

# Programação II

## Introdução ao Arduino

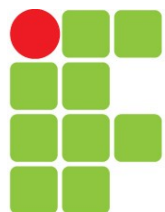
### *Aspectos gerais e prática*



Material produzido por:

Prof. Maurício Edgar Stivanello, DAMM/IFSC

Prof. Delcino Picinin Júnior

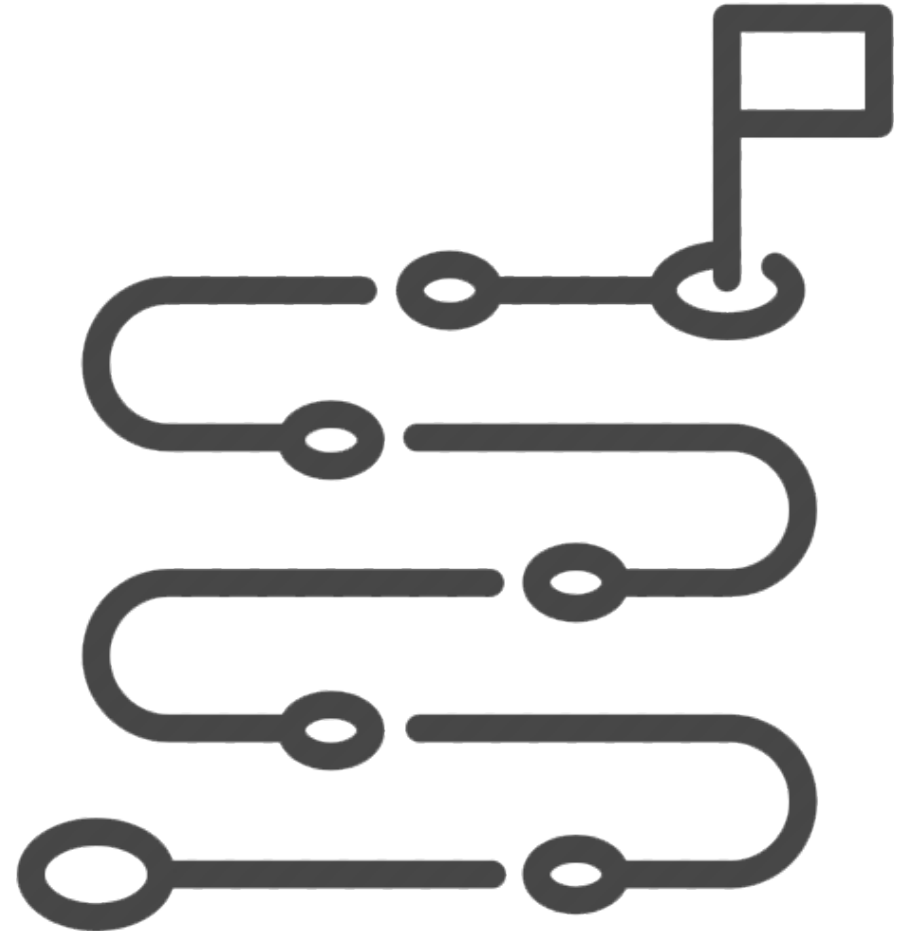


# Agenda

## **1) Introdução ao Arduino**

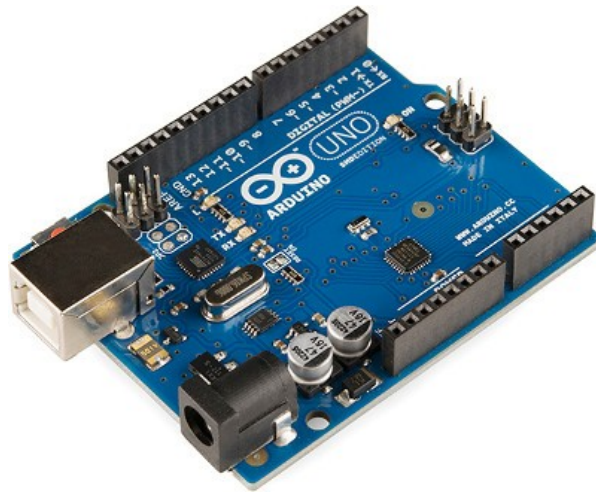
2) Exemplos no Tinkercad

3) Exercícios



# Arduino

O que é um Arduino?



Resposta curta:  
Isso é um Arduino...



# Arduino

O que é um Arduino?

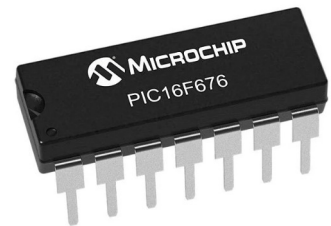
Para compreender melhor o que é um Arduino precisamos compreender o que é um **microcontrolador**



ATtiny 85



ATmega 328

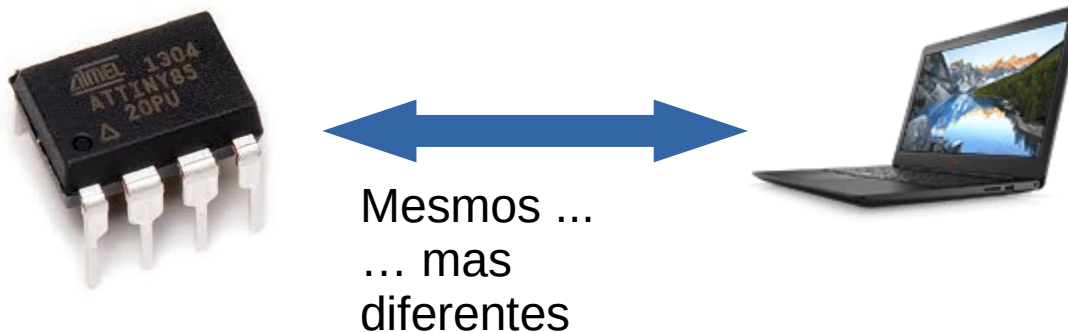


PIC16F676

# Arduino

Microcontroladores são como computadores em miniatura:

- Possuem um processador, Memória RAM, Armazenamento, etc
- Podem ser programados como um computador



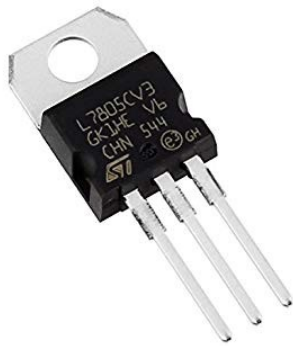
Diferente de computadores os microcontroladores...

- Disponibilizam conexões elétricas diretas a dispositivos externos
- Não rodam Battlefield ou Minecraft ...

# Arduino

Microcontroladores podem ser trabalhosos de se utilizar:

- Requerem componentes externos para funcionarem: reguladores de tensão, cristal oscilador, capacitores, etc.



Regulador de  
tensão



Cristal oscilador

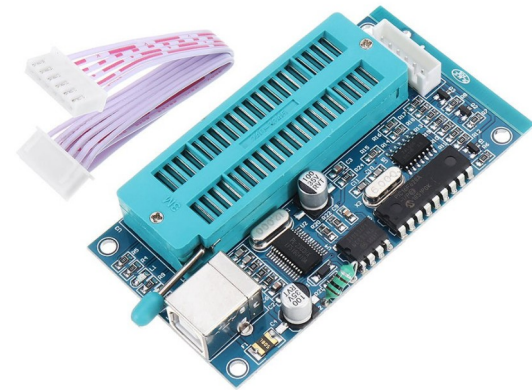
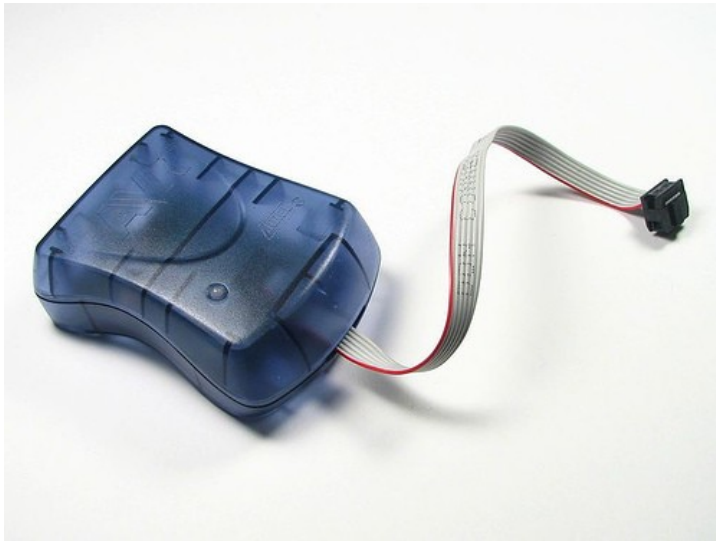


Capacitor

# Arduino

Microcontroladores podem ser trabalhosos de se utilizar:

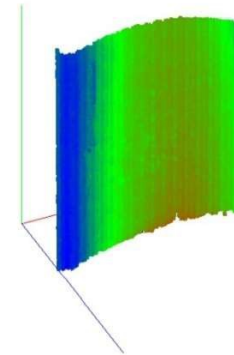
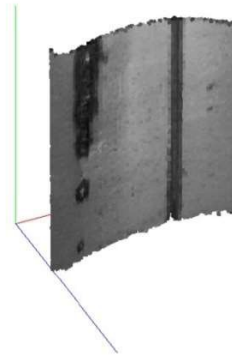
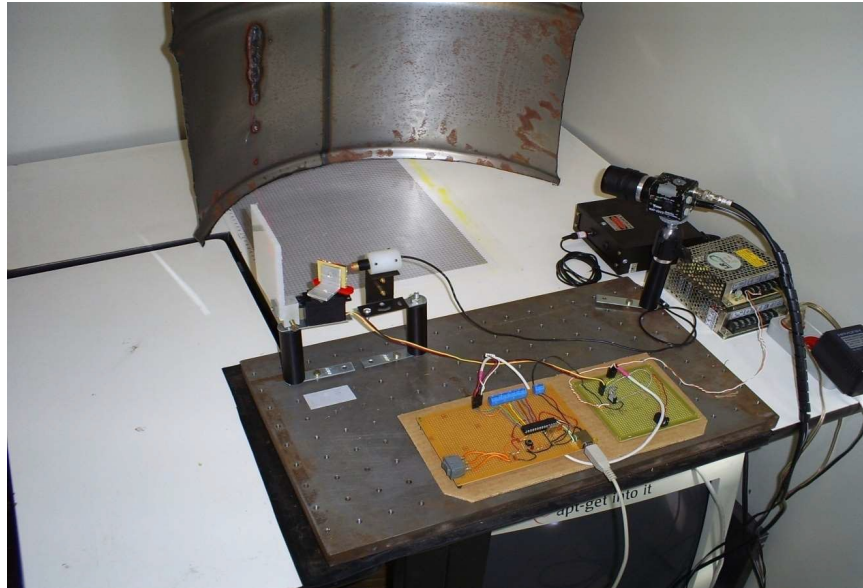
- Requerem dispositivos específicos para programar



Programadores

# Arduino

- Exemplo de projeto sem kit de desenvolvimento



Scanner laser



# Arduino

Então o que é um Arduino?

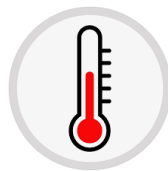
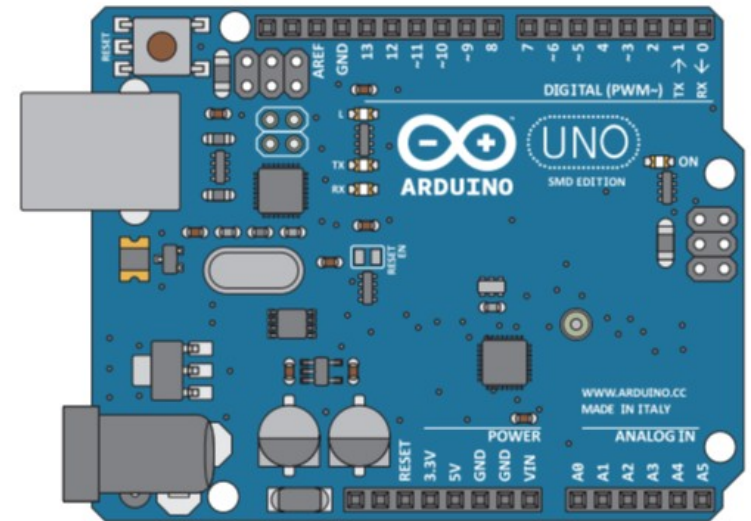
O Arduino combina um microcontrolador com todos os outros componentes necessários para seu funcionamento em um único dispositivo



# Arduino

Então o que é um Arduino?

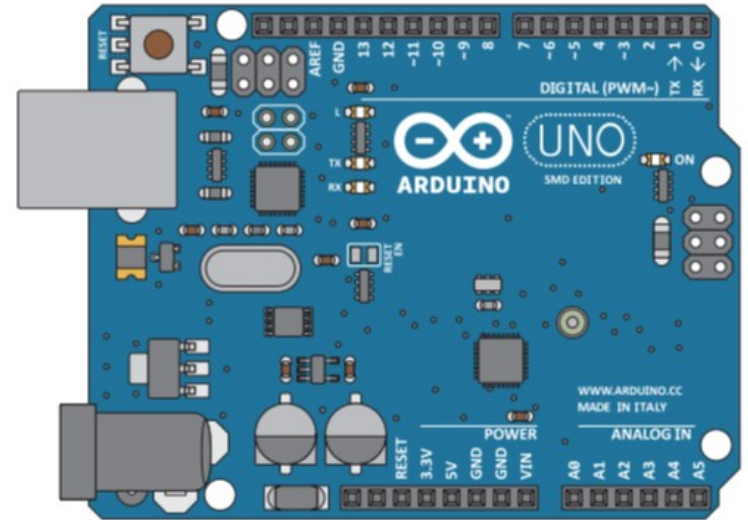
- Uma plataforma aberta e livre que simplifica a criação de dispositivos eletrônicos capazes de realizar medições (sensores) e comandar ações (atuadores)
- Um computador para o mundo físico
  - Capaz de ler entradas: temperatura através de um sensor, pressionamento de um botão
  - Gerar saídas: ligar um led, ativar um motor
- Utiliza informação de entradas para gerar saídas



# Arduino

## Então o que é um Arduino?

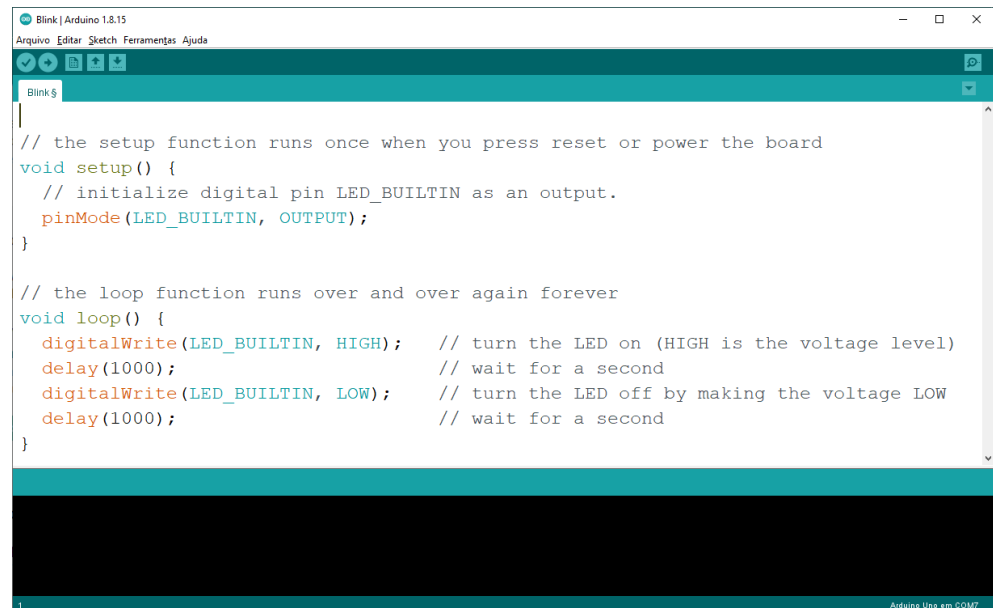
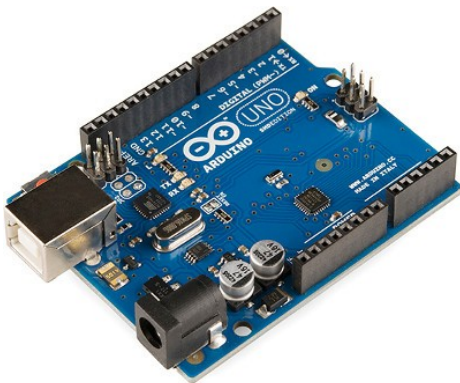
- Criação
  - Time liderado por Massimo Banzi
  - 2005
  - Instituto de design na cidade de Ivrea (Itália)
- Características
  - Aberto
  - Multi-plataforma
  - Ambiente de programação simples
  - Software livre e expansível (Bibliotecas)
  - Hardware livre (Projetos livres que viabilizam muitas placas de terceiros)



# Arduino

## Então o que é um Arduino?

- Plataforma do Arduino
  - Placa (Microcontrolador Atmel ATMEGA 328)
  - Linguagem de programação Arduino (Similar ao C/C++)
  - Ambiente de desenvolvimento (IDE)



```
Blink | Arduino 1.8.15
Arquivo  Editar  Sketch  Ferramentas  Ajuda

Blink.$

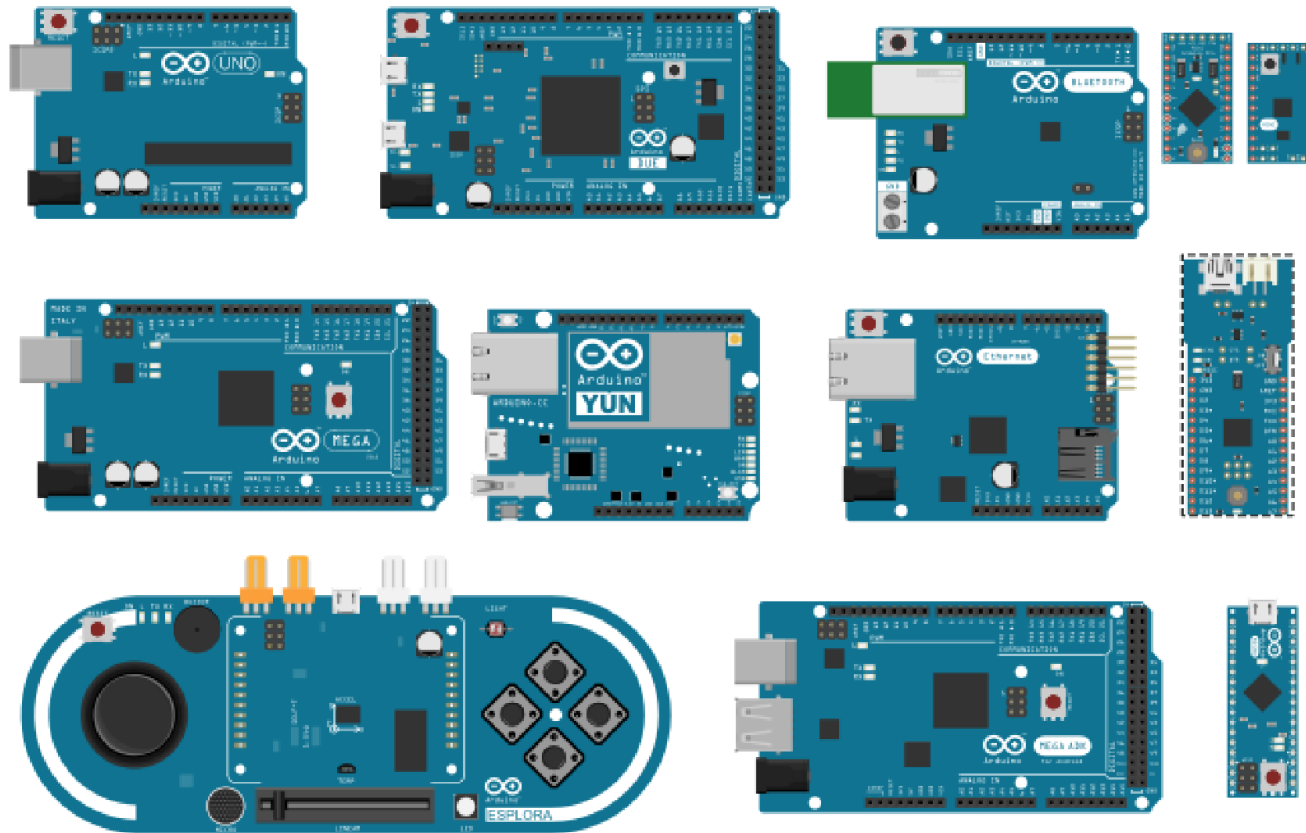
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

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

# Arduino

## Hardware

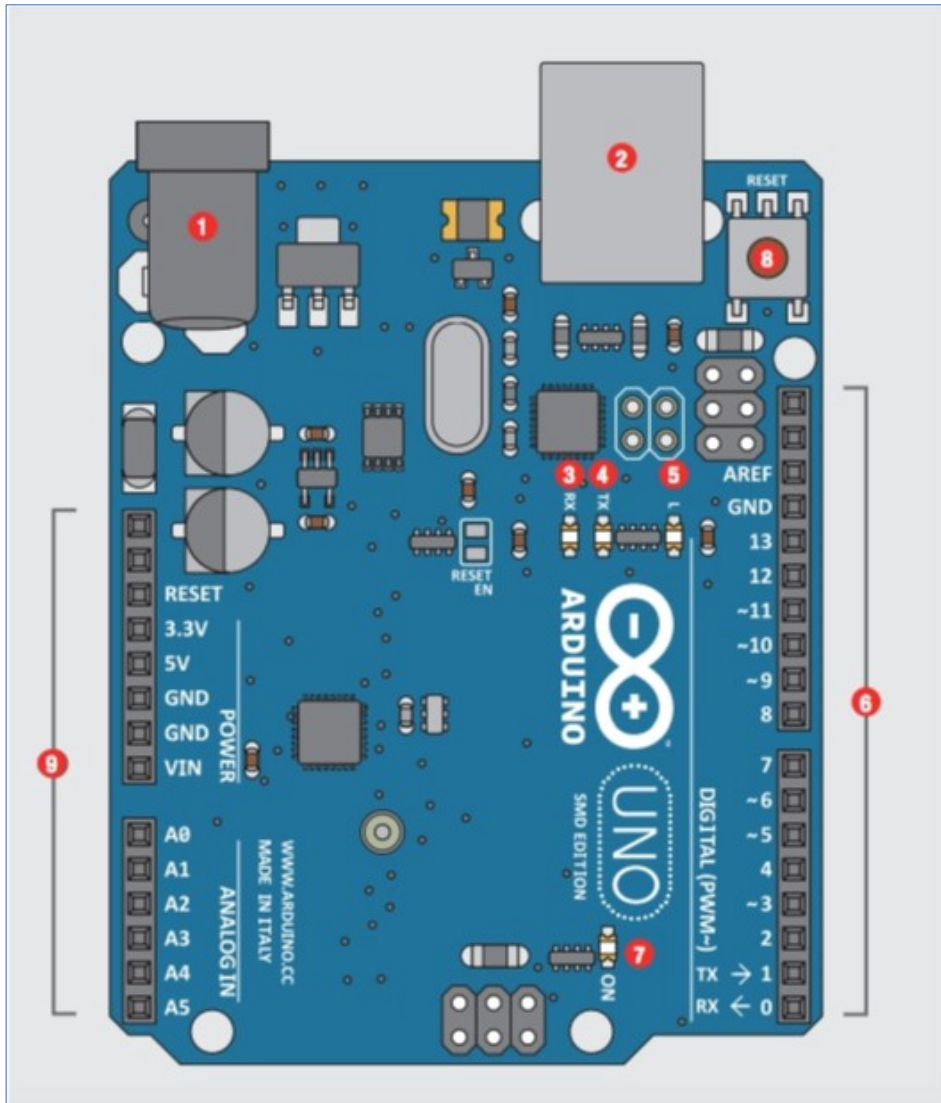
Diversos modelos: UNO, Leonardo, LilyPad, MEGA, Mini, etc





# Arduino

## Hardware: Visão geral do Arduino UNO



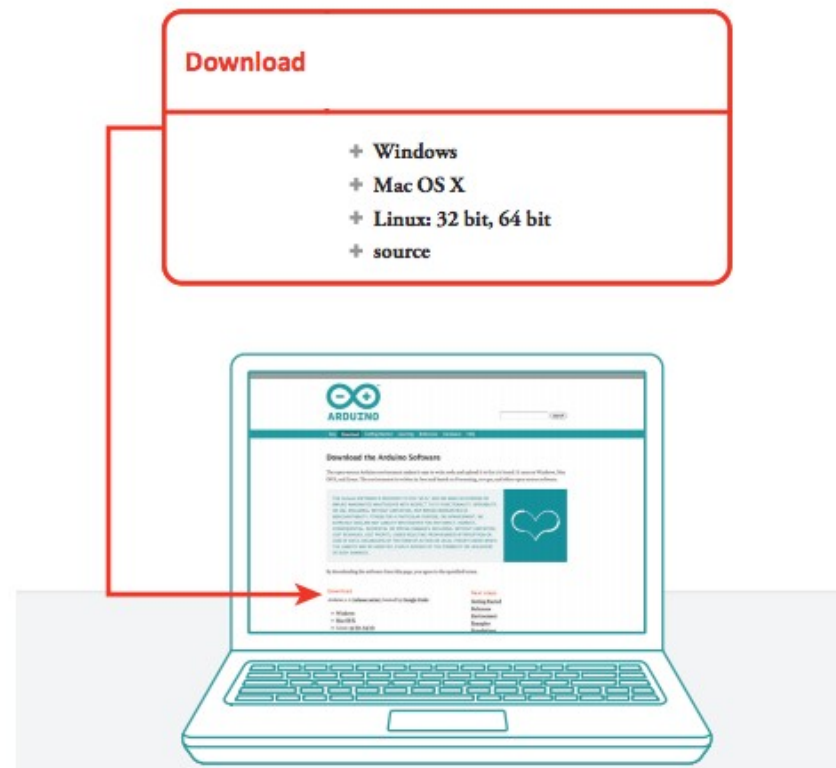
- 1) Entrada de Alimentação: 9V ou 12V
- 2) Porta USB: Alimentação e comunicação com o PC
- 3) LED (RX): Indica recebimento de dados
- 4) LED (TX): Indica transmissão de dados
- 5) LED (Pino 13): Propósito geral
- 6) Pinos: Pinos digitais, alimentação e terra
- 7) LED (ON): Indicação de ligado
- 8) Botão de reinicialização: Reinicia o Arduino
- 9) Pinos: Pinos analógicos, Alimentação, Terra, Reinicialização

# Arduino

Utilizando o Ambiente de Desenvolvimento

Baixe o instalador e instale o ambiente no PC:

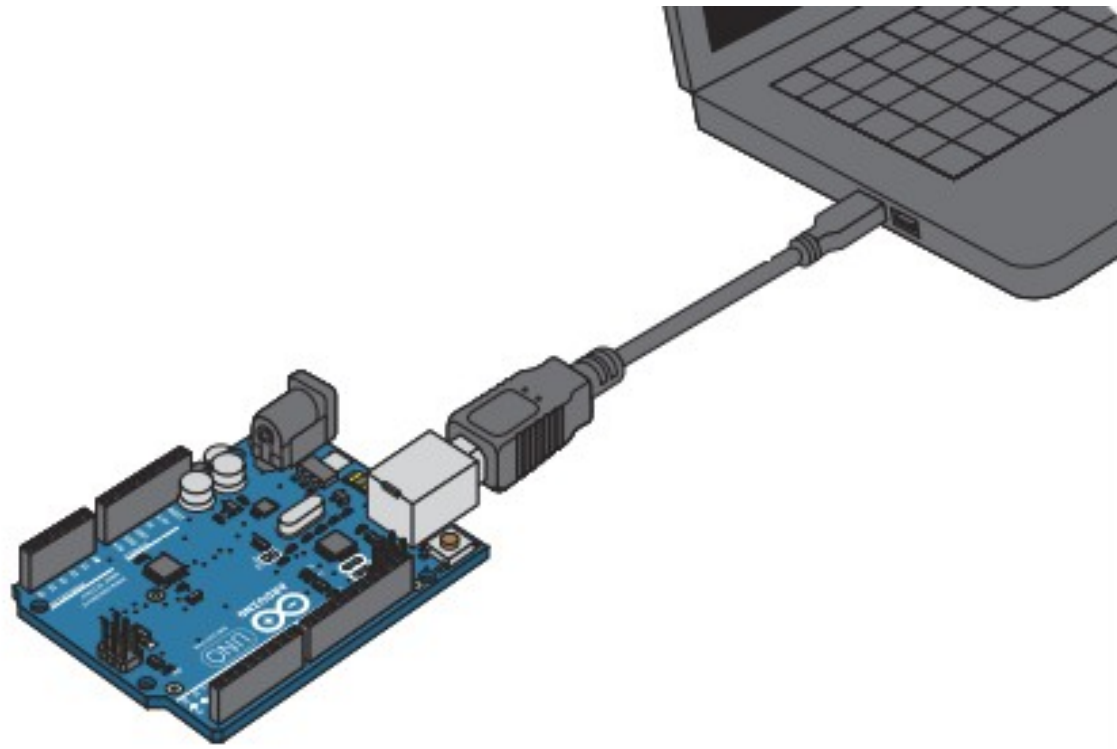
[www.arduino.cc/en/main/software](http://www.arduino.cc/en/main/software)



# Arduino

## Utilizando o Ambiente de Desenvolvimento

Conecte o Arduino ao PC fazendo uso do cabo USB

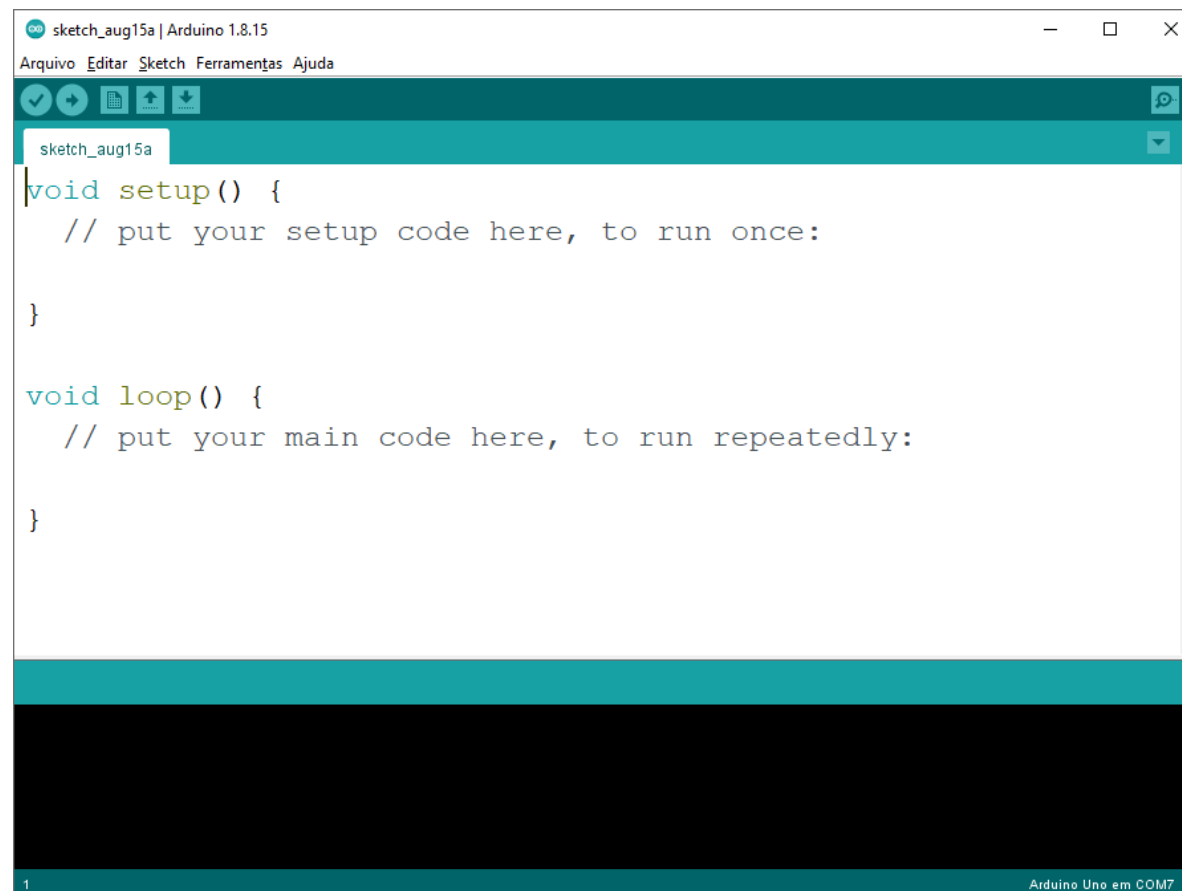




# Arduino

## Utilizando o Ambiente de Desenvolvimento

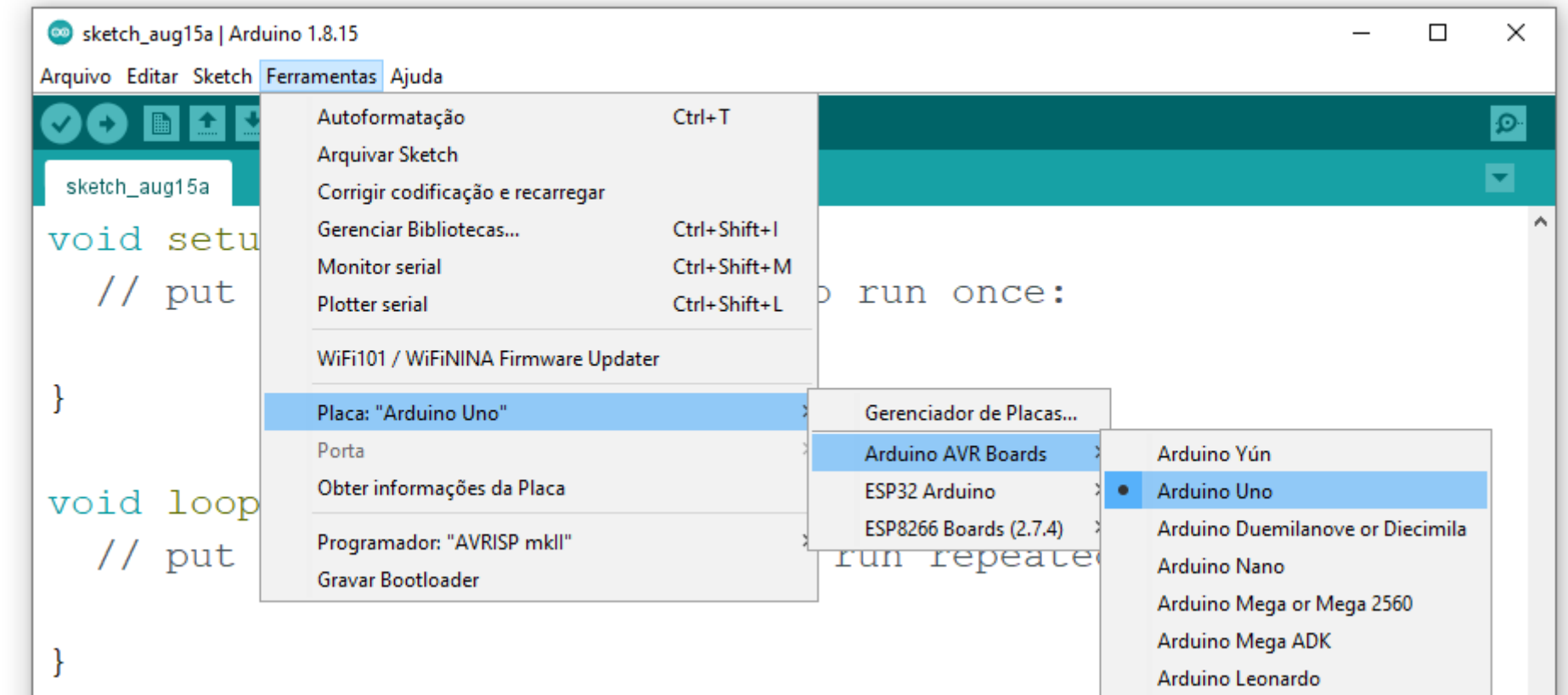
Execute o ambiente de desenvolvimento



# Arduino

## Utilizando o Ambiente de Desenvolvimento

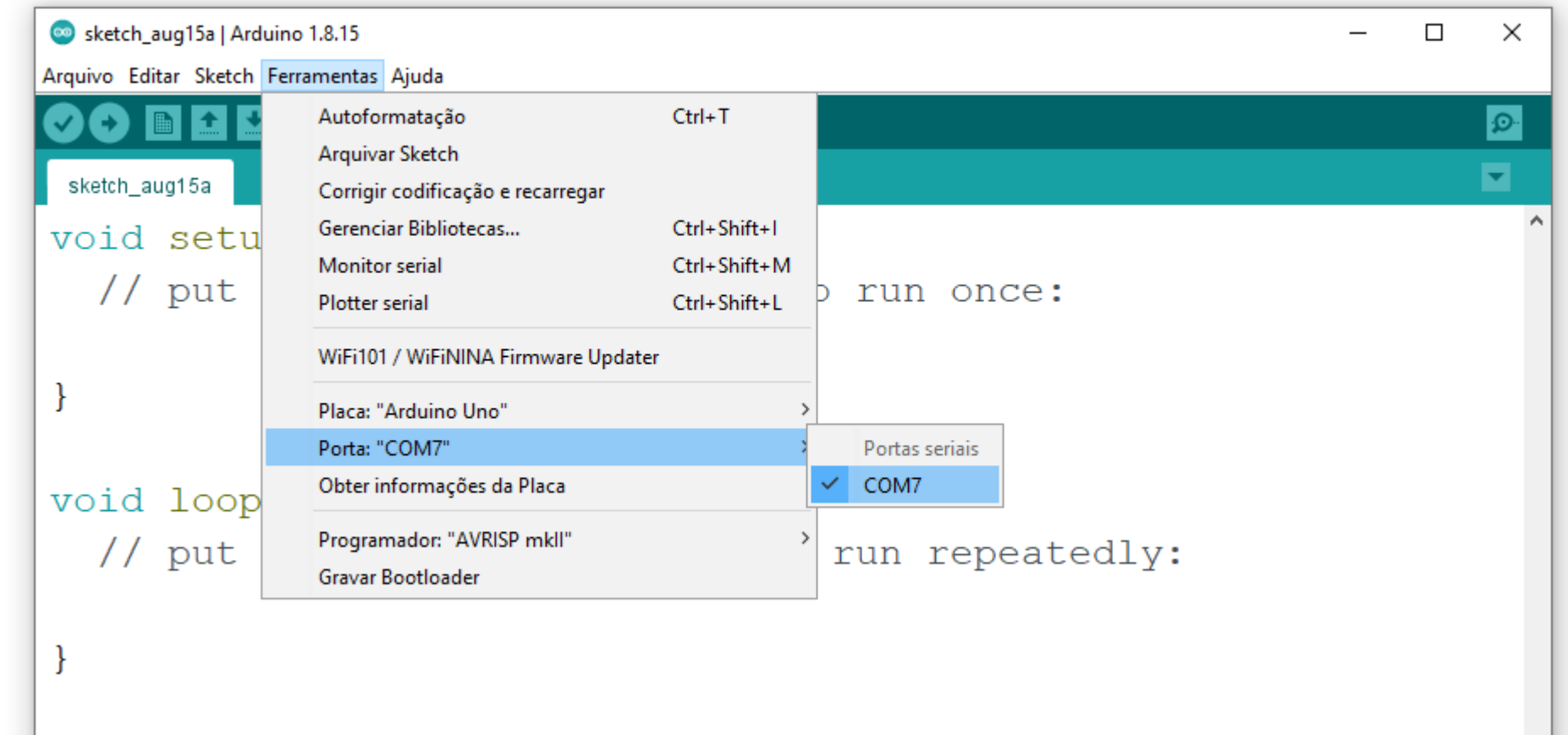
Escolha o modelo da placa utilizada através do menu ...



# Arduino

## Utilizando o Ambiente de Desenvolvimento

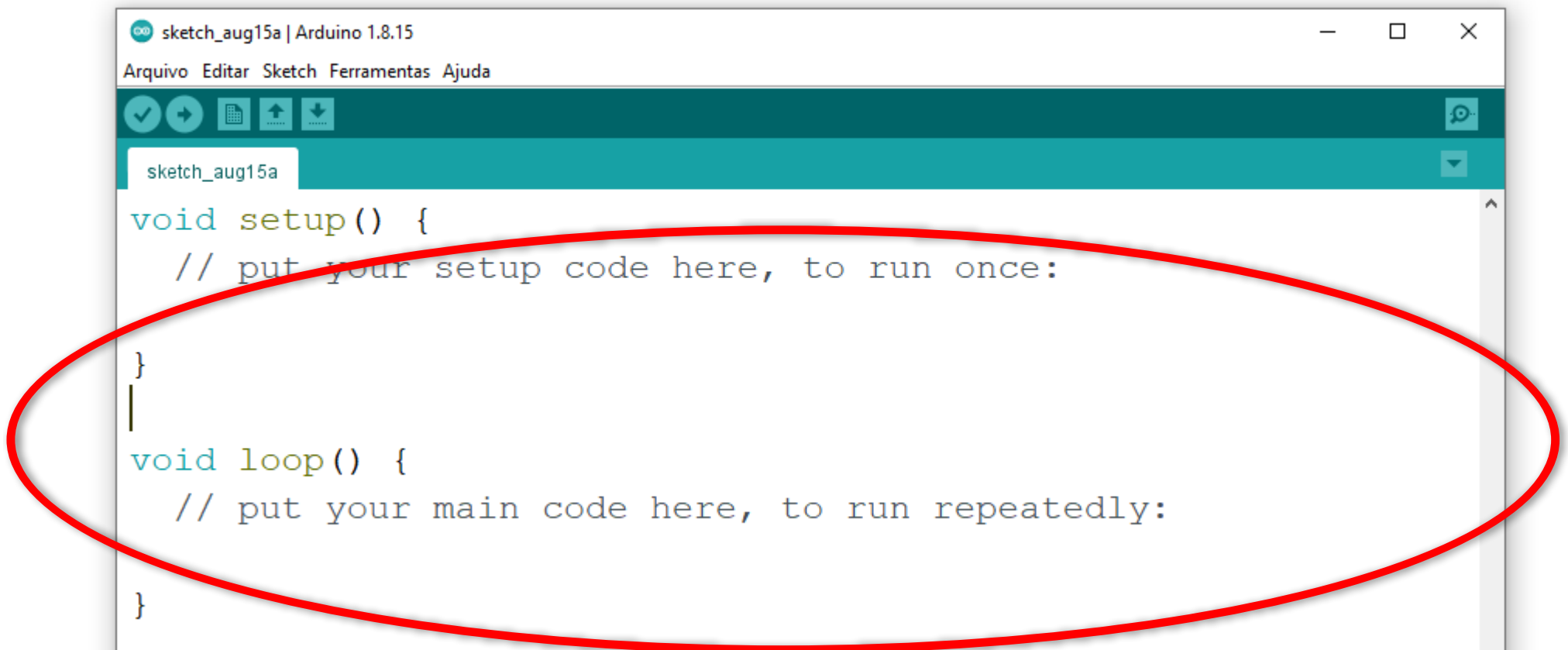
Escolha a porta de comunicação através do menu ...



# Arduino

## Utilizando o Ambiente de Desenvolvimento

Crie o seu código (Sketch) ...



```
sketch_aug15a | Arduino 1.8.15
Arquivo Editar Sketch Ferramentas Ajuda

sketch_aug15a

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

}

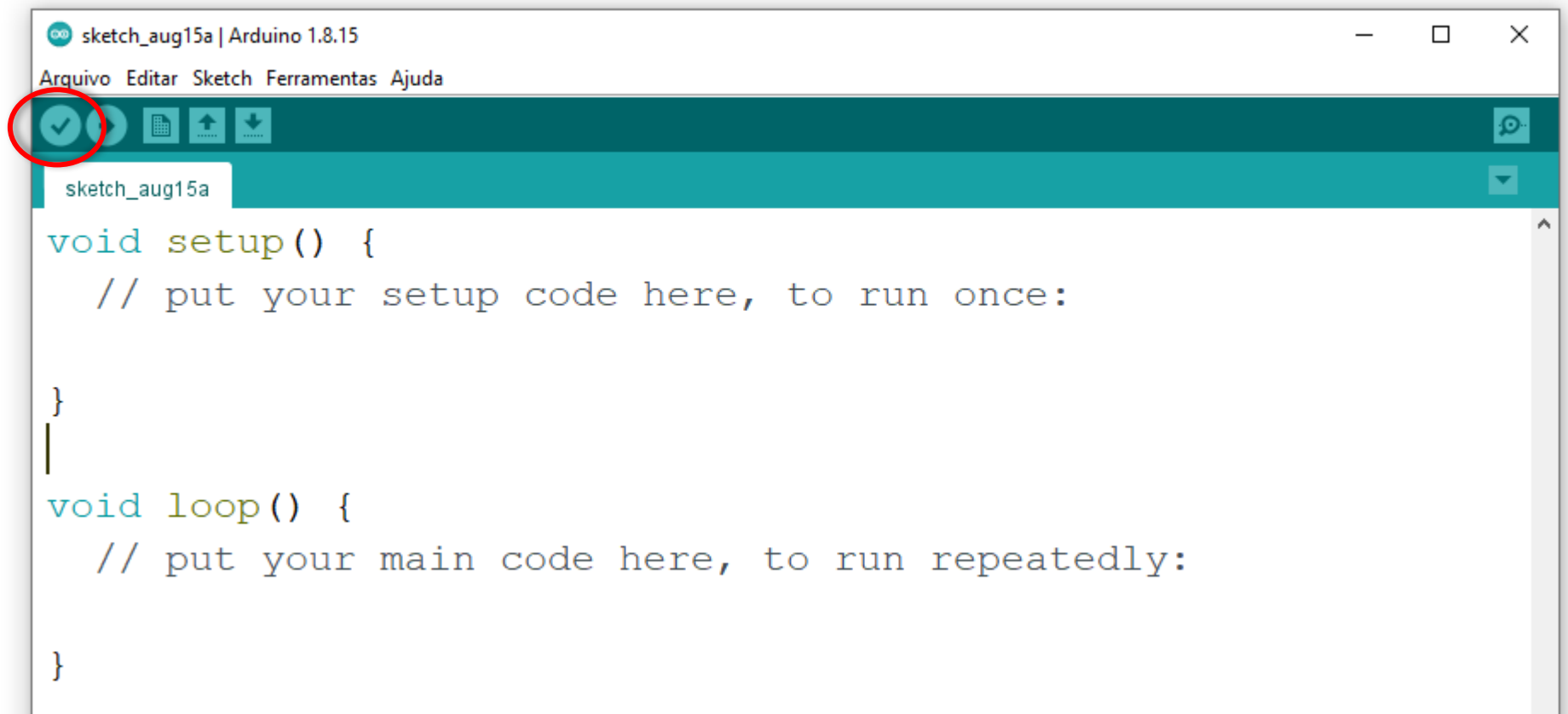
void loop() {
  // put your main code here, to run repeatedly:

}
```

# Arduino

## Utilizando o Ambiente de Desenvolvimento

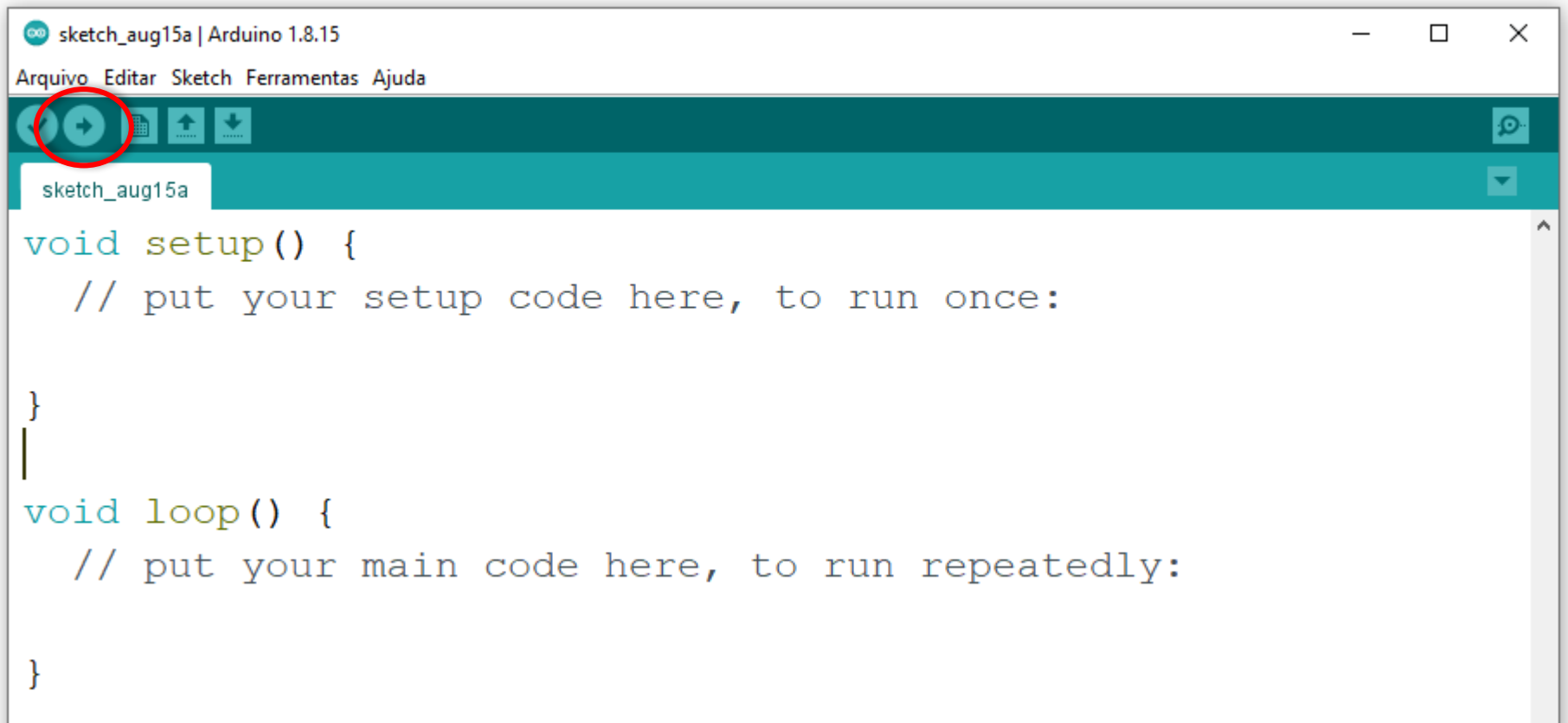
Verifique o seu código (Eventuais problemas serão descritos no painel inferior) ...



# Arduino

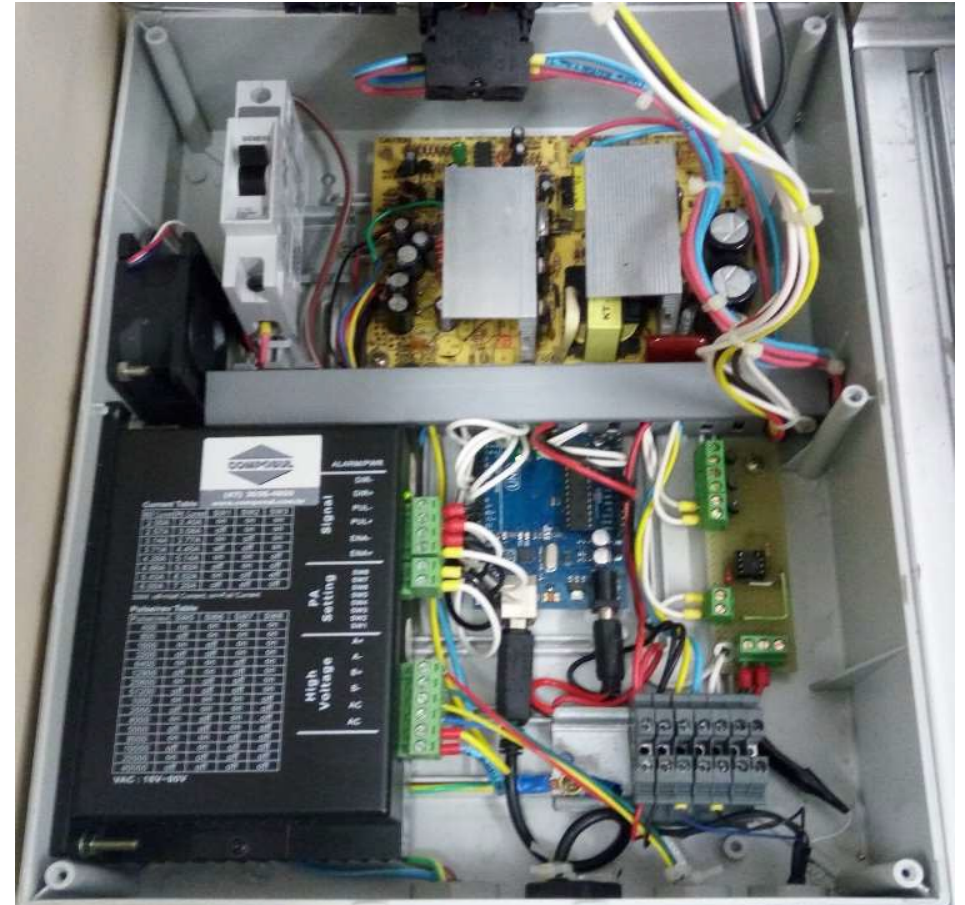
## Utilizando o Ambiente de Desenvolvimento

Envie o seu código ao Arduino ...



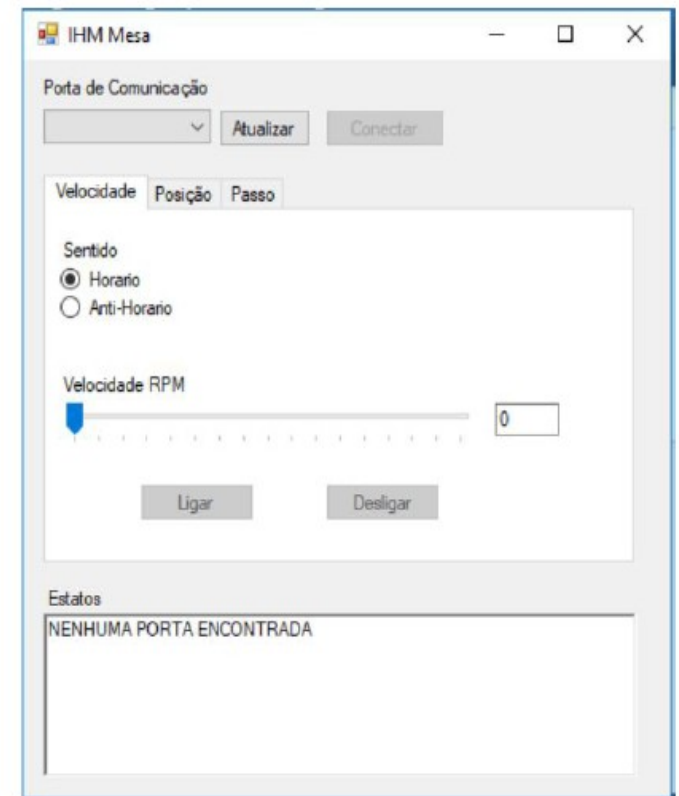
# Arduino

Exemplos de projetos: esteira de movimentação



# Arduino

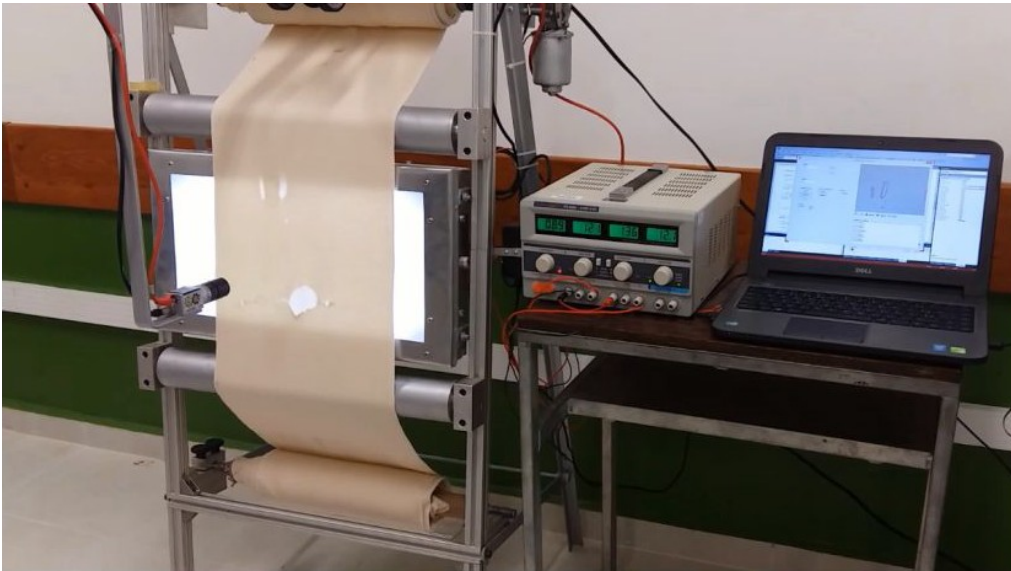
Exemplos de projetos: bancada giratória





# Arduino

Exemplos de projetos: rebobinadeira e robô



# Programando o Arduino

## Os blocos principais

- Função *setup*

Esta função é executada somente quando o Arduino é ligado ou reinicializado. É utilizada para inicializar variáveis, definir os modos de operação dos pinos, etc.

- Função *loop*

Esta função é executada continuamente até que o Arduino seja desligado. A lógica principal do código é escrita neste local.

```
void setup() {  
    // *** código de inicialização (executa 1 vez) ***  
    // inicializações do ambiente / definição do modo dos pinos  
}  
void loop() {  
    // *** código com lógica principal (executa repetidamente) ***  
  
    // Leituras de valores de sensores / cálculos / acionamentos  
}
```

# Arduino

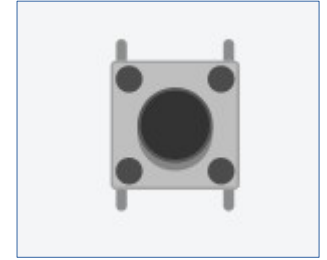
## Função **pinMode**

- Descrição:  
Configura o pino especificado para funcionar como uma entrada ou saída.
- Sintaxe  
`pinMode(pino, modo)`
- Parâmetros  
pino: o número do pino desejado  
modo: o modo do pino. Este pode ser INPUT, OUTPUT

*`pinMode(13, OUTPUT); // define o pino 13 como sendo de saída`*

*`pinMode(13, INPUT); // define o pino 13 como sendo de entrada`*

# Arduino



Ex: estado de um botão

## Função **digitalRead**

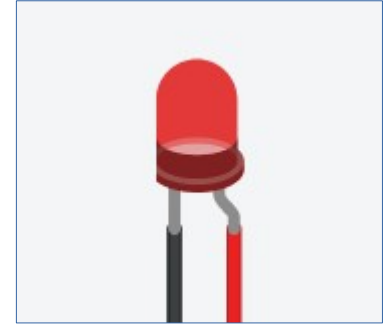
- Descrição:  
Lê o valor de um pino digital especificado, que pode ser HIGH (5V) ou LOW (0V).
- Sintaxe  
`digitalRead(pino)`
- Parâmetros  
pino: o número do pino digital do Arduino que você quiser verificar
- Retorna  
HIGH ou LOW

```
int estadoBotao = digitalRead(2); // lê e armazena o estado do pino 2  
  
if (digitalRead(2) == HIGH) // testa o estado do pino 2  
{  
    ...
```

# Arduino

## Função **digitalWrite**

- Descrição:  
Aciona um valor HIGH (5V) ou LOW (0V) em um pino digital.
- Sintaxe  
`digitalWrite(pino, valor)`
- Parâmetros  
pino: o número do pino do Arduino  
valor: HIGH ou LOW



Ex: acionamento de um LED

```
digitalWrite(13, HIGH); // ativa o pino digital 13
```

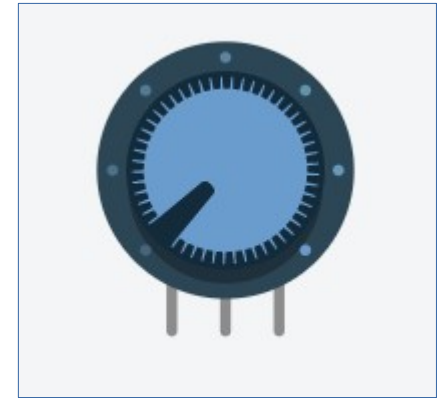
```
digitalWrite(13, LOW); // desativa o pino digital 13
```

# Arduino

## Função **analogRead()**

- Descrição:  
Lê o valor de um pino analógico especificado.  
Converte tensões de 0V a 5V em valores entre 0 e 1023
- Sintaxe  
`analogRead(pino)`
- Parâmetros  
pino: o nome do pino de entrada analógica que se quer ler (A0 a A5 na maioria das placas)
- Retorna  
A leitura analógica no pino (um inteiro que pode assumir de 0 a 1023)

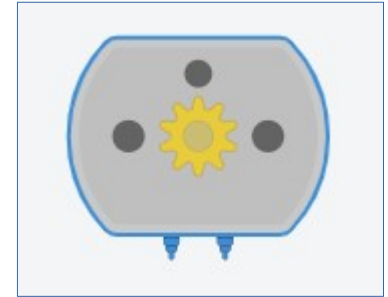
```
int val = analogRead(analogPin); // lê o valor do pino
```



Ex: Posição de um potenciômetro

# Arduino

## Função **analogWrite()**



Ex: Velocidade de um motor

- Descrição:  
Aciona uma onda PWM em um pino.  
Pode ser usada para variar o brilho de um LED ou acionar um motor a diversas velocidades. (somente nos pinos identificados com ~)
- Sintaxe  
`analogWrite(pino, valor)`
- Parâmetros  
pino: o pino escolhido do Arduino.  
valor: o duty cycle: entre 0 (sempre desligado) and 255 (sempre ligado).

```
analogWrite(13, 100);
```

# Arduino

## Função **map()**

- Descrição:  
Remapeia um número de um intervalo para outro.
- Sintaxe  
=map(valor, deMenor, deMaior, paraMenor, paraMaior)
- Parâmetros  
valor: o número a ser mapeado  
deMenor: o menor limite do intervalo atual do valor  
deMaior: o maior limite do intervalo atual do valor  
paraMenor: o menor limite do intervalo alvo  
paraMaior: o maior limite do intervalo alvo

```
// Ex: se val for 511, o resultado será 127  
int novoVal = map(val, 0, 1023, 0, 255);
```



# Arduino

## Função **delay()**

- Descrição:  
Pausa o programa por uma quantidade especificada de tempo (em milissegundos).
- Sintaxe  
`delay(ms)`
- Parâmetros  
ms: o número de milissegundos para pausar o programa

```
delay(1000); // pausa o programa por um segundo
```

# Arduino

## Função **millis()**

- Descrição:  
Retorna o número de milissegundos decorridos desde que a placa Arduino começou a executar o programa atual.
- Sintaxe  
= millis()
- Retorna  
O número de milissegundos passados desde que o programa iniciou (unsigned long)

```
unsigned long time = millis(); // captura o instante atual
```

# Arduino

## Utilizando a Serial como método de depuração

- Inicializar a comunicação serial na função Setup
- Escrever valores na serial indicando trechos de código atingidos ou mesmo valores de variáveis
- Exemplo:

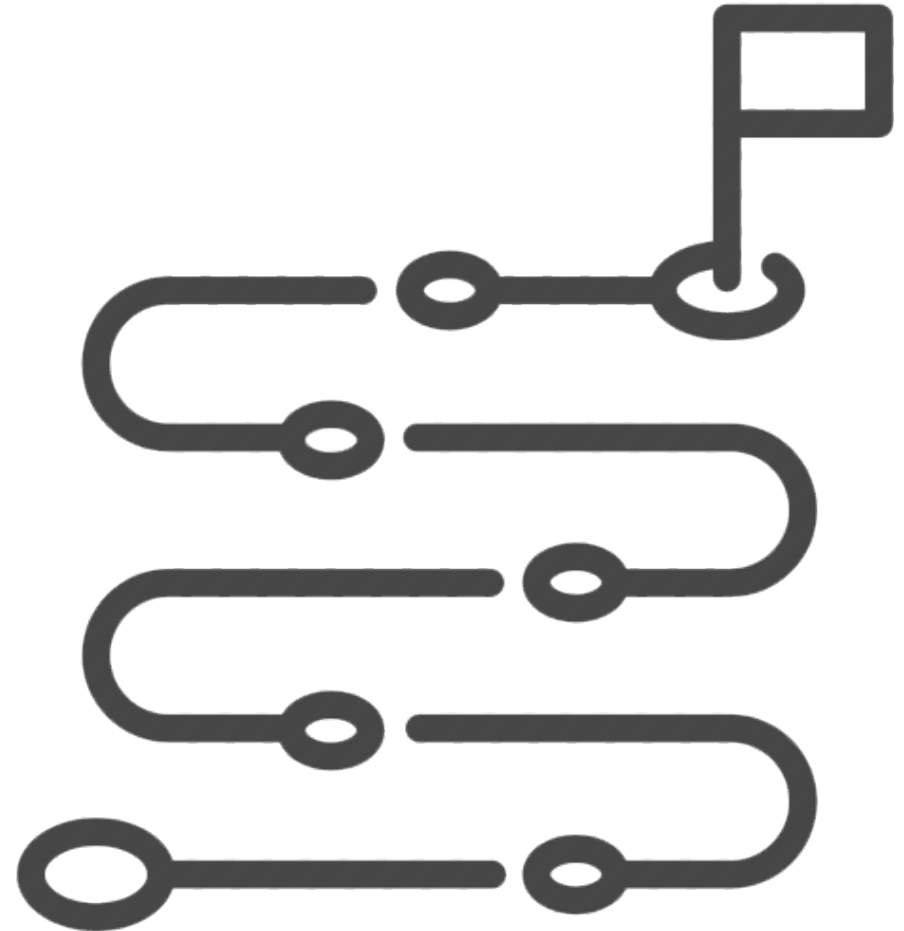
```
void setup() {  
    Serial.begin(9600);  
    ...  
}  
void loop() {  
    ...  
    if (condicao)  
    {  
        Serial.println("Entrou no teste 1");  
        int v = analogRead(A0);  
        Serial.println(String(v));  
    }  
    ...  
}
```

# Agenda

1) Introdução ao Arduino

**2) Exemplos no Tinkercad**

3) Exercícios



## Prática no Tinkercad



# Práticas – Arduino - Tinkercad

## Acesso à Sala de Aula

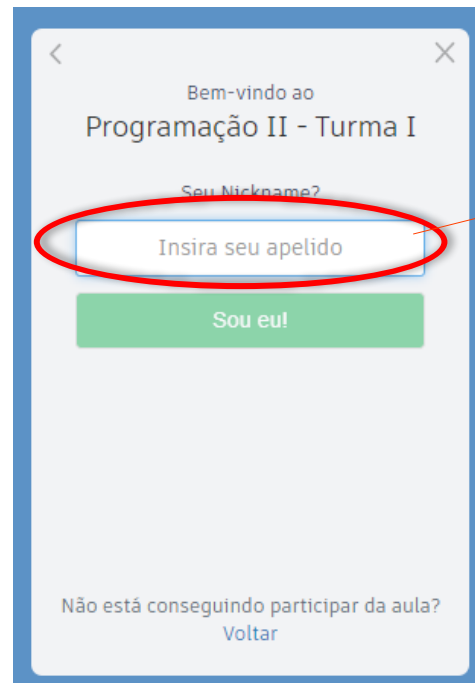
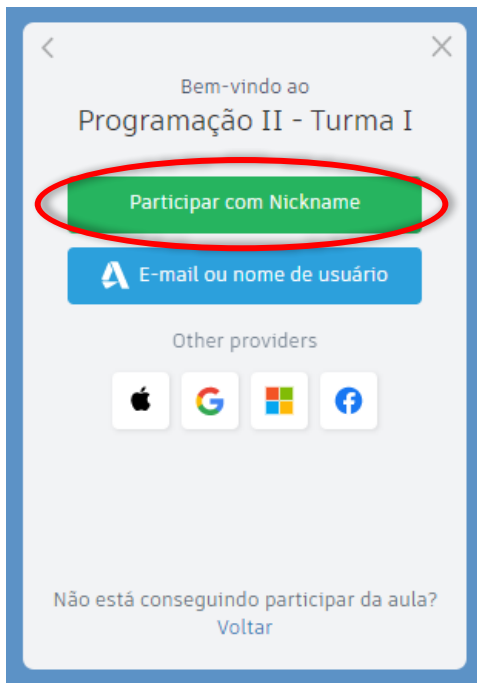
1) Acesse a aula em:

<https://www.tinkercad.com/joinclass/M3AUWETW71DC>

2) Insira o Apelido que seu professor o atribuiu e que está listado na próxima página

# Práticas – Arduino - Tinkercad

## Códigos individuais de acesso



Alunos	Informações de login
<input type="radio"/>	ANDERSON COSTA andersoncosta0960
<input type="radio"/>	ELISA DE VASCONCELOS ... elisadevasconcel3050
<input type="radio"/>	GIOVANE JOSE DA SILVA giovanejosedasil6473
<input type="radio"/>	GUILHERME MATHEUS D... guilhermematheus4919
<input type="radio"/>	HELIO CARVALHO DE AN... heliocarvalhodea9956
<input type="radio"/>	LUCAS FELIPE DE ALMEI... lucasfelipedealm4742
<input type="radio"/>	TAIRINI SANTANA DE SO... tairinisantanade6619

# Práticas – Arduino - Tinkercad

## Acesso ao Circuitos

The screenshot displays the Autodesk Tinkercad web interface. In the top left corner, the Tinkercad logo and 'AUTODESK TINKERCAD' text are visible. Below the logo, the user's name 'MAURICIO EDGAR' is shown. A search bar labeled 'Pesquisar projetos...' is present. The left sidebar contains a menu with options: 'Projetos 3D', 'Circuitos' (highlighted with a red circle and a red '1'), 'Blocos de código' (with a 'NOVO' badge), 'Lições', 'Suas aulas', and 'Projetos' (with a '+ Criar projeto' button). The main area features a 'Circuits' section with a green 'Criar novo Circuito' button (circled in red with a red '2') and a 'Try Circuits' button. A banner at the top right promotes 'Tinkercad Lesson Plans' with a 'Learn more' link. The bottom of the interface has a green bar with the text '42 | Mecatrônica – DAMM – IFSC' on the left and 'TEM – Sistemas de Visão' on the right.

TINKERCAD AUTODESK TINKERCAD

MAURICIO EDGAR

Pesquisar projetos...

Projetos 3D

**Circuitos**

Blocos de código NOVO

Lições

Suas aulas

Projetos + Criar projeto

**Circuits**

Criar novo Circuito

Criar novo Circuito

Tinker with Circuits on Tinkercad!

Try Circuits

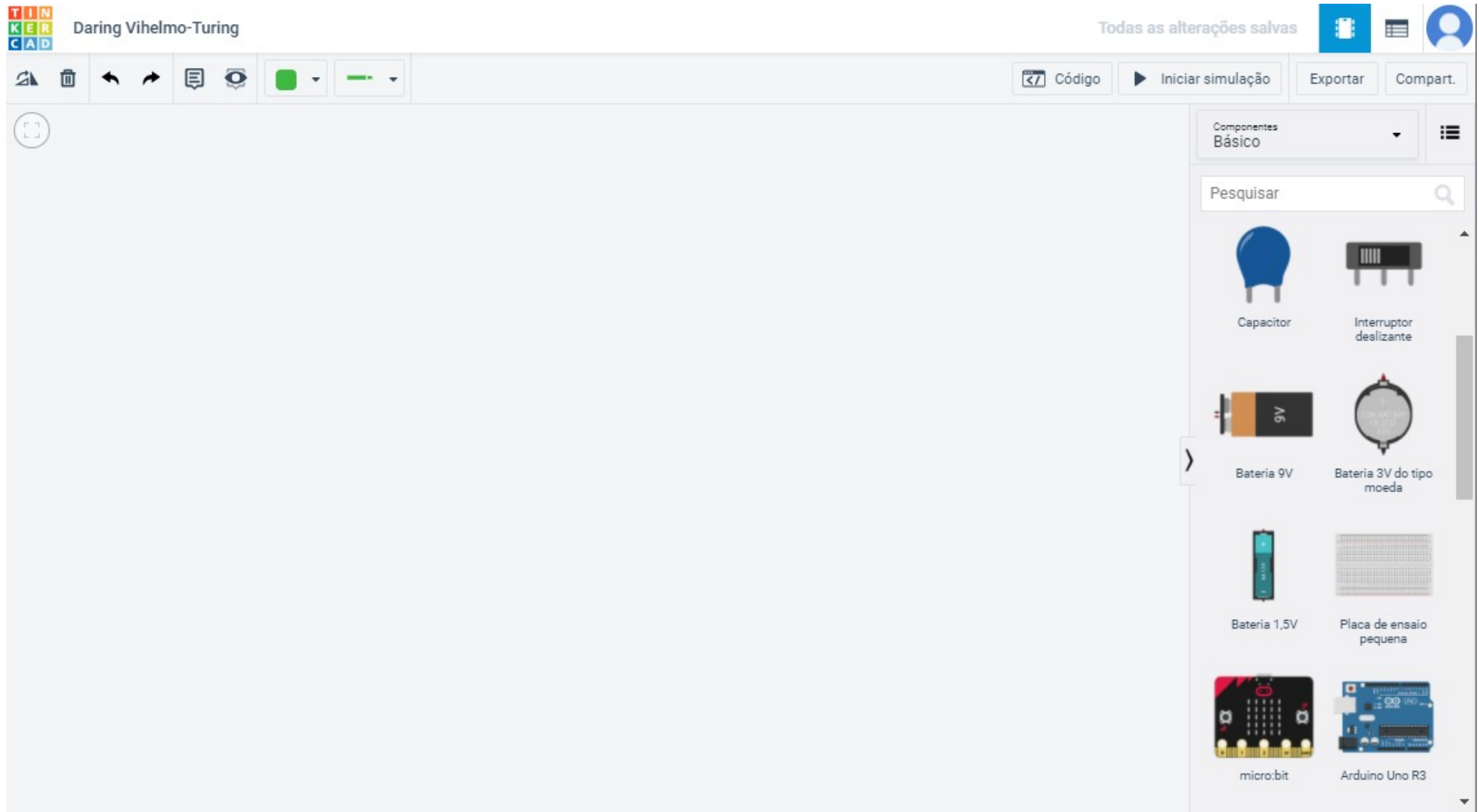
Tinkercad Lesson Plans

Tinkercad lesson plans are ready to use online or in the classroom. Discover curriculum developed in partnership with teachers. [Learn more](#)



# Práticas – Arduino - Tinkercad

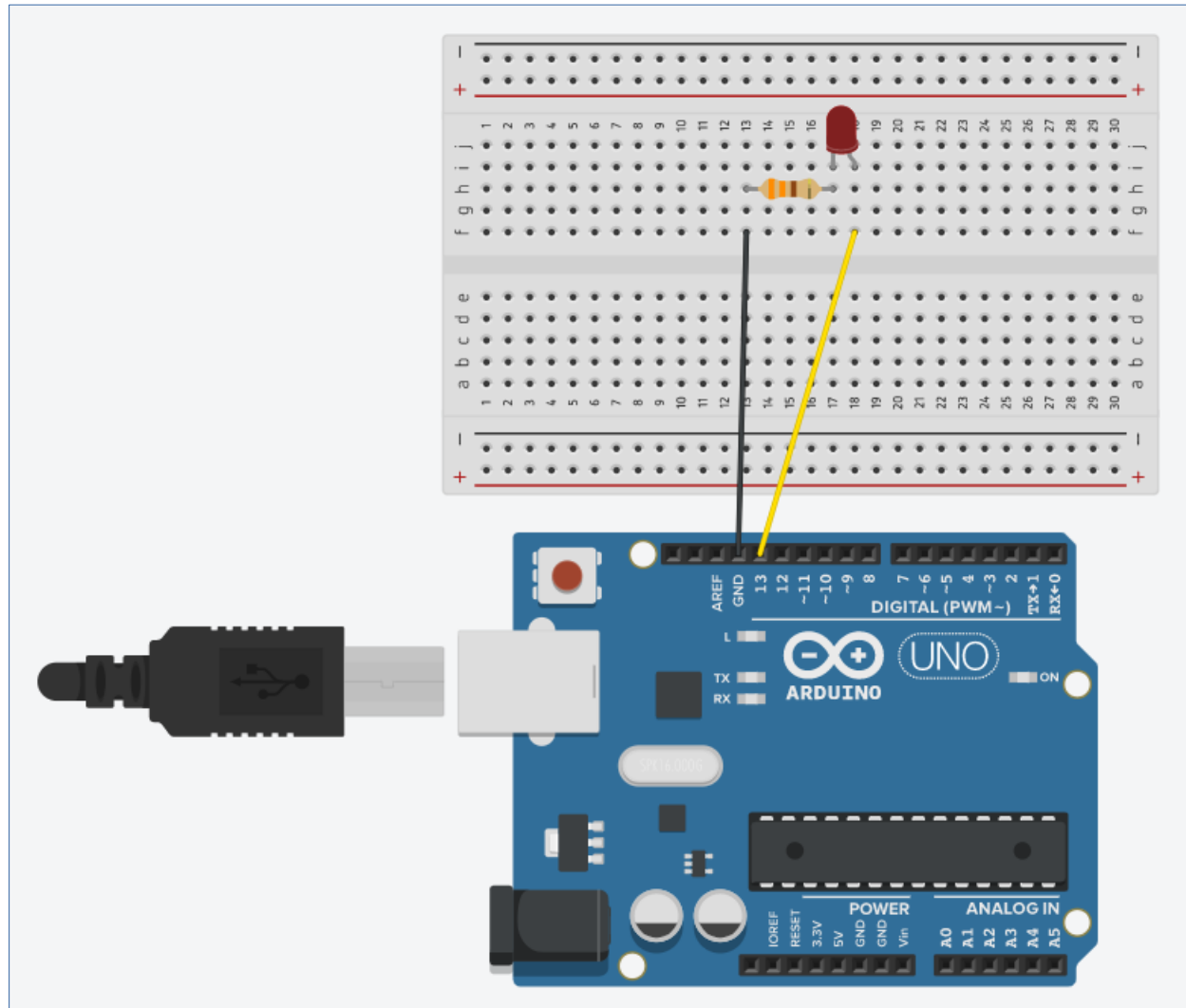
## Visão Geral do Ambiente



# Exemplo #1

## Piscando um LED – Circuito

- Arduino
- Placa de ensaio
- Led
- Resistor (330 $\Omega$ )
- Fios



# Exemplo #1

## Piscando um LED – Código

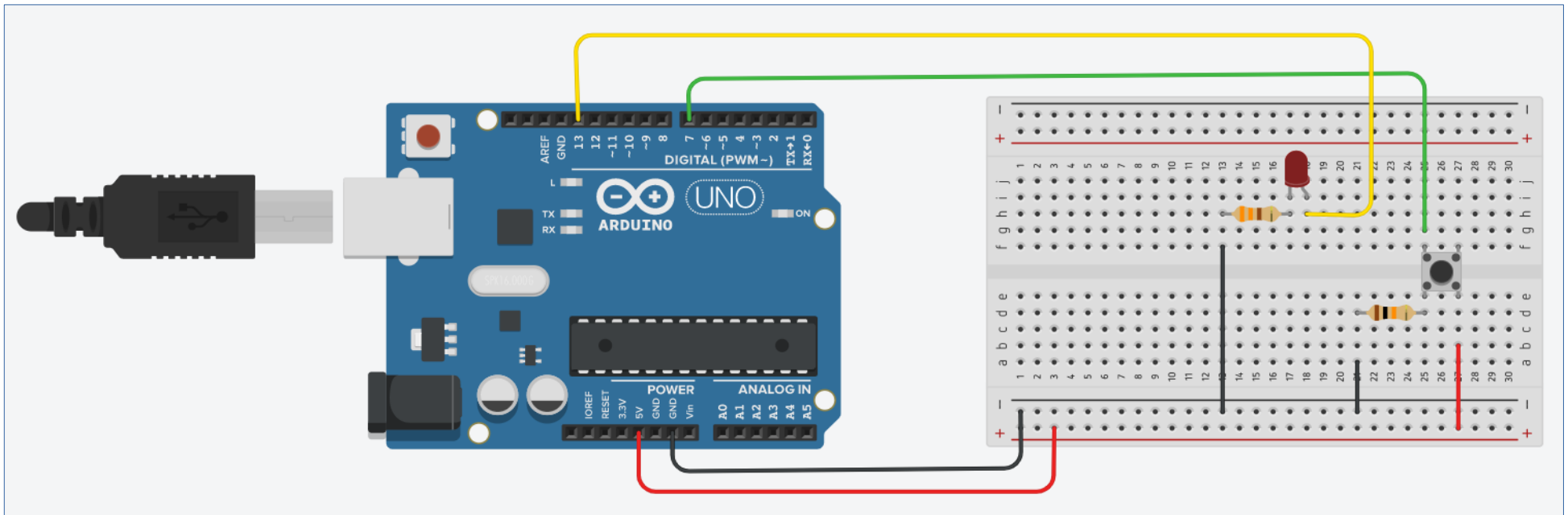
```
// executa 1 vez no início
void setup() {
    // inicializa o pino digital 13 como sendo de saída
    pinMode(13, OUTPUT);
}

// executa repetidamente
void loop() {
    digitalWrite(13, HIGH);    // liga o led
    delay(1000);               // aguarda por 1 segundo
    digitalWrite(13, LOW);     // desliga o led
    delay(1000);               // aguarda por 1 segundo
}
```

## Exemplo #2

### Acendendo um LED ao detectar botão pressionado – Circuito

- Arduino
- Led
- Botão
- Fios
- Placa de ensaio
- Resistor do led ( $330\Omega$ )
- Resistor do botão ( $10k\Omega$ )



# Exemplo #2

## Acendendo um LED ao detectar botão pressionado – Código

```
#define LED_PIN 13 // associa o pino do led a um nome
#define BUTTON_PIN 7 // associa o pino do botão a um nome

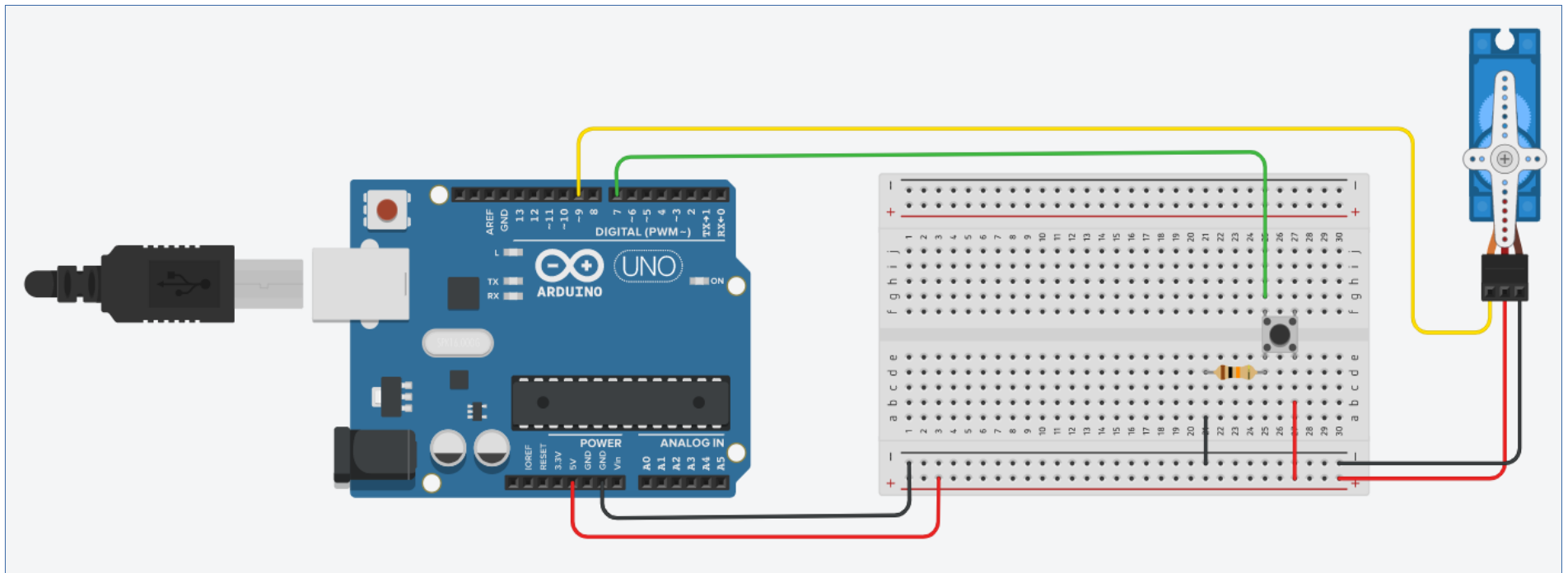
void setup() {
    pinMode(LED_PIN, OUTPUT); // define o pino do led como sendo de saída
    pinMode(BUTTON_PIN, INPUT); // define o pino do botão como sendo de entrada
}

void loop(){
    if (digitalRead(BUTTON_PIN) == HIGH) { // verifica se o botão está pressionado
        digitalWrite(LED_PIN, HIGH); // liga o led
    }
    else {
        digitalWrite(LED_PIN, LOW); // desliga o led
    }
}
```

# Exemplo #3

Posicionar um servomotor ao detectar botão pressionado – Circuito

- Arduino
- Servo motor
- Resistor do botão (10kΩ)
- Placa de ensaio
- Botão
- Fios



# Exemplo #3

## Posicionar um servomotor ao detectar botão pressionado – Código

```
#include <Servo.h> // Importa biblioteca para controle de servomotores

#define SERVO_PIN 9 // associa o pino do servo a um nome
#define BUTTON_PIN 7 // associa o pino do botão a um nome

Servo servo;

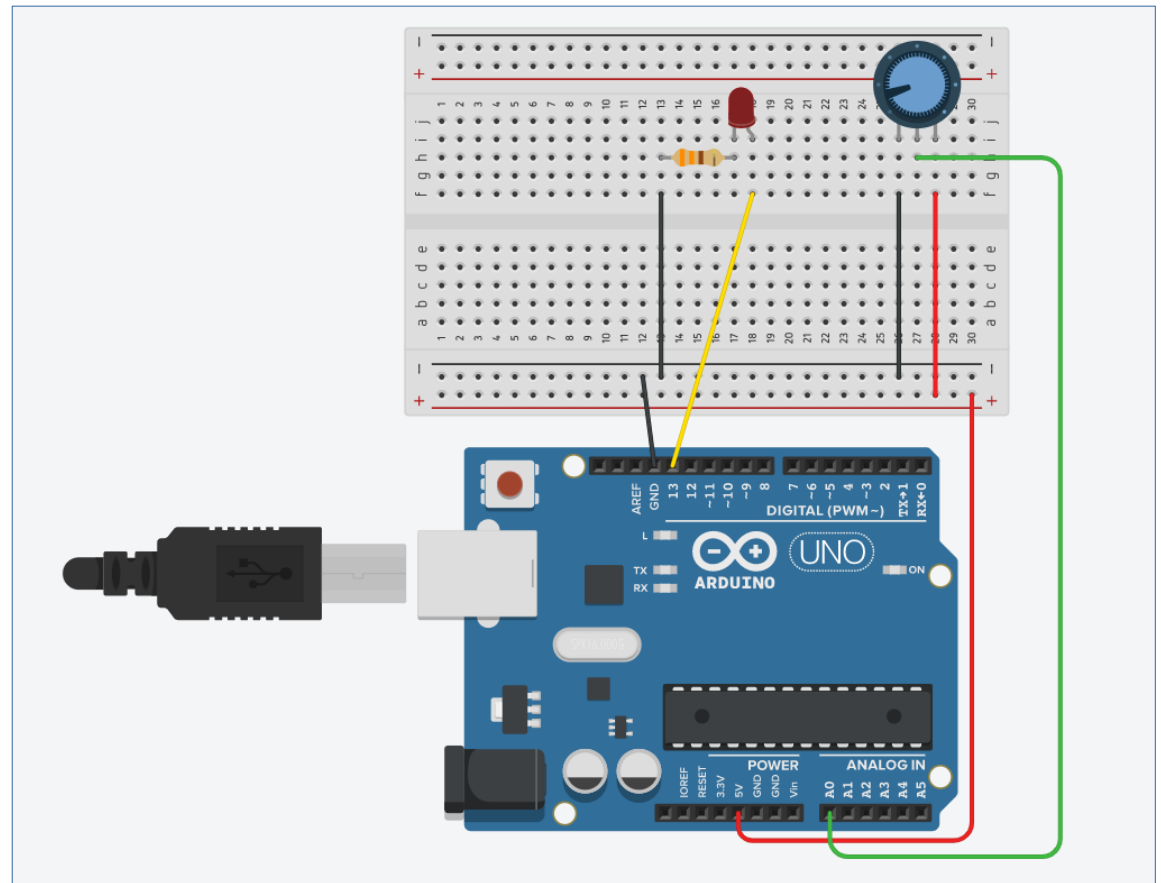
void setup() {
    pinMode(BUTTON_PIN, INPUT); // define o pino do botão como sendo de entrada
    servo.attach(SERVO_PIN); // associa o pino ao servo
}

void loop(){
    if (digitalRead(BUTTON_PIN) == HIGH) { // verifica se o botão está pressionado
        servo.write(180); // define angulo 180 ao servo (pode variar de 0 a 180)
    }
    else {
        servo.write(0); // define angulo 0 ao servo
    }
}
```

## Exemplo #4

Alterar a frequência do acionamento de um led em função do valor de um potenciômetro – Circuito

- Arduino
- Placa de ensaio
- Led
- Resistor (330Ω)
- Potenciômetro
- Fios





# Exemplo #4

## Alterar a frequência do acionamento de um led em função do valor de um potenciômetro – Circuito

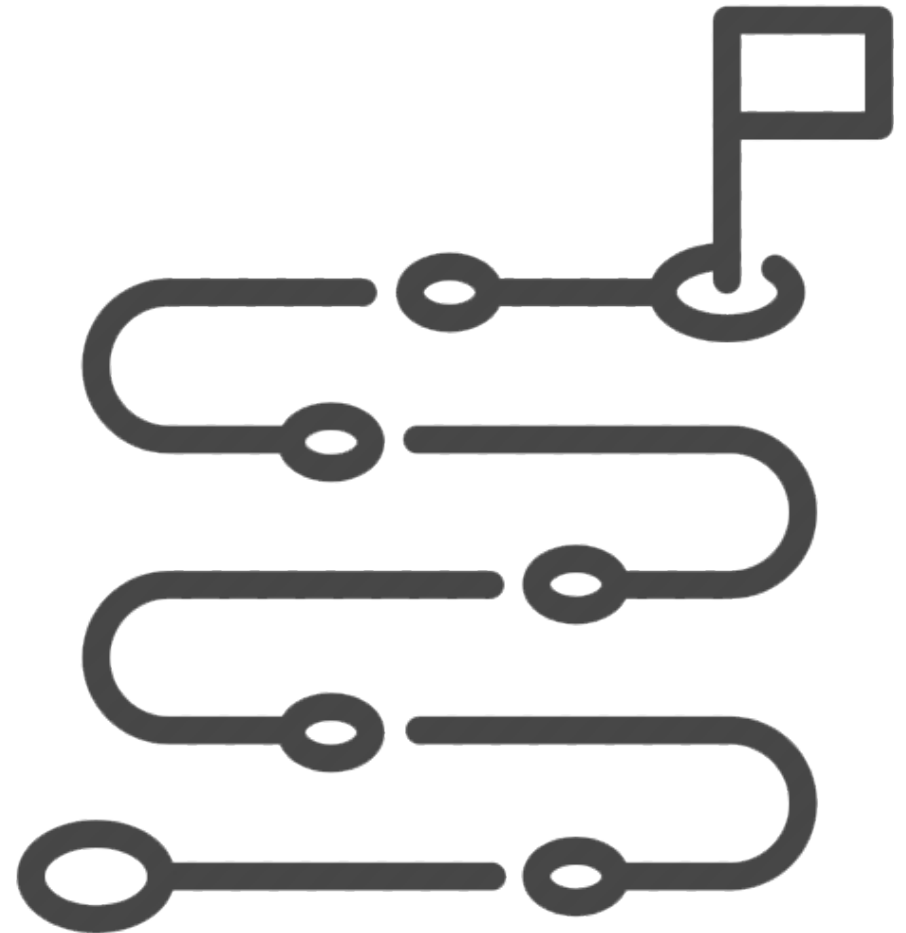
```
int vp; // declaração da variável utilizada na leitura do potenciômetro

void setup() {
    pinMode(13, OUTPUT); // define o pino do LED como sendo de saída
    pinMode(A0, INPUT); // define o pino do potenciômetro como sendo de saída
}

void loop() {
    vp = analogRead(A0); // realiza a leitura do valor do potenciômetro (0 - 1023)
    digitalWrite(LED_BUILTIN, HIGH); // liga o led
    delay(vp); // aguarda tempo em milissegundos associado à leitura do potenciômetro
    digitalWrite(LED_BUILTIN, LOW); // desliga o led
    delay(vp); // aguarda tempo em milissegundos associado à leitura do potenciômetro
}
```

# Agenda

- 1) Introdução ao Arduino
- 2) Exemplos no Tinkercad
- 3) Exercícios**



# Exercícios

- 1) Reproduza os projetos de exemplo da aula no Tinkercad e avalie o funcionamento.
- 2) Crie o projeto de um semáforo com um led verde, um led amarelo e um led vermelho. O led verde permanecerá aceso por 20 segundos, o amarelo por 10 segundos e o vermelho por 20 segundos.
- 3) Crie um projeto contendo um led e um botão. O led piscará 50 vezes com intervalos de 100 ms quando o botão for pressionado.
- 4) Crie um projeto contendo um led e um botão. O estado do led deve ser alternado entre ligado e desligado quando o botão for pressionado.
- 5) Crie um projeto contendo um led e um botão. Quando ocorrem dois pressionamentos consecutivos do botão o led se acenderá e permanecerá aceso pelo mesmo intervalo de tempo que transcorreu entre os pressionamentos do botão.
- 6) Crie um projeto contendo dois botões e um servo motor. Um dos botões serve para incrementar o ângulo do servo motor em 5 graus, enquanto o outro botão serve para decrementar o ângulo na mesma quantidade.

# Exercícios

- 7) Crie um projeto contendo um potenciômetro e um servomotor. O valor do potenciômetro deve ser usado para definir o ângulo do servo motor.
- 8) Crie um projeto contendo um led e um potenciômetro. O valor do potenciômetro deve ser usado para definir a intensidade do led.

# Arduino

Dúvidas?

