



Introdução ao Arduino

Prof. Simão

O que é “Arduino”?



- É uma plataforma para elaboração de protótipos
 - *Hardware* e *software* livres
- Foi criado em 2005 por 5 pesquisadores italianos
 - Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis

Combinação de *hardware* + *software*

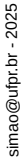


```
Arquivo Editar Sketch Ferramentas Ajuda
[Checkmark] [Run] [Save] [Upload] [Verify] Verificar

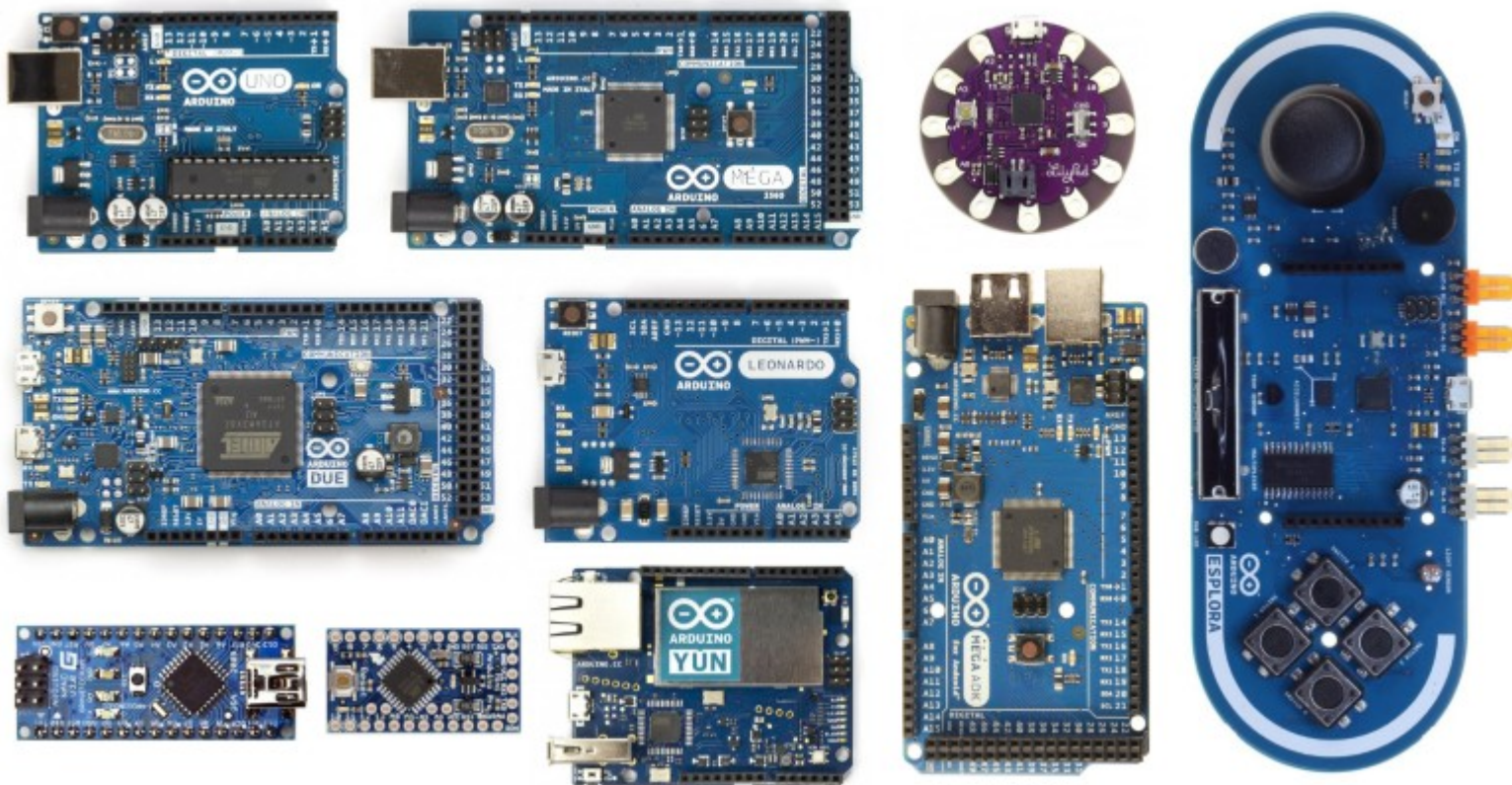
Blink

1 /*
2  * Blink
3  *
4  * Turns an LED on for one second, then off for one second, repeatedly.
5  *
6  * Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
7  * it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
8  * the correct LED pin independent of which board is used.
9  * If you want to know what pin the on-board LED is connected to on your Arduino
10 * model, check the Technical Specs of your board at:
11 * https://www.arduino.cc/en/Main/Products
12 *
13 * modified 8 May 2014
14 * by Scott Fitzgerald
15 * modified 2 Sep 2016
16 * by Arturo Guadalupi
17 * modified 8 Sep 2016
18 * by Colby Newman
19 *
20 * This example code is in the public domain.
21 *
22 * http://www.arduino.cc/en/Tutorial/Blink
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27   // initialize digital pin LED_BUILTIN as an output.
28   pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
33   digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
34   delay(1000);                       // wait for a second
35   digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
36   delay(1000);                       // wait for a second
37 }
```

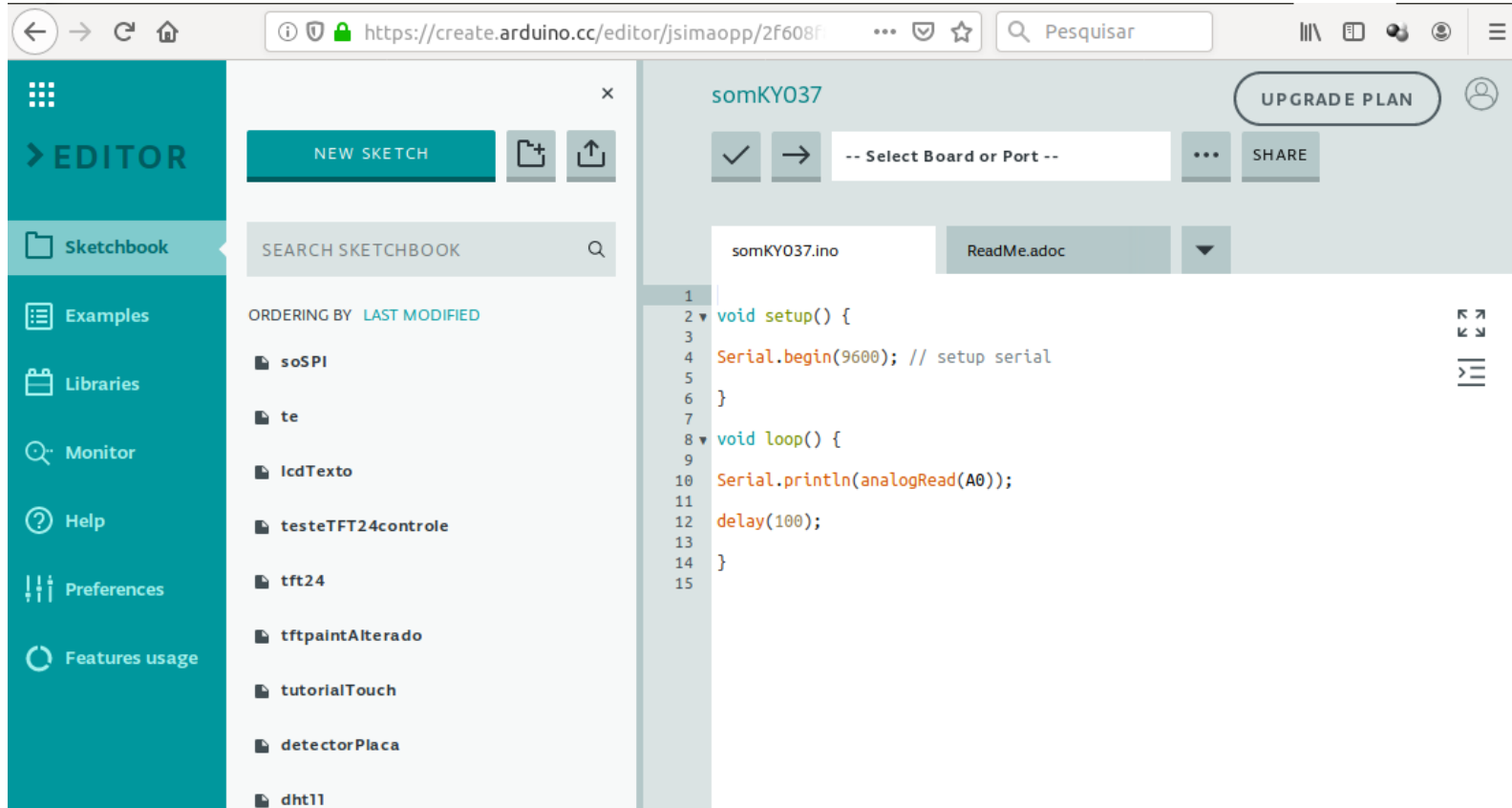
simao@ufpr.br - 2025
1010000 0100000 110010 0100000 1101111 0100000 1010011 0100000 1101001 0100000 1101101 0100000 1101000 0100000 1101111



Muitas variantes



Programação *online* ou Desktop



Programação *online* ou Desktop

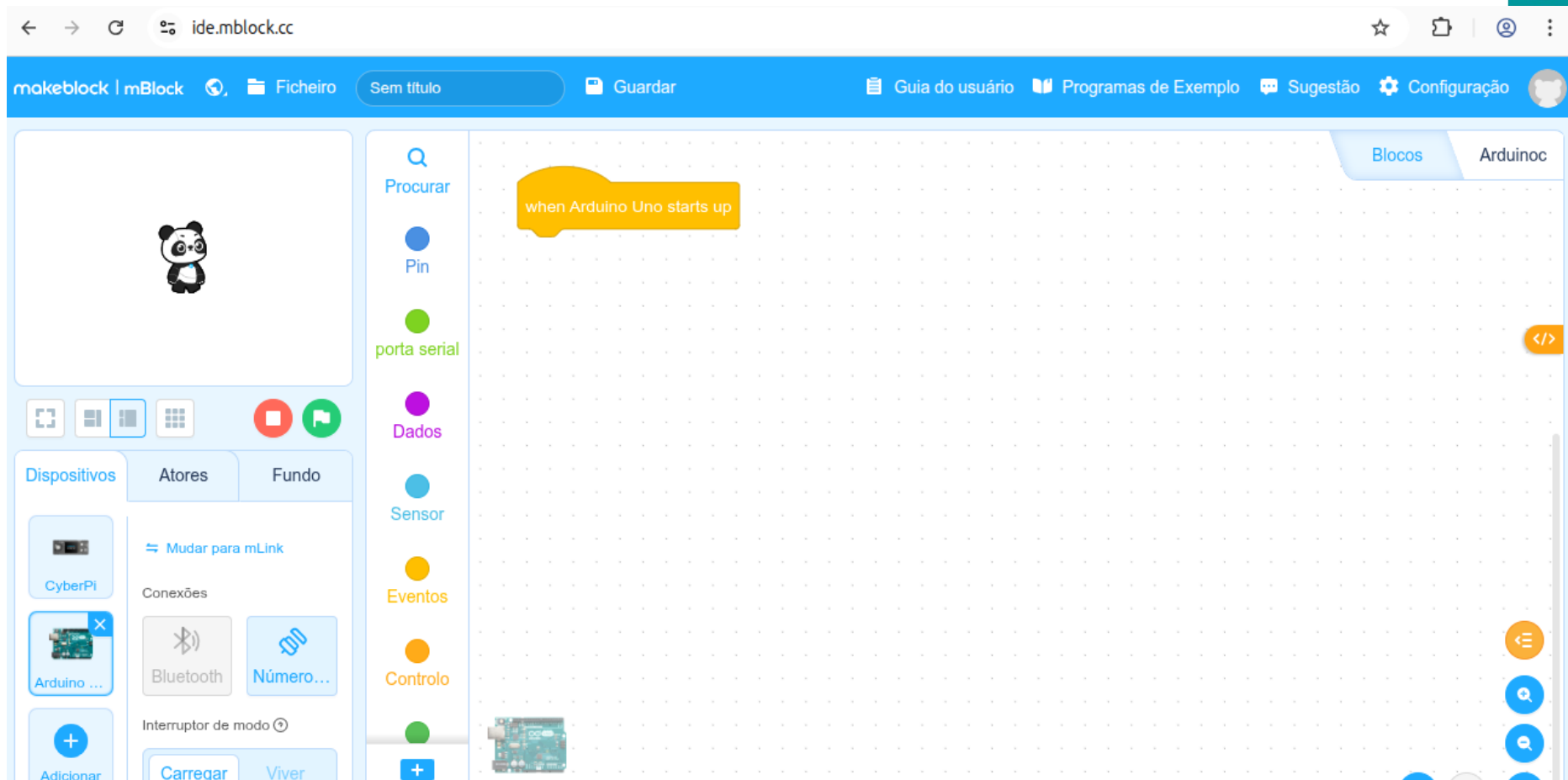


Imagem: cópia de tela efetuada pelo autor

Programação *online* ou Desktop



```
controlePWM | Arduino 1.8.5
Arquivo Editar Sketch Ferramentas Ajuda

controlePWM

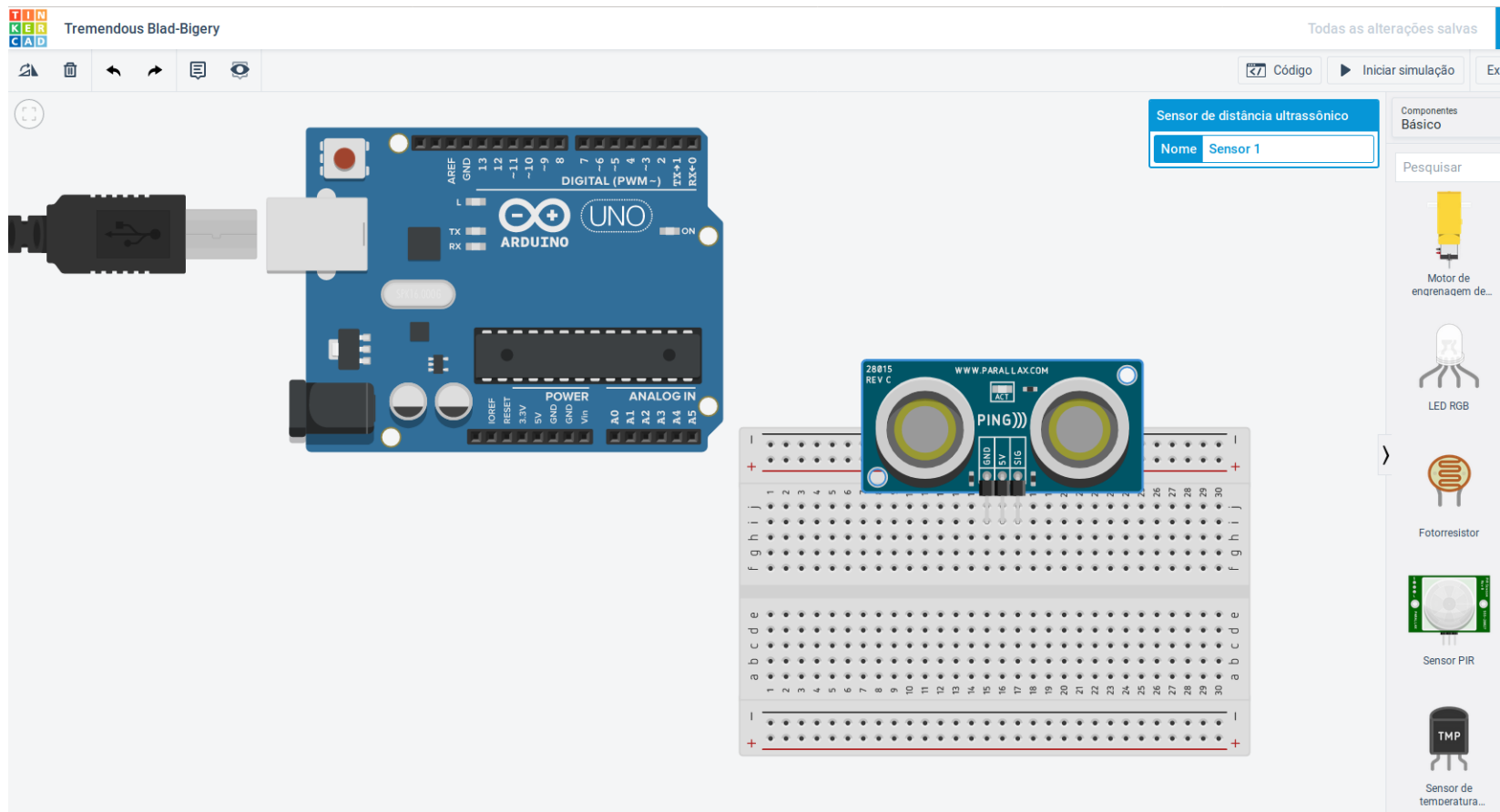
1
2 #define pwmSaida 3 //saída PWM a controlar
3 #define espera 100 //intervalo para alteração do pulso
4 void setup()
5 {
6     Serial.begin(9600);
7     Serial.print("Iniciando...\n\n");
8     pinMode(pwmSaida, OUTPUT); //saída PWM a controlar
9 }
10
11 void loop()
12 {
13     int nivelPWM = 0; //vai de 0% a 100%
14     Serial.print("Aumentando...\n");
15     for (nivelPWM = 0; nivelPWM < 256; nivelPWM++)
16     {
17         analogWrite(pwmSaida, nivelPWM);
18         delay(espera);
19     }
20     Serial.print("Diminuindo...\n"); //Vai de 100% a 0%
21     for (nivelPWM = 255; nivelPWM >= 50; nivelPWM--)
22     {
23         analogWrite(pwmSaida, nivelPWM);
24         delay(espera);
25     }
26 }
```

Compilação terminada.

O sketch usa 2224 bytes (6%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 230 bytes (11%) de memória dinâmica, deixando 1818 bytes para variáveis locais. O máximo são 2048 bytes.

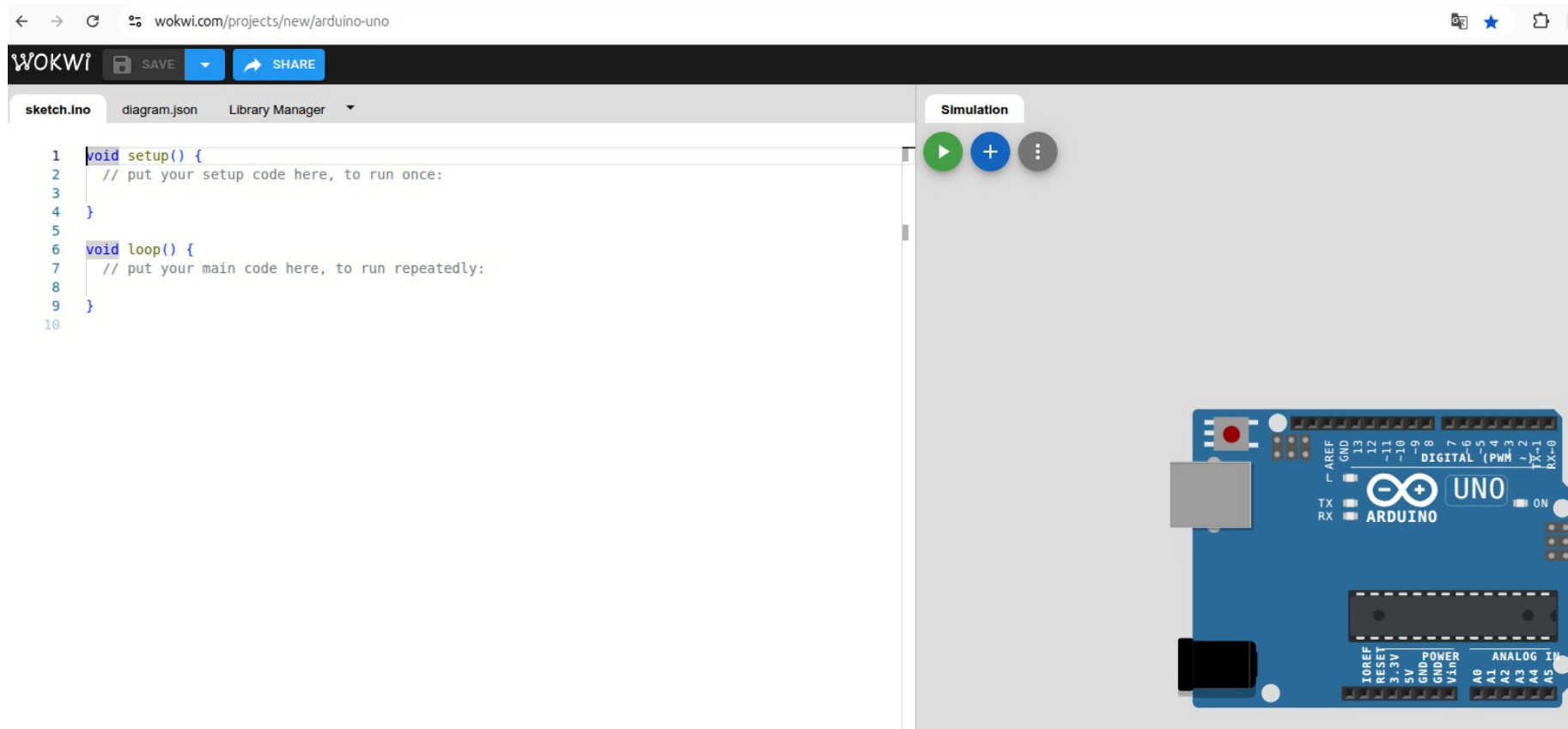
20 Arduino/Genuino Uno em /dev/ttyUSB0

Simulações *online*

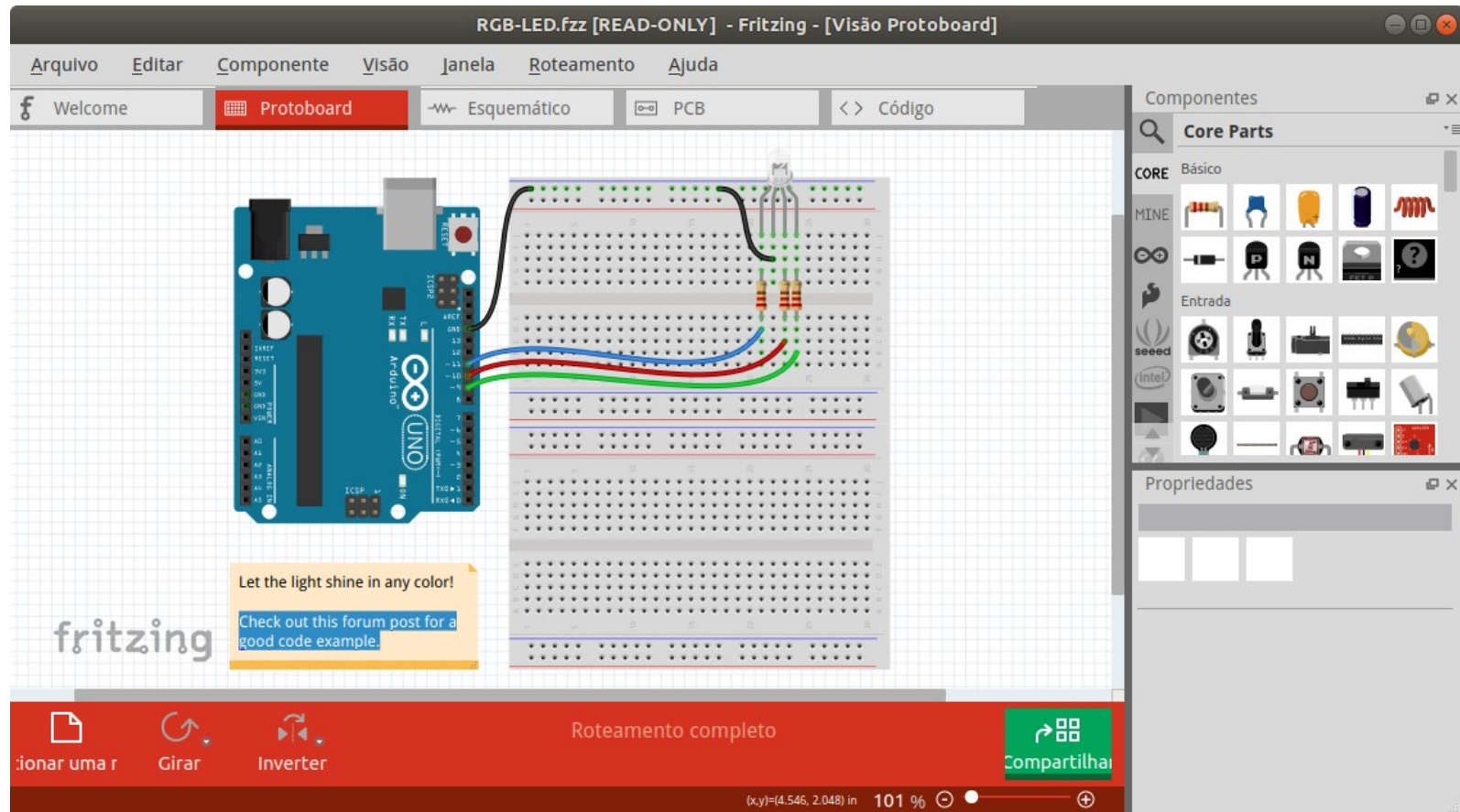


Simulações *online*

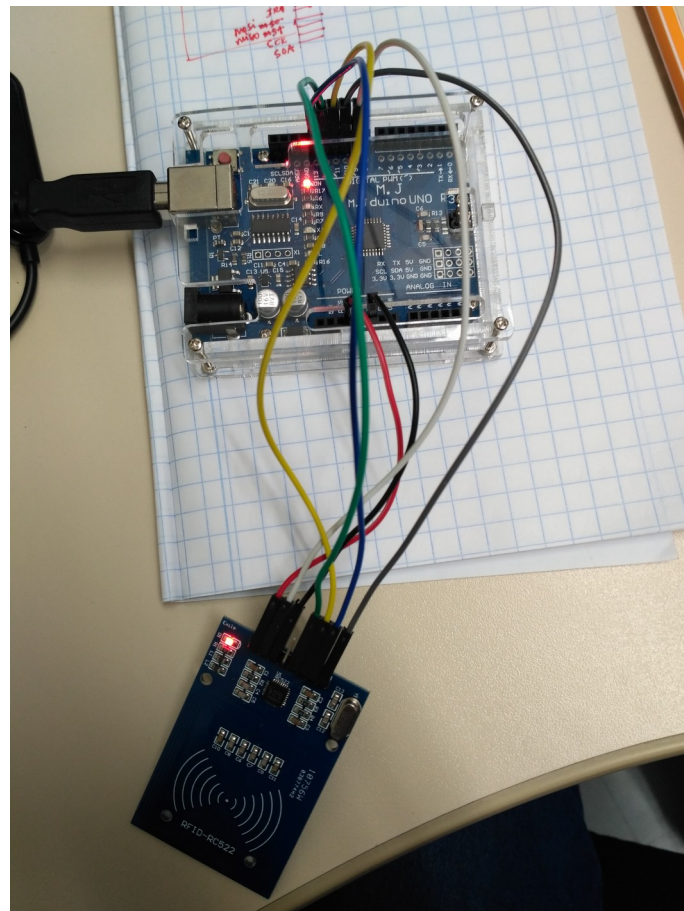
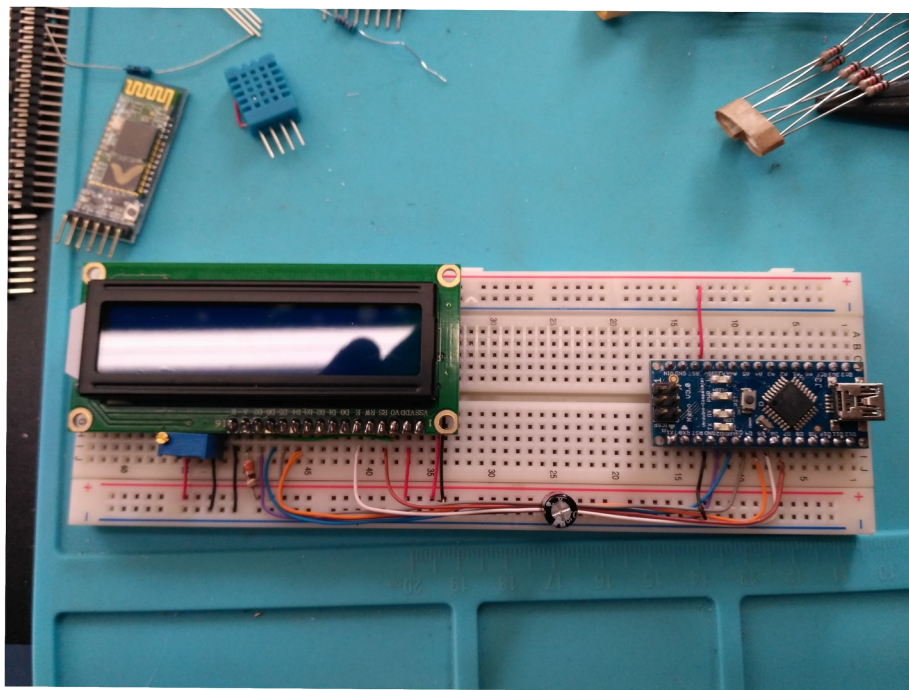
<https://wokwi.com/projects/new/arduino-uno>



Apoio às ligações das ligações



Prototipação



Hands on



Ambiente



- Instale o IDE do Arduino
 - Você terá funcionamento *off line* e liberdade para instalar bibliotecas e placas
<https://www.arduino.cc/en/software/#ide>
- Ou crie uma conta *online*
 - Não precisa instalar nada, porém haverá algumas limitações de placas a serem resolvidas com versão paga

Em qualquer caso...



- Haverá necessidade de comunicação entre a placa e o computador
 - A conexão é por meio de USB, porém haverá um mapeamento interno para conexão serial
 - Poderá ser necessário ajustar *drivers*

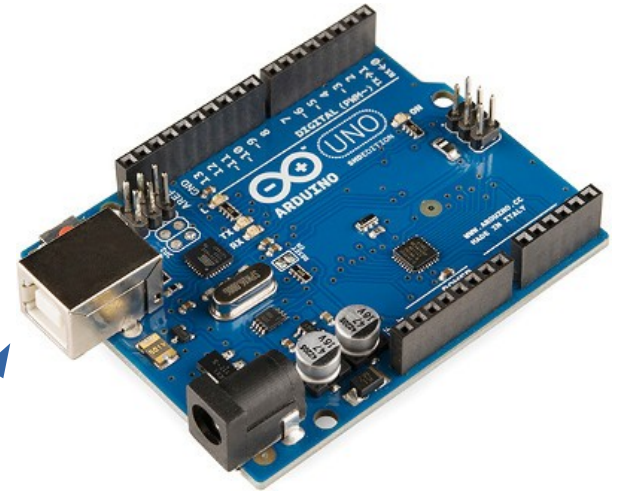
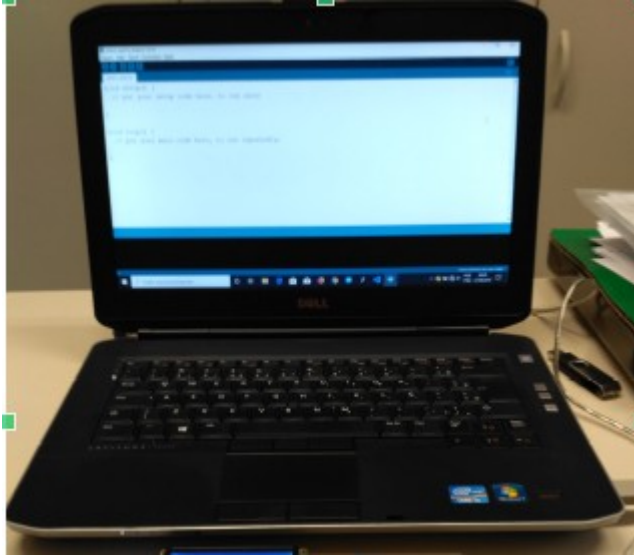
Opções

- Desktop – procure o ícone



- <https://wokwi.com/projects/new/arduino-uno> (simulação)
- <https://create.arduino.cc/> ou
- <https://ide.mblock.cc/> ou
- <https://app.arduino.cc/sketches/>
 - Vai demandar um *driver*

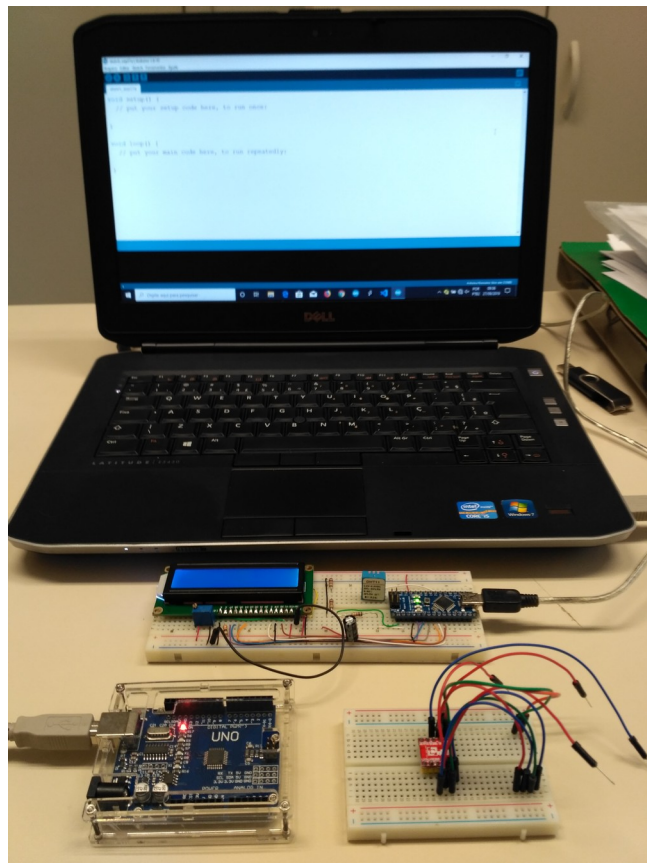
Como ficará



**USB ao
computador**

USB ao Arduino

Como ficará

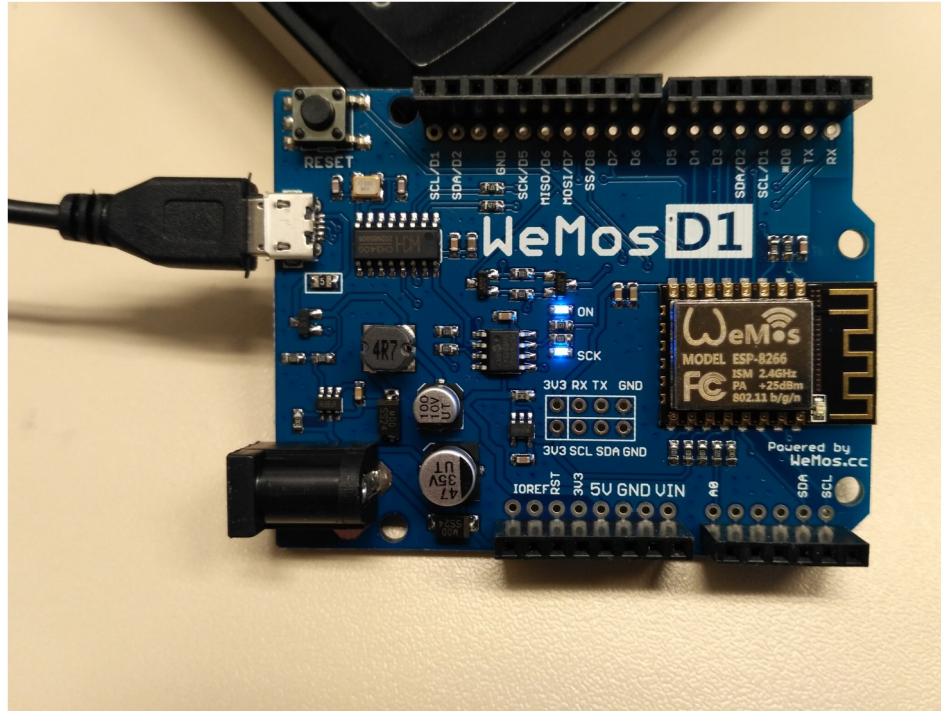


O ambiente de desenvolvimento estará instalado / configurado no computador.

A placa escolhida estará disponível com cabo de comunicação USB.

O computador deverá reconhecer a comunicação com a placa!

Led aceso = energia

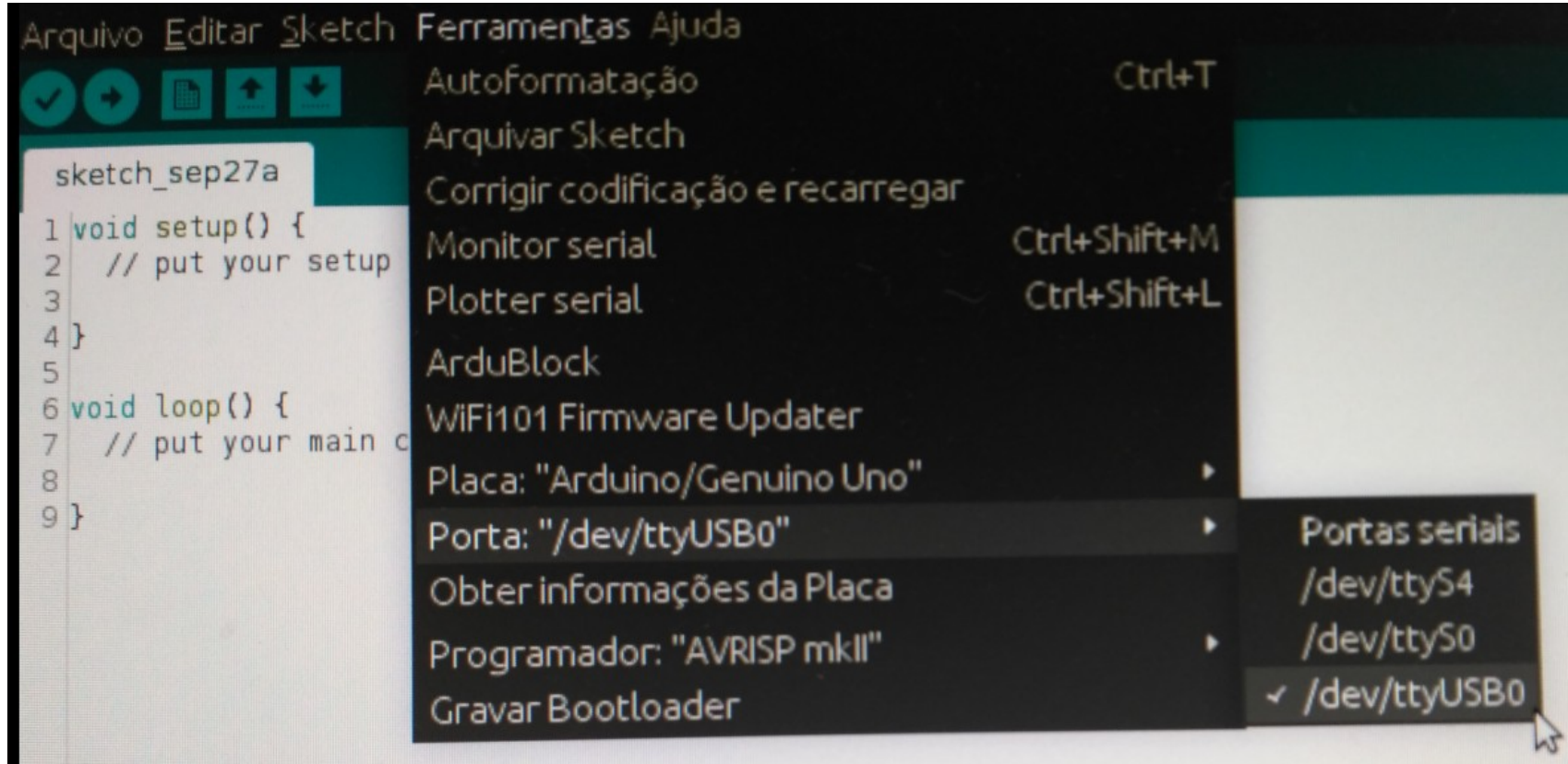


Led aceso na placa é sinal de que a energia está chegando à placa, a conexão USB está OK, mas não garante que a comunicação esteja OK!

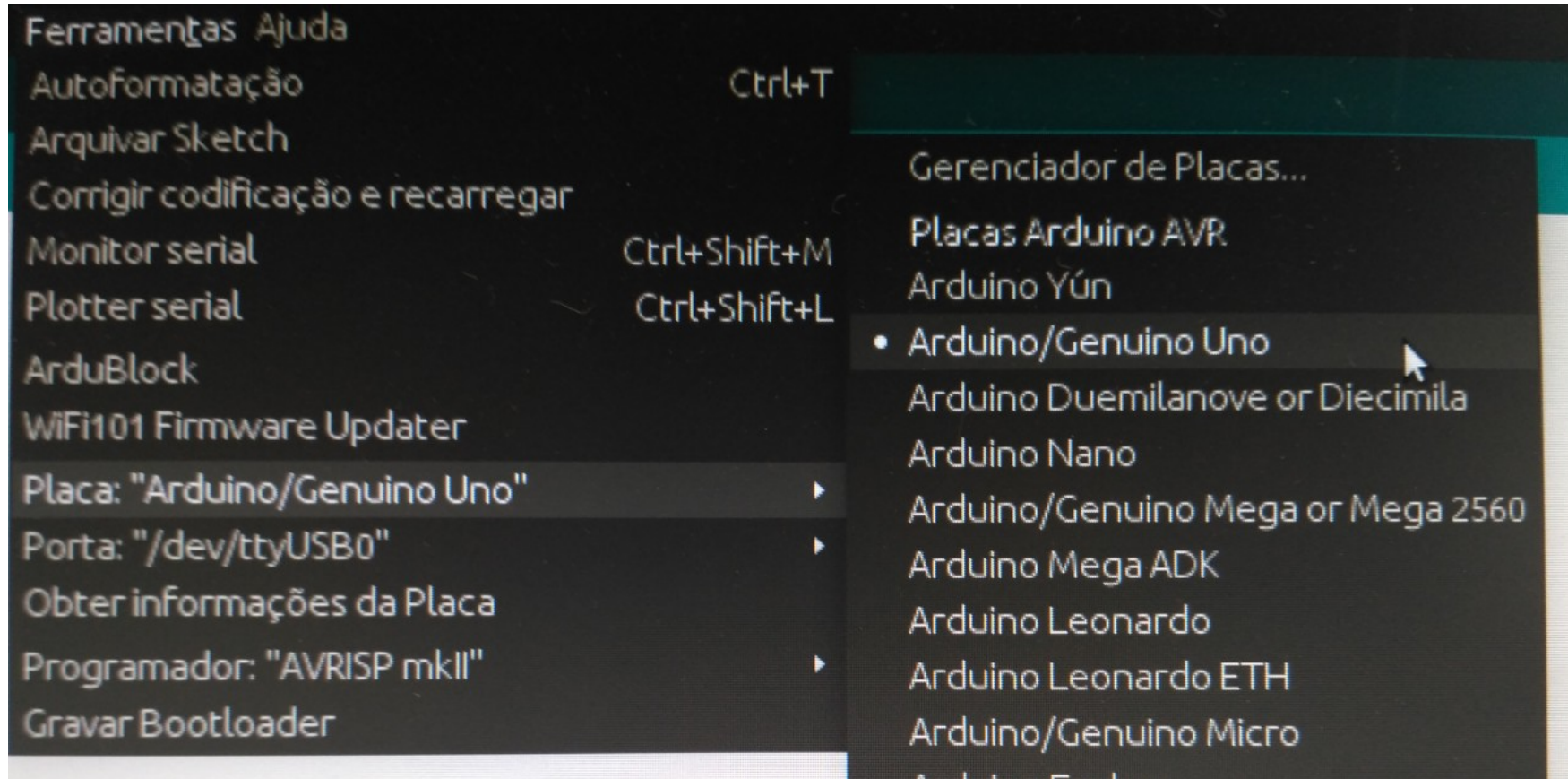
Testando

- Abra o IDE do arduino
- Conecte o cabo USB na placa
- Conecte o cabo USB no computador
- Verifique a identificação da placa no IDE
 - Ferramentas / Portas / Portas seriais
 - Ferramentas / Placas

Identificação da placa no IDE



Confirme a placa



Não sabe em qual porta?

- Verifique no gerenciador de dispositivos do Windows
- Verifique nas conexões se for Chromebook
- Verifique com `lsusb` se for linux

No linux



Verifique se seu usuário tem acesso:

```
ls -l /dev/ttyACM*
```

Se não tiver:

```
sudo usermod -aG dialout nomeDoSeuUsuario
```

No Windows

Muitas vezes não localiza a comunicação serial em função da falta de driver para o conversor CH340

Pode usar este (exemplo, existem vários):
https://sparks.gogo.co.nz/assets/_site_/downloads/CH34x_Install_Windows_v3_4.zip

Tudo ok?



Em seu ambiente a placa Arduino UNO deverá estar disponível e a conexão serial deverá estar disponível e selecionada.

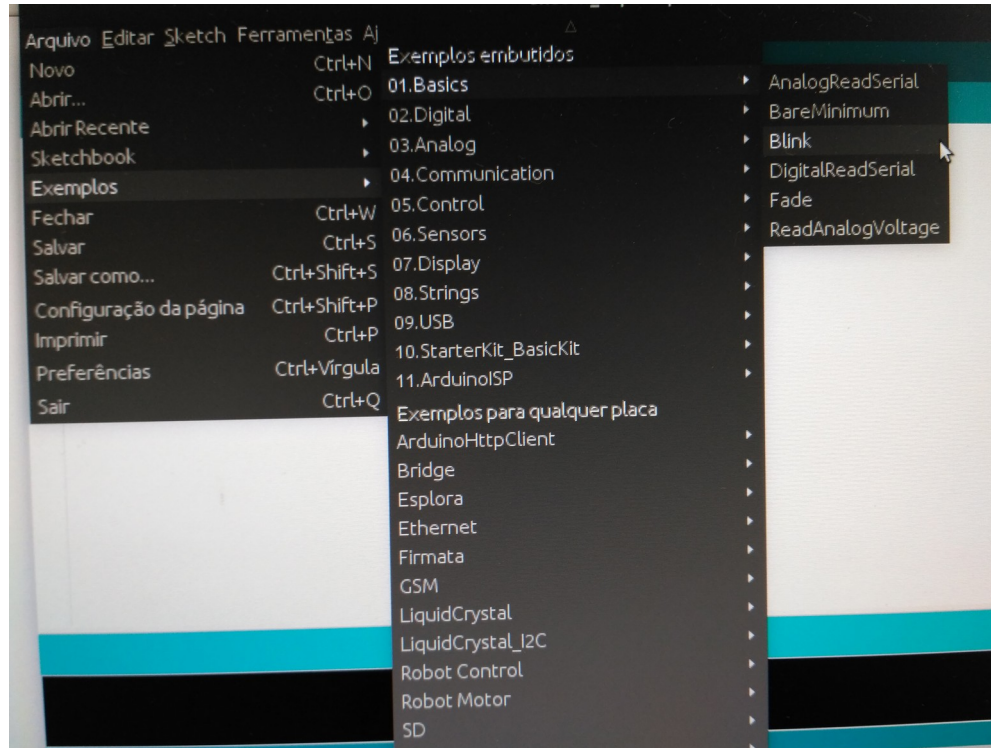
Placa conectada



- Quando a placa está conectada ela vai receber energia pela USB. Neste caso, vai inicializar e executar a última programação enviada. Assim, se, quando você conectar a placa no computador, um LED estiver piscando, não se preocupe – é porque esta foi a última programação enviada (e iremos alterá-la na sequência).

Carregue um exemplo

Arquivo/ Exemplos/ 01.Basics/ Blink



Código do exemplo '*blink*'

A screenshot of the Arduino IDE interface. The title bar reads 'Blink | Arduino 1.8.5'. The menu bar includes 'Arquivo', 'Editar', 'Sketch', 'Ferramentas', and 'Ajuda'. The toolbar shows icons for opening, saving, and running. The main text area displays the 'Blink' example code, which includes a multi-line comment explaining the code's purpose and history, and the start of the C++ code with the 'void setup()' function.

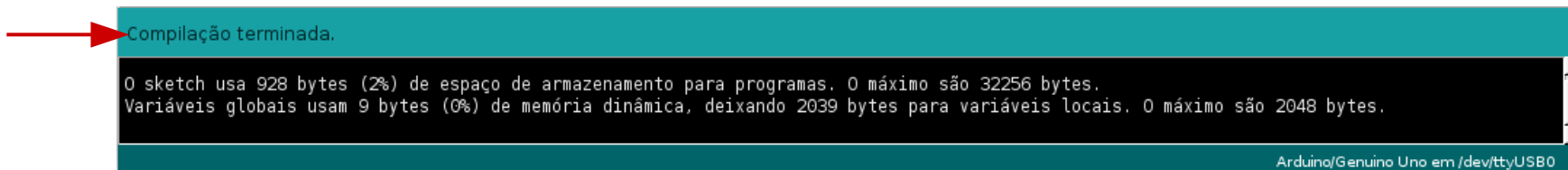
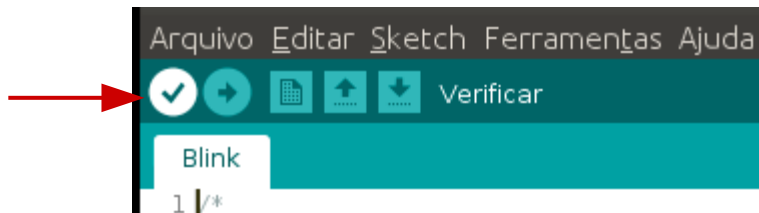
```
1 /*  
2  Blink  
3  
4  Turns an LED on for one second, then off for one second, repeatedly.  
5  
6  Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO  
7  it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to  
8  the correct LED pin independent of which board is used.  
9  If you want to know what pin the on-board LED is connected to on your Arduino  
10 model, check the Technical Specs of your board at:  
11 https://www.arduino.cc/en/Main/Products  
12  
13 modified 8 May 2014  
14 by Scott Fitzgerald  
15 modified 2 Sep 2016  
16 by Arturo Guadalupi  
17 modified 8 Sep 2016  
18 by Colby Newman  
19  
20 This example code is in the public domain.  
21  
22 http://www.arduino.cc/en/Tutorial/Blink  
23 */  
24  
25 // the setup function runs once when you press reset or power the board  
26 void setup() {
```

Arduino/Genuino Uno em /dev/ttyUSB0

Compilar e transferir o código



- Sem alterar a configuração de porta e placa já testadas:
 - Clique no ícone de compilação para testar o código:



Compilar e transferir o código

- Sem alterar a configuração de porta e placa já testadas:
 - Clique no ícone de transferência:

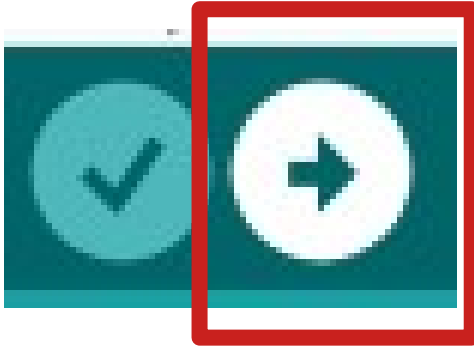


Nota: durante a transferência de dados os leds Rx e Tx da placa irão piscar.

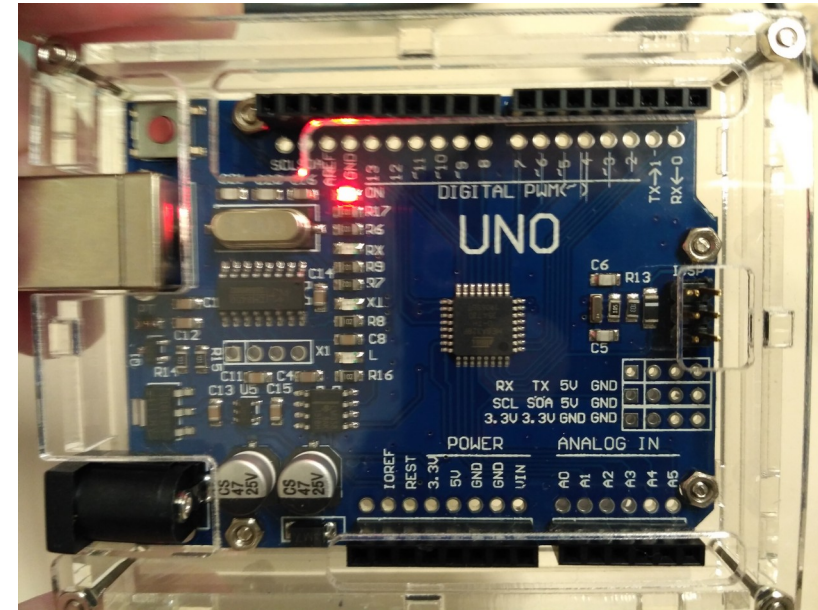
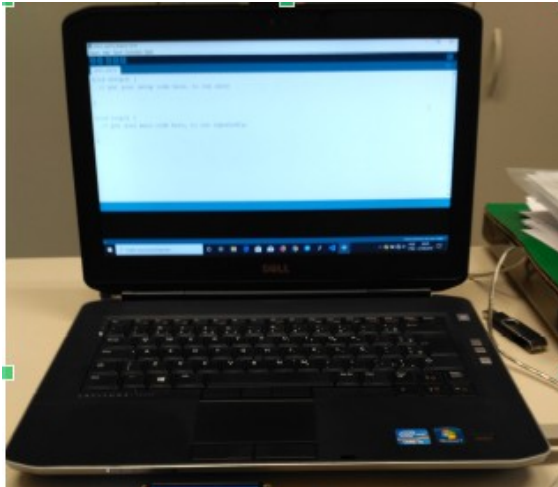
0 sketch usa 928 bytes (2%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 9 bytes (0%) de memória dinâmica, deixando 2039 bytes para variáveis locais. O máximo são 2048 bytes.

Arduino/Genuino Uno em /dev/ttyUSB0

Compilar e transferir o código



Converte ao código -da placa- Arduino que você selecionou e envia ao Arduino

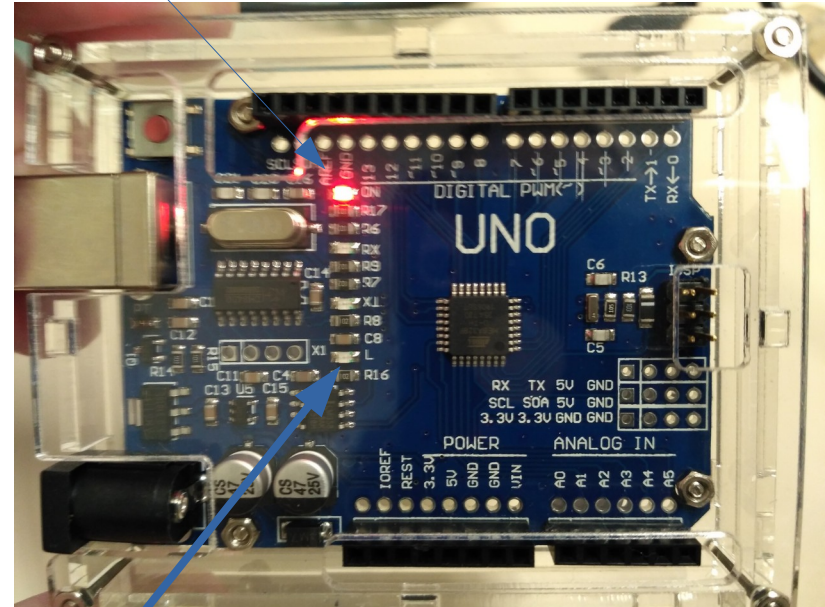
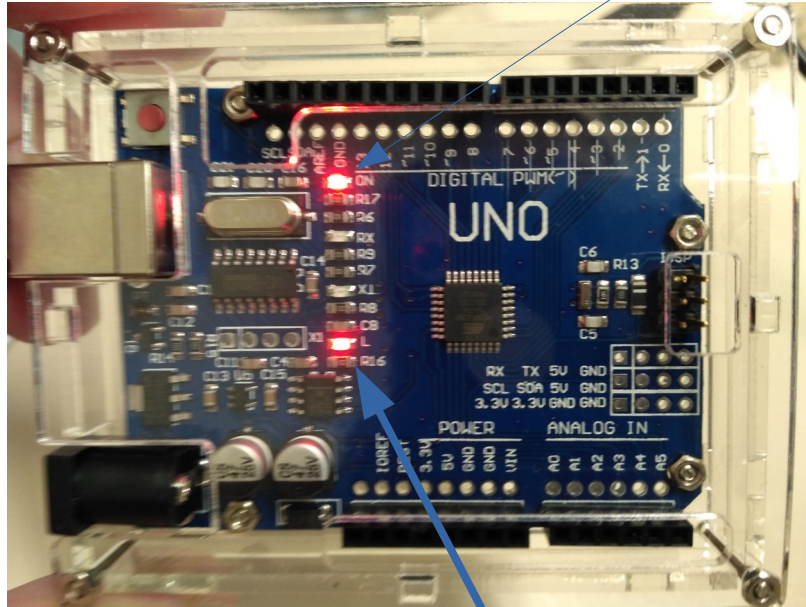


O Arduino recebe e, na sequência, executa o código

O led da placa deverá piscar



LED de energia – sempre aceso



LED da placa (pino 13) deverá piscar

Parabéns!



Continuando...



- 1) Pode ser que sua conexão não tenha dado certo, mas você não percebeu pois a última programação enviada ao Arduino era para piscar um LED e foi o que ele fez...
- 2) Vamos compreender e alterar o código enviado à placa, para termos certeza de que os comandos em execução são os nossos

Compreendendo e alterando...



- Há duas funções principais no código do Arduino:
 - *setup()*
 - Seta parâmetros e condições iniciais de funcionamento
 - *loop()*
 - Executa 'para sempre' as rotinas descritas na função

Compreendendo e alterando...



- ***setup()***

- No código do exemplo, a função `setup()` inicializa a porta 13 da placa (`LED_BUILTIN`, padrão) como sendo uma saída (`OUTPUT`), usando a função `pinMode(pino, direção)`

```
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27     // initialize digital pin LED_BUILTIN as an output.
28     pinMode(LED_BUILTIN, OUTPUT);
29 }
```

Nota: os números de linhas podem ser diferentes.

Compreendendo e alterando...



- *loop()*

- No código do exemplo, a função `loop()` utiliza a função `digitalWrite (porta, valor)` e a função `delay(tempo em milisegundos)` para alternar a energia ao LED, que acenderá ou não

```
31 // the loop function runs over and over again forever
32 void loop() {
33   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
34   delay(1000); // wait for a second
35   digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
36   delay(1000); // wait for a second
37 }
```

Nota: os números de linhas podem ser diferentes.

Compreendendo e alterando...



- **Altere o valor do `delay()`**
 - Altere o valor do `delay()`: tempos menores farão o led piscar mais rápido e vice-versa; troque o intervalo em que o led fica aceso e apagado

```
32 void loop() {  
33     digitalWrite(13, HIGH);           // liga o led (veja o número da porta...)  
34     delay(2000);                      //espera 2 segundos  
35     digitalWrite(LED_BUILTIN, LOW);  // desliga o led  
36     delay(500);                      // espera 0,5 segundos  
37 }
```

Nota: os números de linhas podem ser diferentes.

Tudo ok?



O LED da placa que você está usando deverá estar piscando de acordo com as alterações que você fez no programa.

Altere de novo os valores e envie para a placa.

Linguagem C/ C++



- A linguagem C existe desde os anos 70 e ainda é muito popular e muitos dispositivos IoT e seus *drivers* são desenvolvidos usando C/ C++.
- A quase totalidade dos exemplos disponíveis na Internet para o ambiente Arduino (e muitos outros microcontroladores) é apresentada em C / C++.
- Saber a sintaxe do C possibilita que você se sinta mais à vontade com PHP, Java, JavaScript, Python.

Atenção



- A linguagem é “*case sensitive*”
 - Maiúsculas e minúsculas não são a mesma coisa!
- Comentários
 - Servem para documentação e auxílio
 - No código, um “//” transforma em comentário até o final da linha
 - Para um bloco de comentários iniciamos com “/*” e terminamos com “*/”

O que você aprendeu neste início, que será necessário para os próximos passos e compreende:

1. Usar o IDE para registrar a conexão serial
2. Usar o IDE para registrar a placa Arduino
3. Compilar um código fornecido
4. Enviar o código compilado para o Arduino

→ não fique com dúvidas!!!

Uso do simulador

- Vamos ver como se comporta o simulador, repetindo o código (CTRL C + CTRL V) que você acabou de usar
- Vamos usar <https://wokwi.com/projects/new/arduino-uno>

<https://wokwi.com/projects/new/arduino-uno>



WOKWI

SAVE SHARE

Docs SIGN IN

sketch.ino diagram.json Library Manager

```
1 void setup() {
2   pinMode(LED_BUILTIN, OUTPUT);
3 }
4
5 void loop() {
6   digitalWrite(LED_BUILTIN, HIGH);
7   delay(850);
8   digitalWrite(LED_BUILTIN, LOW);
9   delay(350);
10 }
11
```

Simulation

2 – clique em simulação

1 – digite o código

Dúvidas



Se tiver dúvidas do que foi desenvolvido até o momento, tire-as agora, antes de prosseguirmos.

Caso contrário, vamos em frente...

Parabéns!

