Diretoria de Tecnologia e Inovação

ROBÓTICA

Módulo 2

Arduino: Bibliotecas e Funções

AULA 02

GOVERNO
DO ESTADO DO PARANÁ
SECRETARIA DE EDUCAÇÃO
E DO ESPORTE

GOVERNADOR DO ESTADO DO PARANÁ
Carlos Massa Ratinho Júnior

SECRETÁRIO DE ESTADO DA EDUCAÇÃO
Renato Feder

DIRETOR DE TECNOLOGIA E INOVAÇÃO
Andre Gustavo Souza Garbosa

COORDENADOR DE TECNOLOGIAS EDUCACIONAIS
Marcelo Gasparin

Produção de Conteúdo
Simone Sinara de Souza

Validação de Conteúdo
Cleiton Rosa

Revisão Textual
Adilson Carlos Batista

Projeto Gráfico e Diagramação
Edna do Rocio Becker

Ilustração
Jocelin Vianna

2021

Aula 01
O que já vimos?

Aula 02
Arduino: Bibliotecas e Funções

Aula 03
Código Morse

Aula 04
Semáforo Inteligente com IR

Aula 05
Semáforo Completo com Display

Aula 06
Matriz de LED 8X8

Aula 07
Desenhando na matriz de LEDs

Aula 08
Painel de Senhas

Aula 09
Escrevendo mensagens

Aula 10
Robô Autônomo

Aula 11
Sensor de Chuva

Aula 12
Sensor de Umidade do Solo

Aula 13
Irrigador Automático

Aula 14
Feedbacks + Inventário I

Aula 15
Teclado Matricial de Membrana

Aula 16
Servos Motores

Aula 17
Fechadura Eletrônica

Aula 18
Controlando Servos Motores

Aula 19
JoyStick Shield

Aula 20
Braço Robótico

Aula 21
Sensor de Movimento Presença

Aula 22
Sensor de Som

Aula 23
Sensor de Umidade e Temperatura

Aula 24
Termômetro Digital

Aula 25
Sensor de Gás e Fumaça

Aula 26
Acelerômetro e Giroscópio

Aula 27
Motor de Passo

Aula 28
Feedbacks + Inventário II

Aula 29
Receptor IR e Controle Remoto

Aula 30
Relé

Aula 31
Módulo RF 433mhz - I

Aula 32
Módulo RF 433mhz - II

Aula 33
Projeto CHAT via RF

Aula 34
Módulo Wireless

Aula 35
Comunicação do Módulo WI-FI em HTML

Aula 36
Módulo WI-FI - IoT com Sensores

Aula 37
Módulo WI-FI - IoT com Atuadores (LED)

Aula 38
Módulo WI-FI - IoT com Atuadores (Relé)

Aula 39
Monitor de Sensores em HTML I

Aula 40
Monitor de Sensores em HTML II

Aula 41
Mostra de Robótica

Aula 42
Feedbacks + Inventário III

Sumário

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

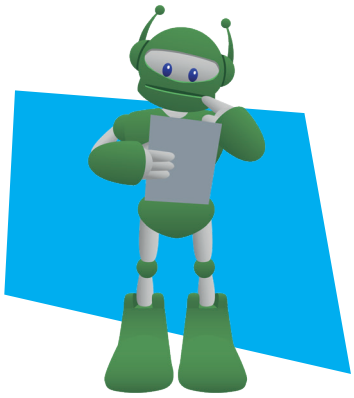
Introdução

No Módulo 1, conhecemos o conceito de software e a utilização desta lógica no campo da robótica, além disso, programamos, por meio da plataforma Arduino IDE, protótipos e robôs para realizarem determinadas funções.

Nesta aula, você terá a oportunidade de entender a aplicabilidade das bibliotecas presentes no Arduino IDE, bem como as etapas para criação de funções neste software, que executem, a partir de um comando, uma mesma tarefa constantemente.

Objetivos desta Aula

- Conceituar função e os elementos que a compõem;
- Entender o mecanismo de criação de funções no Arduino IDE;
- Destacar a aplicabilidade das bibliotecas do Arduino IDE.



AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Competências Gerais Previstas na BNCC

[CG02] - Exercitar a curiosidade intelectual e recorrer à abordagem própria das ciências, incluindo a investigação, a reflexão, a análise crítica, a imaginação e a criatividade, para investigar causas, elaborar e testar hipóteses, formular e resolver problemas e criar soluções (inclusive tecnológicas) com base nos conhecimentos das diferentes áreas.

[CG04] - Utilizar diferentes linguagens – verbal (oral ou visual-motora, como Libras, e escrita), corporal, visual, sonora e digital –, bem como conhecimentos das linguagens artística, matemática e científica, para se expressar e partilhar informações, experiências, ideias e sentimentos em diferentes contextos e produzir sentidos que levem ao entendimento mútuo.

[CG05] - Compreender, utilizar e criar tecnologias digitais de informação e comunicação de forma crítica, significativa, reflexiva e ética nas diversas práticas sociais (incluindo as escolares) para se comunicar, acessar e disseminar informações, produzir conhecimentos, resolver problemas e exercer protagonismo e autoria na vida pessoal e coletiva.

[CG09] - Exercitar a empatia, o diálogo, a resolução de conflitos e a cooperação, fazendo-se respeitar e promovendo o respeito ao outro e aos direitos humanos, com acolhimento e valorização da diversidade de indivíduos e de grupos sociais, seus saberes, identidades, culturas e potencialidades, sem preconceitos de qualquer natureza.

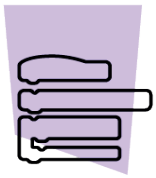
[CG10] - Agir pessoal e coletivamente com autonomia, responsabilidade, flexibilidade, resiliência e determinação, tomando decisões com base em princípios éticos, democráticos, inclusivos, sustentáveis e solidários.

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Habilidades do Século XXI a Serem Desenvolvidas

- Pensamento crítico;
- Afinidade digital;
- Resiliência;
- Resolução de problemas;
- Colaboração;
- Comunicação.



Roteiro da Aula

1. Contextualização (15min):

Durante as aulas do módulo 1, você aprendeu a programar tarefas para que diversos protótipos pudessem funcionar, utilizando, para isso, comandos disponíveis nas bibliotecas do software Arduino IDE. Ao longo da programação destes protótipos, você deve ter observado que alguns comandos se repetiam para que determinada tarefa fosse executada diversas vezes. Talvez, tenha se perguntado: será que não há um modo mais fácil de programar a execução de determinada tarefa sem a necessidade de repetir códigos?

A resposta a esta pergunta é sim. É possível criar trechos de código separados da estrutura principal do programa para executar tarefas repetitivas, sem a necessidade de escrevê-las detalhadamente cada vez que forem necessárias. Estes trechos de código recebem o nome de **função**, mas como isso funciona?

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Para entendermos o conceito e os elementos que compõem uma função, vamos utilizar a seguinte analogia: pense em uma atividade rotineira, repetida pelo menos três vezes ao dia, como, por exemplo, a alimentação, considerando, para tal, café da manhã, almoço e jantar. Como você realiza esta atividade?

2. Conteúdo (60min):

Toda vez que nos alimentamos, realizamos o mesmo procedimento, ou seja, pegamos um prato (de vidro, de porcelana ou de plástico, de tamanho variado conforme refeição, e com base redonda ou quadrada), depositamos o alimento sobre este prato (que pode ser um pão, frios, ovos, macarrão, arroz, feijão, carne, saladas, entre outros) com auxílio de talheres (colher, ou garfo e faca) manuseamos este alimento, abrimos a boca, depositamos o alimento, mastigamos e realizamos a deglutição do bolo alimentar. Cada parte deste processo você aprendeu na primeira infância. Isso significa que quando alguém te convida para almoçar, por exemplo, não precisa te explicar as etapas para este processo, pode simplesmente dizer "vamos almoçar!"

Na programação, esta mensagem "vamos almoçar!" é conhecida por **chamada de função**. Assim, podemos conceituar função como um trecho de códigos que permite realizar uma série de comandos pré-definidos.

O uso de funções ao longo da programação evita a repetição de um mesmo comando, organiza e facilita a leitura do código, deixando claro o conceito do programa, além de reduzir as hipóteses de erros em modificações no código, caso esse precise ser alterado.

Nos códigos do Arduino IDE, encontramos duas funções obrigatórias: a **void setup()** e a **void loop()** com suas respectivas variáveis. A função **void setup** determina a inicialização do programa e a configuração dos pinos da placa Arduino. Já, a função **void loop** é responsável em executar os comandos para a realização das tarefas, repetindo-se enquanto o Arduino estiver ligado (figura 1). Todavia, o software permite que o usuário crie suas próprias funções, através de seu editor de texto (Sketch).

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Figura 1 – Exemplo de código-fonte utilizando as funções void setup e void loop

```
void setup() {
  /* Configura o pino do LED como saída */
  pinMode(Pino_LED, OUTPUT);
  /* Configura o pino do botão como entrada */
  pinMode(Pino_Botao, INPUT);
}

void loop() {
  /* Lê o estado do botão e armazena na variável */
  Estado_Botao = digitalRead(Pino_Botao);

  /* Verifica o estado armazenado na variável e liga ou desliga o LED */
  if(Estado_Botao == HIGH){
    Estado = !Estado;
    digitalWrite(Pino_LED, Estado);
  }
}
```

Para que uma função seja criada, faz-se necessário conhecer os elementos que a compõem (tipo, nome, parâmetros e corpo) e suas etapas (declaração e delimitadores).

Quanto aos elementos, toda função é representada por:

a) Tipo de função - determina se a função retorna ou não um determinado valor. Faz parte da etapa de declaração de uma função, sendo representada por sintaxes, como, por exemplos:

- **int:** retorna valor de número inteiro;
- **float:** utilizada para números de ponto flutuante, ou seja, que possuem ponto decimal (valores fracionados);
- **boolean:** retorna um valor, seja ele verdadeiro (TRUE) ou falso (FALSE);
- **void:** não retorna nenhum valor (informação) para a função da qual foi chamada.

b) Nome da função - palavra utilizada no decorrer do código para chamar a função. Na figura 1, temos duas categorias de função declaradas (chamadas) pelos nomes setup() e loop(), respectivamente.

c) Parâmetros da função - são variáveis declaradas entre parênteses no cabeçalho de uma função. Correspondem às alterações em cada chamada de função, ou seja, enviam algum dado para a função quando ela é chamada.

d) Corpo da função - refere-se à tarefa que deve ser executada com a função. No código, o corpo da função será escrito entre chaves e apresentará todas as informações necessárias para execução da tarefa desejada. É representado, por exemplo, pelas sintaxes:

- **digitalWrite():** escreve um valor alto (HIGH) ou baixo (LOW) em um pino digital. Para pino analógico a sintaxe utilizada é analogWrite();
- **digitalRead()**: faz a leitura do valor de um pino digital especificado, devolvendo como resposta HIGH (alto) ou LOW (baixo). Quando o pino especificado for analógico, utiliza-se a sintaxe analogRead();
- **pinMode()**: faz a configuração do pino digital especificado para que se comporte como pino de entrada (INPUT) ou de saída (OUTPUT);
- **delay():** pausa a execução da atividade por um determinado tempo (em milissegundos) especificado como parâmetro. Cada segundo contém 1000 milissegundos;
- **tone():** gera um sinal de onda básica (quadrada) de frequência especificada em um pino da placa Arduino. Esta sintaxe é utilizada, por exemplo, para acionar frequência de nota musical (tom em Hz) em um Buzzer. Para interromper este sinal na programação, utiliza-se a sintaxe noTone.

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Para Saber Mais...

Na página Documentação de Referência da Linguagem Arduino, você encontra toda a linguagem de programação do Arduino, conforme estrutura, valores e funções.

https://www.arduino.cc/reference/pt/

Quanto às etapas, toda função inicia-se sendo declarada e utiliza os delimitadores, parênteses () e chaves {}, para indicar, respectivamente, a passagem de parâmetros de uma função e o início e fim de um bloco de códigos (figura 2).

Figura 2 – Anatomia de uma função

```
func(parâmetros)
{

  // Linhas de código pertencente à função func

}
```

Vale destacar que uma boa prática para criar funções no Arduino IDE é inseri-las fora das chaves das funções setup e loop, ou seja, antes ou depois da iniciação e configuração dos pinos que serão utilizados e execução de tarefas pré-determinadas no software.

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Agora que conhecemos os elementos necessários e alguns comandos utilizados na criação de uma função, vamos analisar um exemplo de código onde é possível criar função.

2.1 Criando função no software Arduino IDE

No módulo 1, **Aula 24 – Buzzer Passivo**, você conheceu as possibilidades de uso deste componente eletrônico e programou, através dele, a emissão de notas musicais, conforme mostra a figura 3.

Figura 3 – Programando o toque de notas musicais no Arduino IDE

```
/* Programa: Produzindo sons através de um buzzer */

/* Pino ligado ao buzzer */
int buzzer = 8

void setup()
{
  /* Define o pino do buzzer como Saída */
  pinMode(buzzer, OUTPUT);
}

void loop()
{
  /* Aciona o buzzer na frequência relativa ao Dó em Hz */
  tone(buzzer, 261);
  /* Espera um tempo para desativar */
  delay(200);
  /* Desativa o buzzer */
  noTone(buzzer);
  /* Aciona o buzzer na frequência relativa ao Ré em Hz */
  tone(buzzer, 293);
  delay(200);
  noTone(buzzer);
  /* Aciona o buzzer na frequência relativa ao Mi em Hz */
  tone(buzzer, 329);
  delay(200);
  noTone(buzzer);
  /* Aciona o buzzer na frequência relativa ao Fá em Hz */
  tone(buzzer, 349);
```

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

```
30    delay(200);
31    noTone(buzzer);
32    /* Aciona o buzzer na frequência relativa ao Sol em Hz */
33    tone(buzzer, 392);
34    delay(200);
35    noTone(buzzer);
36    delay(2000);
37  }
```

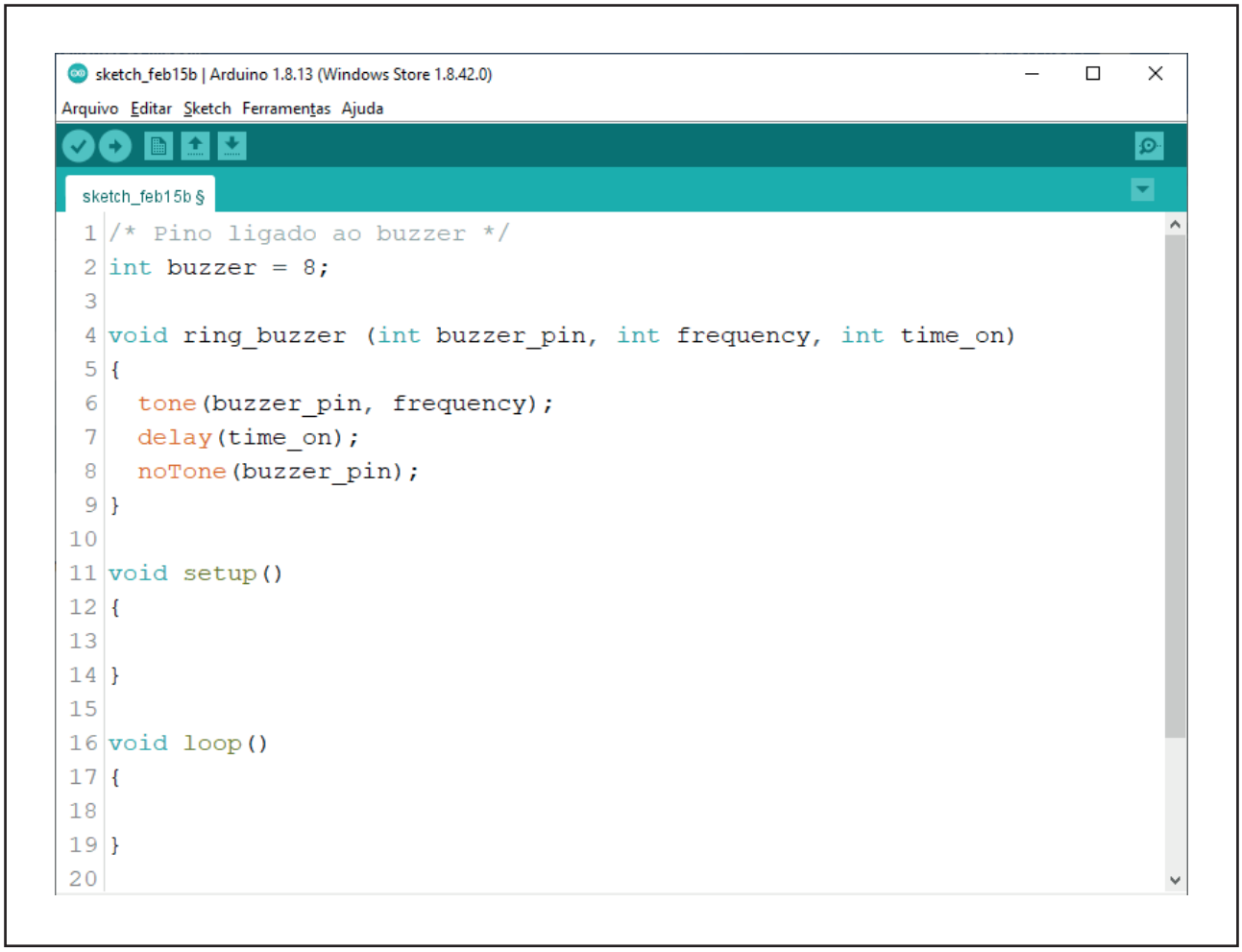
No código de programação da figura 3, é possível observar que as sintaxes tone, delay e noTone repetem-se a cada frequência de nota musical. Esta repetição torna o código extenso e com maior chance de erros, caso haja necessidade de alterá-lo. Assim, o uso de funções é recomendado para facilitar a leitura e reutilização deste código.

A primeira ação para criar uma função é acessar o Arduino IDE e criar uma aba, em branco, para a escrita de sketches (na **Aula 5 – Arduino IDE e mBlock** do Módulo 1, comentamos como criar aba na categoria Sketch). Nesta nova aba, cria-se um novo bloco, antes ou depois dos blocos **setup()** e **loop()**, e define-se a função (figura 4).

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Figura 4 - Declarando uma função no Arduino IDE



Ao analisarmos a figura 4, observamos na linha 4, que a função é declarada pela sintaxe void, com o nome de ring_buzzer e os parâmetros são representados, entre parênteses, pelas sintaxes: int buzzer_pin, int frequency e int time_on. Nas linhas, entre chaves, é apresentado o corpo da função por meio das sintaxes: tone, delay e noTone.

Uma vez criada a função, faz-se necessário informar quais variáveis farão parte desta função. No exemplo utilizado, as variáveis definem a frequência das notas musicais, e são descritas na função void loop (a partir da linha 17), representada na figura 5.

Figura 5 - Variáveis da função ring_buzzer

```
/* Pino ligado ao buzzer */
int buzzer = 8;

void ring_buzzer (int buzzer_pin, int frequency, int time_on)
{
  tone(buzzer_pin, frequency);
  delay(time_on);
  noTone(buzzer_pin);
}

void setup()
{
  /* Define o pino do buzzer como saída */
  pinMode(buzzer, OUTPUT;)
}

void loop()
{
  /* Aciona o buzzer na frequência relativa ao Dó em Hz por 200 ms*/
  ring_buzzer(buzzer, 261, 200);
  /* Aciona o buzzer na frequência relativa ao Ré em Hz por 200 ms*/
  ring_buzzer(buzzer, 293, 200);
  /* Aciona o buzzer na frequência relativa ao Mi em Hz por 200 ms*/
  ring_buzzer(buzzer, 392, 200);
  /* Aciona o buzzer na frequência relativa ao Fá em Hz por 200 ms*/
  ring_buzzer(buzzer, 349, 200);
  /* Aciona o buzzer na frequência relativa ao Sol em Hz por 200 ms*/
  ring_buzzer(buzzer, 349, 200);
}

```

Comparando o código-fonte apresentado na figura 3, com o código da figura 5, podemos notar o encurtamento do código, bem como a organização e facilidade de leitura. Além destas vantagens, a função criada poderá ser reutilizada em outros projetos que utilizam o componente eletrônico Buzzer passivo, tornando ágil a programação.

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

Vale ressaltar que alguns componentes eletrônicos possuem funções pré-estabelecidas por seus fabricantes, as quais facilitam o desenvolvimento de aplicações de tarefas, e estão disponíveis em bibliotecas do software Arduino.

2.2 Utilizando bibliotecas do Arduino IDE

Na programação, o termo biblioteca refere-se a um conjunto de instruções que definem funcionalidades específicas à execução de tarefas para determinado componente eletrônico. Além disso, a biblioteca organiza o código e facilita a leitura dos comandos presentes nesse.

No Arduino IDE, há três modelos de bibliotecas que se diferenciam quanto à usabilidade. Sendo elas:

a. Biblioteca Essencial ou **Core:** conforme o nome indica, este modelo de biblioteca é imprescindível para o desenvolvimento de projetos, desde o mais simples, como o acender de um LED, até os mais complexos, que necessitam de comandos pertencentes a outras bibliotecas para sua realização. Além do mais, as funções presentes neste modelo de biblioteca, como, por exemplo: digitalRead, digitalWrite, analogRead, entre outras, não precisam ser declaradas.

b. Bibliotecas Padrão: este modelo de biblioteca, apesar de estar presente na instalação do Arduino IDE, somente participa do projeto quando declarada no início do código escrito no sketch, por meio da sintaxe #include. Isso porque o software possui recursos limitados de memória.

c. Bibliotecas de Terceiros: correspondem a bibliotecas instaladas no Arduino IDE por desenvolvedores, com o objetivo de fornecer funcionalidades não presentes em nenhuma biblioteca do software ou adicionar novas funções às bibliotecas já existentes.

No módulo anterior, você aprendeu a instalar bibliotecas no Arduino IDE. Para rever as etapas necessárias desta ação, retorne ao conteúdo da **Aula 5 – Arduino IDE** e **mBlock**.

AULA 02

ARDUINO: BIBLIOTECAS E FUNÇÕES

3. Feedback e Finalização (15min):

Você e seus colegas compreenderam a importância de criar funções que possam, a partir de um comando, executar tarefas repetitivas? Conseguem exemplificar em quais outras programações, trabalhadas no Módulo 1, são possíveis criar funções? Compartilhem suas impressões e ideias.