

Robótica Educacional

Módulo 3



Aula 07

Recursos das bibliotecas

GOVERNADOR DO ESTADO DO PARANÁ

Carlos Massa Ratinho Júnior

SECRETÁRIO DE ESTADO DA EDUCAÇÃO

Roni Miranda Vieira

DIRETOR DE TECNOLOGIA E INOVAÇÃO

Claudio Aparecido de Oliveira

COORDENADOR DE TECNOLOGIAS EDUCACIONAIS

Marcelo Gasparin

Produção de Conteúdo

Cleiton Rosa

Darice Alessandra Deckmann Zanardini

Validação de Conteúdo

Cleiton Rosa

Revisão Textual

Kellen Pricila dos Santos Cochinski

Projeto Gráfico, Diagramação e Geração de Imagem I.A.

Edna do Rocio Becker

2024

Sumário

Introdução	2
Objetivos desta aula	3
Roteiro da aula	3
1. Contextualização	3
2. Programação	4
Feedback e finalização	18
Referências	21

Introdução

No decorrer da nossa jornada pela Robótica nos deparamos com diferentes projetos, dos mais simples aos mais sofisticados.

Em muitos desses projetos, a programação do Arduino ficou facilitada devido à utilização de **bibliotecas**, como se estivéssemos utilizando uma caixa de ferramentas com tudo (ou quase tudo) que precisamos para nosso projeto.

Na programação dos robôs, utilizamos as ferramentas para controle dos motores, por exemplo, sem precisar escrever muitas linhas de código para ações como ir para frente, para trás ou virar; na programação de sensores, pudemos ter acesso aos seus dados também com apenas algumas funções; nos projetos de comunicação e IoT, o mesmo!



Objetivos desta aula

- Conhecer funções presentes em bibliotecas pelo recurso de abertura dos arquivos *.h;
- Conhecer o caminho para localização comum de bibliotecas instaladas no computador: **Documentos > Arduino > libraries**.

Lista de materiais

- Computador e/ou notebook;
- Arduino.

Roteiro da aula

1. Contextualização

Na programação com Arduino IDE, as bibliotecas constituem parte fundamental no desenvolvimento de protótipos por fornecerem funcionalidades específicas que podem ser facilmente incorporadas na programação dos componentes relacionados, organizando o código de programação e facilitando a leitura de comandos.

Como já vimos na **Aula 02 – Arduino: Bibliotecas e Funções**, do Módulo 2 de Robótica Educacional, no Arduino IDE tem-se três modelos de bibliotecas: biblioteca **essencial** ou **core**, fundamental para o desenvolvimento de projetos por possuir funções usuais como **digitalWrite()** e **analogRead()**; biblioteca **nativa** do Arduino IDE, a qual não requer instalação posterior, mas precisa ser declarada na programação pela diretiva **#include**, como a **Servo.h**, utilizada para controle de servomotores; e bibliotecas de **terceiros** (externa), como a **Ultrasonic.h**, desenvolvida por Erick Simões para o controle do sensor ultrassônico, que possuem o objetivo de otimizar o código da programação para os componentes correspondentes por fornecer funcionalidades não presentes em outras bibliotecas do software.

Para utilizar uma biblioteca externa em um sketch do Arduino, é preciso, caso você utilize a versão software do Arduino

IDE, instalar a biblioteca pelo gerenciador de bibliotecas da IDE, que pode ser acessado pelo menu **Sketch > Include Library > Manage Libraries**. Caso você utilize a versão online do Arduino IDE, a biblioteca constará em seus repositórios e, para facilitar sua inclusão e visualização de mais informações e exemplos de utilização, poderá ser **favoritada**. Por fim, em ambas as versões também é preciso incluir a biblioteca no sketch da programação com a diretiva **#include**, informando ao Arduino IDE que o código da biblioteca deve ser incorporado na compilação do projeto.



2. Programação

Como neste Módulo 3 estamos incentivando você e seus colegas a se aventurarem cada vez mais pelo mundo da Robótica com o desenvolvimento de projetos e a prototipagem de ideias, este percurso envolve também ampliar o domínio sobre a programação do Arduino. Por isso, vamos explorar os recursos das bibliotecas!

Além das orientações geralmente presentes em nossas aulas quanto à indicação e utilização de bibliotecas para projetos específicos com o Arduino, você pode explorar mais os recursos das bibliotecas para descobrir outras funções e utilidades.

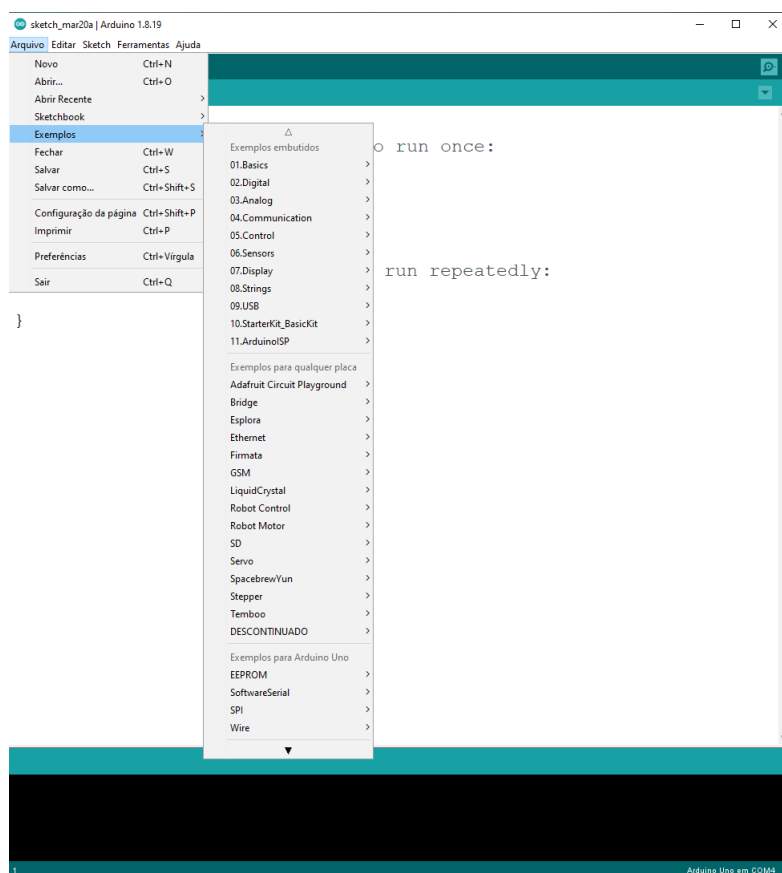
Muitos desenvolvedores compartilham exemplos de programações com suas bibliotecas que podem te inspirar a novos projetos e, com isso, você vai descobrindo outras funcionalidades. Vamos experimentar? Se você estiver usando a versão software do Arduino IDE, pelo Windows, abra um novo

sketch e clique no menu **Arquivo > Exemplos**. Você pode localizar os exemplos também pela pasta da biblioteca salva localmente, geralmente pelo caminho **Documentos > Arduino > libraries > nome_biblioteca > examples**.

Na aba que se abrir, localizando os exemplos pelo Arduino IDE, estarão listadas todas as bibliotecas que

você instalou e outras que o Arduino possui embutida ou sugere e, para cada uma delas, há um conjunto de programações que exploram seus recursos. Muitas vezes, as funções de bibliotecas que trazemos em nossos projetos são apenas uma parte de toda a potencialidade de funções que uma biblioteca possui.

Figura 01- Exemplos Arduino IDE Software versão 1.8.19



Fonte: Arduino IDE

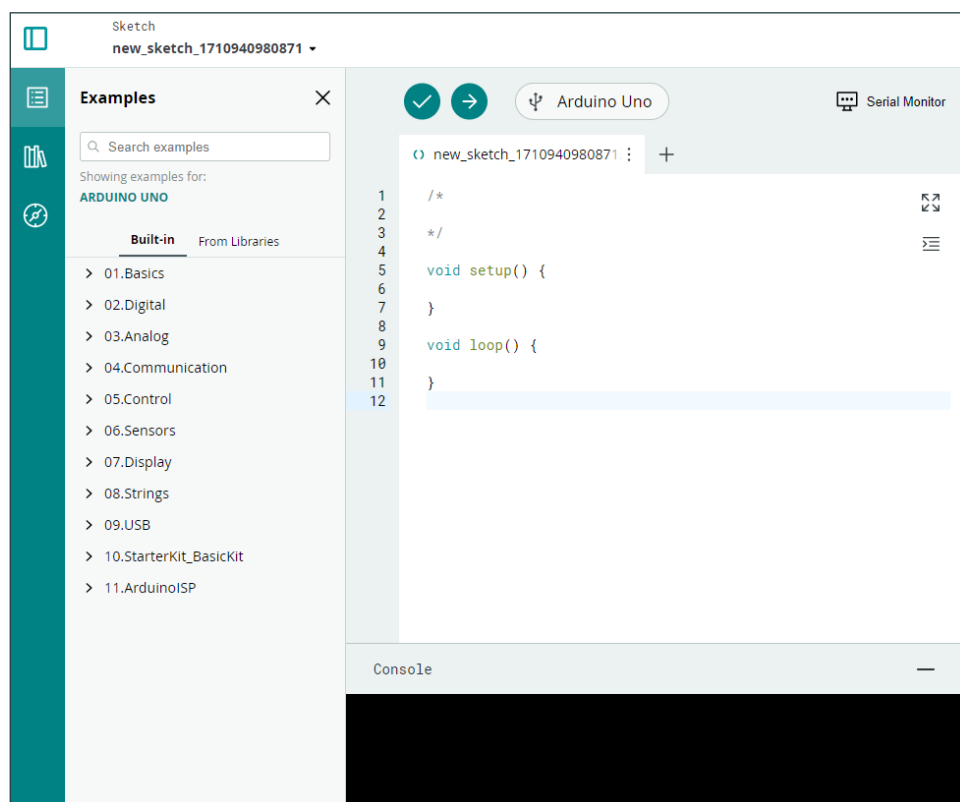
Robótica

Aula 07 Recursos das bibliotecas

Os exemplos estarão divididos em conjuntos:

- **Exemplos embutidos:** referem-se aos exemplos presentes na IDE e destinados a placas variadas.
- **Exemplos para qualquer placa:** apresenta exemplos para diferentes modelos, sem necessidade de alterações no código.
- **Exemplos para Arduino Uno:** traz exemplos das bibliotecas nativas do Arduino para este modelo.
- **Exemplos de Bibliotecas Personalizadas:** refere-se aos exemplos das bibliotecas que você possui instaladas em seu computador.

Figura 02- Exemplos Arduino IDE Online



No Arduino IDE Online, você encontra os exemplos de todas as bibliotecas acessando, no menu lateral, o ícone Examples após indicar o tipo de placa microcontroladora conectada.

Fonte: Arduino IDE Online

Robótica

Aula 07 Recursos das bibliotecas

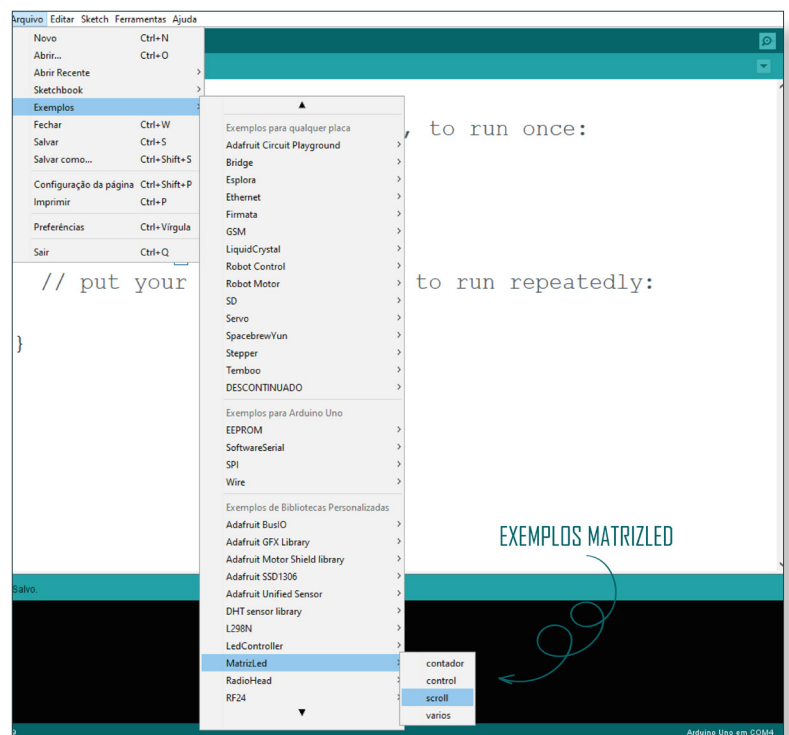
A diferença é que, nesta versão, os exemplos estão divididos em dois conjuntos:

- **Built-in:** Apresenta exemplos integrados de códigos com foco no desenvolvimento da programação do Arduino e montagem dos protótipos. Na página Built-in Examples, você encontra uma coleção de tutoriais com explicações (em inglês) e circuito destes exemplos.
- **From Libraries:** Traz exemplos de programações de todas as bibliotecas, incluindo as de terceiros, disponíveis nos repositórios do Arduino.

Comece a explorar os recursos das bibliotecas selecionando um dos exemplos de programação que desejar. Para facilitar nossa análise inicial, vamos optar pelos exemplos da biblioteca **MatrizLED**, desenvolvida por Daniel Alvarez e disponível em seu [GitHub](#), a qual utilizamos na **Aula 06 - Matriz de LEDs** do Módulo 2 de Robótica Educacional. Além das funções de escrever com efeito rolagem e exibir contagem, utilizadas no projeto da **Aula 06**, podemos explorar outras funções que o desenvolvedor preparou e ampliar nossos projetos com este componente. Vamos lá?

Ao buscar os exemplos da biblioteca **MatrizLED** pelo software Arduino IDE ou sua versão online, o sketch nos apresenta quatro programações: "contador", "control", "scroll" e "varios".

Figura 03- Exemplos da biblioteca MatrizLed no software Arduino IDE



Fonte: Arduino IDE



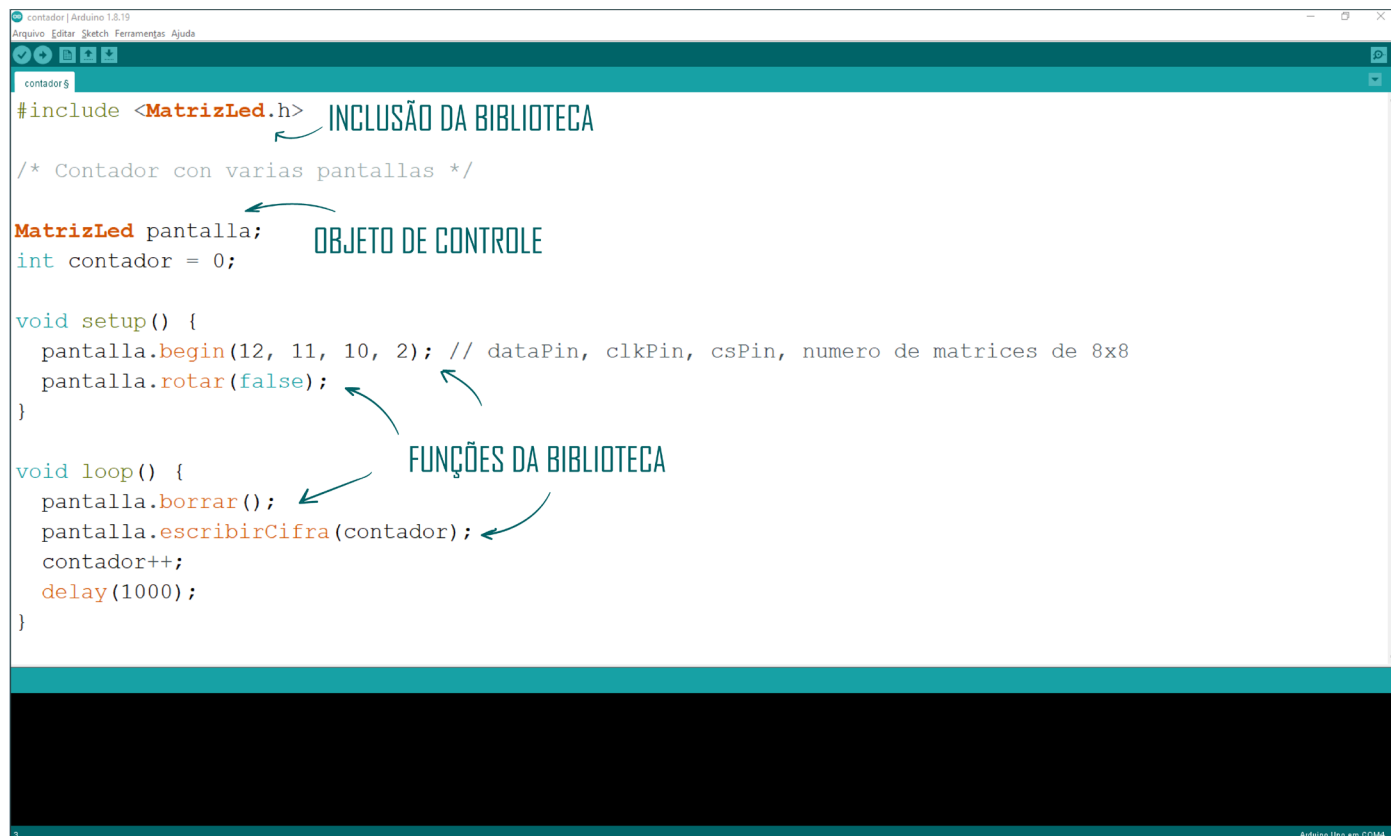
Robótica

Aula 07 Recursos das bibliotecas

Cada um desses exemplos traz a programação de um projeto com a matriz de LEDs. Exemplos de outras bibliotecas trarão modelos de programação para os componentes destinados à sua criação.

Você pode abrir cada um dos exemplos para identificar as funções utilizadas e comparar o código, verificando padrões e diferenças. Esse trabalho de explorar códigos é bem investigativo e exige atenção, estimulando nossa curiosidade, pois envolve a análise das funções descritas, comandos associados, parâmetros e relação com demais sintaxes. Uma dica é: **observe a criação de objetos de controle no início da programação e localize, no decorrer do sketch, o nome atribuído ao dispositivo como início de funções.** Em cada um dos exemplos podemos encontrar novas funções ou novas finalidades.

Figura 04 – Exemplo da programação “contador”



```
contador | Arduino 1.8.19
Arquivo Editar Sketch Ferramentas Ajuda

contador$
#include <MatrizLed.h>  INCLUSÃO DA BIBLIOTECA

/* Contador con varias pantallas */

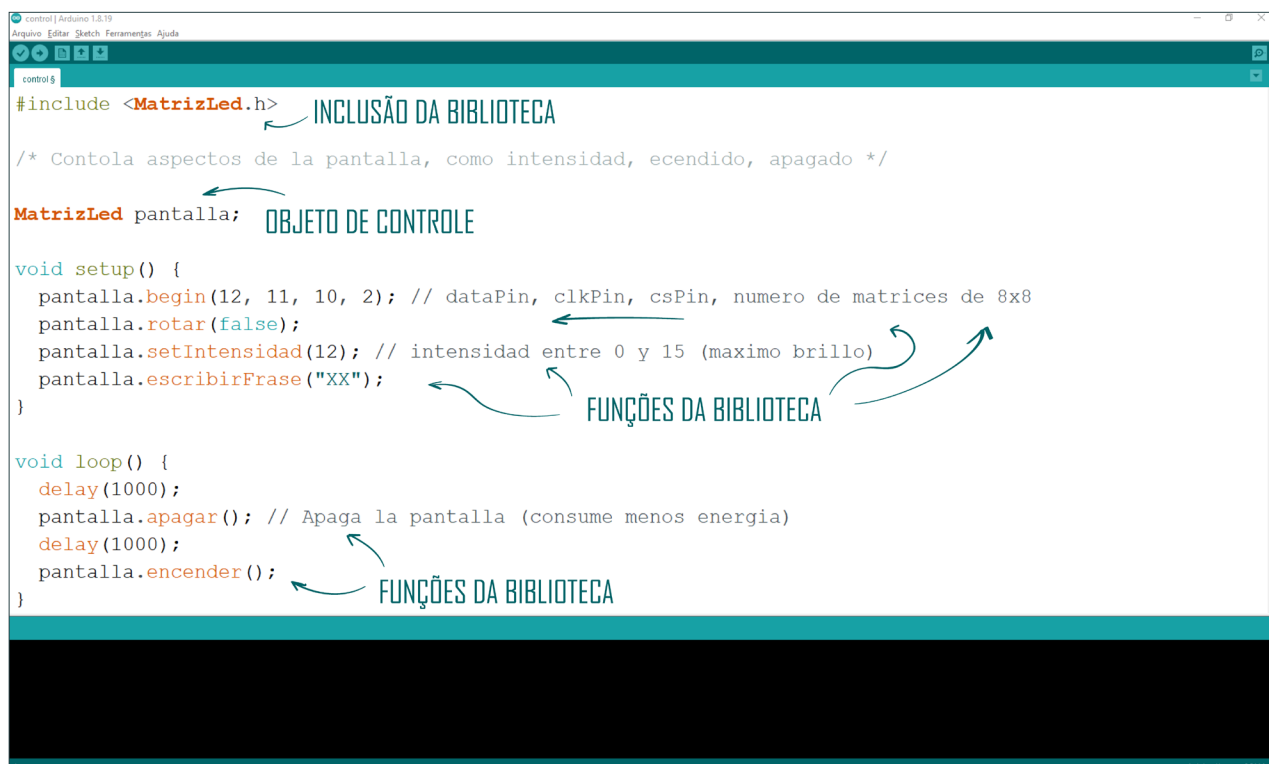
MatrizLed pantalla;     OBJETO DE CONTROLE
int contador = 0;

void setup() {
  pantalla.begin(12, 11, 10, 2); // dataPin, clkPin, csPin, numero de matrices de 8x8
  pantalla.rotar(false);
}

void loop() {
  pantalla.borrar();
  pantalla.escribirCifra(contador);
  contador++;
  delay(1000);
}
```

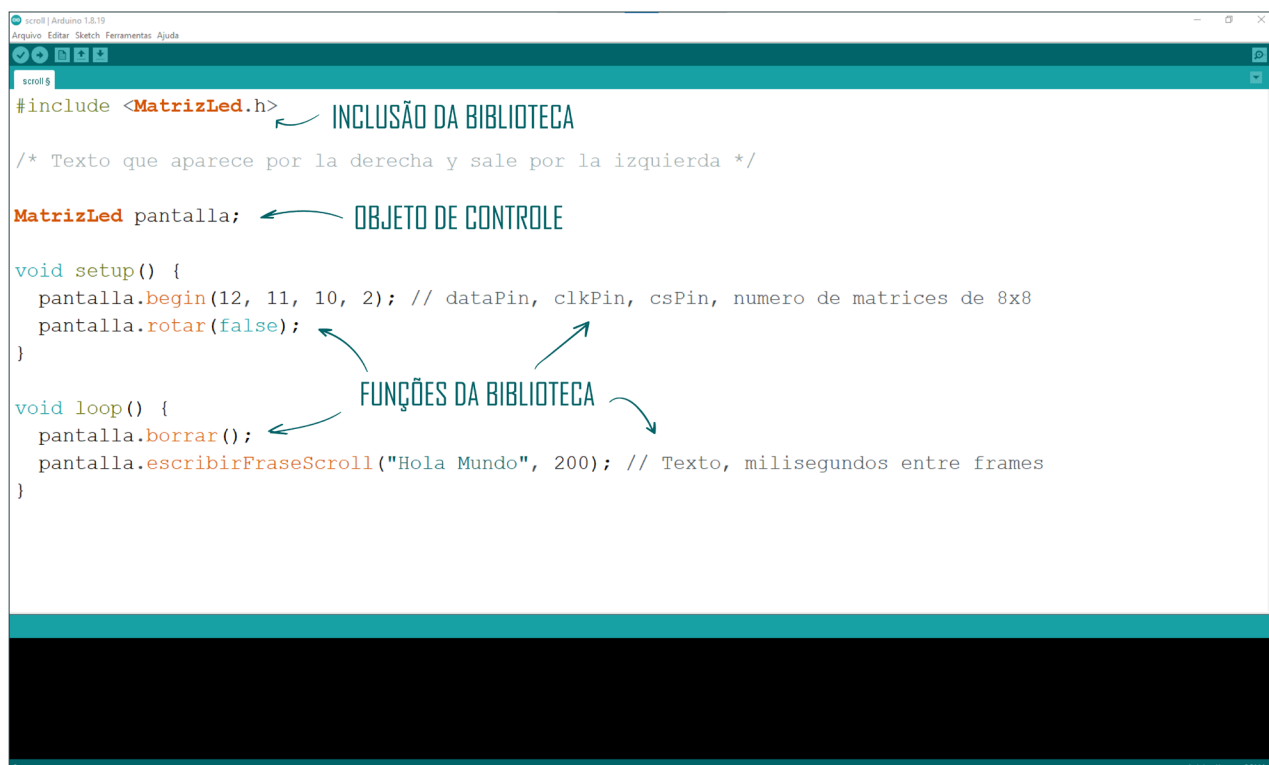
Fonte: Arduino IDE

Figura 05 – Exemplo da programação “control”



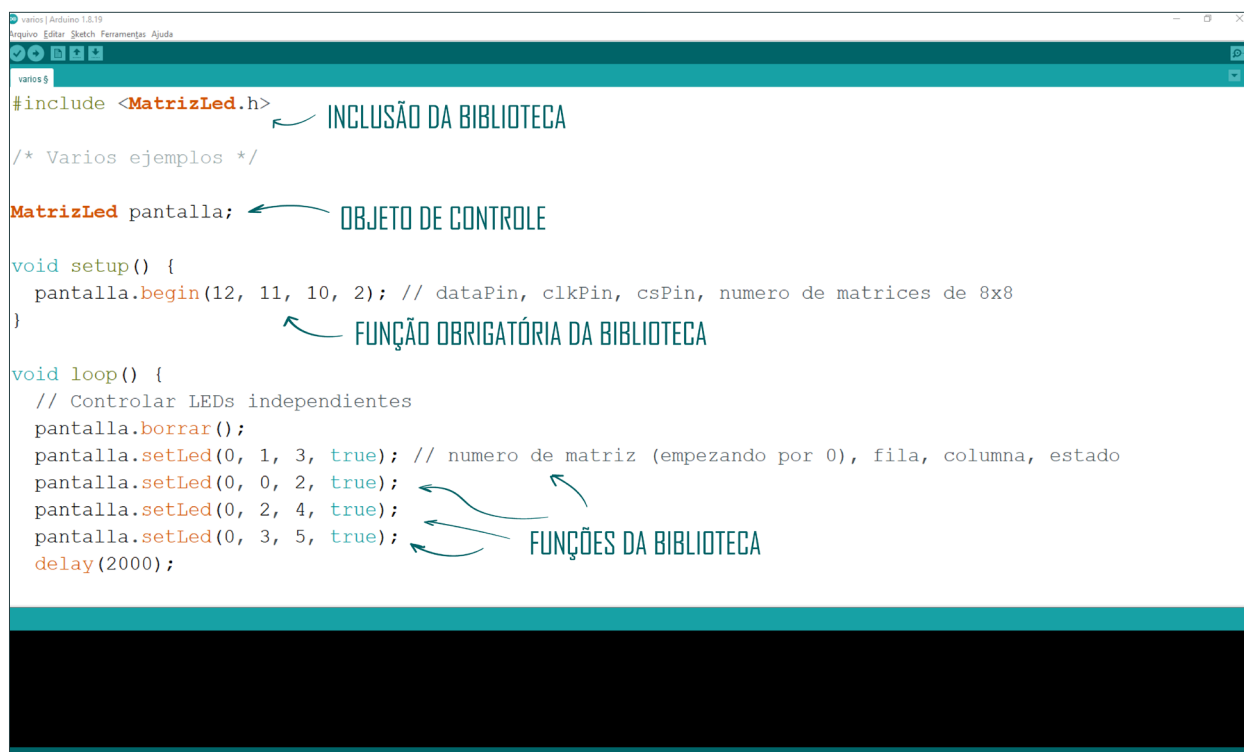
Fonte: Arduino IDE

Figura 06 – Exemplo da programação “scroll”



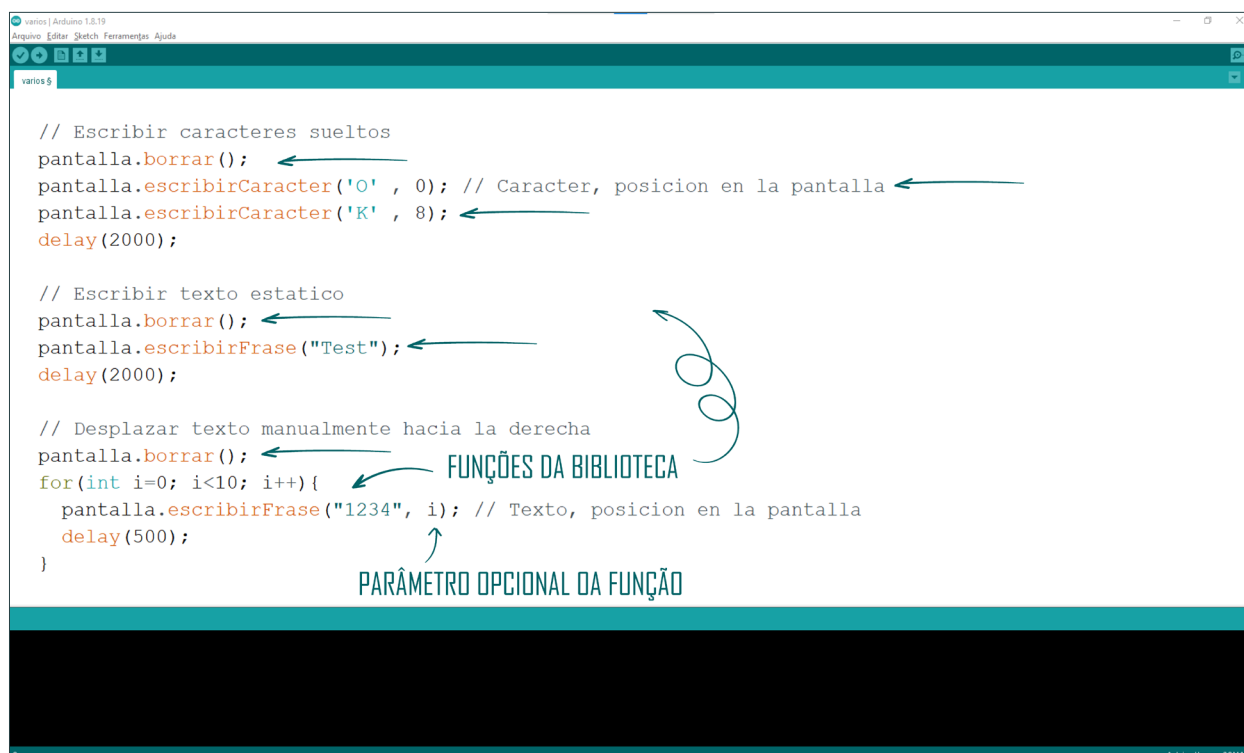
Fonte: Arduino IDE

Figura 07 – Exemplo da programação “varios”, parte 1



Fonte: Arduino IDE

Figura 08 – Exemplo da programação “varios”, parte 2

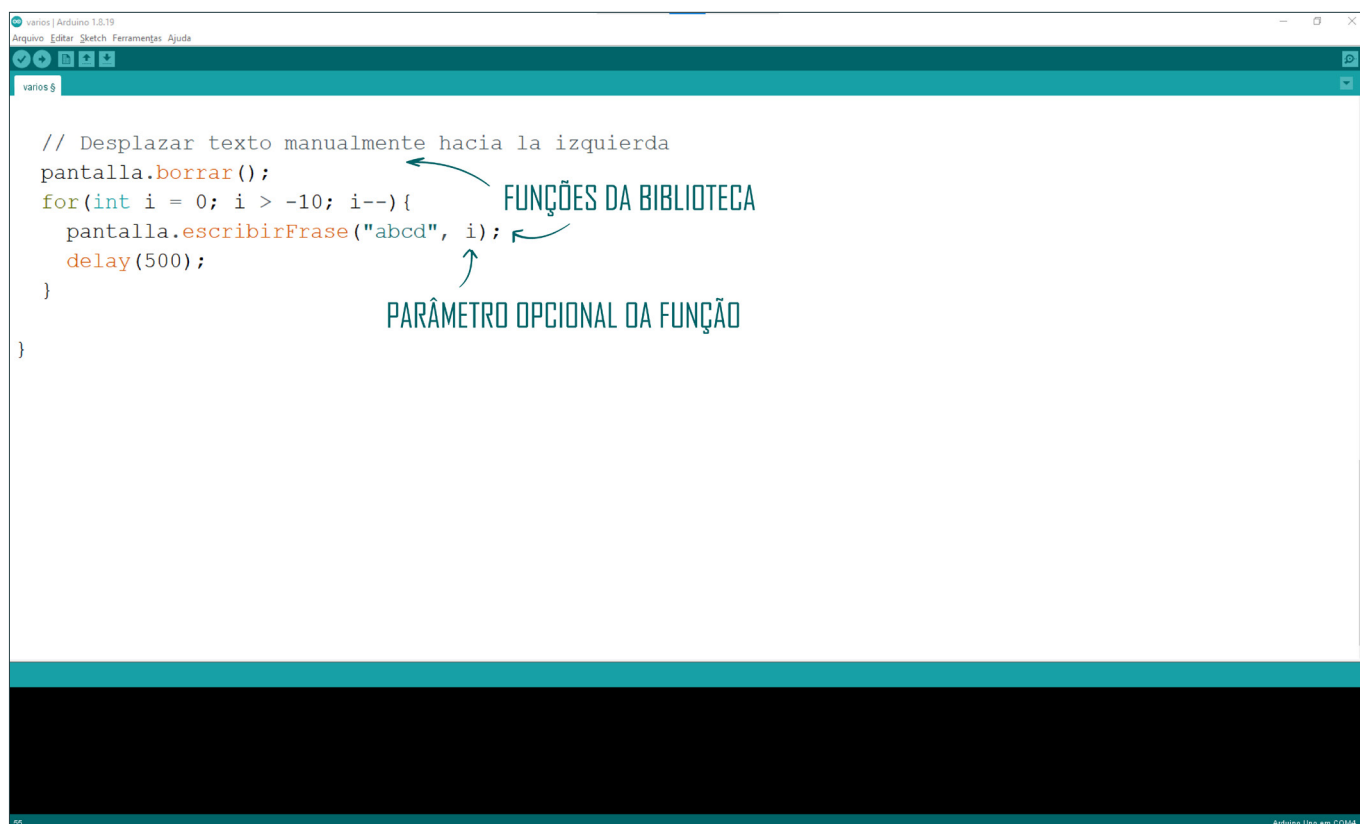


Fonte: Arduino IDE

Robótica

Aula 07 Recursos das bibliotecas

Figura 09 – Exemplo da programação “varios”, parte 3



Fonte: Arduino IDE

Cada uma das setas indicativas, considerando o objeto de controle “pantalla” como referência, indica as funções que o desenvolvedor preparou para a biblioteca e cada exemplo pode contar com funções distintas para seu objeto. Além disso, comparando as funções entre os exemplos de programação, conseguimos perceber três elementos obrigatórios: **1)** inclusão da biblioteca; **2)** criação do objeto de controle; **3)** inicialização da matriz de LEDs pelo comando `.begin()` e seus parâmetros.

Quanto aos parâmetros das demais funções, o desenvolvedor nos mostra outras funções com os comentários acerca de parâme-

tros aplicados e na comparação dos códigos de programação é possível perceber também que algumas funções podem conter outros parâmetros, como no exemplo “**varios**”, no qual o comando `.escribirFrase()` possui o parâmetro opcional da posição da frase na matriz, permitindo programarmos um efeito de rolagem.

Na análise dos exemplos da biblioteca **MatrizLed**, e em outros também, podemos questionar: **Será que a biblioteca apresenta mais recursos?** E quanto às bibliotecas sem exemplos das aplicações de suas funções, **como conhecer suas potencialidades e funções disponíveis?**

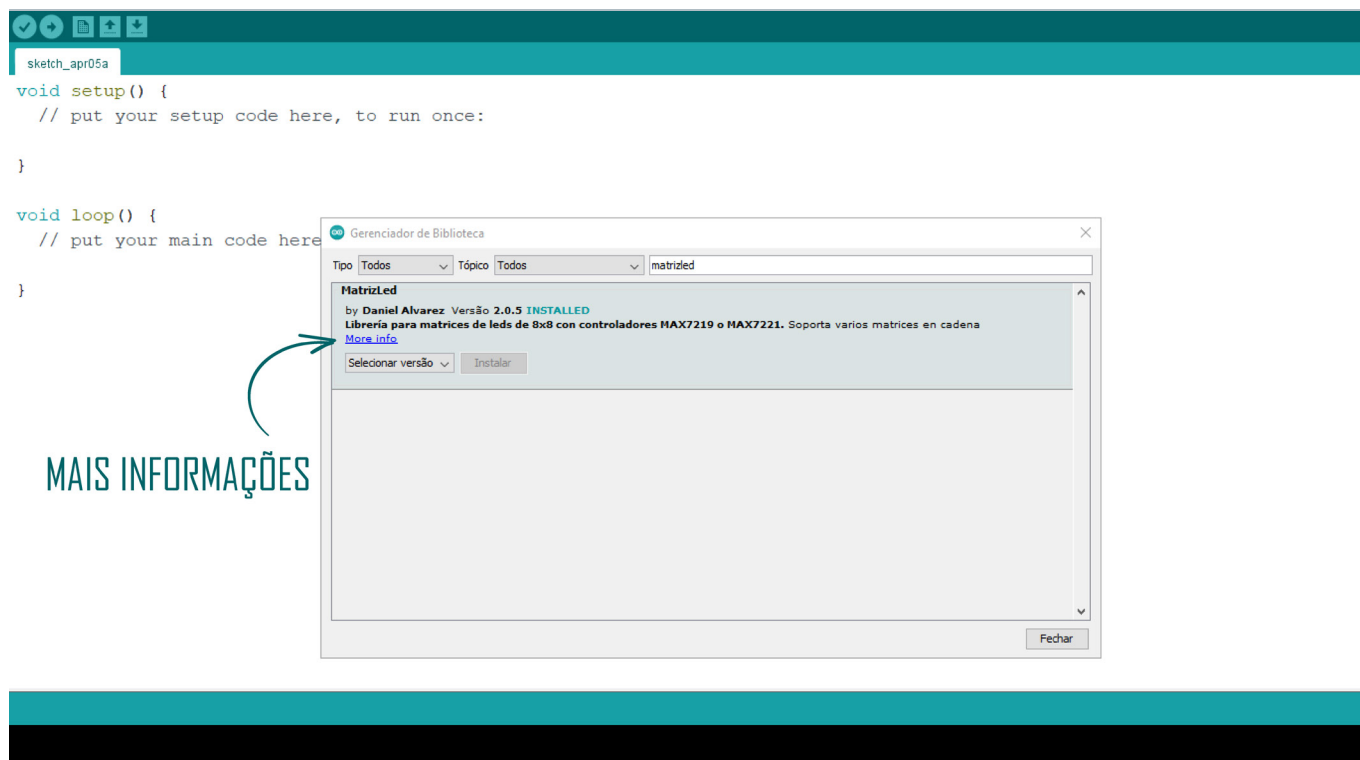
Robótica

Aula 07 Recursos das bibliotecas

Diante destes questionamentos, uma solução é explorar a pasta da biblioteca pelo **GitHub** do desenvolvedor ou pelo **diretório do Arduino IDE** em seu computador.

Geralmente, as programações dos projetos das nossas aulas são compartilhadas com um **comentário inicial** com informações sobre o projeto proposto e, se necessário, a biblioteca utilizada, com link ao **GitHub** do desenvolvedor. No Arduino IDE versões software e online, você localiza este link pela opção **More info** ("mais informações") ao buscar a biblioteca.

Figura 10 – Mais informações de biblioteca pelo software Arduino IDE 1.8.19



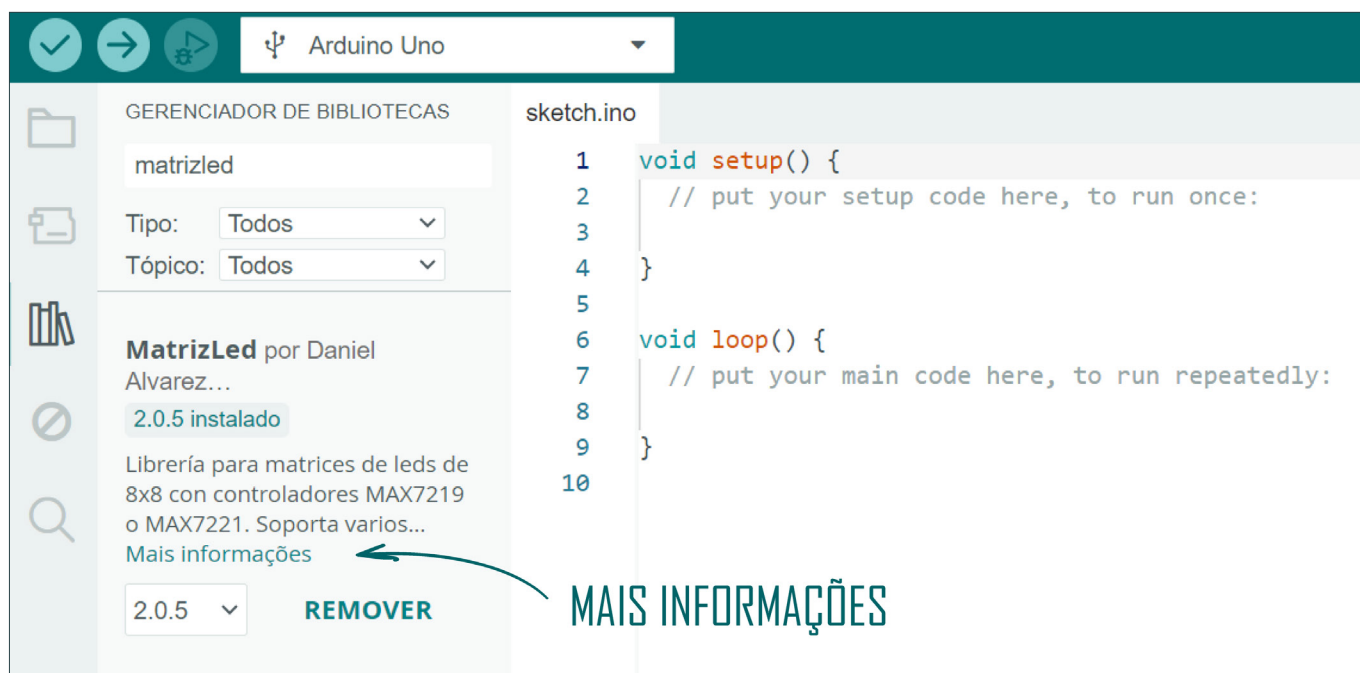
Fonte: Arduino IDE



Robótica

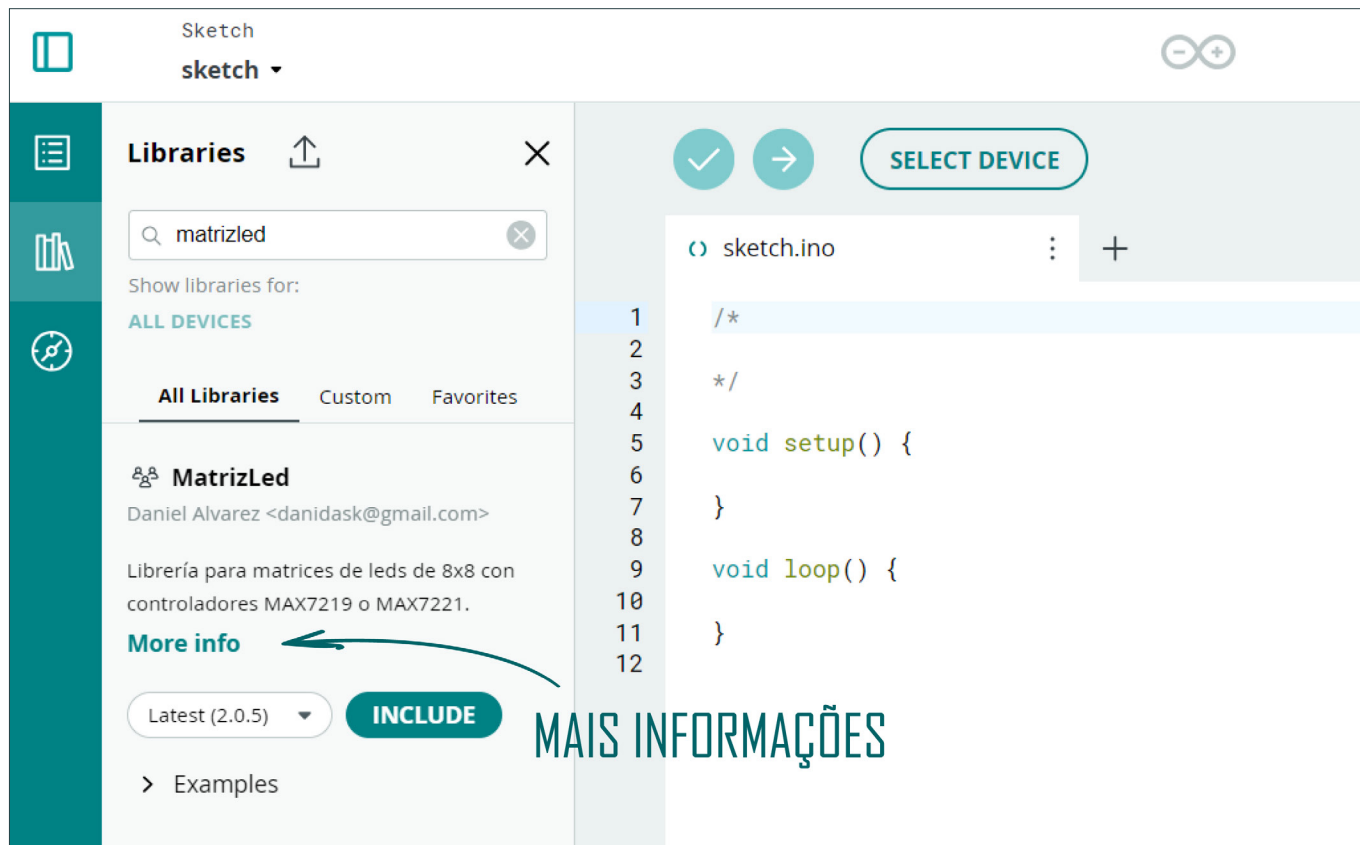
Aula 07 Recursos das bibliotecas

Figura 11 – Mais informações de biblioteca pelo software Arduino IDE 2



Fonte: Arduino IDE

Figura 12 – Mais informações de biblioteca pelo Arduino IDE Online



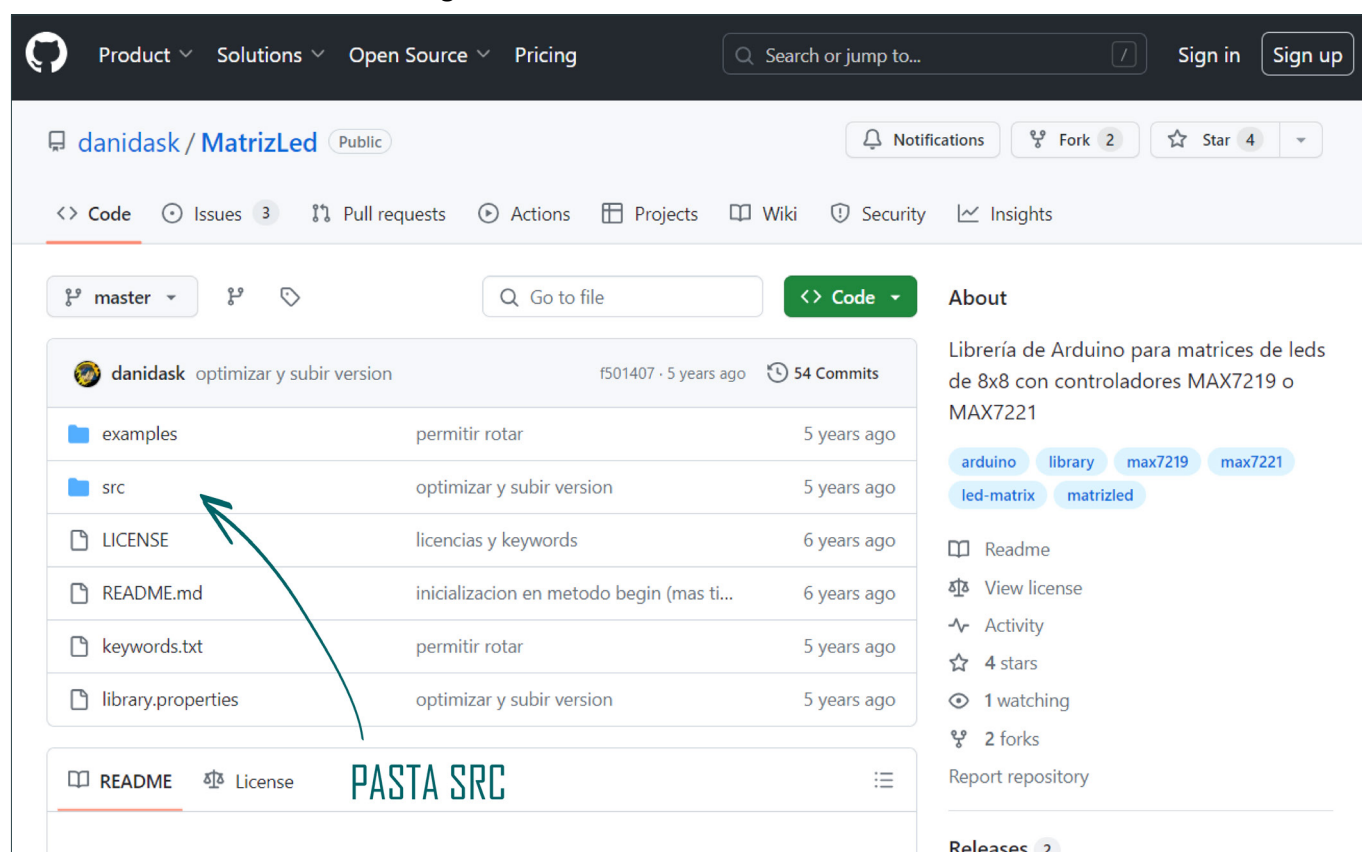
Fonte: Arduino IDE Online

Robótica

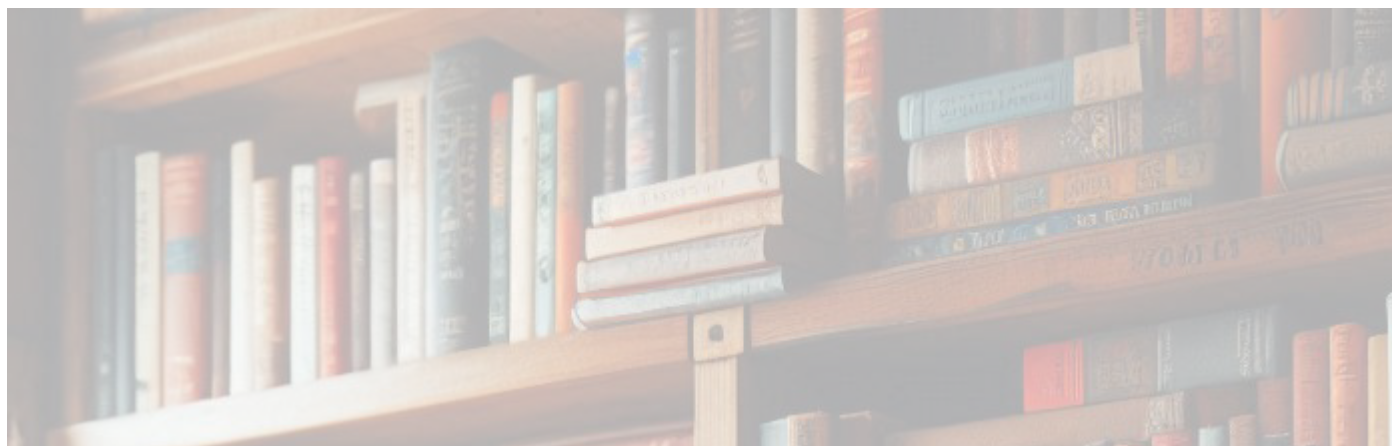
Aula 07 Recursos das bibliotecas

Pelo **GitHub**, localize o arquivo com nome o da biblioteca seguido pela extensão **.h** e abra-o no próprio navegador. Em projetos organizados, esse arquivo tende a ser localizado na pasta **"src"**. Caso o projeto compartilhado pelo desenvolvedor não possua esta pasta, localize o arquivo **.h** no diretório compartilhado – ele estará acompanhado pelo arquivo **.cpp**.

Figura 13 – GitHub da biblioteca MatrizLed



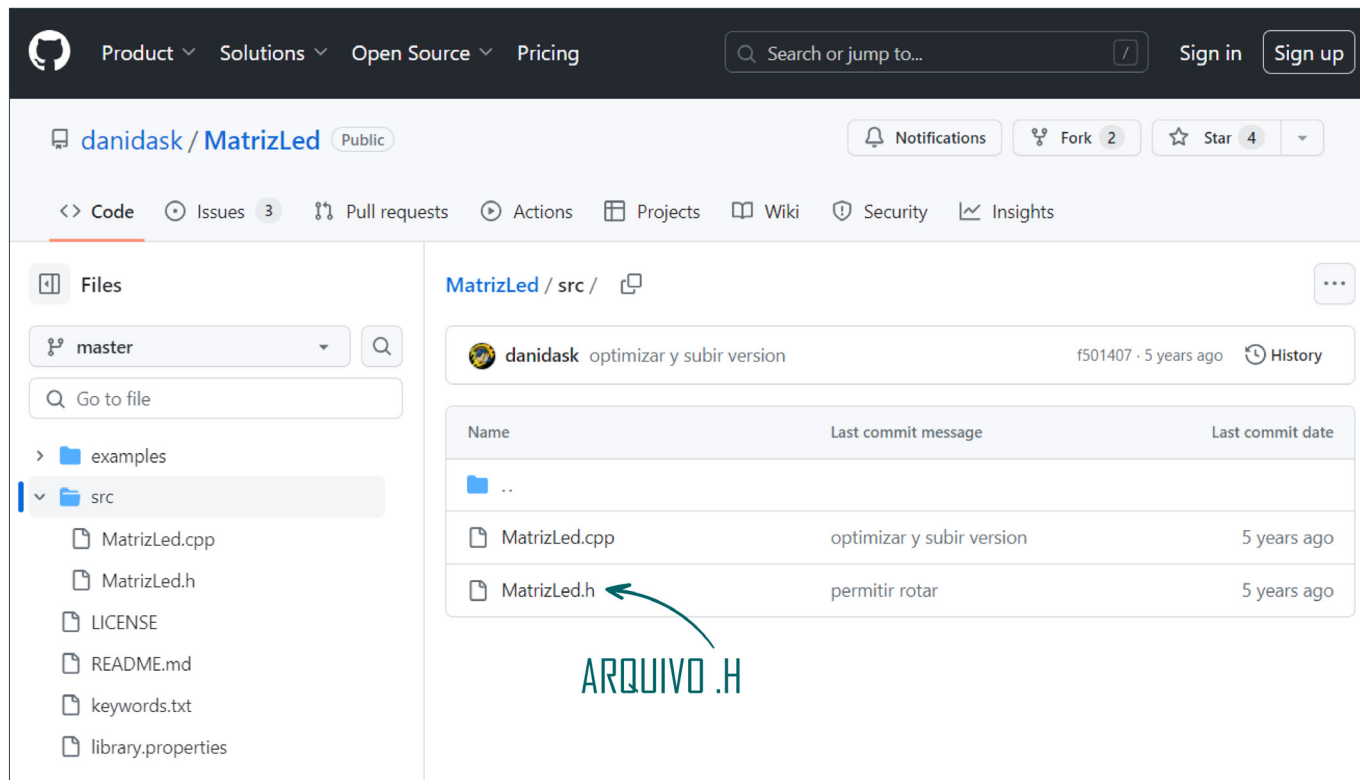
Fonte: GitHub



Robótica

Aula 07 Recursos das bibliotecas

Figura 14 – Arquivo .h da biblioteca MatrizLed



Fonte: GitHub

Abra o arquivo **.h** pelo próprio **GitHub** ou por um bloco de notas, se estiver salvo localmente, para visualizar seu conteúdo. Você encontrará muitas linhas de comentários e outras de estrutura. O arquivo **.h** (**header** ou cabeçalho) é essencial na programação do Arduino - é por ele que se define a interface da biblioteca e a implementação das funções declaradas nesse arquivo é feita no arquivo **.cpp** (código-fonte), o qual codifica os métodos de classe presentes no **.h**. Podemos observar outros detalhes sobre a criação de funções e as escolhas do desenvolvedor. Dentre elas, neste exemplo **MatrizLed**, é a personalização da biblioteca no próprio idioma, com funções em espanhol.

Explorando as pastas das bibliotecas, podemos encontrar também, pela organização do desenvolvedor, outros arquivos, como o arquivo **keywords.txt**, destinado à definição das cores aplicadas às palavras-chave da biblioteca para ficarem em destaque no sketch do Arduino.

Super dica

Para saber quais são **todas** as funções presentes na biblioteca e identificar parâmetros obrigatórios e adicionais, localize no arquivo **.h**, a sequência de comandos presentes na seção **public**.

The screenshot shows a code editor with the file `MatrizLed / src / MatrizLed.h` open. The file contains the following code:

```

115 class MatrizLed {
133
134     public:
135         /*
136          * Create a new controler
137          * Params :
138          * dataPin          pin on the Arduino where data gets shifted out
139          * clockPin         pin for the clock
140          * csPin            pin for selecting the device
141          * numDevices       maximum number of devices that can be controlled
142          */
143         MatrizLed();
144         void begin(int dataPin, int clkPin, int csPin, int numDevices=1);
145         void rotar(bool);
146         void escribirCaracter(char, int);
147         void escribirFrase(const char*);
148         void escribirFrase(const char*, int);
149         void escribirCifra(int);
150         void escribirCifra(int, int);
151         void escribirFraseScroll(const char*, unsigned long);
152         void borrar();
153         void setIntensidad(int);
154         void apagar();
155         void encender();
156
157         /*
158          * Gets the number of devices attached to this MatrizLed.

```

Annotations in the image:

- A green arrow points to the `class MatrizLed` line, with the label **CLASSE**.
- A green arrow points to the `public:` section, with the label **FUNÇÕES PÚBLICAS**.
- A green arrow points to the list of methods, with the label **COMANDOS DAS FUNÇÕES E PARÂMETROS**.

Robótica

Aula 07 Recursos das bibliotecas

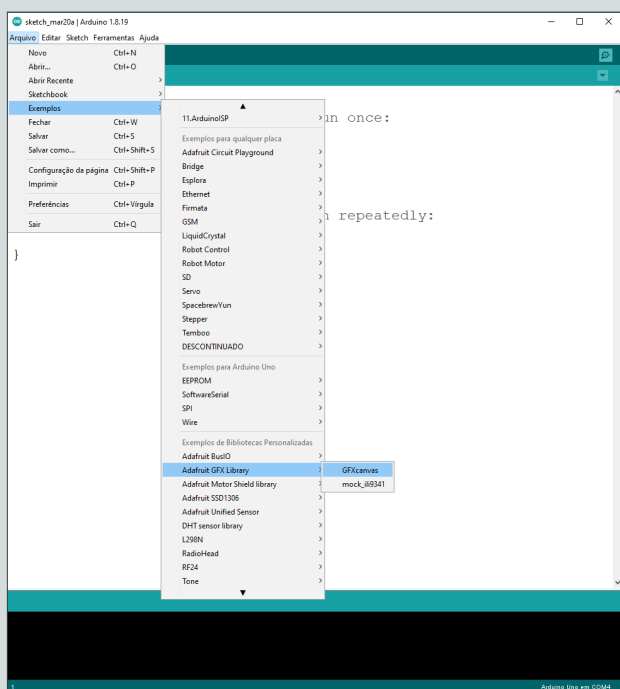
Uma vez explorado o arquivo **.h**, conforme indicado na **super dica**, como podemos utilizar as funções da biblioteca em uma programação? Na programação do seu projeto, após definir o objeto de controle, utilize o nome atribuído ao objeto de controle seguido pelo comando que você localizou na seção **public**, com um ponto entre eles.

Sintaxe de funções da biblioteca com identificação da classe e dos comandos na seção public do arquivo **.h**:

```
nome_atribuído_ao_objeto_de_controle.comando_listado_no_arquivo_h();
```

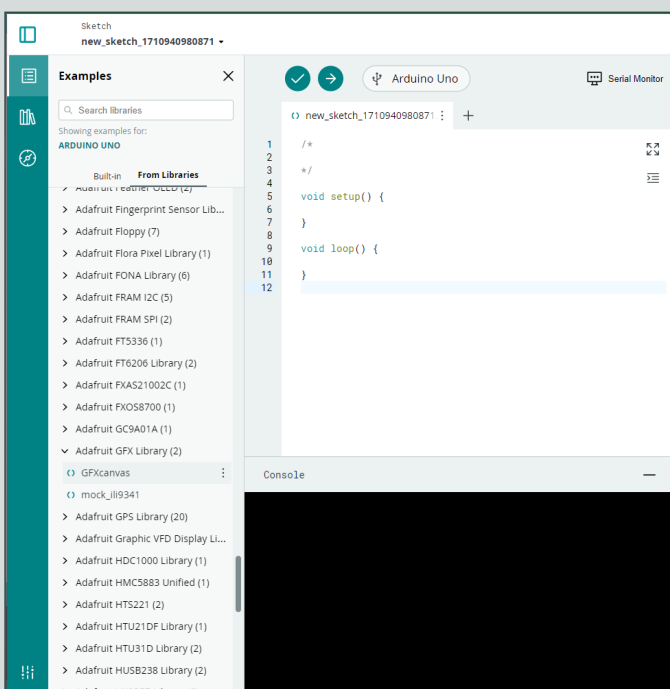
Conforme o exemplo de programação que você selecionar para **conhecer as outras bibliotecas**, o desenvolvedor pode ter adicionado mais linhas ou recursos que podem ser vistas no próprio no sketch da programação, como no exemplo da programação **GFXcanvas**, da **biblioteca Adafruit GFX**, indicada na **Aula 06 – Instax OLED [Parte II]** como complementar, conforme desafio proposto, à biblioteca principal utilizada para impressão das fotografias no display OLED, a **Adafruit SSD1306**.

Exemplo GFXcanvas no software Arduino IDE



Fonte: Arduino IDE Online

Exemplo GFXcanvas no Arduino IDE Online



Fonte: Software Arduino IDE

Neste exemplo da biblioteca **Adafruit GFX**, o sketch da programação é aberto com abas complementares, recurso que exploraremos mais na **Aula 08 - Técnicas de programação: sketch com abas**.

Portanto, dependendo da personalização dos exemplos de programação pelo desenvolvedor, o Arduino IDE poderá abrir diretamente no sketch da programação abas com as extensões **.h** e **.cpp**, além da aba principal com o exemplo da programação.

Como objetivo de ampliar o domínio sobre a programação do Arduino, seguiremos explorando recursos de desenvolvedores em outras aulas. Até lá!

Aba exemplo

```
GFXcanvas  GFXcanvasSerialDemo.cpp  GFXcanvasSerialDemo.h
/**
 * This example is intended to demonstrate the use of getPixel() ver
 * getRawPixel() and the fast horizontal and vertical drawing routines
 * in the GFXcanvas family of classes,
 *
 * When using the GFXcanvas* classes as the image buffer for a hardware
 * there is a need to get individual pixel color values at given physical
 * coordinates. Rather than subclasses or client classes call getBuffer
 * reinterpret the byte layout of the buffer, two methods are added to
 * GFXcanvas* classes that allow fetching of specific pixel values.
 *
 * * getPixel(x, y) : Gets the pixel color value at the rotated
 * the image.
 * * getRawPixel(x,y) : Gets the pixel color value at the unrotated
 * in the image. This is useful for getting the pixel value to map to
 * pixel location. This method was made protected as only the hardware
 * should be accessing it.
 *
 * The GFXcanvas*SerialDemo classes in this example will print to Serial
 * contents of the underlying GFXcanvas buffer in both the current rotation
 * and the underlying physical layout.
 */
#include "GFXcanvasSerialDemo.h"
#include <Arduino.h>

void setup() {
  Serial.begin(115200);

  // first create a rectangular GFXcanvas8SerialDemo object and d
```

Aba .cpp

```
GFXcanvas  GFXcanvasSerialDemo.cpp  GFXcanvasSerialDemo.h
#include "GFXcanvasSerialDemo.h"
#include <Arduino.h>

GFXcanvas1SerialDemo::GFXcanvas1SerialDemo(uint16_t w, uint16_t h
: GFXcanvas1(w, h) {}

void GFXcanvas1SerialDemo::print(bool rotated) {
  char pixel_buffer[8];
  uint16_t width, height;

  if (rotated) {
    width = this->width();
    height = this->height();
  } else {
    width = this->WIDTH;
    height = this->HEIGHT;
  }

  for (uint16_t y = 0; y < height; y++) {
    for (uint16_t x = 0; x < width; x++) {
      bool pixel;
      if (rotated) {
        pixel = this->getPixel(x, y);
      } else {
        pixel = this->getRawPixel(x, y);
      }
      sprintf(pixel_buffer, "%d", pixel);
      Serial.print(pixel_buffer);
    }
    Serial.print("\n");
  }
}
```

Aba .h

```
GFXcanvas  GFXcanvasSerialDemo.cpp  GFXcanvasSerialDemo.h
#ifndef __GFXcanvasSerialDemo__
#define __GFXcanvasSerialDemo__
#include <Adafruit_GFX.h>

/*****
 * @brief Demonstrates using the GFXcanvas classes as the backing
 * for a device driver.
 */
/*****
 * class GFXcanvas1SerialDemo : public GFXcanvas1 {
 * public:
 *   GFXcanvas1SerialDemo(uint16_t w, uint16_t h);
 *
 *   /*****
 *    * @brief Prints the current contents of the canvas to Serial
 *    * @param rotated true to print according to the current GFX
 *    * false to print to the native rotation of the canvas (or unrotated)
 *    */
 *   void print(bool rotated);
 * };
 *
 * /*****
 * @brief Demonstrates using the GFXcanvas classes as the backing
 * for a device driver.
 */
/****
```

Robótica

Aula 07 Recursos das bibliotecas

Desafios:

Que tal explorar os arquivos de outras bibliotecas e ver como cada desenvolvedor dá atenção a este importante recurso? Procure analisar ele apresenta seus códigos, tece comentários às linhas de código e estrutura sua programação.

E se você desenvolver uma biblioteca própria, mesmo para um código de programação simples? Pense em um projeto que já tenha realizado e nas funções utilizadas. A partir disto, revise o roteiro desta aula quanto à estrutura das bibliotecas e as dicas de sites para pensar no desenvolvimento da sua biblioteca. Converse com seus colegas sobre este desafio para que vocês possam trabalhar juntos neste desenvolvimento, planejando seu objetivo e funcionamento.

E se...

Eu não localizar informações relevantes ou não encontrar exemplos claros sobre as bibliotecas que tenho instalado e usado? Procure localizar a documentação dessa biblioteca no próprio site do Arduino. Outra opção é pesquisar o GitHub do desenvolvedor, utilizando as palavras-chave nome da biblioteca + nome do desenvolvedor, conforme descrito no Arduino IDE quando você localiza a biblioteca tanto na versão online quanto na versão software.

Feedback e finalização

O aprimoramento da programação do Arduino e desenvolvimento de projetos de Robótica vem da exploração de recursos e das tentativas de acerto e erro. Neste ponto em que você se encontra com a Robótica, já deve ter passado por várias experiências e desafios! Continue nesta jornada para descobrir e criar coisas cada vez mais incríveis!



Quer se aprofundar mais?

O arquivo **.h** contém as definições das classes, métodos, constantes e variáveis que são disponibilizados pela biblioteca. É aqui que são declaradas todas as funções “públicas” que os usuários da biblioteca poderão chamar, como, por exemplo, **MinhaBiblioteca(int pin)** e **void fazerAlgo()**, facilitando suas programações. São declaradas também as funções “privadas”, internas para funcionamento correto da biblioteca e que não utilizamos na programação principal.

As diretivas **#ifndef**, **#define** e **#endif** são utilizadas para evitar inclusões múltiplas, o que pode causar erros de compilação. E se a biblioteca depender de outras bibliotecas ou da biblioteca essencial, ou core do Arduino, elas precisarão ser incluídas no arquivo **.h** pela diretiva **#include**. Por exemplo, **#include <Arduino.h>** é comum para se ter acesso às definições padrão e funcionalidades do Arduino, como **pinMode()** e **digitalRead()**.

```
/*Exemplo de definição de classe e seus membros, incluindo
construtores, métodos e variáveis no arquivo .h */

#ifndef MinhaBiblioteca_h
#define MinhaBiblioteca_h

#include <Arduino.h>

class MinhaBiblioteca {
public:
    MinhaBiblioteca (int pin);
    void fazerAlgo();
private:
    int _pin;
};

#endif
```

```
/*Exemplo de codificação dos métodos da classe do exemplo
MinhaBiblioteca e função fazerAlgo() no arquivo .cpp */

#include"MinhaBiblioteca.h"

MinhaBiblioteca::MinhaBiblioteca(int pin) {

    pinMode(pin, OUTPUT);

    _pin = pin;
}

void MinhaBiblioteca::fazerAlgo() {

    digitalWrite(_pin,1);

}
```

Nestes exemplos de um arquivo **.h** e um arquivo **.cpp**, o código pode ser melhorado tanto para a própria organização do desenvolvedor quanto para a visualização de outros desenvolvedores com a inserção de comentários pelos recursos **//** (comentário em linha) ou **/* */** (comentário em bloco), o que auxilia na explicação do propósito de cada função e parâmetro.

Dica!

Confira os seguintes roteiros, com exemplos simplificados, para a criação de bibliotecas!



[Como fazer uma biblioteca para Arduino?](#)



[Criando suas próprias bibliotecas para Arduino](#)

REFERÊNCIAS

ARDUINO. **Built-in Examples**. Disponível em: <https://docs.arduino.cc/built-in-examples/>. Acesso em: 18 mar de 2024.

ARDUINO. **Documentação de Referência da Linguagem do Arduino**. Disponível em: <https://www.arduino.cc/reference/pt/>. Acesso em: 18 mar de 2024.

ARDUINO DESDE CERO. **Incluir Arduino.h: Cómo utilizarlo en tus proyectos**. Disponível em: <https://arduinodesdecero.com/cursos/incluir-arduino-h-como-utilizarlo-en-tus-proyectos/>. Acesso em: 22 mar de 2024.

ELETROGATE. **Criando uma Biblioteca para Arduino**. Disponível em: <https://blog.eletrogate.com/criando-uma-biblioteca-para-arduino/>. Acesso em: 22 mar de 2024.

EMBARCADOS. **Criando suas próprias bibliotecas para Arduino**. Disponível em: <https://embarcados.com.br/criando-bibliotecas-para-arduino/>. Acesso em: 22 mar de 2024.

MEDIUM. **Como fazer uma biblioteca para Arduino?** Disponível em: <https://medium.com/roboino/como-fazer-uma-biblioteca-para-arduino-d5da8bbb65d8>. Acesso em: 22 mar de 2024.



DIRETORIA DE TECNOLOGIAS E INOVAÇÃO (DTI)
COORDENAÇÃO DE TECNOLOGIAS EDUCACIONAIS (CTE)

EQUIPE ROBÓTICA PARANÁ

- Ailton Lopes
- Andrea da Silva Castagini Padilha
- Cleiton Rosa
- Darice Alessandra Deckmann Zanardini
- Edna do Rocio Becker
- Kellen Pricila dos Santos Cochinski
- Marcelo Gasparin
- Michele Serpe Fernandes
- Michelle dos Santos
- Roberto Carlos Rodrigues
- Sandra Aguera Alcova Silva

Os materiais, aulas e projetos da “Robótica Paraná”, foram produzidos pela Coordenação de Tecnologias Educacionais (CTE), da Diretoria de Tecnologia e Inovação (DTI), da Secretaria de Estado da Educação do Paraná (SEED), com o objetivo de subsidiar as práticas docentes com os estudantes por meio da Robótica. Este material foi produzido para uso didático-pedagógico exclusivo em sala de aula.



Este trabalho está licenciado com uma Licença
Creative Commons – CC BY-NC-SA
[Atribuição - NãoComercial - CompartilhaIgual 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/)

