado ao GND).

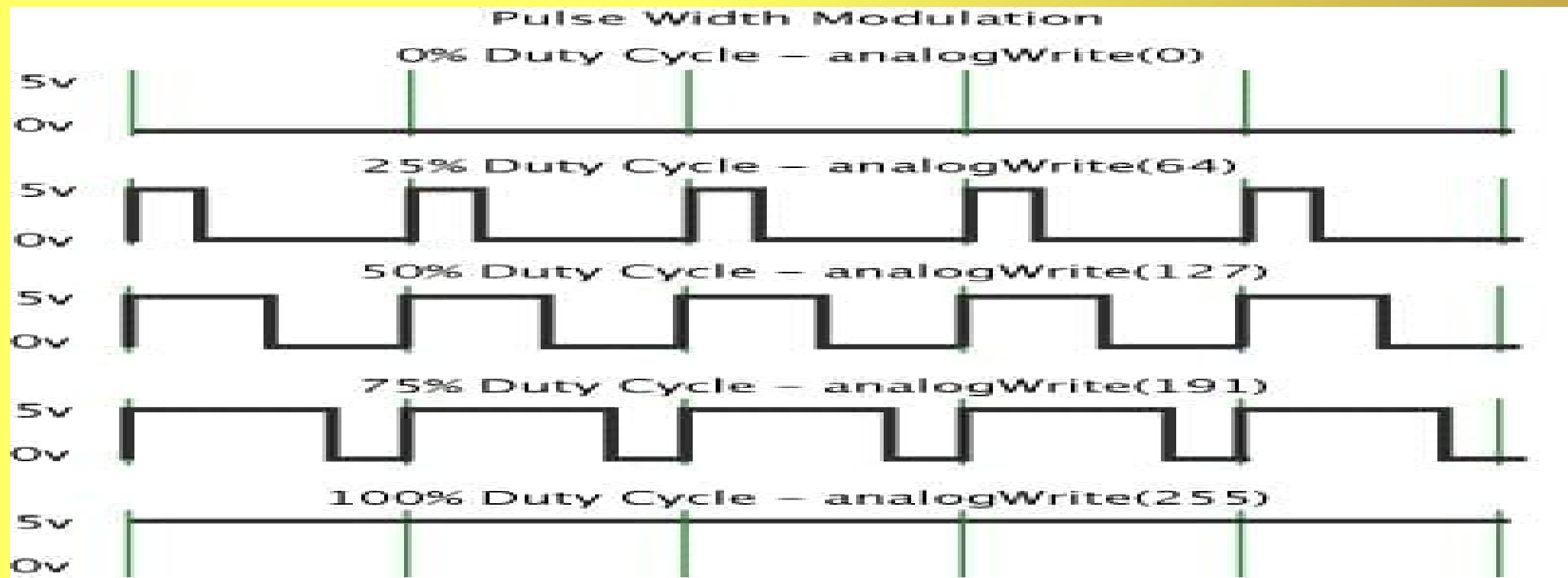
As portas digitais são comumente chamadas de I/O ports, que em inglês significa “portas de entrada e saída”. Esse nome vem do fato que uma porta digital poder assumir dois possíveis modos de operação, o modo de **entrada** e o modo de **saída**.

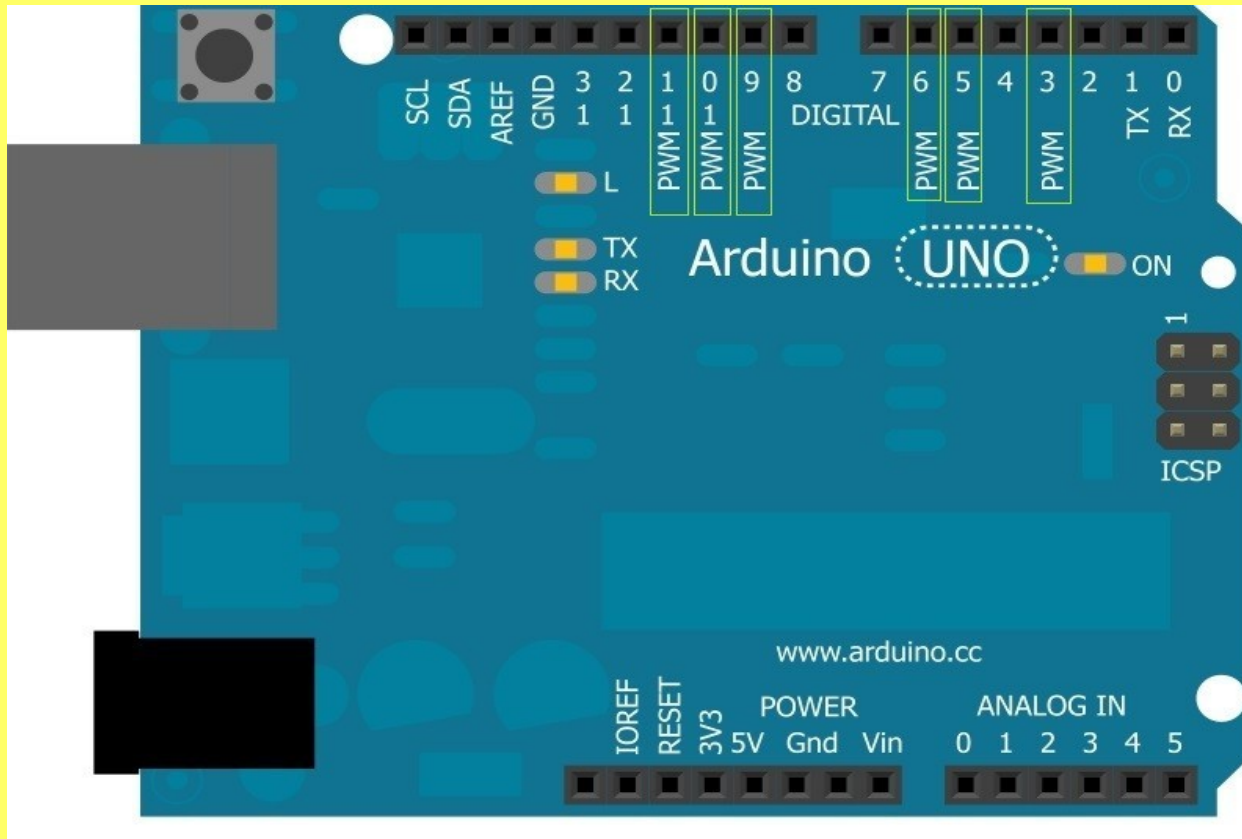
Portas PWM

PWM, do inglês Pulse Width Modulation, é uma técnica utilizada por sistemas digitais para variação do valor médio de uma forma de onda periódica.

A técnica consiste em manter a frequência de uma onda quadrada fixa e variar o tempo que o sinal fica em nível lógico alto. Esse tempo é chamado de duty cycle, ou seja, o ciclo ativo da forma de onda. No gráfico abaixo são exibidas algumas modulações PWM:

- Quando o duty cycle está em 0% o valor médio da saída encontra-se em 0 V e consequentemente para um duty cycle de 100% a saída assume seu valor máximo, que no caso é 5V.
- Para um duty cycle de 50% a saída assumirá 50% do valor da tensão, 2,5 V e assim sucessivamente para cada variação no duty cycle.





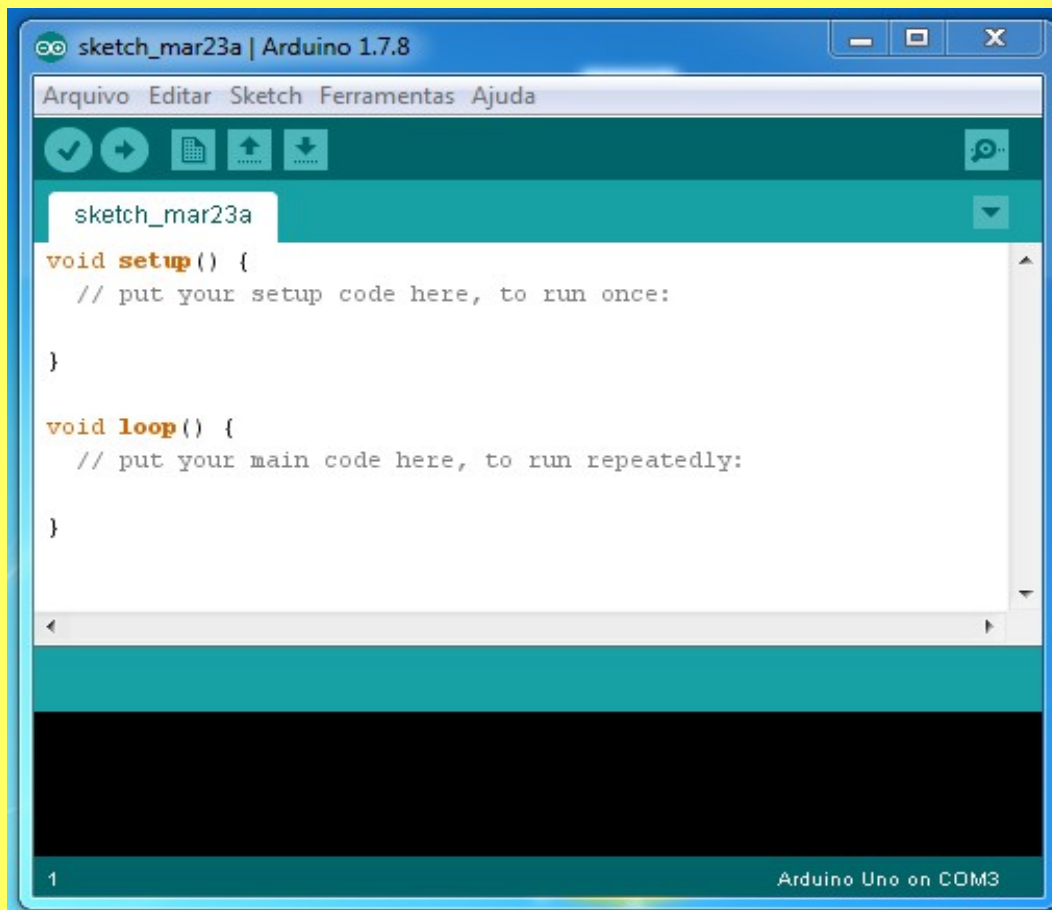
PWM pode ser usada para diversas aplicações, como por exemplo:

- controle de velocidade de motores;
- variação da luminosidade de leds;
- geração de sinais analógicos;
- geração de sinais de áudio.

Primeiro de tudo é importante explicar que a linguagem de programação (c) Arduino é sensível a maiúsculas: uma letra maiúscula não é o mesmo que uma letra minúscula.

O código a seguir representa o mínimo para que um programa possa ser compilado:

O "void setup ()" é normalmente usado para inicializar variáveis , modos de pinos , definição a velocidade de transmissão serial, etc. O software só vai executar uma vez.



```
sketch_mar23a | Arduino 1.7.8
Arquivo  Editar  Sketch  Ferramentas  Ajuda

sketch_mar23a
void setup() {
  // put your setup code here, to run once:
}

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

1
Arduino Uno on COM3
```

O "void loop ()" é a parte do código em que se faz um loop, ou seja, cria-se um ciclo de repetição para que as instruções dentro do mesmo sejam repetidas.



sketch_apr17a | Arduino 1.7.8



Arquivo Editar Sketch Ferramentas Ajuda



sketch_apr17a



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



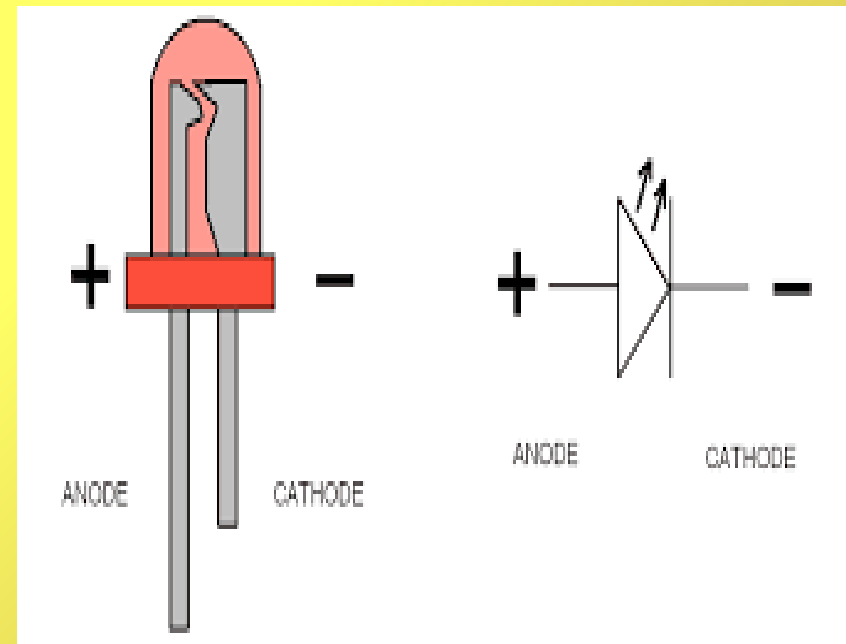
Desafio 'pisca pisca'

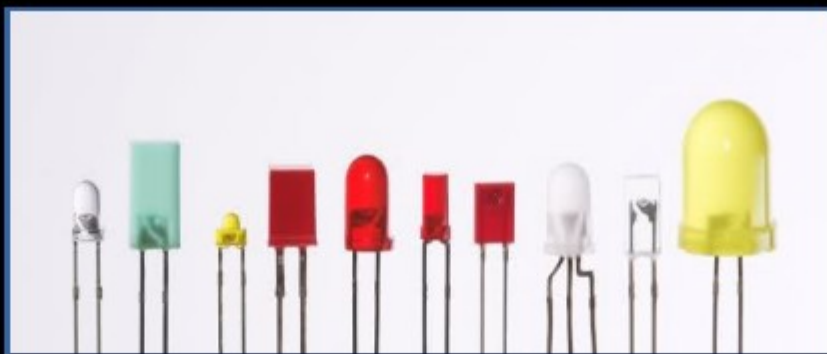
Este desafio consiste em ligar e desligar o Led interno da placa, simulando um pisca pisca.

```
1 void setup() {  
2   // put your setup code here, to run once:  
3   pinMode(LED_BUILTIN, OUTPUT);  
4 }  
5  
6 void loop() {  
7   // put your main code here, to run repeatedly:  
8   digitalWrite(LED_BUILTIN, HIGH);  
9 }
```

O comando PinMode configura o pino de saída para o Led interno;
O comando digitalWrite serve para ligar o Led;

Lâmpadas X LED





O "**diodo emissor de luz**" também é conhecido pela sigla em inglês **LED** (Light Emitting Diode). Sua funcionalidade básica é a emissão de luz em locais e instrumentos onde se torna mais conveniente a sua utilização no lugar de uma lâmpada.



Tensão dos Leds

Tensão: 3.1
Amp: 0.02



1.8
0.02



3.1
0.02

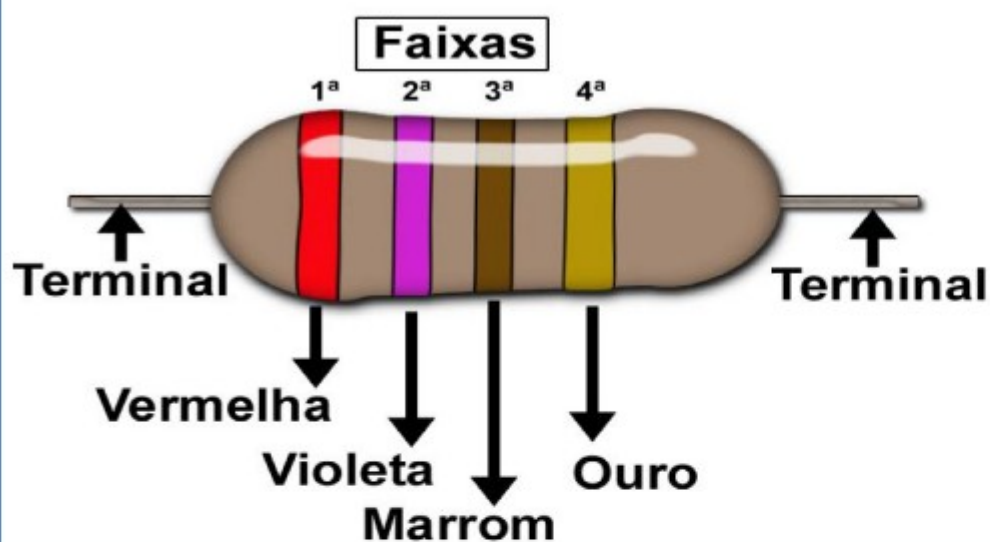


2.0
0.015



2.1
0.02





O valor da
resistência é
medido em OHM
e seu símbolo é o
Ômega Grego Ω

1 ohm ou 1Ω
1000 ohms = $1K\Omega$
1000 $K\Omega$ = $1 M\Omega$

Um **Resistor** (frequentemente chamado de resistência, que é na verdade a sua medida) é um dispositivo elétrico muito utilizado em eletrônica, ora com a finalidade de transformar energia elétrica em energia térmica por meio do efeito joule, ora com a finalidade de limitar a corrente elétrica em um circuito.

Cálculo de Resistores para Leds

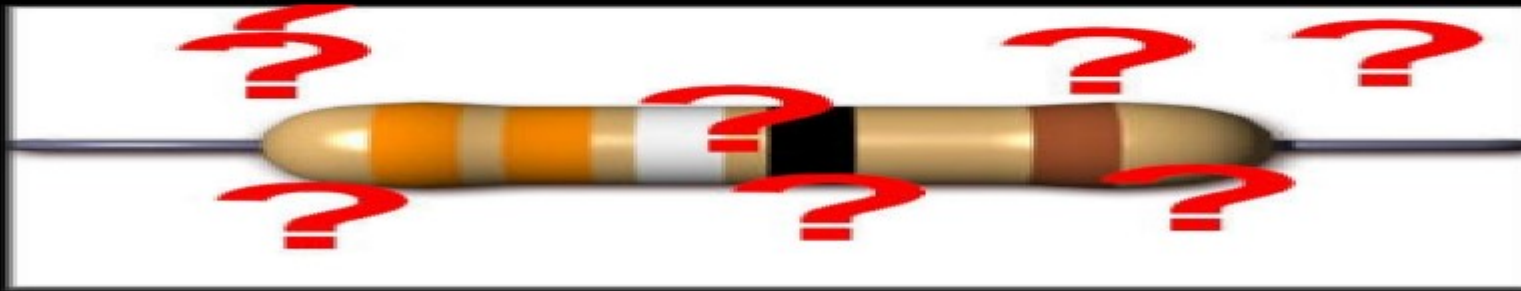
$$\text{Fórmula: } R = (V_F - V_L) / CML$$

R = Resistor necessário

VF = Voltagem fornecida pela fonte de energia

VL = Voltagem do Led

CML = Corrente máxima do Led



Exemplo: Um led que precisa de 2.1v e sua corrente máxima é de 20 milliampéres, qual resistor deveremos usar?

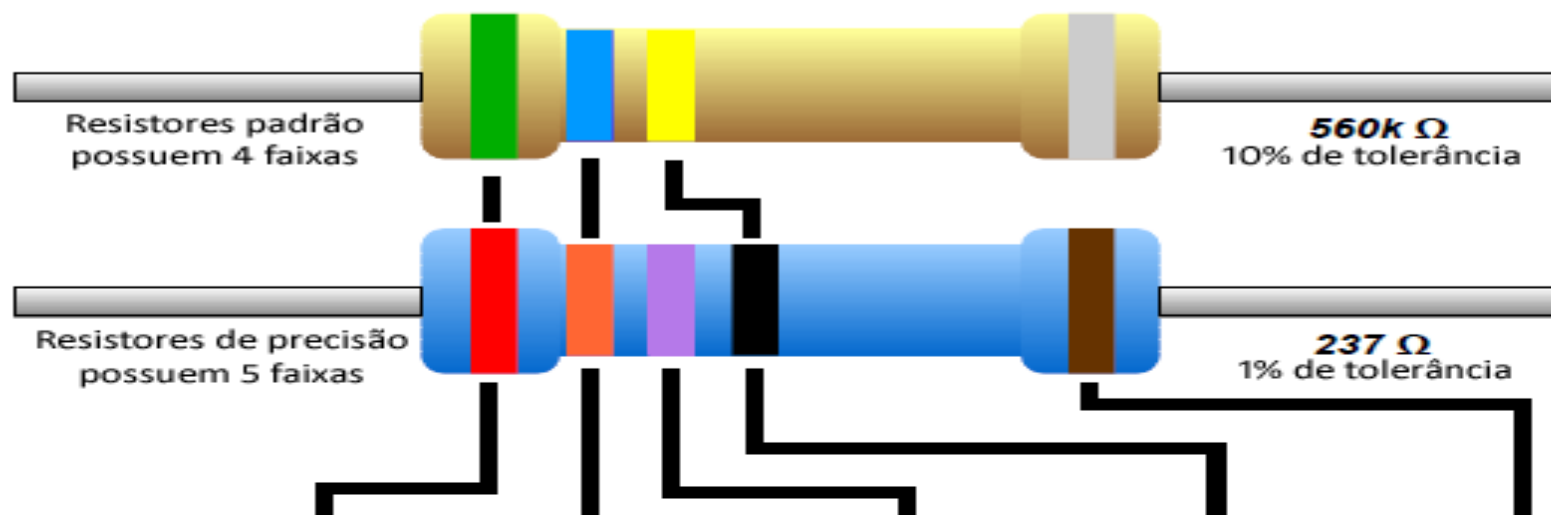
Tabela de tensão(v) para led's					
	Azul	Vermelho	Branco	Amarelo	Verde
Tensão(v)	3,1	1,8	3,1	2	2,1
Corrente(a)	0,020	0,020	0,020	0,015	0,020
Fonte	5	5	5	5	5
Tensão(v)	1,9	3,2	1,9	3	2,9
(R)Ohms	94	159	94	199	144

http://www.audioacustica.com.br/exemplos/Valores_Resistores/Calculadora_Grafica_Resistores_4-bandas.html

Tabela de resistores

Código de Cores

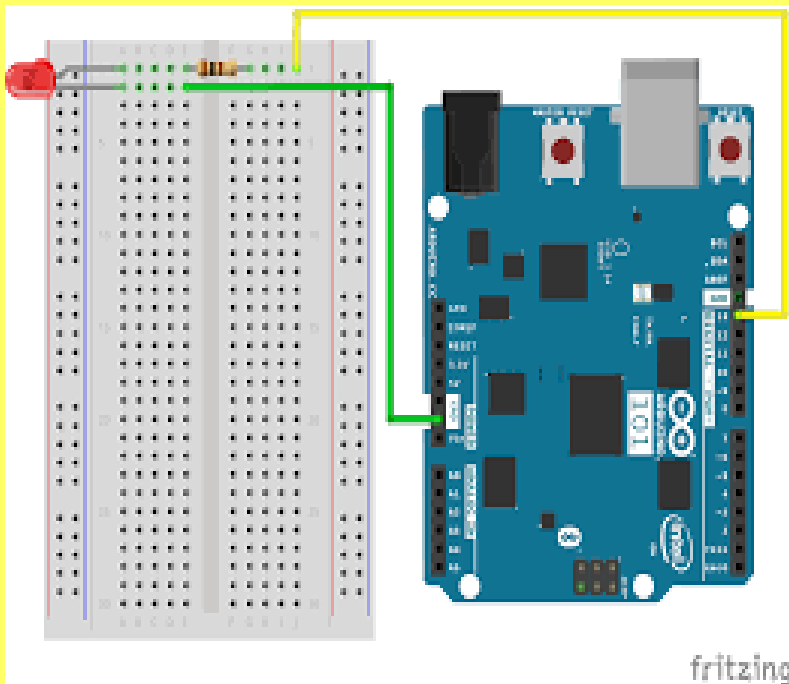
A extremidade com mais faixas deve apontar para a esquerda



Cor	1ª Faixa	2ª Faixa	3ª Faixa	Multiplicador	Tolerância
Preto	0	0	0	$\times 1 \Omega$	
Marrom	1	1	1	$\times 10 \Omega$	+/- 1%
Vermelho	2	2	2	$\times 100 \Omega$	+/- 2%
Laranja	3	3	3	$\times 1K \Omega$	
Amarelo	4	4	4	$\times 10K \Omega$	
Verde	5	5	5	$\times 100K \Omega$	+/- .5%
Azul	6	6	6	$\times 1M \Omega$	+/- .25%
Violeta	7	7	7	$\times 10M \Omega$	+/- .1%
Cinza	8	8	8		+/- .05%
Branco	9	9	9		
Dourado				$\times .1 \Omega$	+/- 5%
Prateado				$\times .01 \Omega$	+/- 10%

1º desafio: acendendo o LED

Nossa primeira atividade será acender um Led através do Arduino e protoboard. *Mãos à obra...*



Monte o circuito da seguinte forma:

- a) o pino (-) do Led no resistor e o outro pino do resistor no GND da placa Arduino;
- b) o pino (+) do Led deverá ser conectado à porta 5v da placa Arduino;
- c) Conecte o Arduino pela porta USB para alimentação;
O Led deverá acender.

2º desafio... 'blink'

- Através do circuito anterior, vamos montar um 'blink'(pisca).
- Para esta montagem, será necessário configurar uma porta digital para saída da tensão elétrica e juntamente com o código, determinar o tempo em que o led ficará ligado/desligado, em torno de 1s cada;

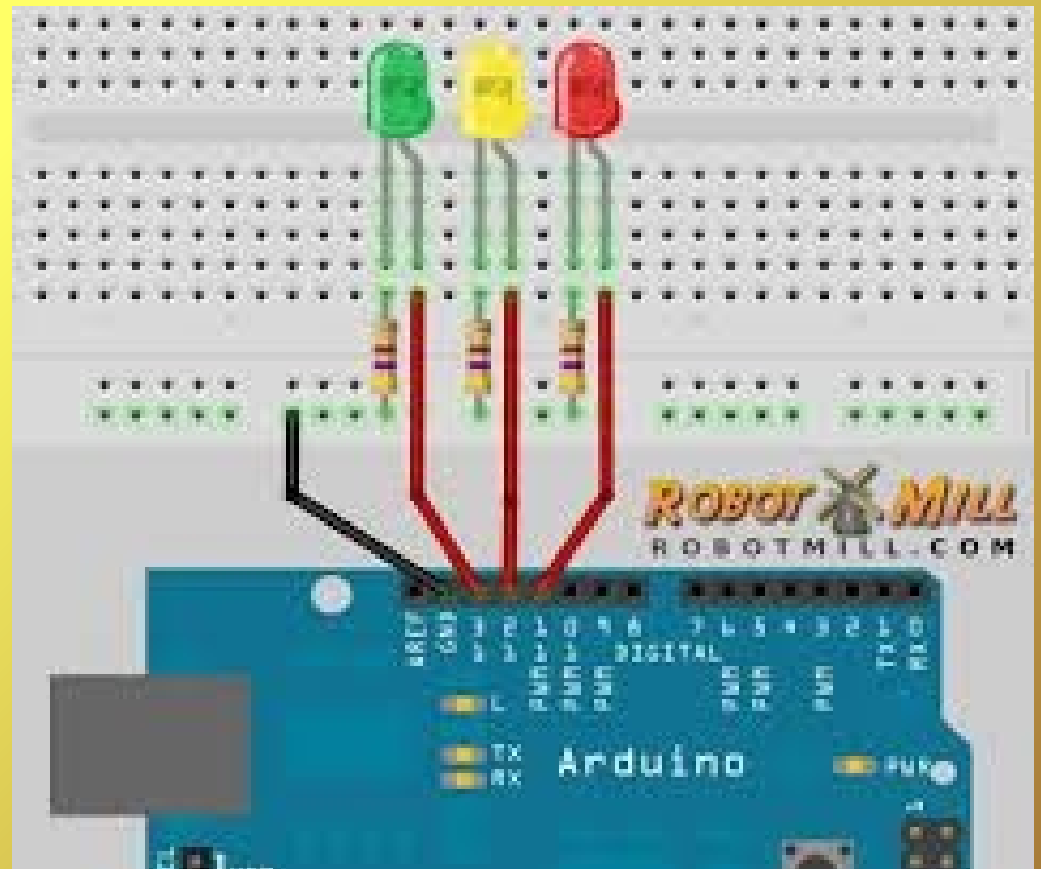
```
int ledVerde = 2; // define a variável na porta digital 2
int tempo=1000; // define o tempo para ligar e desligar

void setup() {
  pinMode(ledVerde ,OUTPUT); // define a porta (2) saída
}

void loop() {
  digitalWrite(ledVerde , HIGH); // envia o sinal (+)
  delay(tempo); // aguarda 1000 milisegundos
  digitalWrite(ledVerde , LOW); // envia o sinal (-)
  delay(tempo); // aguarda 1000 milisegundos
}
```


3º desafio

- Montar um 'blink' com três led's:
- Verde
- Amarelo
- Vermelho
- #partiuarduino



```
int ledGreen = 2; // define a variável na porta digital 2
int ledYellow = 4; // define a variável na porta digital 2
int ledRed = 6; // define a variável na porta digital 2
int tempo=1000; // define o tempo para ligar e desligar
```

```
void setup() {
  pinMode(ledGreen ,OUTPUT); // define a porta (2) saída
  pinMode(ledYellow ,OUTPUT); // define a porta (2) saída
  pinMode(ledRed ,OUTPUT); // define a porta (2) saída
}
```

```
void loop() {
  digitalWrite(ledGreen , HIGH); // envia o sinal (+)
  digitalWrite(ledYellow , LOW); // envia o sinal (-)
  digitalWrite(ledRed , LOW); // envia o sinal (-)
  delay(tempo); // aguarda 1000 milisegundos
```

```
  digitalWrite(ledGreen , LOW); // envia o sinal (-)
  digitalWrite(ledYellow , HIGH); // envia o sinal (+)
  digitalWrite(ledRed , LOW); // envia o sinal (-)
  delay(tempo); // aguarda 1000 milisegundos
```

```
  digitalWrite(ledGreen , LOW); // envia o sinal (-)
  digitalWrite(ledYellow , LOW); // envia o sinal (-)
  digitalWrite(ledRed , HIGH); // envia o sinal (+)
  delay(tempo); // aguarda 1000 milisegundos
```

```
}
```