Interação com Botões

Interação com Botões

O que são botões?

Podem ser definidos como contatos mecânicos cujo objetivo é controlar a passagem de corrente em um sistema a partir de uma força externa.

Tipos de botões

- Momentâneos
- Memorizados
- Magnéticos

Interação com Botões

pushbutton

O botão *pushbutton* é um interruptor que conduz a corrente elétrica apenas quando é pressionado. Geralmente, é o mais utilizado na prototipagem de projetos eletrônicos.

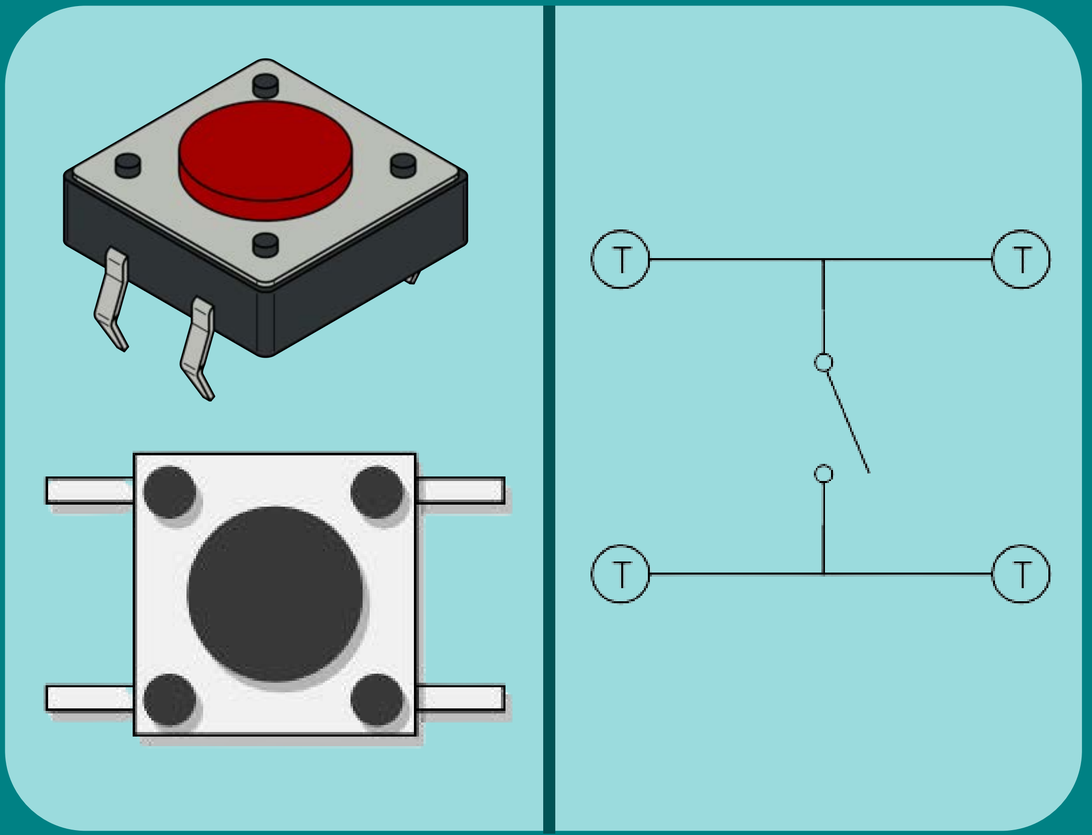

Figura: Desenho do *pushbutton*

Interação com Botões

Como usar o botão?

Para usar o *pushbutton* com a placa do arduino é necessário configurar o modo do pino conectado ao botão como entrada (*INPUT*),

O problema

Configurando um pino como *INPUT,* quando o botão estiver em estado aberto, a entrada estará "flutuando", resultando em leituras imprevisíveis.

Interação com Botões

Solução

Para assegurar uma leitura correta quando o botão estiver aberto, um resistor de pull-up ou pull-down deve ser usado. O propósito desse resistor é colocar o pino em um estado conhecido quando o botão estiver aberto

Pull-down

Se um resistor de pull-down é usado, o pino de entrada estará em LOW quando o botão estiver aberto e HIGH quando o btoão estiver pressionado.

Pull-up

Se um resistor de pull-up é usado, o pino de entrada estará em HIGH quando o botão estiver aberto e LOW quando o btoão estiver pressionado.

PET Tele

Interação com Botões

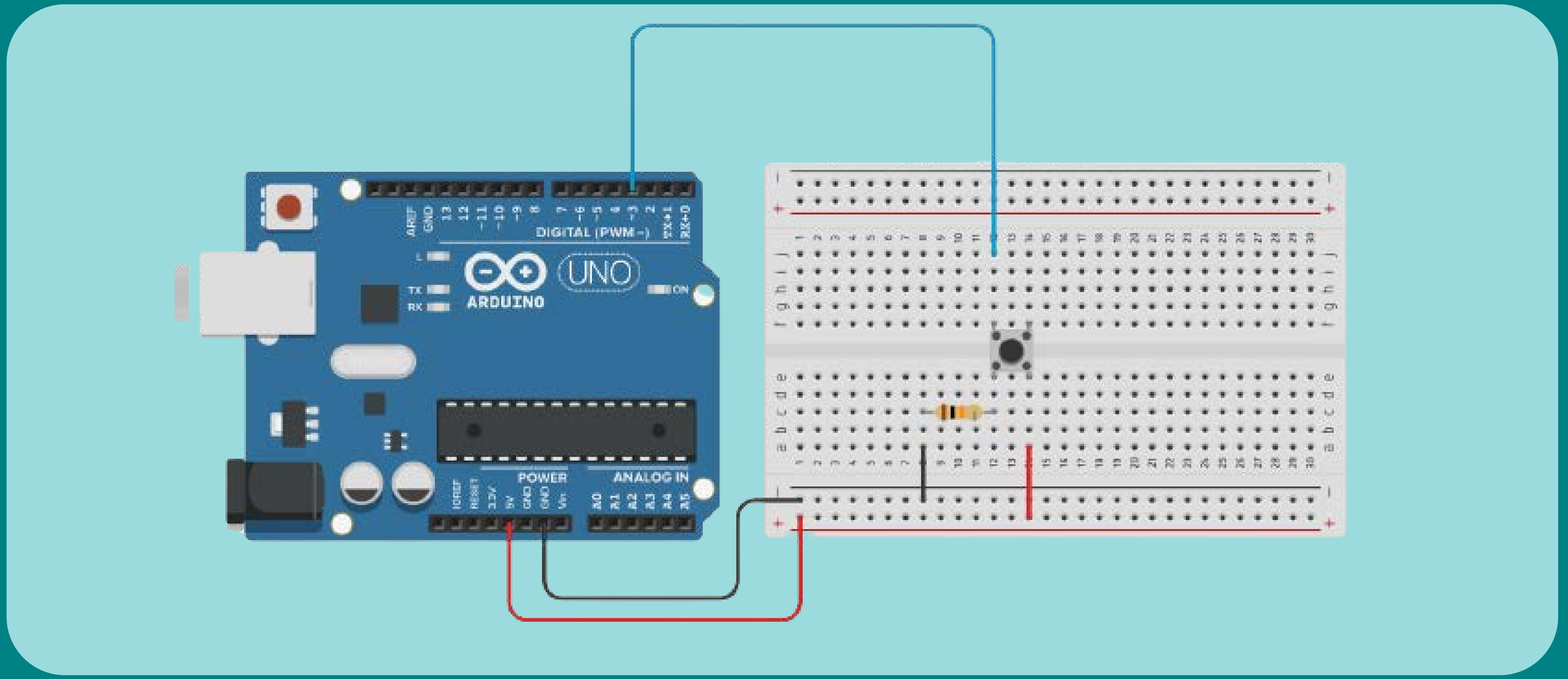

Figura: Circuito com resistor de **pull-down**

Interação com Botões

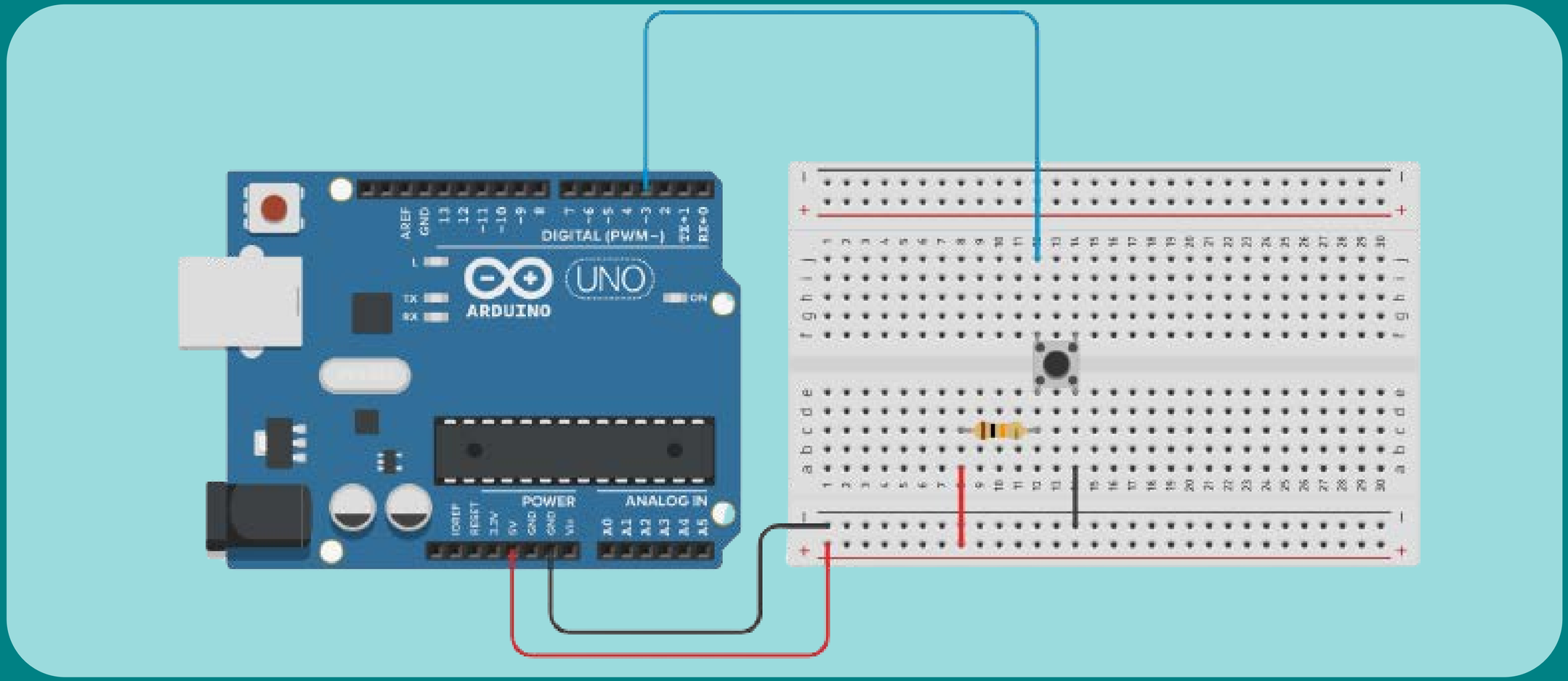

Figura: Circuito com resistor de **pull-up**

Interação com Botões

Verificando o botão

Para verificar o estado de um botão, é necessário usar a função *digitalRead().*

digitalRead(pino)

A função digitalRead() lê o valor de um pino configurado como *INPUT* e retorna 0 (*LOW*) ou 1 (*HIGH*).

```
void loop(){

  int estadoBotao = digitalRead(pinoBotao);

}
```

Figura: Exemplo da função **digitalRead()**

Interação com Botões

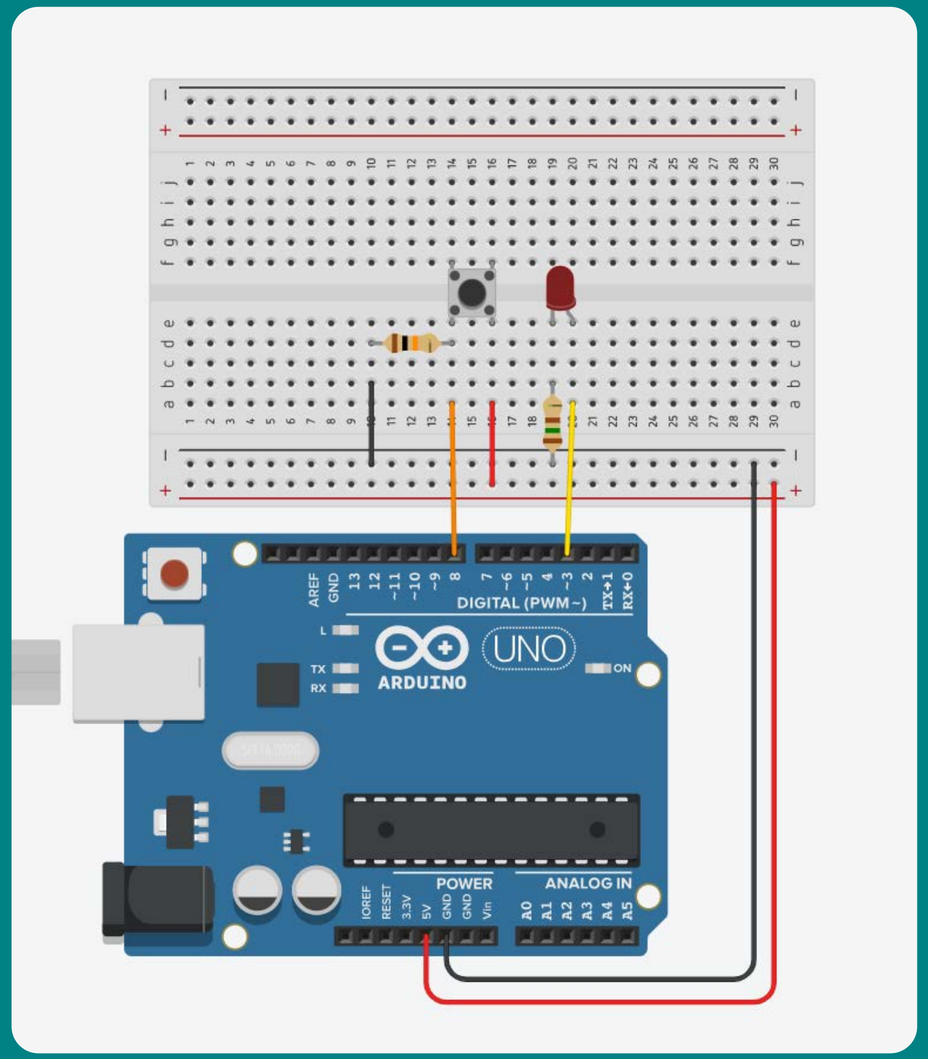

Figura: Circuito Botão (pull-down)

```arduino
int pinoBotao = 8;
int pinoLed = 3;

void setup(){
  pinMode(pinoBotao, INPUT);
  pinMode(pinoLed, OUTPUT);
}

void loop(){
  int estadoBotao = digitalRead(pinoBotao);

  if (estadoBotao == HIGH){

    digitalWrite(pinoLed, HIGH);

  }
  else if (estadoBotao == LOW){

    digitalWrite(pinoLed, LOW);

  }

}
```

Figura: Código Botão

Faça agora um botão para acender o LED e outro para apagar!

Interação com Botões

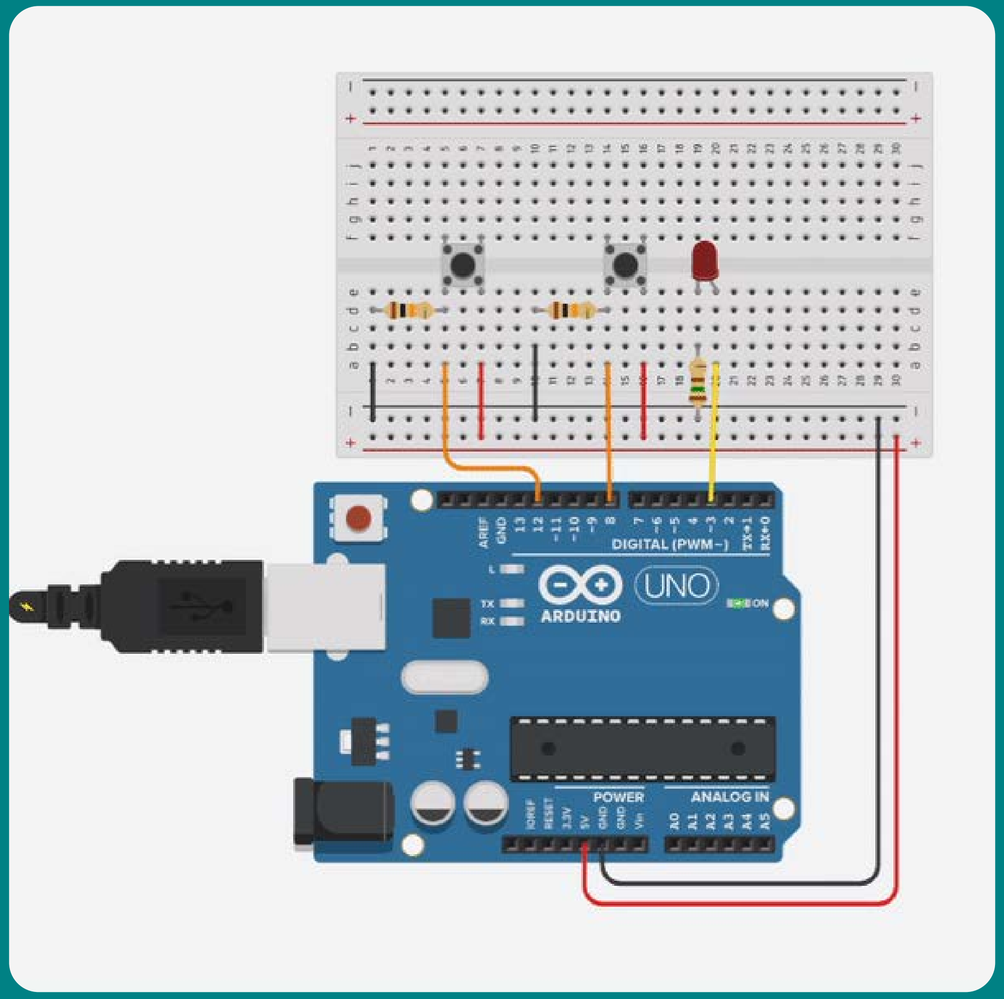

Figura: Circuito Exercício

```arduino
int pinoBotao1 = 8;
int pinoBotao2 = 12;
int pinoLed = 3;

void setup()
{
  pinMode(pinoBotao1, INPUT);
  pinMode(pinoBotao2, INPUT);
  pinMode(pinoLed, OUTPUT);
}

void loop()
{
  int estadoBotao1 = digitalRead(pinoBotao1);

  int estadoBotao2 = digitalRead(pinoBotao2);

  if (estadoBotao1 == HIGH){
    digitalWrite(pinoLed, HIGH);
  }
  if (estadoBotao2 == HIGH){
    digitalWrite(pinoLed, LOW);
  }
}
```

Figura: Código Exercício

Funções e Monitor Serial

Funções e Monitor Serial

Monitor Serial

O monitor serial é um programa que proporciona uma ligação de alto nível entre o Arduino e o computador.

Através dele, pode-se trocar mensagens entre o computador e o Arduino e, com isso, manter um monitoramento do sistema.

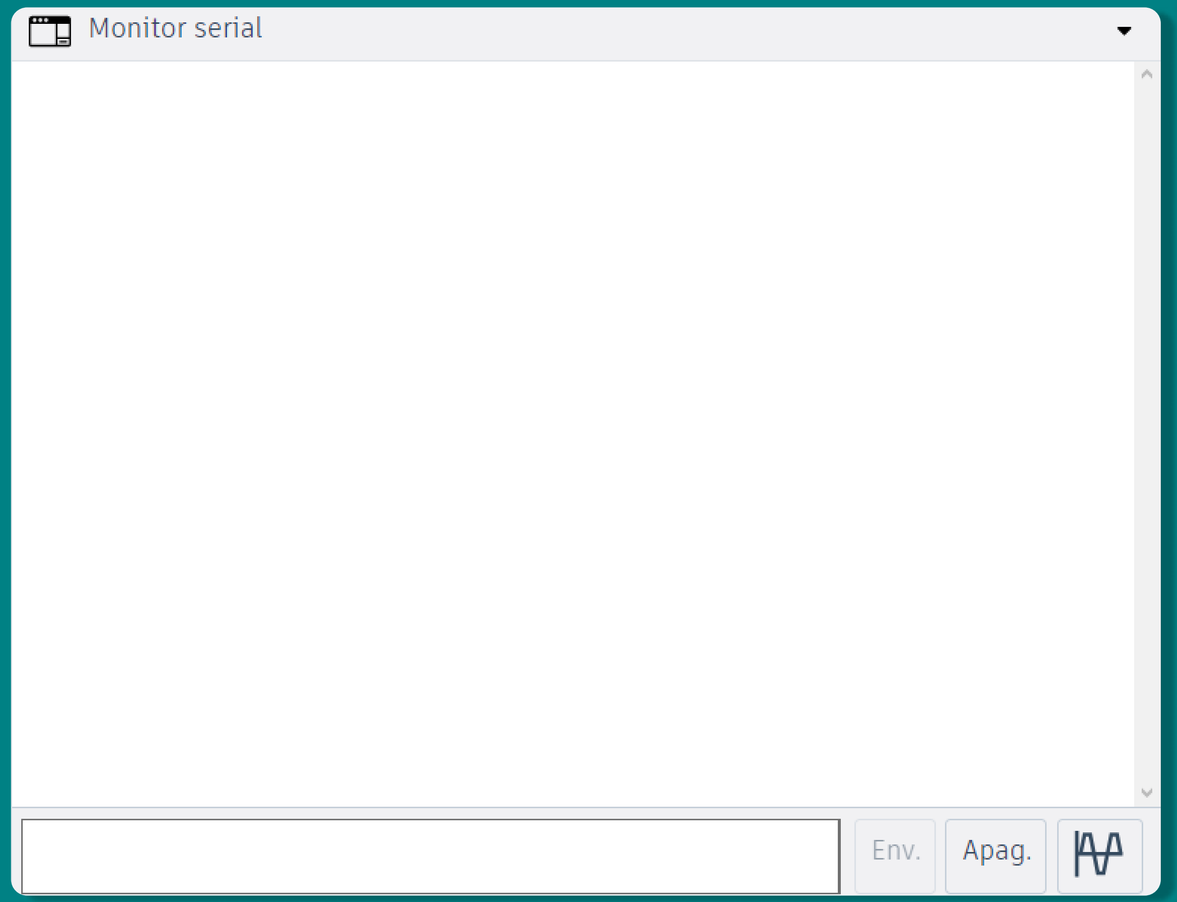

Figura: Monitor Serial no Tinkercad

Funções e Monitor Serial

Principais Funções

- ***Serial.begin()***
- ***Serial.print()***
- ***Serial.println()***
- ***Serial.available()***
- ***Serial.read()***

Serial.begin(taxa)

Configura a taxa de transferência em bits por segundo (*baud rate*) para transmissão serial.

```
void setup()
{
  Serial.begin(9600);
}
```

Figura: Exemplo da função **Serial.begin()**

Funções e Monitor Serial

Serial.print(val)

Imprime dados na porta serial como texto ASCII (facilmente legível, diferentemente dos valores binários).

***val:*o valor a ser impresso - qualquer tipo de dado.**

É possível usar a funcao *Serial.println(val)* para imprimir pulando uma linha.

```
void loop()
{
  Seral.print("Grupo:");

  Serial.println("PET-Tele");
}
```

Figura: Exemplo da função **Serial.print()**

Funções e Monitor Serial

Serial.available()

Retorna o número de bytes (caracteres) disponíveis para leitura na porta serial.

Serial.read()

Lê dados recebidos na porta serial.

```
void loop()
{
  if(Serial.available() > 0) {

    char dado = Serial.read();

  }
}
```

Figura: Exemplo da função **Serial.read()** e **Serial.available().**

Funções e Monitor Serial

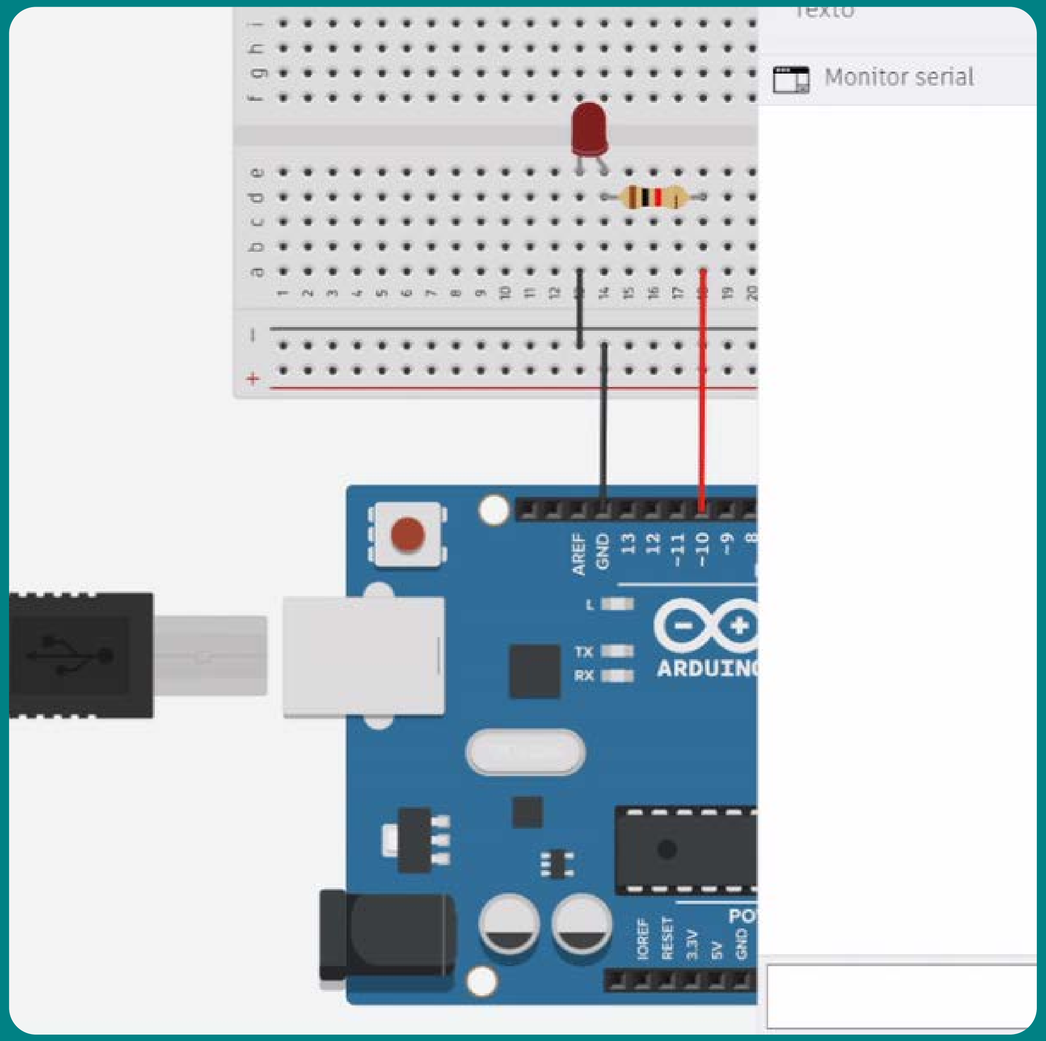

Figura: Circuito Monitor Serial

```ino
int pinoLed = 10;

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

void loop()
{
  if(Serial.available() > 0 ){

    char dado = Serial.read();

    if (dado == 'l') {

        digitalWrite(pinoLed, HIGH);
        Serial.println("LED Ativado!");
    }
  }
}
```

Figura: Código Monitor Serial

Complete o exemplo anterior com um caractere para apagar o LED, e imprima no Monitor Serial que o LED foi apagado.

Funções e Monitor Serial

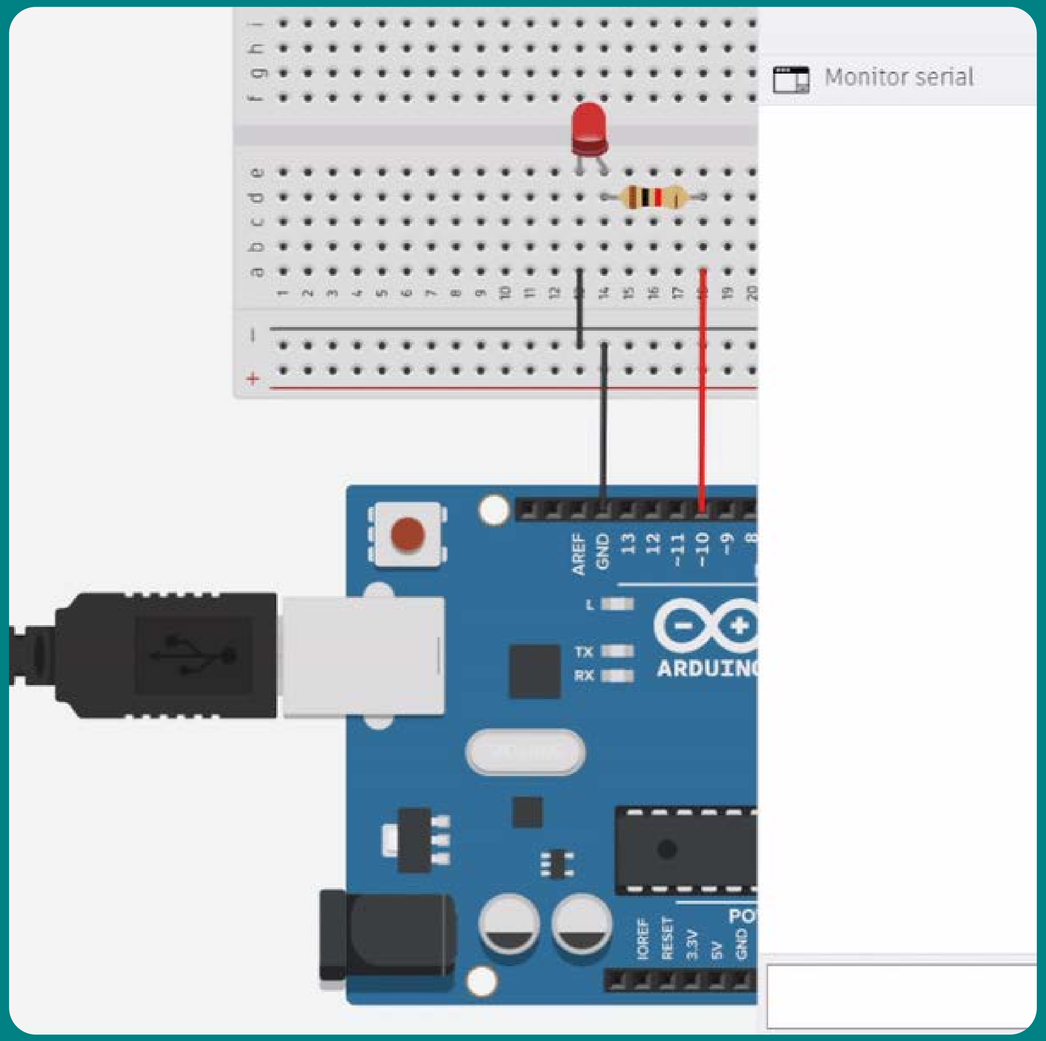

Figura: Circuito Exercício

```arduino
int pinoLed = 10;

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

void loop()
{
  if(Serial.available() > 0 ){

    char dado = Serial.read();

    if (dado == 'l') {

        digitalWrite(pinoLed, HIGH);
        Serial.println("LED Ativado!");

    }
    if (dado == 'k') {

        digitalWrite(pinoLed, LOW);
        Serial.println("LED Desativado!");
    }
  }
}
```

Figura: Código Exercício

Interação com Sensores

Interação com Sensores

Sensores de Movimento (PIR)

São capazes de detectar movimento de objetos que exalam calor e que estejam dentro do seu raio de detecção que alcança até 7 metros. Com o sensor atuando, qualquer objeto que se movimentar dentro do seu campo de detecção, fará com que a saída do mesmo seja ativada.

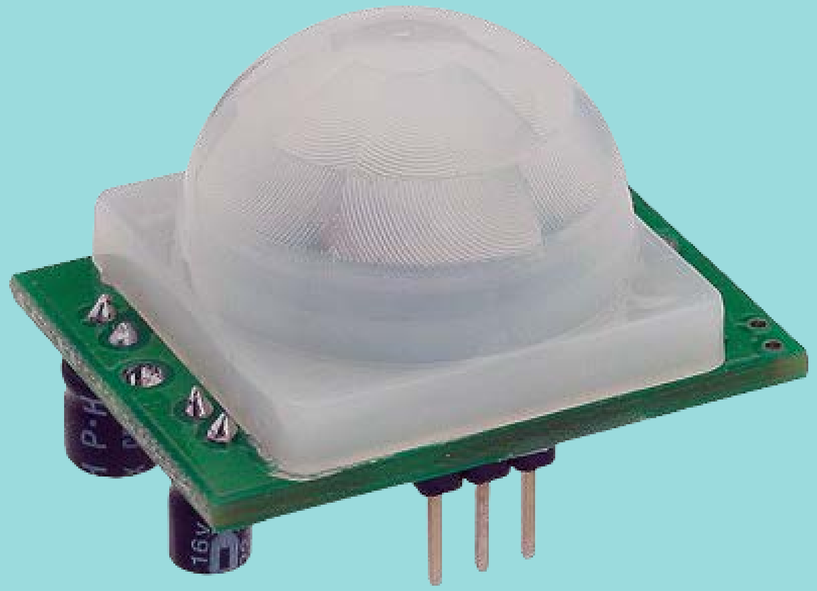

Figura: Sensor de Movimento PIR

Interação com Sensores

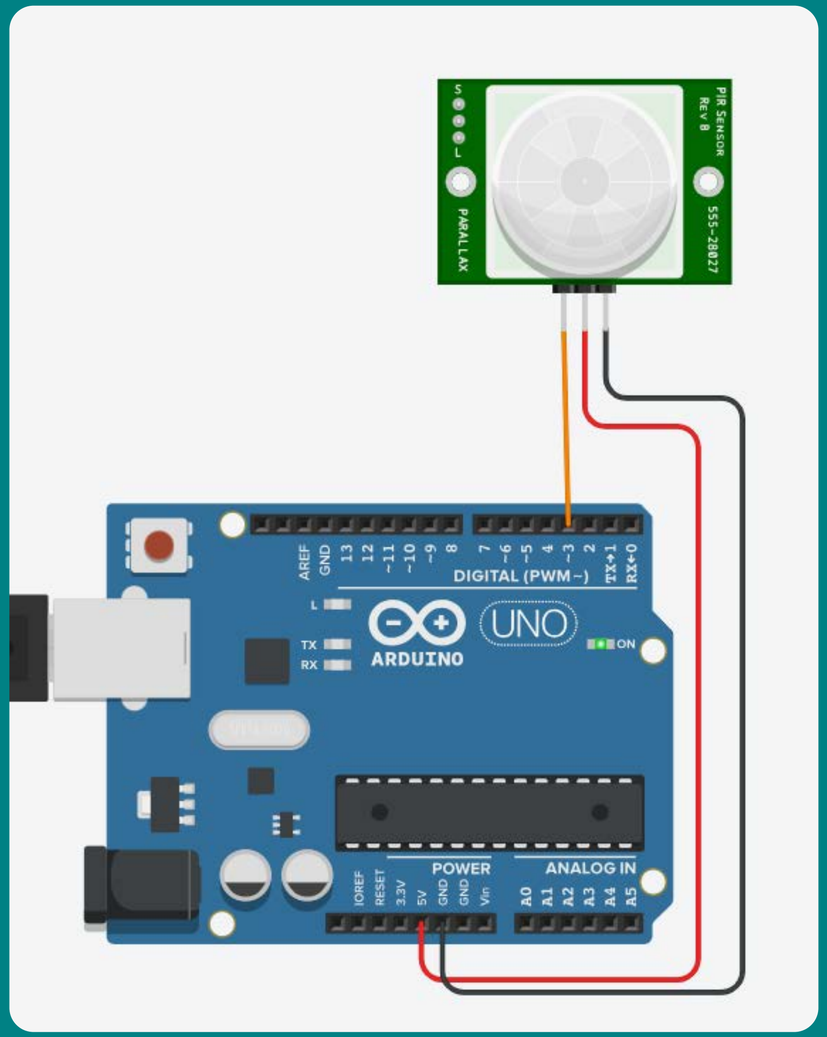

Figura: Circuito Sensor PIR

```ino
int pinoSensor = 3;

void setup()
{
  Serial.begin(9600);
  pinMode(pinoSensor, INPUT);
}

void loop()
{

  int valorLido = digitalRead(pinoSensor);


  if(valorLido == HIGH){

    Serial.println("Detectado");
  }

  delay(2000);
}
```

Figura: Código Sensor PIR

Faça com um LED ativado sempre que houver movimento e outro LED ativado enquanto não houver movimento.

Interação com Sensores

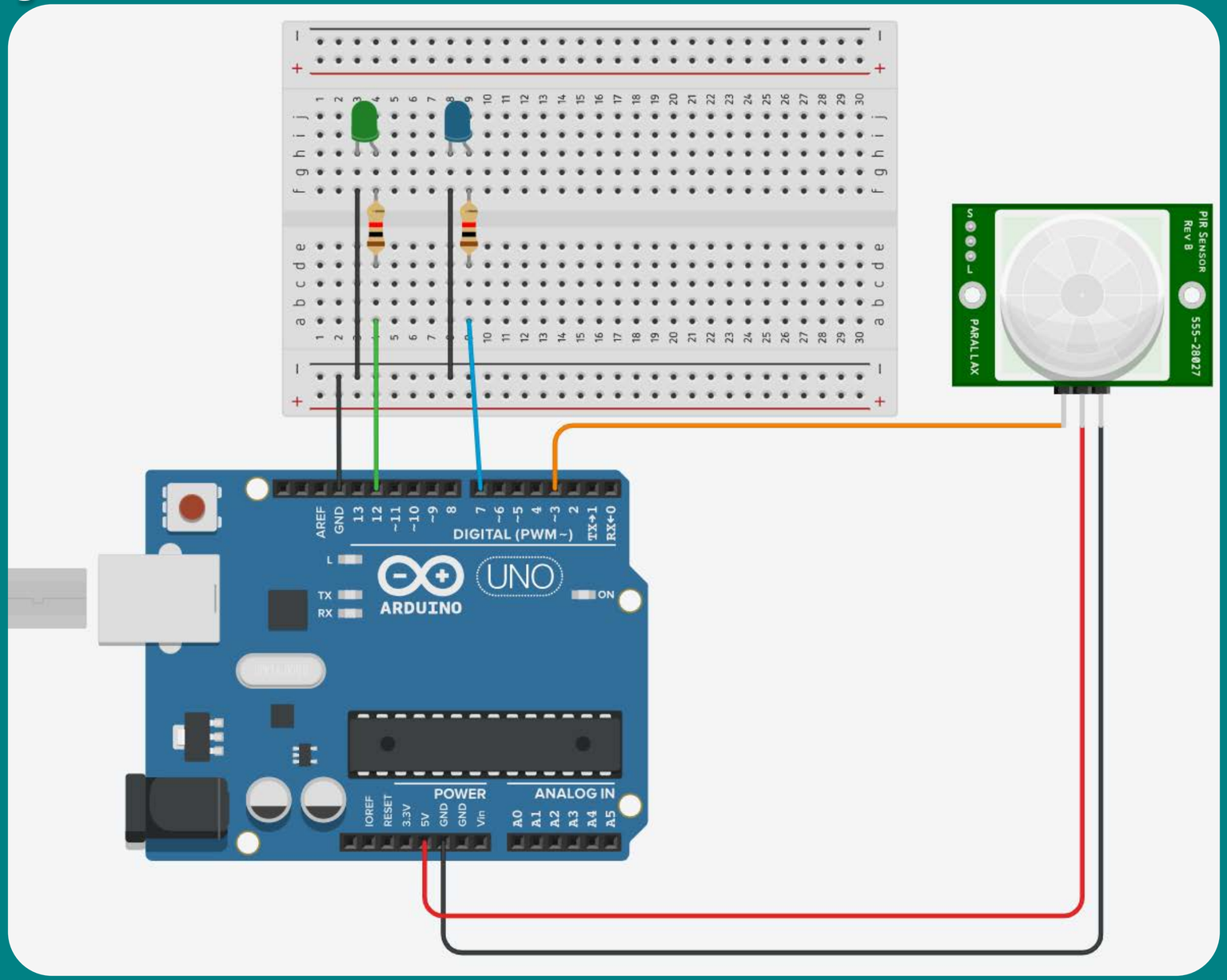

Figura: Circuito Exercício

Referências:

- https://www.arduino.cc/

- https://www.telecom.uff.br/pet/petws/index.php?pagina=downloads/curso_arduino .Curso de Introdução ao kit Arduino. **PET-Tele**