

# **Programação do Arduino**

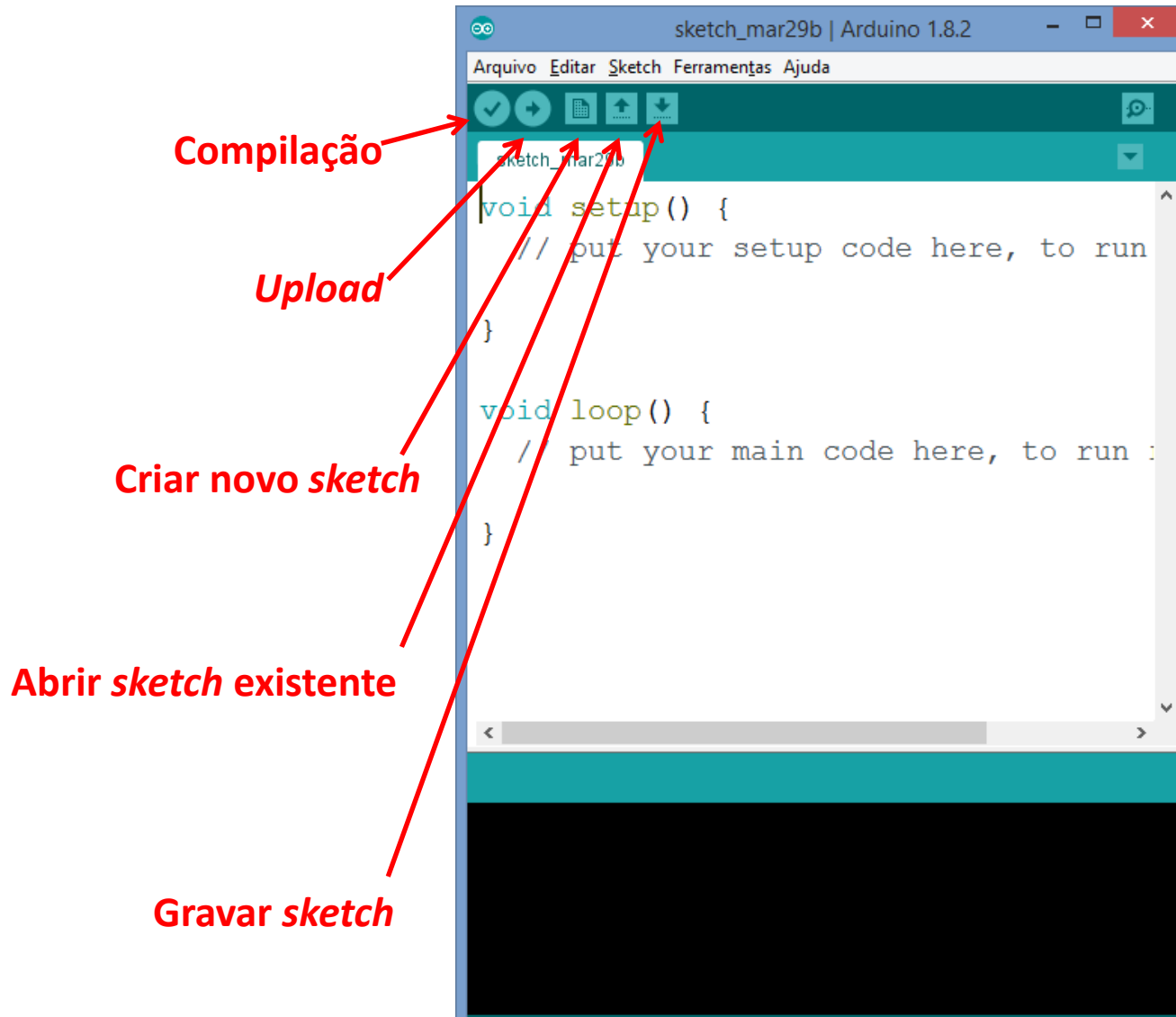
## **Parte 1**

### **IDE Wiring**

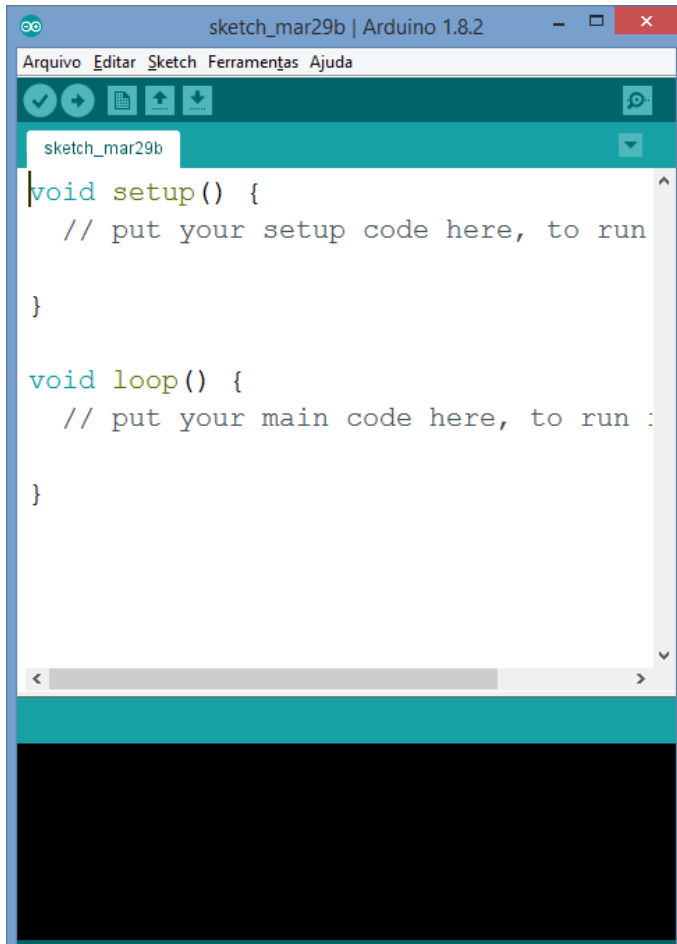
# Ambiente de programação: IDE *Wiring*



# Ambiente de programação: IDE *Wiring*



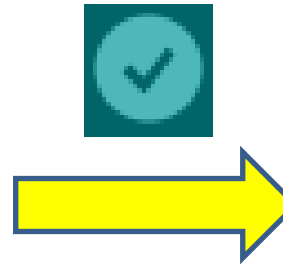
# Ambiente de programação: IDE Wiring



```
sketch_mar29b | Arduino 1.8.2
Arquivo Editar Sketch Ferramentas Ajuda
sketch_mar29b
void setup() {
  // put your setup code here, to run
}

void loop() {
  // put your main code here, to run
}
```

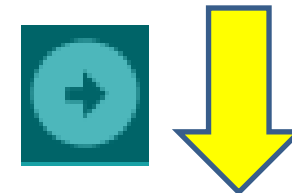
**Código fonte (*sketch*)**



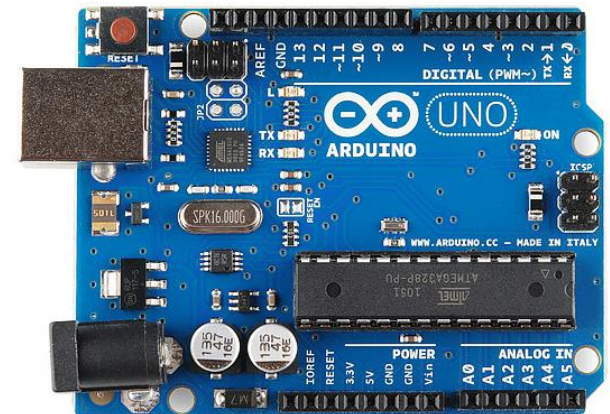
**Compilação**



**Código binário**

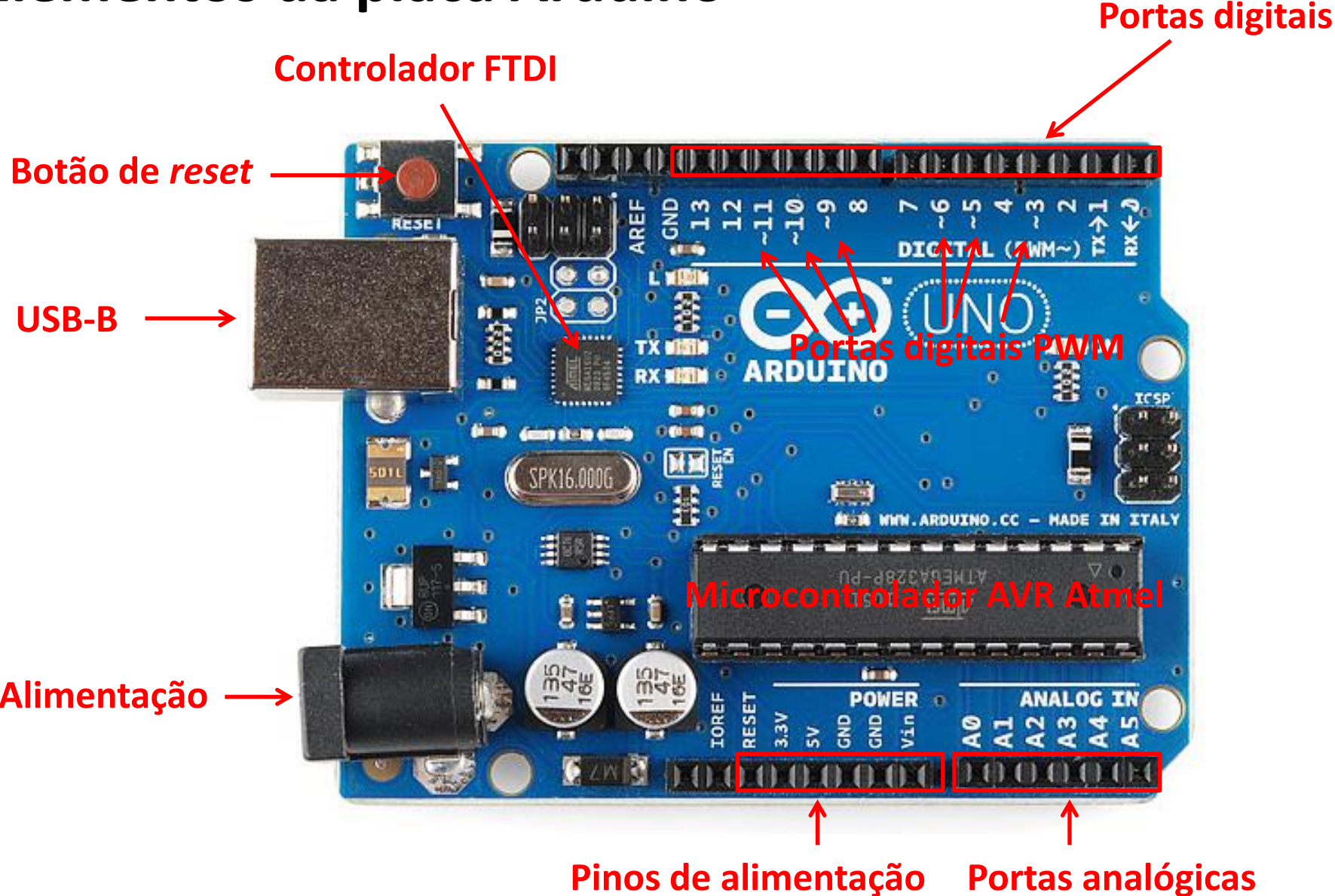


**Upload**



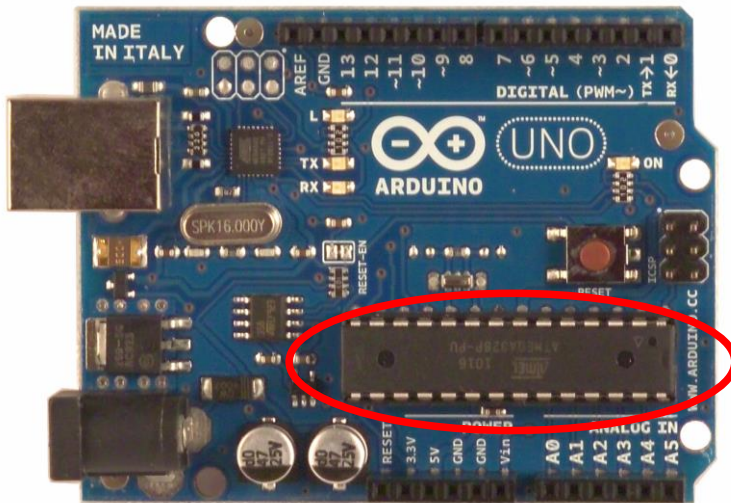
**Arduino**

# Elementos da placa Arduino



# O que é o Arduino?

- O Arduino é uma plataforma de prototipagem de eletrônicos de código aberto
  - Todos os diagramas e fontes de programação estão disponíveis sob licenças livres



Microcontrolador AVR  
Atmega328 da Atmel

O Arduino Uno é baseado no Atmega328 e contém pinos digitais de entrada e saída, entradas analógicas. A conexão USB é realizada por um chip separado

# Cuidados

## ▣ Antes de começar:

- ▣ Limitações do Arduino:
  - ▣ Corrente máxima: 500 mA total, 50 mA por porta
- ▣ Certifique-se que seu circuito não requer mais corrente do que o Arduino pode oferecer
- ▣ Cuidado com curto-circuitos, pode queimar o microcontrolador
- ▣ Sempre desligue o Arduino quando for mexer no circuito eletrônico

# Entrada e saída digital

- ▣ Entradas e saídas digitais são portas programáveis para leitura ou "gravação" de um sinal digital (0 ou 1 – ligado ou desligado – zero ou 5 volts, HIGH e LOW)
  - ▣ Entrada: botão pressionado, porta aberta...
  - ▣ Saída: ligar e desligar lâmpadas, motores... (interruptor)



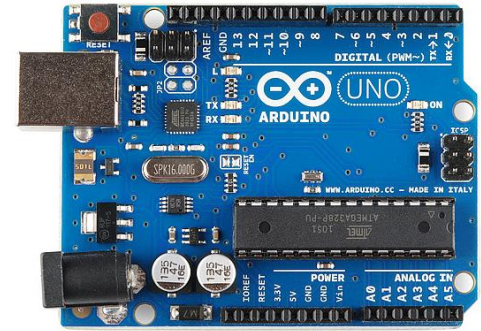
# Comunicação serial



010010001101...



Porta USB

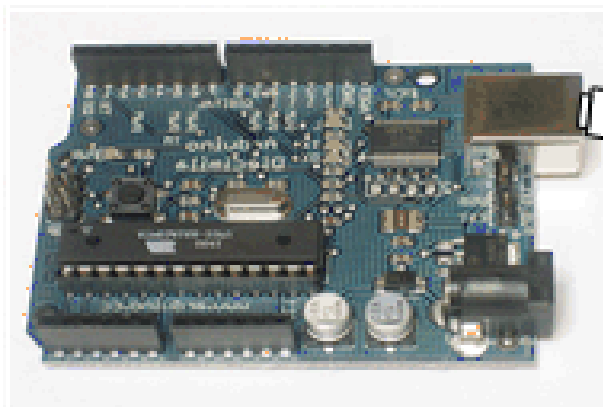


# Comunicação serial



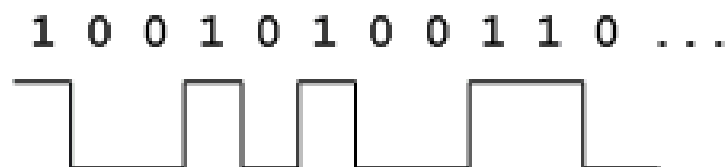
*Notebook*

Cabo USB



Arduino

A informação passa entre o Arduino e o computador por meio do cabo USB. A informação é transmitida como dígitos binários (bits) zeros ('0') e uns ('1').



# Comunicação serial

Permite receber e enviar informações entre o Arduino e um computador

- ▣ Deve ser configurado na função setup:
  - ▣ `Serial.begin(9600); // Inicializa a porta serial para uma taxa de 9600 baud`
- ▣ Dados são enviados para o PC pelo comando:
  - ▣ `Serial.println("Olá Arduino");`
- ▣ Na IDE utilize o monitor serial

# Meu primeiro *sketch* Arduino

```
/* olaArduino
   Primeiro sketch Arduino
*/

void setup() {
    Serial.begin(9600);
    Serial.println("Ola Arduino!");
}

void loop() {

}
```

# Gravando o *sketch* Arduino

The screenshot shows the Arduino IDE interface with the following elements:

- IDE Window:** Title bar "aloArduino | Arduino 1.8.2". Menu bar: "Arquivo", "Editar", "Sketch", "Ferramentas", "Ajuda". Toolbar with icons for check, undo, redo, compile, upload, and download. The sketch editor contains the following code:

```
/*  olaArduino.ino
   Meu primeiro sketch
*/

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

void loop() {
  Serial.println("Alo Arduino!");
}
```
- Save Dialog:** Title "Salvar a pasta de sketches como...". "Salvar em:" dropdown set to "Aulas". A table of files in the folder:

Nome	Data de modificaç...	Tipo	Tarr
aloArduino	29/03/2017 09:46	Pasta de arquivos	
blinkLED13	29/03/2017 10:27	Pasta de arquivos	

The "Nome:" field is set to "aloArduino" and the "Tipo:" dropdown is set to "Todos os Arquivos (\*.\*)". Buttons for "Salvar" and "Cancelar" are visible.
- Status Bar:** Shows "10" on the left and "Arduino/Genuino Uno em COM4" on the right.

# Executando o *sketch* Arduino



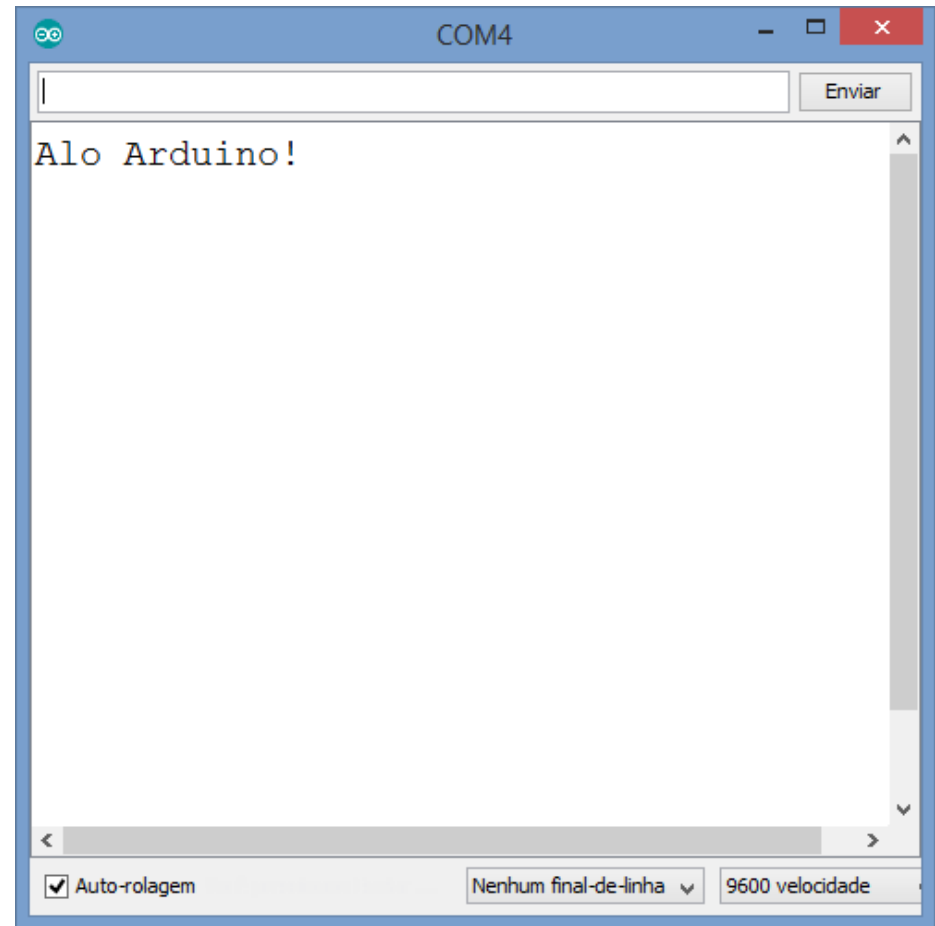
The screenshot shows the Arduino IDE editor window titled 'aloArduino | Arduino 1.8.2'. The menu bar includes 'Arquivo', 'Editar', 'Sketch', 'Ferramentas', and 'Ajuda'. The toolbar contains icons for saving, undo, redo, and other functions. The sketch name 'aloArduino \$' is displayed in the top left. The code in the editor is as follows:

```
/* olaArduino.ino
   Meu primeiro sketch
 */

void setup() {
  Serial.begin(9600);
  Serial.println("Alo Arduino!");
}

void loop() {
}
```

At the bottom of the editor, a status bar indicates 'Carregado.' and provides memory usage information: 'O sketch usa 1438 bytes (4%) de espaço' and 'Variáveis globais usam 198 bytes (9%) d'. The bottom right corner shows 'Arduino/Genuino Uno em COM4'.



The screenshot shows the serial monitor window titled 'COM4'. It features a text input field at the top with an 'Enviar' button. The main area displays the output 'Alo Arduino!'. At the bottom, there are settings for 'Auto-rolagem' (checked), 'Nenhum final-de-linha', and '9600 velocidade'.

**Janela do monitor serial**

# Executando a instrução `Serial.println` no *loop*

```
/* olaArduino.ino
   Primeiro sketch Arduino
*/

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

void loop() {
    Serial.println("Ola Arduino!");
}
```

# Executando o *sketch* Arduino



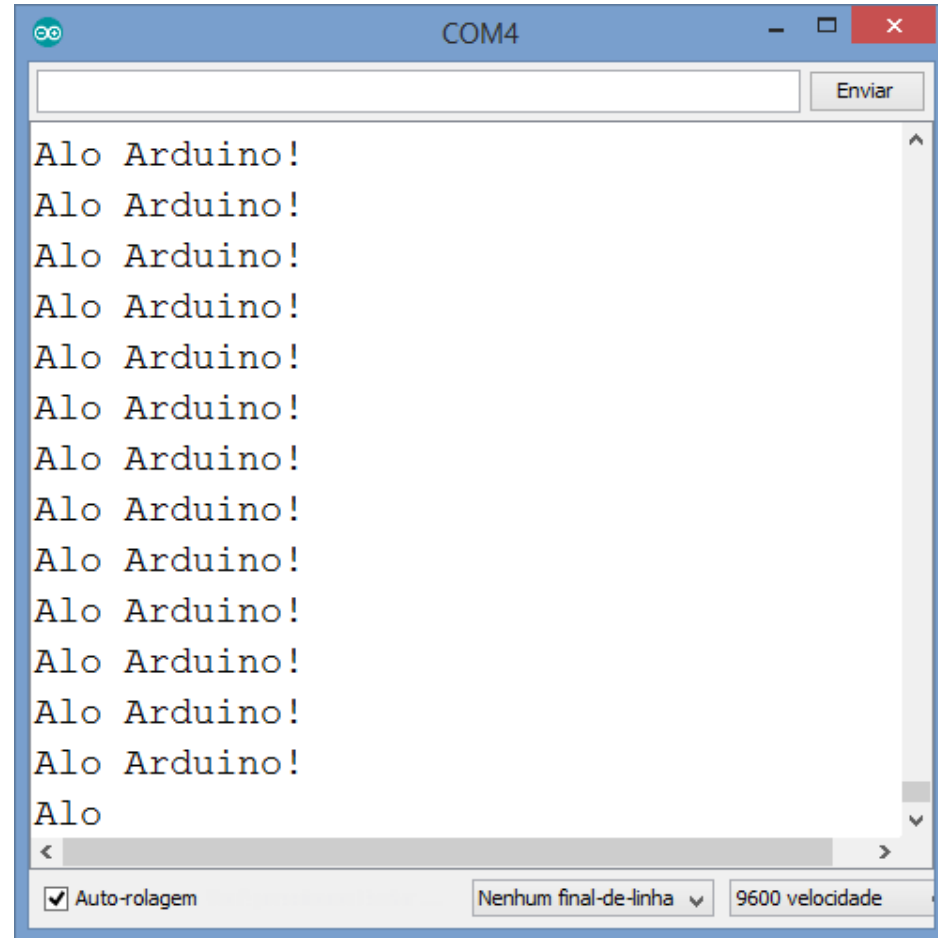
The screenshot shows the Arduino IDE interface. The title bar reads "aloArduino | Arduino 1.8.2". The menu bar includes "Arquivo", "Editar", "Sketch", "Ferramentas", and "Ajuda". The toolbar contains icons for saving, running, uploading, and downloading. The main editor displays the following code:

```
/*  olaArduino.ino
    Meu primeiro sketch
 */

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

void loop() {
  Serial.println("Alo Arduino!");
}
```

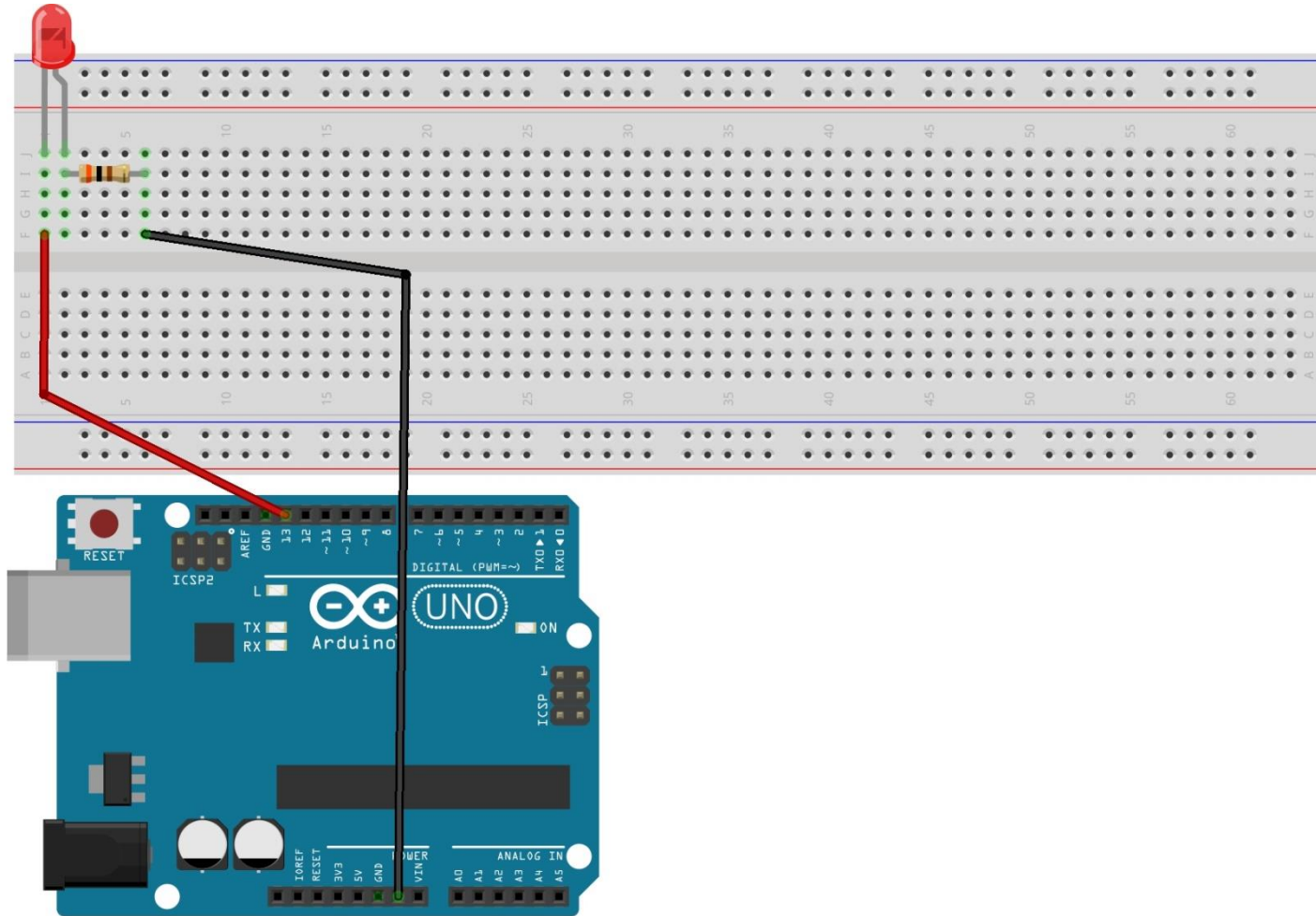
Below the code editor, a status bar indicates "Carregado." and provides memory usage information: "O sketch usa 1438 bytes (4%) de espaço" and "Variáveis globais usam 198 bytes (9%) d". At the bottom, the status bar shows "10" and "Arduino/Genuino Uno em COM4".



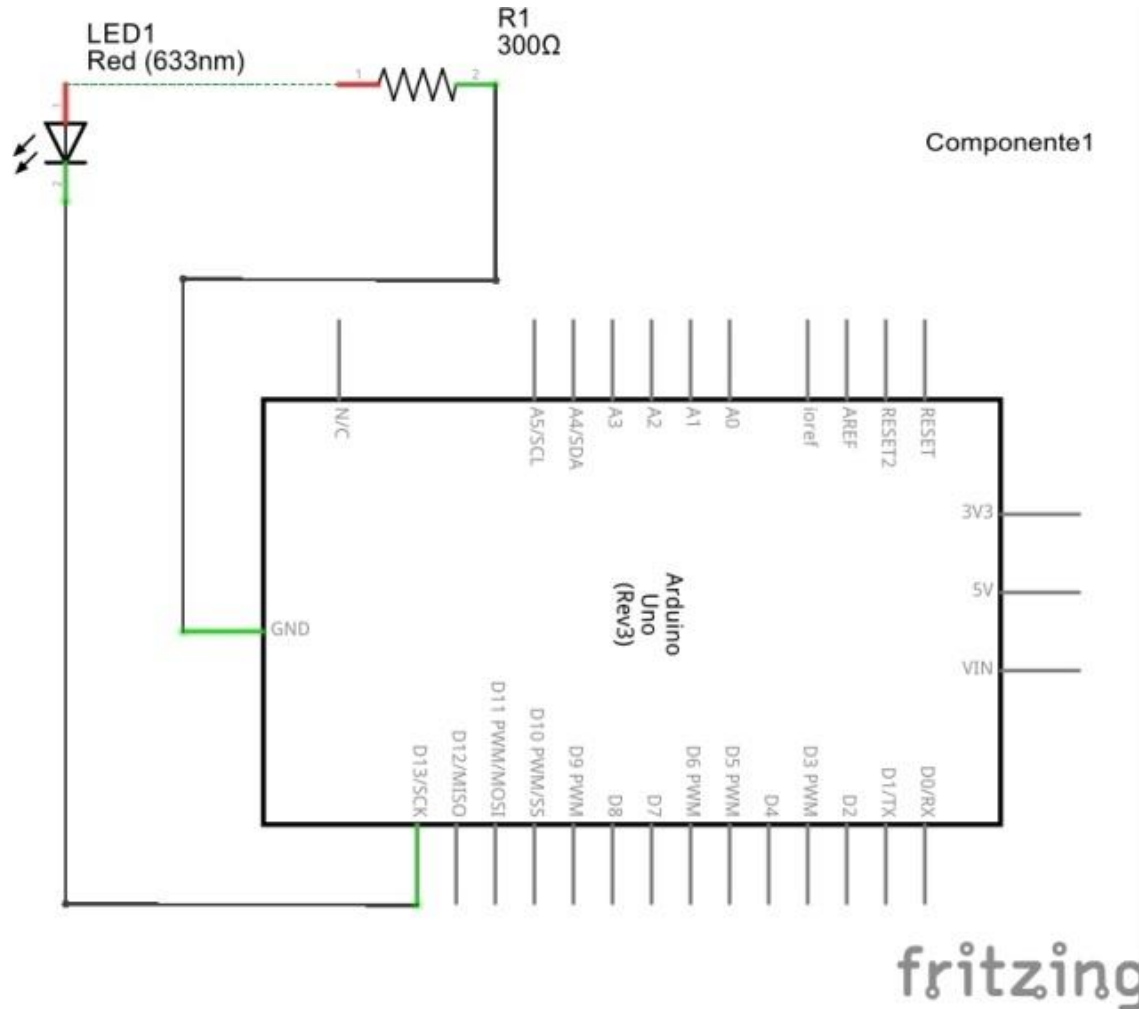
The screenshot shows the Arduino Serial Monitor window titled "COM4". It features a text input field at the top with an "Enviar" button. The main area displays the output of the sketch, which consists of the text "Alo Arduino!" repeated 13 times, followed by "Alo". At the bottom, there are three controls: a checked "Auto-rolagem" checkbox, a dropdown menu set to "Nenhum final-de-linha", and a dropdown menu set to "9600 velocidade".



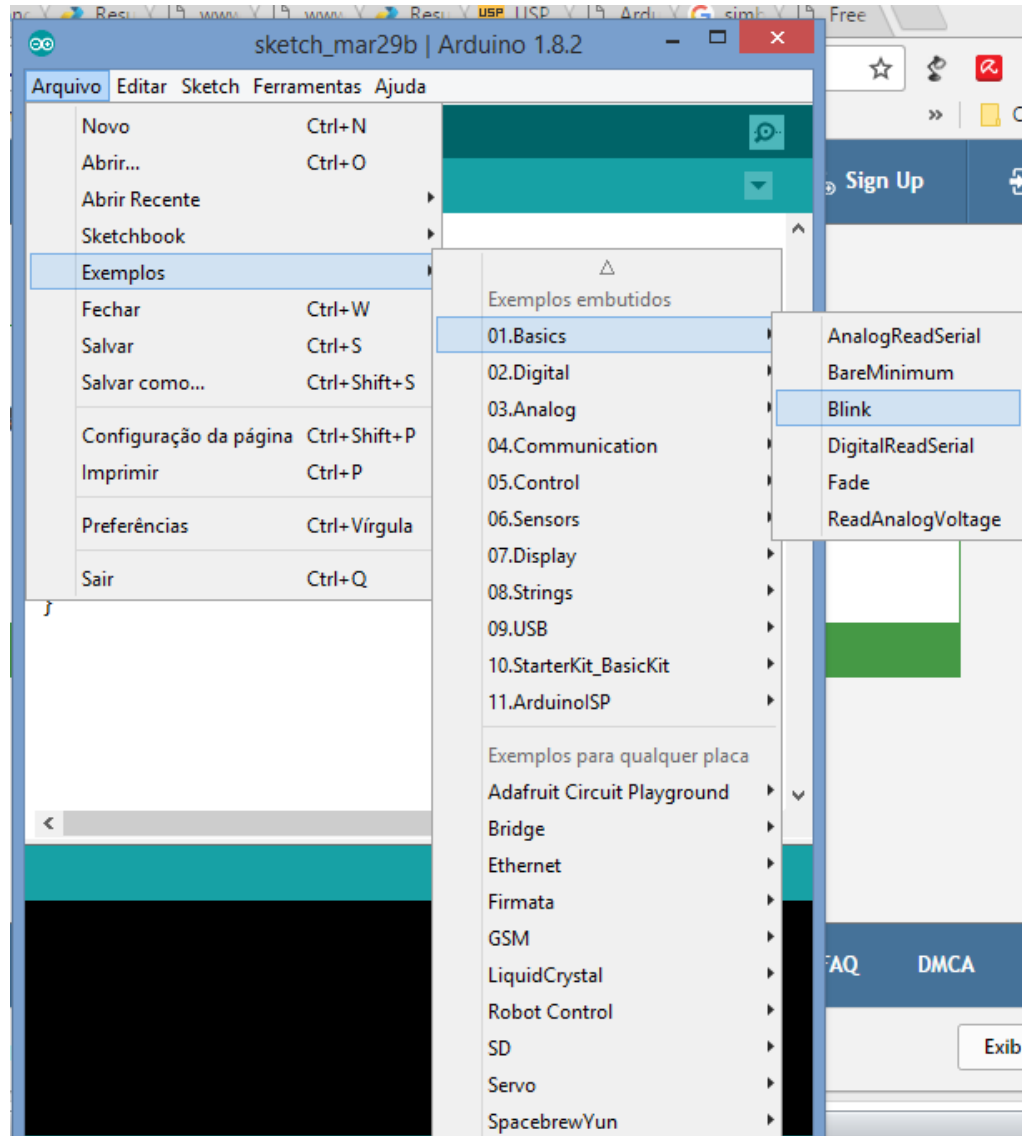
# Montagem do LED com resistor



# Circuito esquemático do LED com resistor



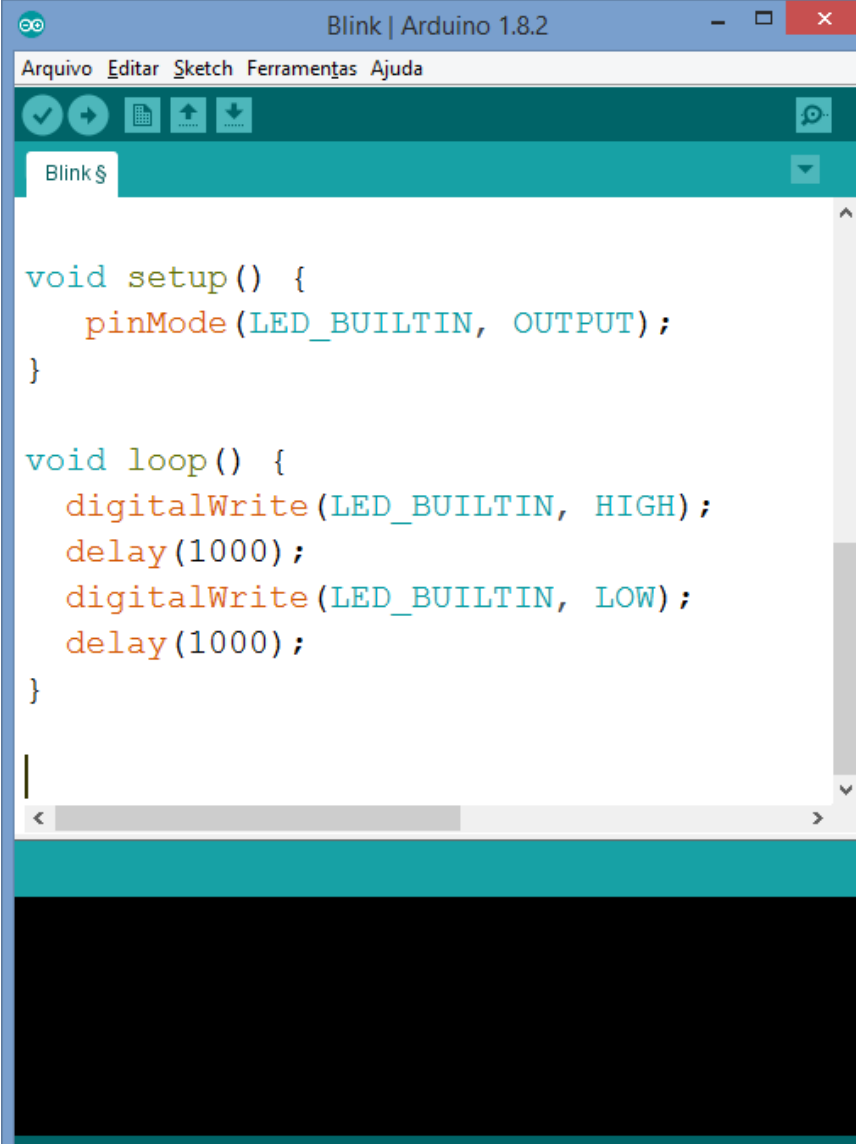
# Abrindo a janela de exemplos do IDE Wiring



# Sketch blink (exemplo do Arduino)

```
/*  
  Blink  
  Turns on an LED on for one second, then off for one second, repeatedly.  
  
  This example code is in the public domain.  
  
  modified 8 May 2014  
  by Scott Fitzgerald  
  
  modified 2 Sep 2016  
  by Arturo Guadalupi  
  
  modified 8 Sep 2016  
  by Colby Newman  
*/  
  
// the setup function runs once when you press reset or power the board  
void setup() {  
  // initialize digital pin LED_BUILTIN as an output.  
  pinMode(LED_BUILTIN, OUTPUT);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)  
  delay(1000); // wait for a second  
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW  
  delay(1000); // wait for a second  
}
```

# Executando o *sketch* Arduino

A screenshot of the Arduino IDE interface. The window title is "Blink | Arduino 1.8.2". The menu bar includes "Arquivo", "Editar", "Sketch", "Ferramentas", and "Ajuda". The toolbar contains icons for saving, running, uploading, and downloading. The current sketch is named "Blink\$". The code editor displays the following C++ code:

```
void setup() {  
    pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(1000);  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(1000);  
}
```

The bottom of the IDE shows a dark area, likely representing the serial monitor or a terminal window.

# Sketch blink modificado

```
/*  
  Blink  
  
  modified 8 May 2014  
  by Scott Fitzgerald  
  
  modified 2 Sep 2016  
  by Arturo Guadalupi  
  
  modified 8 Sep 2016  
  by Colby Newman  
*/  
  
void setup() {  
  pinMode(13, OUTPUT);  
}  

```

# Sketch original

```
void setup() {  
  pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(LED_BUILTIN, HIGH);  
  delay(1000);  
  digitalWrite(LED_BUILTIN, LOW);  
  delay(1000);  
}
```

# Modificado

```
void setup() {  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(13, HIGH);  
  delay(1000);  
  digitalWrite(13, LOW);  
  delay(1000);  
}
```

Carregado.

Opções de compilação alteradas, recompilar...  
O sketch usa 928 bytes (2%) de espaço de a...  
Variáveis globais usam 9 bytes (0%) de mer...

# Tipos de dados

O que são tipos de dados?

Tipos de dados são dados que possuem uma natureza pré-definida. Exemplos de tipos de dados são: números naturais, números inteiros, números reais, variáveis alfanuméricas ou textuais (palavras ou caracteres individuais).

Tipo	Uso	Exemplos
<b>int</b>	Número inteiro	1, 2, 100, 300, 1000
<b>float</b>	Número real (ponto flutuante) ou decimal	3.14, $10^{-9}$ , $6.023 \times 10^{23}$
<b>char</b>	Representa um único caractere	'A', 'B', 'a', 'b', 65, 66, 97, 98
<b>String</b>	Representa uma cadeia de caracteres	"Alo Arduino!"



# Tipos de dados numéricos

Tipos de dados numéricos são usados para manipular variáveis numéricas por meio de operações aritméticas.

Os principais tipos de variáveis numéricas são o tipo inteiro (`int`) e real de precisão simples (`float`) e dupla precisão (`double`).

Tipo	Tamanho (bits)	Algarismos significativos	Valor mínimo	Valor máximo
<code>int</code>	16	5	-32768	32767
<code>long</code>	32	10	-2.147.483.648	2.147.483.647
<code>float</code>	32	7	$-1.1755 \times 10^{-38}$	$3.4028 \times 10^{38}$
<code>double</code>	64	15	$-2.225 \times 10^{-308}$	$2.225 \times 10^{308}$

# Tipos de dados

- **boolean**: valor verdadeiro (true) ou falso (false);
- **char**: um caractere;
- **byte**: um byte, ou sequência de 8 bits;
- **int**: número inteiro de 16 bits com sinal (-32768 a 32767);
- **unsigned int**: número inteiro de 16 bits sem sinal (0 a 65535);
- **long**: número inteiro de 16 bits com sinal (-2147483648 a 2147483647);
- **unsigned long**: número inteiro de 16 bits sem sinal (0 a 4294967295);
- **float**: número real de precisão simples (ponto flutuante);
- **double**: número real de precisão dupla (ponto flutuante);
- **string**: sequência de caracteres;
- **void**: tipo vazio (não tem tipo). <sup>2</sup>

# Constantes e Variáveis

## O que são constantes?

Constantes são tipos de dados pré-definidos fixos que não podem ser alterados. Exemplo de constantes são os tipos `HIGH` e `LOW` no *sketch* blink.

Constantes são valores predefinidos da linguagem e não podem ser alteradas. Existem três tipos na linguagem C do Arduino, são elas:

- **HIGH | LOW**: definem as tensões nos pinos, sendo 0V baixo (low) e 5V alto (high);
- **INPUT | OUTPUT**: usadas em algumas funções para determinar se é entrada ou saída;
- **true | false**: constantes booleanas, onde false é sempre zero e qualquer outro valor é true.

# Constantes e Variáveis

## O que são variáveis?

Em contraposição às constantes, as variáveis são usadas para armazenar informação temporariamente. O valor de uma variável pode mudar, por exemplo, como resultado de uma operação aritmética:

```
int A = 5;
```

```
int B = 10;
```

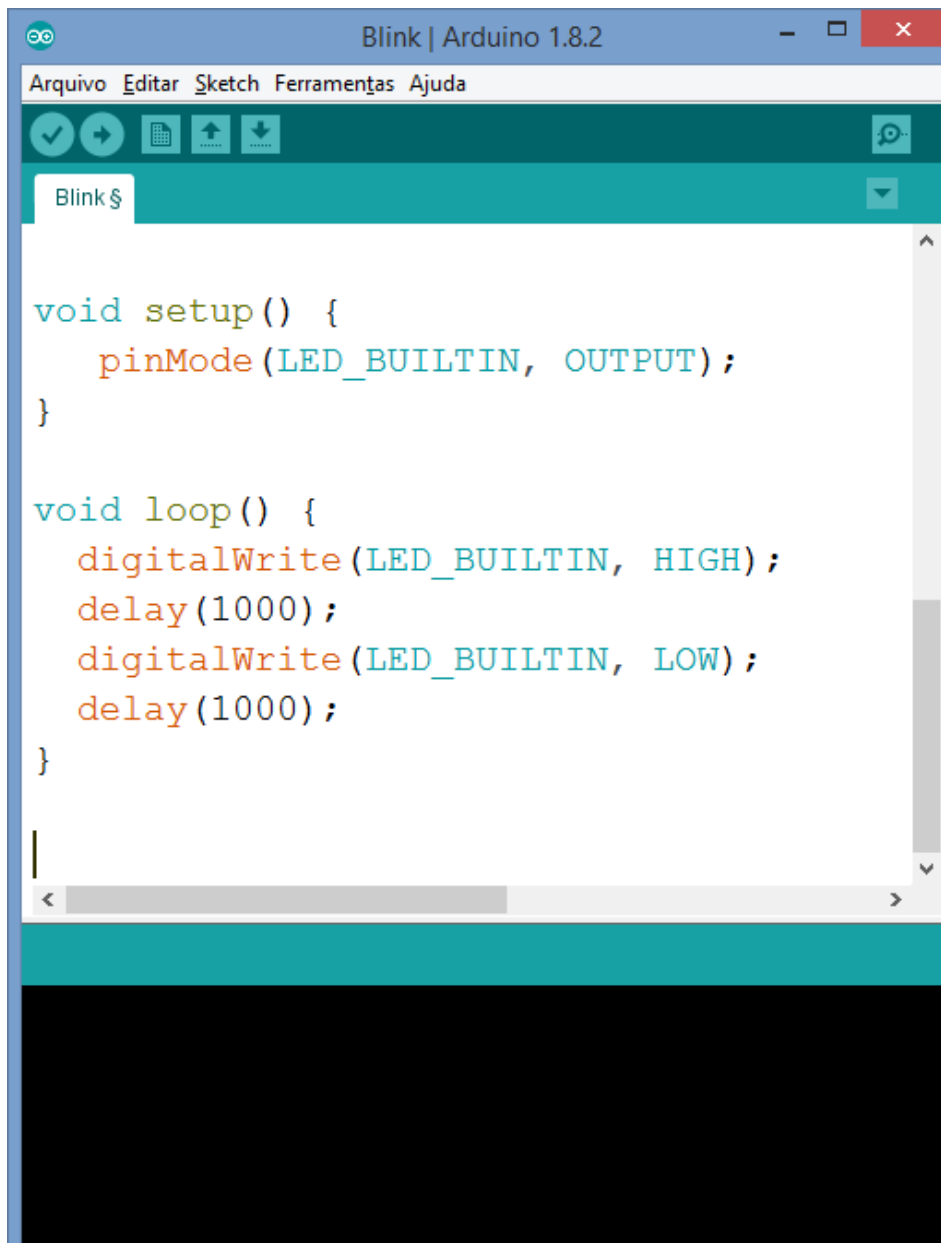
```
A = A + B;
```

# Sketch blink com variável LED

```
/*  
  Blink  
  
  modified 8 May 2014  
  by Scott Fitzgerald  
  
  modified 2 Sep 2016  
  by Arturo Guadalupi  
  
  modified 8 Sep 2016  
  by Colby Newman  
*/  
  
int LED = 13;  
  
void setup() {  

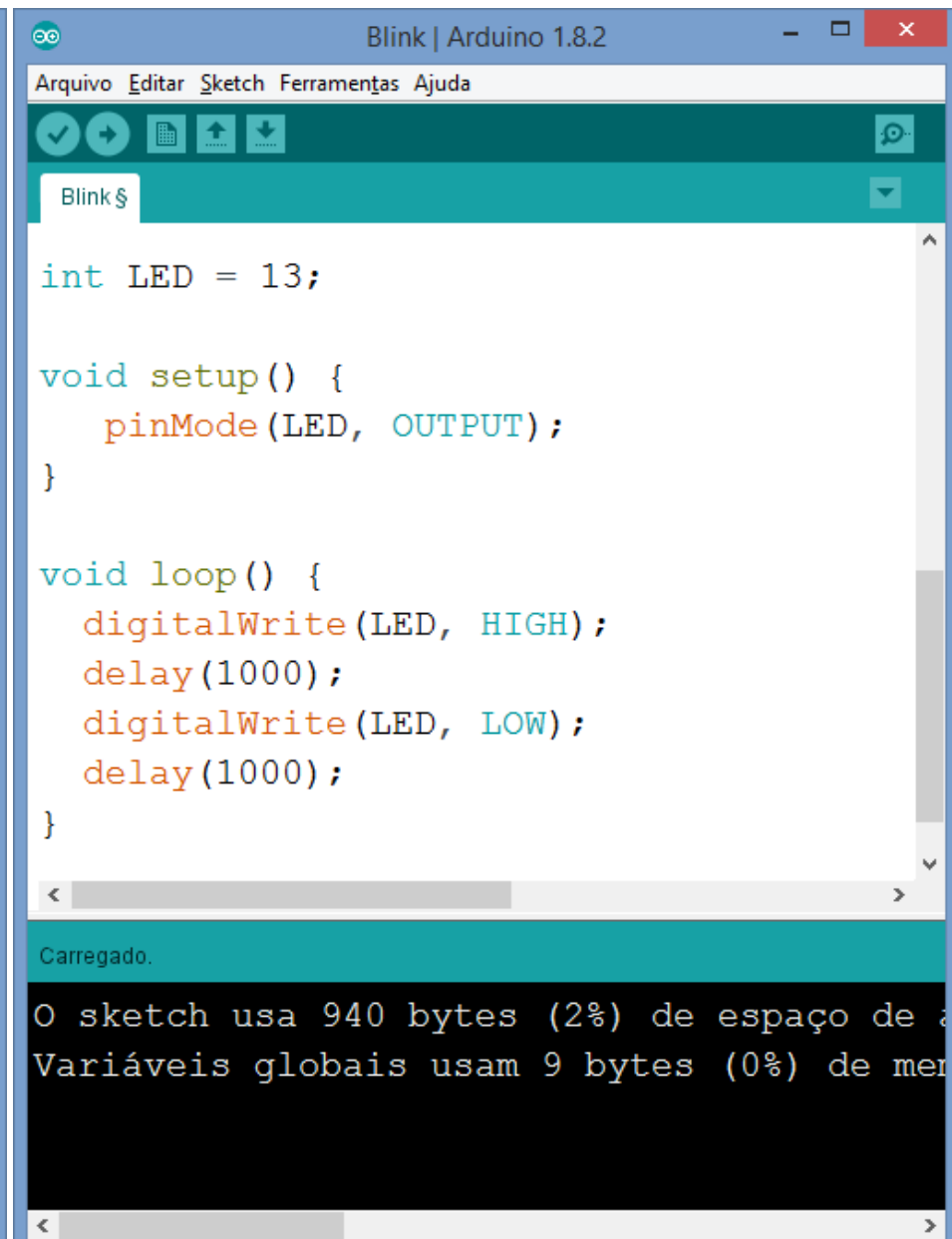
```

# Sketch original



```
void setup() {  
  pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(LED_BUILTIN, HIGH);  
  delay(1000);  
  digitalWrite(LED_BUILTIN, LOW);  
  delay(1000);  
}
```

# Modificado



```
int LED = 13;  
  
void setup() {  
  pinMode(LED, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(LED, HIGH);  
  delay(1000);  
  digitalWrite(LED, LOW);  
  delay(1000);  
}
```

Carregado.

O sketch usa 940 bytes (2%) de espaço de a  
Variáveis globais usam 9 bytes (0%) de mer

# Prática de programação do Arduino

Montar um circuito de quatro LEDs com acendimento sequencial, usando as portas digitais 8, 9, 10 e 11

