

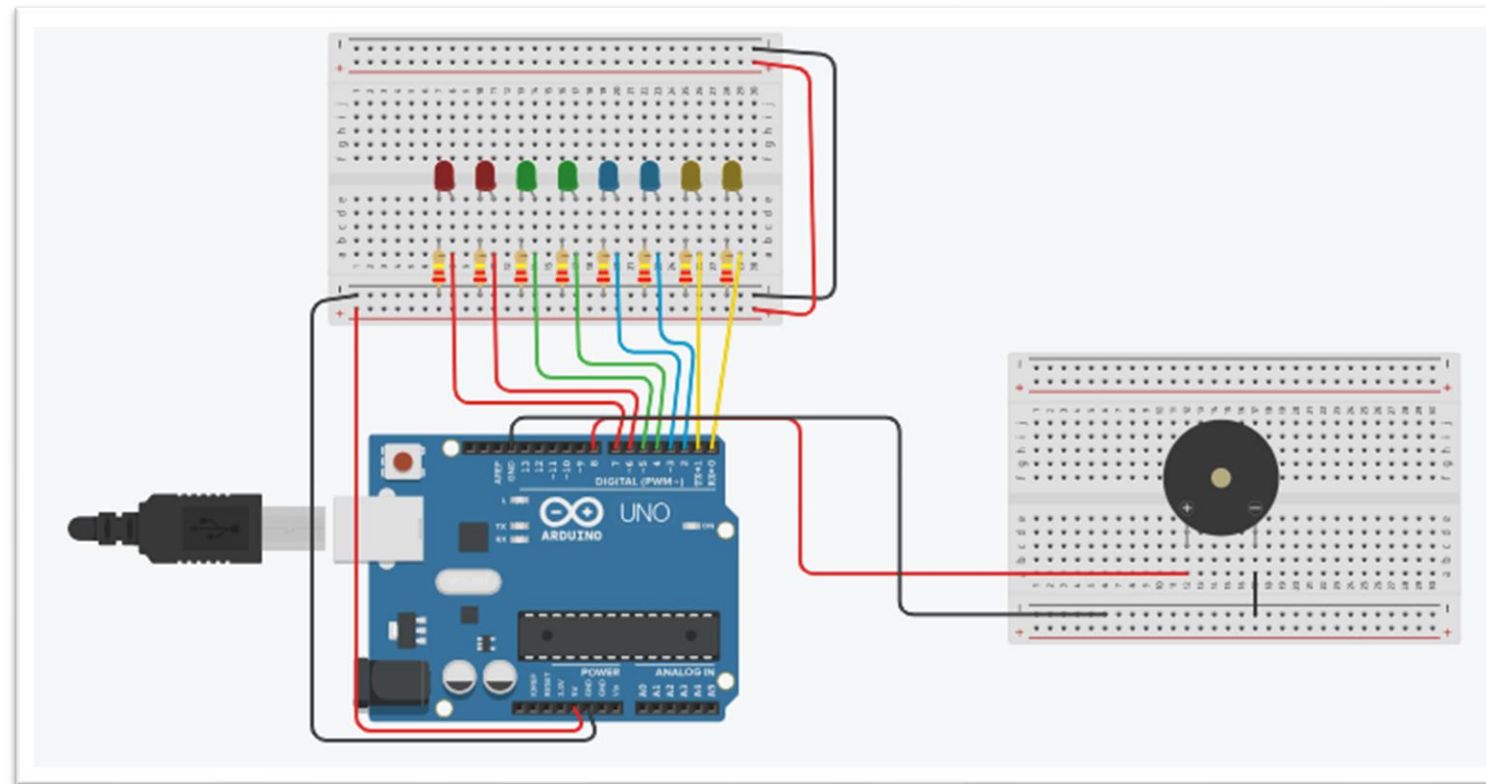
# PLACA ARDUÍNO

## CIRCUITOS – SHOW DE LEDS COM MÚSICA

# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MI FÁ

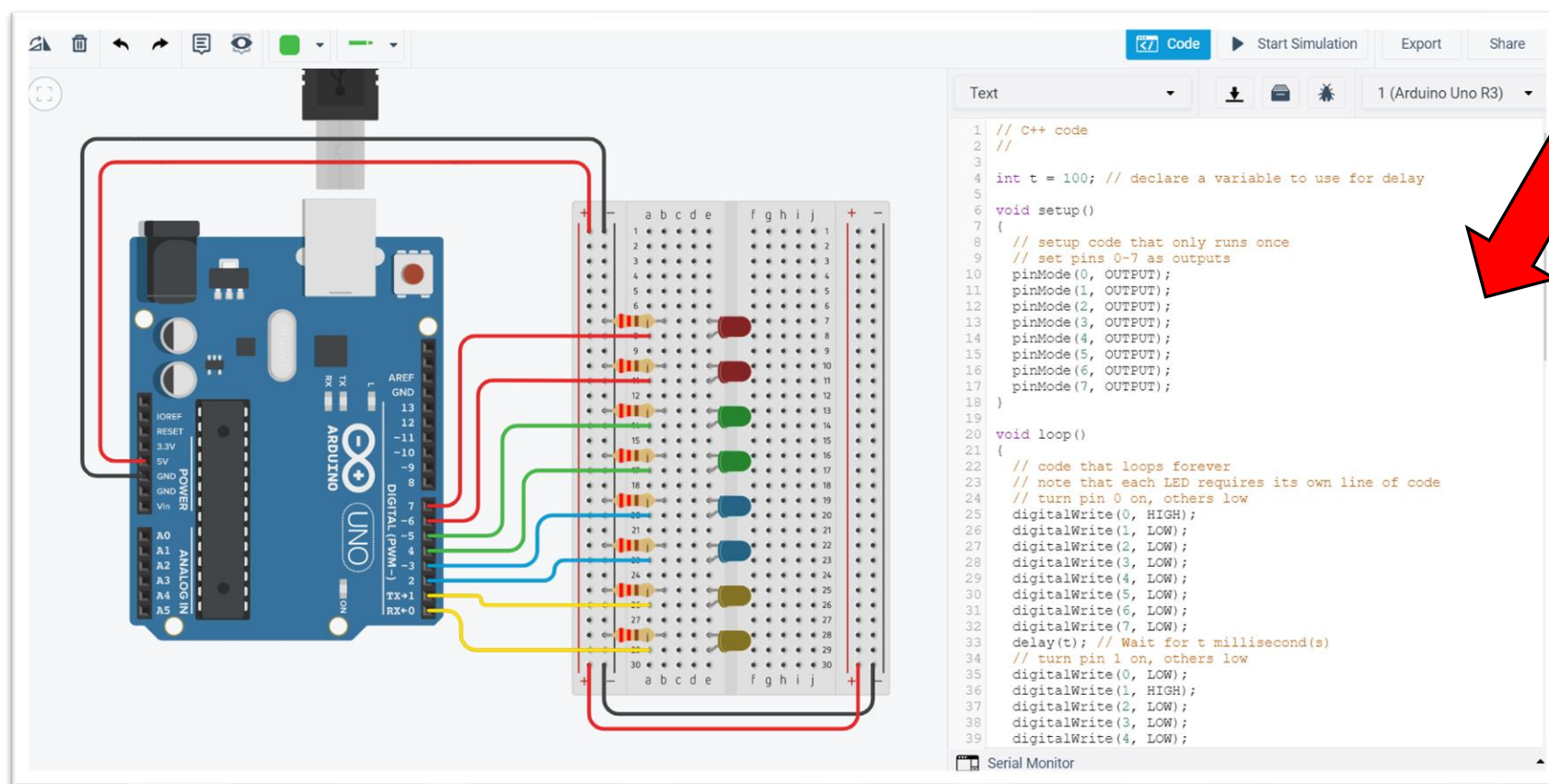
### TINKERCAD



# SHOW DE LEDS

## TINKERCAD

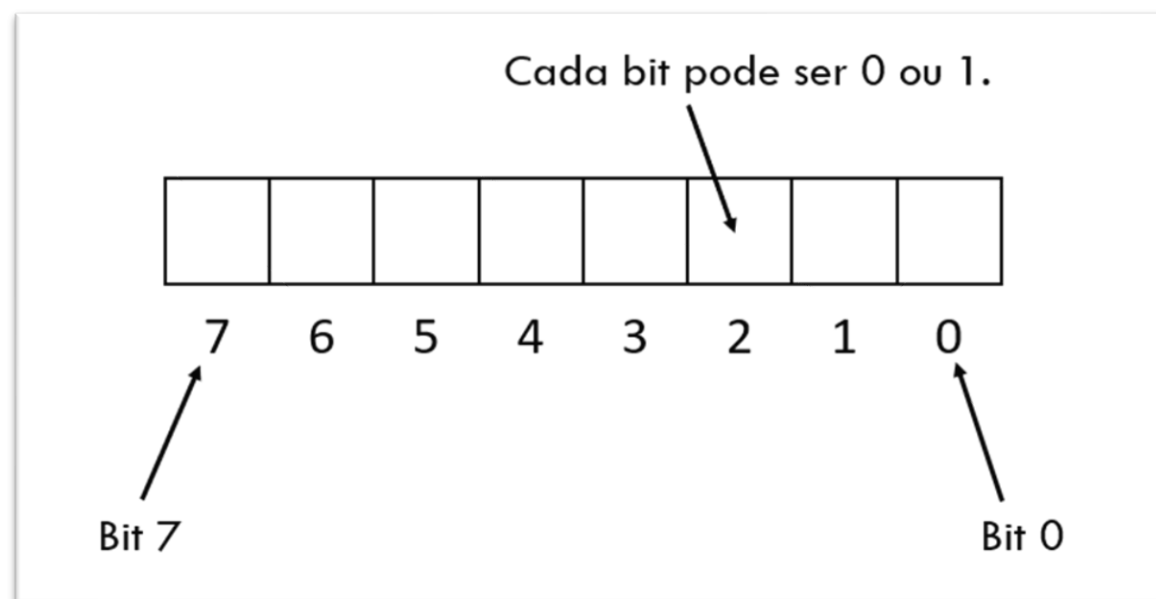
Usar `pinMode()` e `digitalWrite()` para controlar LEDs requer uma linha de código por pino para programar animações de circuitos com muitos LEDs.



# SHOW DE LEDS

## TINKERCAD

Cada registo tem 8 bits:



Conte os bits da direita para a esquerda, começando em 0.

# SHOW DE LEDS

## TINKERCAD

Os bits no registo correspondem a pinos físicos no Arduino:



Bit 7

Bit 6

Bit 5

Bit 4

Bit 3

Bit 2

Bit 1

Bit 0

O registo DDRD substitui o comando `pinMode()`.

Definir um pouco como 1 define o pino como uma saída.

Definir um pouco como 0 define o pino como uma entrada.

Por exemplo:

```
DDRD = 0b11111111;
```

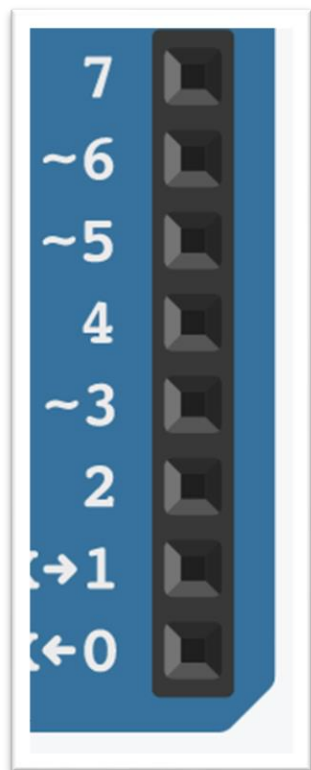
Esta linha de código define pinos de 0 a 7 como saídas.

Nota: o "0b" diz ao Arduino que este número está em binário.

# SHOW DE LEDS

## TINKERCAD

O registro PORTD substitui o comando digitalWrite().  
Definir um pouco para 1 define o pino HIGH.  
Definir um pouco como 0 define o pino BAIXO.



Por exemplo:

```
PORTD = 0b10000000;
```

Esta linha de código define o pino 7 alto e todos os outros pinos baixos.

# SHOW DE LEDS

## TINKERCAD

Agora você pode reduzir oito linhas de código em uma única linha!

```
pinMode(0, OUTPUT);  
pinMode(1, OUTPUT);  
pinMode(2, OUTPUT);  
pinMode(3, OUTPUT);  
pinMode(4, OUTPUT);  
pinMode(5, OUTPUT);  
pinMode(6, OUTPUT);  
pinMode(7, OUTPUT);
```

→ `DDRD = 0b11111111;`

```
digitalWrite(0, HIGH);  
digitalWrite(1, LOW);  
digitalWrite(2, LOW);  
digitalWrite(3, LOW);  
digitalWrite(4, LOW);  
digitalWrite(5, LOW);  
digitalWrite(6, LOW);  
digitalWrite(7, LOW);
```

→ `PORTD = 0b10000000;`



# SHOW DE LEDS

## TINKERCAD

Eles são controlados pelos registros DDRB e PORTB.

DDRB: definir o modo de pino (entrada ou saída)

PORTB: definir pino alto ou baixo se for uma saída



**Atenção:** os números de bits nos registos não correspondem aos números de pinos do Arduino! Isso pode ser confuso, por isso é mais fácil trabalhar com pinos de 0 a 7 quando você está aprendendo a usar registradores pela primeira vez.

Por exemplo, essas duas linhas de código definiriam todos os pinos como saídas e, em seguida, definiriam o pino 13 (bit 5) de altura:

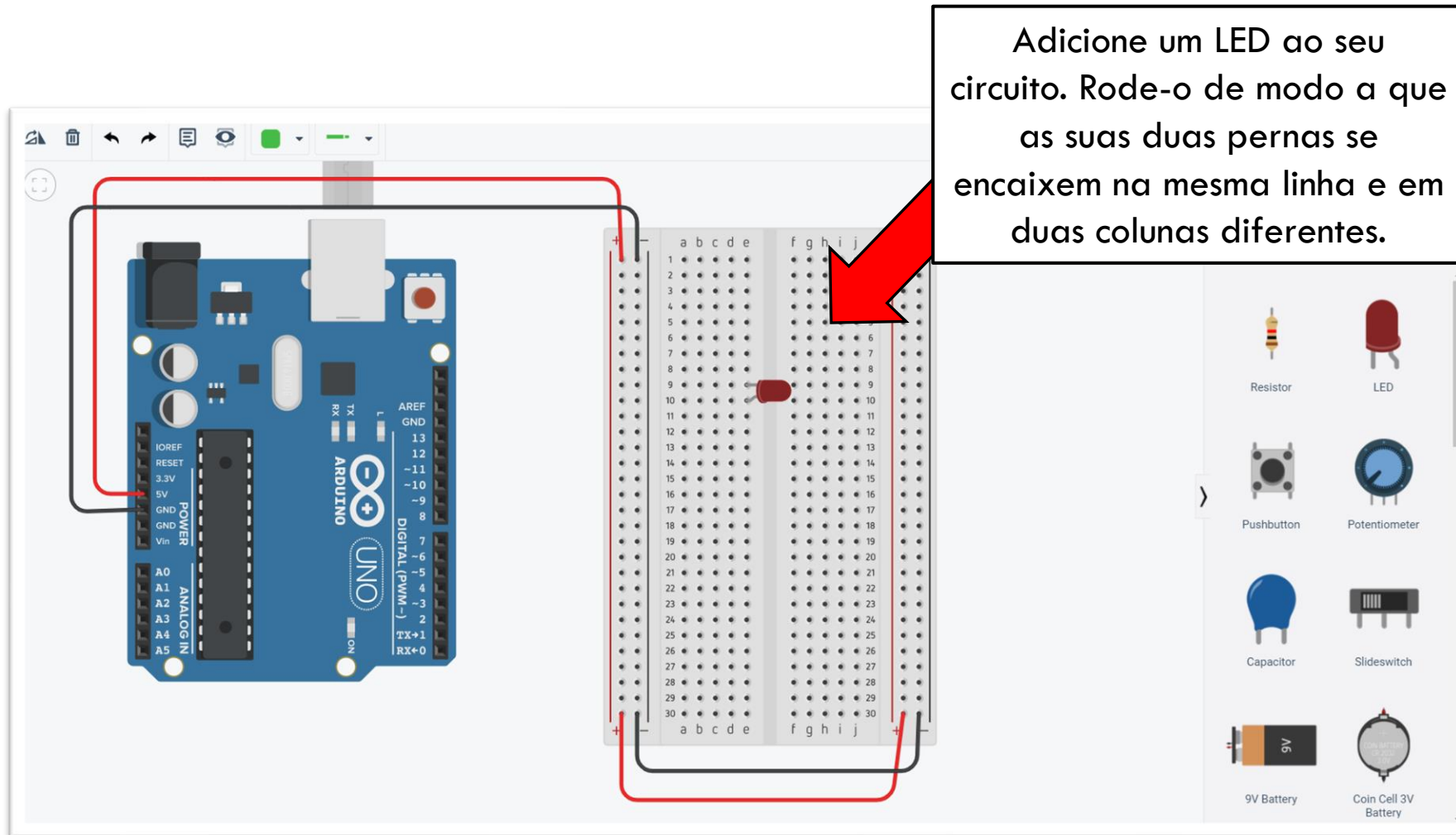
```
DDRB = 0b00111111;
```

```
PORTB = 0b00100000;
```



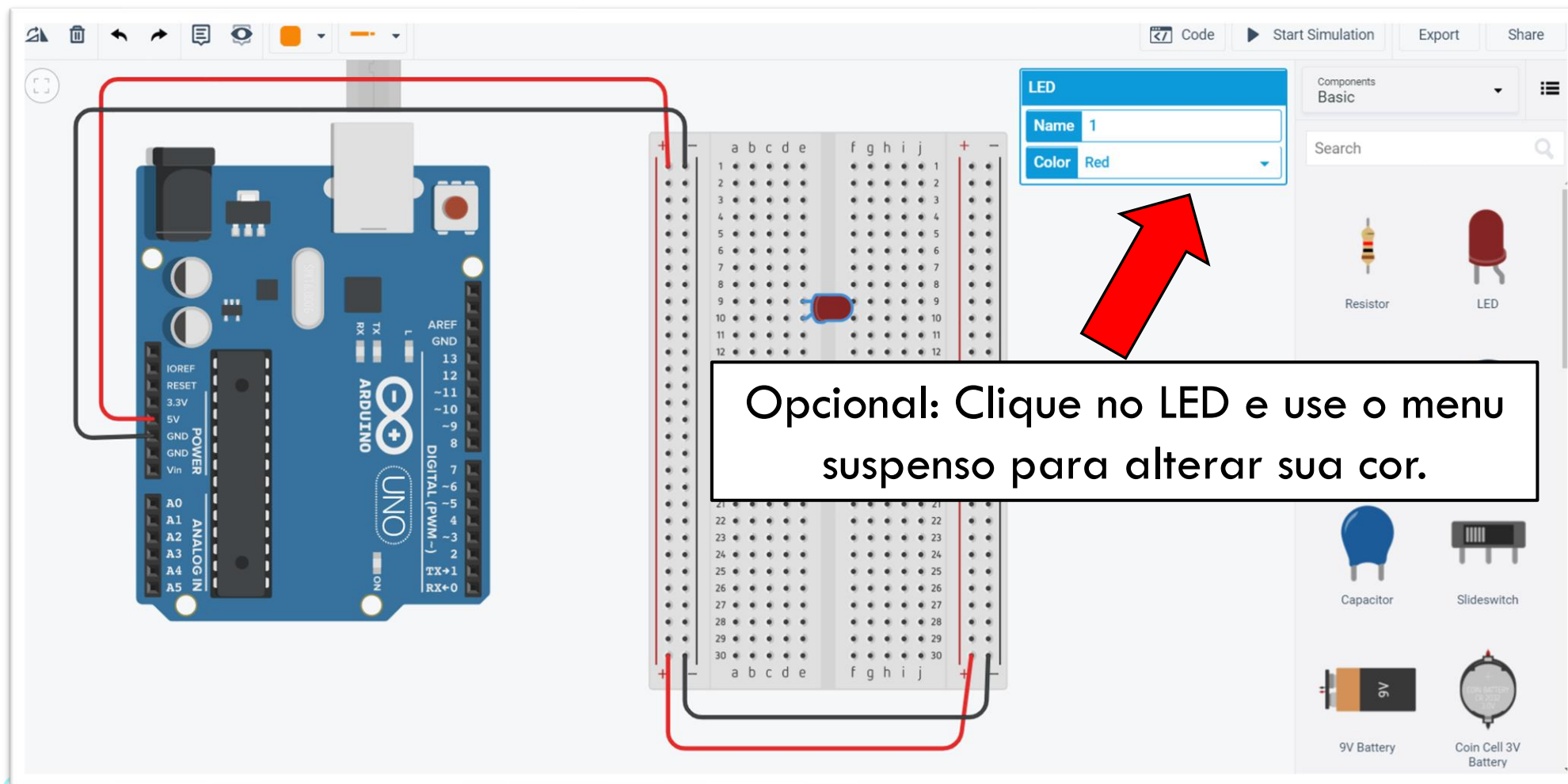
# SHOW DE LEDS

## TINKERCAD



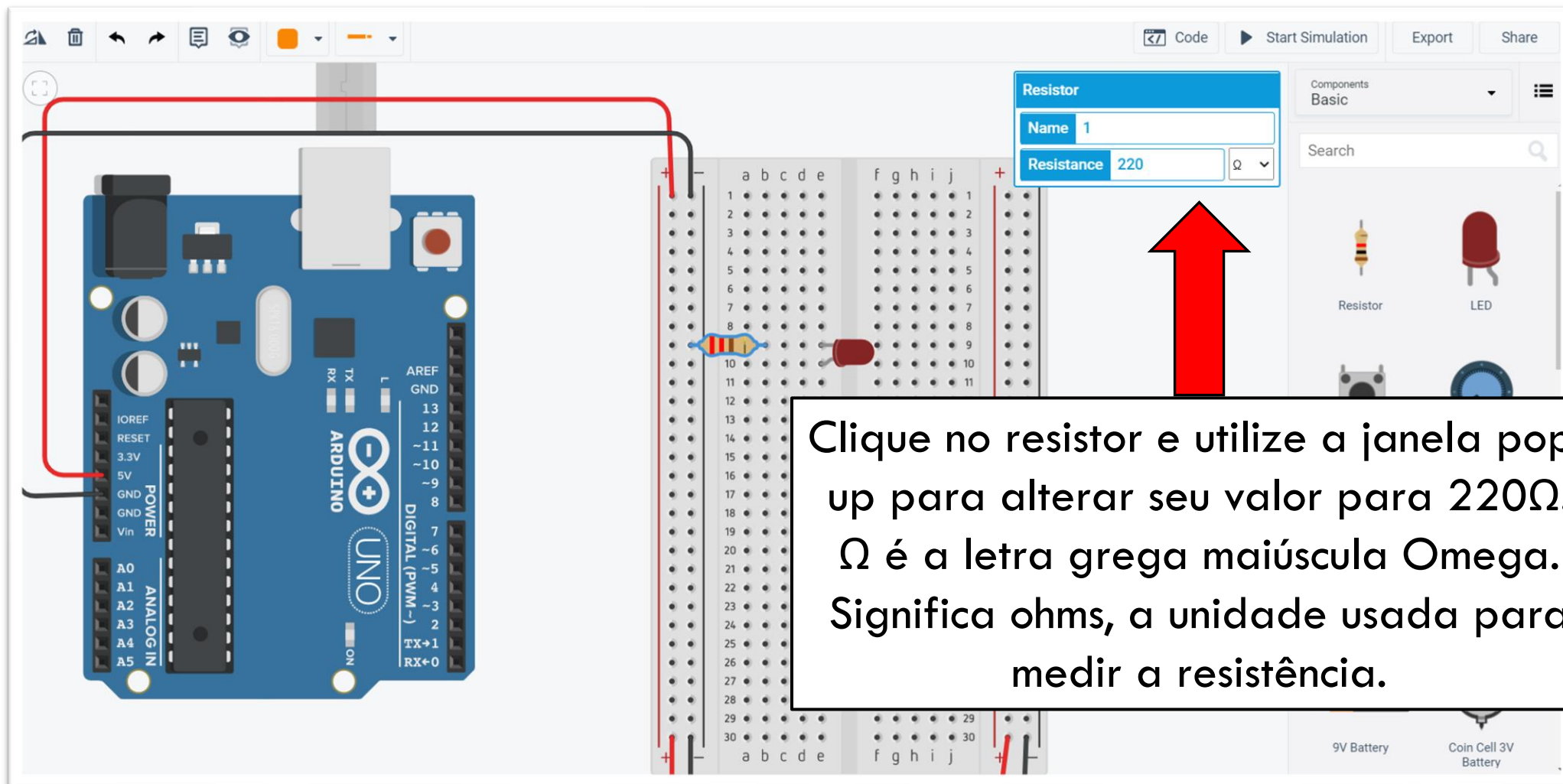
# SHOW DE LEDS

## TINKERCAD



# SHOW DE LEDS

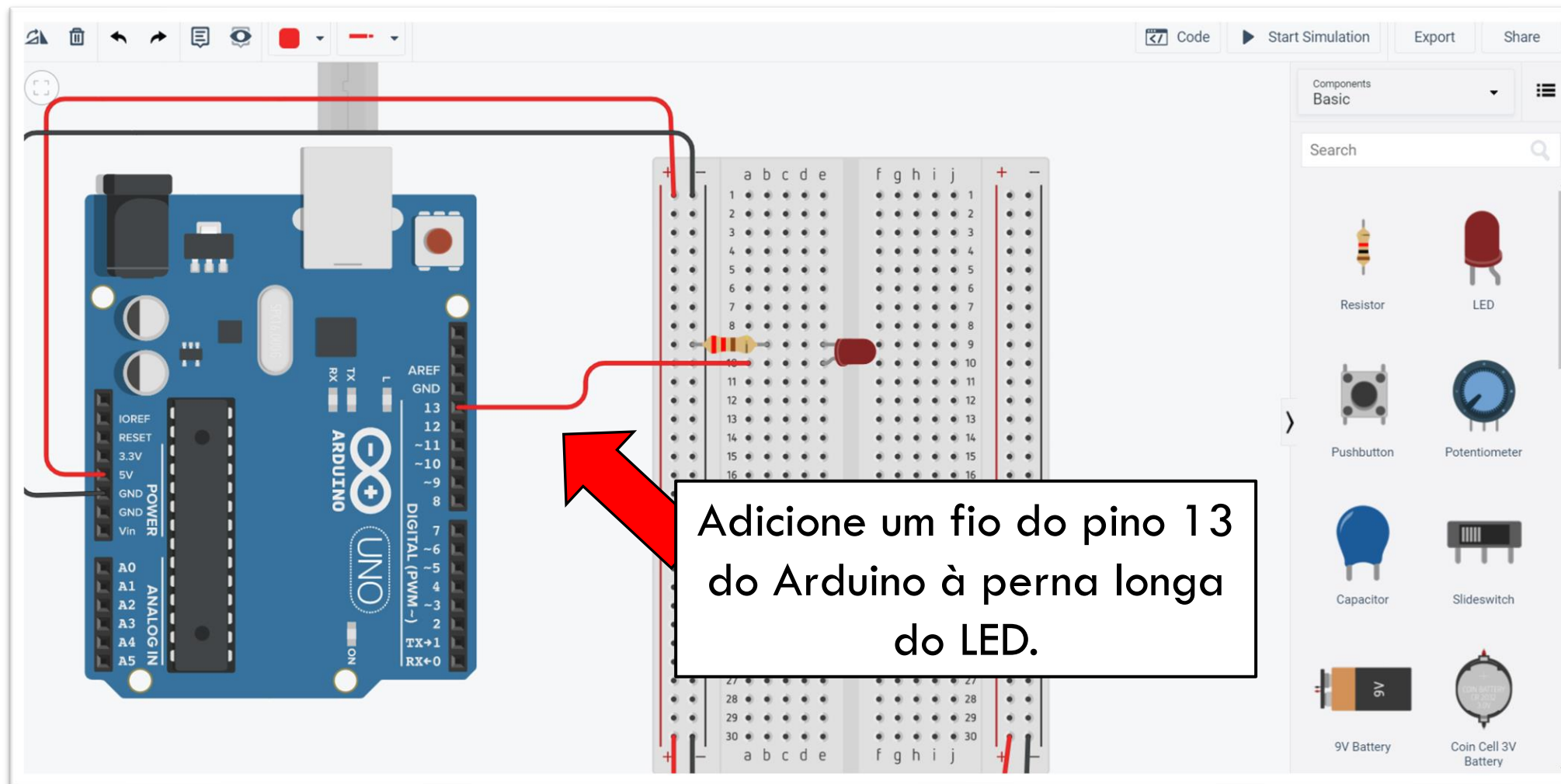
## TINKERCAD



Clique no resistor e utilize a janela pop-up para alterar seu valor para 220Ω. Ω é a letra grega maiúscula Omega. Significa ohms, a unidade usada para medir a resistência.

# SHOW DE LEDS

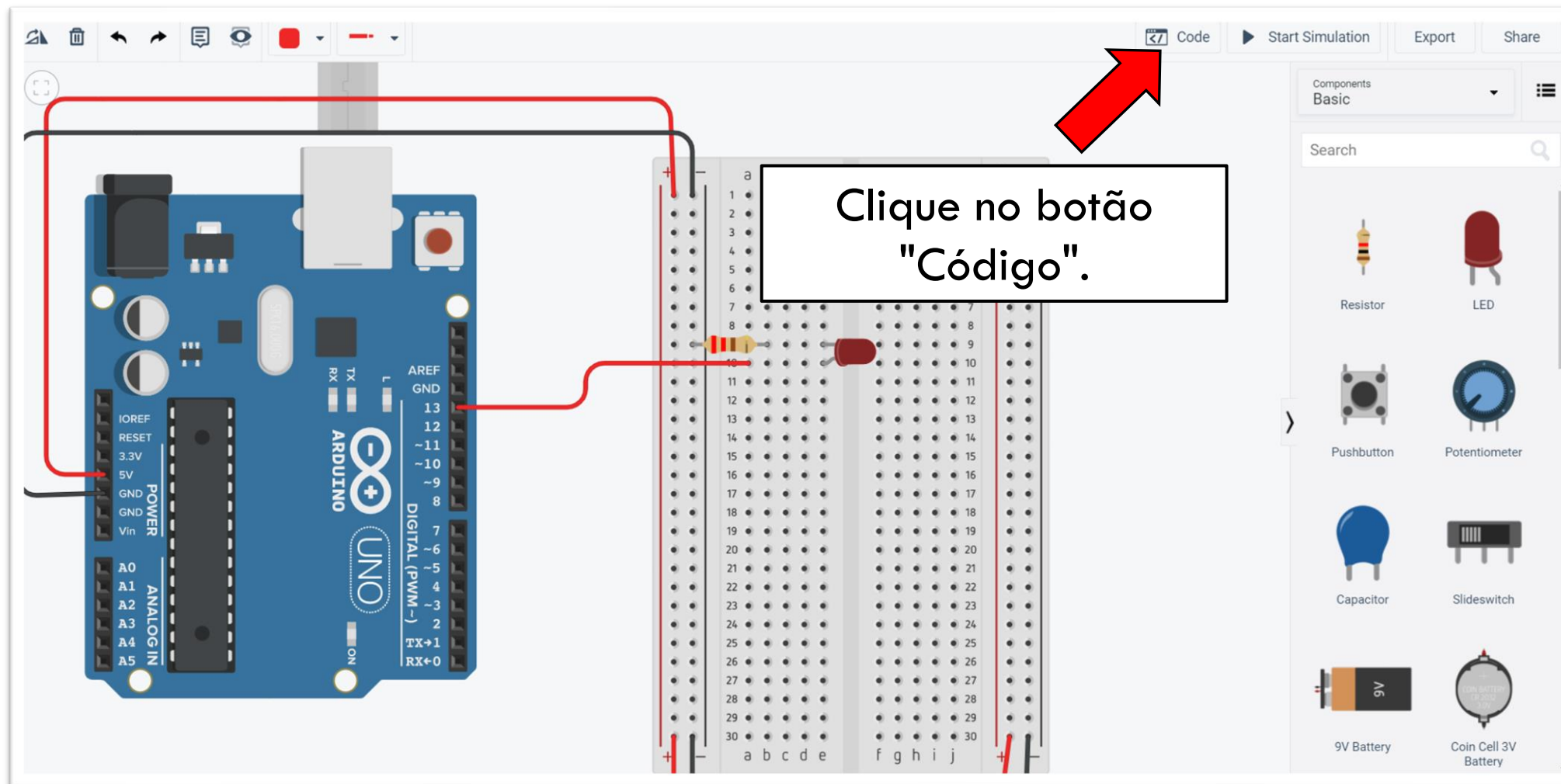
## TINKERCAD





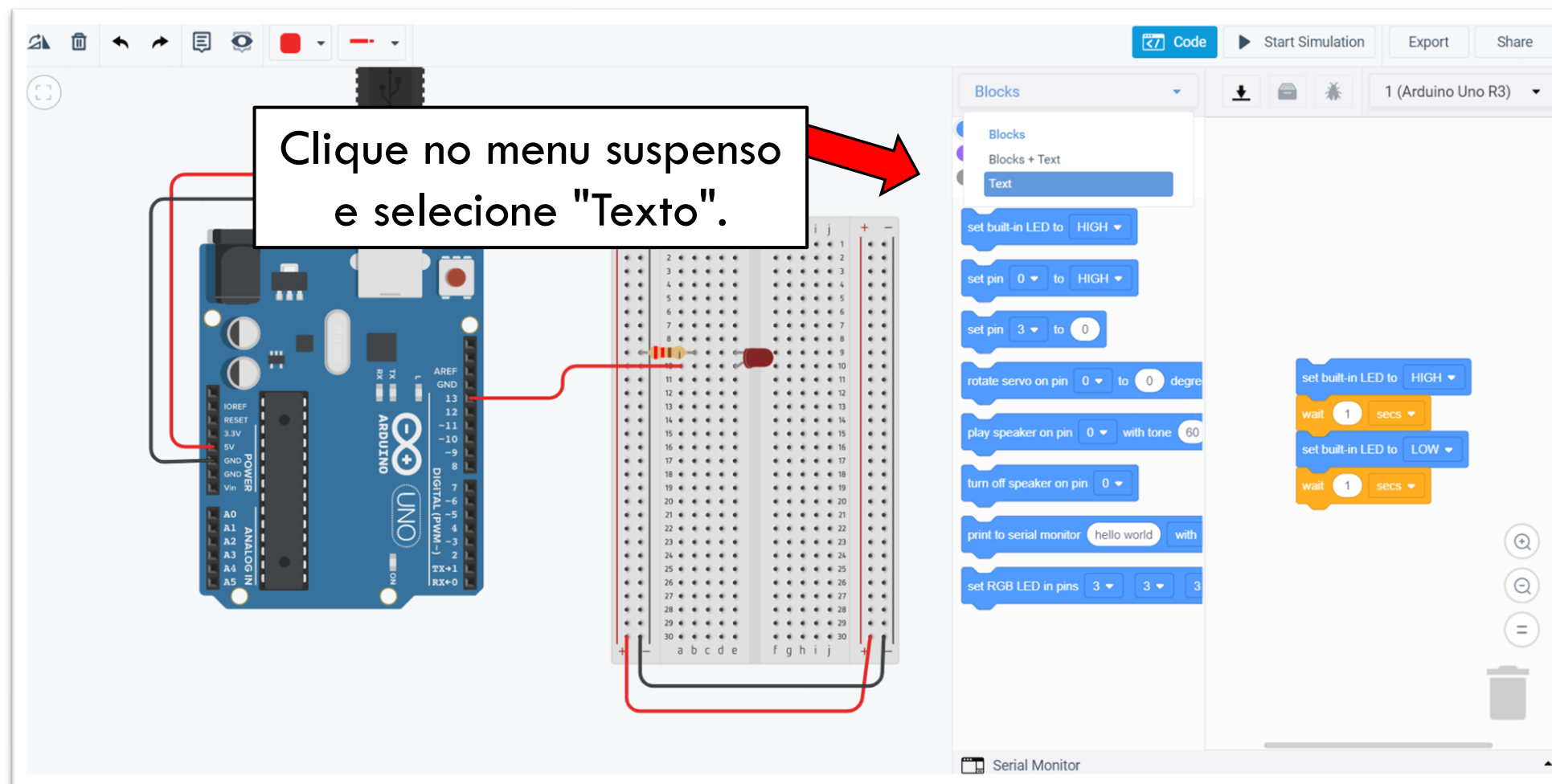
# SHOW DE LEDS

## TINKERCAD



# SHOW DE LEDS

## TINKERCAD



# SHOW DE LEDS

## TINKERCAD

O código na função de configuração só é executado uma vez. Ele usa o comando `pinMode` para dizer ao Arduino para usar o pino 13 como saída.

O código na função `loop()` é executado repetidamente para sempre. Ele usa o comando `digitalWrite` para ligar ou desligar o LED (alto ou baixo) e o comando `delay` para fazer o código aguardar por um determinado período de tempo. Este código diz ao Arduino para ligar o LED, esperar um segundo (1000 milissegundos), desligar o LED, esperar um segundo e repetir.

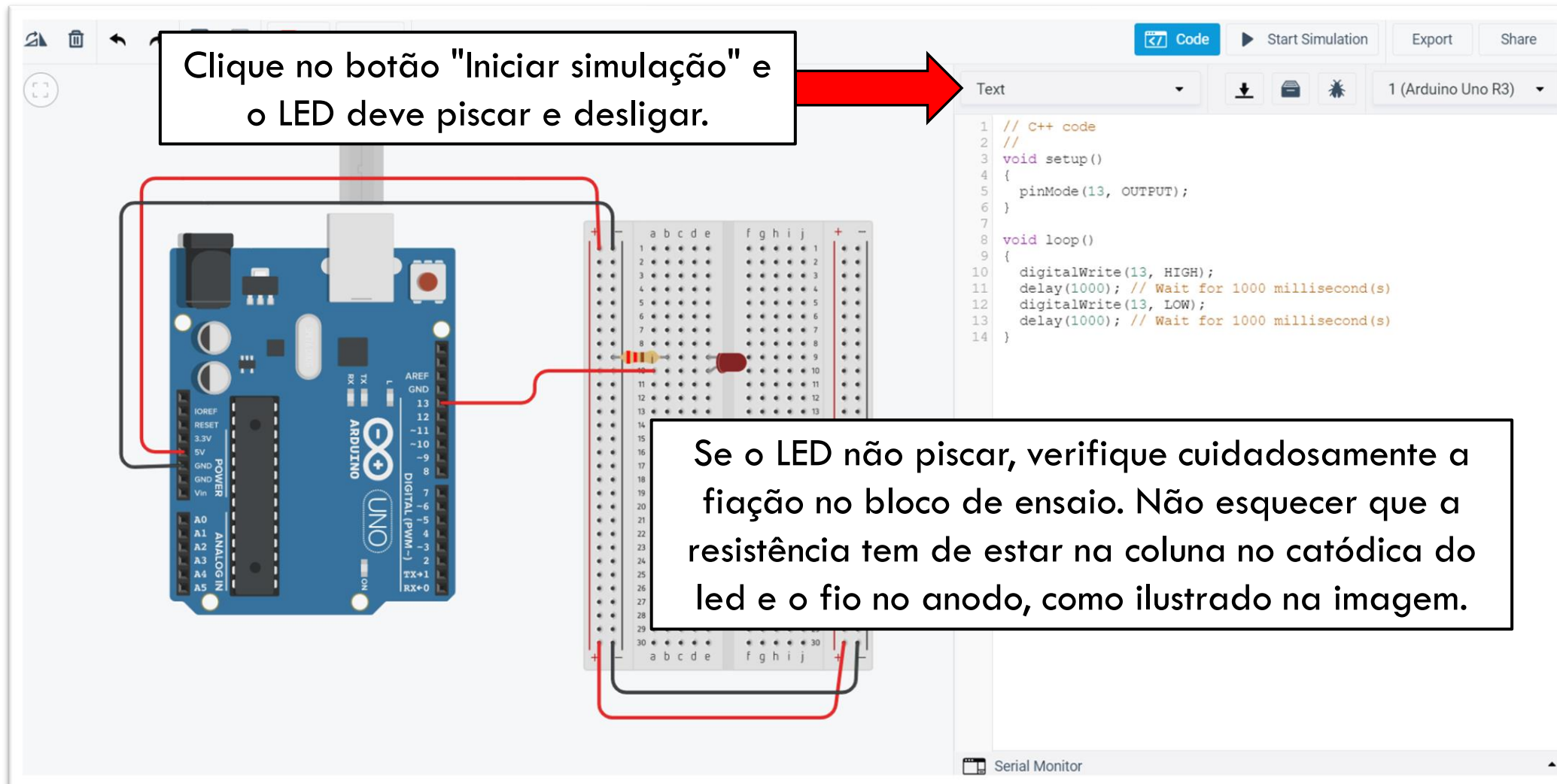
```
1 // C++ code
2 //
3 void setup()
4 {
5   pinMode(13, OUTPUT);
6 }
7
8 void loop()
9 {
10  digitalWrite(13, HIGH);
11  delay(1000); // Wait for 1000 millisecond(s)
12  digitalWrite(13, LOW);
13  delay(1000); // Wait for 1000 millisecond(s)
14 }
```



# SHOW DE LEDS

## TINKERCAD

Clique no botão "Iniciar simulação" e o LED deve piscar e desligar.



```
// C++ code
//
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

Se o LED não piscar, verifique cuidadosamente a fiação no bloco de ensaio. Não esquecer que a resistência tem de estar na coluna no catódica do led e o fio no anodo, como ilustrado na imagem.

# SHOW DE LEDS

## TINKERCAD

**Com os conhecimentos adquiridos, programe o seu próprio show de luzes LED.**

Pode conectar até 14 LEDs ao seu Arduino (observe o exemplo, imagem seguinte, como os pinos começam a contar em 0).

Lembre-se:

Use o comando `pinMode` para definir pinos como saídas.

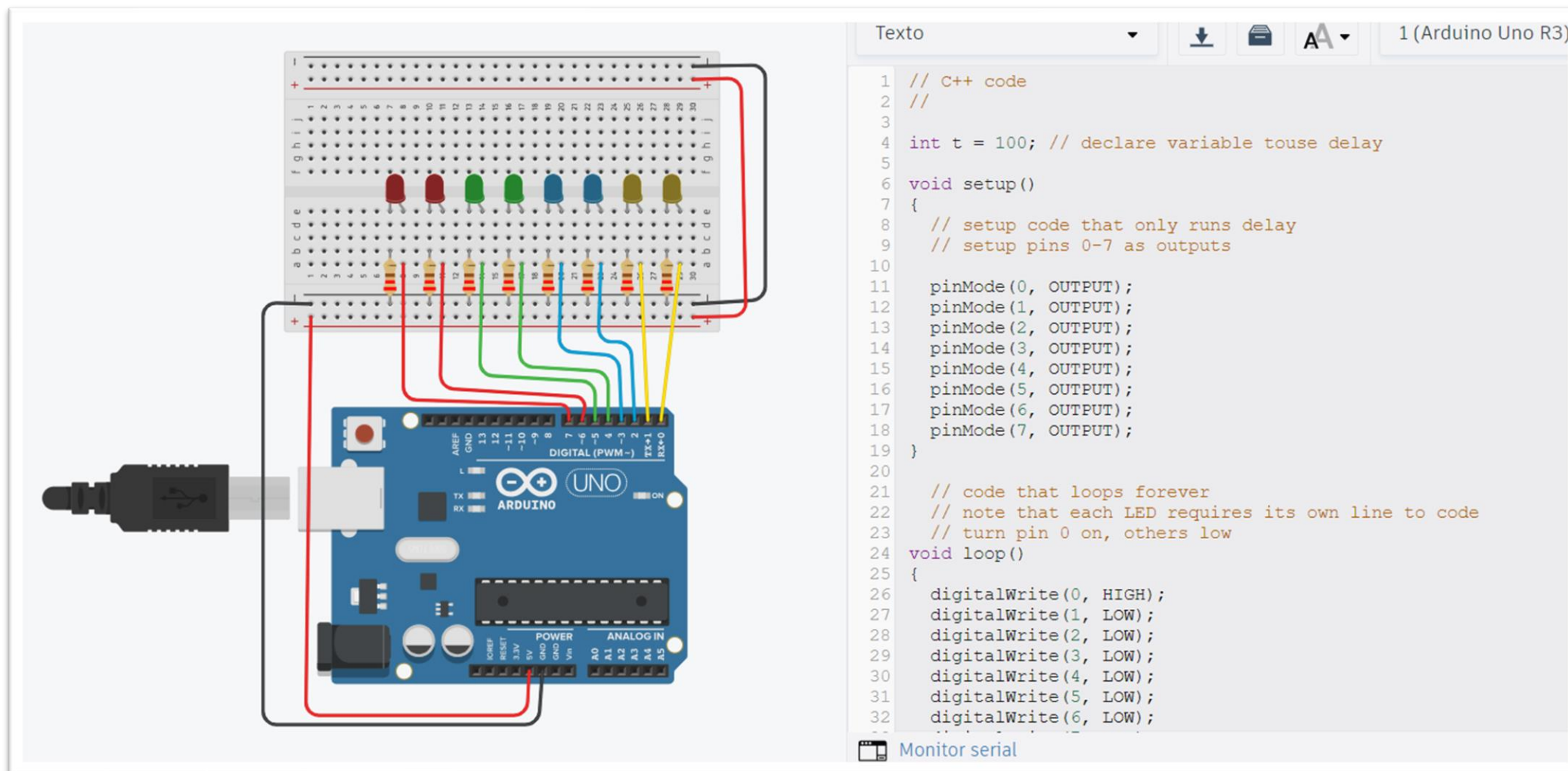
Use o comando `digitalWrite` para ativar e desativar os pinos.

Use o comando `delay` para fazer o programa esperar.

Você pode usar variáveis para facilitar o ajuste dos tempos de atraso.

# SHOW DE LEDS

## TINKERCAD



The screenshot displays the Tinkercad environment. On the left, an Arduino Uno R3 is connected to a breadboard. Eight LEDs are connected to the breadboard: two red LEDs to pins 0 and 1, two green LEDs to pins 2 and 3, two blue LEDs to pins 4 and 5, and two yellow LEDs to pins 6 and 7. The breadboard is connected to the Arduino's power pins (VCC and GND). On the right, the code editor shows the following C++ code:

```
1 // C++ code
2 //
3
4 int t = 100; // declare variable touse delay
5
6 void setup()
7 {
8   // setup code that only runs delay
9   // setup pins 0-7 as outputs
10
11   pinMode(0, OUTPUT);
12   pinMode(1, OUTPUT);
13   pinMode(2, OUTPUT);
14   pinMode(3, OUTPUT);
15   pinMode(4, OUTPUT);
16   pinMode(5, OUTPUT);
17   pinMode(6, OUTPUT);
18   pinMode(7, OUTPUT);
19 }
20
21 // code that loops forever
22 // note that each LED requires its own line to code
23 // turn pin 0 on, others low
24 void loop()
25 {
26   digitalWrite(0, HIGH);
27   digitalWrite(1, LOW);
28   digitalWrite(2, LOW);
29   digitalWrite(3, LOW);
30   digitalWrite(4, LOW);
31   digitalWrite(5, LOW);
32   digitalWrite(6, LOW);
```

Below the code editor, the 'Monitor serial' button is visible.

# SHOW DE LEDS

## TINKERCAD

```
33 digitalWrite(7, LOW);
34 delay(t); // Wait for t millisecond(s)
35
36 digitalWrite(0, LOW);
37 digitalWrite(1, HIGH);
38 digitalWrite(2, LOW);
39 digitalWrite(3, LOW);
40 digitalWrite(4, LOW);
41 digitalWrite(5, LOW);
42 digitalWrite(6, LOW);
43 digitalWrite(7, LOW);
44 delay(t); // Wait for t millisecond(s)
45
46 digitalWrite(0, LOW);
47 digitalWrite(1, LOW);
48 digitalWrite(2, HIGH);
49 digitalWrite(3, LOW);
50 digitalWrite(4, LOW);
51 digitalWrite(5, LOW);
52 digitalWrite(6, LOW);
53 digitalWrite(7, LOW);
54 delay(t); // Wait for t millisecond(s));
55
56 digitalWrite(0, LOW);
57 digitalWrite(1, LOW);
58 digitalWrite(2, LOW);
59 digitalWrite(3, HIGH);
60 digitalWrite(4, LOW);
61 digitalWrite(5, LOW);
62 digitalWrite(6, LOW);
63 digitalWrite(7, LOW);
```



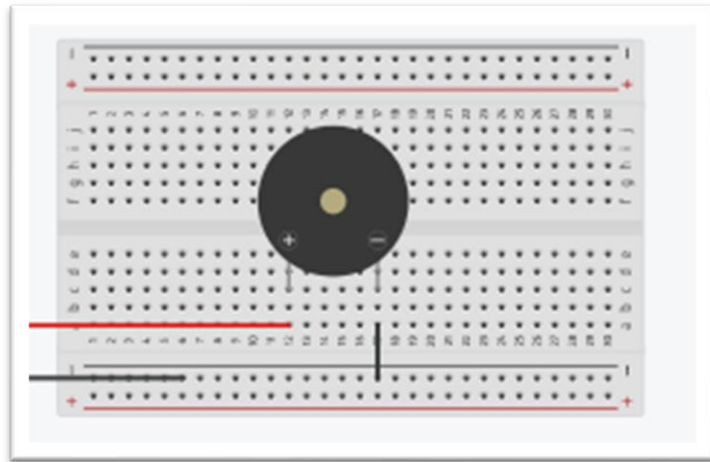
E assim, sucessivamente.

# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MÍ FÁ

### TINKERCAD

Introduzir o buzzer na placa de ensaio e as frequências das notas musicais em Hz.



```
Texto [v] [Download] [Save] [Font] 1 (Arduino Uno R3) [v]
1 // C++ code
2 //
3
4 int t= 300; // declare variable
5 int buzzer = 8; //Atribui o valor 8 a variável buzzer, que repre
6
7 // Frequências das notas musicais (em Hz)
8 const int notaDo = 262; // Dó
9 const int notaRe = 294; // Ré
10 const int notaMi = 330; // Mi
11 const int notaFa = 349; // Fá
12 const int notaSol = 392; // Sol
13
```

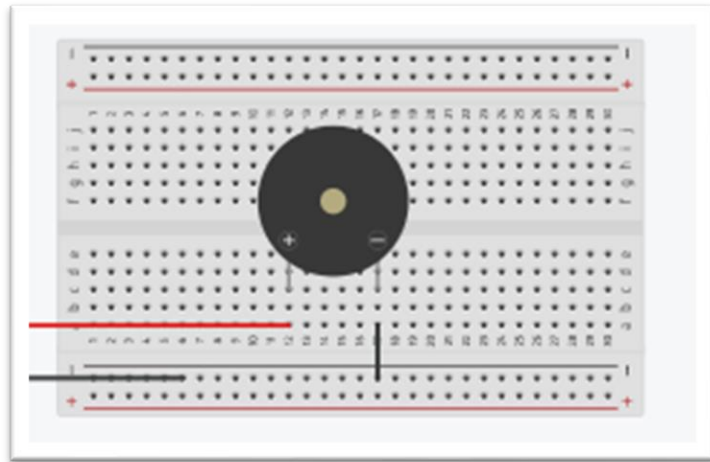


# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MÍ FÁ

### TINKERCAD

A função void setup determina a inicialização do programa e a configuração dos pinos da placa Arduino



```

13
14 void setup()
15     // setup code that only runs delay
16     // setup pins 0 a 7 as outputs
17 {
18
19     pinMode (0, OUTPUT);
20     pinMode (1, OUTPUT);
21     pinMode (2, OUTPUT);
22     pinMode (3, OUTPUT);
23     pinMode (4, OUTPUT);
24     pinMode (5, OUTPUT);
25     pinMode (6, OUTPUT);
26     pinMode (7, OUTPUT);
27
28     pinMode(buzzer, OUTPUT); //Definindo o pino buzzer como de saída
29
30     // Frequências das notas musicais (em Hz)
31     const int notaDo = 262; // Dó
32     const int notaRe = 294; // Ré
33     const int notaMi = 330; // Mi
34     const int notaFa = 349; // Fá
35     const int notaSol = 392; // Sol
36 }

```

# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MI FÁ

### TINKERCAD

A função void loop é responsável em executar os comandos para a realização das tarefas, repetindo-se enquanto o Arduino estiver ligado.

```
38 void loop()
39 {
40   digitalWrite(0, HIGH);
41   digitalWrite(1, LOW);
42   digitalWrite(2, LOW);
43   digitalWrite(3, LOW);
44   digitalWrite(4, LOW);
45   digitalWrite(5, LOW);
46   digitalWrite(6, LOW);
47   digitalWrite(7, LOW);
48   delay(t); // Wait for t millisecond(s)
49
50   digitalWrite(0, LOW);
51   digitalWrite(1, HIGH);
52   digitalWrite(2, LOW);
53   digitalWrite(3, LOW);
54   digitalWrite(4, LOW);
55   digitalWrite(5, LOW);
56   digitalWrite(6, LOW);
57   digitalWrite(7, LOW);
58   delay(t); // Wait for t millisecond(s)
```

...



# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MÍ FÁ

### TINKERCAD

Frequência e duração das notas musicais definidas na função void loop () após definição dos show de LEDs.

```
Texto [v] [Download] [Save] [Font Size: A] [1 (Arduino Uno R3)]
120 //Dó
121 tone(buzzer, notaDo, 300); //Frequência e duração da nota Dó
122 delay(t); //Intervalo de 200 milissegundos
123 noTone(buzzer);
124
125 //Ré
126 tone(buzzer, notaRe, 300); //Frequência e duração da nota Ré
127 delay(t); //Intervalo de 200 milissegundos
128 noTone(buzzer);
129
130 //Mi
131 tone(buzzer, notaMi, 300); //Frequência e duração da nota Mi
132 delay(t); //Intervalo de 200 milissegundos
133 noTone(buzzer);
134
135 //Fá
136 tone(buzzer, notaFa, 300); //Frequência e duração da nota Fá
137 delay(t); //Intervalo de 200 milissegundos
138 noTone(buzzer);
139
140 //Fá Fá
141 for (int a = 0; a < 2; a++) {
142     tone(buzzer, notaFa, 200); //Frequência e duração da nota Fá
143     delay(t); //Intervalo de 300 milissegundos
144     noTone(buzzer);
145 }
146
147 //Dó
148 tone(buzzer, notaDo, 300); //Frequência e duração da nota Dó
149 delay(t); //Intervalo de 200 milissegundos
150 noTone(buzzer);
```

# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MÍ FÁ

### TINKERCAD

Frequência e duração das notas musicais definidas na função void loop () após definição dos show de LEDs.

```
Texto [v] [Download] [Save] [Font Size] 1 (Arduino Uno R3) [v]
152 //Ré
153 tone(buzzer, notaRe, 300); //Frequência e duração da nota Ré
154 delay(t); //Intervalo de 200 milissegundos
155 noTone(buzzer);
156
157 //Dó
158 tone(buzzer, notaDo, 300); //Frequência e duração da nota Dó
159 delay(t); //Intervalo de 200 milissegundos
160 noTone(buzzer);
161
162 //Ré
163 tone(buzzer, notaRe, 300); //Frequência e duração da nota Ré
164 delay(t); //Intervalo de 200 milissegundos
165 noTone(buzzer);
166
167 //Ré Ré
168 for (int b = 0; b < 2; b++) {
169     tone(buzzer, notaRe, 200); //Frequência e duração da nota Ré
170     delay(t); //Intervalo de 300 milissegundos
171     noTone(buzzer);
172 }
173
174 //Dó
175 tone(buzzer, notaDo, 300); //Frequência e duração da nota Dó
176 delay(t); //Intervalo de 200 milissegundos
177 noTone(buzzer);
178
179 //Sol
180 tone(buzzer, notaSol, 300); //Frequência e duração da nota Sol
181 delay(t); //Intervalo de 200 milissegundos
182 noTone(buzzer);
```

# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MÍ FÁ

### TINKERCAD

Frequência e duração das notas musicais definidas na função void loop () após definição dos show de LEDs.

```
Texto [v] [Download] [Save] [Font Size] 1 (Arduino Uno R3)
184 //Fa
185 tone(buzzer, notaFa, 300); //Frequência e duração da nota Fá
186 delay(t); //Intervalo de 200 milissegundos
187 noTone(buzzer);
188
189 //Mi
190 tone(buzzer, notaMi, 300); //Frequência e duração da nota Fá
191 delay(t); //Intervalo de 200 milissegundos
192 noTone(buzzer);
193
194 //Mi Mi
195 for (int c = 0; c < 2; c++) {
196     tone(buzzer, notaMi, 200); //Frequência e duração da nota
197     delay(t); //Intervalo de 300 milissegundos
198     noTone(buzzer);
199 }
200
201 //Dó
202 tone(buzzer, notaDo, 300); //Frequência e duração da nota Dó
203 delay(t); //Intervalo de 200 milissegundos
204 noTone(buzzer);
205
206 //Ré
207 tone(buzzer, notaRe, 300); //Frequência e duração da nota Ré
208 delay(t); //Intervalo de 200 milissegundos
209 noTone(buzzer);
210
211 //Mi
212 tone(buzzer, notaMi, 300); //Frequência e duração da nota Mi
213 delay(t); //Intervalo de 200 milissegundos
214 noTone(buzzer);
```

# SHOW DE LEDS COM MÚSICA

## DÓ RÉ MÍ FÁ

### TINKERCAD

Frequência e duração das notas musicais definidas na função void loop () após definição dos show de LEDs.

```
216 //fa
217 tone(buzzer, notaFa, 300); //Frequência e duração da nota Fá
218 delay(t); //Intervalo de 200 milissegundos
219 noTone(buzzer);
220
221 for (int d = 0; d < 2; d++) {
222     tone(buzzer, notaFa, 200); //Frequência e duração da nota Fá
223     delay(t); //Intervalo de 300 milissegundos
224     noTone(buzzer);
225 }
226
227 delay(t);
228
229 }
```

# SÍNTESE

- Continuação da lição anterior inserindo música no circuito

# SUGESTÕES

- Crie um parque de estacionamento.

Esta proposta educativa foi traduzida e/ou adaptada do projeto  
[Arduíno Education](#)

Link de projetos no TinKerCAD

<https://www.tinkercad.com/projects/Building-a-Simple-Electronic-Piano-Using-Tinkercad>