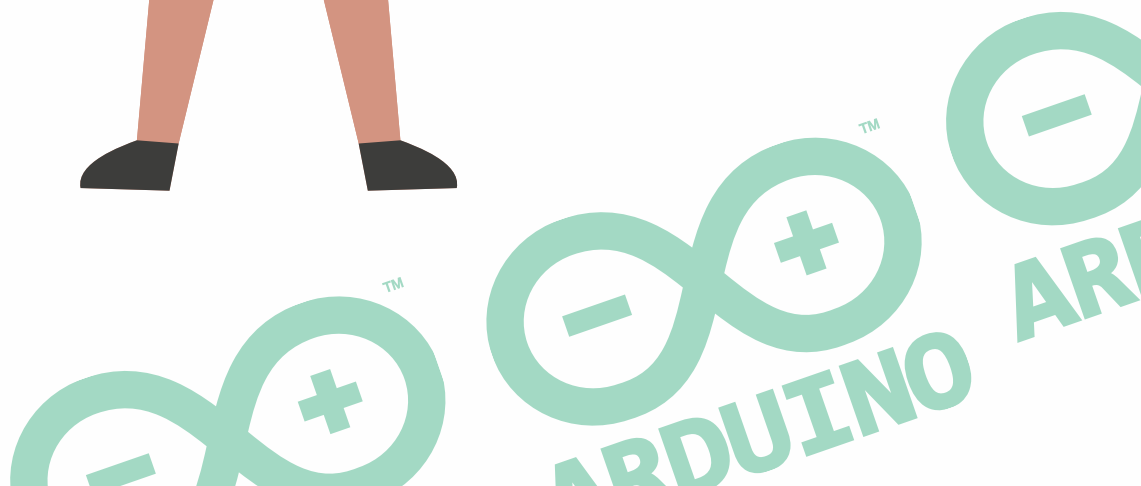
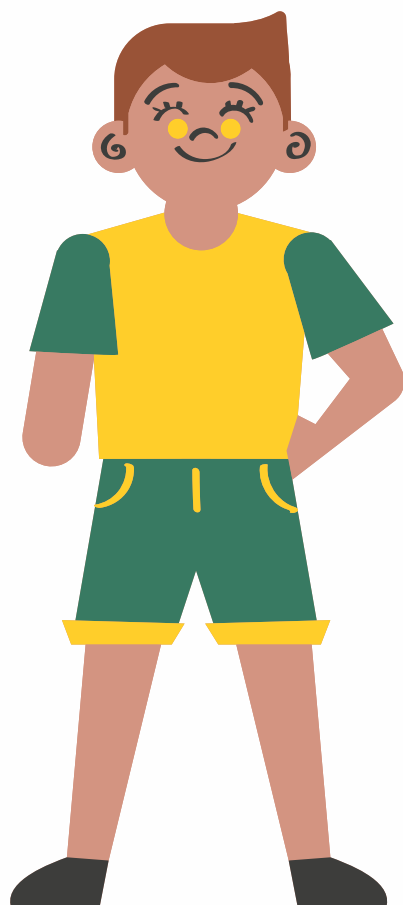


Manual Didático

SIGABEM

APOIO AO EMBARQUE

DAS OFICINAS DO PROJETO
NAS ESCOLAS PÚBLICAS



APRESENTAÇÃO:

ESTE MANUAL DIDÁTICO É FRUTO DO PROJETO FINANCIADO PELA SETEC/MEC POR MEIO DO EDITAL Nº 67/2021- CHAMADA PÚBLICA DE PROJETOS PARA APOIO À IMPLEMENTAÇÃO DAS OFICINAS 4.0 INTITULADO DESENVOLVIMENTO DE SOLUÇÃO TECNOLÓGICA PARA SOLICITAÇÃO DE APOIO AO EMBARQUE DE PESSOAS COM DEFICIÊNCIA AO TRANSPORTE PÚBLICO NA REGIÃO METROPOLITANA DO RECIFE (SIGABEM - APOIO AO EMBARQUE).

UM DOS OBJETIVOS ESPECÍFICOS DO PROJETO É MULTIPLICAR OS CONTEÚDOS E CONHECIMENTOS ADQUIRIDOS NAS OFICINAS 4.0 EM ESCOLAS PÚBLICAS MUNICIPAIS, DE FORMA A ESTIMULAR A IMERSÃO DE ESTUDANTES EM TEMÁTICAS COMO EMPREENDEDORISMO, INOVAÇÃO E USO DAS TECNOLOGIAS DIGITAIS PARA SOLUÇÃO DE PROBLEMAS REAIS PARA ACESSIBILIDADE DE PESSOAS COM DEFICIÊNCIA.

ESTA PUBLICAÇÃO CONTÉM CONCEITOS BÁSICOS DE ELETRICIDADE, PROGRAMAÇÃO COM ARDUÍNO, LORA, ENTRE OUTROS, ALÉM DA APRESENTAÇÃO DOS COMPONENTES BÁSICOS UTILIZADOS PARA O DESENVOLVIMENTO DO HARDWARE (DISPOSITIVO) RESULTADO DO PROJETO. POR FIM, O MANUAL APRESENTA O PASSO A PASSO DE COMO MONTAR UM PEQUENO PROJETO COM ARDUÍNO- PISCA LED COM EXEMPLO PRÁTICO DE APLICAÇÃO DOS CONCEITOS TRABALHADOS, ONDE, DURANTE AS OFICINAS NAS ESCOLAS PÚBLICAS SELECIONAS, OS ESTUDANTES PUDEAM EXECUTÁ-LO.

Dados Internacionais de Catalogação na Publicação (CIP) (Câmara Brasileira do Livro, SP, Brasil)

Manual didático sigabem : apoio ao embarque [livro eletrônico] : das oficinas do projeto nas escolas públicas. -- 1. ed. -- Recife, PE : Ed. dos Autores, 2024.
PDF

Vários autores.
ISBN 978-65-01-18746-4

1. Arduino (Linguagem de computador)
2. Eletricidade (Ensino fundamental) 3. Hardware.

24-232720

CDD-372.35

Índices para catálogo sistemático:

1. Eletricidade : Ciências : Ensino fundamental
372.35

Aline Grazielle Benitez - Bibliotecária - CRB-1/3129

AUTORES:

Ioná Maria Beltrão Rameh Barbosa

Aida Araújo Ferreira

Vânia Soares de Carvalho

Gilmar Gonçalves de Brito

Mizael Claudino Lins da Silva

Maria Eduarda Soares da Silva

Tales Matheus Alves dos Passos

IFPE Campus Recife - Coordenadora do projeto

IFPE Campus Recife - Coordenadores

IFPE Campus Recife - Coordenadores

IFPE Campus Recife - Coordenadores

Estudante: Superior de Engenharia Mecânica do IFPE

Estudante: Técnico de Eletrônica do IFPE

Estudante: Técnico de Eletrônica do IFPE

Sumário

- O que é o projeto Sigabem?
- Eletricidade básica
 - Resistor
 - Potenciômetro
 - Jumpers
 - Protoboard
 - Lei de ohm
- Arduino
 - Pinagem
 - Energização
 - Características
- Programação Arduino
 - IDE
 - Circuito led
 - Comentários
 - Variáveis
 - Tipos de dados
 - Bibliotecas
- LoRa
 - Rede
- Display
 - Ligações
 - programação
- Teclado Matricial
 - Ligações
 - Programação
- Autofalante
 - Ligações
 - Programação
- Módulo SD
 - Ligações
 - Programação
 - Cartão SD
- Nosso circuito
 - Ligações
 - Programação



Projeto Sigabem

Apoio ao embarque

MOTIVAÇÃO



POR QUE INICIAMOS ESSE PROJETO DE PESQUISA?

A média diária de usuários do transporte coletivo no Recife é de cerca de 2 milhões de pessoas. Durante os horários de maior fluxo, especialmente nas regiões centrais, as condições de embarque tornam-se caóticas.

Observe na figura:



Fonte: Mobilize Brasil (Recife).

Projeto Sigabem

Apoio ao embarque

Considerando o cenário de embarque nas grandes cidades, imagine a dificuldade enfrentada pelas pessoas com deficiência ao tentar acessar o transporte coletivo em um ambiente tão caótico.



COMO PODEMOS MELHORAR AS CONDIÇÕES DE EMBARQUE NO TRANSPORTE PÚBLICO PARA PESSOAS COM DEFICIÊNCIA?

SOLUÇÃO

Para facilitar o embarque de pessoas com deficiência nos pontos de ônibus, desenvolvemos um dispositivo que permite ao usuário, de forma independente e segura, solicitar o ônibus desejado, estabelecendo uma comunicação antecipada com o veículo.

Já solicitou um motorista por aplicativo?

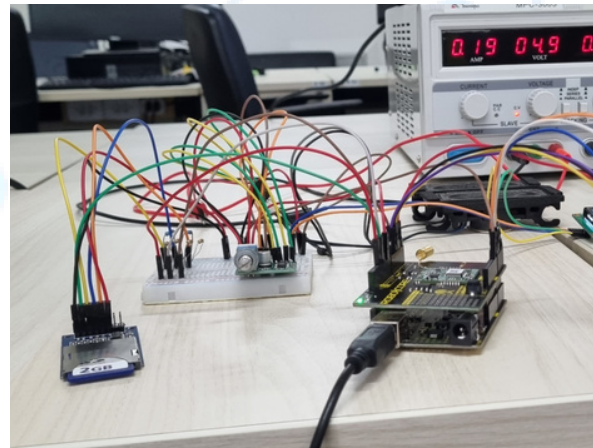
A dinâmica é parecida: a pessoa com deficiência (PCD) se comunica com o coletivo através do nosso dispositivo, expressando seu interesse, e logo em seguida, é informada sobre o tempo estimado de chegada do transporte.

ELETRICIDADE

BÁSICA

O QUE É UM CIRCUITO:

Um circuito eletrônico pode ser definido como um sistema de elementos que conduzem eletricidade, transformando-a em energia, dados e força. Há uma interdependência entre os componentes, que necessitam de conexões e configurações adequadas para garantir o funcionamento eficaz do sistema.



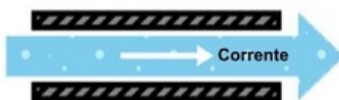
RESISTOR

Resistência elétrica é a capacidade de um material se opor a passagem da corrente elétrica. Já o resistor é um componente elétrico passivo que tem a função primária de limitar o fluxo da corrente elétrica em um circuito.

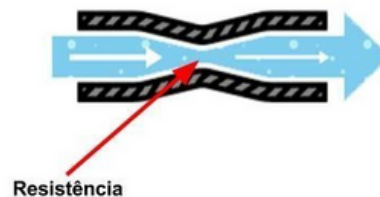


Podemos pensar no resistor como:

Cano de água
sem resistência permite
corrente máxima.



Cano de água
com resistência que
limita a corrente!



Precisamos selecionar o resistor com atenção para não haver sobrecarga no circuito. Isto é: se a corrente aplicada for maior que a resistência fornecida, o circuito tem seu funcionamento prejudicado.

Para selecionarmos o resistor adequado, possuímos a tabela de cores.

COR	1ª FAIXA	2ª FAIXA	3ª FAIXA	4ª FAIXA	5ª FAIXA
	VALOR	VALOR	VALOR	MULTIPLICADOR	TOLERÂNCIA
PRETO		0	0	X 1 Ω	
MARROM	1	1	1	X 10 Ω	Mais ou menos 1 %
VERMELHO	2	2	2	X 100 Ω	Mais ou menos 2 %
LARANJA	3	3	3	X 1.000 Ω	
AMARELO	4	4	4	X 10.000 Ω	
VERDE	5	5	5	X 100.000 Ω	Mais ou menos 0,5%
AZUL	6	6	6	X 1.000.000 Ω	Mais ou menos 0,25%
VIOLETA	7	7	7	X 10.000.000 Ω	Mais ou menos 0,1%
CINZA	8	8	8		Mais ou menos 0,05%
BRANCO	9	9	9		
DOURADO				X 0.1 Ω	Mais ou menos 5 %
PRATA				X 0.01 Ω	Mais ou menos 10 %
SEM COR					Mais ou menos 20 %

POTENCIÔMETRO

Os potenciômetros são resistores variáveis com uma derivação no centro que permite por meio de movimentos lineares ou rotativos mudar o valor de resistência podendo ser configurado como potenciômetro propriamente dito (variando a tensão) ou reostato que varia a corrente elétrica.



JUMPERS

Jumpers são os cabos que utilizamos para efetuar as ligações do nosso circuito. Podendo diferenciar-se entre macho e fêmea.



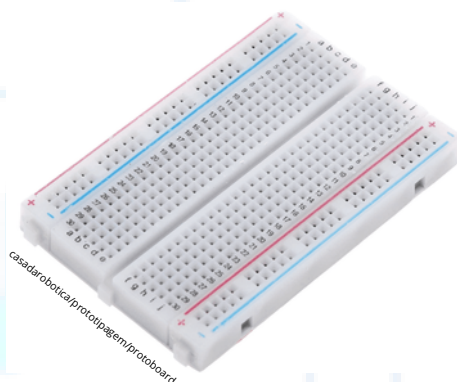
<https://www.farnell.com/jumpers-machos-femeas-x-40-unidades-30cm>



MUITO UTILIZADOS EM PROTOBOARDS, OS COMPONENTES ELETRÔNICOS SÃO ADEQUADOS A ESTES CABOS.

PROTOBOARD

Protoboard é uma placa que quando alimentada com eletricidade, nos permite criar circuitos práticos e autônomos. Permitindo a mudança dos *jumpers* e ligações do circuito.



Tensão...

A diferença de potencial entre dois pontos, conhecida como DDP, é responsável pela movimentação das cargas elétricas, gerando a corrente.

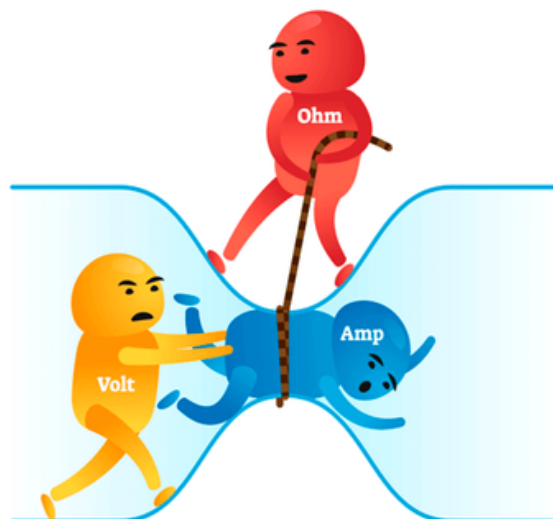
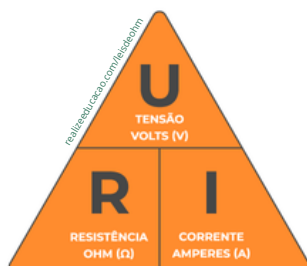
Corrente...

É o movimento organizado de elétrons causado pela diferença de potencial elétrico entre as extremidades do condutor.

1º LEI DE OHM

A lei de ohm nos permite calcular grandezas físicas como a tensão, corrente e resistência elétrica. A lei de Ohm nos afirma que a tensão elétrica é igual ao produto da resistência elétrica pela corrente elétrica.

$$U=R.I$$

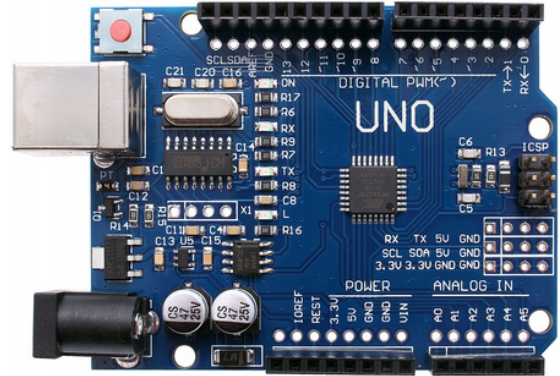


intensivacursos.com

ARDUINO

O Arduino é uma placa de prototipagem eletrônica de código aberto baseada em *hardware* e *software* fáceis de usar.

Através do arduino podemos acionar leds, enviar instruções para o microcontrolador, ler entradas em sensor.



produto.mercadolivre.com.br/MLB-1815702587-placa-compativel-arduino-uno

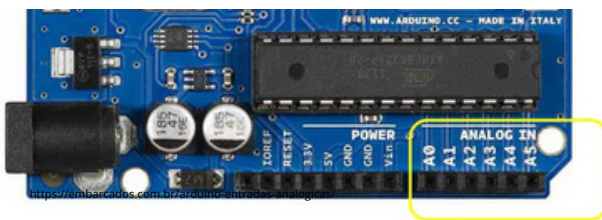
PORTAS DO ARDUINO...

As portas do arduino são os meios pelos quais ele consegue trocar interações com os meios externos.



<https://embarcados.com.br/pinos-digitais-do-arduino/>

O arduino uno oferece 14 portas digitais, que podem ser usadas para entradas e saídas, e que podem comandar 14 dispositivos externos.



Entradas Analógicas

O Arduino possui 6 portas analógicas, que funcionam exclusivamente como entradas. Essas portas recebem sinais com amplitude variável entre 0 e 5 volts, geralmente provenientes de sensores.

ALGUMAS PORTAS DIGITAIS SÃO ESPECÍFICAS PARA PWM (MODULAÇÃO POR LARGURA DE PULSO), UM MÉTODO QUE PERMITE GERAR TENSÕES INTERMEDIÁRIAS ENTRE 0 E 5 VOLTS. ESSA TÉCNICA É ESPECIALMENTE ÚTIL PARA CONTROLAR DISPOSITIVOS COMO LEDS, MOTORES, ENTRE OUTROS.

PROGRAMAÇÃO

ARDUINO

ARDUINO IDE

Vamos falar agora sobre o ambiente de desenvolvimento integrado (IDE) do Arduino! Mas calma, o que seria isso?



O código no Arduino é escrito em uma versão modificada de C++, uma linguagem de alto nível.

ALTO NÍVEL



BAIXO NÍVEL

É o tipo de linguagem que mais se aproxima a linguagem computacional

É o tipo de linguagem que mais se aproxima a linguagem humana

Apesar de escrevermos em linguagens de alto ou baixo nível, o microcontrolador não as compreende diretamente. Então, por que utilizamos essas linguagens? Simples: para que possam ser convertidas em linguagem de máquina, um processo conhecido como compilação.

O Arduino IDE é um ambiente de desenvolvimento voltado especificamente para o Arduino. Nele, podemos escrever, compilar e enviar o código diretamente para o dispositivo.

Ele oferece todas as ferramentas necessárias integradas, incluindo um corretor de sintaxe, um compilador, entre outras funcionalidades.

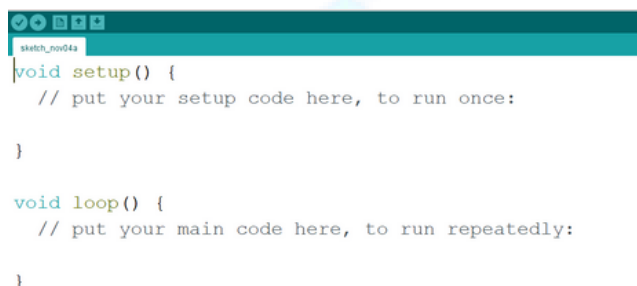
Como baixar o Arduino IDE? É simples. Qualquer computador com Windows, macOS ou Linux e acesso à internet pode realizar o download. Basta acessar o site 'www.arduino.cc/en/software', selecionar a versão mais recente do Arduino e fazer o download do arquivo conforme mostrado na imagem.



Em seguida, clique em 'Just Download' para baixar o Arduino IDE gratuitamente. Após o download, descompacte a pasta e crie um atalho do arquivo do Arduino IDE na sua área de trabalho.

ENTENDENDO O AMBIENTE DE DESENVOLVIMENTO INTEGRADO (IDE)

Após realizar o download e abrir o programa, você encontrará a seguinte tela:



```
sketch_nov04a
void setup() {
  // put your setup code here, to run once:
}

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



Na parte superior esquerda, encontramos cinco botões: o botão para verificar o código, o botão para carregar o código, o de criar um novo arquivo, o de abrir um arquivo existente e o de salvar o arquivo atual. Esses são os principais recursos que você precisa conhecer inicialmente.

Os últimos três botões são autoexplicativos: um, cria um novo arquivo, outro abre um arquivo existente em seu dispositivo e o último salva as alterações feitas. O primeiro botão verifica se o código contém erros de sintaxe, enquanto o segundo envia tudo que você escreveu no IDE para a placa através do conector USB mencionado anteriormente.

PRIMEIRAS LINHAS DE CÓDIGO

Após explicar os botões que usaremos com mais frequência e os conceitos básicos de programação, que tal começarmos a explorar as primeiras linhas de código? Vamos primeiro entender o que são 'void setup' e 'void loop'.

VOID SETUP

O void setup é uma função que executa todas as suas linhas apenas uma vez, sempre que o Arduino é ligado ou reiniciado.



VOID LOOP

O void loop é uma função executada repetidamente em um loop, a menos que ocorra um 'break', interrompendo a execução do código.

As funções são 'abertas' e 'fechadas' com dois colchetes '{}'. Tudo que estiver dentro desses colchetes faz parte do corpo da função correspondente.

Mas como será que isso funciona na prática?



NOSSO PRIMEIRO PROJETO ARDUINO

PISCAL LED

Neste exemplo, vamos apenas acender e apagar um LED repetidamente. Para isso, primeiro você precisará escolher um pino que será responsável pelo sinal do Arduino para o LED. Vamos escolher o pino 2. Assim, no setup, você deve escrever:

```
pinMode( 2, OUTPUT);
```

A função `pinMode()` é utilizada para definir se o pino será configurado como saída ou entrada. Os argumentos dessa função especificam, primeiro, o pino escolhido e, em segundo lugar, se ele será configurado como INPUT (por exemplo, um botão) ou OUTPUT (como no nosso caso, um LED).

Não se esqueça de adicionar um ponto e vírgula ao final da função. No `void loop`, você irá definir o que acontecerá na porta digital escolhida. Assim, escreveremos entre os colchetes dessa função:

```
digitalWrite( 2, HIGH);  
delay( 500);  
digitalWrite( 2, LOW);  
delay( 500);
```

A função `digitalWrite` utiliza o primeiro argumento para especificar o número do pino e o segundo para definir o estado da porta digital (HIGH para ligado e LOW para desligado). A função `delay`, por sua vez, introduz uma pausa entre as linhas de código, em milissegundos. Durante esse intervalo, o sistema permanece inativo, sem realizar cálculos ou quaisquer outras ações.

O código ficará assim:

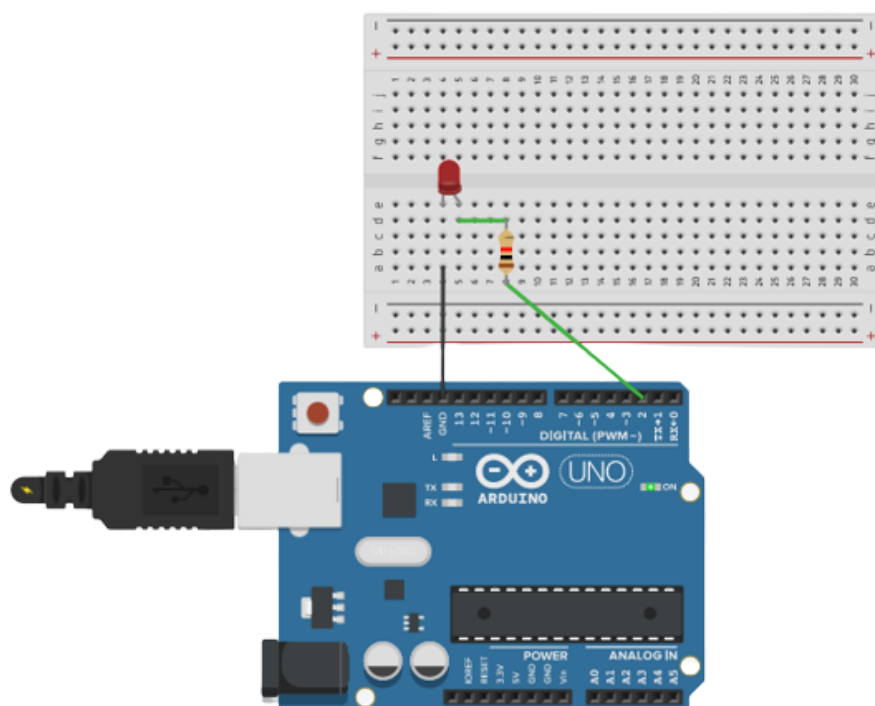
```
sketch_nov04a  
  
void setup()  
{  
    pinMode(2, OUTPUT);  
}  
  
void loop()  
{  
    digitalWrite(2, HIGH);  
    delay(500);  
    digitalWrite(2, LOW);  
    delay(500);  
}
```

NOSSO PRIMEIRO PROJETO ARDUINO

PISCAL LED

O LED possui uma parte maior e uma parte menor para conexão na protoboard. Vamos conectar a parte menor (polo negativo) ao GND (terra) do Arduino. A parte maior (polo positivo) deve ser ligada a um resistor, e utilizaremos um jumper para conectar esse resistor ao pino 2 do Arduino.

Para o circuito basta seguir a imagem e orientações abaixo:



AGORA, BASTA COMPILAR E CARREGAR O CÓDIGO NO ARDUINO. VOCÊ VERÁ O LED PISCANDO REPETIDAMENTE A CADA 0,5 SEGUNDOS.

BIBLIOTECAS ARDUINO

As bibliotecas são, na verdade, 'extensões' das funções básicas disponíveis no Arduino. Elas oferecem uma variedade de funções e tipos de dados adicionais, além dos que já estão incluídos no IDE.



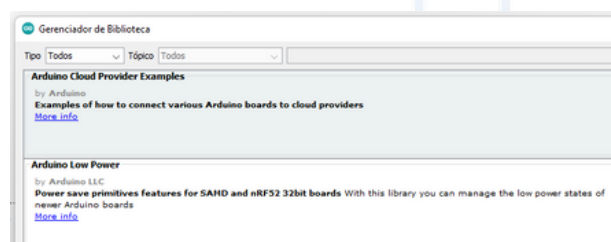
Primeiramente, é importante saber qual biblioteca você deseja utilizar. Por exemplo, se quiser usar a biblioteca 'Wire', basta incluir a linha `#include <Wire.h>` no início do código. Dessa forma, a biblioteca será 'importada' e você poderá acessar seus recursos a partir desse ponto.

```
#include <Wire.h>
```

Outra maneira de fazer isso é abrir o IDE do Arduino, clicar na opção 'Sketch' no canto superior esquerdo, selecionar 'Incluir Biblioteca' e, em seguida, escolher a biblioteca 'Wire'.

Baixando bibliotecas

Caso não encontre a biblioteca desejada nas bibliotecas básicas do IDE, basta ir em "Sketch", clicar, "incluir biblioteca" e "gerenciar bibliotecas". Algo assim irá aparecer:



Aqui, você encontrará uma vasta seleção de bibliotecas de diversos fornecedores, além da opção de atualizar ou baixar versões anteriores das bibliotecas já instaladas. Assim que o download for concluído, a biblioteca estará disponível para inclusão.

Se você ainda não encontrar a biblioteca que procura, é possível baixar um arquivo ZIP da internet e importá-lo para a IDE. Para fazer isso, siga o caminho 'Sketch', selecione 'Incluir Biblioteca' e, por fim, clique em 'Adicionar Biblioteca ZIP'. Após selecionar o arquivo, ele estará disponível em 'Incluir Biblioteca'.



COMENTÁRIOS

Você também pode adicionar informações para consulta futura ao programar um código no Arduino. Essas áreas são irrelevantes para o sistema, mas extremamente úteis para o programador, permitindo que ele faça comentários, anotações ou registre dúvidas sem interferir no funcionamento do código. Existem duas maneiras de escrever comentários: para uma única linha, use `//comentário`, e para múltiplas linhas, utilize `/*comentário*/`.

```
void setup()
{
  pinMode(2, OUTPUT);
  //comentário de uma linha
}

void loop()
{
  /*comentários
  de múltiplas linhas
  */
  digitalWrite(2, HIGH);
  delay(500);
}
```

VARIÁVEIS

Variáveis são identificadores que usamos para armazenar dados ou partes do nosso código. Por exemplo:

```
int x = 2;
```

Neste caso, eu defini o tipo de dado da variável. Se quiser saber quais tipos estão disponíveis, basta consultar a tabela ao lado.

Assim, a variável 'x' é um inteiro com o valor atribuído de 2. Ao longo do código, posso usar o nome da variável em vez de escrever o valor diretamente. Por exemplo:

```
int x = 2;
void setup()
{
  pinMode(x, OUTPUT);
}
```

ATRIBUIÇÃO DE VARIÁVEIS DE DIFERENTES NATUREZAS

```
char letra = 'D';
String texto = "Hello";
int inteiro = 5;
double realGrande = 9.98;
float realPequeno = 5.99;
bool verdadeiroOuFalso = true;
```

TIPOS DE DADOS

O tipo de dado varia dependendo do valor que ele guarda. Entre os tipos de dados temos alguns principais:

INT: Esse guarda dados de números inteiros: 1, 2, -1, 110, etc.

CHAR: Esse tipo de dado guarda informação quanto aos caracteres, como a, b, F, T, u, etc.

STRING: esse tipo de dado guarda informações de texto como "Arduino", "olá", "tudo bem?", etc.

FLOAT: Guarda informações de números Reais com até 6 casas decimais como 1.48, 7.3333, 5.87, etc.

DOUBLE: como o float, porém com até 10 casas decimais.

BOOL: guarda valores de verdadeiro ou falso (true, false).

O QUE É LORA?

Agora que entendemos melhor a parte de software, vamos estudar uma ferramenta muito útil para o Arduino: a tecnologia LoRa. LoRa, que significa 'Long Range', é uma tecnologia de modulação de espectro espalhado. Em outras palavras, trata-se de uma tecnologia de comunicação por radiofrequência.



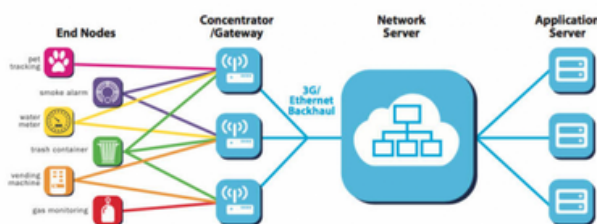
LoRa (Long Range) é uma tecnologia de rádio relativamente nova destinada à comunicação de pequenas quantidades de dados em distâncias maiores do que alcançáveis com Bluetooth ou Wi-Fi

LoRaWAN

O LoRa é, na verdade, a camada física e a tecnologia em si. Já o LoRaWAN é o protocolo que define a arquitetura de rede, ou seja, a camada lógica. O LoRaWAN pode ser visto como uma 'caixa de ferramentas' que utiliza a tecnologia LoRa, incluindo todos os recursos lógicos necessários, como frameworks, bibliotecas e outros componentes essenciais.

LORA

A arquitetura e o funcionamento das diferentes partes que a compõem podem ser um tema extenso, mas é possível fornecer uma breve visão geral da arquitetura, destacando seus nós e gateways:



Fonte: Profissionais TI

Cada nó é responsável por receber ou emitir informações 'em campo' realizando medições, ações, cálculos e coleta de dados. Esses nós possuem um módulo LoRa que envia as informações para o Gateway. Por sua vez, os Gateways transmitem esses dados para a internet, onde podem ser direcionados a uma 'aplicação final' ou seguir o caminho inverso. Nesse último caso, os Gateways recebem informações e as retransmitem de volta para os nós.

Quais as vantagens desse tipo de tecnologia?

1. Alta imunidade a ruídos;
2. Baixo consumo de energia;
3. Longo alcance, como já mencionado, podendo chegar a até 15km;
4. Capacidade para espalhar milhares de nós.

HARDWARE

SIGABEM

MAS, O QUE É *HARDWARE*?

Podemos chamar de hardware qualquer componente físico presente dentro ou fora de um aparelho tecnológico. Exemplos incluem celulares, televisores e, claro, videogames. Portanto, nosso equipamento também é classificado como um conjunto de hardware, pois é composto por:



TECLADO MATRICIAL

Uma maneira simples de entender este teclado é compará-lo ao teclado de uma maquininha de cartão de crédito, cujos botões estão organizados em colunas e linhas, formando uma matriz — daí o seu nome. Quando pressionamos qualquer botão, ele conecta a linha à coluna correspondente.

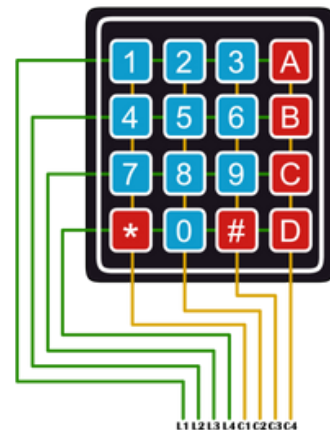
A pinagem é a seguinte:

Pino 1 (esquerda) - Primeira Linha (L1)

Pino 2 - Segunda Linha (L2)

Pino 5 - Primeira Coluna (C1)

Pino 8 - Quarta Coluna (C4)



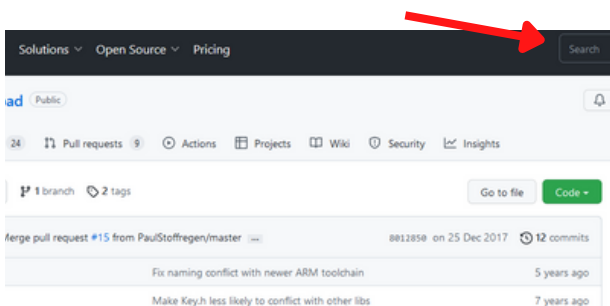
E quanto ao nosso teclado? Ele funciona da seguinte forma: na programação, cada linha de ônibus está associada a um botão do teclado (com 5 linhas dispostas nos botões de 1 a 5). O botão 'Hashtag' nos leva a uma tela onde podemos escolher a linha desejada. Assim, ao pressionar um botão de 1 a 5, o sistema reconhece que queremos acessar a linha correspondente a esse botão. Por exemplo, se pressionarmos o botão 1, as informações sobre a linha 202 serão exibidas e selecionadas, e em seguida, essas informações serão enviadas ao motorista.

PROGRAMAÇÃO

Para utilizarmos o teclado matricial, precisaremos baixar uma biblioteca específica para ele. Para o teclado em questão, obtivemos a biblioteca do GitHub, um site amplamente utilizado por programadores.

<https://github.com/Chris--A/Keypad>

Para fazer o download pelo link acima, basta acessar o site e procurar pelo botão 'Code':



Em seguida, clique em 'Download ZIP' e, depois, inclua a biblioteca no IDE Arduino, conforme explicado anteriormente.

Agora podemos 'importar' a biblioteca no IDE e utilizar suas funções. As principais funções dessa biblioteca estão disponíveis no seguinte link:

<https://playground.arduino.cc/Code/Keypad/#Functions>

No Arduino IDE, vamos começar criando algo semelhante a uma variável: um arranjo. Pense nisso como uma variável que pode armazenar mais de um valor. O código, portanto, deve ser assim:

```
#include <Keypad.h>
```

```
byte pinosLinhas[] = {3,2,5,0};
```

```
byte pinosColunas[] = {A0,A1,A2};
```

Agora, vamos criar um 'mapa' do teclado com suas respectivas teclas utilizando um recurso matemático chamado matriz, o que justifica o termo 'teclado matricial'. Essa etapa é essencial para que o computador identifique qual símbolo corresponde a cada tecla. Portanto, o código ficará assim:

```
char teclas[4][3] = {{ '1', '2', '3' },  
                     { '4', '5', '6' },  
                     { '7', '8', '9' },  
                     { '*', '0', '#' } };
```

Com todos os arranjos prontos, agora basta chamar a função que irá reconhecer essas três partes como um teclado:

```
Keypad teclado1 = Keypad( makeKeymap(teclas),  
                           pinosLinhas, pinosColunas, 4, 3);
```

Assim, o teclado já se tornou parte do nosso código. Agora, basta aplicar as funções que podem ser encontradas no link mencionado ao lado. Esse link também contém exemplos de aplicações e trechos de código que ajudam a entender melhor como a biblioteca funciona.

DISPLAY

Display é o termo usado para designar a pequena tela onde as informações do equipamento serão exibidas. Pense na tela do seu smartphone, mas sem o vidro protetor e a parte digitalizadora do touch, que permite a interação direta. Existem diversos tipos de displays, e o nosso é do tipo LCD (Liquid Crystal Display). O nome se refere ao fato de que ele é composto por uma matriz de cristais líquidos que são iluminados por uma luz de fundo.



eletrogate.com/display-lcd

Ao iniciar, a tela exibirá 'SigaBem - Apoio ao Embarque'. Ao pressionar a tecla 'Hashtag' no teclado, poderemos visualizar todas as linhas de transporte disponíveis, uma a uma. Ao apertar uma tecla, a linha correspondente à tecla pressionada aparecerá na tela, seguida das mensagens 'Enviando sua solicitação' e 'Solicitação recebida'.

PROGRAMAÇÃO

A biblioteca que utilizamos é a 'LiquidCrystalI2C', disponível no link a seguir:

<https://drive.google.com/file/d/1bSa5tyCN3MOFJj4zkT1RD-bnLCxr6tFv/view>

Após baixar, instalar e incluir a biblioteca no código, estaremos prontos para começar a programar:

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd (0x27, 16,2);
```

Na segunda função, o primeiro argumento entre parênteses representa o endereço do display, o segundo indica o número de colunas (neste caso, 16) e o terceiro, o número de linhas disponíveis (2).

```
lcd.init();
lcd.setBacklight(HIGH);
lcd.setCursor(0,0);
lcd.print ("Projeto SigaBem");
delay(500);
```

Em ordem, o processo consiste em inicializar o LCD, ativar a luz de fundo, definir a coluna e a linha (1º e 2º argumentos) e, por fim, escrever o que se deseja exibir no display.

AUTO-FALANTE

O alto-falante é o dispositivo responsável por anunciar as informações de forma auditiva, utilizando uma voz. Além de facilitar o uso do totem, ele também auxilia pessoas com deficiência visual que, de outra forma, teriam dificuldade em ler as informações exibidas no display. Assim, a função do alto-falante é reproduzir mensagens de áudio, que vão desde a introdução do equipamento e instruções de uso até a escolha e confirmação das linhas.

E O ALTO-FALANTE, É SÓ ISSO?



Seu funcionamento vai além de simplesmente reproduzir som. O volume do alto-falante pode ser parcialmente controlado por um dos componentes mencionados no início da nossa apostila. Adivinhou qual é? É o potenciômetro!

Além disso, temos outro componente que complementa o alto-falante, e, curiosamente, este é o próximo tópico que vamos abordar: o módulo SD.

PROGRAMAÇÃO

Primeiramente, é necessário baixar uma biblioteca chamada TMRpcm, disponível no link:

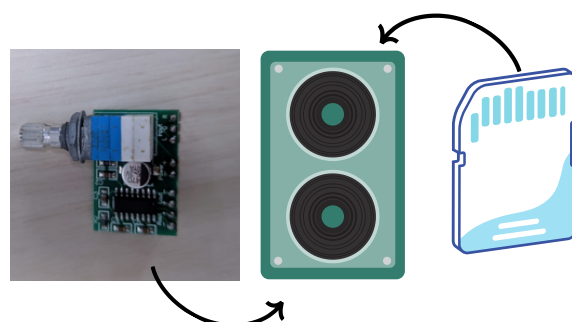
<https://github.com/TMRh20/TMRpcm>

Além dessa biblioteca, precisamos incluir mais duas que já estão disponíveis na IDE: SPI e SD. Também é importante definir o pino ao qual o módulo SD está conectado. Para isso, criamos uma constante com o valor do pino (diferente de uma variável, esse valor não pode ser alterado).

Além disso, definimos uma variável do tipo TMRpcm chamada 'som', tudo isso fora do void setup.

```
#include <SD.h>
#define SD_ChipSelectPin 4
#include <TMRpcm.h>
#include <SPI.h>
```

```
TMRpcm som;
```



MÓDULO MICRO SD

E qual é a função do módulo SD? De forma simples, ele é essencial para que o alto-falante possa transmitir as mensagens, já que armazena os arquivos de áudio reproduzidos quando o sistema está em operação. Com 2 GB de memória, ele oferece espaço suficiente para esse armazenamento. Podemos dizer, então, que o módulo SD é a 'voz' do nosso hardware. Os arquivos de áudio armazenados nele estão no formato '.wav'. Mas o que isso significa?



Os arquivos de áudio no formato WAV (Waveform Audio File Format) ocupam mais espaço de armazenamento, mas oferecem a máxima qualidade de som, reproduzindo fielmente o que foi gravado. Por esse motivo, é importante que o cartão SD tenha 2 GB de memória, garantindo espaço suficiente para armazenar esses arquivos sem comprometer a performance do sistema.

PROGRAMAÇÃO

Agora, daremos continuidade à programação iniciando o módulo SD, selecionando o pino para o áudio e verificando se o cartão de memória está corretamente inserido e pronto para inicialização.

```
void setup() {  
  
  SD.begin(SD_ChipSelectPin);  
  som.speakerPin = 9; //5,6,11 ou 46 para o Arduino Mega  
                    //9 para o Arduino Uno, Nano, etc  
  
  if (!SD.begin(SD_ChipSelectPin)) {  
    return;  
  }  
}
```

Agora que tudo está pronto, basta utilizar as funções disponíveis na Wiki mencionada anteriormente e garantir que os arquivos de áudio estejam salvos no cartão. Em nosso projeto, integramos essa funcionalidade ao display LCD, resultando no seguinte:

```
lcd.init();  
lcd.setBacklight(HIGH);  
lcd.setCursor(0,0);  
lcd.print("Projeto SigaBem");  
delay(500);  
som.play("siga.wav");  
delay(2500);  
lcd.clear();
```

Lembrando que o nome do áudio não deve exceder 8 caracteres.