

Unoverse: Estudante

Bem-vindo(a)!



Monitor serial no Tinkercad

Acesse o site Tinkercad (simulador de robótica), através do link:
<https://www.tinkercad.com/>



Passo 1

Criando conta no Tinkercad



Vamos seguir os passos da seta.

Passo 2

Selecione o tipo de conta
(pessoal)

Iniciar edição

Como você usará o Tinkercad?

Na escola?

[Os educadores começam aqui](#)

[Alunos, entrem em uma turma](#)

Por conta própria

[Criar uma conta pessoal](#)

Já tem uma conta?

[Entrar](#)

[Declaração de Privacidade dos Filhos](#) [Privacy settings](#)

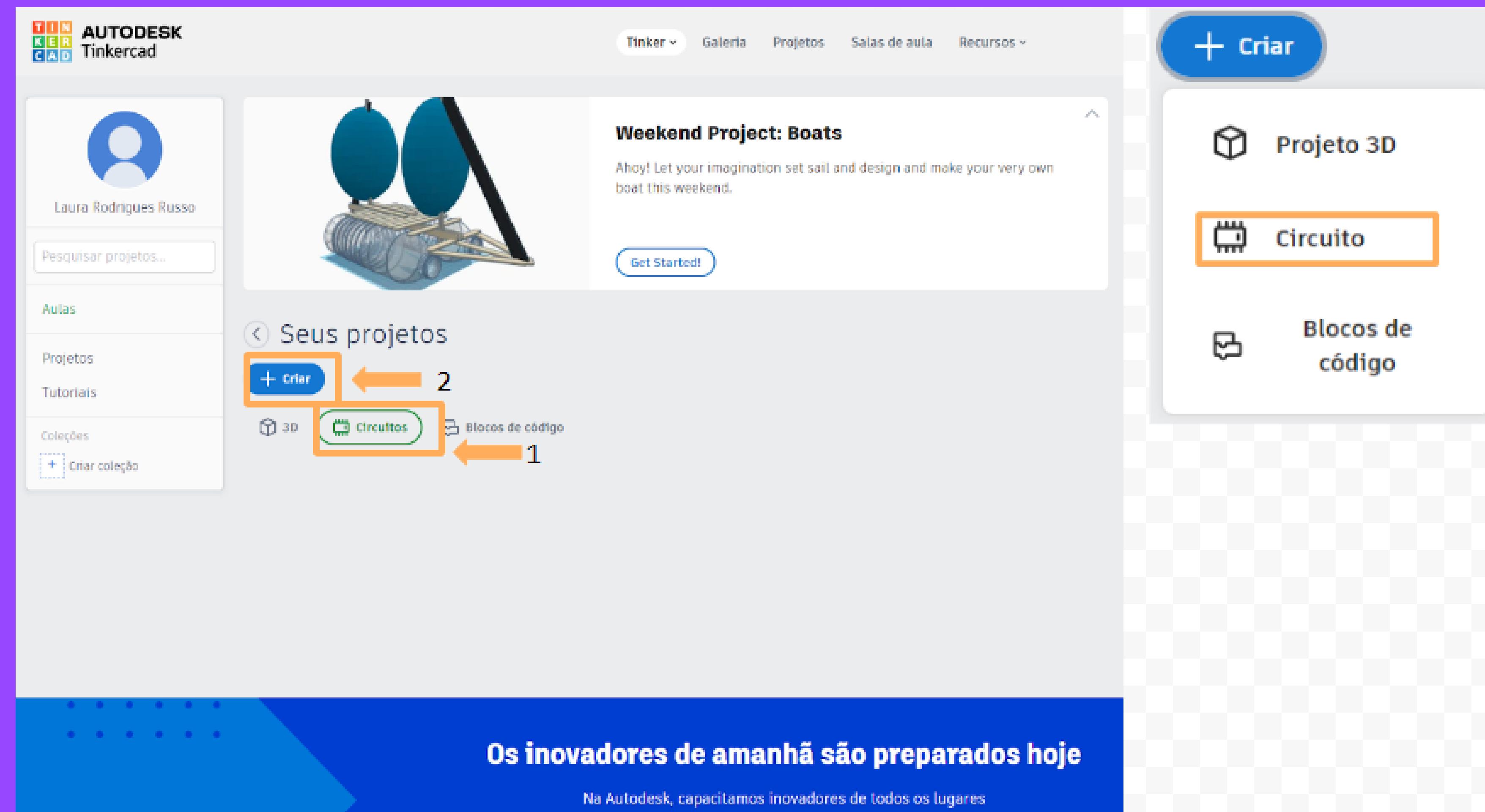
Passo 3

Escolha o tipo de conta

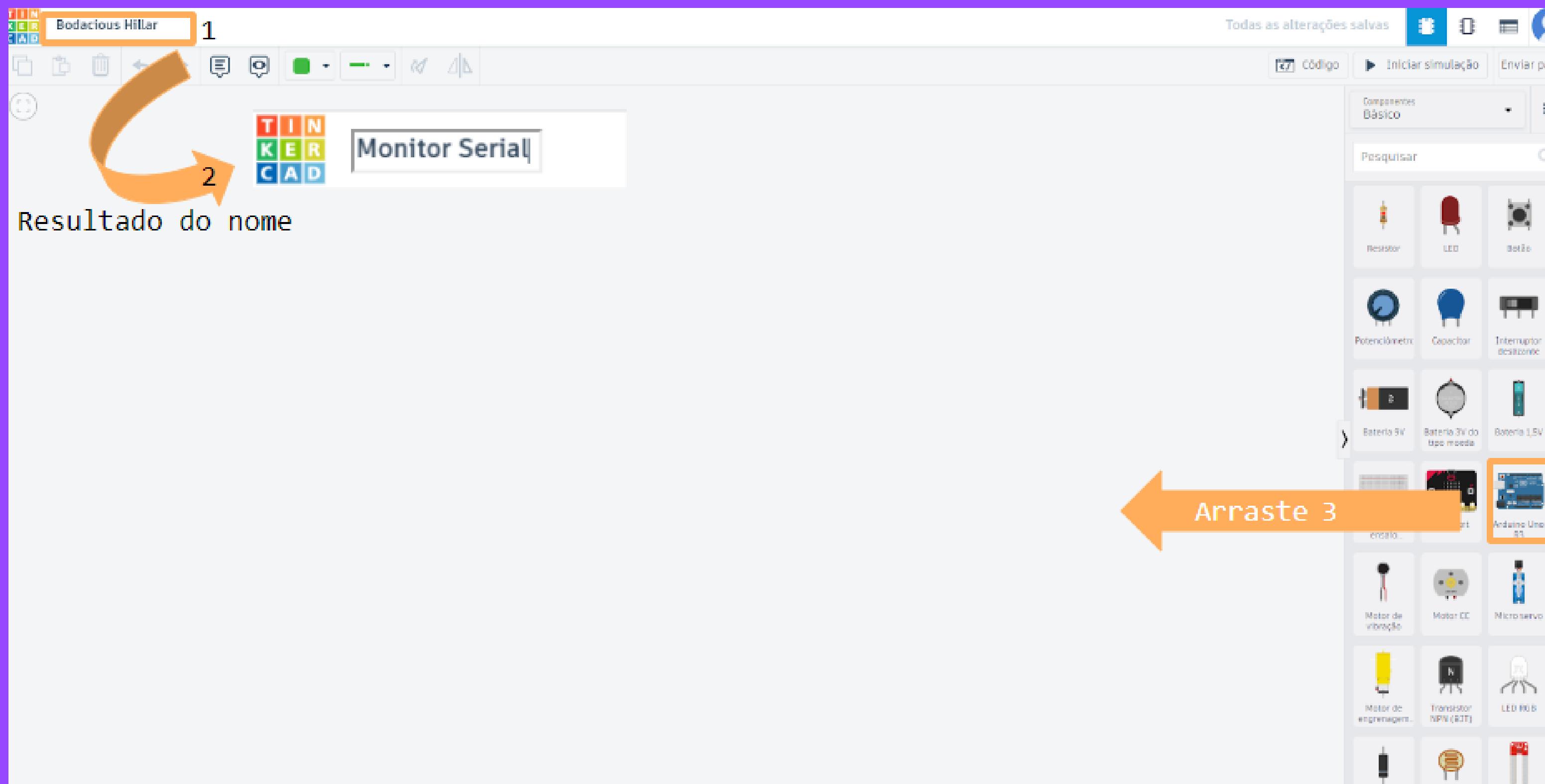


Passo 4

Selecione a opção “circuitos” e vamos começar a criar!



Passo 5

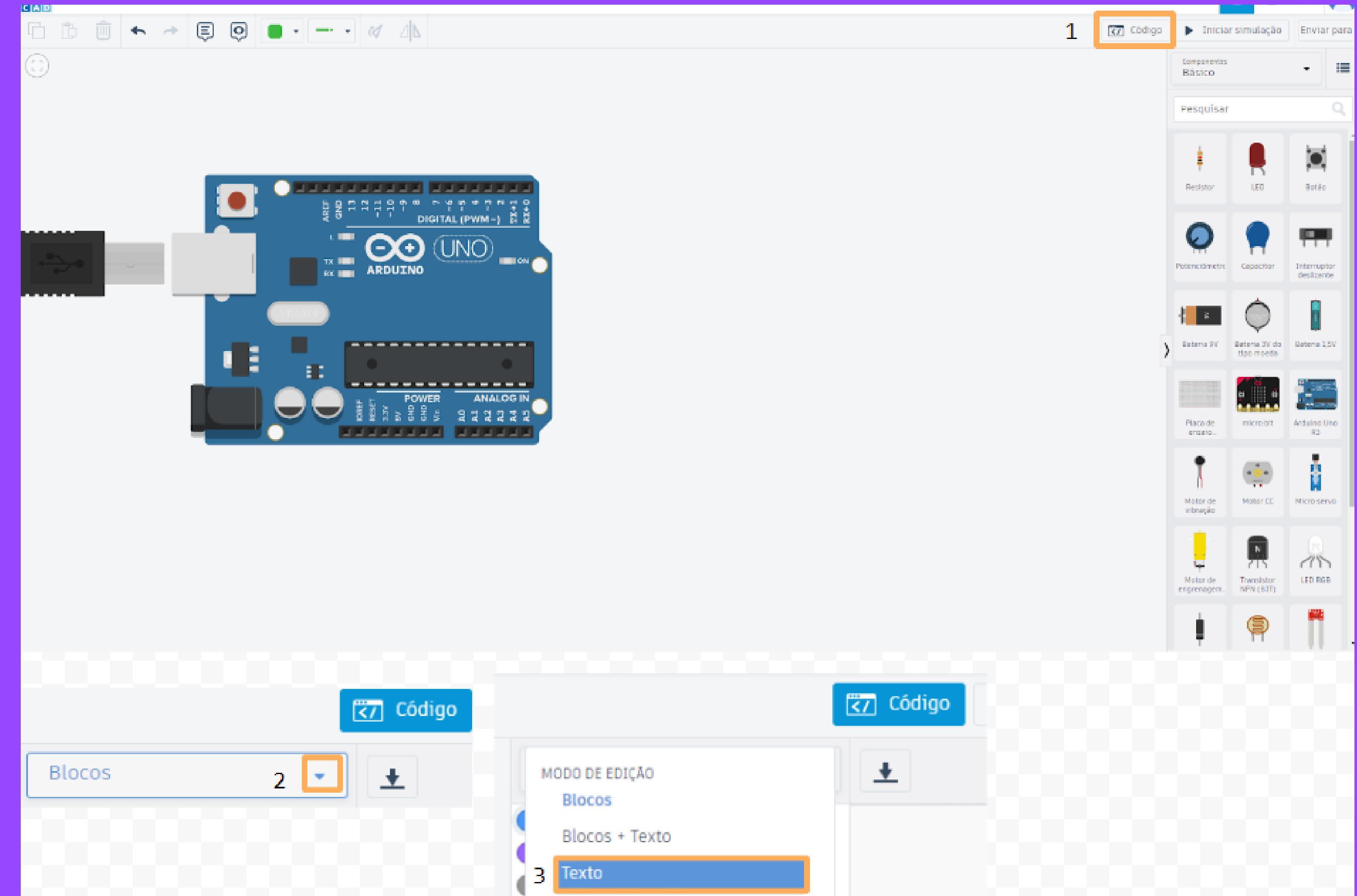


Após a abertura da tela, mude o nome do programa e arraste o componente “Arduino Uno R3” para o espaço em branco

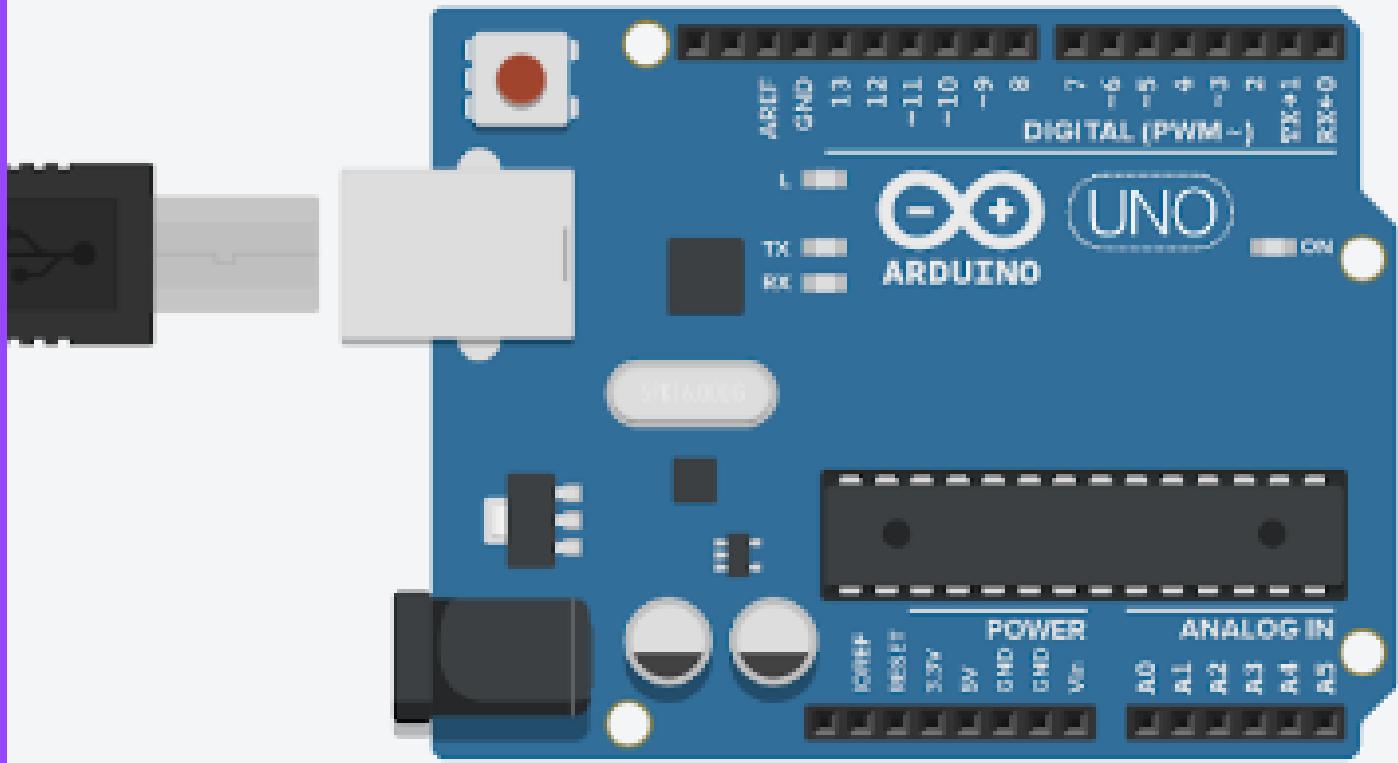
Passo 6

Com o Arduino em sua área de trabalho...

Selecione a opção de
“Código” e logo após
a opção “Texto”



Resultado



The image shows the Arduino Uno R3 board connected to a computer via a USB cable. The board is blue with various components like the ATmega328 microcontroller, capacitors, and resistors. It has several pins labeled: AREF, GND, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, TX, RX, GND, POWER, 5V, ANALOG IN, A0, A1, A2, A3, A4, A5, GND, 3.3V, IOREF, and GND.

Código | Iniciar simulação | Enviar | I (Arduino Uno R3)

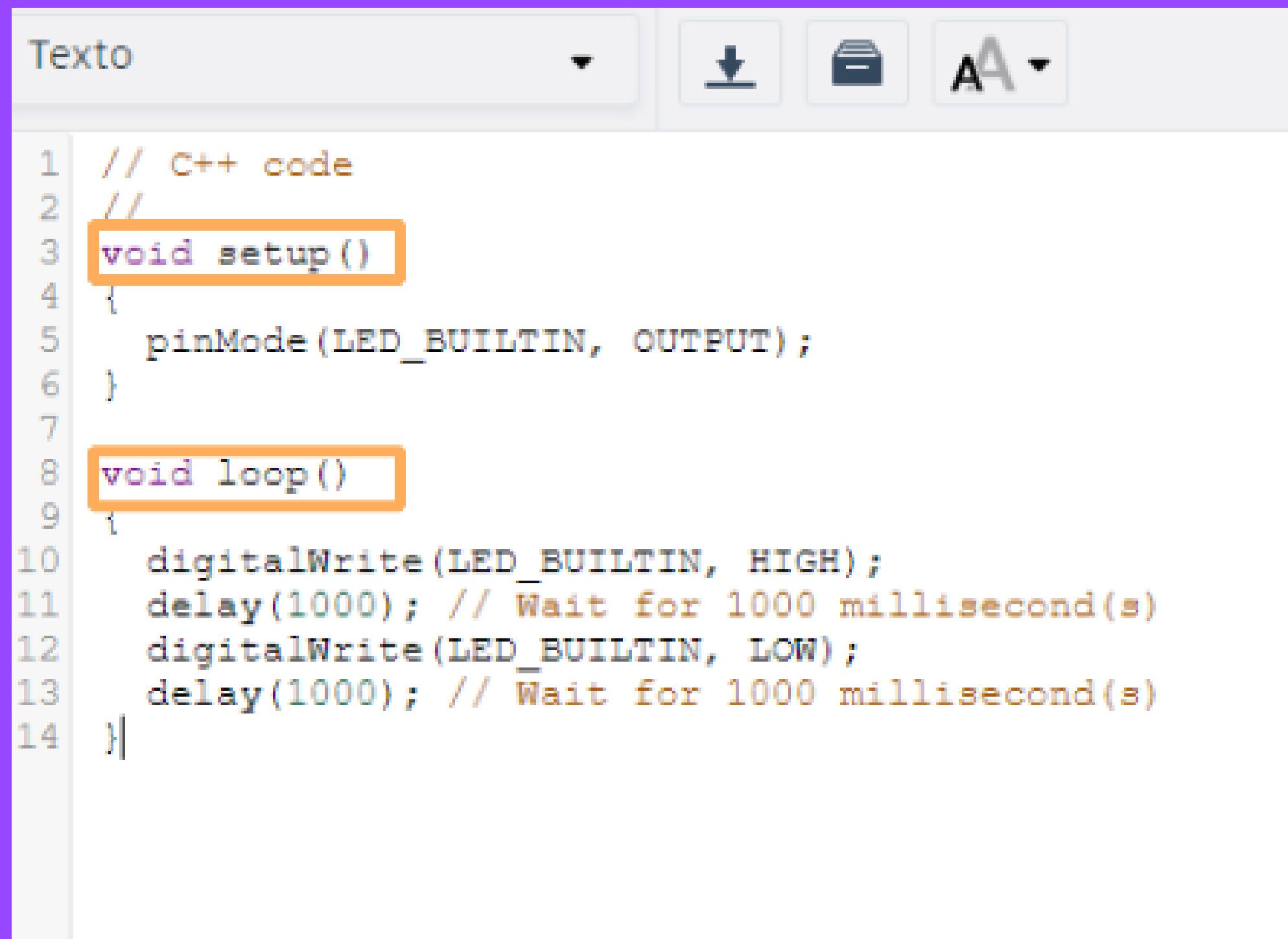
Texto

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

Monitor serial

Passo 7

Entenda o código...



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

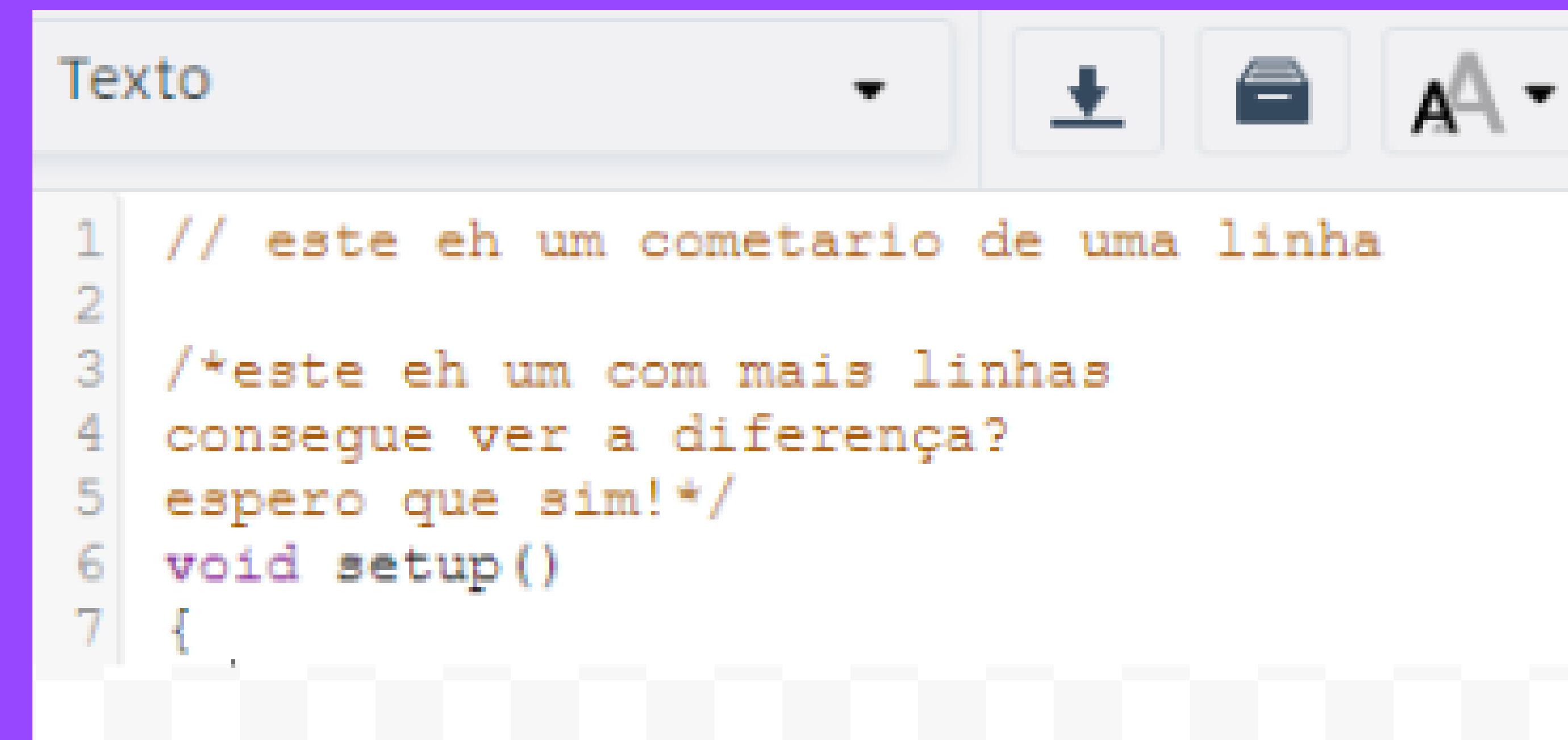
A função **setup**: Fazer a “ligação” da parte “física” (simulador) com a parte de lógica (que escreverá que o led estará ligado no pino 9 da placa, por exemplo)

A função **loop**: nela colocaremos o código principal, que irá se repetir em loop (como acender e apagar um led)

Passo 8

Para fazer comentários auxiliando o programador a se localizar, utilizamos “// comentário” para uma linha e para mais linhas usamos “/*comentário*/”. Eles ficam de uma cor única, podendo diferenciá-los do resto do programa.

Hora de fazer o código!

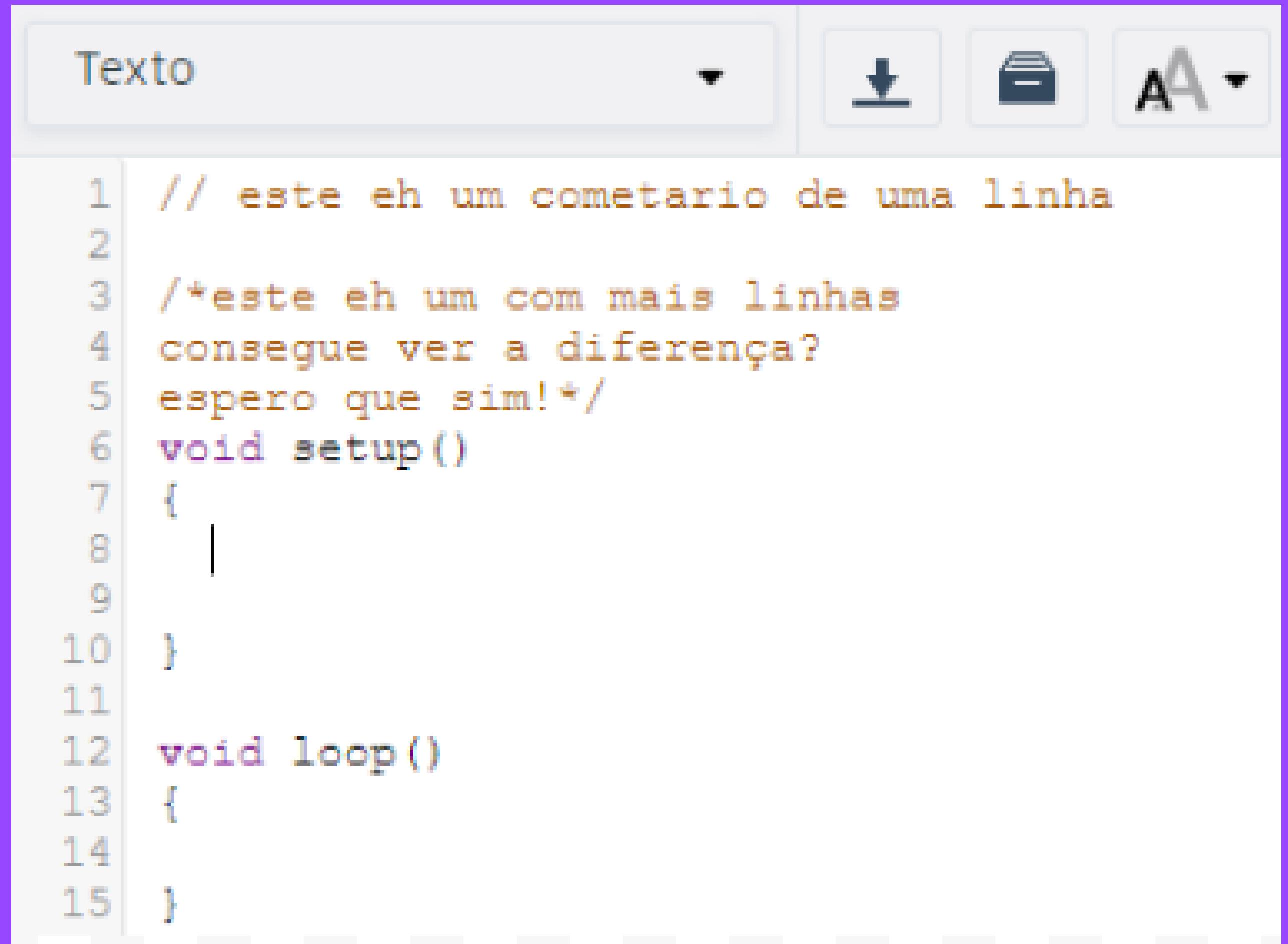


The image shows a screenshot of a text editor window titled "Texto". The editor has a toolbar with icons for download, file, and font size. The code area contains the following text:

```
1 // este eh um comentario de uma linha  
2 /*este eh um com mais linhas  
3 conseguue ver a diferenca?  
4 espero que sim!*/  
5 void setup()  
6 {  
7 }
```

Passo 8

Para isso, vamos apagar os comandos dentro do setup e do loop, deixando - o desse modo:



The image shows a screenshot of a text editor window titled "Texto". The content of the editor is a code snippet in Portuguese:

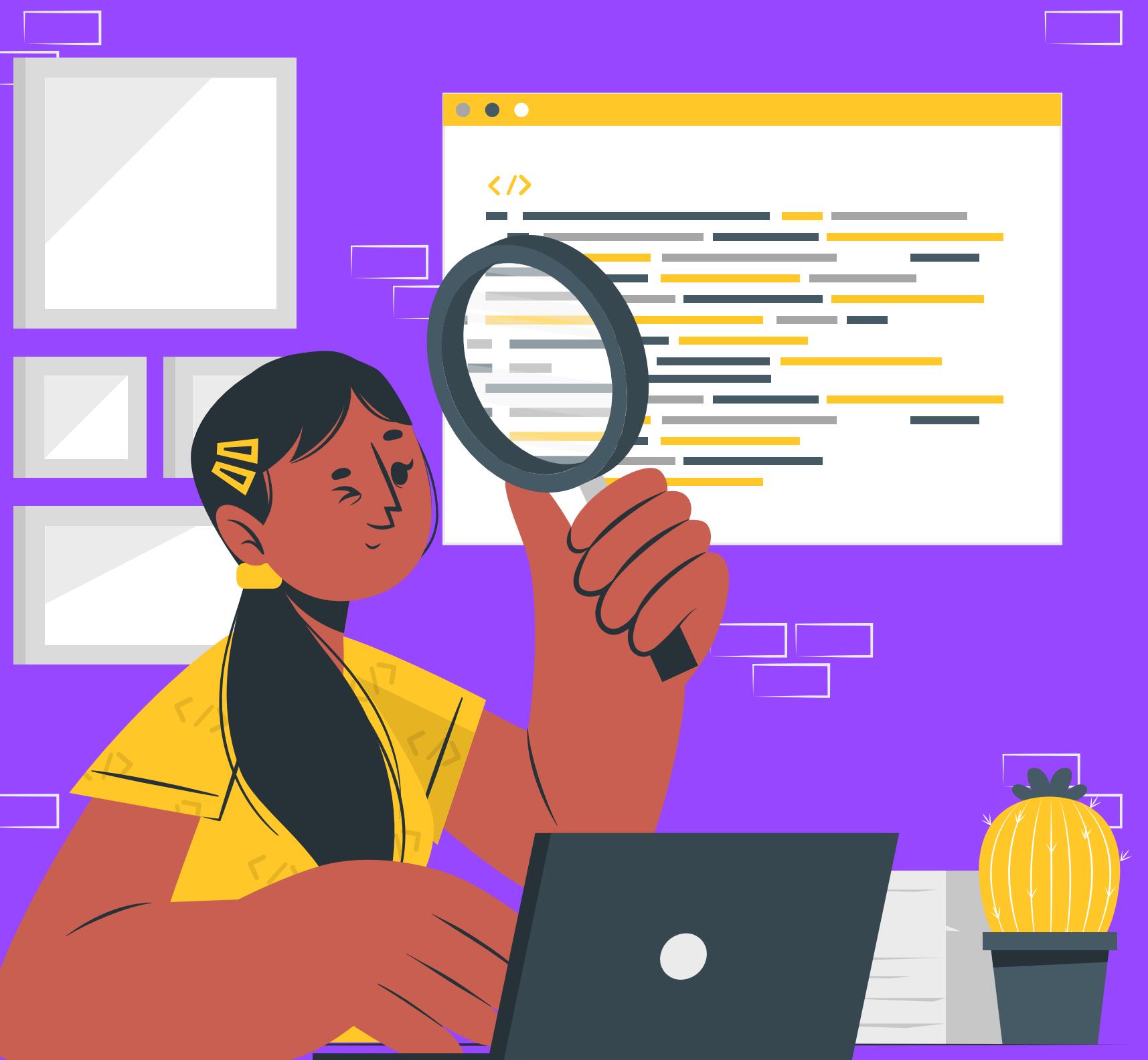
```
1 // este eh um comentario de uma linha
2
3 /*este eh um com mais linhas
4 consegue ver a diferenca?
5 espero que sim!*/
6 void setup()
7 {
8
9 }
10
11 void loop()
12 {
13
14 }
```

The code consists of two main sections: a multi-line comment starting with `/*` and ending with `*/`, and two empty function bodies for `setup()` and `loop()`.

Passo 8

Vamos iniciar nosso código, que será “`Serial.begin(9600);`”, no `setup` com o comando que inicializa o Monitor Serial

Não se esqueça do melhor amigo do programador, o ponto e vírgula!! Se não utilizar, será alertado.

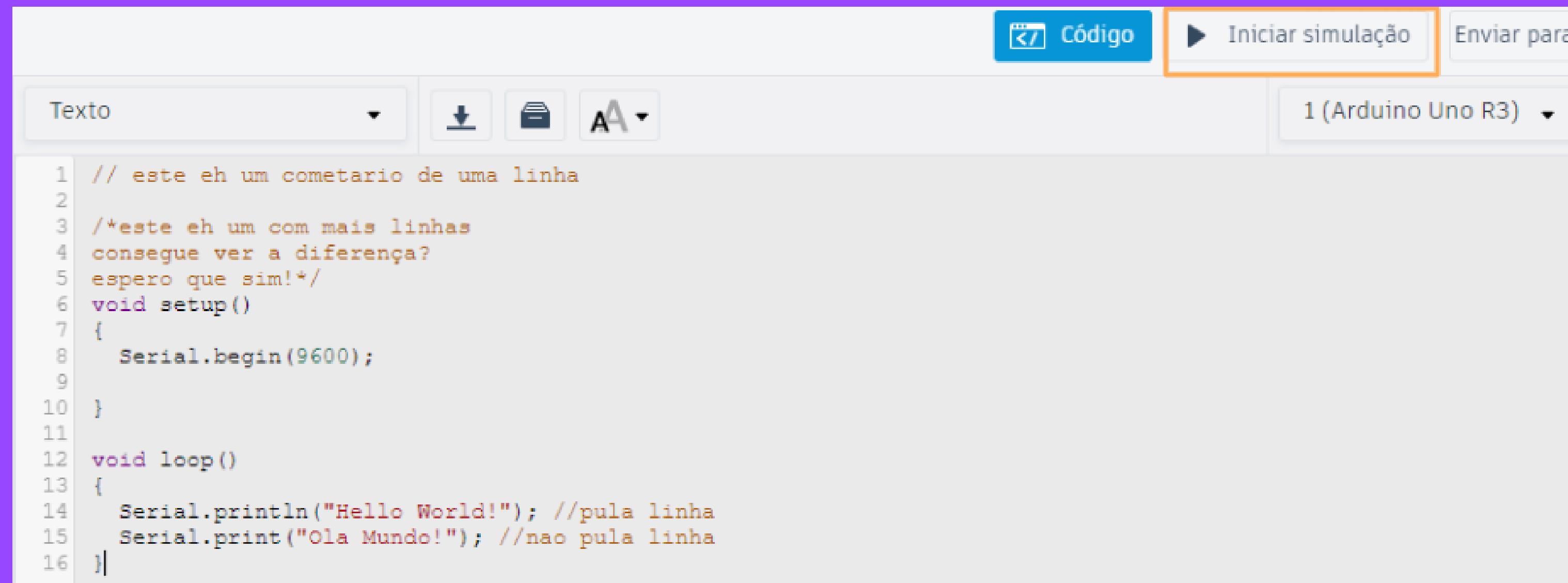


Passo 8

Na função loop, colocaremos comandos relacionados ao Monitor Serial, que exibirá as seguintes frases:

Serial.println("Hello World!"); -> quando quisermos pular uma linha

Serial.print("Ola Mundo!"); -> quando não quisermos pular uma linha



The screenshot shows the Arduino IDE interface. The top bar includes tabs for 'Código' (Code), 'Iniciar simulação' (Start Simulation) which is highlighted with an orange border, and 'Enviar para' (Send to). Below the tabs, there's a toolbar with icons for 'Texto' (Text), download, upload, and font size. A dropdown menu shows '1 (Arduino Uno R3)'. The main area contains the following code:

```
1 // este eh um comentario de uma linha
2
3 /*este eh um com mais linhas
4 consegue ver a diferenca?
5 espero que sim!*/
6 void setup()
7 {
8     Serial.begin(9600);
9 }
10
11 void loop()
12 {
13     Serial.println("Hello World!"); //pula linha
14     Serial.print("Ola Mundo!"); //nao pula linha
15 }
16 }
```

Dica

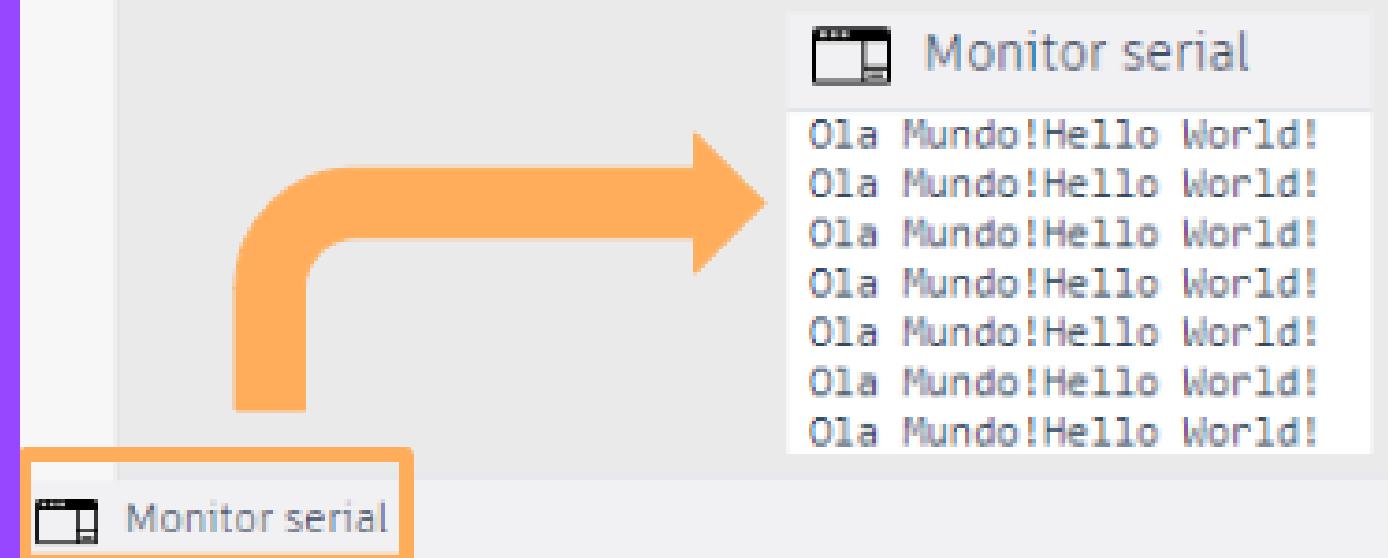
**Não coloque acentos
nas palavras, já
que os caracteres
especiais não são
lidos pelo código
pela programação na
língua estrangeira
(inglês)**



Resultado

Observe como o “Olá mundo!” não pula linha, enquanto o “Hello World!” pula, isso acontece devido as diferenças dos comandos explanados.

```
1 // este eh um comentario de uma linha
2
3 /*este eh um com mais linhas
4 consegue ver a diferenca?
5 espero que sim!*/
6 void setup()
7 {
8     Serial.begin(9600);
9
10 }
11
12 void loop()
13 {
14     Serial.println("Hello World!"); //pula linha
15     Serial.print("Olá Mundo!"); //não pula linha
16 }
```



Passo 9

Vamos aprender o comando `delay`?

Você percebeu que os comandos foram um atropelados pelo outro, sem pausa?

Para resolvemos esse problema de visualização prejudicada, vamos parar a simulação:



The screenshot shows the Arduino IDE interface. At the top right, there are three buttons: 'Código' (Code) in blue, 'Parar simulação' (Stop simulation) in green, and a third button which appears to be 'Iniciar simulação' (Start simulation). Below these buttons, there are icons for play, stop, and refresh. To the right of the play/stop icons, it says '1 (Arduino)'. The main area displays the following Arduino code:

```
1 // este eh um comentario de uma linha
2
3 /*este eh um com mais linhas
4 consegue ver a diferenca?
5 espero que sim!*/
6 void setup()
7 {
8     Serial.begin(9600);
9 }
10 void loop()
11 {
```

Passo 9

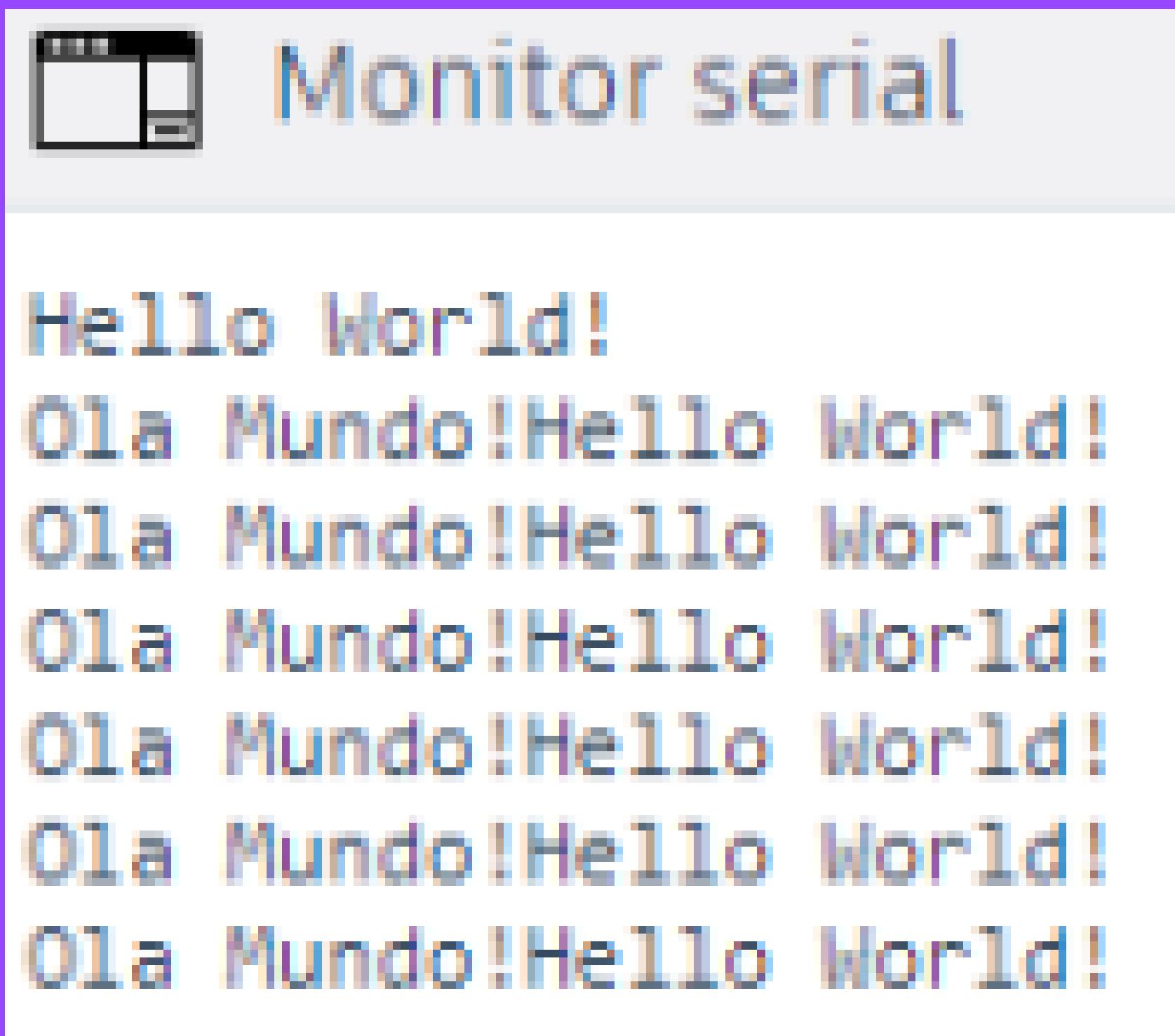
Adicione o comando do Monitor Serial, responsável por atrasar o próximo processo, sendo ele o “delay(1000);”.

```
1 // este eh um comentario de uma linha
2
3 /*este eh um com mais linhas
4 consegue ver a diferenca?
5 espero que sim!*/
6 void setup()
7 {
8     Serial.begin(9600);
9
10 }
11
12 void loop()
13 {
14     Serial.println("Hello World!"); //pula linha
15     delay(1000); //espera 1 segundo
16     Serial.print("Ola Mundo!"); //nao pula linha
```

1000 ->referente a milissegundos, indicando que o programa atrasará o próximo comando em 1 segundo.

Resultado

Inicie a simulação novamente e abra o Monitor Serial para ver esse resultado:



The screenshot shows a window titled "Monitor serial". Inside the window, there are two distinct sets of text. The first set consists of the message "Hello World!" repeated seven times. The second set consists of the message "Olá Mundo!" repeated seven times. Both sets of messages are displayed in a monospaced font.

```
Hello World!  
Olá Mundo!Hello World!
```

-> Observe a mudança de tempo,
tendo uma visualização bem
melhor.

Agora você sabe o que como criar uma conta no Tinkercad e utilizar o Monitor Serial. Continue para as próximas aulas para aprender mais sobre Robótica!!

