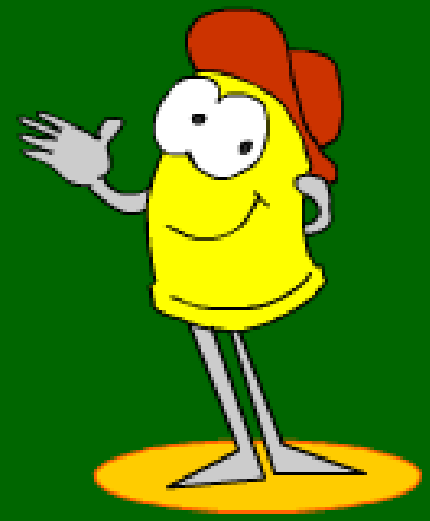
PESQUISAR ELETRÔNICA

TUTORIAL 08
UNO BÁSICO 01 – PROCESSANDO UMA ENTRADA DIGITAL



07 Processando uma entrada digital

- Este tutorial você vai abordar os seguintes tópicos:
- Como processar uma entrada digital.
- Como estabelecer uma comunicação serial entre o Arduino e um PC!

Professor Roberto Bairros dos Santos.
27/10/2015

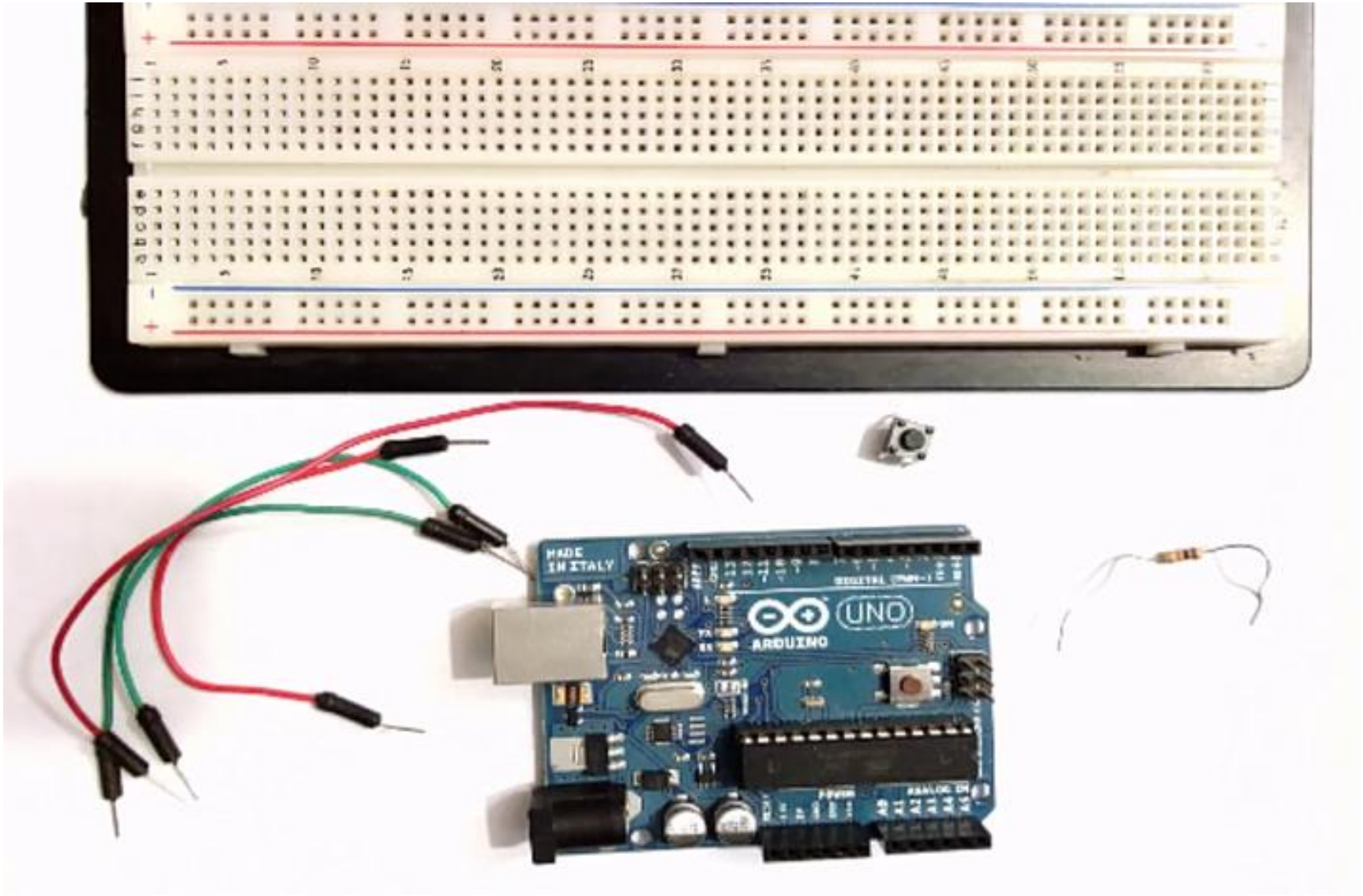
- No programa exemplo abaixo a chave é chamada “pushbotton” que significa uma chave sem retenção!

```
// digital pin 2 has a pushbutton attached to it. Give it a name:
int pushButton = 2;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);        // delay in between reads for stability
}
```

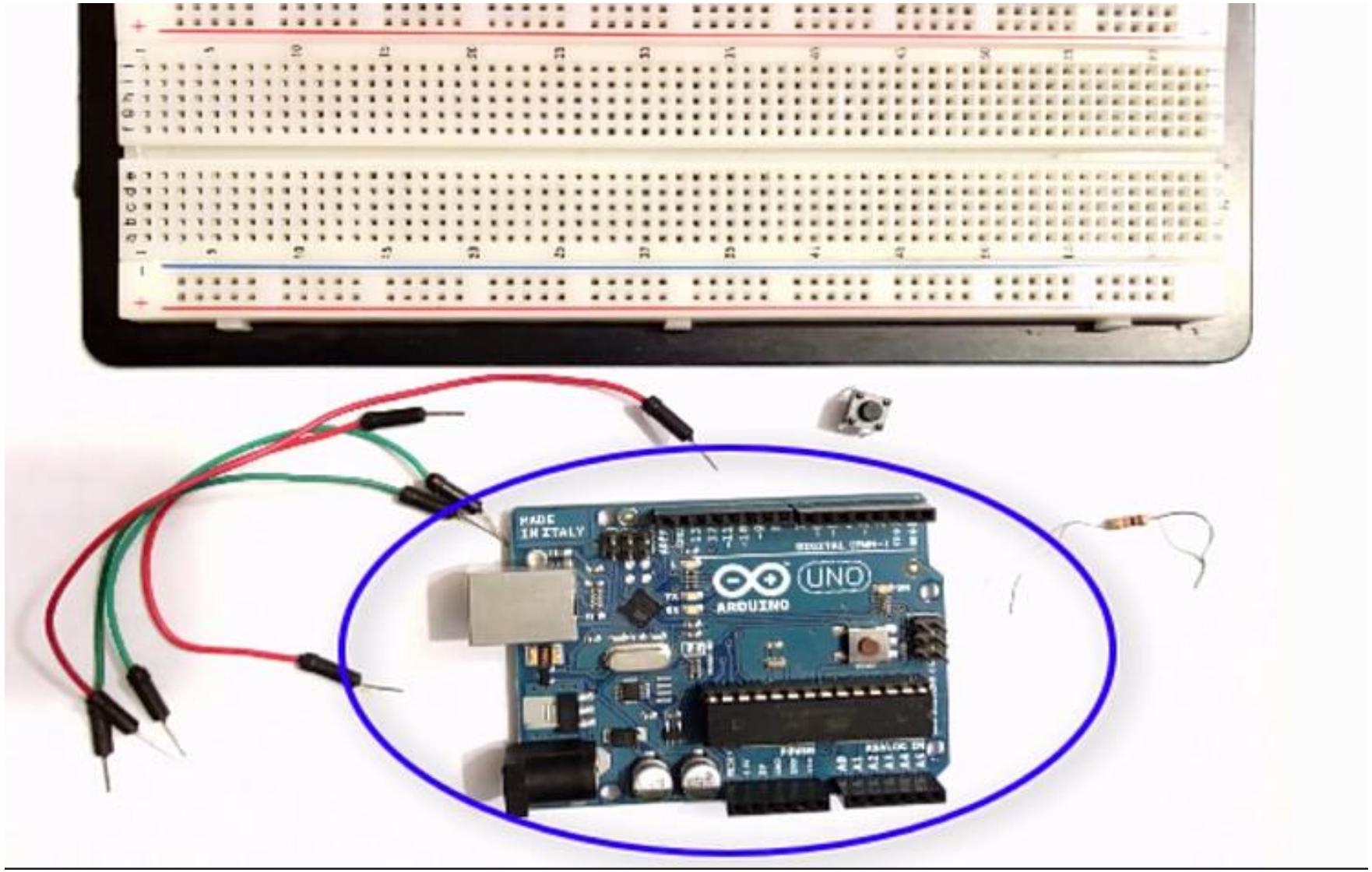
- A comunicação com o PC é feita através da função Serial()!

```
11 // the setup routine runs once when you press reset:
12 void setup() {
13   // initialize serial communication at 9600 bits per second:
14   Serial.begin(9600);
15   // make the pushbutton's pin an input:
16   pinMode(pushButton, INPUT);
17 }
```

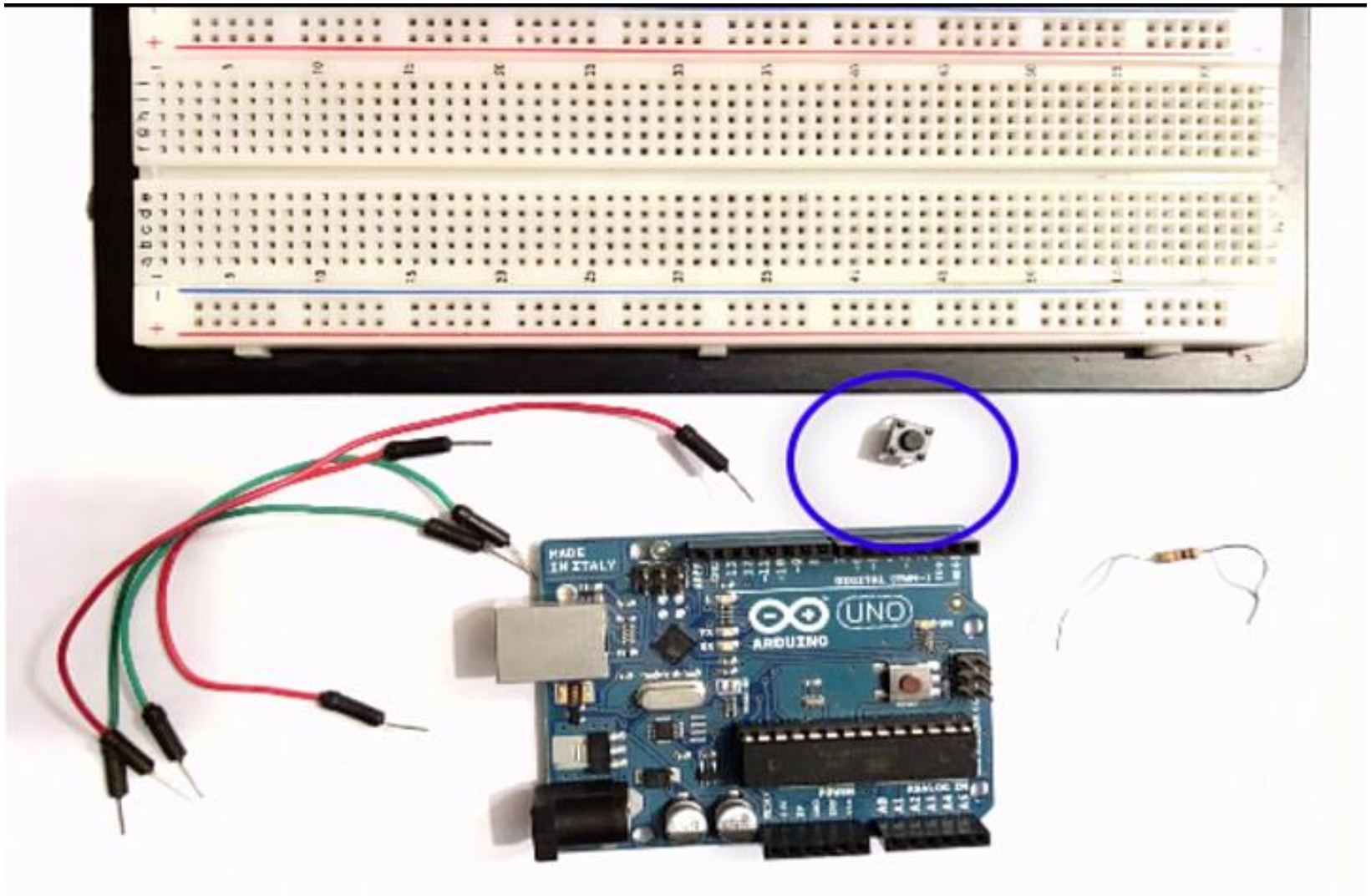
- O material para o teste da entrada digital é mostrado abaixo!



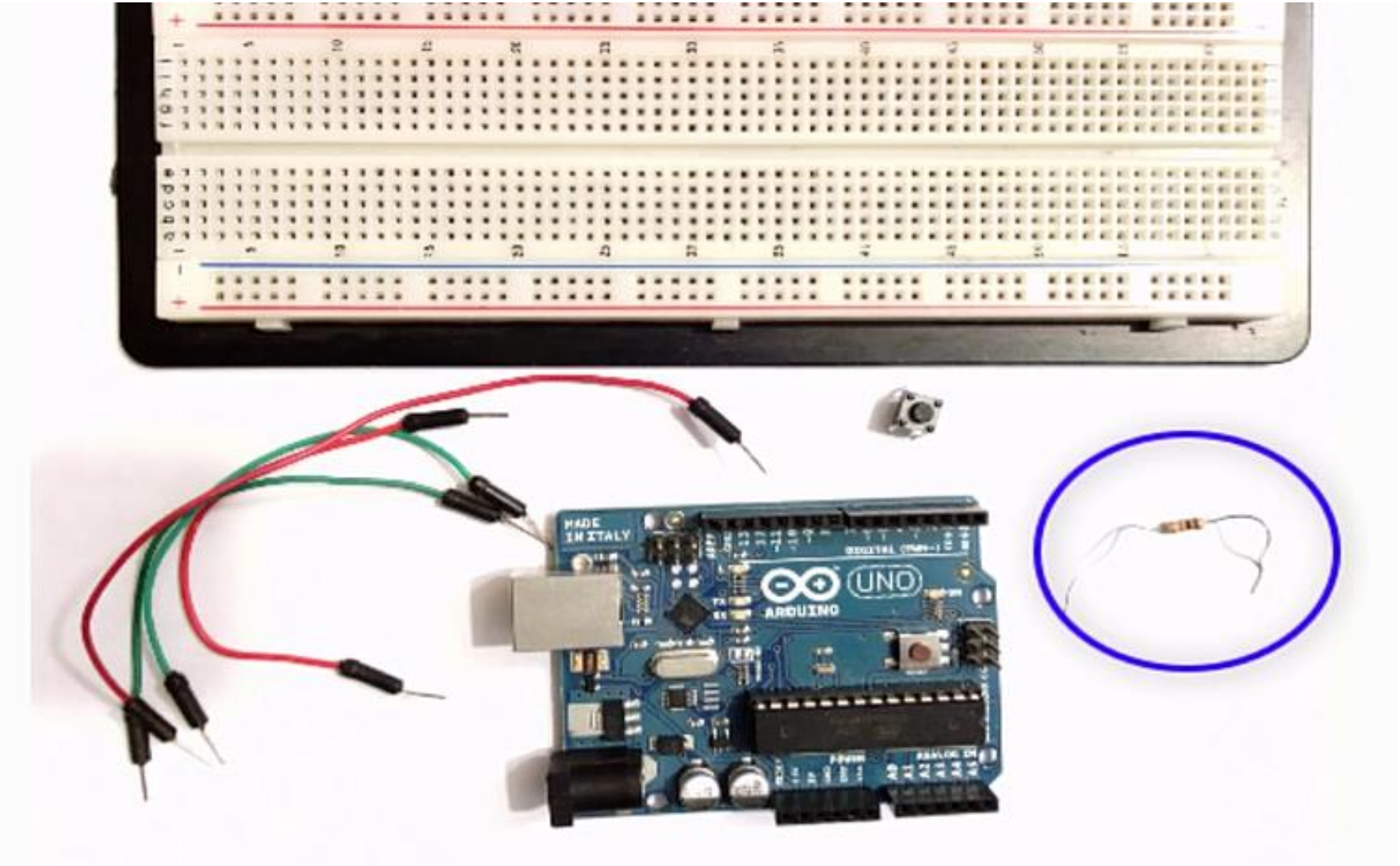
- Placa UNO!



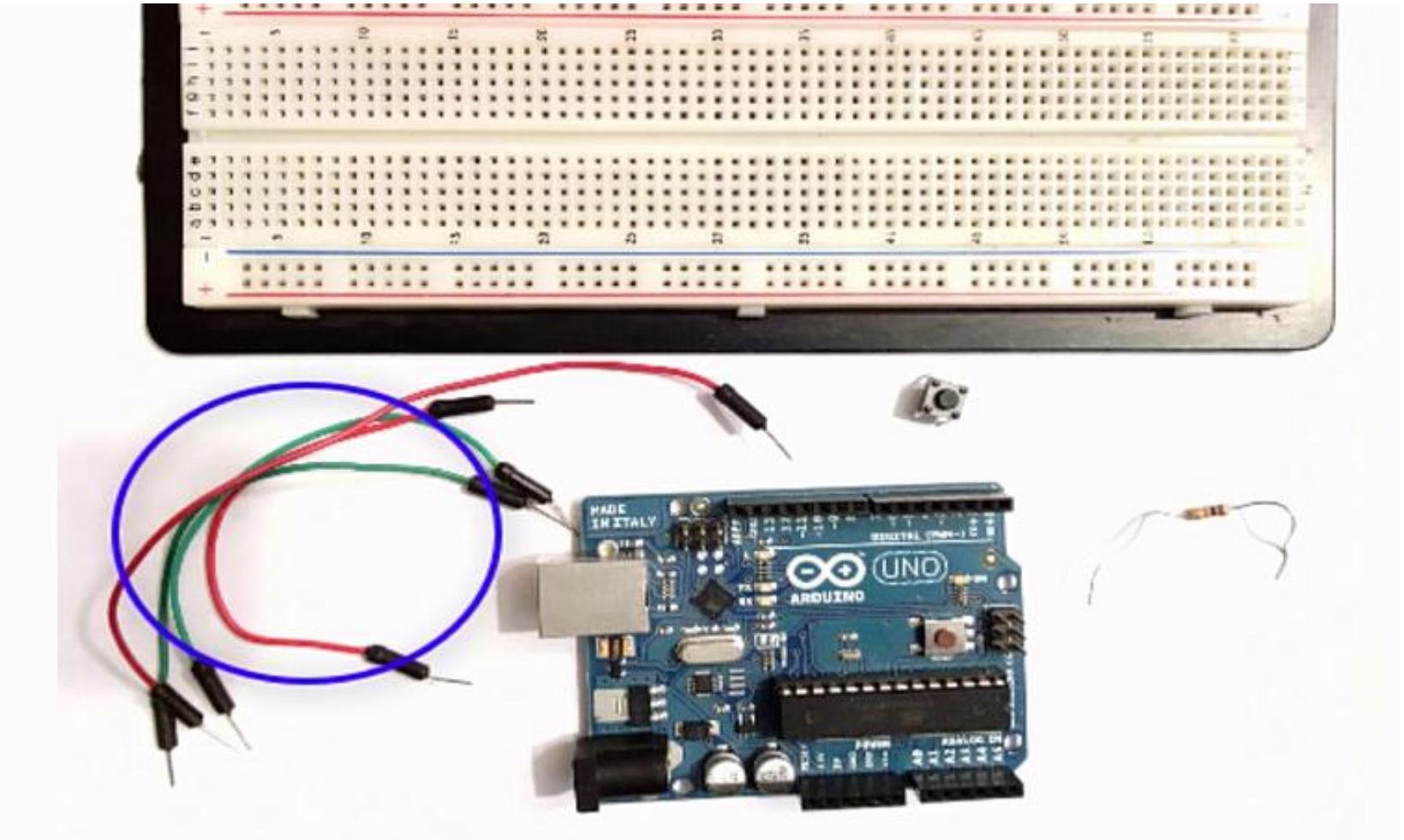
- A chave de impulso (push botton)!



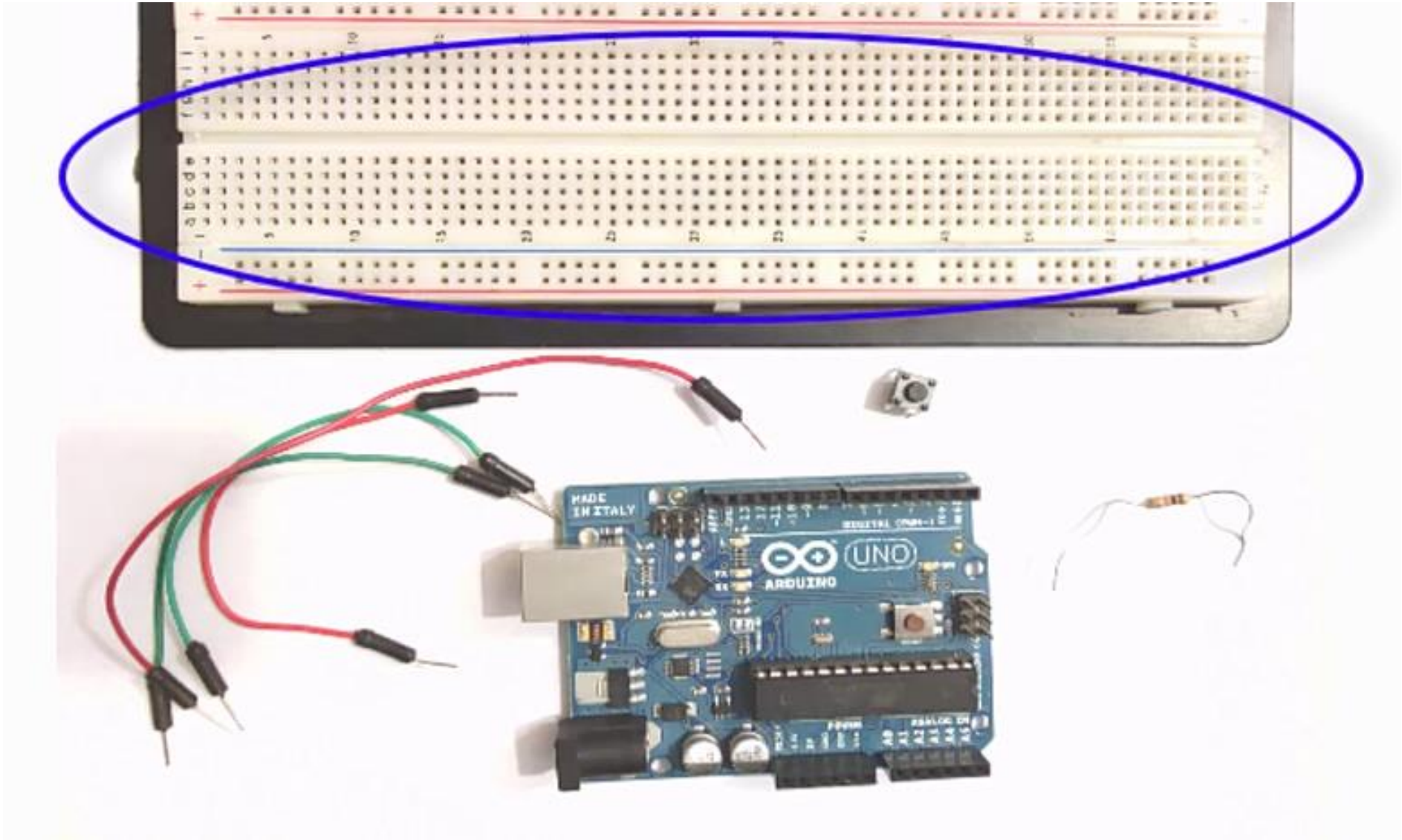
- Resistor de 10 KOhm!



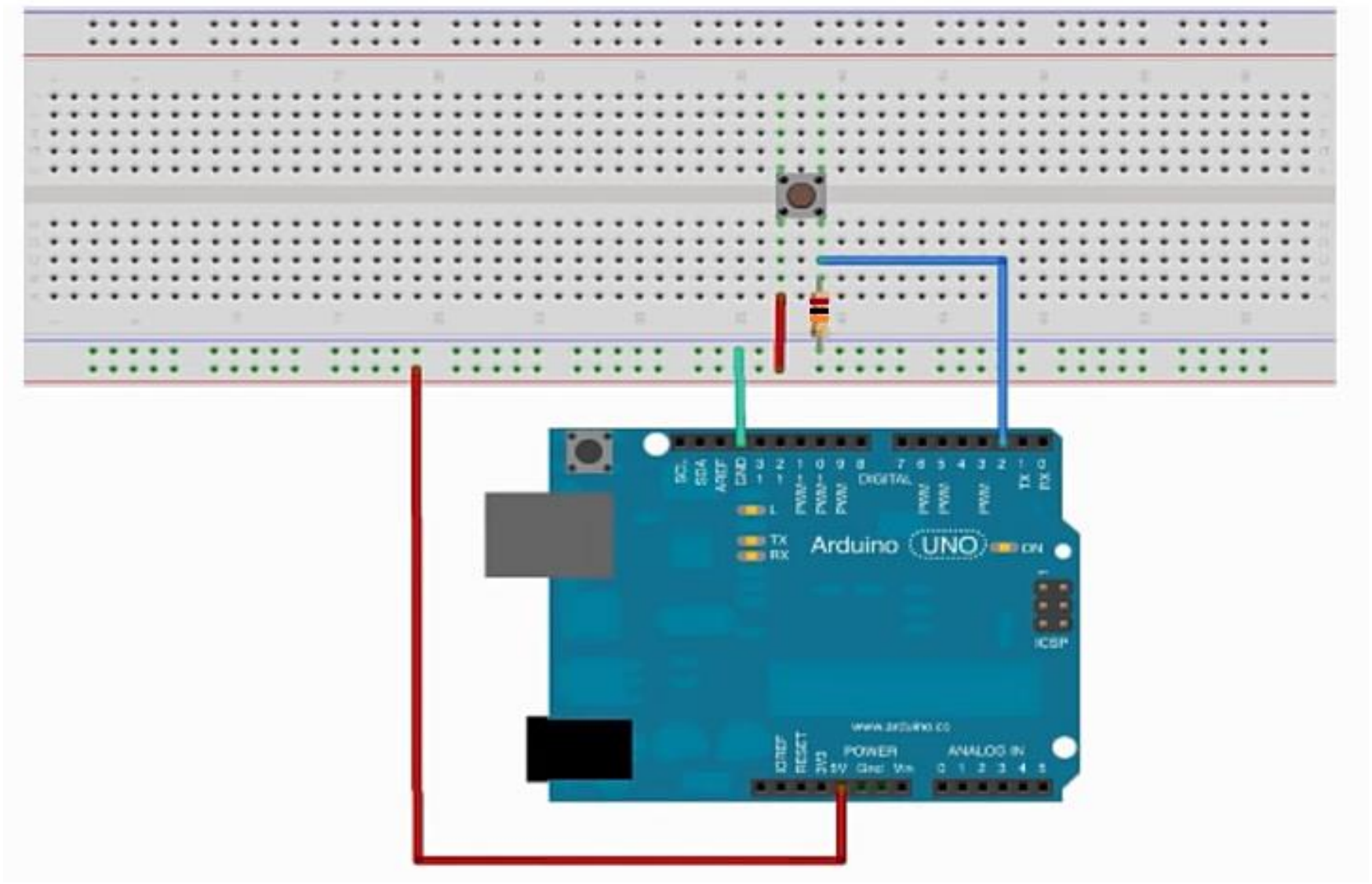
- Cabos de interligações!



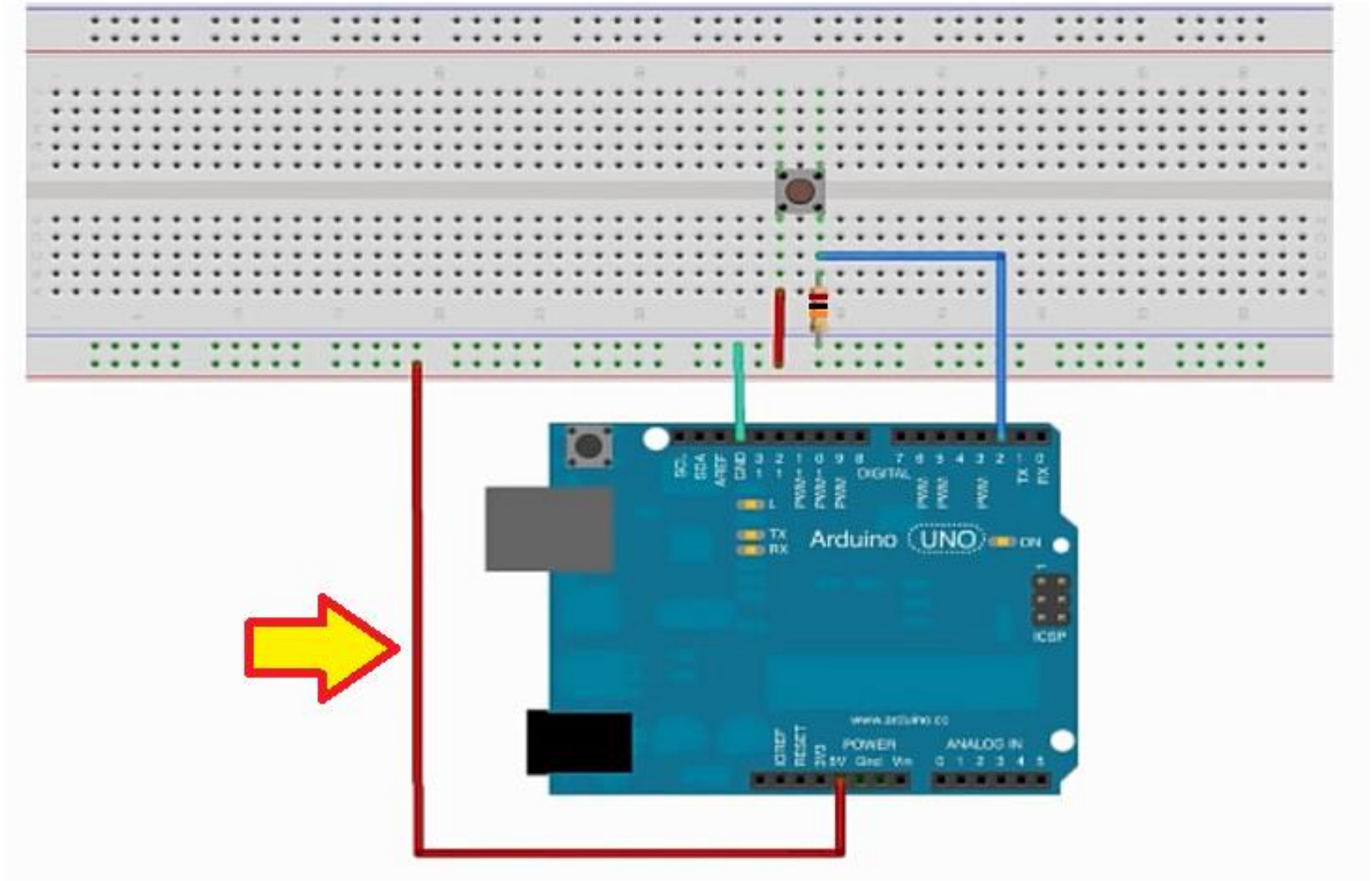
- Uma protoboard!



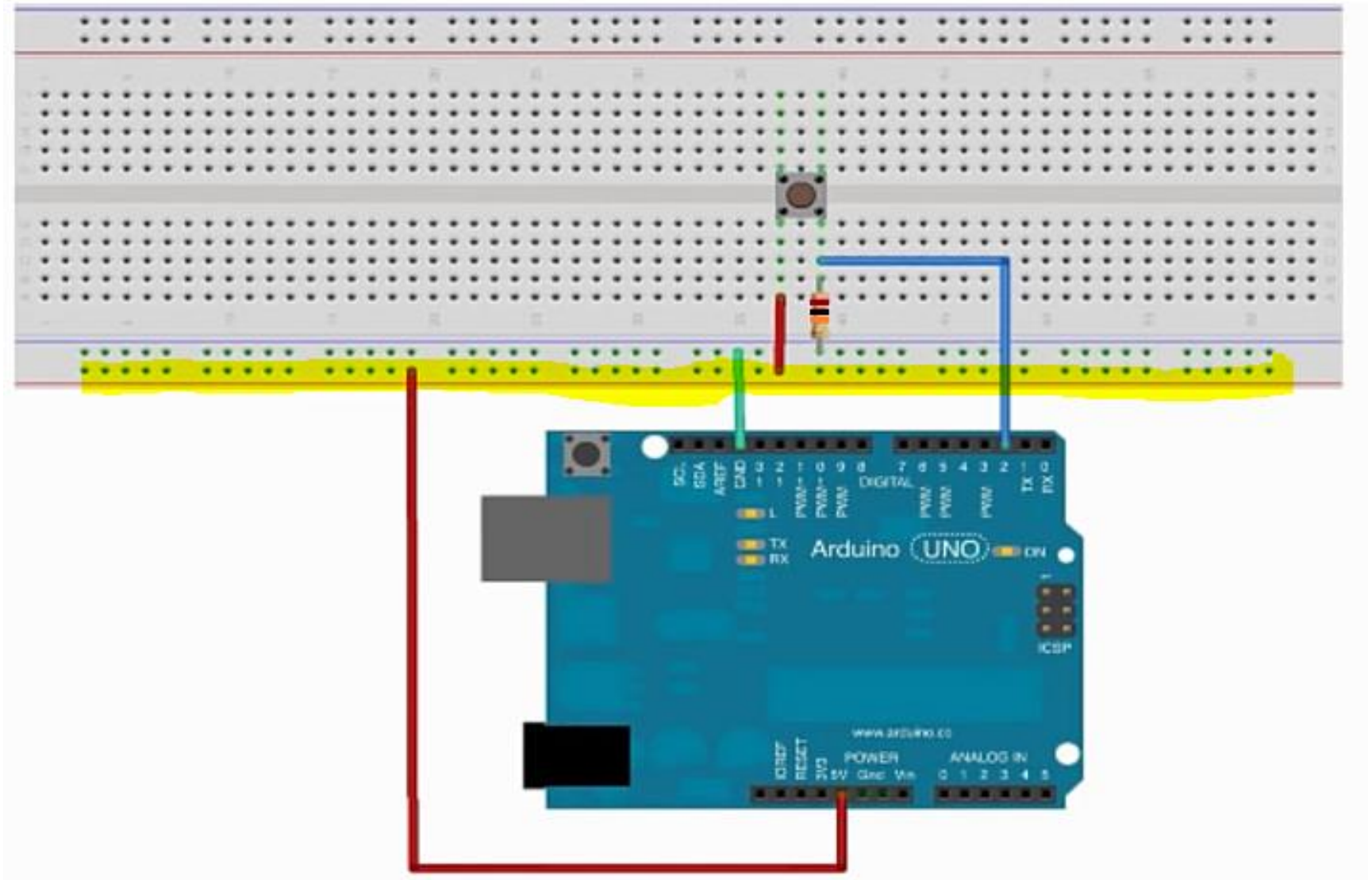
- A interligação é mostrada abaixo!



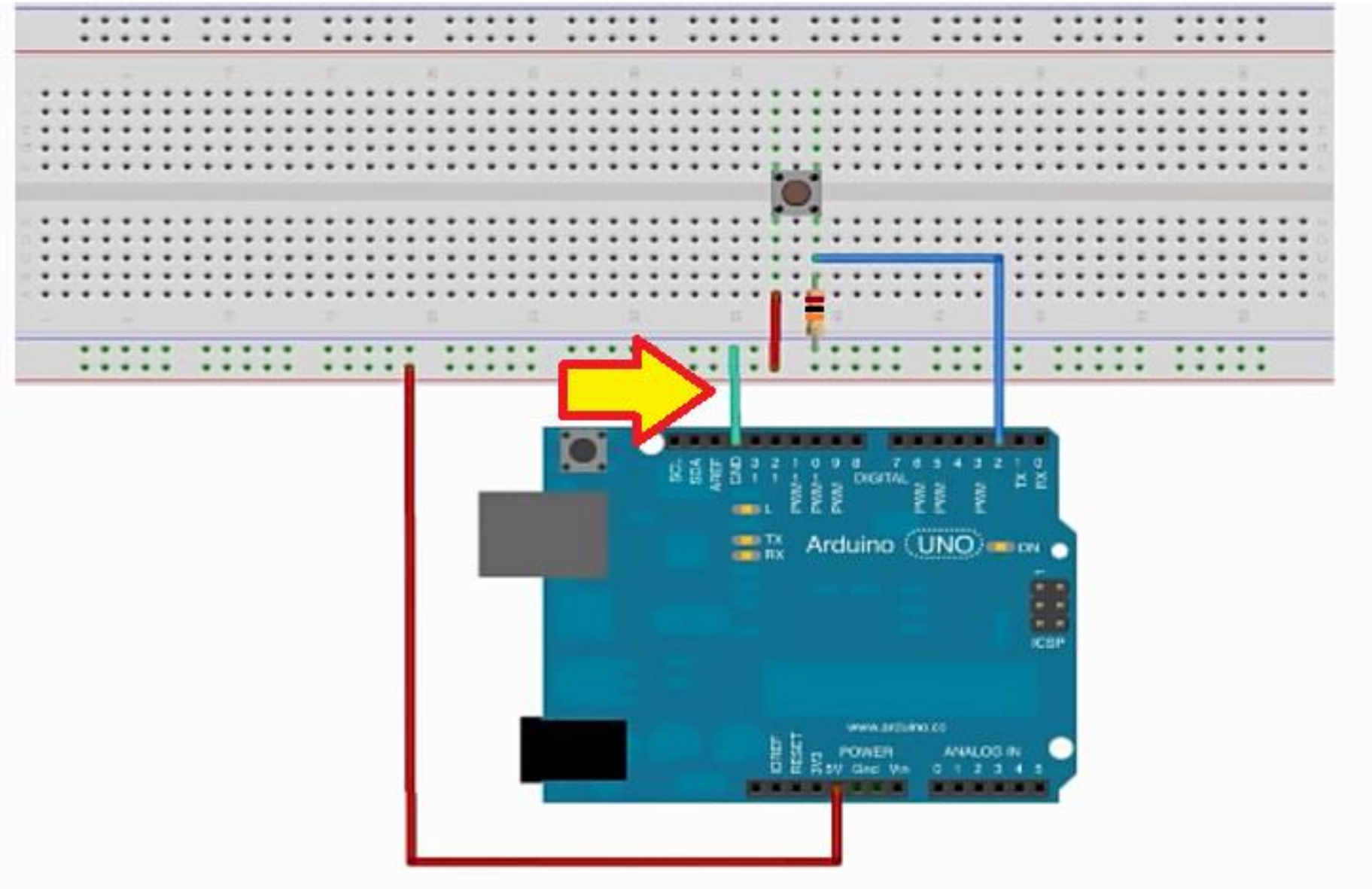
- Conexão da alimentação do +5Vcc!



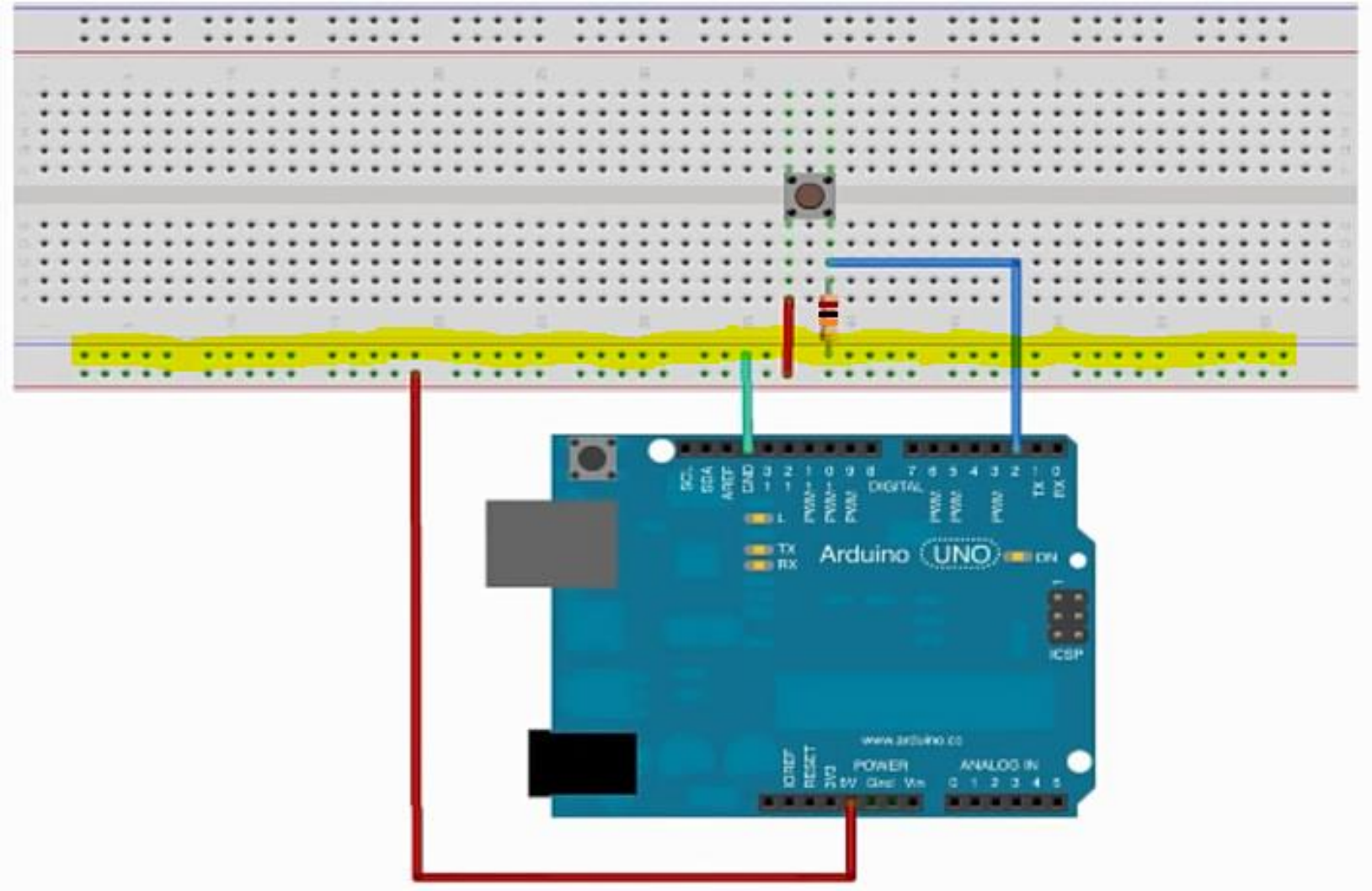
- A alimentação Vcc com cabos vermelhos é comum a todos os pinos da protoboard indicados e alimenta um dos pinos da chave!



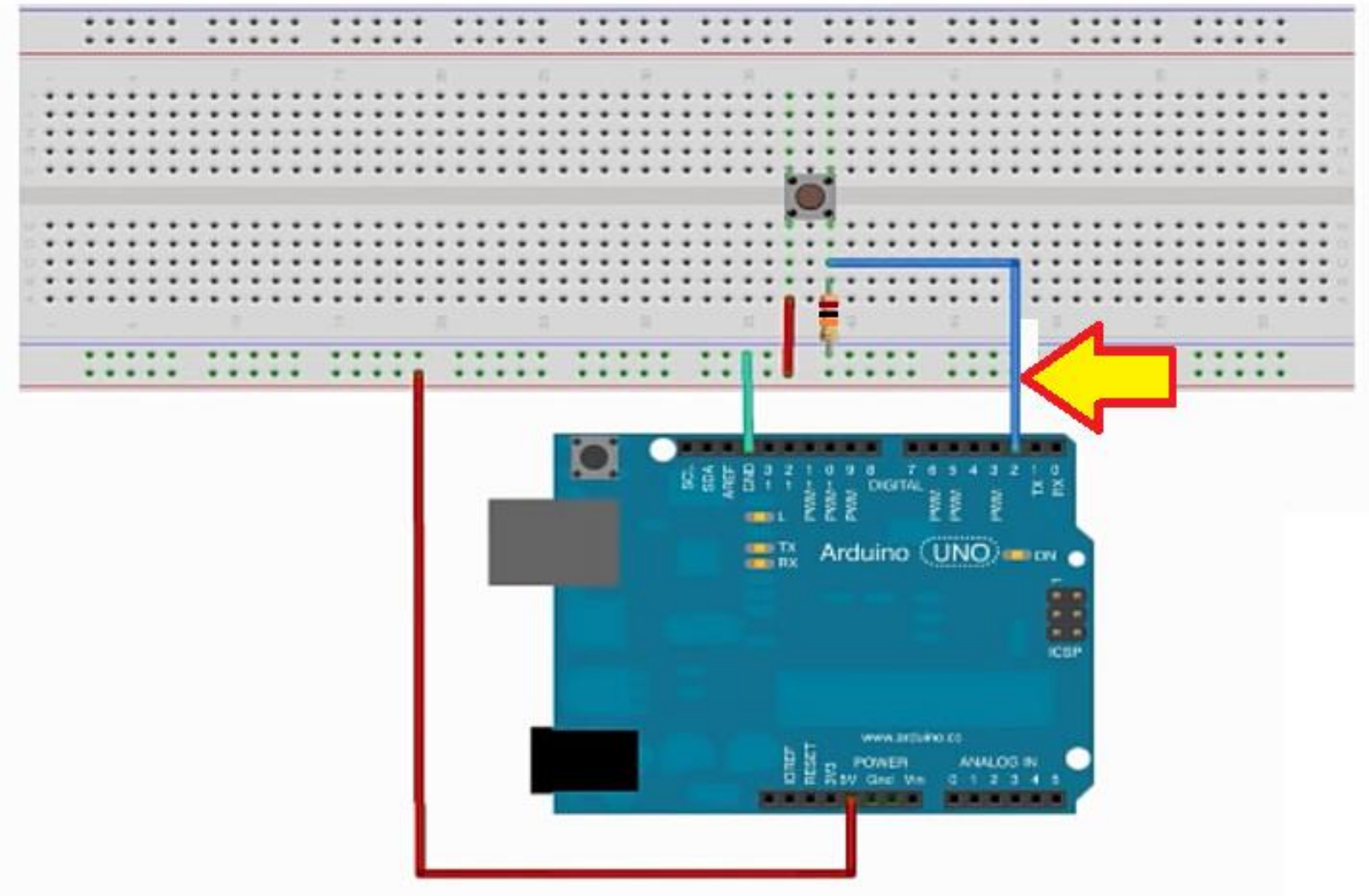
- A ligação do terra é mostrada abaixo!



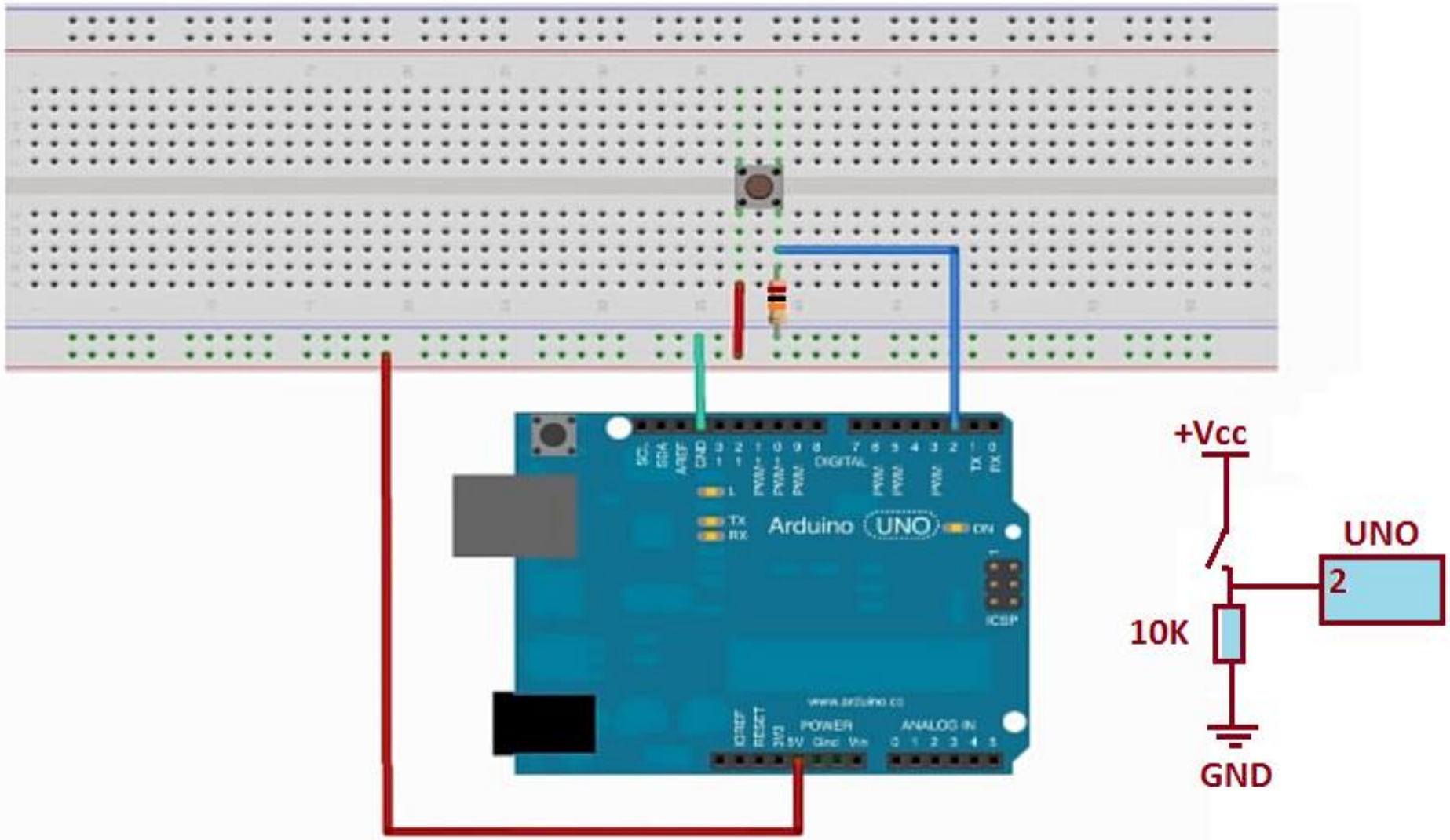
- O resistor de 10 kOhm vai ligado entre a saída da chave e o terra, este tipo de conexão é chamado de pulldown!



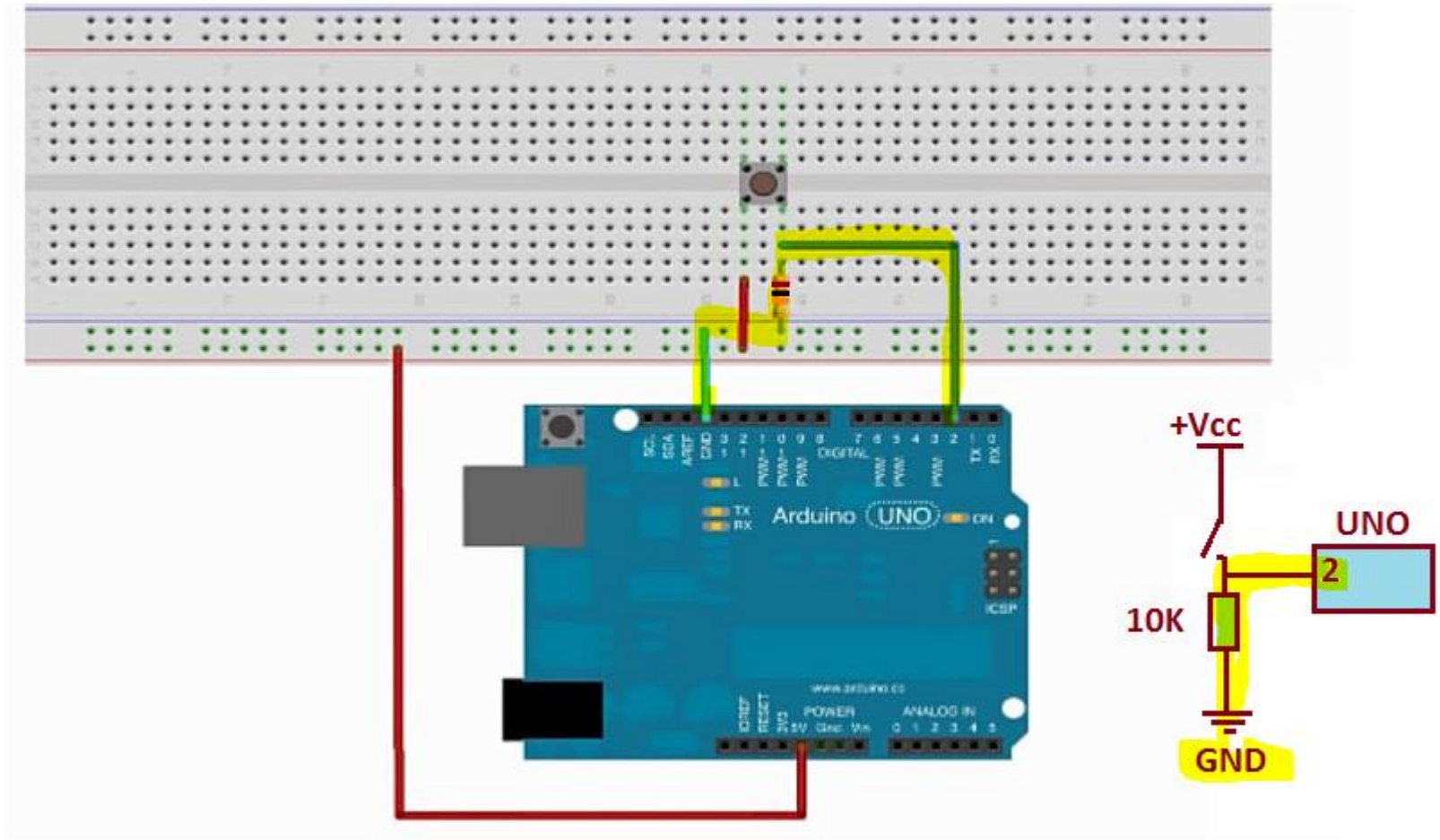
- A ligação da saída da chave é ligada a entrada 2 da placa UNO como indicado abaixo!



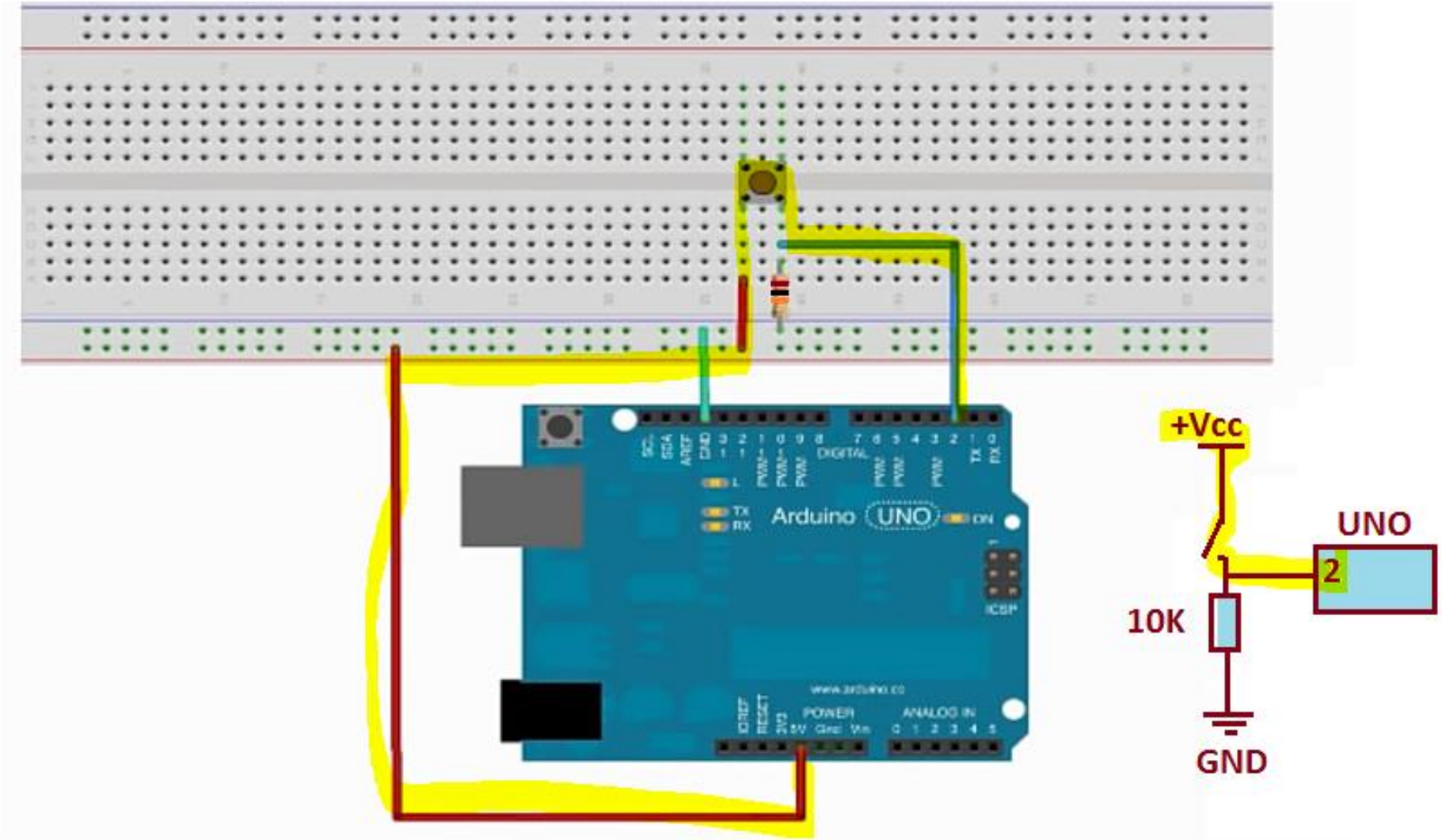
- O diagrama é desenhado ao lado!



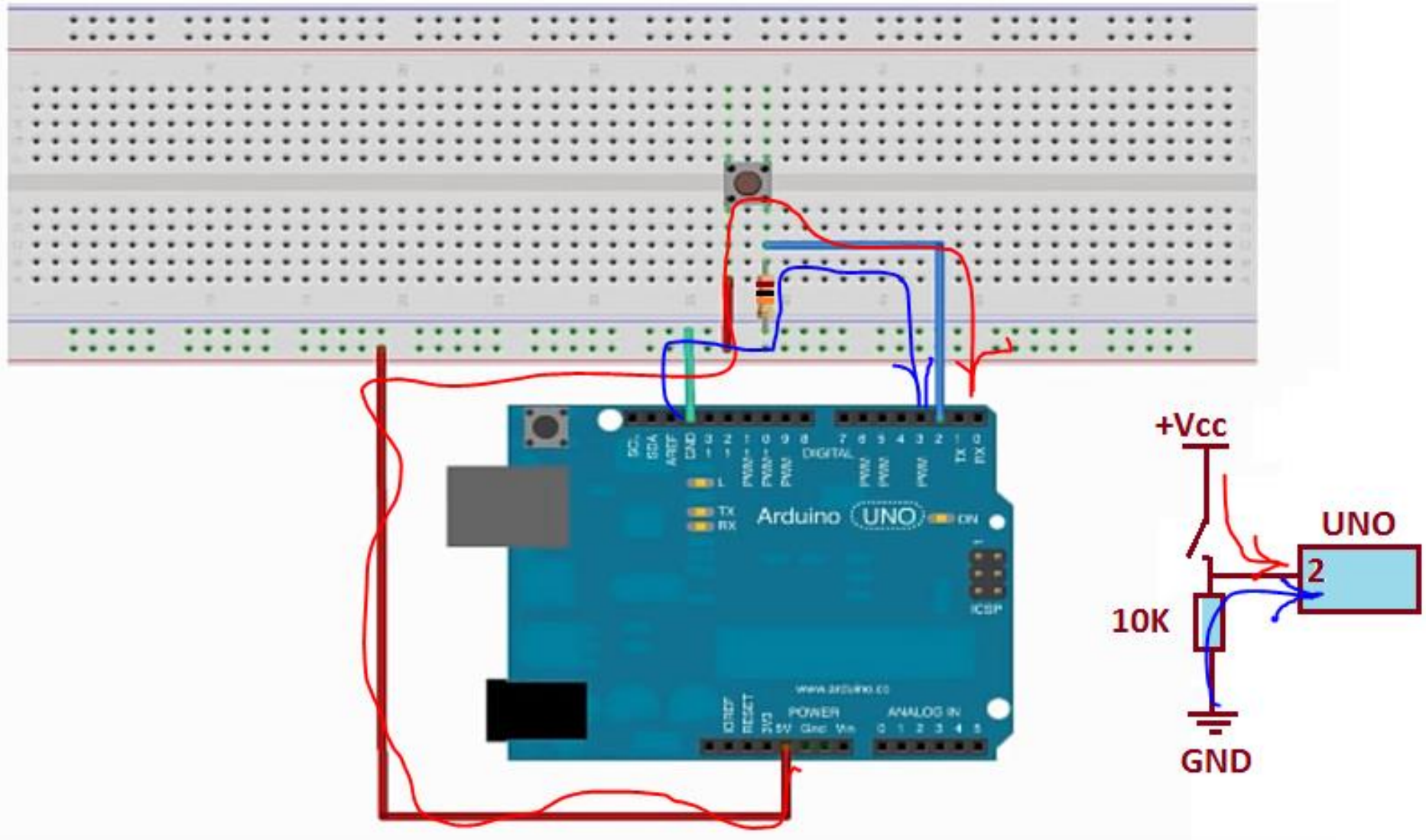
- Quando o botão não está pressionado a entrada 2 da placa está aterrada, isto é, a tensão na entrada é zero volt!



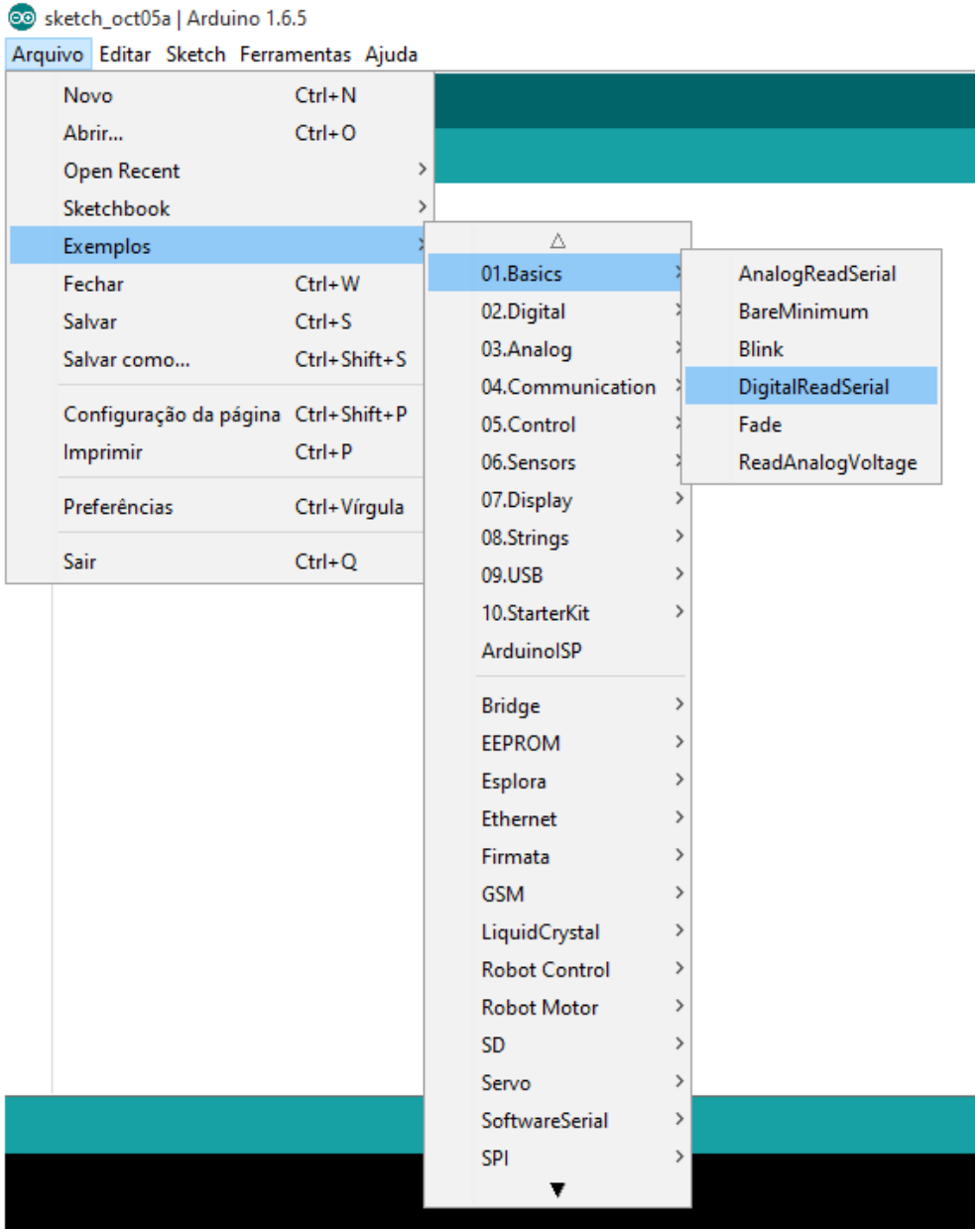
- Quando a chave é pressionada a tensão de 5 Volts é aplicada a entrada 2 da placa do Arduino UNO!



- Quando você pressiona a chave um nível alto é aplicado a entrada do Arduino, quando a chave não está pressionada um nível baixo é aplicado!



- Neste tutorial será usado o exemplo de programa abaixo, Digital Read Serial!



- Neste programa serão analisadas as instruções de entrada digital e comunicação serial!

```
8  // digital pin 2 has a pushbutton attached to it. Give it a name:
9  int pushButton = 2;
10
11 // the setup routine runs once when you press reset:
12 void setup() {
13   // initialize serial communication at 9600 bits per second:
14   Serial.begin(9600);
15   // make the pushbutton's pin an input:
16   pinMode(pushButton, INPUT);
17 }
18
19 // the loop routine runs over and over again forever:
20 void loop() {
21   // read the input pin:
22   int buttonState = digitalRead(pushButton);
23   // print out the state of the button:
24   Serial.println(buttonState);
25   delay(1);        // delay in between reads for stability
26 }
```

- A linha abaixo declara a variável com o endereço do pino de entrada da placa, neste caso será usado o pino 2!

```
  DigitalReadSerial
 Reads a digital input on pin 2, prints the result to the serial m

 This example code is in the public domain.
 */

// digital pin 2 has a pushbutton attached to it. Give it a name
int pushButton = 2;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

- Esta é uma instrução similar a usada no exemplo do LED!

```
  DigitalReadSerial
 Reads a digital input on pin 2, prints the result to the serial m

 This example code is in the public domain.
 */

// digital pin 2 has a pushbutton attached to it. Give it a name
int pushButton = 2;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

- A função setup() tem duas instruções, uma que configura a porta serial e outra que configura a porta como entrada digital!

```
*/

// digital pin 2 has a pushbutton attached to it. Give it a nam
int pushButton = 2;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
```

- A linha abaixo mostra a função que configura a porta serial!

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

- Esta é uma função que faz parte da livraria do Arduino e serve para estabelecer uma comunicação entre a placa e outro PC, ou outros equipamentos!

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

- Neste tutorial esta comunicação se dará com o PC e o cabo de comunicação será o próprio cabo de programação da placa Arduino!

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

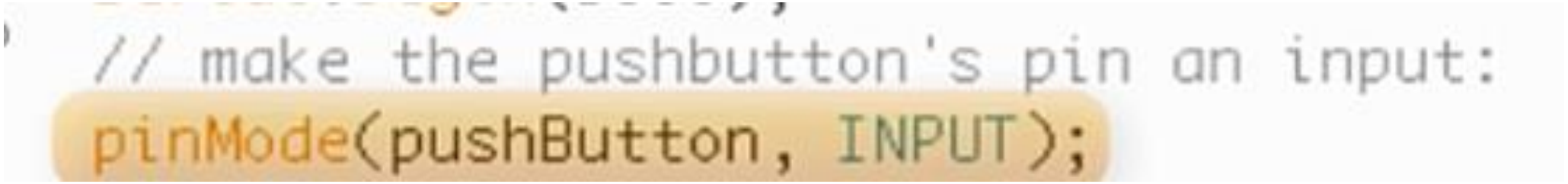
- A instrução serial begin() inicia a comunicação entre a Placa Arduino e o PC, a partir desta instrução dados podem ser transmitidos para o PC ou recebidos do PC!

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

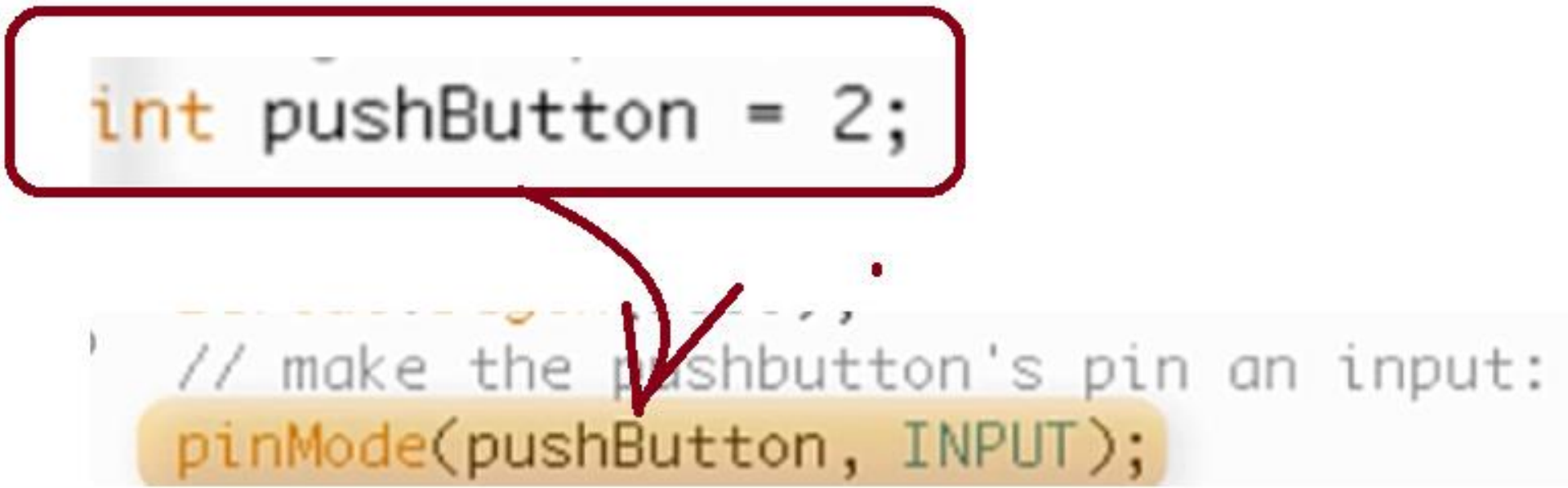
- A função serial.begin tem somente um argumento que é o Baud Rate, este valor deve ser o mesmo da outra porta da comunicação, neste caso foi ajustado para 9600.

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}
```

- A próxima instrução dentro do setup() é a instrução que configura a porta digital, neste caso o botão chamado pushButton
- Está instrução possui dois argumentos!



- No primeiro argumento você deve colocar o número do pino da porta, ou o nome declarado para este pino!



- Na verdade oque o compilador faz é substituir a variável “pushButton” pelo número 2 ao compilar o programa!

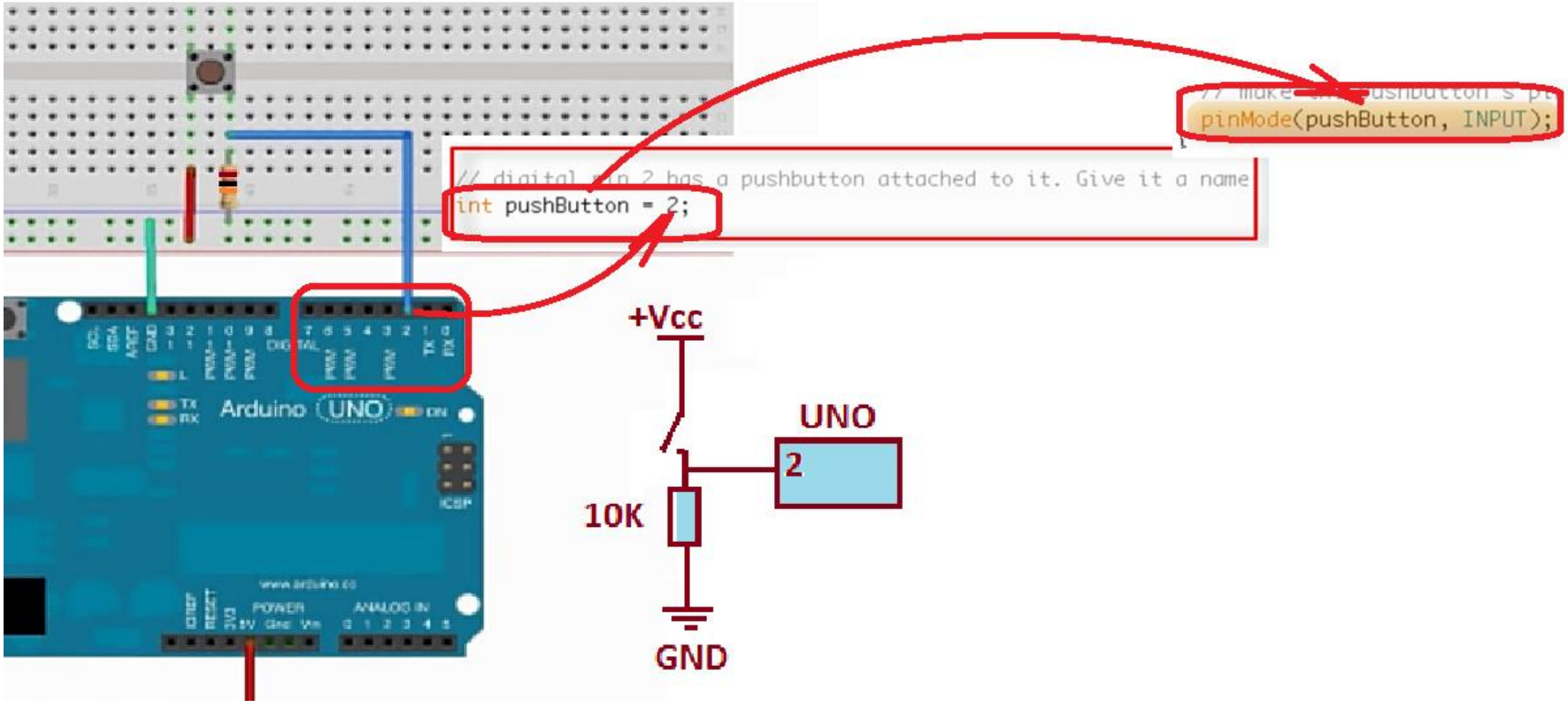
```
pinMode(pushButton, INPUT);  =  pinMode(2,INPUT);
```

- Você pode escrever a instrução das duas maneiras!

```
pinMode(pushButton, INPUT);
```

```
pinMode(2,INPUT);
```

- O primeiro argumento deve ser ajustado para o número do pino de entrada da placa, neste caso a variável pushBotton foi declarada com o valor 2!



- O segundo argumento INPUT ajusta o pino 2 para ser uma entrada digital!

```
// make the pushbutton's pin an input:
pinMode(pushButton, INPUT);
```

- O próximo bloco de código é a função loop()!

```
// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);        // delay in between reads for stability
}
```

- Nesta instrução a variável é declarada com o valor inicial igual ao valor de leitura da porta de entrada!

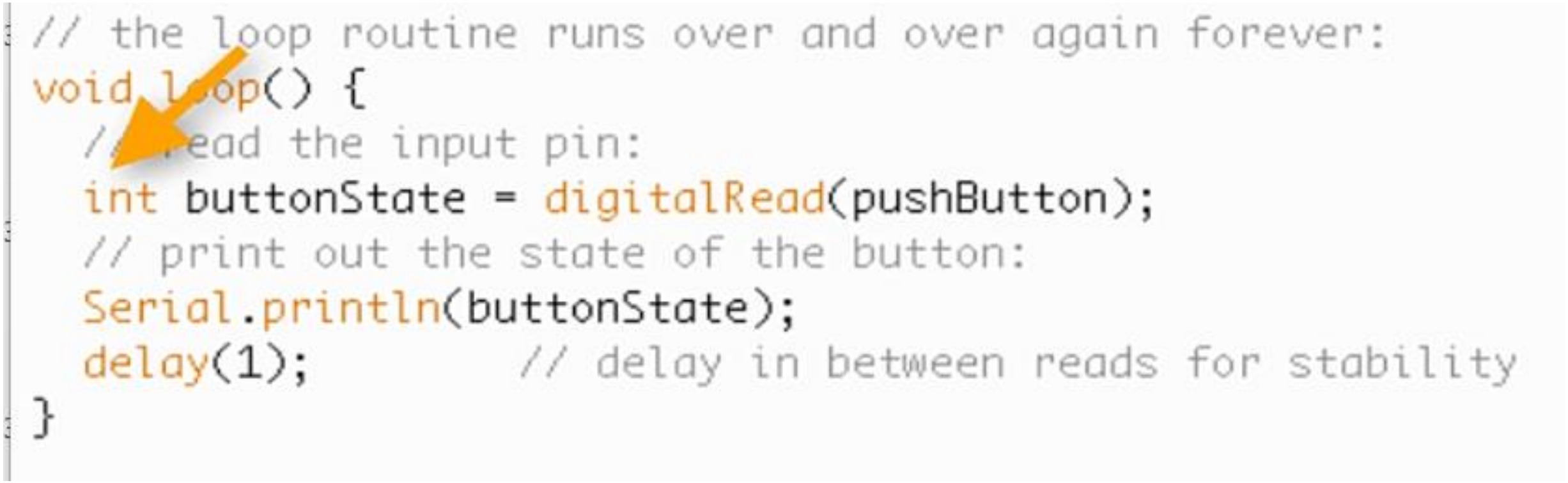
```
// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);        // delay in between reads for stability
}
```

- Você pode declarar e inicializar uma variável a qualquer momento no programa!

```
// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);        // delay in between reads for stability
}
```

- O que você está fazendo na verdade é declarando uma variável com o valor inicial igual ao estado da chave!

- Quando o programa passar por esta instrução a variável vai assumir o valor da entrada digital!



```
// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);        // delay in between reads for stability
}
```

- O valor será “0” se a chave estiver desligada ou “1” se ligada, por isto o tipo da variável deverá ser inteiro!

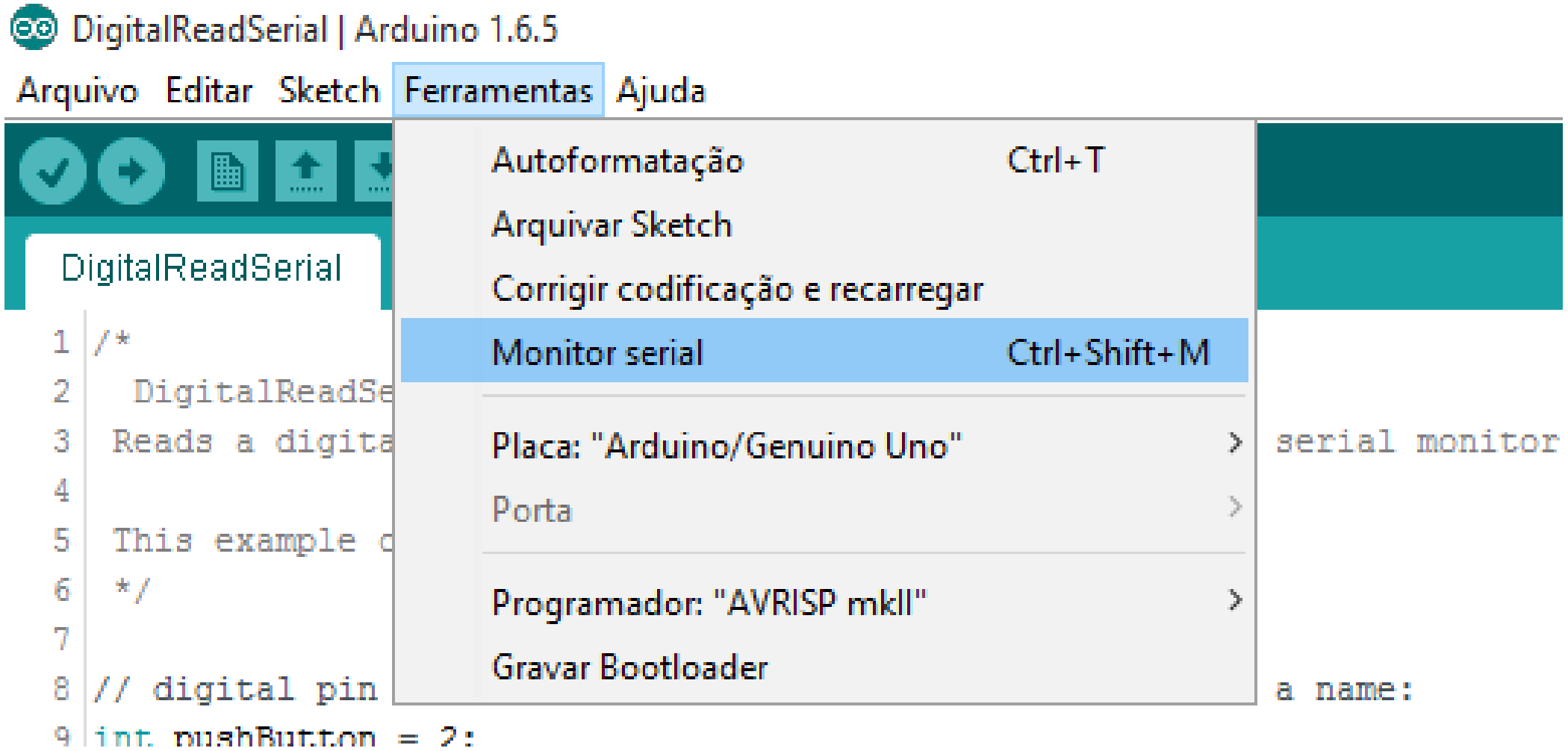
- “0”se a entrada digital for zero volt, isto é, chave NÃO está sendo pressionada!

- Ou “1” se a entrada digital for cinco volts, isto é, chave está sendo pressionada!

- A função serial printline() escreve na serial o valor do argumento, neste caso o valor da chave!

```
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);          // delay in between reads for stability
}
```

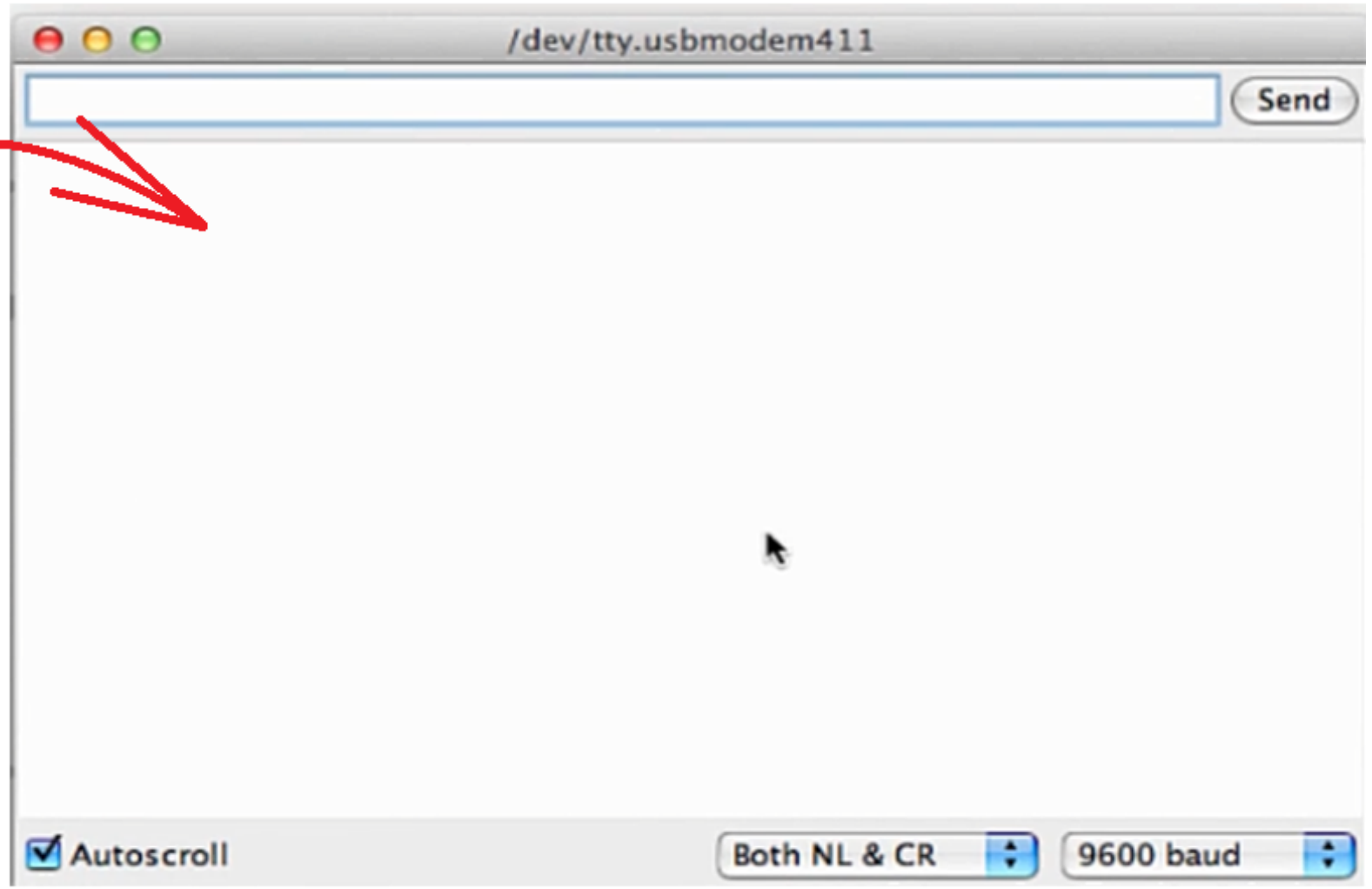
- Para ver este valor no seu programa você deverá selecionar a opção abaixo no menu, que é o componente de Monitor da serial!



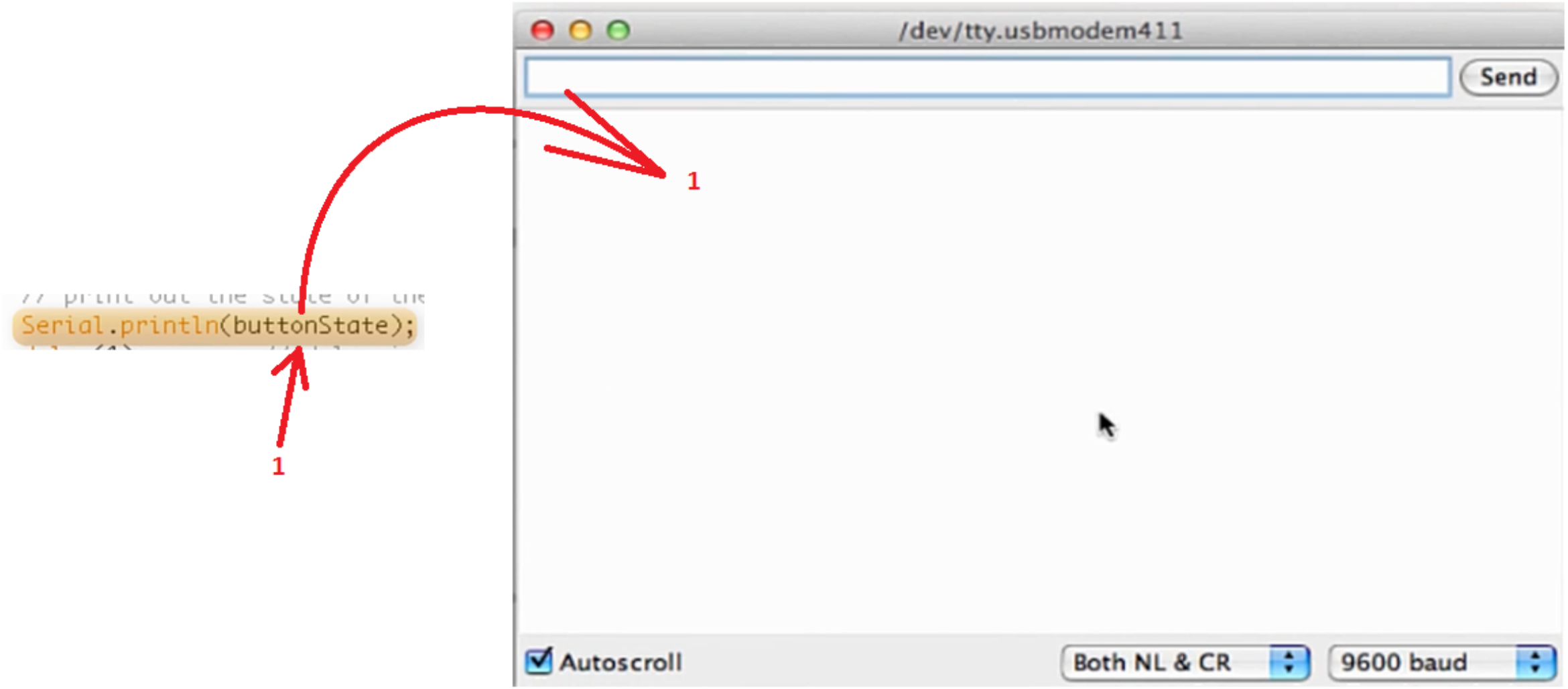
- Se o cabo da placa estiver ligado será aberto uma janela no seu PC similar a da figura abaixo!

- O que a função serial printline() faz é pegar o valor do argumento e mandar para o PC!

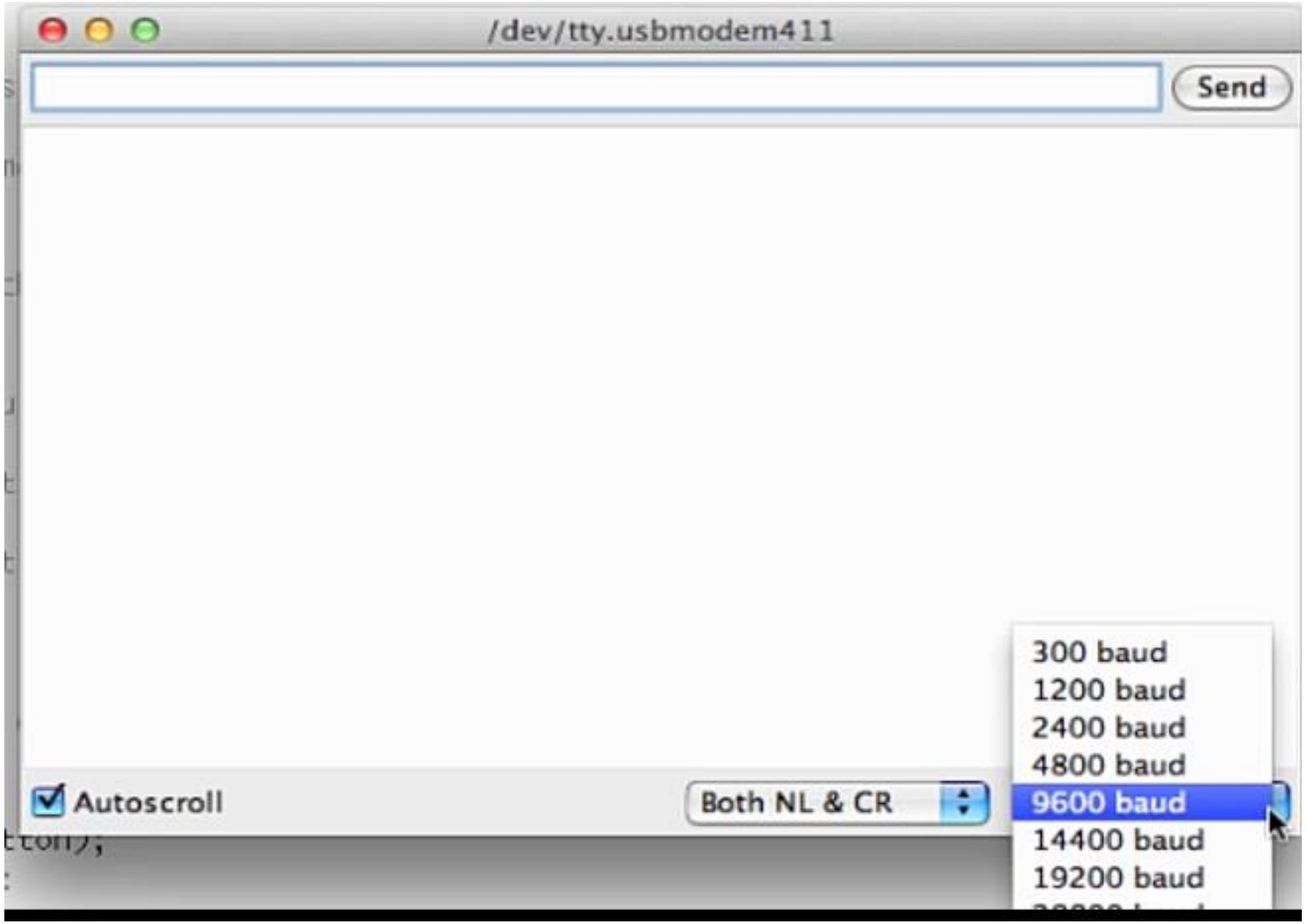
```
Serial.println(buttonState);
```



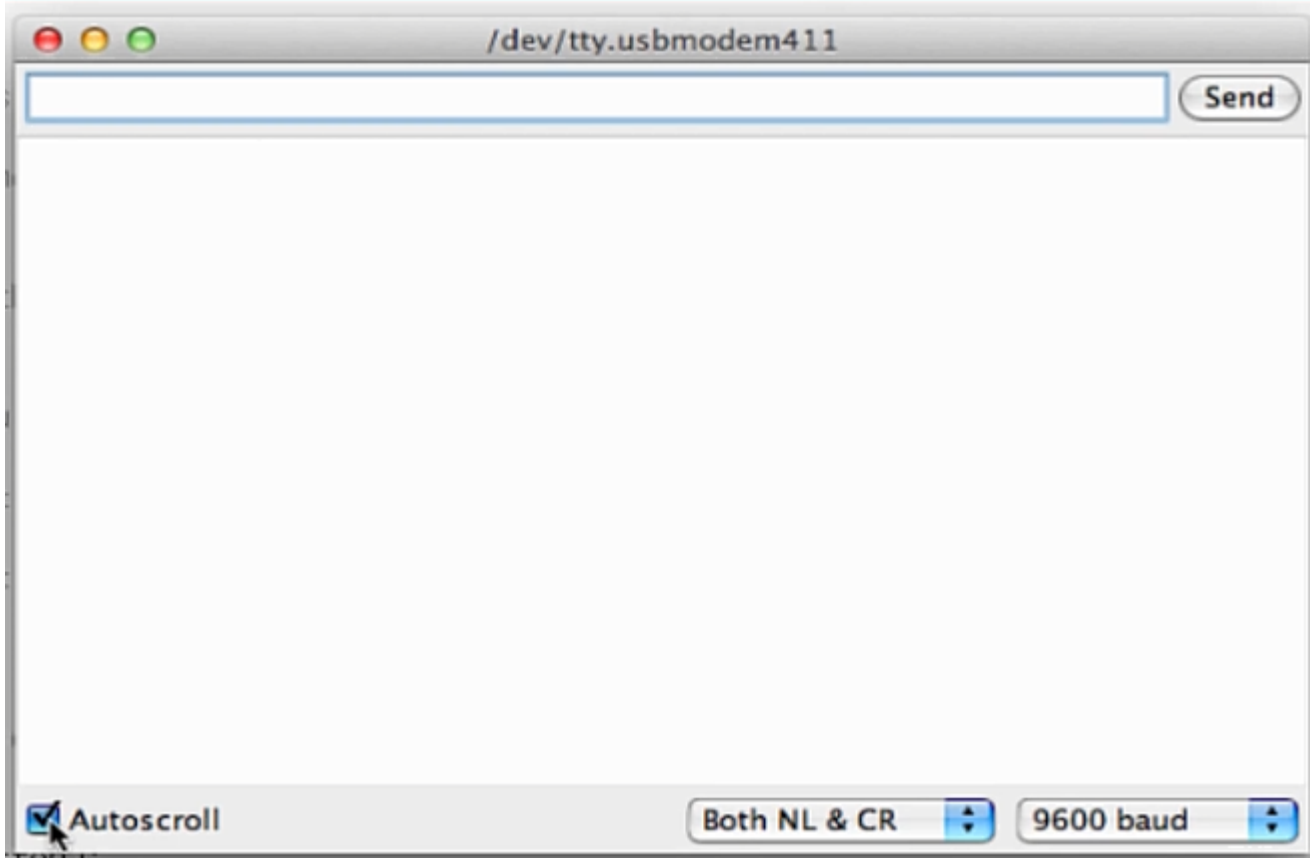
- Se o valor da variável for “1” ou “0” você poderá ver na janela do monitor da serial!



- Observe que na janela do monitor da serial você pode ajustar o BaudRate para o valor da sua instrução no programa!



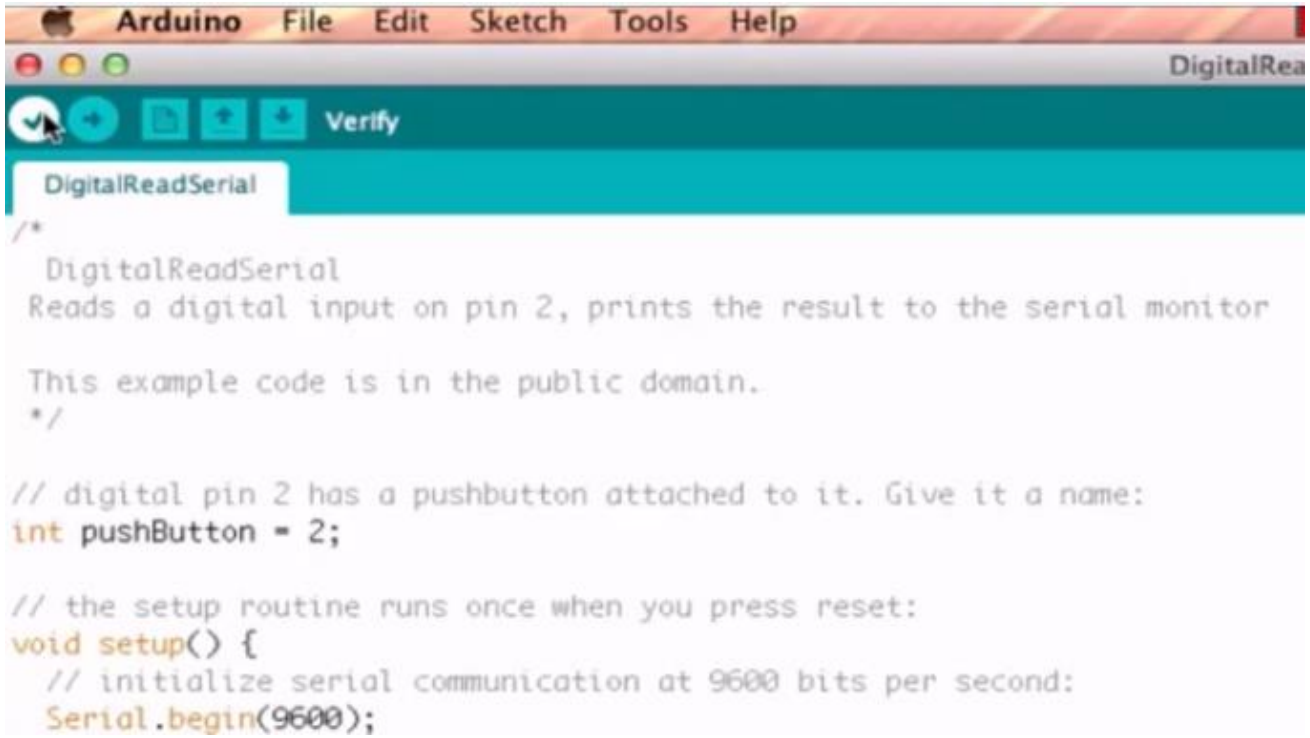
- O botão AutoScroll habilita a rolagem da tela!



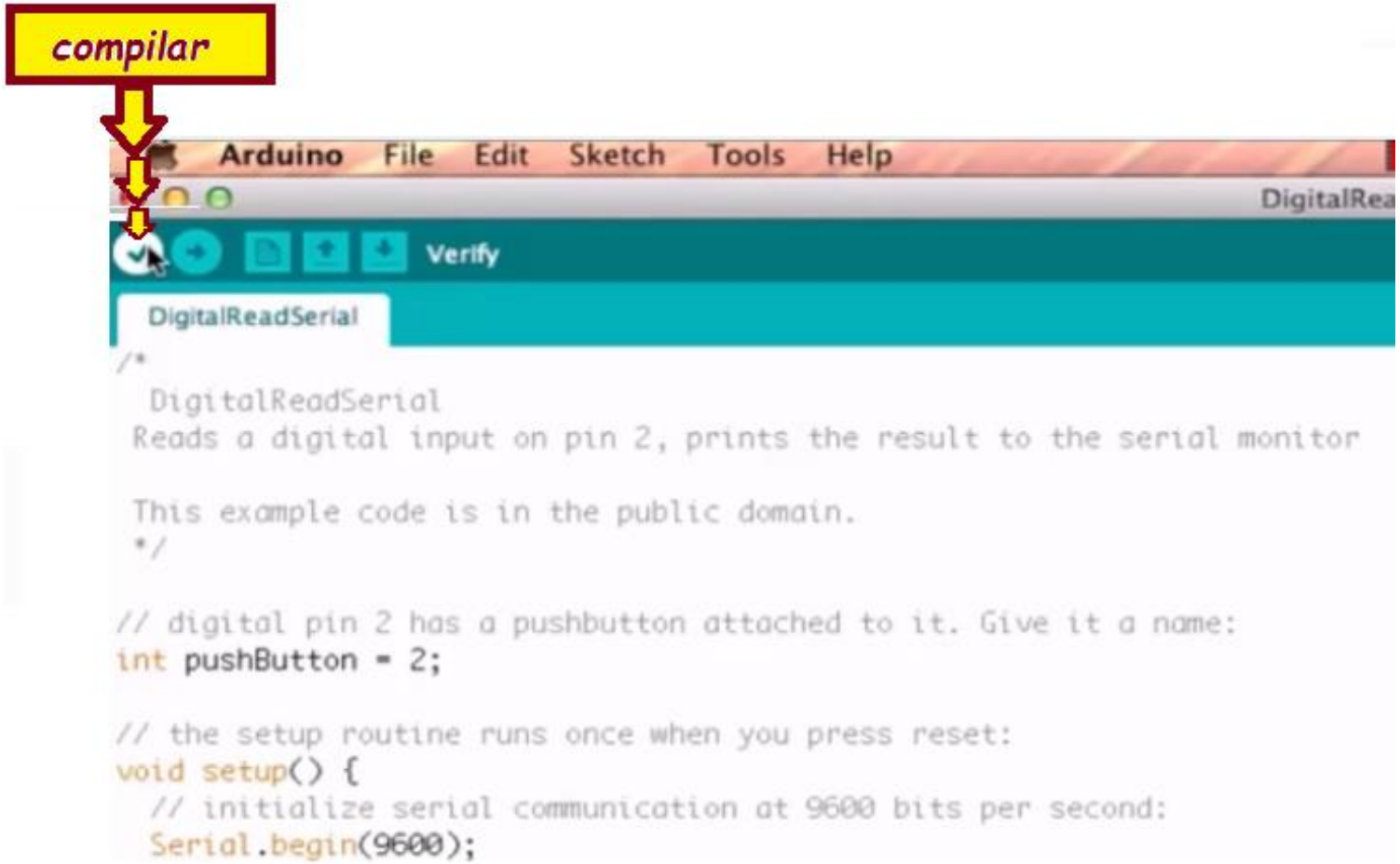
- A última instrução é um atraso, o objetivo desta instrução é estabilizar a leitura da chave!

```
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);        // delay in between reads for stability
}
```

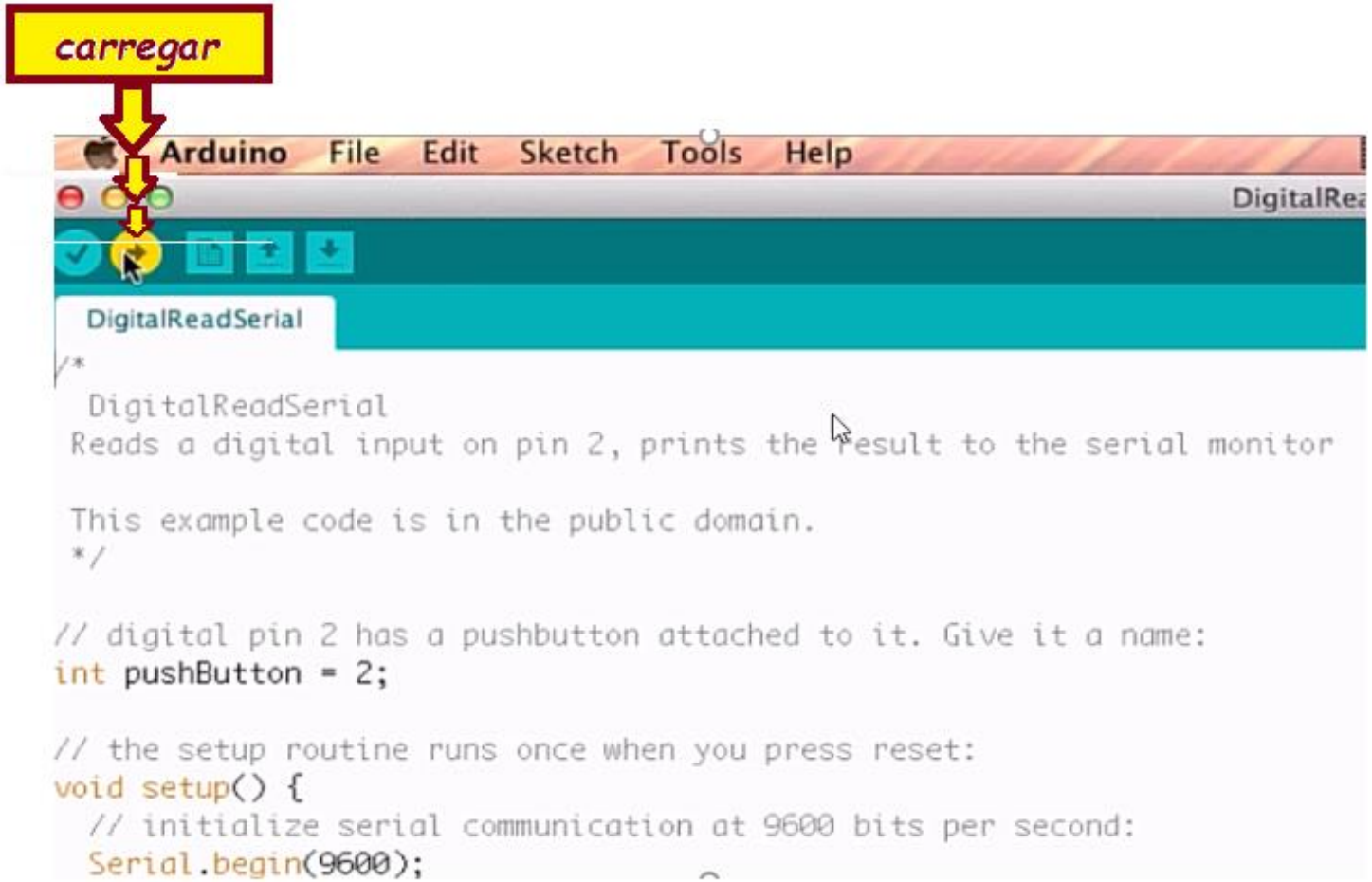
- A última instrução é um atraso, o objetivo desta instrução é estabilizar a leitura da chave!



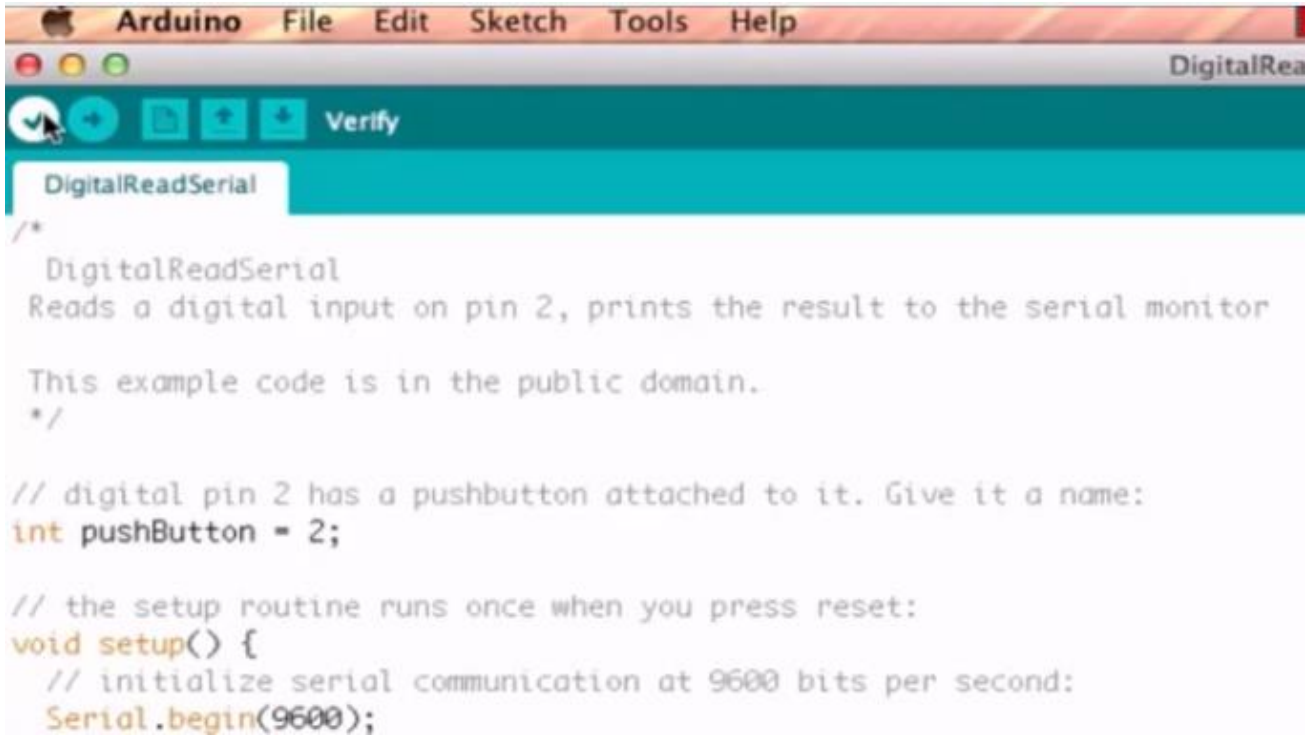
- Para testar, compile!



- E carregue o programa na placa Arduino Uno!



- Agora é só verificar o programa funcionando!



- No próximo tutorial você verá como ler um sinal analógico!