CURSO DE INTRODUÇÃO À PROGRAMAÇÃO POR MEIO DA ROBÓTICA

PEDRO LEMOS DE ALMEIDA JÚNIOR
ORGANIZADOR

EDITORA
IFSertãoPE

INSTITUTO FEDERAL
Sertão Pernambucano

EDITORA
IFSertãoPE

Presidente da República
Luis Inácio Lula da Silva
Ministro da Educação
Camilo Sobreira de Santana
Secretário de Educação Profissional e Tecnológica
Marcelo Bregagnoli

INSTITUTO FEDERAL
Sertão Pernambucano

EDITORA
IFSertãoPE

Contato
Rua Aristarco Lopes, 240 - Centro
CEP: 56302-100 | Petrolina/PE – Brasil
E-mail: editora@ifsertaope.edu.br

ACADEMIA MAKER – CURSO DE INTRODUÇÃO À PROGRAMAÇÃO POR MEIO DA ROBÓTICA.

Organizador
Pedro Lemos de Almeida Júnior

Dados Internacionais de Catalogação na Publicação (CIP)

A168 Academia Maker: curso de introdução à programação por meio da robótica / Pedro Lemos de Almeida Junior (Org.). - Petrolina: IFSertãoPE, 2024.

PDF ; 35,9 MB ; 157p. il.

ISBN: 978-65-89380-39-9

1. Metodologias ativas de aprendizagem. 2. Programação 3. Robótica.

I. ALMEIDA JUNIOR, Pedro Lemos.

CDD 600

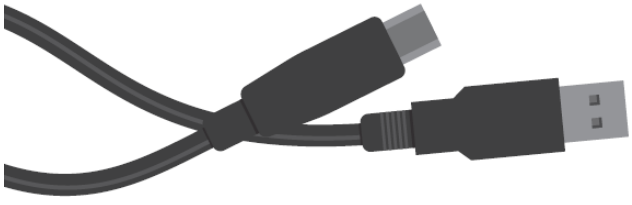
Ficha Catalográfica Elaborada pela Bibliotecária Ana Christina Bezerra CRB4-2311

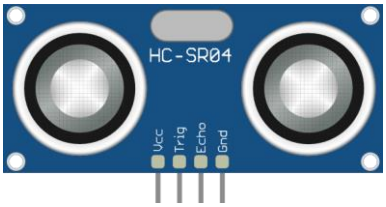
SUMÁRIO

ACADEMIA MAKER

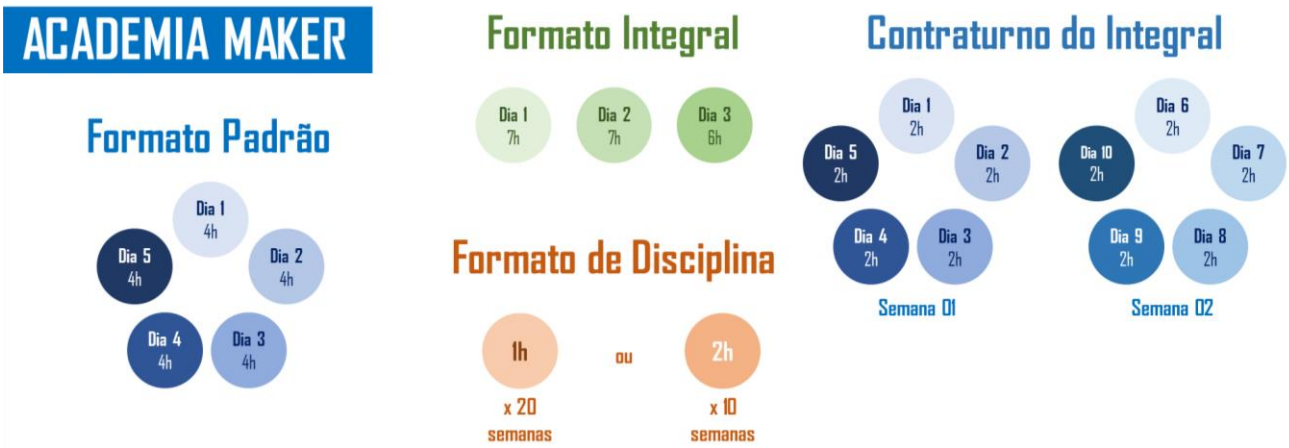
A ACADEMIA MAKER é fruto de um projeto de extensão aprovado junto aos editais 01/2021 do IFES, edital de apoio à iniciação tecnológica com foco no ensino de programação aplicada, e 04/2022 do IFSertãoPE, processo seletivo de programas e núcleos de extensão. No ano de 2022, o projeto em questão foi desenvolvido por 12 estudantes de cursos nível médio técnico e superior, que realizaram a formação de mais de 300 estudantes de nível fundamental da rede pública de ensino nas cidades de Salgueiro e Terra nova.

Esse projeto objetiva o desenvolvimento e aplicação de um curso de introdução à programação por meio da robótica. Para tanto, fez-se uso de metodologias de ensino ativas, principalmente a aprendizagem baseada em projetos, e da metodologia de introdução de conhecimento sob demanda, dessa forma, conceitos de eletrônica e programação são apresentados de acordo com as demandas dos alunos para a resolução dos projetos propostos. Por essa razão, na estruturação desse material você não irá se deparar com um capítulo inicial com conceitos de lógica de programação, por exemplo. Os conceitos que seriam trabalhados nesse capítulo são dissolvidos ao longo do curso, de modo que os estudantes possam trabalhar esses quando deles demandas e entender, na prática, seu princípio de funcionamento. O mesmo aplica-se aos conceitos de eletrônica e aos componentes eletrônicos, apesar desses serem apresentados, brevemente, no início desse material, uma abordagem mais profunda é realizada no primeiro projeto na qual aquele componente é utilizado.





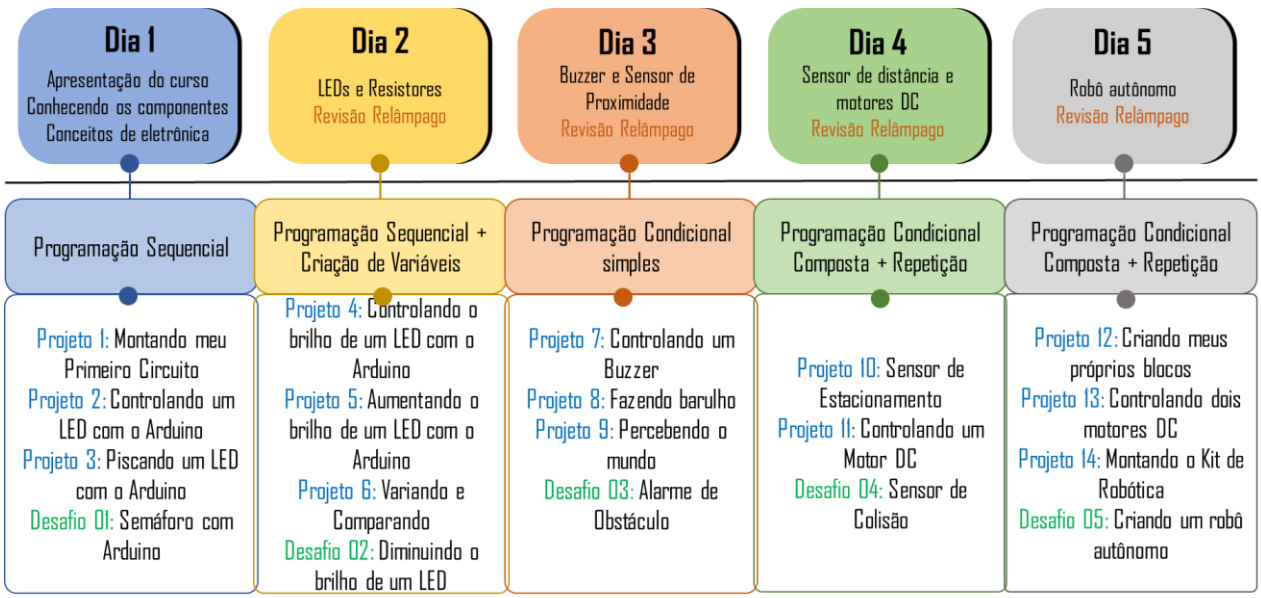
Idealmente, projeta-se o curso para ser realizado em 5 dias corridos, com quatro horas diárias, totalizando uma semana de curso intensivo que ocorre, geralmente, no horário regular desses alunos(as). Quando aplicado em escolas com ensino em horário integral, o curso pode ser aplicado em três dias corridos, utilizando o horário letivo dos alunos, ou em duas semanas, com duas horas diárias, geralmente ocorrendo no contraturno das aulas dos alunos, quando esses podem ficar além do seu horário habitual. Por fim, em casos específicos, o curso pode ser integrado em uma dada disciplina a ser executada ao longo de 10 ou 20 semanas, com uma carga horária semanal de uma ou duas horas aulas.



Esse material é utilizado como material didático e de referência para os estudantes e professores que aplicam esse curso em suas escolas. Ao longo desse material são apresentados 17 projetos que podem ser desenvolvidos a partir dos materiais listados no capítulo a seguir. Esse projetos tornam-se mais complexos em nível de abstração exigidos e no número de componentes eletrônicos empregados. Ao longo de cada capítulo apresentam-se os componentes, a complexidade de cada projeto, o passo a passo da montagem do circuito e o processo de criação do software de controle.

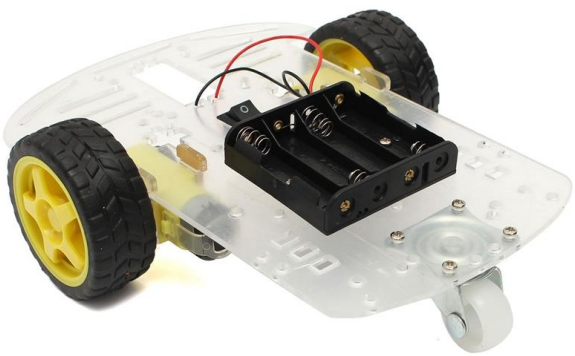
Em termos práticos, quando aplicados em sala, empregamos a apresentação desses como projetos guiados, nos quais são apresentados conceitos e eletrônica e programação, por fim os alunos são desafiados em um projeto mais complexo, que emprega os conceitos e componentes trabalhados ao longo do dia. Esse formato premia as equipes que conseguirem, de forma autônoma, realizar o desafio proposto em um intervalo de tempo determinado. Aquelas que não conseguem serão guiadas a resolvê-lo, de modo a compreender a resolução e refletir sobre as etapas que impediram seu sucesso na resolução no tempo proposto. . Essa abordagem envolve, além da aprendizagem baseada em projeto, técnicas de gamificação buscando um maior engajamento dos alunos. Vale ressaltar, porém, que nenhuma equipe deixa de ser pontuada ao fim do projeto ou deixa de realizá-lo.

Considerando a experiência da equipe no ano de 2022, distribuição dos projetos ocorreu como descrito abaixo.



A partir desses 16 projetos são elaborados os materiais para o curso, como slides e apostilas. De modo a facilitar o trabalho dos alunos que estão fazendo o curso, sempre que uma equipe conclui um projeto ela recebe a resolução desse, impressa, de modo a compor uma apostila que, ao fim do curso, contará com a resolução de todos os projetos e desafios. Essa estratégia facilita a resolução de projetos mais complexos utilizando os conceitos aprendidos nos projetos mais simples.

Esse curso toma como base um kit Arduino básico, com um chassis e dois motores DC, para o desenvolvimento dos projetos. Os projetos, bem como os desafios, foram propostos de acordo com os componentes eletrônicos disponíveis no kit utilizado, porém com aplicações em diversos projetos. Por exemplo, o sensor infravermelho que é, corriqueiramente, utilizado para detectar a linha guia para robôs seguidores de linhas, foi aplicado em um primeiro projeto para ilustrar o funcionamento de sensores e da entrada digital da placa Arduino. O sensor ultrassônico de distância foi aplicado em projetos para desenvolvimento de um sensor de estacionamento, juntamente com um *buzzer*, ou um sensor de colisão que freia um dado motor. Dessa forma, a aplicação desses componentes não se limitam apenas aos projetos aqui listados, porém esses projetos foram pensados seguindo a metodologia didática já descrita, por essa razão recomenda-se muito seu uso na sequência aqui apresentada.



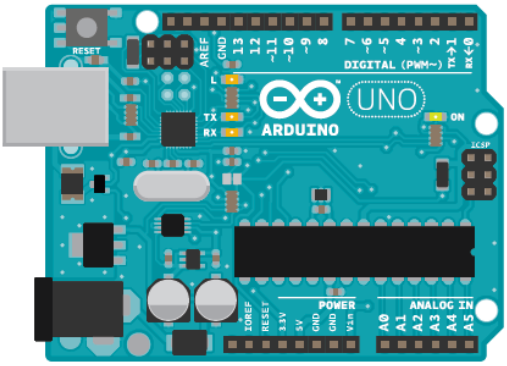
Como se trata de um curso que tem como principal objetivo o ensino de programação, empregou-se uma plataforma de programação gráfica, em blocos. Nesse caso, optou-se pela plataforma mBlock, que possui versão traduzida para o português, o que facilita a compreensão das funções e comandos de cada bloco, interface nativa com o Arduino, permitindo sua programação em tempo real ou por meio da gravação dos comandos na placa. Essa plataforma permite, ainda, aos estudantes relacionarem os componentes eletrônicos do hardware, com componentes gráficos do software, permitindo, por exemplo, controlar a posição de um dado personagem na tela por meio de um sensor de distância.

Como destacado anteriormente, o curso em questão utiliza um kit com um chassis de acrílico, porém, buscando garantir uma maior resistência mecânica e facilidade de montagem, os kits foram personalizados, utilizando peças plásticas oriundas de impressoras 3D. A personalização permite, além de uma maior robustez do kit quando comparamos as peças impressas aquelas equivalentes de acrílico, um processo de montagem mais simples, permitindo que os estudantes apenas encaixem os motores e os componentes eletrônicos, sem uso de parafusos e porcas, o que agiliza e simplifica o processo, otimizando o tempo para a resolução dos projetos e desafios proposto.

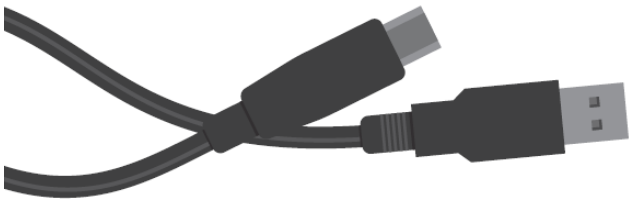
Todos materiais didáticos, como slides, apostilas e modelos 3D empregados na personalização dos kits de acrílico, podem ser acessados pelo QRcode.

Todos as imagens foram criadas pelo autor, utilizando os softwares Frintzing, mBlock e Blender.

COMPONENTES DO KIT DE ROBÓTICA

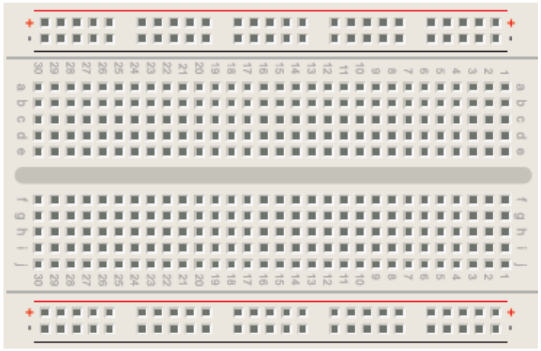


Arduino Uno: Placa microcontroladora que será o coração dos seus projetos. Trata-se de um pequeno computador, com ele você criará circuitos e programará como ele vai interagir com os outros componentes. (1 unidade)

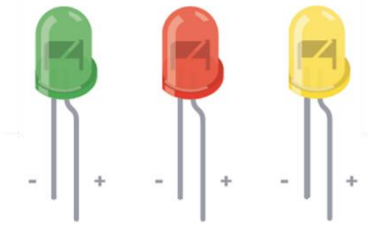


Cabo USB: Utilizado para a comunicação entre o Arduino e computador. Além disso, em vários exemplos o cabo USB é utilizado para alimentação elétrica dos circuitos. (1 unidade)

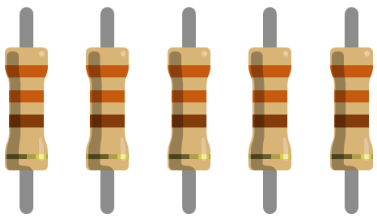
Cabos Jumper: Cabos destinados à conexão entre componentes. (30 unidades)



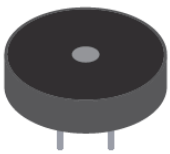
Protoboard ou Placa de Testes: Ou "matriz de contatos", é uma placa reutilizável para montagem de circuitos e conexão de componentes eletrônicos sem necessidade do uso de soldas. (1 unidade)



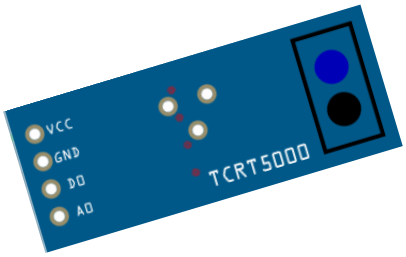
LEDS de 5mm (Diodos Emissores de Luz): Um tipo de diodo que emite luz com a passagem de uma corrente elétrica, muito utilizado para sinalização visual. (15 unidade, 5 de cada cor)



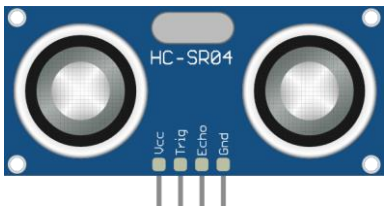
Resistores: Limitam a corrente de um determinado ponto do circuito. (20 unidades)



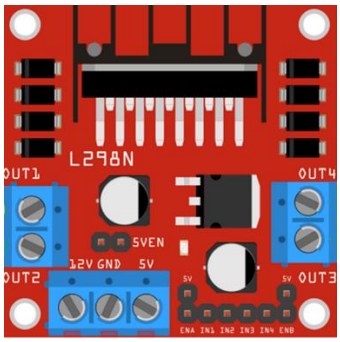
Buzzer: Componente elétrico para emissão de sons e melodias, possibilitando o controle dos tons gerados. (1 unidade)



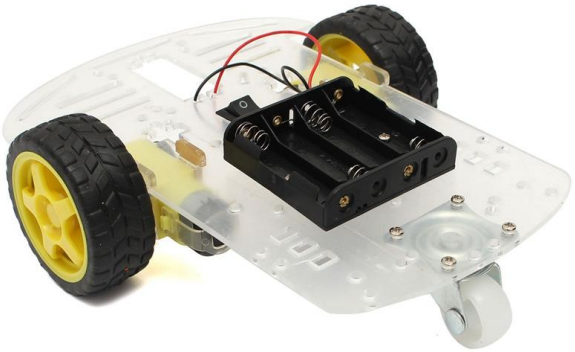
Sensor de Obstáculo Infravermelho: Sensor utilizado para a detecção, por infravermelho, de obstáculos. Seu circuito possui um emissor e um receptor: diante de algum obstáculo no ângulo e distância ajustados, o sinal IR é refletido e o sensor detecta o obstáculo. (2 unidades)



Sensor Ultrassônico de Distância (HCSR04): Sensor que usa sinal ultrassônico para identificar a distância, entre 2 cm e 4 m, até um objeto.(1 und.)



Módulo Ponte H (L298N): Placa utilizada para o acionamento e controle da velocidade de até 2 motores. (1 unidade)



Kit Robô para Arduino (Chassi 2WD): Destinado à montagem de projetos, como robô seguidor de linha, é composto por chassi, rodas conectadas a motores DC, roda boba e compartimento para pilhas AA. (1 unidade)

Observe que todos os componentes do kit de robótica estão em caixa plástica organizadora. Isso facilitará a guarda e manejo dos componentes. Por isso, tenha sempre atenção, quando receber o kit para desenvolver os projetos, em como os componentes estão guardados para devolvê-los ao lugar quando a aula encerrar. Isso manterá a ordem tanto para as próximas turmas quanto para que você, ao receber novamente os kits para o desenvolvimento dos projetos, encontre facilmente os componentes que precisa.

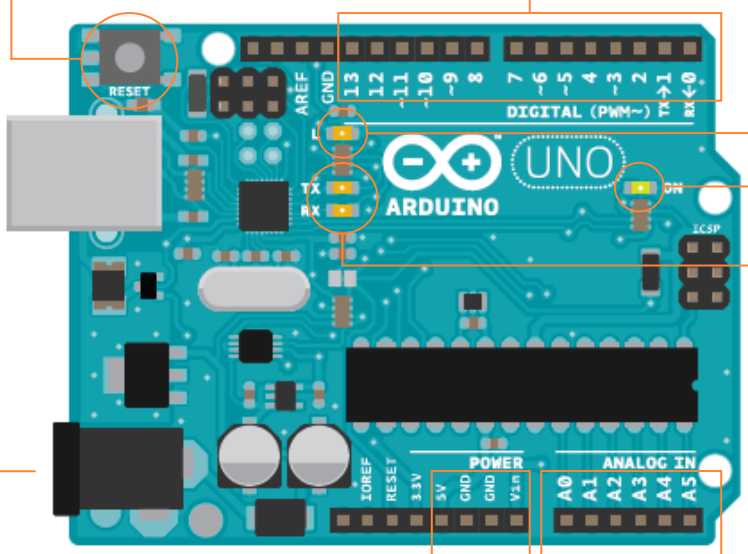
PLACA ARDUINO UNO

O Arduino é uma plataforma de prototipagem eletrônica que contempla, tanto o hardware, que são a placas eletrônicas, como também um ambiente integrado de desenvolvimento (IDE), que é, basicamente, onde escrevemos os comandos para controlar nossa placa. Existem várias placas Arduino, nesse kit vamos trabalhar com a placa Arduino Uno, que funciona como um minicomputador capaz de realizar tarefas, processar e armazenar informações.

Conector USB
Utilizado como fonte de energia à placa, mas tem seu principal uso na interface para programação da placa.

Botão de Reset
Botão que permite resetar a placa, reiniciando a execução do código.

Pinos Digitais
São pinos para entrada e saída de dados. Funcionam apenas nos estados "alto", +5V, e "baixo", 0V (terra ou GND). A placa Arduino Uno possui 14 pinos digitais, desses 6 são pinos PWM (destacados pelo símbolo ~).



LEDs do pino 13
LED conectado diretamente ao pino 13 da placa.

LED de alimentação
LED que indica quando a placa está energizada.

LEDs RxTx
Esses dois LEDs da placa indicam a comunicação entre o Arduino e o computador. É esperado que, ao enviar um código esses LEDs pisquem rapidamente.

Pinos Analógicos
São pinos para entrada de dados analógicos, ou seja, que variam entre 0 e 5V. A placa Arduino Uno possui 6 pinos analógicos, geralmente utilizados para leitura de dados obtidos por sensores.

Pinos de alimentação
Pinos que fornecem 5V e 0V (GND) de saída. Esses pinos são utilizados para alimentação elétrica dos seus circuitos. O pino Vin é um pino que, junto com o GND, pode ser utilizado para alimentação do Arduino.

Conector de alimentação
Local onde energizamos o Arduino, quando ele não está ligado na porta USB, com tensões entre 7-12V.

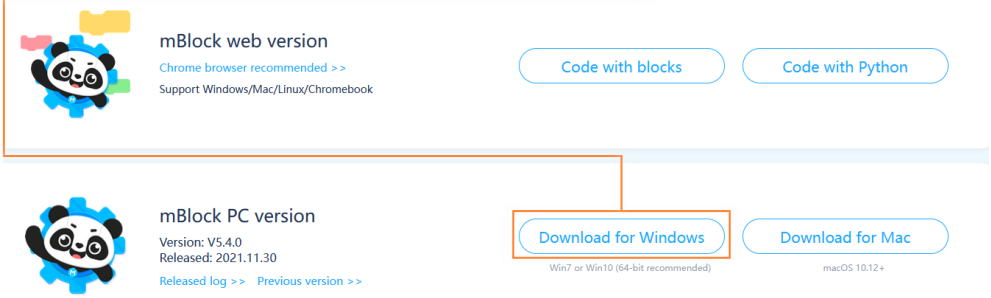
MBLOCK

O mBlock é uma plataforma que suporta linguagens de programação gráfica e textual, permitindo, assim, programar arrastando e soltando blocos de construção, semelhante a outra plataforma muito utilizada para essa finalidade, o Scratch. Utilizaremos o mBlock na versão offline, que pode ser instalado como descrito abaixo.

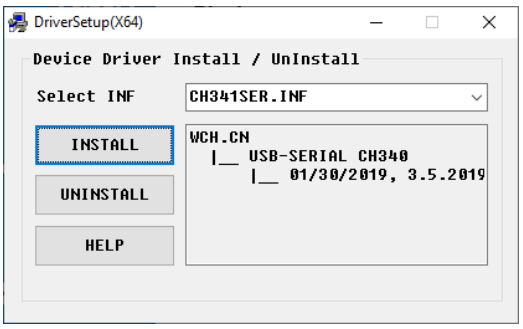
INSTALANDO NO WINDOWS

Acesse o site
http://www.mblock.cc/en-us/download

1. Escolha a opção mBlock PC version, Download for Windows.



2. Após do download, inicie o instalador. É possível que você tenha que instalar um *driver*, clique em *install*, como na imagem abaixo. Essa etapa dura alguns minutos.



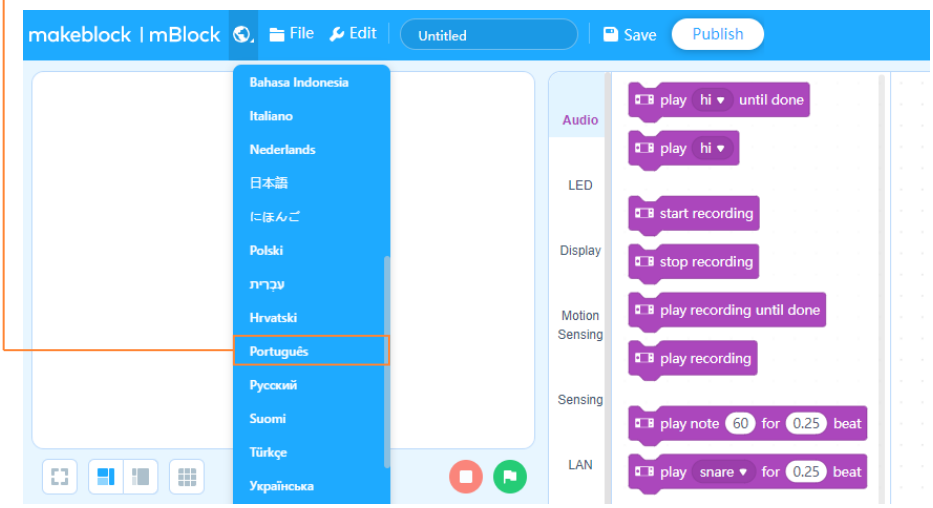
3. Após essas etapas o mBlock já está instalado e pronto para uso.

CONFIGURANDO O MBLOCK

Agora que você já instalou o mBlock no seu computador, vamos configurar a plataforma para facilitar o nosso trabalho.

1

Caso o idioma esteja completamente em inglês, vamos mudar clicando no primeiro ícone no menu superior, semelhante à um mapa-múndi e escolher o idioma português.

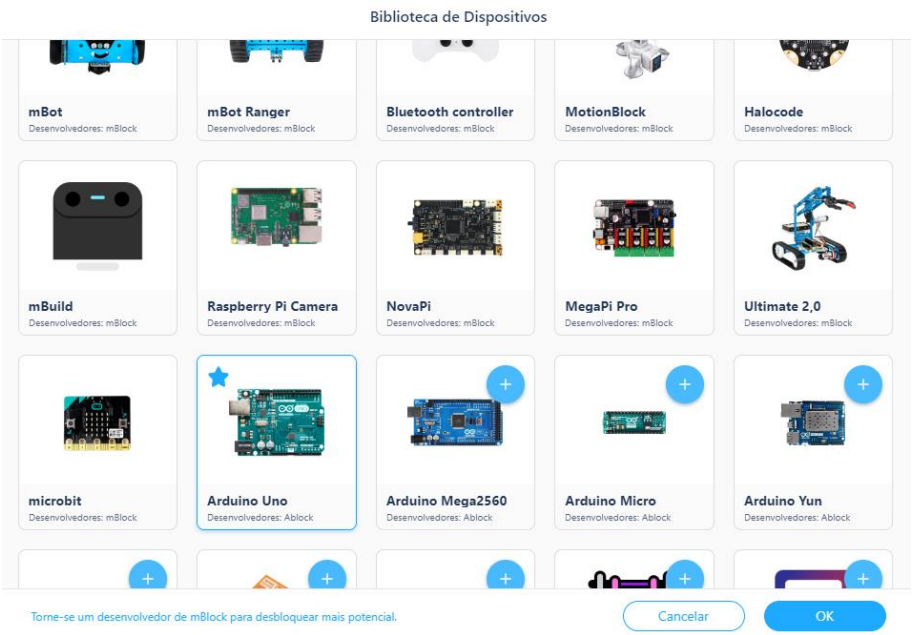


2

Em seguida, na aba dispositivos, clique em adicionar (+), em seguida procure a placa Arduino Uno, na Biblioteca de Dispositivos. Importante, é necessário ter acesso a internet nessa etapa. Clique no ícone (+) para iniciar o download.

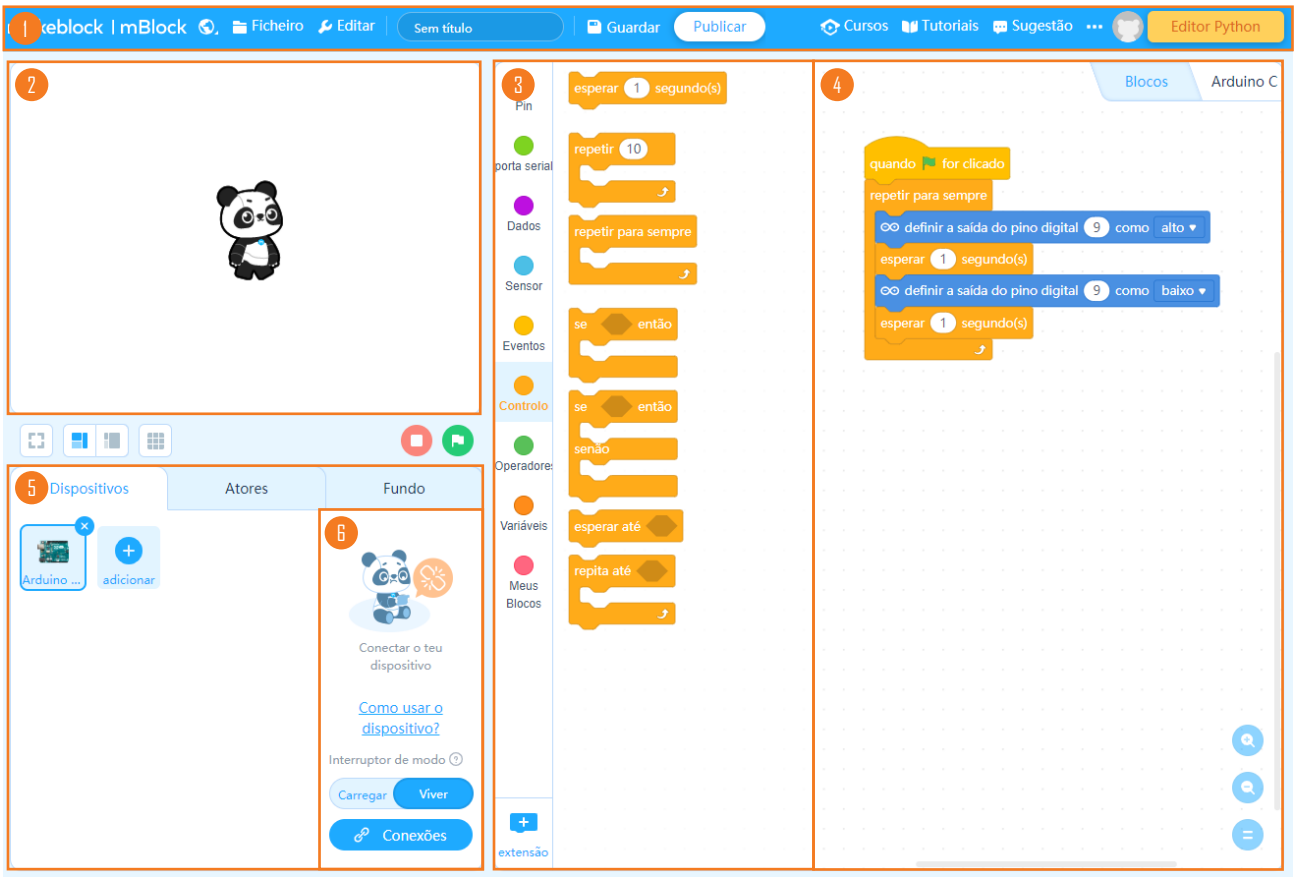
3

Por fim, selecione a placa Arduino Uno e clique no botão "ok".



CONHECENDO O MBLOCK

Agora que você já instalou e configurou o mBlock no seu computador, vamos conhecer um pouco sobre essa plataforma. Para facilitar vamos separar essa a tela em 7 partes:



1. No menu superior temos opções como mudar o idioma, carregar um arquivo salvo ou salvar o nosso arquivo, acessar cursos e tutoriais,

2. O cenário ou palco pode ser utilizado para observar as ações realizadas pelo Panda. Essa janela também é útil para acompanhar o valor de variáveis e a comunicação serial.

3. O conjunto de comandos é uma área onde estão localizados os comandos, classificados por cor e em categorias de acordo com as suas funções, além da aba extensão.

4. A área de programação é o espaço para onde arrastamos os blocos com os comandos e criamos a nossa programação. Nessa área também é possível programar na linguagem Arduino C, utilizando linhas de comando.

5. As abas de Dispositivos, Atores e Fundo, são importantes para a seleção dos elementos que irão compor a programação, os dispositivos para o caso de programação de placas como o Arduino e kits robótica, ou a criação de jogos e animações com atores e fundos.

6. Dentro da aba, dispositivos, temos a tela para conexão com os dispositivos.

CONECTANDO O ARDUINO

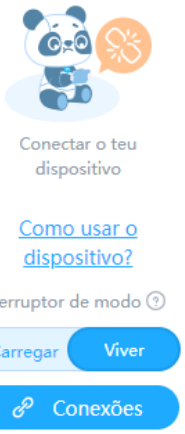
Abra o mBlock, já com o Arduino Uno como dispositivo selecionado, conforme imagem da página anterior.

1

Conecte a placa Arduino Uno ao computador, utilizando o cabo USB. É esperado que o LED de alimentação acenda e o LED do pino 13 pisque.

2

Nesse momento temos que definir como vamos programar o Arduino Uno. O mBlock possui duas opções, o modo "**carregar**" e o modo "**viver**".

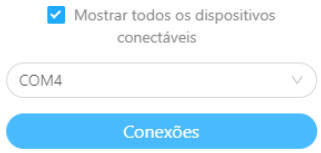


No modo carregar nós gravamos um programa na memória do Arduino e esse programa sempre será executado, basta energizarmos a placa, mesmo que ela não esteja conectada ao computados, esteja conectada à uma bateria, por exemplo. Nesse modo, os comandos só serão executados após serem carregados na memória da placa e qualquer modificação nos comandos, esses deverão ser carregados novamente.

Já o modo "viver", as mudanças que são realizadas na área de programação serão interpretadas imediatamente pela placa e já serão implementadas. Por outro lado, o código criado funcionará apenas quando conectado ao computador com o mBlock aberto.

Nesse primeiro momento, vamos trabalhar com a conexão no modo "**viver**", sendo assim, selecione o modo "viver" e em seguida no botão "Conexões".

3

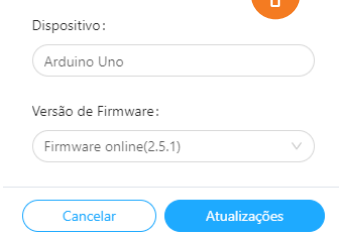


Na janela que se abre selecione a caixa "Mostrar todos os dispositivos conectáveis" e em seguida escolha a porta na qual o Arduino Uno está conectado. Geralmente são as portas com numeração COM3 acima (COM4, COM5, COM6 etc.). Clique em "Conexões".

4

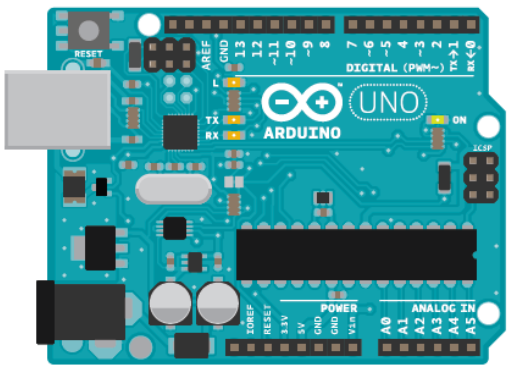
Caso seja a primeira vez que a placa é conectada nesse modo a comunicação ainda não funcionará, aguarde alguns segundos até que apareça o botão atualizar, logo abaixo do botão "Desconectar". Clique em "Atualizar", em seguida clique em "Atualizar Firmware".

5

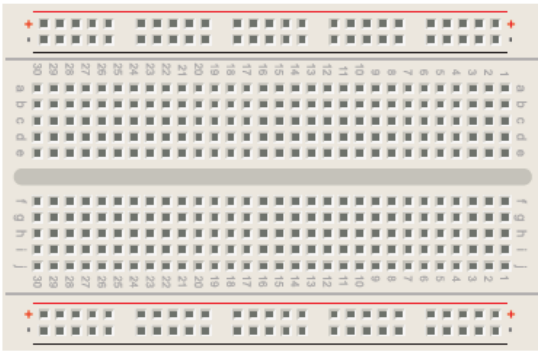


Na janela que se abre, clique em "Atualizações", confira o dispositivo, aguarde alguns segundos até que o firmware seja atualizado, durante esse processo não desconecte o Arduino ou feche o mBlock, para não correr o risco de errar na gravação do firmware, o que pode fazer com que a placa pare de funcionar. É normal que, ao término da gravação do firmware, o mBlock solicite que o Arduino seja conectado mais uma vez.

01. LIGANDO UM LED COM O ARDUINO

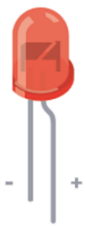


Arduino + Cabo USB (x1)



Placa de testes (x1)

Resistor (x1)



LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

COMPONENTES

CONHECENDO MEUS COMPONENTES

NESSE PROJETO VOCÊ VAI CONSTRUIR UM CIRCUITO SIMPLES COM UM LED E UM RESISTOR, ALÉM DO ARDUINO, PARA CONHECER MELHOR ESSAS FERRAMENTAS.

> Descobertas: Teoria básica da eletricidade, como funciona a placa de testes (protoboard) e como ligar os componentes em série e em paralelo.

> Tempo: **45 MINUTOS**
>
> Dificuldade: 

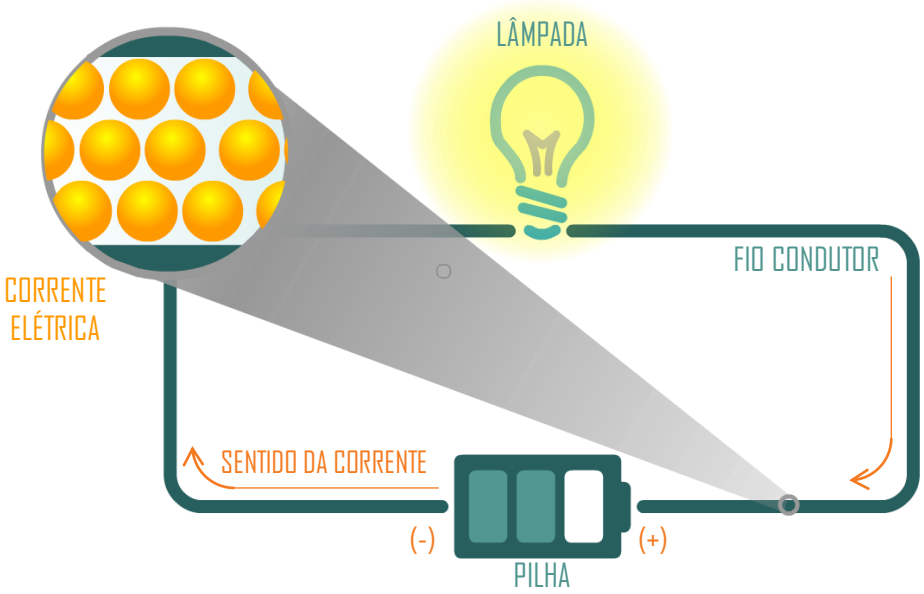
Ao longo de todo esse curso você será desafiado a realizar uma série de projetos que envolverão, em sua maioria, a montagem de circuitos elétricos e a programação de comandos para controlar os componentes desses circuitos. Dessa forma, vamos começar nossos estudos entendendo o que são circuitos elétricos e quais as grandezas envolvidas na montagem desses circuitos.

É importante entendermos que os nossos circuitos serão alimentados pela eletricidade, mas o que é eletricidade? É normal vir a nossa cabeça respostas como: "eletricidade é a energia que acende a luz". Essa resposta não está errada, muito pelo contrário, a energia elétrica, ou eletricidade, acende a luz da sua casa, acende a tela do seu celular, faz com que o notebook funcione, permite a TV receber sinal e projetar na tela para que possamos assistir. Nos nossos projetos vamos utilizar essa energia para controlar diversos componentes, transformar essa energia em luz, ao acender LEDs, em movimento, ao fazer motores girarem, em som, ao acionar um Buzzer.

Mas, até agora, não respondemos a pergunta, o que é eletricidade? Podemos dizer que eletricidade é um tipo de energia, como calor ou luz. Essa energia flui em condutores, como os metais. Você pode converter essa energia em outras formas de energia, de acordo com o seu interesse. Pode utilizar a eletricidade para acender uma luz, convertendo energia elétrica em luz, energia luminosa, ou fazer com que a eletricidade produza algum som, em uma caixa de som, para citar alguns exemplos. Por outro lado, para utilizar a eletricidade de forma correta, temos que conhecer algumas grandezas envolvidas no uso dessa, começando pela corrente elétrica.

CORRENTE ELÉTRICA

Corrente elétrica é o deslocamento de partículas minúsculas, chamados de elétrons, ao longo de algum material condutor. Lembre-se que todos os materiais têm elétrons, já que os elétrons são parte dos átomos e é dos átomos que são feitas todas as coisas. Esse material, você, o seu celular, todas as coisas são feitas de átomos, ou seja, todas as coisas possuem elétrons.



CONDUTORES

Em alguns materiais, como os metais, os elétrons podem ser mover mais facilmente com o fornecimento de energia elétrica. Essas substâncias nas quais os elétrons se movam facilmente tem o nome de condutores.

PILHAS E BATERIAS

As pilhas e baterias são fontes de alimentação elétrica para o circuito com dois polos, um com carga positiva e outro negativa. Na prática temos um polo com elevado potencial energético, que geralmente nos referimos como o polo positivo (+), e outro com baixo potencial energético, o qual nos referimos como polo negativo (-) ou terra (GND). Os elétrons são repelidos pelo polo negativo e atraídos pelo polo positivo, isso faz com que se movam através do fio e formem a corrente elétrica.

TENSÃO

A força que "empurra" esses elétrons nos condutores é chamada de força eletromotriz ou tensão. Essa força é medida em volts.

CIRCUITO ELÉTRICO

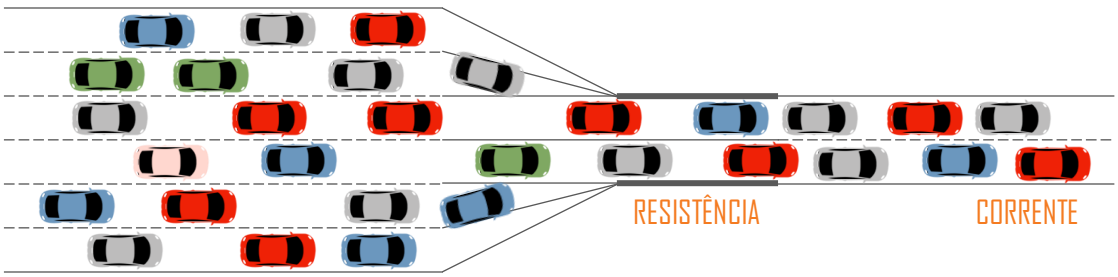
Um circuito é o caminho pelo qual a corrente elétrica pode passar. Alguns elementos são obrigatórios nos circuitos elétricos, como a fonte de alimentação e algum material condutor. Além disso, os circuitos elétricos devem ser fechados, para que os elétrons circulem ao longo do circuito.

Outro ponto que deve ser observado é que o circuito sempre deve ter algum componente que consuma corrente elétrica, caso contrário você estaria ligando o polo positivo no polo negativo da sua bateria o que resultaria em um curto-circuito, onde os elétrons passariam rapidamente de um polo para o outro, o que pode causar aquecimento dos fios e da bateria, queima de componentes eletrônicos, como a placa Arduino, até pequenos incêndios. SEMPRE CONFIRA SEU CIRCUITOS PARA EVITAR A OCORRÊNCIA DE CURTOS.

RESISTÊNCIA

Corrente e tensão são dois dos principais elementos de um circuito elétrico, o terceiro é a resistência. A resistência é a capacidade de alguns materiais de restringirem a passagem de elétrons, e assim reduzirem a corrente elétrica.

Podemos compreender corrente, tensão e resistência imaginando uma estrada muito movimentada. Caso em um dos pontos dessa via exista um pedágio, ou uma ponte, o número de faixas dessa via reduz e o número de carros que se deslocam nessa via também será reduzido. Podemos imaginar esses carros como elétrons, a via como o material condutor, o número de carros antes da ponte como a tensão e o número de carros após a ponte como a corrente elétrica.

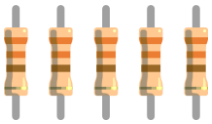


LEI DE OHM

Como descrito anteriormente, a corrente, a tensão e a resistência se relacionam, sendo possível calcular um, conhecendo os outros dois segundo a Lei de Ohm.

Tensão = Corrente x Resistência

RESISTORES

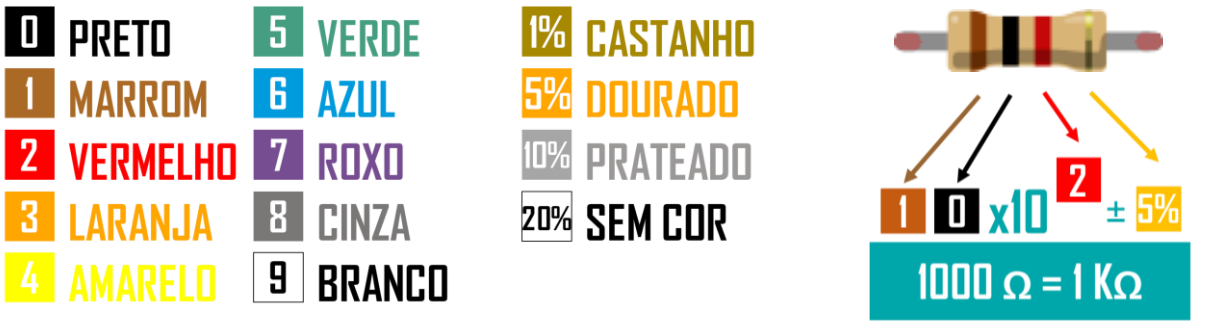


Todos os componentes apresentam um determinado grau de resistência à corrente elétrica, porém existem alguns componentes chamados de resistores que são feitos especialmente para controlar a corrente, e a tensão, de um outro componentes, ou para reduzir a corrente de um circuito e evitar que outro componente seja queimado.

O valor da resistência, medida em ohms (Ω), pode variar de alguns poucos ohms até milhões de ohms. Por outro lado, devido ao seu tamanho, é inviável imprimir nos resistores os valores de suas resistências. Por essa razão se criou um código de cores de acordo com a listras coloridas pintadas no corpo dos resistores.

CÓDIGO DE CORES

No código de cores, as três listras mais próximas umas das outras são as que determinam a resistência. A última está relacionada com a variação da medida. As duas cores das primeiras listras, no exemplo abaixo Marrom e Preto, estão relacionadas com os valores 1 e 0, respectivamente. A terceira lista determina a escala, ou, em outras palavras, a quantidade de zeros que deve ser adicionada no cálculo da resistência. Dessa forma temos 1, da primeira listra, 0, da segunda listra, mais dois zeros da terceira listra, o valor da resistência será 1000 Ω, ou 1 KΩ, o K significa 1000.



AGORA É A SUA VEZ

Utilizando o código de cores, calcule o valor da resistência dos resistores que você tem disponível no seu kit e que serão utilizados nos nossos projetos.

Cores: Laranja Laranja Marrom
3 3 1

Resistência: 33 x 10 = 330 Ω

Resposta da atividade

LEDs

Um LED, ou diodo emissor de luz, é um componente que converte energia elétrica em Luz. LEDs são componentes polarizados, ou seja, só permitem a passagem de corrente elétrica em um único sentido. Os polos dos LEDs podem ser identificados pelas suas pernas, geralmente, a perna maior é o polo positivo e a perna menor o polo negativo. Quando uma voltagem é aplicada no polo positivo e o polo negativo é conectado ao terra, GND, o LED emitirá luz. Pode-se observar também o interior do LED, o condutor menor corresponde ao lado positivos e o maior ao lado negativo.

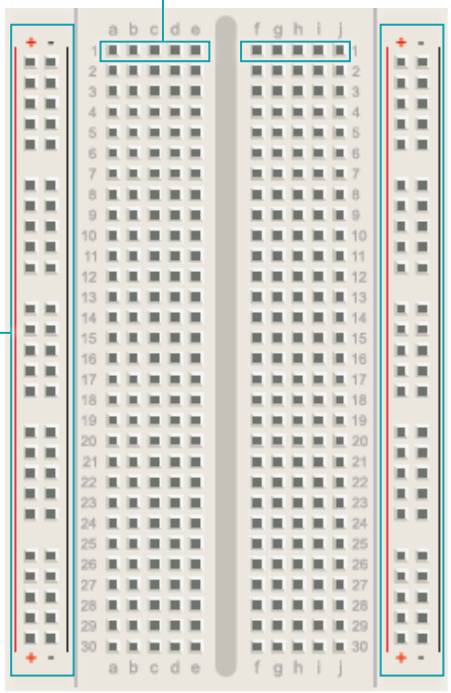
Ao utilizar a placa Arduino para alimentar o LED é importante, sempre, utilizar um resistor para reduzir a corrente fornecida. Ao ligar o LED diretamente ao Arduino, a corrente fará com que o LED "queime", logo a corrente fornecida é superior a suportada pelo componente.

PLACA DE TESTES

A placa de testes, também chamada de matriz de contatos ou protoboard, é o primeiro local onde você monta seus circuitos. Essa placa tem como principal finalidade permitir a montagem e desmontagem de circuitos sem a necessidade de soldar os componentes.

Os cinco furos na linha horizontal são conectados entre si por uma fita de metal condutor no interior da placa. Porém são separados dos outros cinco furos abaixo e dos laterais.

Os furos das colunas verticais também são conectados entre si. Porém são separados dos outros furos laterais. Essas duas filas de furos são, geralmente, utilizadas para conectar os polos positivos e negativos do circuito.



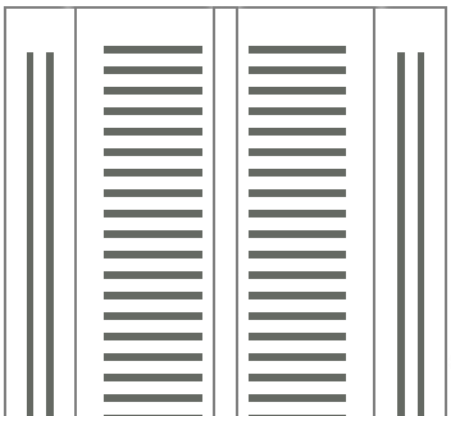
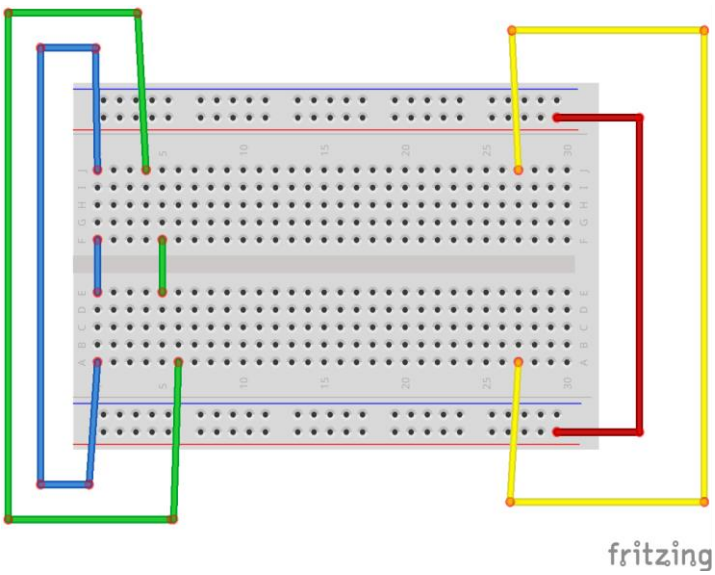


Ilustração das placas condutoras presentes no interior da placa de teste.

JUMPERS

Jumpers são cabos condutores responsáveis por realizar as conexões entre os componentes dos circuitos eletrônicos. São utilizados juntamente com as placas de teste na montagem dos circuitos. Repare que no exemplo abaixo os jumpers azuis estão conectados em um circuito fechado, já que os 5 furos nas linhas horizontais estão conectados entre si. Já os jumpers verdes não estão em circuito, já que estão em linhas horizontais diferentes. Assim como os jumpers amarelos, que não estão em circuito fechado já que no centro da placa os furos não estão conectados. O jumper vermelho está conectado de tal forma que todos os furos das colunas verticais, nas quais ele está conectado, de alimentação estão conectados entre si.



MONTANDO O CIRCUITO

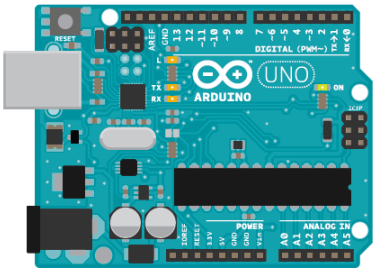
Antes de iniciar a montagem do circuito de qualquer projeto é importante seguir os passos abaixo:

1. Separar os componentes necessários;
2. Garantir que a alimentação elétrica do circuito está desligada;
3. Montar o circuito na placa de testes;
4. Conferir o circuito, todas as conexões, antes de ligar a alimentação;
5. Testar o circuito montado.

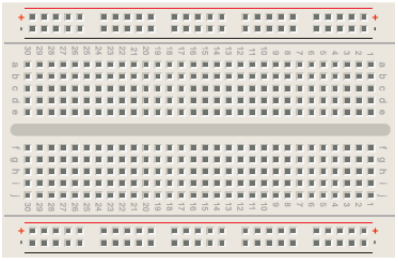
Caso não funcione, desligue novamente a alimentação elétrica e confira todo o circuito, ponto a ponto, de acordo com a ilustração da atividade.

MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

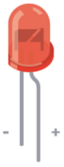


Arduino + Cabo USB (x1)



Placa de testes (x1)

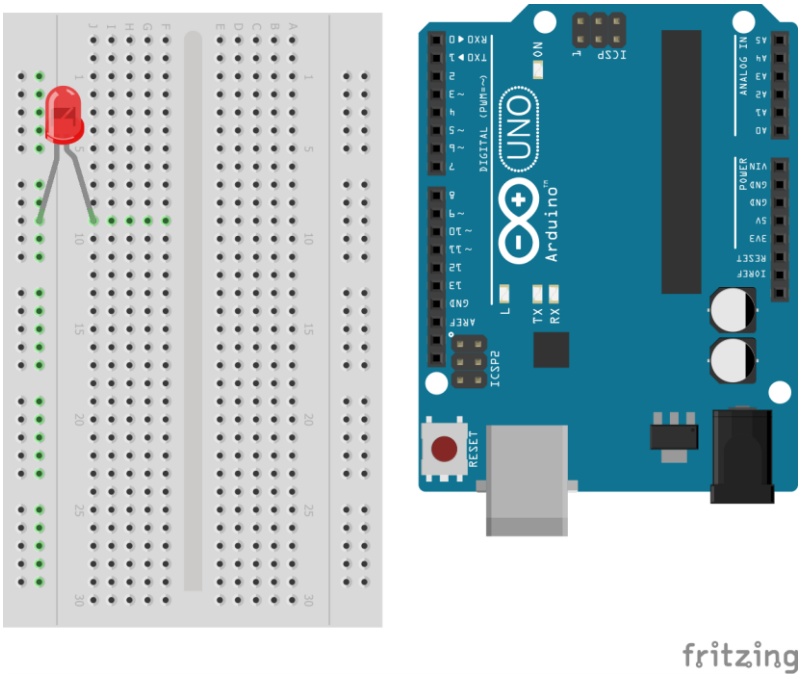
Resistor (x1)



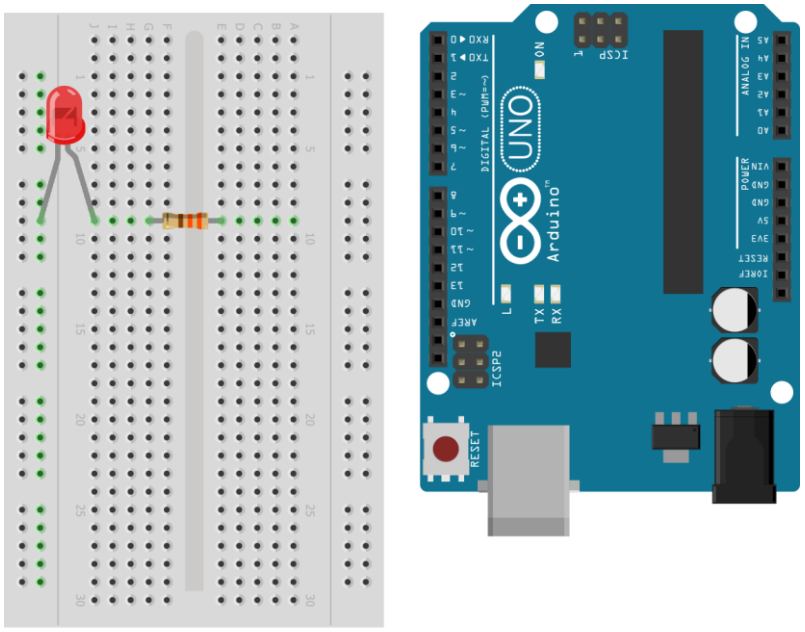
LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

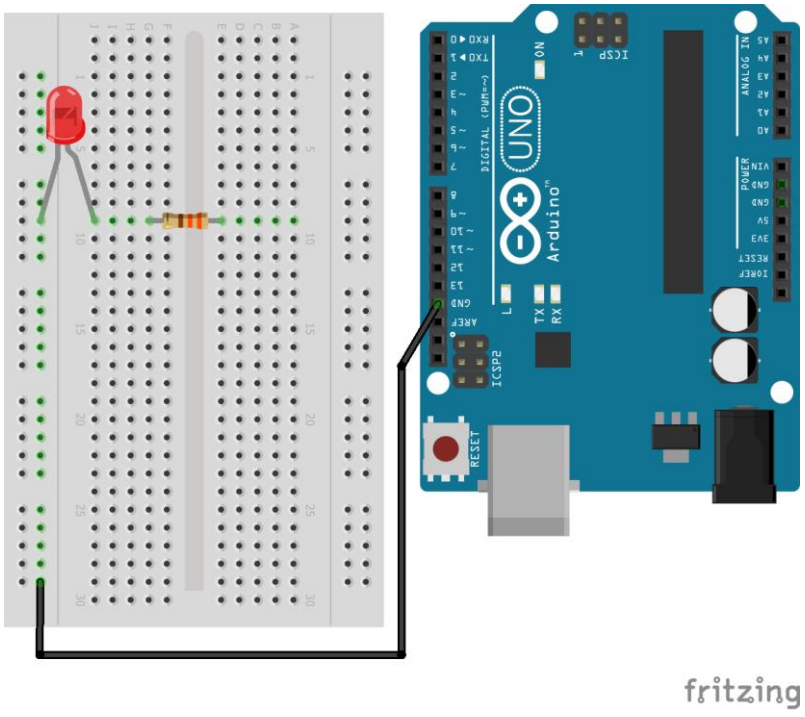
Etapa 02: Conecte o LED vermelho na protoboard com a perna maior, perna positiva, na linha horizontal e a perna menor, perna negativa, na coluna vertical.



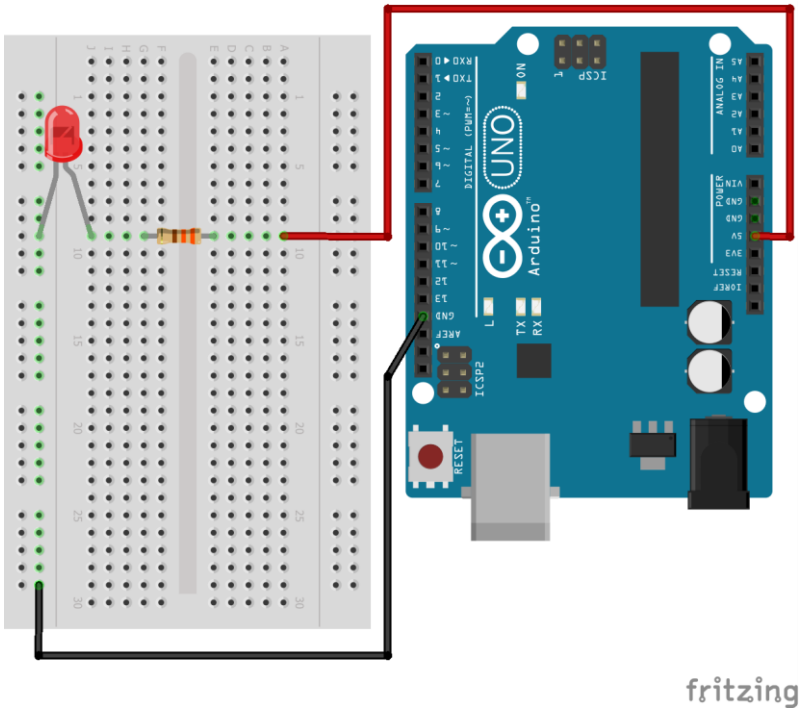
Etapa 03: Como você já sabe, para utilizar um LED com um Arduino deve-se utilizar um resistor para reduzir a corrente, sendo assim, insira um resistor em série com a perna maior do LED. Resistores não possuem polaridade, ou seja, não importa qual perna do resistor será conectada.



Etapa 04: Nessa etapa vamos começar a fechar o nosso circuito, vamos iniciar com a perna negativa do LED, a coluna na qual ela está ligada será conectada, utilizando um **jumper preto**, com o furo GND da placa Arduino, que significa terra, ou seja, ausência de tensão.



Etapa 05: Vamos agora fechar o circuito fornecendo tensão positiva para o LED, para tanto conectamos os pinos nos quais o resistor está ligado com o furo 5V da placa Arduino, utilizando um jumper vermelho.



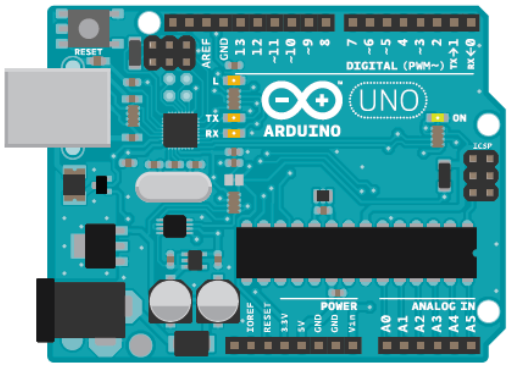
Etapa 06: Antes de alimentarmos o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos analisar o circuito, considerando que a corrente elétrica sai dos 5V do Arduino, ela vai passar inicialmente pelo jumper vermelho, em seguida vai para a linha horizontal da placa de teste, passa pelo resistor, pela linha em série com o fio maior, positivo, do LED, e por fim, passa pela coluna horizontal da placa de teste e pelo **jumper preto**, voltando para o Arduino, fechando assim o circuito.

Etapa 07: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. É esperado que, da forma que o circuito está montado, o LED acenda.

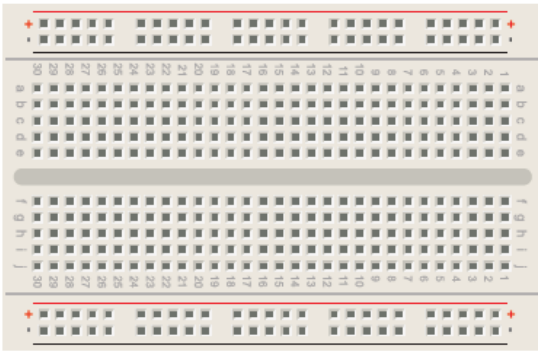
O LED acendeu? Parabéns, você acaba de montar o seu primeiro circuito elétrico.

O LED não acendeu? Confira novamente todas as ligações, pode ser apenas um pino conectado no furo errado.

02. CONTROLANDO UM LED COM O ARDUINO

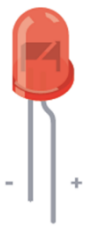


Arduino + Cabo USB (x1)



Placa de testes (x1)

Resistor (x1)



LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

COMPONENTES

ASSUMINDO O CONTROLE

NO PROJETO ANTERIOR VOCÊ CONSEGUIU LIGAR UM LED, AGORA É HORA DE ASSUMIR O CONTROLE.

Descobertas: Portas digitais, criando seu primeiro programa.

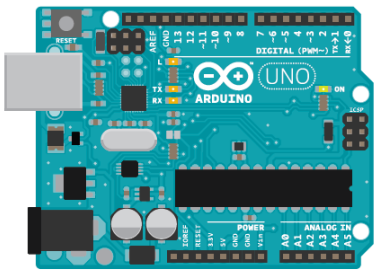
Tempo: **30 MINUTOS**

Dificuldade:

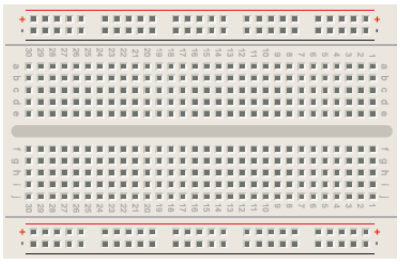
Agora que você já conhece alguns conceitos básicos de eletrônica e já montou o seu primeiro circuito, está na hora de controlar as coisas com o Arduino. Para tanto, vamos criar nosso primeiro programa para controlar um LED conectado à porta 6 do Arduino Uno.

MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

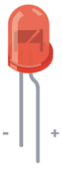


Arduino + Cabo USB (x1)



Placa de testes (x1)

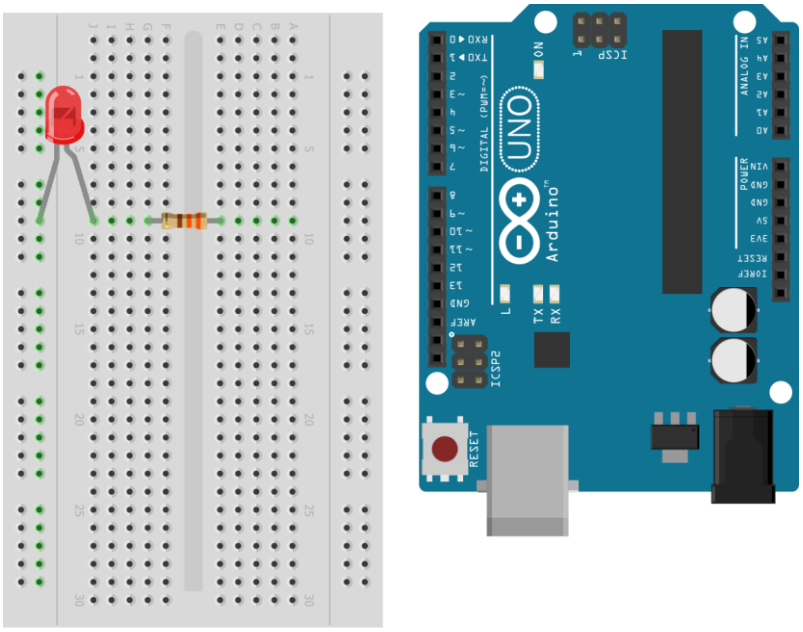
Resistor (x1)



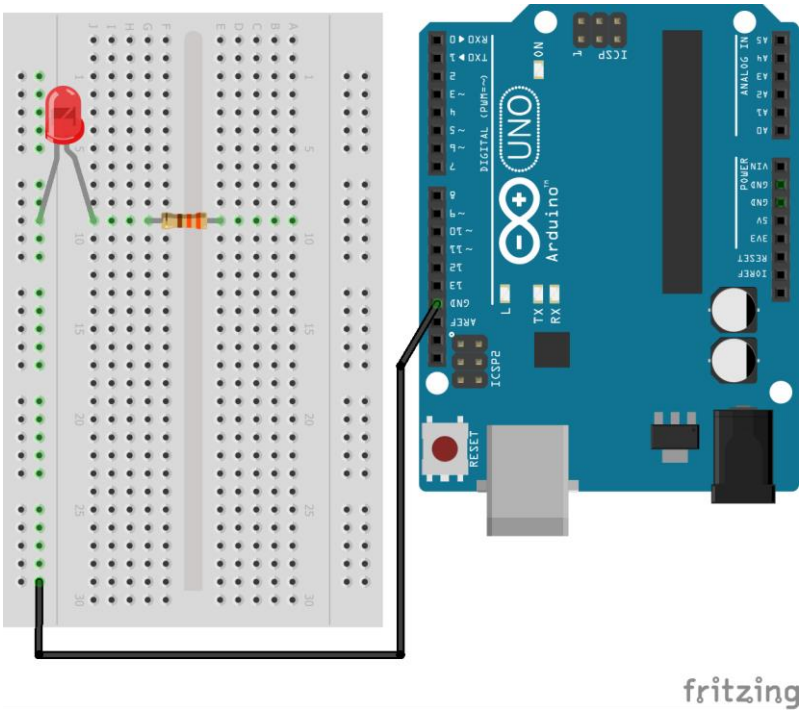
LED vermelho (x1)

Jumpers:
Vermelho (x1)
Preto (x1)

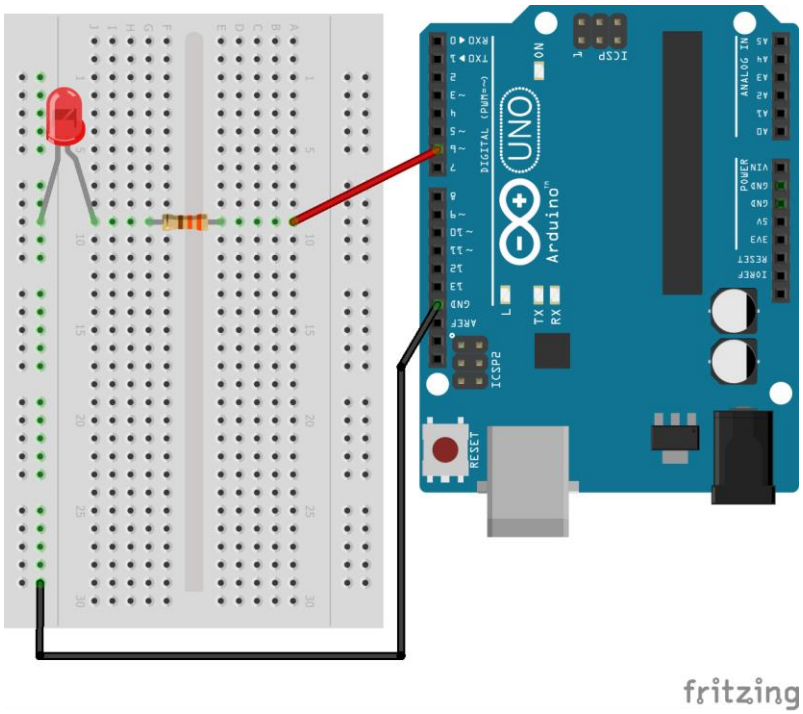
Etapa 02: Conecte o LED vermelho na protoboard com a perna maior, perna positiva, na linha horizontal e a perna menor, perna negativa, na coluna vertical. Como você já sabe, para utilizar um LED com um Arduino deve-se utilizar um resistor para reduzir a corrente, sendo assim, insira um resistor em série com a perna maior do LED.



Etapa 03: Conecte a coluna na qual está ligada a perna negativa do LED, utilizando um **jumper preto**, com o furo GND da placa Arduino, que significa terra, ou seja, ausência de tensão.



Etapa 04: Vamos agora fechar o circuito, como você quer controlar quando o LED irá acender, utilizando um jumper vermelho, conecte a linha na qual está a perna do resistor à pino 6 do Arduino UNO.

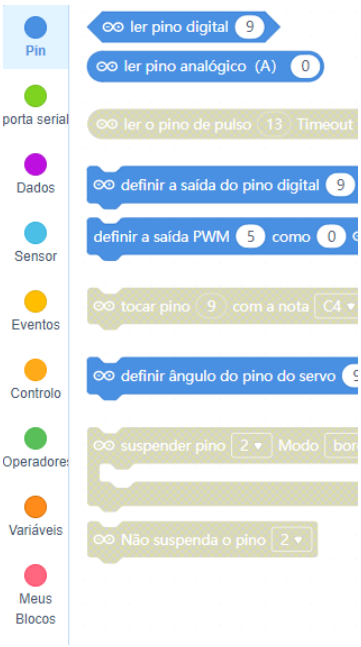


Etapa 05: Antes de alimentarmos o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

- Jumper vermelho saindo do pino 6 do Arduino e conectado à linha horizontal junto com o resistor de 330R;
- Resistor fazendo uma "ponte" conectando o jumper vermelho à perna maior do LED;
- LED vermelho conectado em série com o resistor e com a perna menor, negativa, na coluna vertical da placa de testes;
- **Jumper preto**, conectado na mesma coluna vertical que a perna menor do LED vermelho e com o pino GND, terra, do Arduino.

Etapa 06: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. É esperado que, da forma que o circuito está montado, o LED não acenda, já que ainda não fizemos a programação do Arduino para controlar o LED.

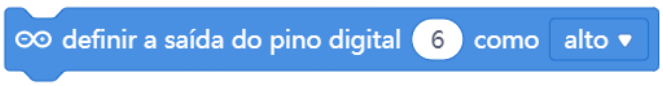
CRIANDO O PROGRAMA



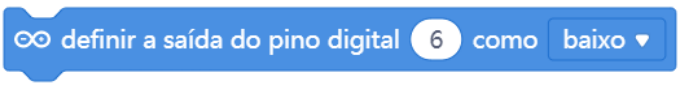
Etapa 07: Abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Viver".

Etapa 08: No conjunto de comando observe que, por padrão, está selecionado o grupo "Pin", de coloração Azul. Esses comandos controlam os pinos do Arduino. Os que estão com uma cor mais clara são aqueles que não funcionam com o modo de conexão "Viver". Como você deseja controlar um pino do Arduino, no qual está conectado o LED, clique e arraste para a área de programação o bloco "definir a saída do pino digital (9) como | alto ▼|".

Etapa 09: O bloco que você arrastou para a área de programação faz com que o pino digital 9 fique no estado "alto". Como você já estudou, o estado "alto" significa aquele com maior tensão, ou seja, com 5V de saída. O estado baixo, por sua vez, significa a menor tensão de saída, ou seja, 0V. Porém, nosso programa ainda não liga o LED, já que o pino que você está controlando é o pino 9 e o pino que o LED está conectado no seu circuito é o 6. Então, mude o valor do pino de 9 para 6 no bloco e clique sobre o bloco.



Etapa 10: Ao clicar no bloco você deve observar que o LED acende, ou seja, você conseguiu ligar ele apenas com a programação, parabéns! O desafio agora é desligar o LED, para fazer isso você deve mudar o valor da saída do pino digital 6 para "baixo" e clicar sobre o bloco, ou seja, como a saída será de 0V, sem alimentação, o LED irá desligar.

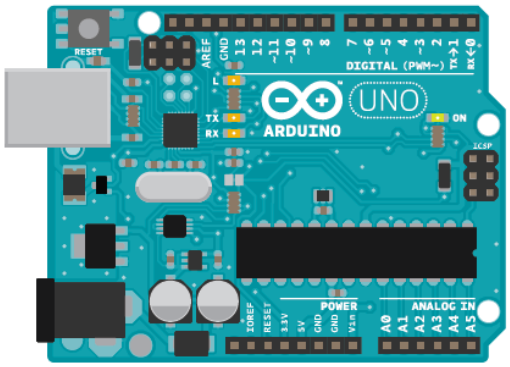


AGORA É A SUA VEZ

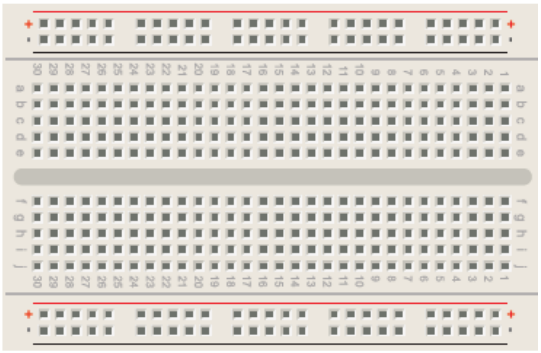
Assim você já consegue ligar e desligar um LED conectado no pino 6 do Arduino. Agora tente controlar um segundo LED, crie um circuito igual a esse criado nesse projeto, porém adicione um LED amarelo conectado no pino 7 e tente ligá-lo e desligá-lo.

Não funcionou? Confira novamente todas as ligações, pode ser apenas um pino conectado no furo errado, além disso, confira o valor do pino digital no bloco e se o Arduino está conectado com o mBlock e no modo "Viver"

03. PISCANDO UM LED COM O ARDUINO

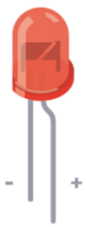


Arduino + Cabo USB (x1)



Placa de testes (x1)

Resistor (x1)



LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

COMPONENTES

CONTROLANDO O LED

NO PROJETO ANTERIOR VOCÊ CONSEGUIU LIGAR E DESLIGAR O LED COM O ARDUINO, CHEGOU A HORA DE ENSINAR ELE A LIGAR E DESLIGAR AUTOMATICAMENTE.

Descobertas: comandos de controle e repetição, *delay* e *loop*.

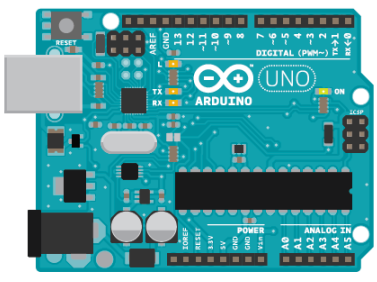
Tempo: **45 MINUTOS**

Dificuldade:

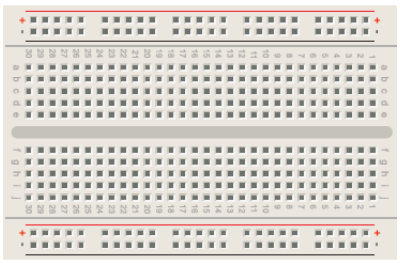
No projeto anterior você já conseguiu ligar e desligar o LED utilizando o código produzido no mBlock, agora é hora de fazer com que o próprio Arduino ligue e desligue o LED, fazendo o LED piscas, no ritmo que você deseja.

MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

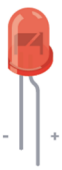


Arduino + Cabo USB (x1)



Placa de testes (x1)

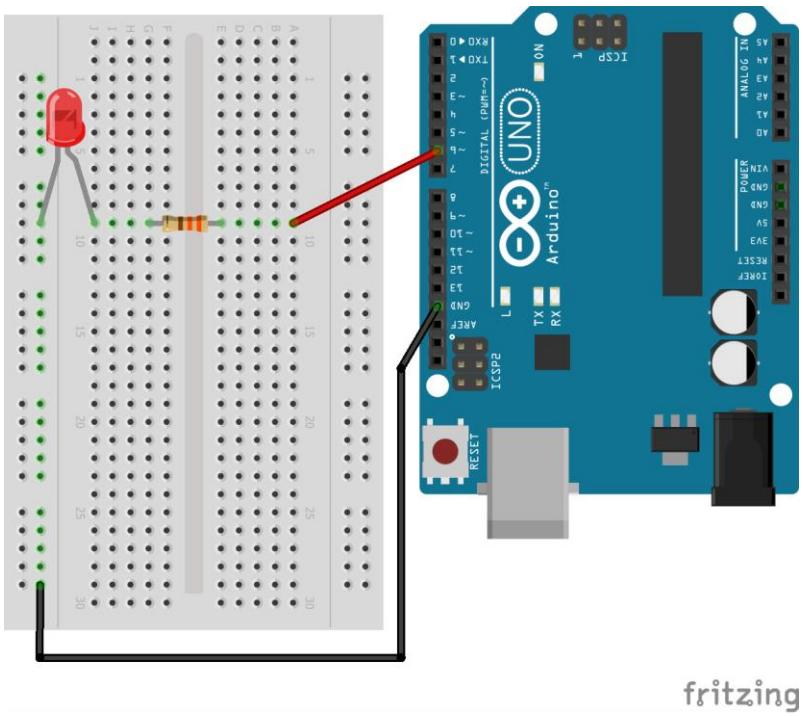
Resistor (x1)



LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

Etapa 02: Monte um circuito exatamente igual ao montado no exemplo anterior, com o LED em série com um resistor de 330 ohms, ligado ao pino 6 do Arduino e a outra perna do LED, a perna menor, ligada ao pino GND.



Etapa 03: Antes de alimentarmos o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

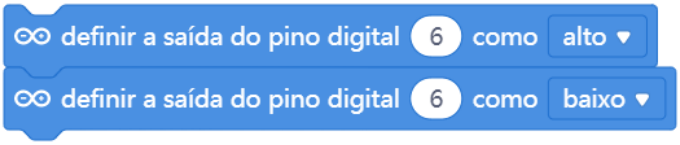
- Jumper vermelho saindo do pino 6 do Arduino e conectado à linha horizontal junto com o resistor de 330R;
- Resistor fazendo uma "ponte" conectando o jumper vermelho à perna maior do LED;
- LED vermelho conectado em série com o resistor e com a perna menor, negativa, na coluna vertical da placa de testes;
- **Jumper preto**, conectado na mesma coluna vertical que a perna menor do LED vermelho e com o pino GND, terra, do Arduino.

Etapa 04: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Viver".

CRIANDO O PROGRAMA

Agora que você já consegue ligar e desligar o LED, vamos fazer esse mesmo processo automaticamente. Utilizando o mesmo circuito elétrico, crie um programa para ligar o LED e depois de um tempo desligá-lo, fazendo o LED piscar.

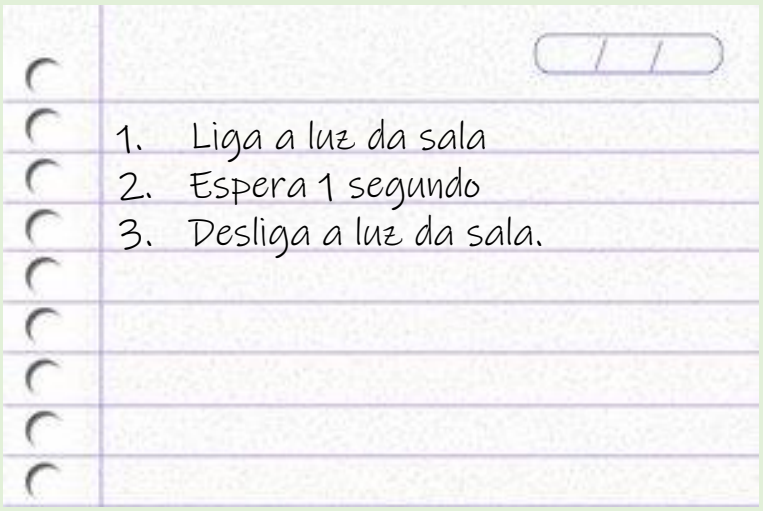
A primeira ideia que pode vir a sua cabeça é, simplesmente, adicionar um bloco definindo a saída do pino digital para alto e outro bloco definindo a saída para baixo. Repare que ao colocar um bloco abaixo do outro eles se encaixam e executam as funções da sequência, primeiro o que está mais acima e depois o de baixo.



Ao clicar sobre os blocos você percebe que o código funciona, porém o LED fica acesso por um intervalo de tempo muito pequeno, o desafio agora é: fazer o LED ligar por um segundo e depois desligar.

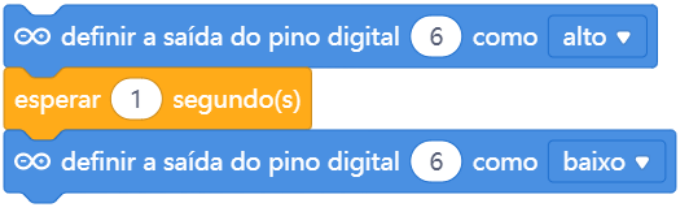
AGORA É A SUA VEZ

Se o seu desafio fosse ligar a luz da sala por apenas 1 segundo o que você faria? Coloque em uma folha de papel as etapas você teria que fazer para cumprir essa tarefa.



Resposta da atividade

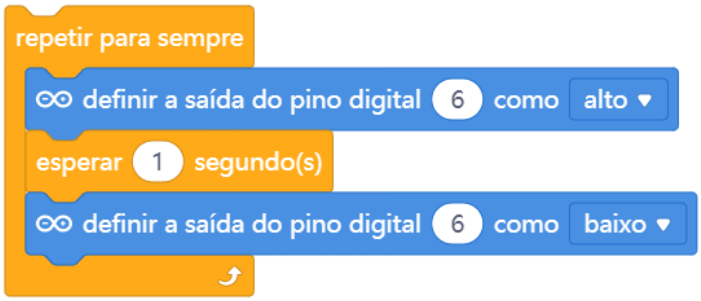
Etapa 05: Agora que você já conseguiu pensar em uma estratégia para superar o desafio, crie os comandos para fazer esse mesmo processo com o LED conectado no pino 6 do Arduino. Os comandos para fazer o LED ligar e desligar você já conhece, basta definir a saída do pino 6 como "alta" e depois "baixa". Para fazer o programa esperar 1 segundo, vamos utilizar um bloco presente no grupo "Controle", clique no bloco "esperar (1) segundo(s)" e arraste para a área de programação, logo após o comando para definir a saída do pino digital 6 como alto. Agora clique no primeiro bloco para que o programa seja executado.



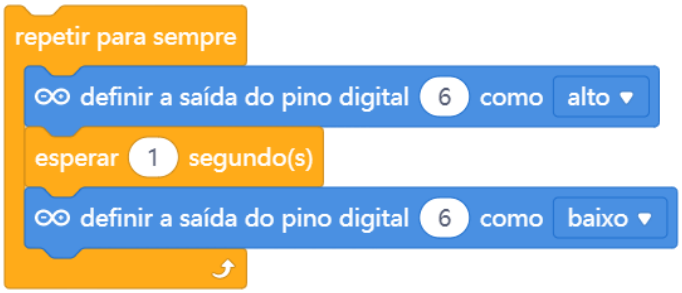
É esperado que o LED acenda, fique acesso por 1 segundo, e então apague. Se você quiser fazer um pisca-pisca, ou seja, o LED acende, apaga, acende, apaga, repetidas vezes; você terá que ficar clicando sempre no primeiro bloco para executar o programa.

Um dos principais usos dos componentes eletrônicos é para controlar dispositivos e automatizar tarefas. Dessa forma, você pode criar um código para fazer o LED piscar, várias vezes, sem que você precise ficar clicando o tempo todo. Para isso você deve utilizar o bloco "repetir para sempre". Com esse bloco, todos os comandos que são encaixados no seu interior serão executados e quando chegarem ao fim, voltarão a ser executados novamente.

Etapa 06: Arraste o bloco "repetir para sempre" para a área de programação e arraste os outros blocos para o seu interior.



Etapa 07: Repare que o LED pisca, muito rapidamente, a cada 1 segundo. Analise o código, linha a linha para tentar entender o que está acontecendo.

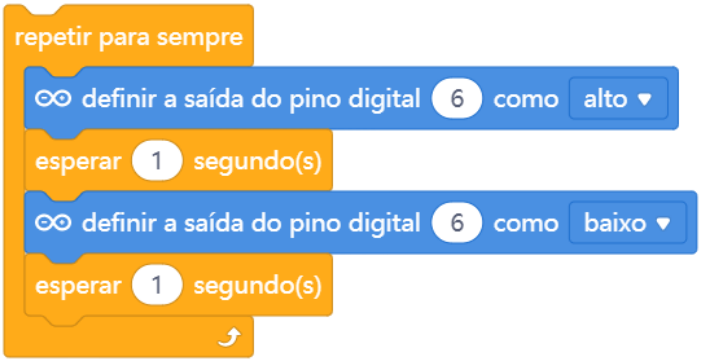


Repete para sempre
    Liga o LED no pino 6
    Espera 1 segundo
    Desliga o LED no pino 6
Volta para o começo

Resposta da atividade

No código que você construiu o LED no pino 6 é ligado, espera 1 segundo e é desligado, aí repete tudo novamente. Ou seja, ele é desligado no fim do código e volta pro começo, sendo ligado novamente, isso acontece muito rápido. Então, como fazer para que o LED fique um tempo desligado, antes de ligar novamente?

Etapa 08: Arraste outro bloco "esperar (1) segundo(s)" para a área de programação, abaixo do bloco que define a saída do pino digital 6 como baixo.

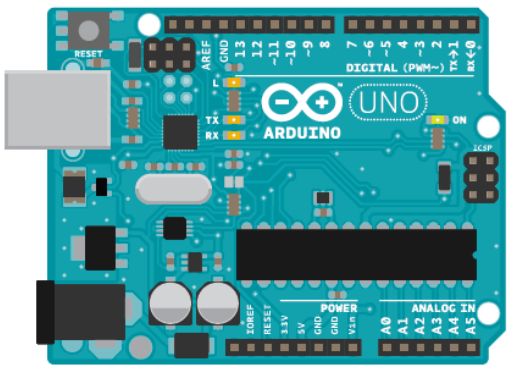


AGORA É A SUA VEZ

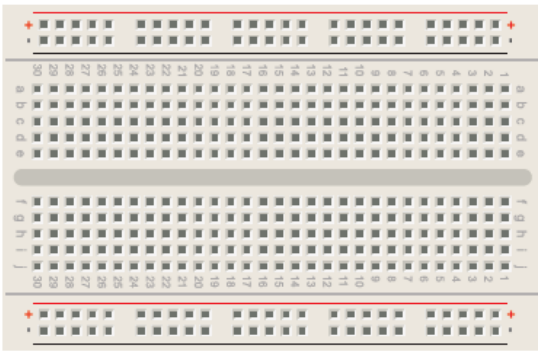
Agora você já consegue observar que o LED fica ligado por 1 segundo e depois, desligado por 1 segundo, e o ciclo sempre se repete. Agora experimente mudar os tempos de espera, tornando o pisca-pisca mais rápido e mais lento.

Não funcionou? Confira novamente todas as ligações, pode ser apenas um pino conectado no furo errado, além disso, confira o valor do pino digital nos blocos, se eles estão encaixados corretamente e se o Arduino está conectado com o mBlock e no modo "Viver".

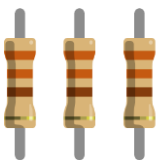
04. SEMÁFORO COM ARDUINO



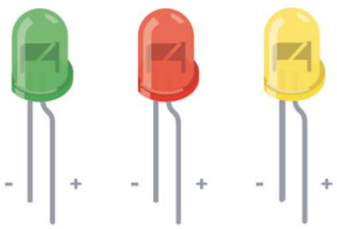
Arduino + Cabo USB (x1)



Placa de testes (x1)



Resistor (x3)



LED verde (x1)
LED vermelho (x1)
LED amarelo (x1)

Jumpers:
Vermelho (x1)
Preto (x1)
Amarelo (x1)
Verde (x1)

COMPONENTES

CONTROLANDO O TRÂNSITO

VOCÊ VAI CRIAR UM SEMÁFORO, IGUAL AQUELES QUE CONTROLAM O TRÂNSITO DA SUA CIDADE.

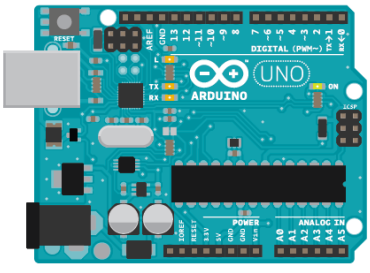
> Descobertas: criação de algorítmicos, programação sequencial, lógica booleana.

> Tempo: **60 MINUTOS**
>
> Dificuldade:

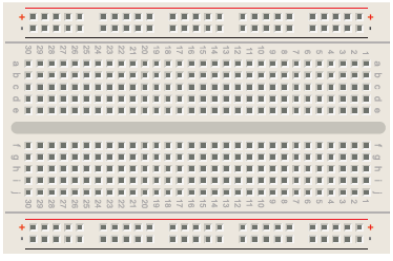
Nos projetos anteriores você já conseguiu controlar o acionamento de LEDs, agora é a hora de colocar na prática esses conhecimentos construindo um semáforo de trânsito.

MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

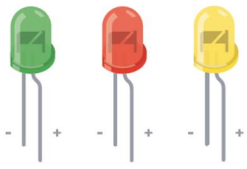


Arduino + Cabo USB (x1)



Placa de testes (x1)

Resistor (x3)



LED verde (x1)
LED vermelho (x1)
LED amarelo (x1)

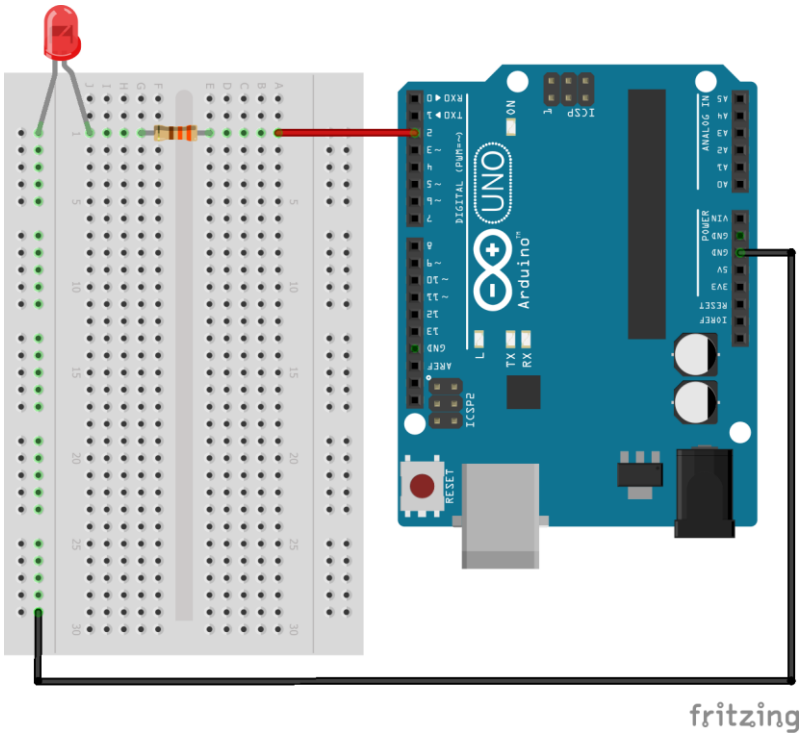
Jumpers:
Vermelho (x1) Amarelo (x1)
Preto (x1) Verde (x1)

Etapa 02: Antes de você iniciar a montagem do circuito escreva no seu caderno quais as partes que compõem um semáforo de trânsito.

1 Luz (lâmpada) vermelha
1 Luz (lâmpada) amarela
1 Luz (lâmpada) verde

Resposta da atividade

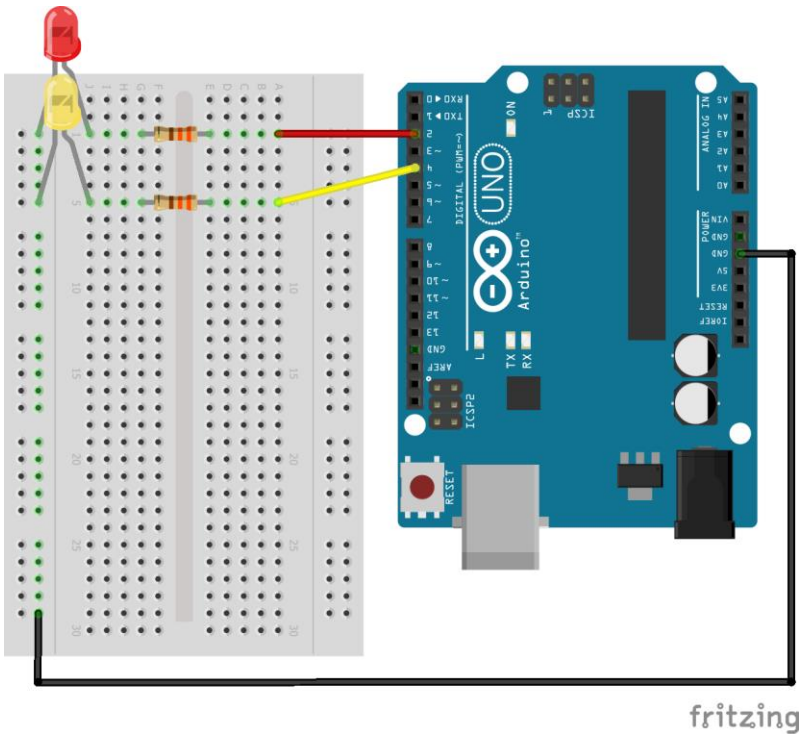
Etapa 03: Agora que você já sabe as partes que compõem um semáforo, já deve imaginar como será realizada a montagem do circuito. Como o semáforo possui três cores, precisaremos de três LEDs, um vermelho, um amarelo e um verde. Nessa primeira etapa, monte um circuito para controlar um LED vermelho, com o pino 2 do Arduino.



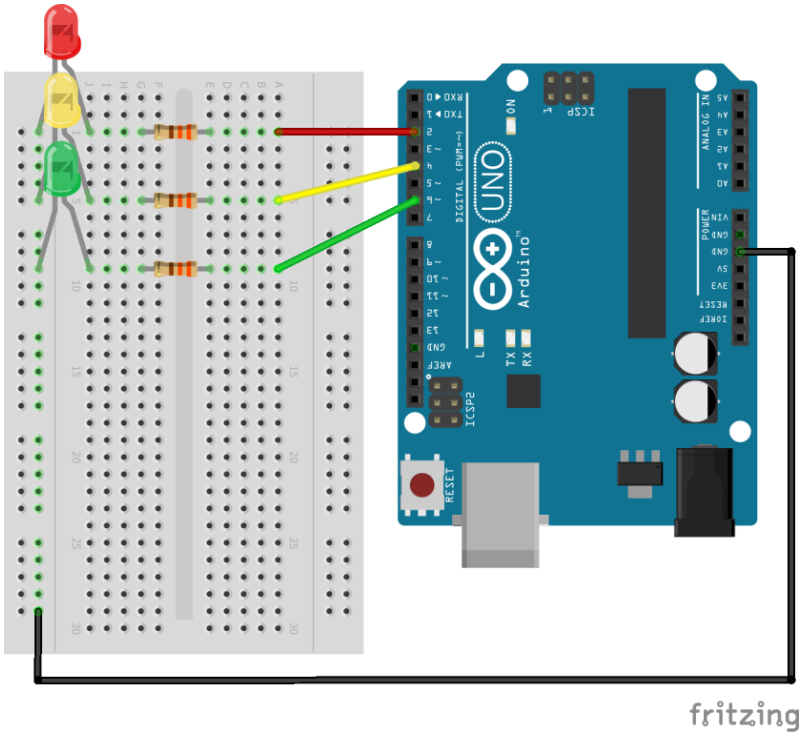
Nesse momento não se preocupe com a programação, ainda, você deve focar apenas na montagem do circuito elétrico.

É importante observar a posição dos componentes para facilitar a montagem, já que teremos muitos dispositivos.

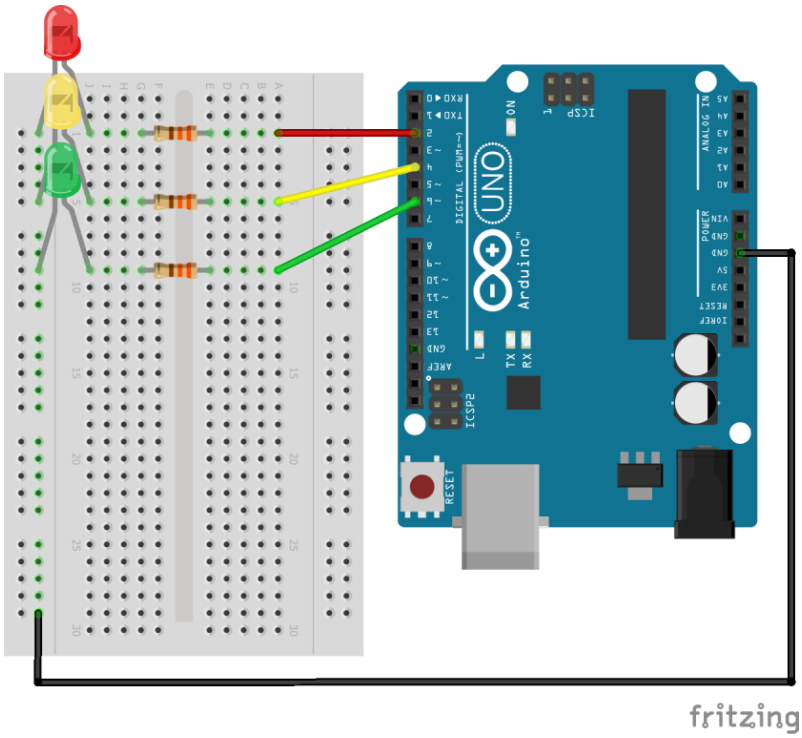
Etapa 04: Adicione ao seu circuito, agora, um LED amarelo, juntamente com o seu resistor, conectado ao pino 4 do Arduino.



Etapa 05: Por fim, adicione ao seu circuito um LED verde, juntamente com o seu resistor, conectado ao pino 6, do Arduino.



Etapa 06: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:



- Jumper vermelho saindo do pino 2 do Arduino e conectado à linha horizontal junto com o resistor de 330R, que faz uma "ponte" conectando o jumper vermelho à perna maior do LED vermelho, que tem sua perna menor conectada na coluna vertical da placa de testes;
- Jumper amarelo saindo do pino 4 do Arduino e conectado à linha horizontal junto com o resistor de 330R, que faz uma "ponte" conectando o jumper vermelho à perna maior do LED amarelo, que tem sua perna menor conectada na coluna vertical da placa de testes;
- Jumper verde saindo do pino 6 do Arduino e conectado à linha horizontal junto com o resistor de 330R, que faz uma "ponte" conectando o jumper vermelho à perna maior do LED verde, que tem sua perna menor conectada na coluna vertical da placa de testes;
- **Jumper preto**, conectado na mesma coluna vertical que a perna menor dos LEDs e com o pino GND, terra, do Arduino.

Etapa 07: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Viver".

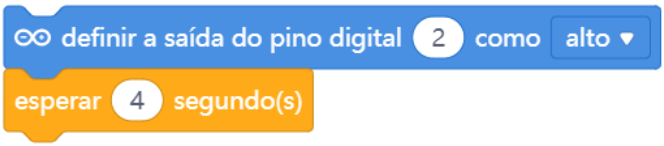
CRIANDO O PROGRAMA

Agora que você já montou o circuito do semáforo, resta apenas controlar os LEDs de acordo com o funcionamento de um semáforo, mas para fazer isso você deve compreender como funciona um semáforo. Analise os três estados do semáforo abaixo e crie, no caderno, a sequência de comandos necessárias para que o semáforo funcione.

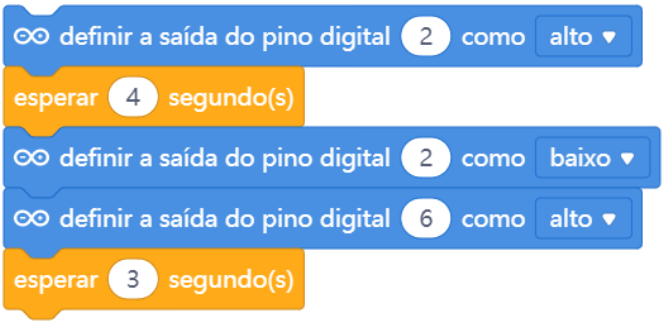
Ligar LED vermelho
Esperar 4 segundos
Desligar LED vermelho
Ligar LED verde
Esperar 3 segundos
Desligar LED verde
Ligar LED amarelo
Esperar 1 segundo
Desligar LED amarelo
Repetir o código

Possível resposta da atividade

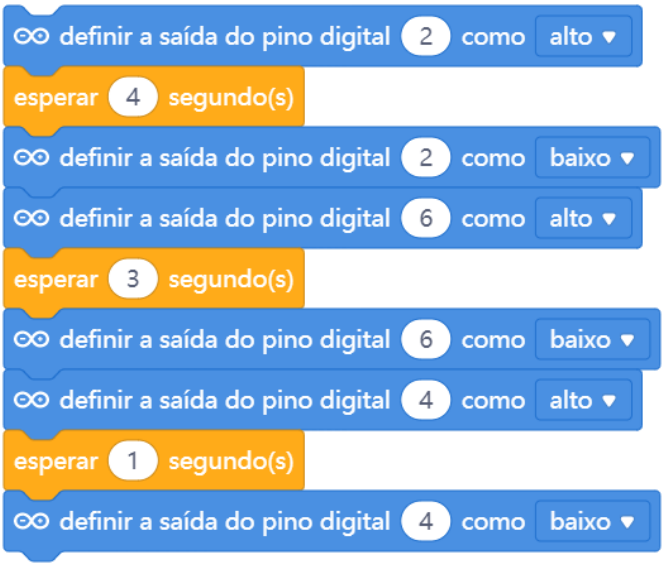
Etapa 08: Como você já sabe a sequência de comandos, chegou a hora de colocar a mão na massa. A primeira etapa é ligar o sinal vermelho. Dessa forma comece o código ligando o LED vermelho, conectado no pino 2 e aguardando 4 segundos, em seguida clique no bloco para testar.



Etapa 09: A próxima etapa é desligar o sinal vermelho e ligar o sinal verde. Ou seja, o pino 2 ficará como baixo na saída e o pino 6 como alto, ligando o LED verde e esperando 3 segundos. Clique no bloco para testar se, após 4 segundos o LED vermelho desliga e o LED verde acende.



Etapa 10: Agora que você já conseguiu acionar o sinal verde, basta desligar o sinal verde e ligar o amarelo por 1 segundo. Para fazer isso você deve colocar a saída do pino 6, do LED verde, no estado baixo e a do pino 4, do LED amarelo, no estado alto, aguardar 1 segundo e desligar o LED amarelo.



Etapa 11: Para finalizar, para que o semáforo funcione corretamente, é importante que após apagar o LED amarelo, o vermelho seja acesso e todos os comandos se repitam, para isso basta você utilizar o comando "repetir para sempre".

Confira se todos os comandos estão sendo realizados de forma correta, olhe para o seu circuito e confirme se:

1. LED vermelho fica ligado por 4 segundos e depois desliga;
2. LED verde é ligado na sequência, permanece 3 segundos ligado e desliga;
3. LED amarelo é ligado na sequência, permanece 1 segundo ligado e desliga.

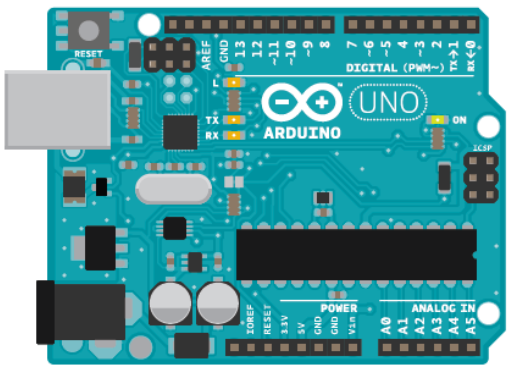
Em programação, diferentes códigos podem chegar ao mesmo resultado. O código proposto para a montagem do semáforo é apenas um exemplo, os blocos poderiam ser organizados de outra forma e o resultado obtido seria o mesmo, como no exemplo abaixo.

AGORA É A SUA VEZ

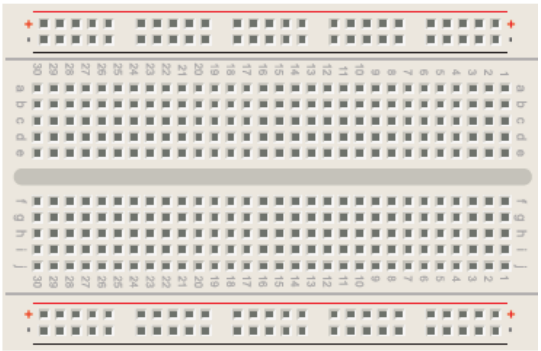
Agora que você conseguiu mudar o semáforo, teste mudanças no tempo de acionamento dos LEDs e na ordem em que eles são acionados. Que tal também projetar uma maquete do semáforo? Isto é, a construção fora da placa de prototipagem, simulando um semáforo real?

Não funcionou? Confira novamente as ligações, pode ser apenas um pino conectado errado, além disso, confira o valor dos pinos digitais nos blocos, se eles estão encaixados corretamente e se o Arduino está no modo "Viver".

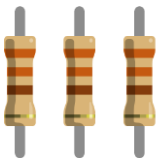
05. SEMÁFORO DUPLO COM ARDUINO



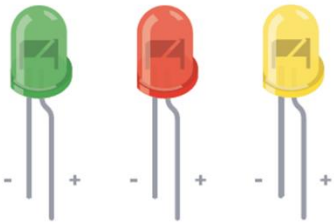
Arduino + Cabo USB (x1)



Placa de testes (x1)



Resistor (x6)



LED verde (x2)
LED vermelho (x2)
LED amarelo (x2)

Jumpers:
Vermelho (x2)
Preto (x1)
Amarelo (x2)
Verde (x2)

COMPONENTES

CONTROLANDO O TRÂNSITO EM UM CRUZAMENTO

VOCÊ JÁ CONSEGUIU CRIAR UM SEMÁFORO, MAS VOCÊ JÁ REPAROU QUE, GERALMENTE, SÃO DOIS SEMÁFOROS? UM EM CADA RUA DE UM CRUZAMENTO?

Descobertas: criação de algorítmicos, programação sequencial, raciocínio lógico.

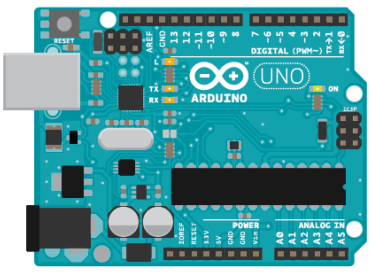
Tempo: **60 MINUTOS**

Dificuldade:

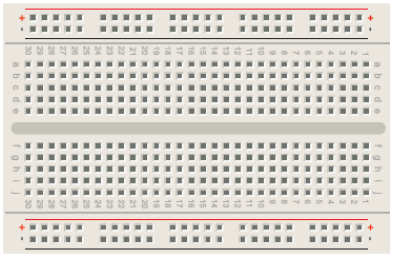
No projeto anterior você conseguiu criar o circuito e o programa para controlar um semáforo, porém, geralmente são dois semáforos em um cruzamento, está na hora de você criar um semáforo duplo.

MONTANDO O CIRCUITO

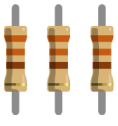
Etapa 01: Separe os componentes necessários para a montagem do circuito.



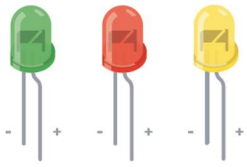
Arduino + Cabo USB (x1)



Placa de testes (x1)

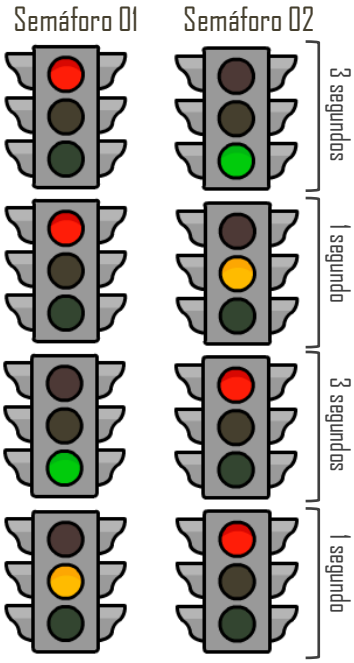


Resistor (x6)



LED verde (x2)
LED vermelho (x2)
LED amarelo (x2)

Jumpers:
Vermelho (x2) Amarelo (x2)
Preto (x7) Verde (x2)

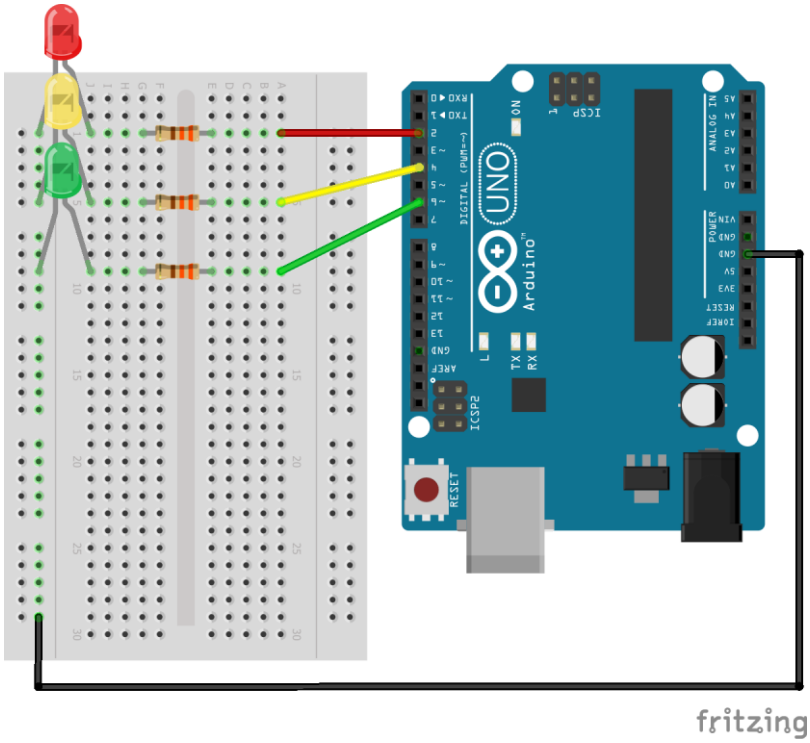


Etapa 02: Antes de você iniciar a montagem do circuito escreva no seu caderno quais as partes de dois semáforos de trânsito.

2 Luzes (lâmpadas) vermelha
2 Luzes (lâmpadas) amarela
2 Luzes (lâmpadas) verde

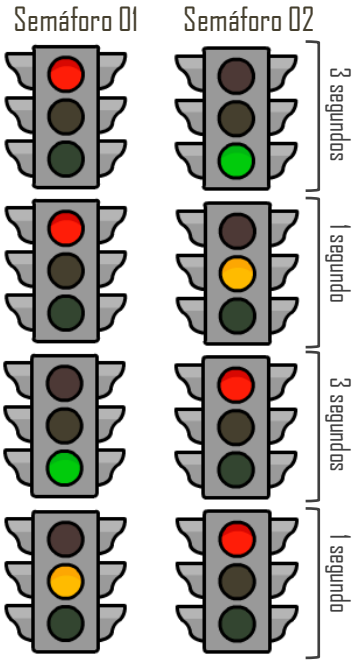
Resposta da atividade

Etapa 03: Agora que você já sabe as partes que compõem os dois semáforos, já deve imaginar como será realizada a montagem do circuito. Como cada semáforo possui três cores, precisaremos de seis LEDs, dois vermelhos, dois amarelos e dois verdes. Nessa primeira etapa, monte um circuito para controlar os três LEDs do primeiro semáforo, exatamente como o circuito montado no último projeto.

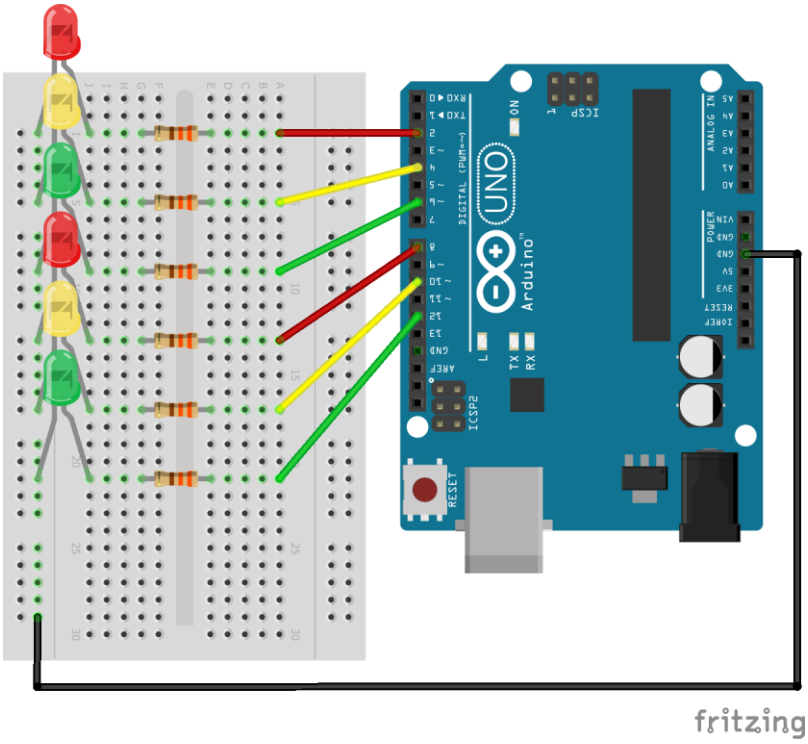


AGORA É A SUA VEZ

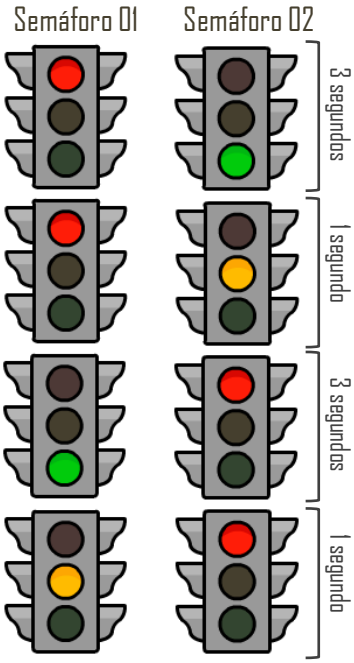
Agora que você já montou o circuito com os três LEDs que compõem o primeiro semáforo, adicione ao seu circuito outros três LEDs, um de cada cor, nos pinos 8, 10 e 12, para compor o segundo semáforo.



Etapa 04: Antes de alimentar o circuito, confira todas as conexões.



- Jumper vermelho saindo do pino 2 do Arduino e conectado à linha horizontal junto com o resistor conectado à perna maior do LED vermelho, que tem sua perna menor na coluna vertical da placa;
- Jumper amarelo saindo do pino 4 do Arduino e conectado à linha horizontal junto com o resistor conectado à perna maior do LED amarelo, que tem sua perna menor na coluna vertical da placa;
- Jumper verde saindo do pino 6 do Arduino e conectado à linha horizontal junto com o resistor conectado à perna maior do LED verde, que tem sua perna menor na coluna vertical da placa;
- Jumper vermelho saindo do pino 8 do Arduino e conectado à linha horizontal junto com o resistor conectado à perna maior do LED vermelho, que tem sua perna menor na coluna vertical da placa;
- Jumper amarelo saindo do pino 10 do Arduino e conectado à linha horizontal junto com o resistor conectado à perna maior do LED amarelo, que tem sua perna menor na coluna vertical da placa;
- Jumper verde saindo do pino 12 do Arduino e conectado à linha horizontal junto com o resistor conectado à perna maior do LED verde, que tem sua perna menor na coluna vertical da placa;
- **Jumper preto**, conectado na mesma coluna vertical que a perna menor dos LEDs e com o pino GND, terra, do Arduino.

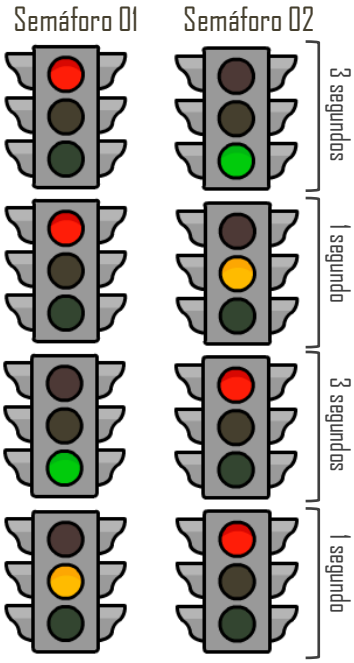


Etapa 05: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Viver".

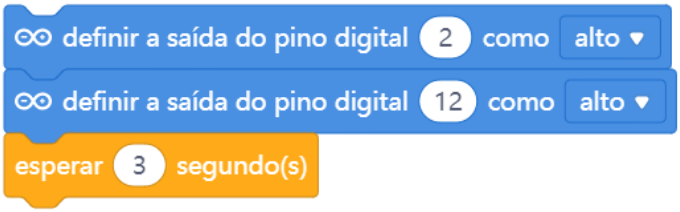
Etapa 06: Antes de você iniciar a programação analise todas as etapas envolvidas no funcionamento dos dois semáforos. Repare que, sempre que o Semáforo 01 está com sinal verde ou amarelo, o Semáforo 02 deve ficar com o sinal vermelho, e vice-versa. Crie, no caderno, a sequência de comandos necessárias para que os semáforos funcionem de forma conjunta.

Ligar LED vermelho do Semáforo 01
Ligar LED verde do Semáforo 02
Esperar 3 segundos
Desligar LED verde do Semáforo 02
Ligar LED amarelo do Semáforo 02
Esperar 1 segundo
Desligar LED vermelho do Semáforo 01
Desligar LED amarelo do Semáforo 02
Ligar LED verde do Semáforo 01
Ligar LED vermelho do Semáforo 02
Esperar 3 segundos
Desligar LED verde do Semáforo 01
Ligar LED amarelo do Semáforo 01
Esperar 1 segundos
Desligar LED amarelo do Semáforo 01
Desligar LED amarelo do Semáforo 02
Repetir o código

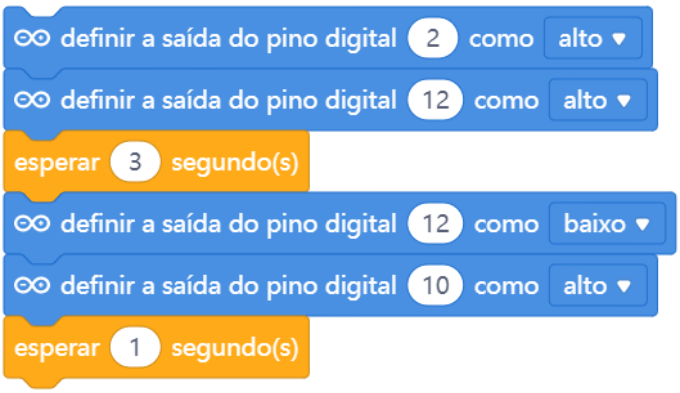
Possível resposta da atividade

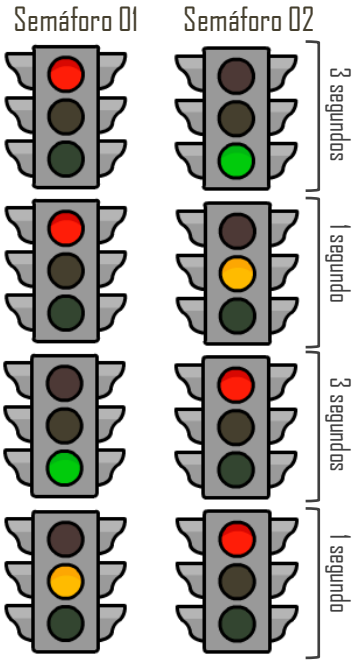


Etapa 07: Como você já sabe a sequência de comandos, chegou a hora de colocar a mão na massa. A primeira etapa é ligar o sinal vermelho do Semáforo 01 e o sinal verde do Semáforo 02. Dessa forma comece o código ligando o primeiro LED vermelho, conectado no pino 2, e o segundo LED verde, conectado no pino 12, e aguardando 3 segundos, em seguida clique no primeiro bloco para testar se os LEDs, vermelho e verde, acenderam.

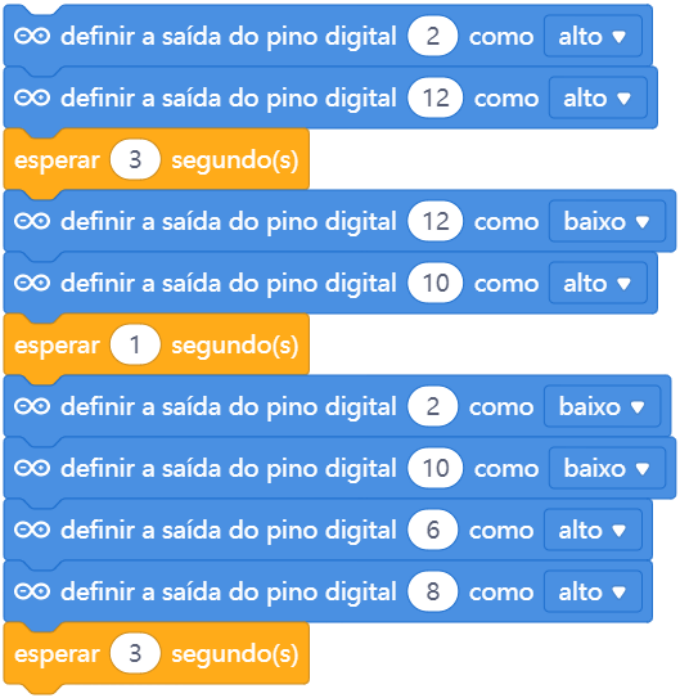


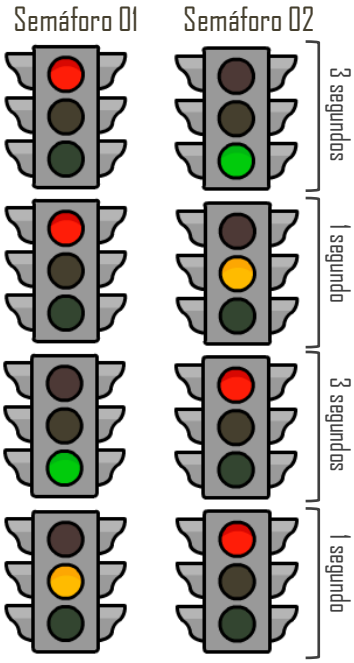
Etapa 08: A próxima etapa é desligar o sinal verde do semáforo 02 e ligar o sinal amarelo. Para tanto você deve definir a saída do pino 12, LED verde, como baixo e a saída do pino 10, LED amarelo, para alto, e aguardar 1 segundo. Perceba que, nesse momento, o sinal vermelho do semáforo 01 permanece inalterado. Clique no bloco para testar se o LED verde do semáforo 2 apaga após 3 segundos, ligando o LED vermelho do semáforo 2, enquanto o LED vermelho do semáforo 1 permanece ligado.



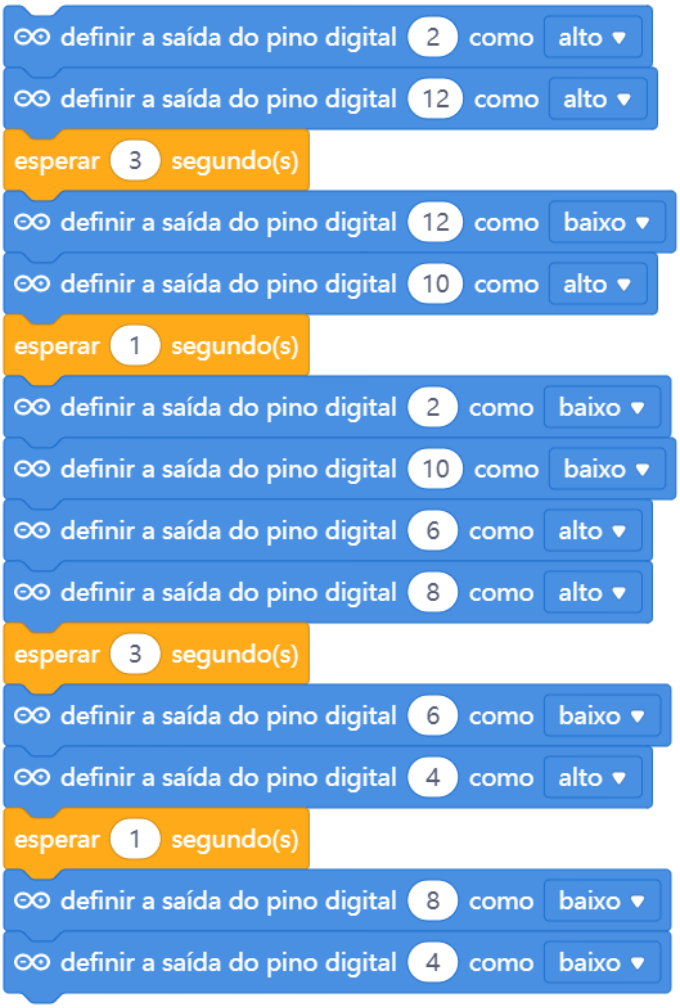


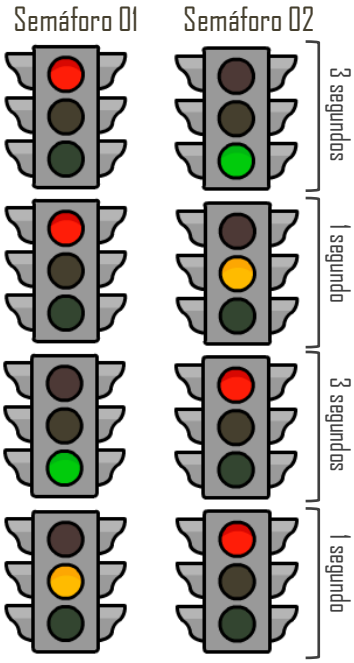
Etapa 09: Agora você tem que desligar os sinais vermelho, do semáforo 01, e amarelo, do semáforo 02, e ligar os sinais verde, do semáforo 01, e vermelho, do semáforo 02. Para tanto você deve definir as saídas dos pinos 2 e 10, LEDs vermelho e amarelo, como baixo e a saída dos pinos 6 e 8, LEDs verde e vermelho, para alto, e aguardar 3 segundos. Clique no bloco para testar o programa, até esse momento.





Etapa 10: Por fim, você deve apagar o sinal verde, do semáforo 01, e ligar o sinal amarelo, do semáforo 01, aguardar 1 segundo e desligar os sinais dos dois semáforos. Para fazer isso você deve definir a saída do pino 6, LED verde, como baixo e do pino 4, LED amarelo, como alto, aguardar 1 segundo, e então definir os dois pinos 8, LED vermelho, e 4, LED amarelo, para nível baixo.





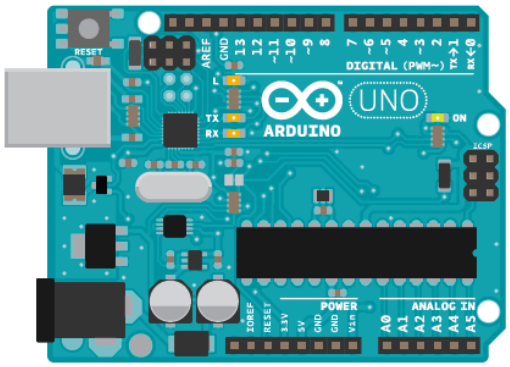
Etapa 11: Para concluir, teste o programa e confira todas as etapas, se estão de acordo com a representação na figura dos dois semáforos. Caso todas as etapas estejam corretas, adicione o comando "repetir para sempre", para que os comandos se repitam continuamente.

AGORA É A SUA VEZ

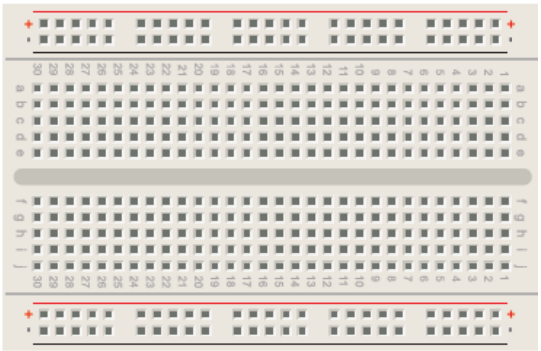
Que tal incluir, no seu projeto, uma sinalização para os pedestres? Utilizando LEDs verde e vermelho, você pode sinalizar a passagem segura para os pedestres, quando o sinal estiver vermelho.

Não funcionou? Confira novamente todas as ligações, pode ser apenas um pino conectado no furo errado, além disso, confira o valor dos pinos digitais nos blocos, se eles estão encaixados corretamente e se o Arduino está conectado com o mBlock e no modo "Viver".

06. CONTROLANDO A LUMINOSIDADE DO LED

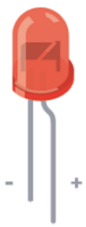


Arduino + Cabo USB (x1)



Placa de testes (x1)

Resistor (x1)



LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

COMPONENTES

CONTROLANDO O BRILHO DO LED

NESSE PROJETO VOCÊ VAI NÃO VAI SIMPLESMENTE LIGAR E DESLIGAR O LED, VOCÊ VAI CONTROLAR A INTENSIDADE DO BRILHO DO LED.

Descobertas: pinos digitais PWM, comandos de controle e repetição.

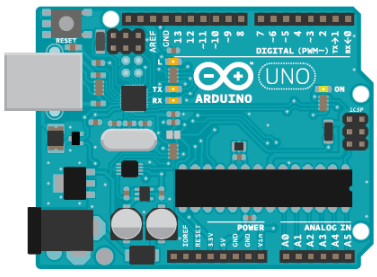
Tempo: **30 MINUTOS**

Dificuldade:

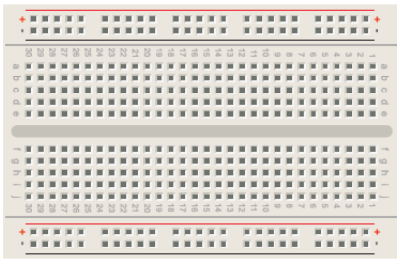
Nos projetos anteriores você já conseguiu ligar e desligar LED utilizando o código produzido no mBlock, agora é hora de aprender a controlar a intensidade da luminosidade do produzida pelo LED.

MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

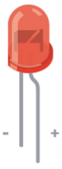


Arduino + Cabo USB (x1)



Placa de testes (x1)

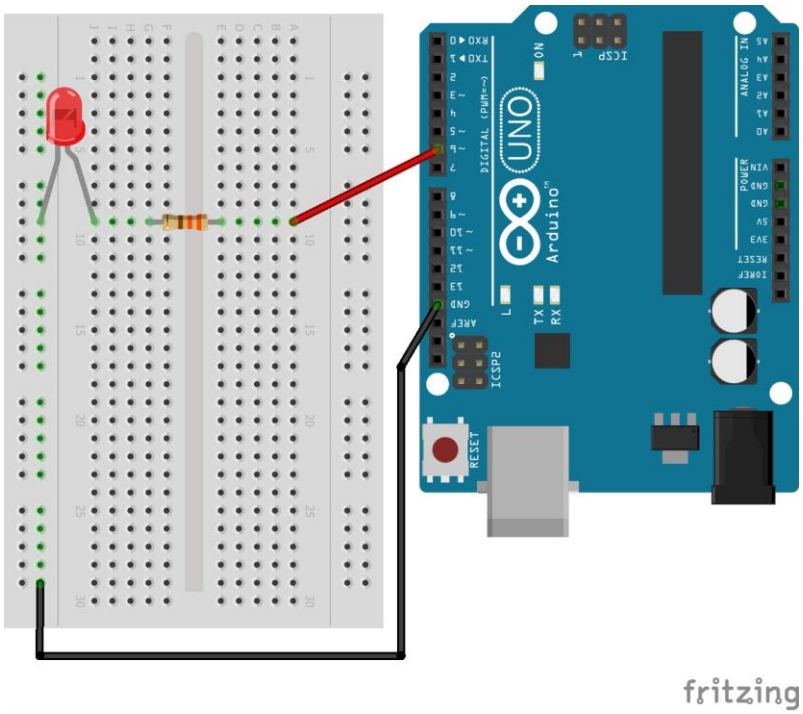
Resistor (x1)



LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

Etapa 02: Monte um circuito exatamente igual ao montado no projeto 2, com o LED em série com um resistor de 330 ohms, ligado ao pino 6 do Arduino e a outra perna do LED, a perna menor, ligada ao pino GND.

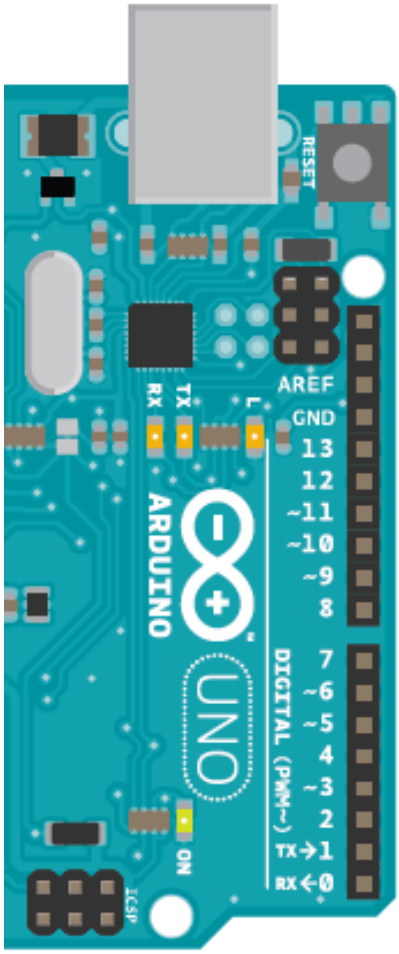


Etapa 03: Antes de alimentarmos o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

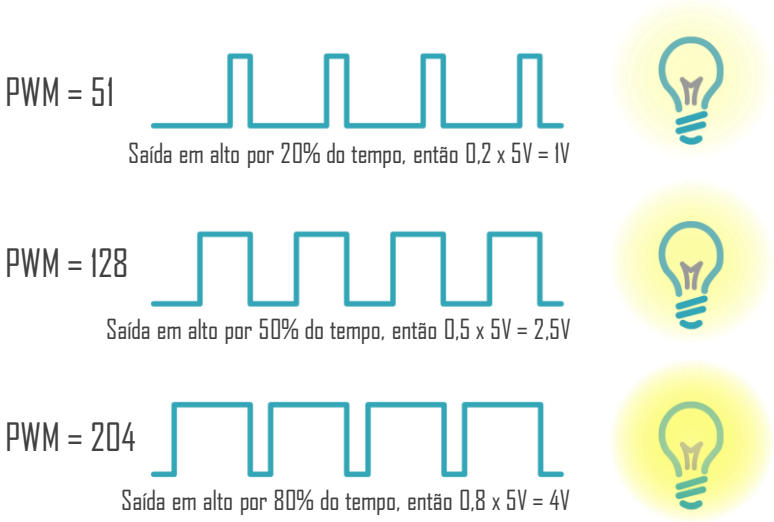
- Jumper vermelho saindo do pino 6 do Arduino e conectado à linha horizontal junto com o resistor de 330R;
- Resistor fazendo uma "ponte" conectando o jumper vermelho à perna maior do LED;
- LED vermelho conectado em série com o resistor e com a perna menor, negativa, na coluna vertical da placa de testes;
- **Jumper preto**, conectado na mesma coluna vertical que a perna menor do LED vermelho e com o pino GND, terra, do Arduino.

Etapa 04: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Viver".

SAÍDA PWM



Da mesma forma que nos projetos anteriores, você deve utilizar os pinos digitais do Arduino, porém, dessa vez, você vai utilizá-los no modo PWM. PWM significa pulso com modulação, em outras palavras, o a saída do pino não será constantemente alta ou baixa, como acontece com as saídas digitais normalmente, o que ocorre é que o sinal será pulsado ao longo do tempo. Observe, na imagem abaixo que, quanto maior o pulso, mais tempo o pino fica no estado alto, maior a voltagem de saída e maior a intensidade da luz da lâmpada.



Por padrão os pulsos variam em uma frequência de 0 a 255, sendo 0 quando não há a ocorrência de pulsos e PWM quando a amplitude do pulso é máxima.

Nas placas Arduino, os pinos PWM são sinalizados com o sinal de til (~), antes do seu número. Apenas os pinos com esse sinal podem ser utilizados no modo PWM. O Arduino Uno possui 6 pinos PWM, os pinos 3, 5, 6, 9, 10 e 11.

CONTROLE DE INTENSIDADE

Etapa 05: No mBlock, o comando utilizado para controlar a intensidade dos pulsos na saída de um pino PWM é o comando "definir a saída PWM (5) como (0) ∞" presente no grupo "Pin".

Etapa 06: Agora basta que você informe em qual pino o seu LED está conectado, nesse caso o pino 6, e qual a intensidade do sinal PWM. Teste seu bloco com a intensidade com valor 50, caso esteja tudo certo o LED deve ligar, porém com uma intensidade mais baixa do que a usual.

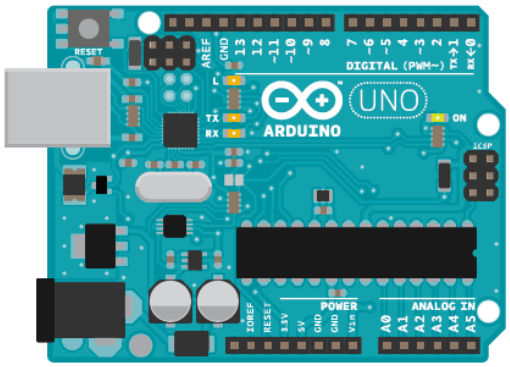
AGORA É A SUA VEZ

Experimente variar a intensidade do PWM e observe o que acontece com o brilho do LED. Crie um programa que mude a intensidade do LED do LED aos poucos, aumento de 0 até 255, variando 51 unidades a cada segundo.

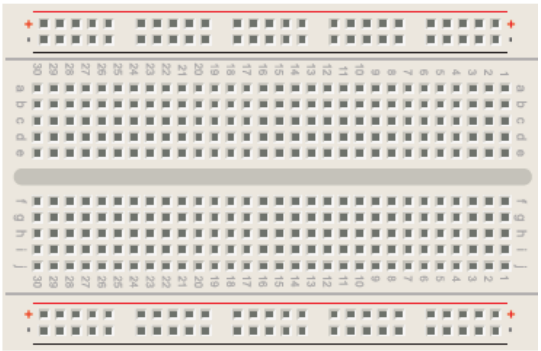
Não funcionou? Confira novamente todas as ligações, pode ser apenas um pino conectado no furo errado, além disso, confira o valor do pino digital no bloco, se o circuito está correto e se o Arduino está conectado com o mBlock e no modo "Viver".

Possível resposta da atividade

07. AUMENTANDO A LUMINOSIDADE DO LED

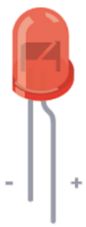


Arduino + Cabo USB (x1)



Placa de testes (x1)

Resistor (x1)



LED vermelho (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

COMPONENTES

VARIANDO E COMPARANDO

NESSE PROJETO VOCÊ VAI AUMENTAR A INTENSIDADE DA LUMINOSIDADE DO LED, AOS POUCOS E DE FORMA AUTOMÁTICA, SEM PRECISAR REPETIR BLOCOS.

Descobertas: variáveis, operadores e condicionais.

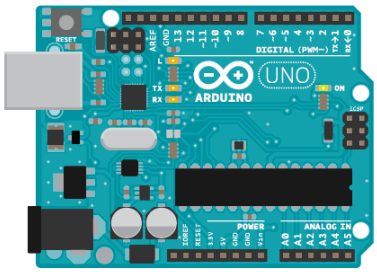
Tempo: **60 MINUTOS**

Dificuldade:

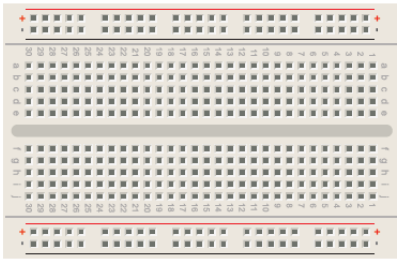
No desafio do projeto anterior você já conseguiu aumentar a intensidade da luminosidade de um LED, porém utilizando um código repetitivo, que lhe dá muito trabalho para montar e para alterar. Chegou a hora de você aprender a fazer esse mesmo processo com um código mais enxuto e que não vai te dar tanto trabalho na hora de fazer alguma alteração.

MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

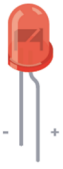


Arduino + Cabo USB (x1)



Placa de testes (x1)

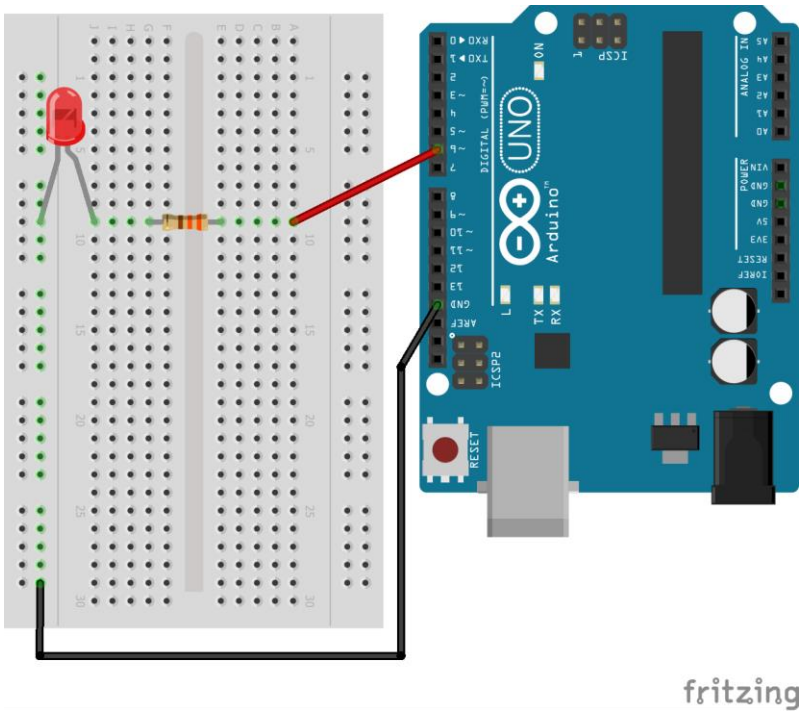
Resistor (x1)



LED vermelho (x1)

Jumpers:
Vermelho (x1)
Preto (x1)

Etapa 02: Monte um circuito exatamente igual ao montado no projeto anterior, com o LED em série com um resistor de 330 ohms, ligado ao pino 6 do Arduino e a outra perna do LED, a perna menor, ligada ao pino GND.



Etapa 03: Antes de alimentarmos o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

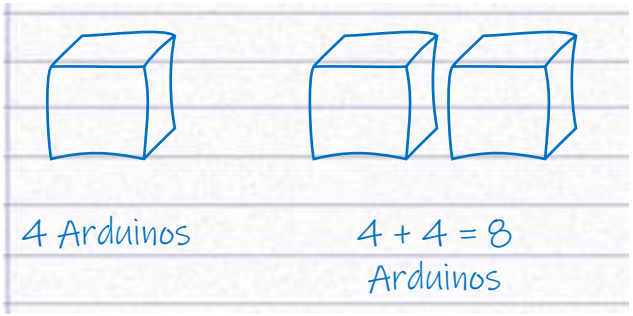
- Jumper vermelho saindo do pino 6 do Arduino e conectado à linha horizontal junto com o resistor de 330R;
- Resistor fazendo uma "ponte" conectando o jumper vermelho à perna maior do LED;
- LED vermelho conectado em série com o resistor e com a perna menor, negativa, na coluna vertical da placa de testes;
- **Jumper preto**, conectado na mesma coluna vertical que a perna menor do LED vermelho e com o pino GND, terra, do Arduino.

Etapa 04: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Viver".

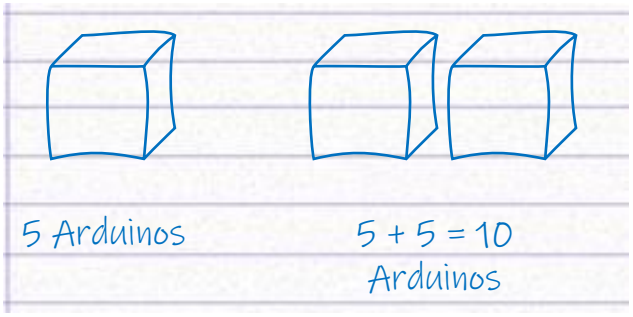
VARIÁVEIS

Analisando o código do exemplo ao lado, caso você deseje alterar o valor do tempo de espera entre a alteração nos valores PWM, ou alterar o pino ao qual o LED está conectado, teria que alterar esses valores em cada um dos blocos, o que é muito trabalhoso, principalmente para projetos maiores. Para evitar esse trabalho podemos usar variáveis, que são blocos aos quais definimos um valor e sempre vamos utilizar esses blocos na nossa programação, sempre que desejamos mudar o valor de uma variável basta mudar no bloco em que definimos o valor dela e a mudança ocorrerá nos demais.

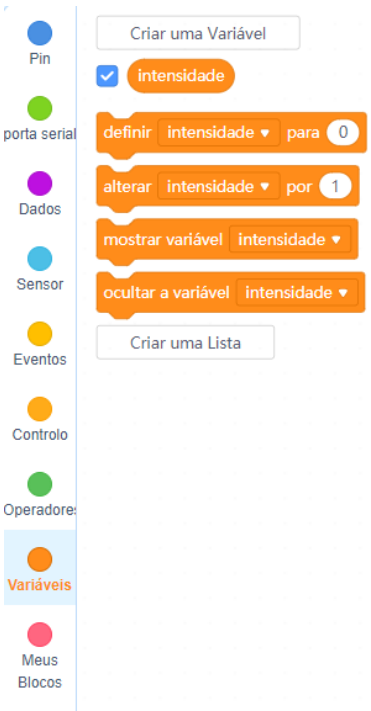
Para facilitar a compreensão do conceito de variável imagine o seguinte exemplo, digamos que você tenha uma caixa fechada e percebe que consegue guardar nessa caixa 4 Arduinos. Se você tem duas caixas iguais, e em cada caixa cabem 4 Arduinos, quantos Arduinos cabem nas duas caixas?



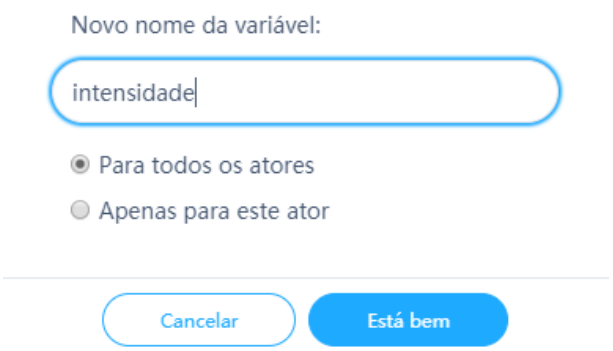
Digamos que agora você percebeu que as caixas eram um pouco maiores do que você pensou e, ao invés de 4, em cada caixa cabem 5 Arduinos, quantos Arduinos cabem nas duas caixas?



Podemos entender variáveis de forma semelhante a esse exemplo. Teríamos uma variável caixa, que inicialmente teve seu valor igual a 4, em seguida seu valor foi modificado para 5, e as operações que envolvem essa variável serão alteradas de acordo com a mudança no seu valor.



Etapa 05: Crie uma nova variável clicando no grupo "Variáveis" e em seguida no botão "Criar uma Variável". Na janela que se abre nomeie sua variável como "intensidade" e clique no botão "Está bem".

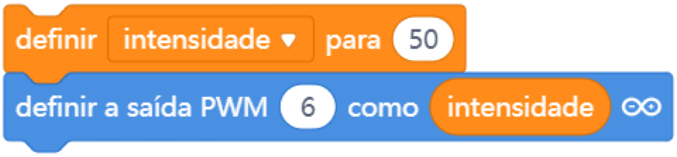


Etapa 06: Agora atribua um valor a nossa variável, para tanto, utilize o bloco "definir |intensidade ▼| para (0)", arraste para a área de programação. A partir desse momento a variável "intensidade" tem o valor igual a zero.

Etapa 07: Chegou a hora de você utilizar sua variável. Abaixo desse bloco onde é definido o valor de intensidade, adicione um bloco para controlar um pino PWM e já defina o valor do pino para 6, conforme o circuito montado.

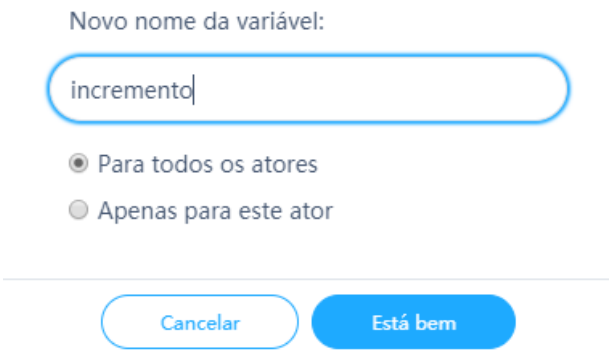
Etapa 08: Agora utilize a variável "intensidade", para definir a intensidade da saída PWM do pino 6. Para fazer isso, no grupo variáveis, clique e arraste o bloco "intensidade" e solte ele dentro do espaço destinado à intensidade do PWM.

Etapa 09: O bloco intensidade deve ser inserido dentro do bloco que define a saída PWM, como na imagem abaixo. Dessa forma o valor da luminosidade do LED irá variar de acordo com o valor definido para a variável "intensidade". Mude o valor definido para "intensidade" e observe o que acontece com a luminosidade do LED.



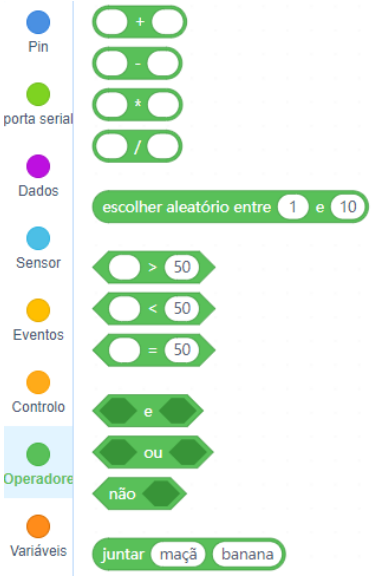
Observe que o valor das variáveis aparece no palco, junto ao Panda. Isso é muito bom para você acompanhar a execução do seu programa, de acordo com os valores das variáveis.

Etapa 10: Como o seu desafio nesse projeto é aumentar o brilho do LED, ao longo do tempo, crie mais uma variável, que vai definir o quanto o brilho deverá aumentar ao longo do tempo e nomeie essa variável de "incremento".



Etapa 11: Defina o valor da variável incremento como sendo 5.

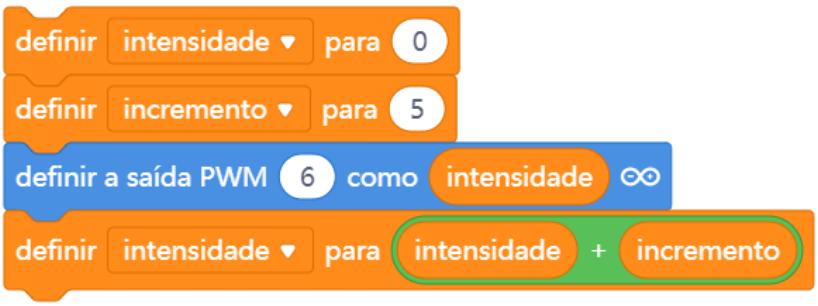
OPERADORES



Etapa 12: A variável incremento determina o valor no qual a variável intensidade deverá aumentar, para aumentar o brilho do LED. Porém, para fazer esse processo temos que entender que o novo valor da intensidade deve ser igual ao valor anterior, somado ao valor do incremento. Para fazer operações, nesse caso será uma operação de soma, devemos utilizar o bloco de soma, (( ) + ( )), presente dentro do grupo **Operadores**. Adicione as duas variáveis nos espaços destinados a soma. O resultado desse bloco será a soma entre o valor da intensidade e o incremento.

Os operadores podem ser utilizados para a realização de operações matemáticas, comparação entre valores, operações lógicas, além de operações com palavras e frases.

Etapa 13: Defina agora, abaixo do bloco de controle do PWM, o valor da variável intensidade, como sendo o bloco da soma entre intensidade e incremento.

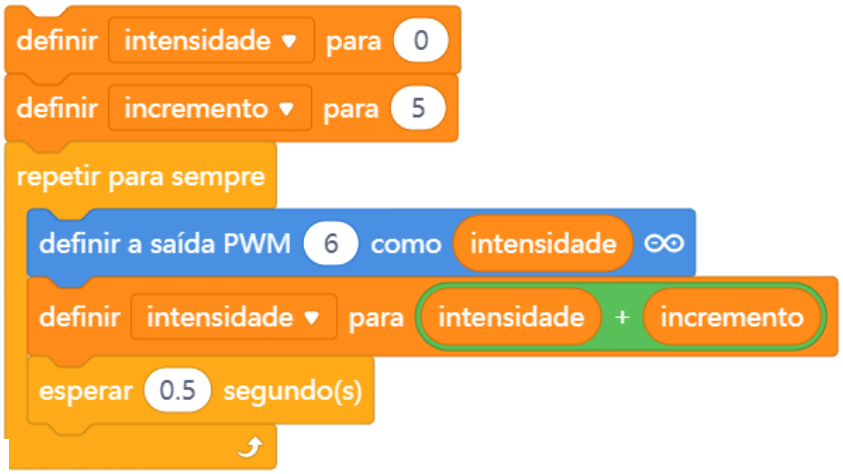


Para entender o procedimento acima, observe o valor da variável intensidade no palco, antes da execução do programa o valor é 0, após a execução, o valor passa a ser 5, logo o valor da intensidade passou a ser o valor anterior somado ao incremento, ou seja, 0 + 5 = 5. Se você analisar o código o que ocorre com as variáveis é o seguinte:

```
intensidade = 0
incremento = 5

intensidade = intensidade + incremento
intensidade =      0      +      5
intensidade = 5
```

Etapa 13: Agora, para que o seu LED tenha seu brilho aumentado aos poucos, não basta aumentar o valor da intensidade apenas uma vez, esse valor tem que aumentar até chegar no valor máximo, em 255. Para fazer isso basta você repetir as etapas que definem o PWM do pino 6 e a etapa que realiza a soma entre a intensidade e o incremento. Para que o aumento do brilho ocorra de forma suave, você deve esperar um tempinho (0.5 segundos) entre os incrementos. Ao executar o programa, observe como o brilho do LED aumenta, com o passar do tempo, e como o valor da variável intensidade aumenta.



Ao observar o valor da variável intensidade no palco, você deve ter percebido que esse valor supera o valor máximo que pode ser atribuído à um pino PWM, que é de 255. O desafio agora é, como aumentar o valor da intensidade apenas até chegar em 255? Depois de 255 não executar mais o código, já que já se obteve o valor máximo para o pino PWM.

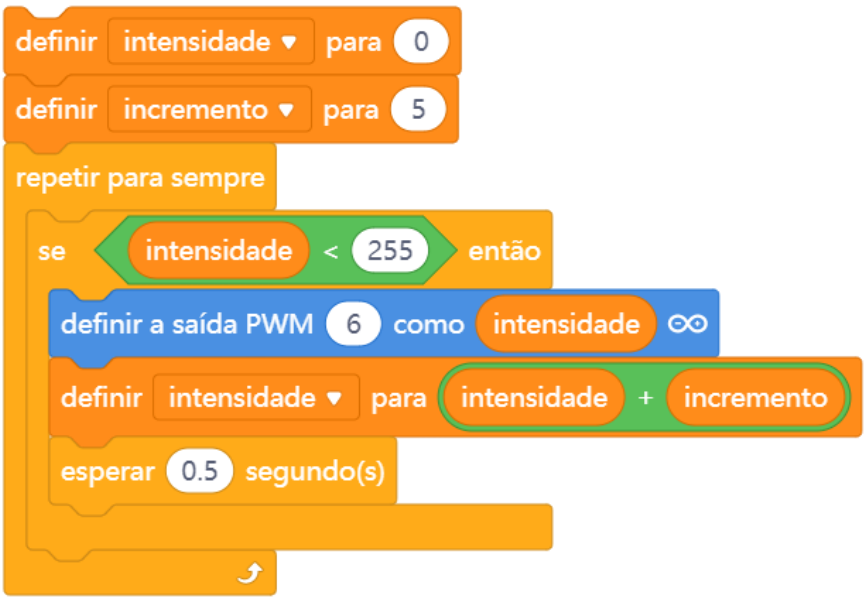
CONDICIONAIS

Para superar esse desafio vamos utilizar uma estrutura de **condicionais**. Essas estruturas permitem que o programa execute um determinado comando de acordo com algumas condições estabelecidas, como no exemplo abaixo, a função SE é uma condicional e tudo que está dentro dela só será executado se a condição for obedecida, nesse exemplo, a condição é que a idade seja maior que 16 anos, caso ela seja obedecida você poderá votar, caso contrário, não poderá.

```
SE sua idade for maior que 16 anos
    Você pode votar
```

Etapa 14: Encontre a condicional "se" no grupo "Controle" e arraste para a área de programação. Esse bloco funciona de uma forma bem simples, tudo que estiver encaixado dentro dele será executado sempre que a condição, inserida após o "se", for obedecida. É comum, em condicionais, o uso de operadores de relação, principalmente o maior que (>), menor que (<) e a igualdade (=). Como nesse projeto você quer que o valor da variável intensidade seja aumentado até 255, podemos utilizar como condição que intensidade tem que ser menor que 255 para que o código seja executado.

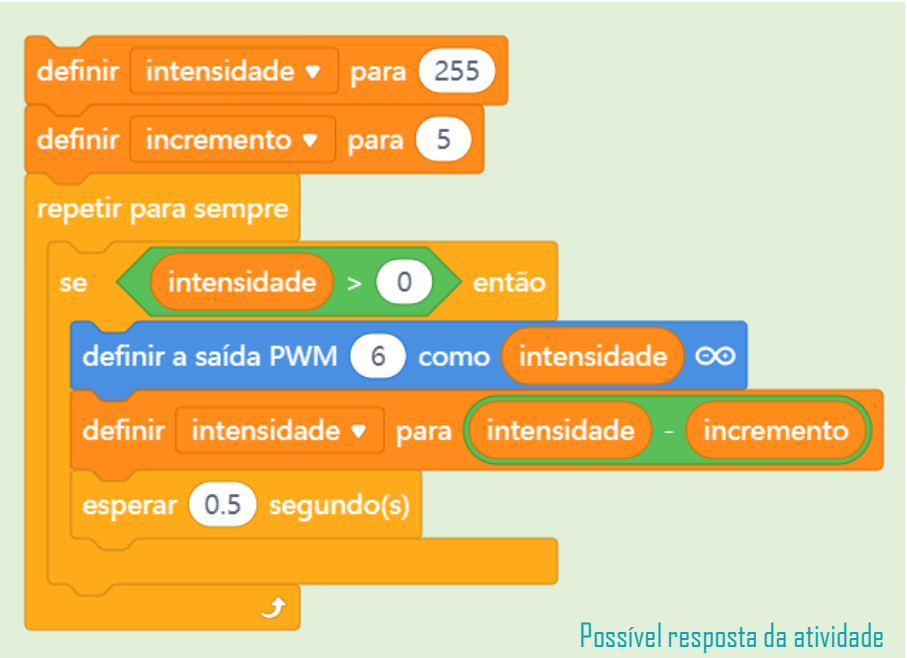
Etapa 15: Arraste agora o operador acima e o coloque como condição para o bloco condicional "se". Em seguida, arraste para o interior do bloco "se" os comandos que devem ser executados apenas quando a condicional for verdadeira, ou seja, quando a intensidade for menor que 255. Teste o código novamente, acompanhando o aumento do valor da variável intensidade, até parar em 255.



AGORA É A SUA VEZ

Experimente variar o valor do incremento e do tempo de espera, para deixar a redução do brilho mais suave. Crie um programa que reduza a intensidade do LED, defina a intensidade inicial como sendo 255, vá reduzindo o valor até zero. Lembrando que o valor da intensidade não pode ser menor que zero.

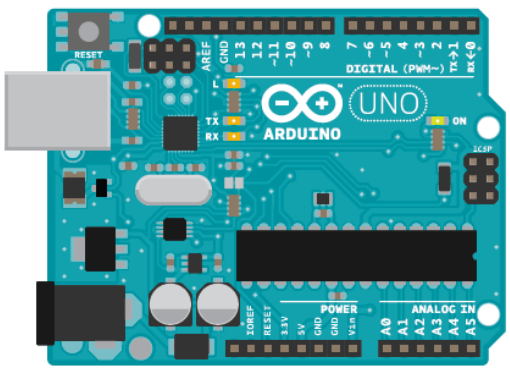
Não funcionou? Confira novamente todas as ligações, pode ser apenas um pino conectado no furo errado, além disso, confira o valor do pino digital no bloco, se o circuito está correto e se o Arduino está conectado com o mBlock e no modo "Viver".



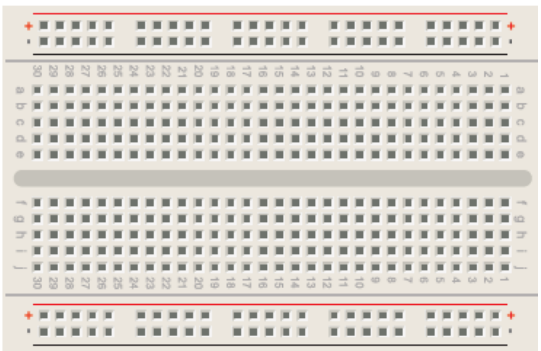
Possível resposta da atividade

Lembre que em programação diferentes códigos podem chegar ao mesmo resultado. O código proposto para esse projeto é apenas um exemplo, os blocos poderiam ser organizados de outra forma e o resultado obtido seria o mesmo.

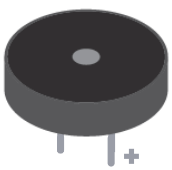
08. CONTROLANDO UM BUZZER



Arduino + Cabo USB (x1)



Placa de testes (x1)



Buzzer (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

COMPONENTES

FAZENDO BARULHO

CUIDADO! CHEGOU A HORA DE VOCÊ PRODUZIR SONS, UTILIZANDO O ARDUINO E UM BUZZER CRIAR UM ALARME!

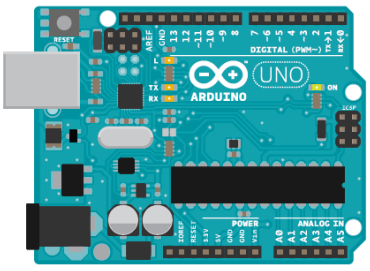
> Descobertas: funcionamento de buzzer, programação sequencial, gravação permanente na placa Arduino.

> Tempo: **30 MINUTOS**
>
> Dificuldade: ■□□□□

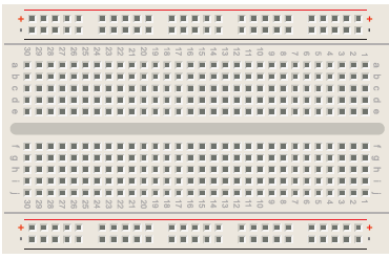
Sabe aquele alerta sonoro do micro-ondas quando o tempo de aquecimento acabou? Esse som é emitido por um dispositivo eletrônico chamado Buzzer, hoje você vai aprender a controlar o Buzzer para criar alarmes.

MONTANDO O CIRCUITO

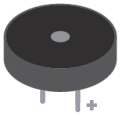
Etapa 01: Separe os componentes necessários para a montagem do circuito.



Arduino + Cabo USB (xl)



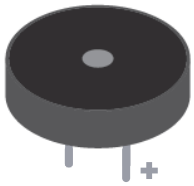
Placa de testes (xl)



Buzzer (xl)

Jumpers:
- Vermelho (xl)
- Preto (xl)

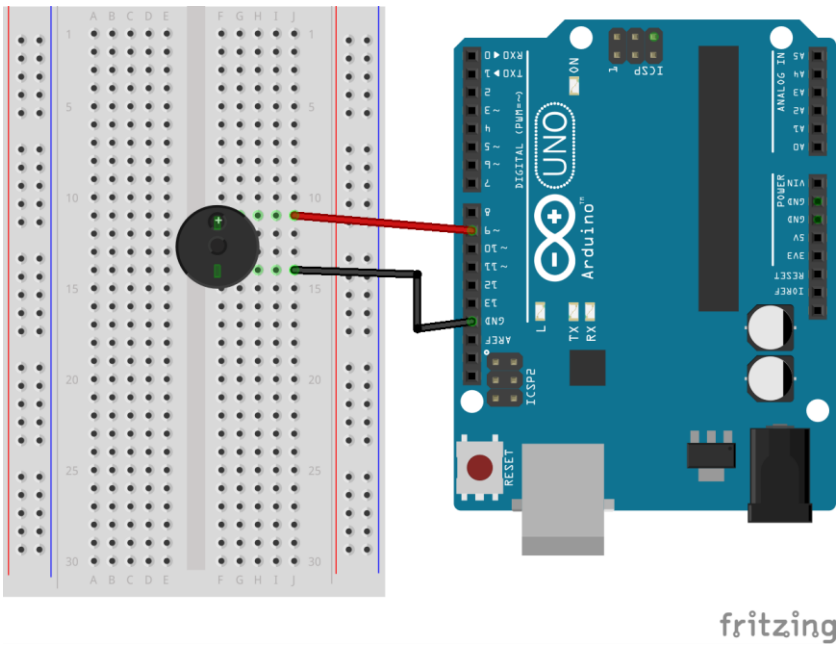
BUZZER



O Buzzer é um dispositivo piezoelétrico, ou seja, ele vibra quando recebe corrente elétrica. Essa vibração move o ar na superfície do buzzer, criando ondas sonoras.

Esse dispositivo é muito utilizado em alertas sonoros de equipamentos eletrônicos. Outro uso muito importante é como alerta sonoro para pessoas com deficiência visual ou com baixa visão, em semáforos, por exemplo.

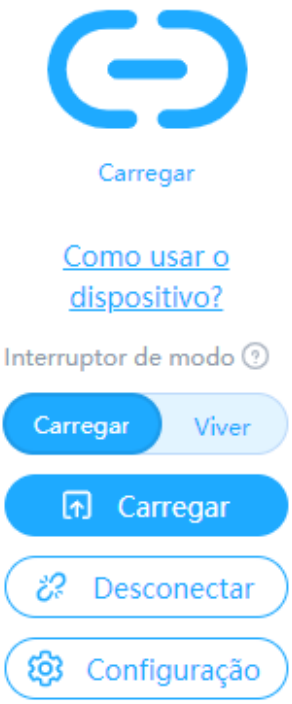
Etapa 02: Antes montar o circuito, identifique o pino positivo da placa, geralmente é o pino com a perna maior, após identificar encaixe o buzzer na placa de testes e, utilizando um jumper vermelho, conecte o lado positivo ao pino 9 do Arduino Uno e o lado negativo, utilizando um **jumper preto**, ao pino GND.



Etapa 03: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

- Jumper vermelho saindo do pino 9 do Arduino e conectado à linha horizontal junto na qual está conectado o pino positivo (maior) do buzzer. A linha na qual está conectado o pino menor do buzzer está conectada, com o **jumper preto**, ao pino GND do Arduino UNO.

MODO CARREGAR

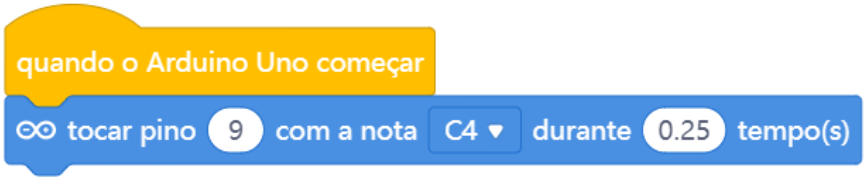


Etapa 04: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Carregar".

Ao trabalhar com algum componentes eletrônicos, como o buzzer, temos que utilizar o modo "Carregar". Nesse modo, as mudanças que realizamos na programação só serão realizadas quando carregamos essa programação para o Arduino. Ao contrário do modo "Viver", no modo "Carregar", essa programação fica salva no Arduino e será executada mesmo que a placa não esteja conectada em um computador, alimentada por pilhas, por exemplo.

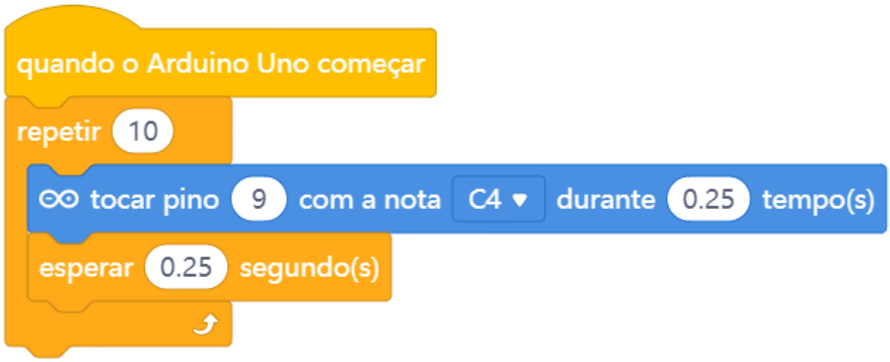
Etapa 05: No grupo "Pin", você deve perceber que, no modo carregar, todos os blocos estão disponíveis, o modo carregar permite que você tenha mais blocos para a criação dos seus programas. Para controlar o buzzer escolha o bloco "tocar pino (9) com a nota | C4 ▼| durante (0.25) tempo (s)" e arraste para a área de programação. Com esse bloco você irá tocar, no pino 9, a nota musical C4, dó maior com 4ª, durante 0,25 segundos.

Etapa 06: Porém, antes de você carregar o código para o Arduino, você tem que informar ao Arduino quando esse código será executado, para isso vá no grupo "Eventos" e encaixe o bloco "quando o Arduino Uno começar" antes do bloco anterior. Dessa forma, quando o Arduino Uno começar, será tocada a nota C4 por 0,25 segundos, no buzzer conectado ao pino 9.



Etapa 07: Clique agora no botão "Carregar" para que os comandos sejam transferidos para a placa. Se tudo der certo você ouvirá um alerta sonoro por 0,25 segundos.

Etapa 08: Pense agora em como funciona um alarme, o som é emitido, tem-se alguns segundos sem o som e, então, ele volta a ser emitido, repetidas vezes. Para programar esse efeito você deve inserir uma estrutura de repetição, para que o som não fique sendo tocado para sempre, utilize o bloco "repetir (10)". Com esse bloco, os comandos que estão no seu interior serão executados 10 vezes seguidas. Não esqueça que após soar o alarme você tem que deixar um tempinho de silêncio, para isso adicione um bloco para esperar 0,25 segundos. Ao carregar para o Arduino você deverá ouvir um alarme com 10 alertas sonoros.



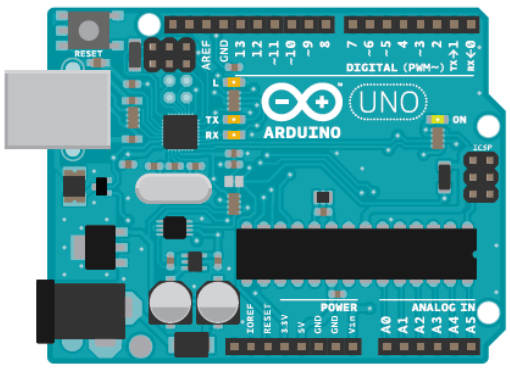
AGORA É A SUA VEZ

Experimente mudar as notas musicais, a duração da execução dessas e o tempo de espera, observando como o perfil do alarme muda com essas alterações.

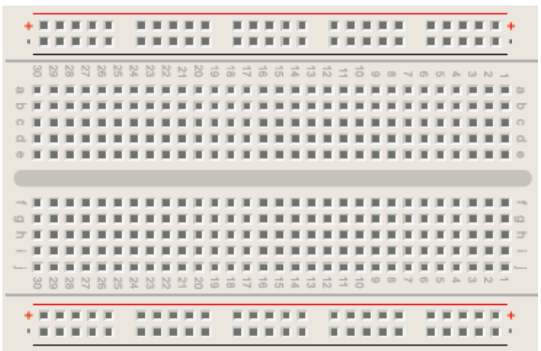
Será que o buzzer poderia ser utilizado nos semáforos para que as pessoas com deficiência visual ou baixa visão soubessem o momento correto de atravessar a faixa de pedestres? Recrie o projeto do semáforo, adicionando agora um buzzer para emitir um alerta sonoro sempre que o sinal estiver vermelho.

Não funcionou? Confira novamente todas as ligações, além disso, confira o valor do pino digital no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar".

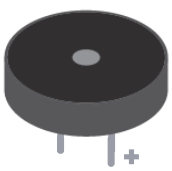
09. MELODIAS COM O BUZZER



Arduino + Cabo USB (x1)



Placa de testes (x1)



Buzzer (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

COMPONENTES

SOLTA O SOM

AUMENTA O SOM! CHEGOU A HORA DE VOCÊ PRODUZIR MELODIAS, UTILIZANDO O ARDUINO E, QUEM SABE, TOCAR A SUA MÚSICA FAVORITA.

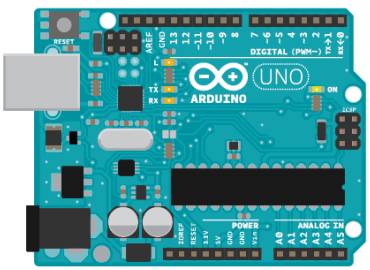
> Descobertas: programação sequencial, gravação permanente na placa Arduino, teoria musical básica.

> Tempo: **60 MINUTOS**
>
> Dificuldade: ■■■□□

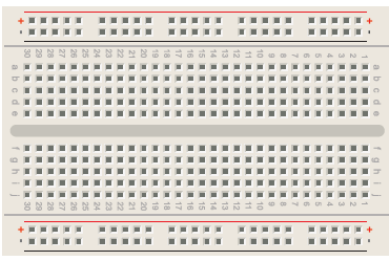
Sabe aquela música que você gosta? Você poderá tocar ela utilizando o buzzer conectado ao Arduino Uno, para isso deverá aprender um pouco sobre as notas musicais e como ler partituras musicais.

MONTANDO O CIRCUITO

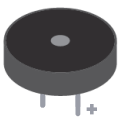
Etapa 01: Separe os componentes necessários para a montagem do circuito.



Arduino + Cabo USB (x1)



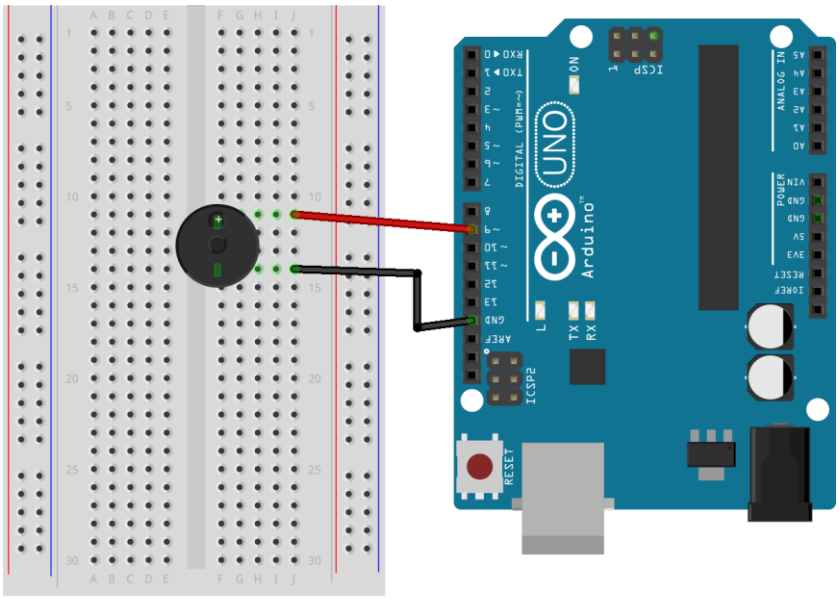
Placa de testes (x1)



Buzzer (x1)

Jumpers:
- Vermelho (x1)
- Preto (x1)

Etapa 02: Monte o circuito do Buzzer, conectado ao pino 9 do Arduino Uno, igual ao circuito do projeto anterior.



Etapa 03: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

- Jumper vermelho saindo do pino 9 do Arduino e conectado à linha horizontal junto na qual está conectado o pino positivo (maior) do buzzer. A linha na qual está conectado o pino menor do buzzer está conectada, com o **jumper preto**, ao pino GND do Arduino UNO.

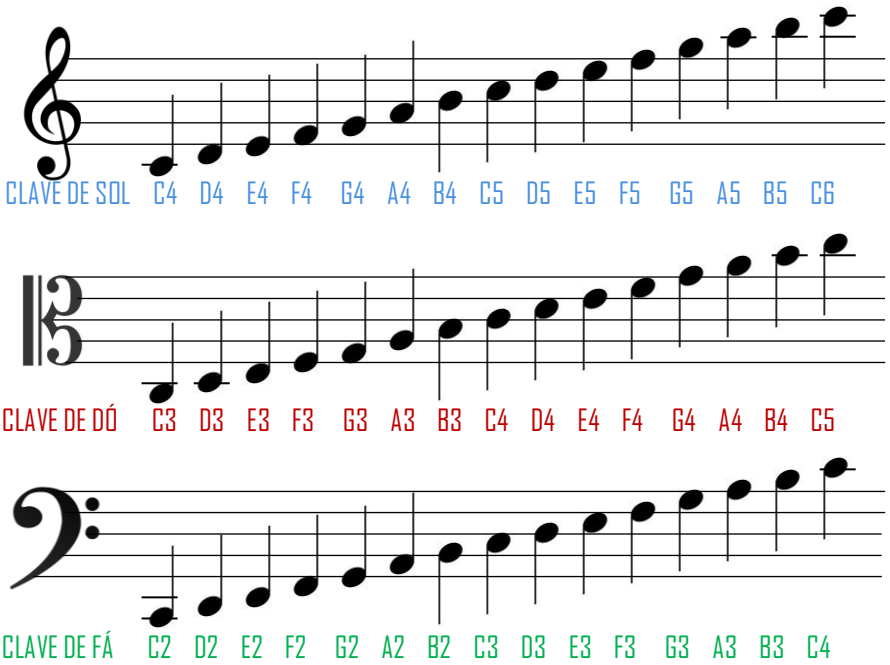
Etapa 04: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB e inicie a conexão com o mBlock, no modo "Carregar".

CRIANDO MELODIAS

Etapa 05: Como o buzzer reproduz notas musicais, você pode, naturalmente, utilizá-lo para reproduzir melodias de músicas. Para fazer isso você vai precisar da partitura da música. Para facilitar as coisas as músicas devem ter apenas uma clave e tocar uma nota por vez, para que o buzzer e Arduino consigam executar essa música. Para identificar as notas e as durações você deve utilizar o guia a seguir:

NOTAS MUSICAIS

Nota Musical:	Lá	Si	Dó	Ré	Mi	Fá	Sol
Cifra:	A	B	C	D	E	F	G



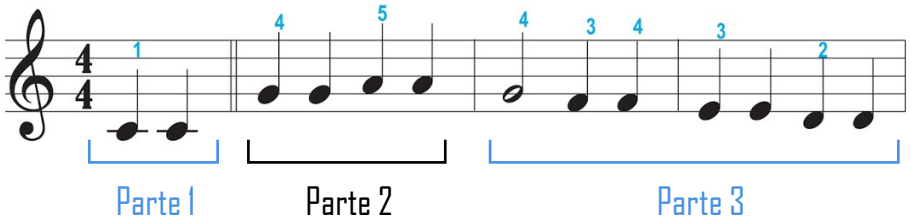
TEMPO

NOME	SEMIBREVE	MÍNIMA	SEMIMÍNIMA	COLCHEIA	SEMICOLCHEIA	FUSA	SEMIFUSA
SÍMBOLO							
DURAÇÃO	1	1/2	1/4	1/8	1/16	1/32	1/64
SEGUNDOS	2	1	0,5	0,25	0,125	0,0625	0,03125

ENTENDENDO PARTITURAS

Etapa 06: Utilizando os guias da página anterior, analise a partitura abaixo e liste as notas musicais e o tempo de cada nota no seu caderno.

Partitura: Brilha Bilha Estrelinha (Parte 01)



Música: Brilha Brilha Estrelinha (parte 1)

Parte 1
C4 por 0,5 segundos
C4 por 0,5 segundos

Parte 2
G4 por 0,5 segundos
G4 por 0,5 segundos
A4 por 0,5 segundos
A4 por 0,5 segundos

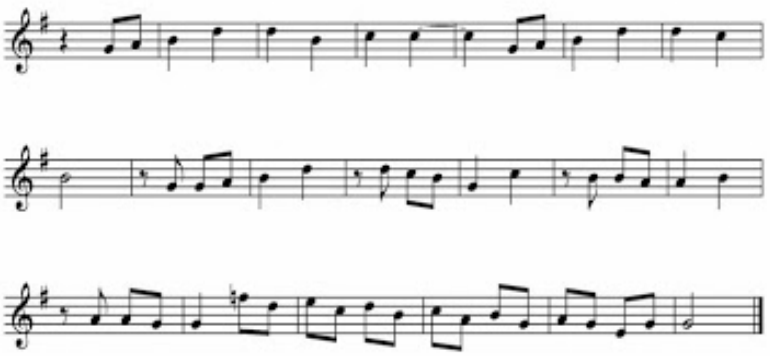
Parte 3
G4 por 1 segundo
F4 por 0,5 segundos
F4 por 0,5 segundos
E4 por 0,5 segundos
E4 por 0,5 segundos
D4 por 0,5 segundos
D4 por 0,5 segundos

CRIANDO O PROGRAMA

Etapa 07: Agora que você já conhece as notas e os tempos, crie um programa no mBlock com todas as notas, na sequência da partitura. Carregue para o Arduino Uno e confira que o Buzzer reproduz a música.

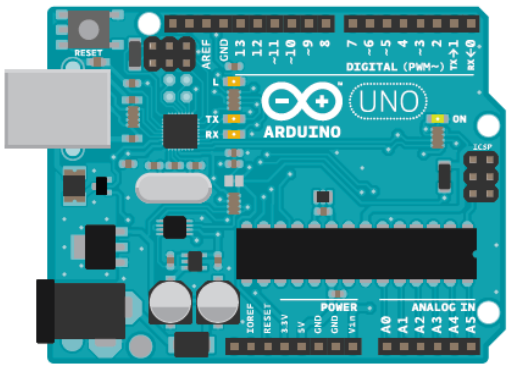
AGORA É A SUA VEZ

Pesquisa a partitura de alguma música que você gosta e crie um programa para tocar essa melodia com o Buzzer. Uma opção é a música Asa Branca de Luiz Gonzaga, que possui a partitura abaixo.



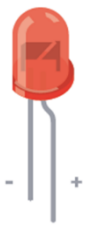
Não funcionou? Confira novamente todas as ligações, além disso, confira o valor do pino digital no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar".

10. DETECTANDO OBSTÁCULOS

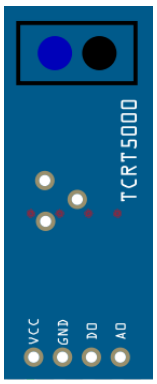


Arduino + Cabo USB (x1)

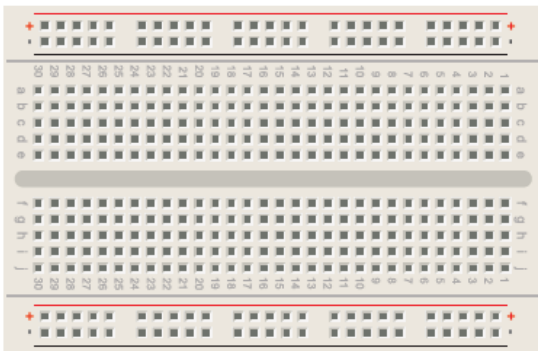
Resistor (x1)



LED vermelho (x1)



Sensor de Obstáculo (x1)



Placa de testes (x1)

Jumpers:
- Vermelho (x2)
- Preto (x3)
- Verde (x1)

COMPONENTES

PERCEBENDO O MUNDO

CHEGOU A HORA DO ARDUINO COMEÇAR A PERCEBER O MUNDO, NESSE PROJETO VOCÊ VAI DESENVOLVER UM DETECTOR DE OBSTÁCULO.

Descobertas: uso de sensores, leitura de sinal em pinos digitais, condicional dupla.

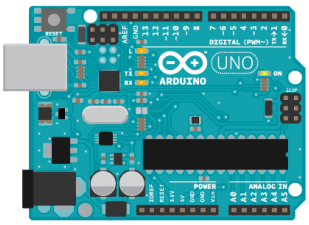
Tempo: **45 MINUTOS**

Dificuldade:

Nos projetos anteriores você utilizou o Arduino para controlar LEDs e Buzzer, nesses projetos as portas digitais foram sempre utilizadas no modo de "saída", nesse projeto, ao utilizar um sensor de proximidade, você vai aprender como funciona a "entrada" de dados em pinos digitais.

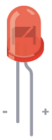
MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

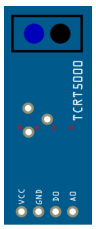


Arduino + Cabo USB (x1)

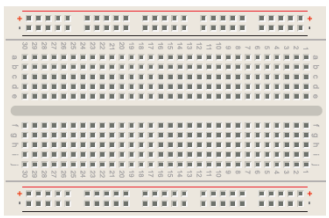
Resistor (x1)



LED vermelho (x1)



Sensor de Obstáculo (x1)

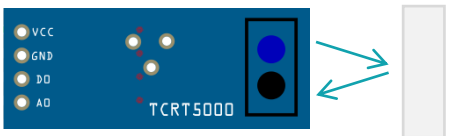


Placa de testes (x1)

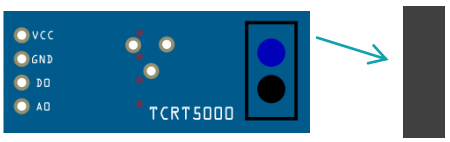
Jumpers:
- Vermelho (x2)
- Preto (x3)
- Verde (x1)

SENSOR DE OBSTÁCULO

O Sensor de Obstáculo IR funciona baseado na reflexão do sinal emitido por um LED infravermelho e detectado por um sensor. Dessa forma, quando não há nenhum obstáculo na frente do sensor, ou há um obstáculo de cor preta, que absorve a radiação infravermelha emitida, o sinal de saída no pino DO será alto, ou seja, 5V. Por outro lado, ao se aproximar de uma superfície, principalmente superfícies brancas, o sinal emitido pelo LED é refletido e o sensor libera um sinal baixo (0V) no pino DO.

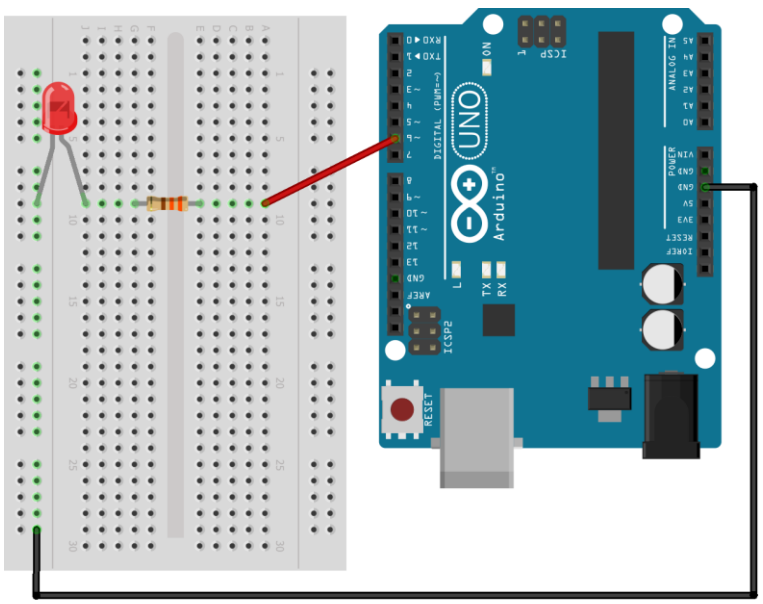


obstáculo detectado

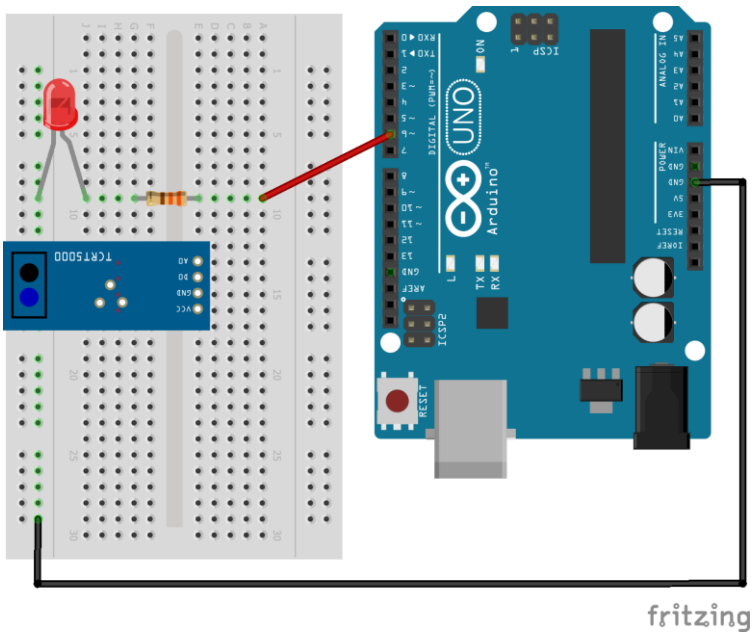


obstáculo não detectado

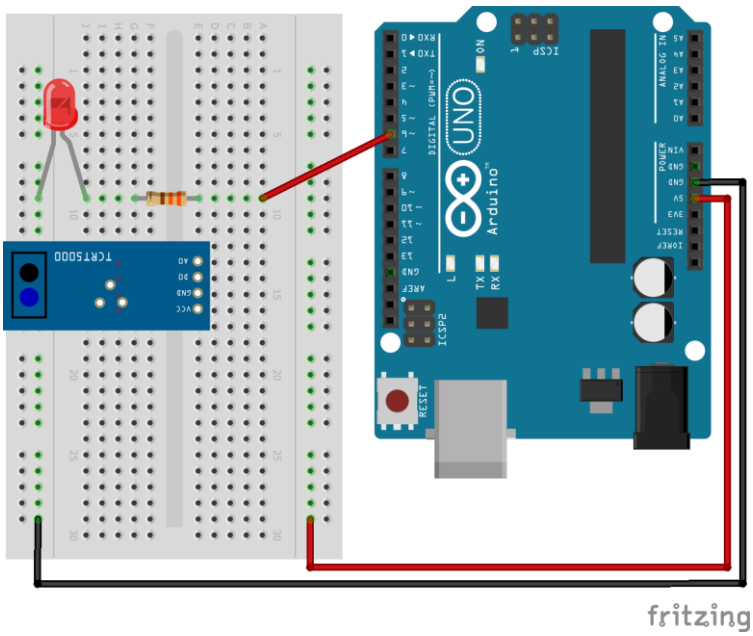
Etapa 02: Antes de começar a montagem do circuito do sensor de obstáculos, monte um circuito para acionamento de um LED vermelho, no pino 6 do Arduino.



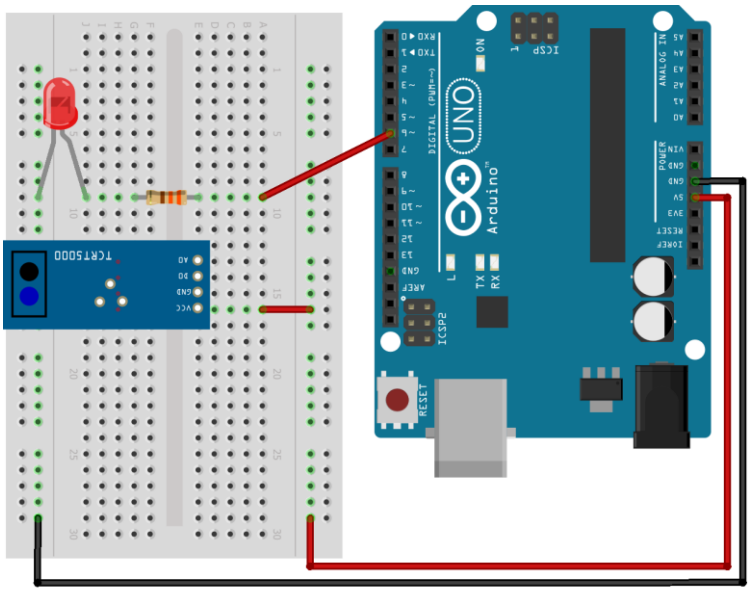
Etapa 03: Agora conecte os três pinos do sensor em três linhas paralelas da protoboard.



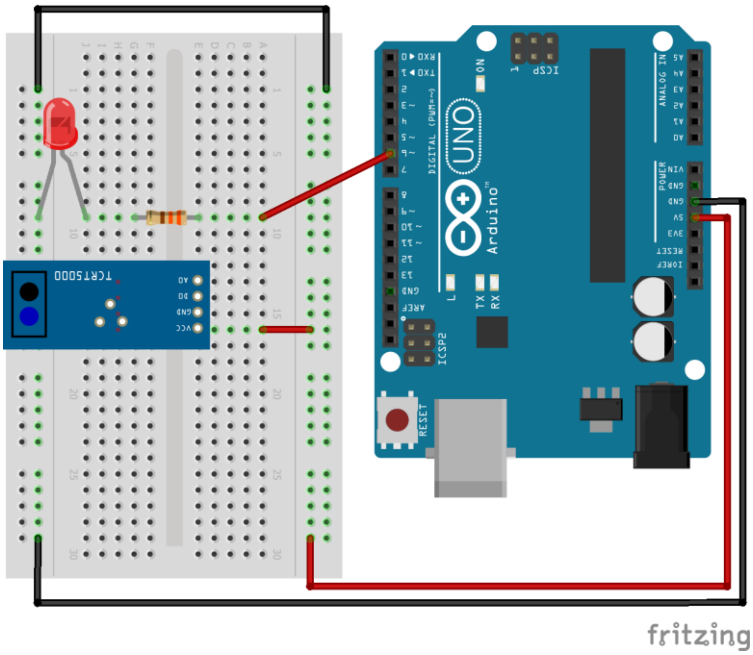
Etapa 04: As conexões dos pinos do Sensor de Obstáculo são VCC para 5V do Arduino, GND para o GND do Arduino e VCC para o pino 12 do Arduino. Para começar, adicione um jumper vermelho saindo do pino 5V do Arduino para a coluna vertical da placa de testes.



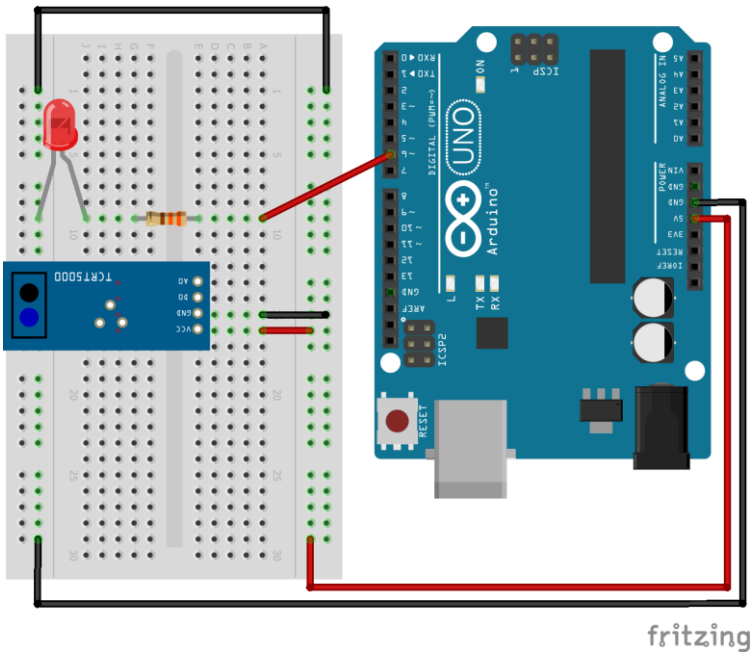
Etapa 05: Com outro jumper vermelho, conecte a linha na qual está o pino VCC do Sensor de Obstáculo na mesma coluna do outro jumper vermelho ligado aos 5V da placa Arduino.



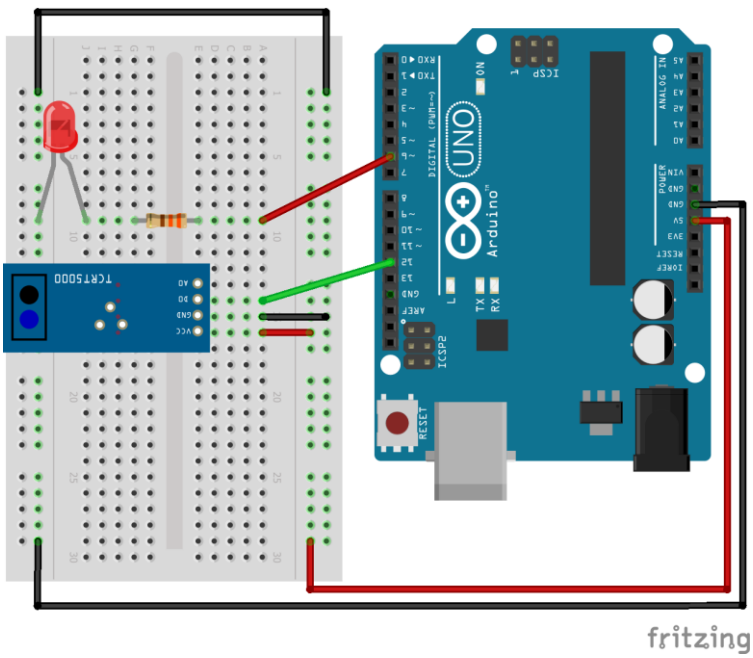
Etapa 06: Para você ligar o pino GND, perceba que você já possui um **jumper preto**, ligado ao pino GND do Arduino, você pode aproveitar essa conexão. Para isso, adicione outro jumper preto, da coluna com o outro **jumper preto**, até a coluna do outro lado da placa, ficando as duas colunas como GND.



Etapa 07: Com outro **jumper preto**, conecte a linha na qual está o pino GND do Sensor de Obstáculo na mesma coluna do outro **jumper preto** ligado ao GND da placa Arduino.



Etapa 08: Por fim, utilizando um jumper verde, conecte o pino 12 do Arduino na mesma coluna do pino DO, do Sensor.



Etapa 10: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

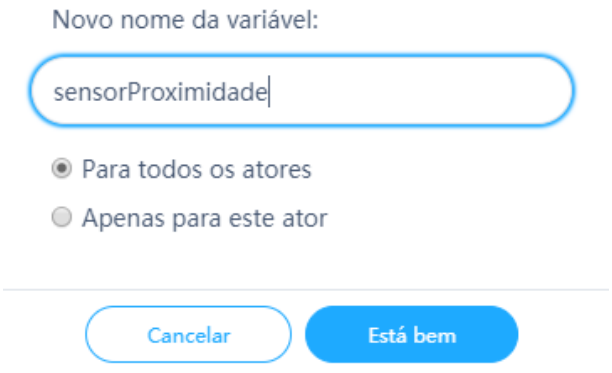
- Jumper vermelho saindo do pino 9 do Arduino e conectado à linha horizontal junto na qual está conectado em um resistor, de 33 ohms, fazendo a ponte para o pino positivo (maior) do LED vermelho, com a perna menor conectada em uma coluna que está conectada, com o **jumper preto**, ao pino GND do Arduino UNO.
- Sensor de proximidade com o pino VCC conectado, por um jumper vermelho, em uma coluna conectada, por outro jumper vermelho no pino, 5V do Arduino. Pino GND conectado, por um **jumper preto**, na coluna conectada, por outro **jumper preto**, no pino GND do Arduino. Por fim, pino OUT conectado, por um jumper verde, no pino 12 do Arduino.

Etapa 11: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB, e inicie a conexão com o mBlock, no modo "Viver".

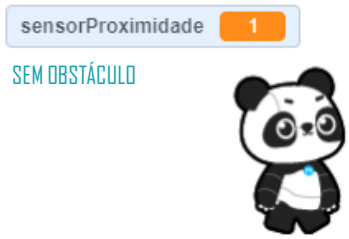
É de se esperar que, ao conectar o Arduino ao computador, o LED da placa do sensor se acenda. Geralmente a placa possui dois LEDs, um indica a alimentação da placa e o outro a detecção de obstáculos. Tente passar a mão sobre os LEDs do sensor e observe se um dos LEDs da placa se acende, ou apaga.

CRIANDO O PROGRAMA

Etapa 12: Inicie a programação criando um variável chamada "sensorProximidade".



Etapa 13: Em seguida, arraste para a área de programação o bloco que define o valor da variável "sensorProximidade". Arraste também um bloco para repetir o código para sempre e coloque o bloco anterior dentro da repetição. Perceba que, até esse momento o valor da variável é zero e não se altera com obstáculos na frente do sensor.



Etapa 14: Para que você consiga observar o valor da variável "sensorProximidade" mudando de acordo com o sinal do sensor, você deve definir o valor dessa variável como sendo o valor lido pelo sensor. Para fazer isso, no grupo "Pin", arraste o bloco "ler pino digital (9)" para dentro do bloco que define o valor da variável. Lembre de mudar o pino digital para 12, que é o pino conectado a saída, DO, do sensor. Teste o código passando a mão sobre os LEDs do sensor e observando a alteração no valor da variável "sensorProximidade".

Perceba que, nesse projeto, ao contrário dos anteriores, você está trabalhando com a entrada de dados no pino digital. Analisando o funcionamento do programa você pode perceber que:

- Quando o sensor não detecta nenhum objeto o valor que é recebido pelo Arduino no pino digital 12 é igual a 1;
- Já quando algum obstáculo passa próximo ao sensor, refletindo o sinal emitido pelo LED, o valor recebido pelo Arduino, no pino 12, é igual a 0.

Esse exemplo define muito bem como é a entrada de dados nos pinos digitais, da mesma forma que a saída, os pinos digitais podem receber apenas sinais nos estados alto, 5V, e baixo, 0V ou GND.

Etapa 15: Agora, você deve usar o valor da variável "sensorProximidade" que é igual ao valor emitido pelo sensor, para ligar ou desligar um LED quando algum obstáculo se aproxima, seguindo a lógica abaixo:

```
Se tiver obstáculo
        Ligar o LED
Senão tiver obstáculo
        Desligar o LED
```

Para saber se há, ou não, algum obstáculo o Arduino utilizará o sinal obtido pelo sensor de obstáculo, dessa forma podemos reescrever a lógica acima como:

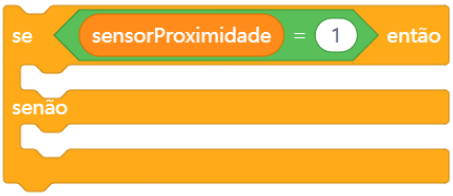
```
Se sensorProximidade = 0
        Ligar o LED
Senão
        Desligar o LED
```

Dessa forma, se o valor do sensor for igual a zero, quando é detectado algum objeto, o LED deverá ser ligado, caso o valor seja igual a 1, o LED deverá ser desligado.

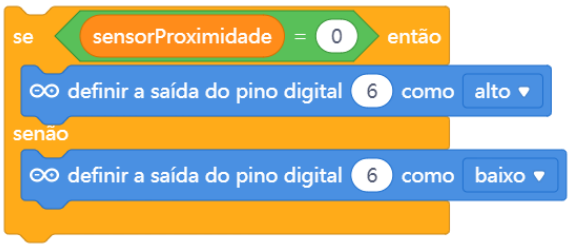
Etapa 16: Como você já sabe, para realizar projetos que envolvam comparações entre valores devemos utilizar operadores e para tomar decisões utilizamos as condicionais. Para começar, crie a estrutura do "se" e "senão". Nesse caso, como você precisará de duas condicionais, "se o tiver obstáculo" e "senão tiver obstáculo", arraste o bloco "se < > então – senão", presente no grupo "Controle", para a área de programação.

Como você tem que fazer o Arduino compreender quando há, ou não, obstáculo. Para fazer isso utilize o bloco de igualdade, que pode ser encontrado no grupo "operadores".

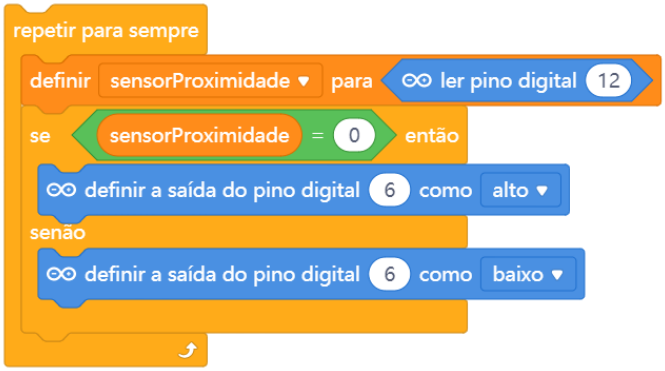
Etapa 17: Arraste esse bloco para dentro do bloco, se senão, após o se. Como argumentos, você quer que os comandos dentro do SE sejam executados apenas se o valor de "sensorProximidade" for igual a 1, então coloque antes da igualdade a variável e depois o valor 1.



Etapa 18: Arraste agora o bloco responsável por ligar o LED no pino 6 do Arduino e coloque dentro do bloco SE. Dessa forma, se o sensor de proximidade for igual a 1 então ligue o LED. Senão for igual a um o Arduino deve desligar o LED, assim, arraste outro bloco, esse para desligar o LED no pino 6 do Arduino.



Etapa 19: Para finalizar, arraste todos os comandos para dentro do repetir para sempre, abaixo do comando que define a variável e teste se, ao aproximar um obstáculo do sensor, o LED acende, e ao afastar o obstáculo, o LED apaga.

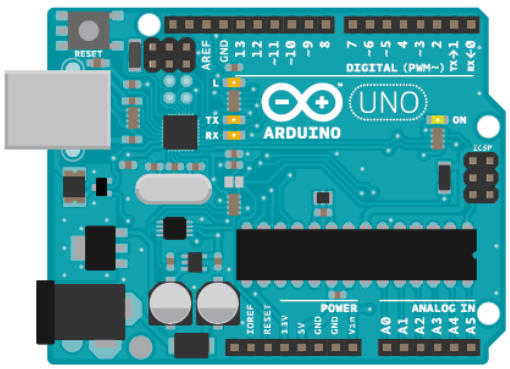


AGORA É A SUA VEZ

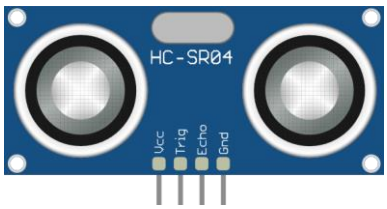
Que tal substituir o LED por um buzzer e fazer um alarme de proximidade?, utilize sua criatividade para experimentar outros componentes que foram trabalhados nos projetos anteriores.

Não funcionou? Confira novamente todas as ligações, além disso, confira o valor do pino digital no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar".

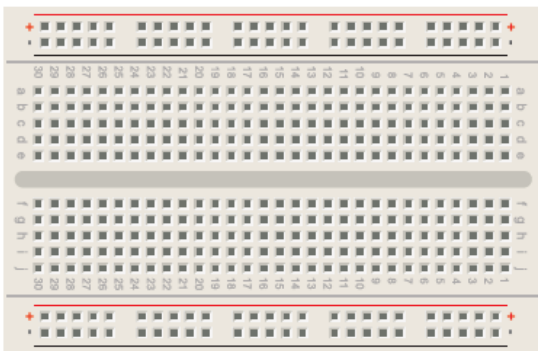
11. SENSOR ULTRASSÔNICO DE DISTÂNCIA



Arduino + Cabo USB (x1)



Sensor Ultrassônico de Distância (HCSR04) (x1)



Placa de testes (x1)

Jumpers:

- Vermelho (x2)
- Preto (x2)
- Verde (x1)
- Amarelo (x1)

COMPONENTES

ESTOU PERTO OU LONGE?

NO ÚLTIMO PROJETO VOCÊ VIU COMO DETECTAR A PRESENÇA DE UM OBSTÁCULO, CHEGOU A HORA DE CALCULAR A DISTÂNCIA ATÉ O OBSTÁCULO.

Descobertas: uso de um sensor ultrassônico, comunicação serial, uso de extensões.

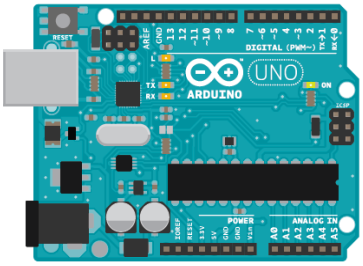
Tempo: **60 MINUTOS**

Dificuldade:

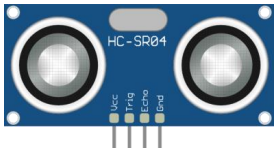
No projeto anterior você utilizou um sensor de obstáculo para detectar um obstáculo próximo ao sensor. Nesse projeto você utilizará um sensor de proximidade para medir a distância desse obstáculo até o sensor.

MONTANDO O CIRCUITO

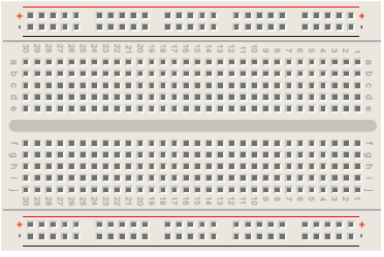
Etapa 01: Separe os componentes necessários para a montagem do circuito.



Arduino + Cabo USB (x1)



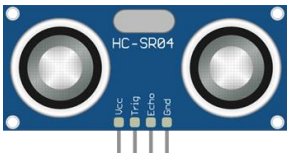
Sensor Ultrassônico de Distância (HCSR04) (x1)



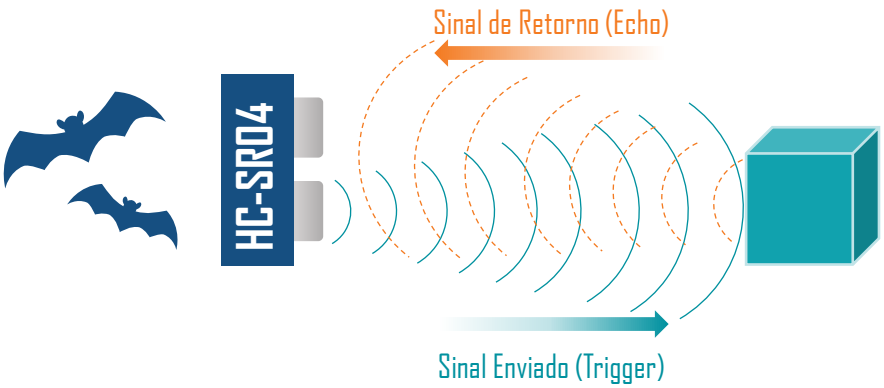
Placa de testes (x1)

Jumpers:
Vermelho (x1)
Preto (x1)
Verde (x1)
Amarelo (x1)

SENSOR DE DISTÂNCIA



O Sensor de Ultrassônico HC-SR04 é um sensor que permite medir distâncias entre 2cm e 4 metros, com uma precisão de 3 mm. A medida de distância pode ser utilizada em vários projetos como acionar um alarme, abrir uma porta automaticamente, fazer um robô desviar de um obstáculo, fazer uma trena eletrônica etc. O funcionamento do sensor de distância se baseia no uso envio de sinais ultrassônicos, como os emitidos por morcegos, o sensor aguarda o retorno do sinal e, com base no tempo entre envio e retorno, calcula a distância entre o sensor e o objeto detectado.

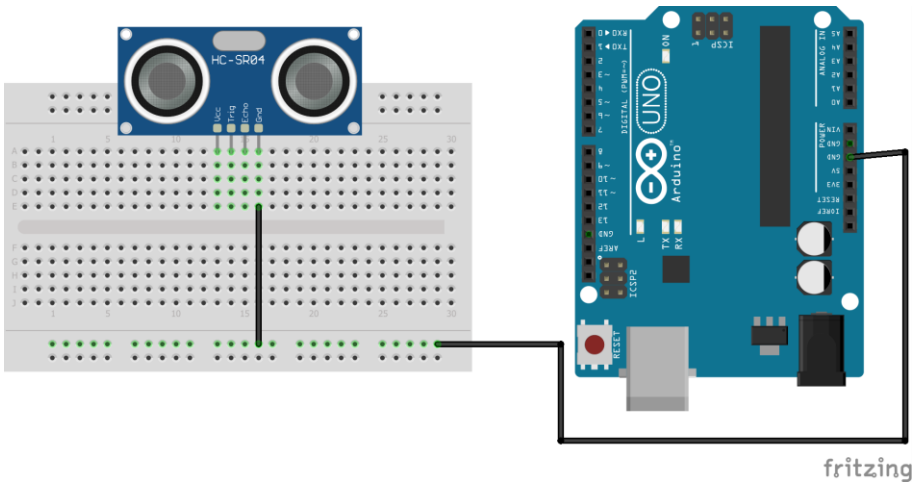


De forma mais detalhada, o sensor envia uma onda ultrassônica na forma de pulsos e aguarda o tempo em que esses pulsos serão detectados pelo sensor ultrassônico que fica do lado do emissor. Para determinar a distância entre o objeto e o sensor utiliza-se a equação:

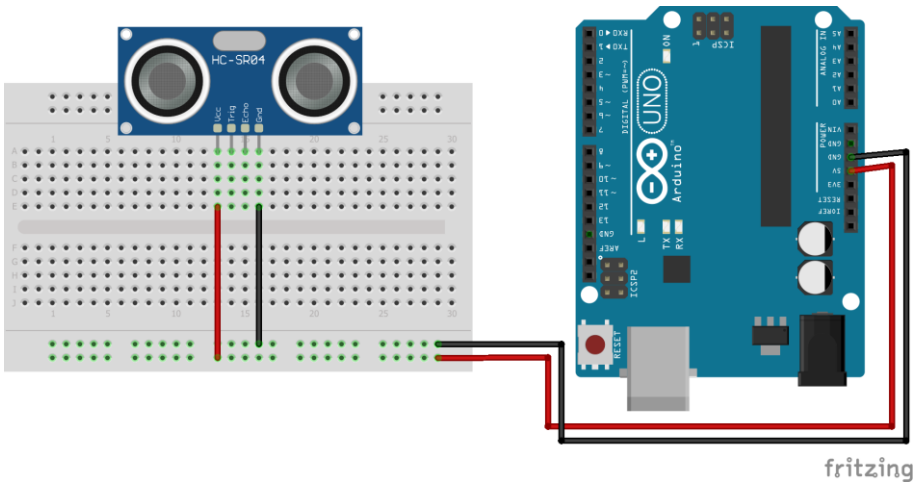
$$Distância = \frac{Tempo\ do\ Echo\ * Velocidade\ do\ Som}{2}$$

De acordo com essa equação a distância será calculada de acordo com o tempo do Echo, ou seja, o tempo até que o sinal enviado seja refletido pelo objeto e retorne, multiplicado pela velocidade do som, que é a velocidade das ondas ultrassônicas que serão emitidas, dividido por 2. Essa divisão por dois acontece porque o sinal faz o caminho até o obstáculo, reflete e retorna até o sensor, dessa forma esse sinal percorreu a distância até o objeto duas vezes, na ida e na volta.

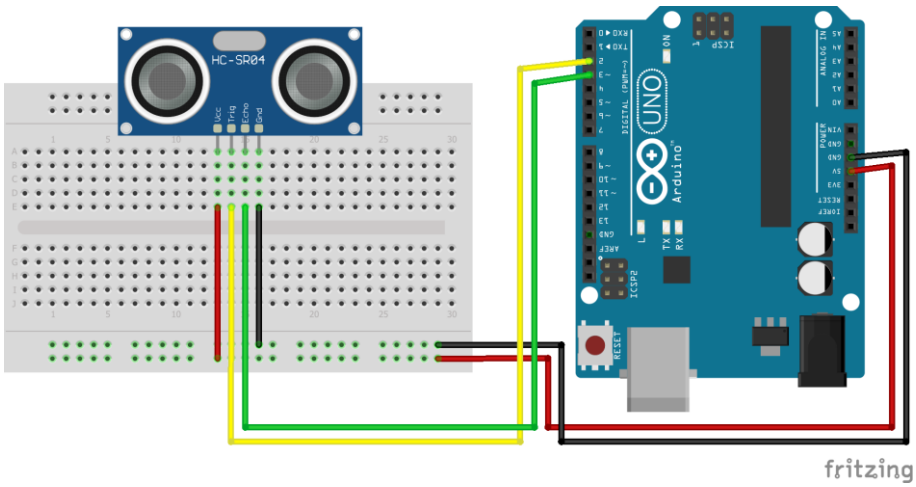
Etapa 02: Para conectar o sensor de distância ao Arduino você deve, inicialmente, conectar o sensor na placa protoboard, em quatro linhas horizontais paralelas. Em seguida conecte o pino GND do sensor ao pino GND do Arduino, utilizando um **jumper preto**, conectado à uma coluna da protoboard e, utilizando outro **jumper preto**, conecte essa mesma coluna ao GND do Arduino.



Etapa 03: Em seguida, conecte o pino VCC do sensor ao pino de 5V do Arduino. Para fazer isso utilize-se de um jumper vermelho, conectando a linha do pino VCC à outra coluna da protoboard e, com outro jumper vermelho, conecte essa coluna no Arduino.



Etapa 04: Chegou a hora dos pinos Trigger, que emite as ondas ultrassônicas, e Echo, que recebe o sinal refletido pelo obstáculo. Comece fazendo a ligação entre o pino Echo, do sensor, com o pino 3 do Arduino, utilizando um jumper verde. Para finalizar, ligue o pino Trig do sensor com o pino 2 do Arduino, utilizando um jumper amarelo.



Etapa 05: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito do sensor de distância:

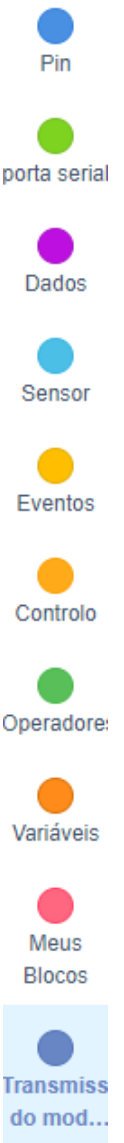
- Pino VCC conectado, por um jumper vermelho, em uma coluna conectada, por outro jumper vermelho no pino 5V do Arduino.
- Pino GND conectado, por um **jumper preto**, na coluna conectada, por outro **jumper preto**, no pino GND do Arduino.
- Pino Echo conectado, por um jumper verde, ao pino 3 do Arduino.
- Pino Trig conectado, por um jumper amarelo, ao pino 2 do Arduino.

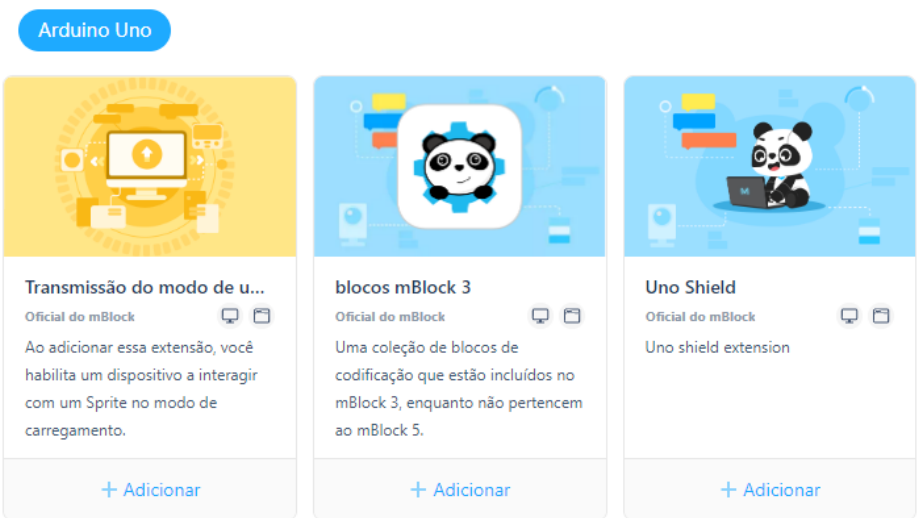
Etapa 06: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB, e inicie a conexão com o mBlock, no modo "Carregar".

UTILIZANDO EXTENSÕES

Para criar o programa do sensor de distância você vai precisar utilizar uma Extensão para ler os valores de distância medidos no próprio mBlock. Extensões, no mBlock, são pacotes de blocos já criados por outros programadores. As extensões permitem o uso de outros dispositivos, além daqueles os quais o mBlock já possui os blocos.

Para utilizar uma nova extensão você deve, com o dispositivo Arduino Uno selecionado, clicar no ícone "extensão", presente na parte inferior da lista dos blocos. Em seguida abre-se uma janela com várias extensões, para esse projeto procure a extensão "transmissão do modo de upload", conforme ilustrado na figura abaixo. Clique no ícone de (+) em seguida em Adicionar. Para realizar esse processo é necessário conexão com a internet.





Ao adicionar a extensão ao seu projeto você perceberá que um novo grupo de blocos, com o nome da extensão, "transmissão do modo de upload", foi adicionado no conjunto de comandos. Dessa forma, temos novos blocos disponíveis para nossa programação.

Não funcionou? Caso os novos blocos não tenha aparecido, confira novamente se a extensão foi adicionada, conforme descrito no parágrafo anterior.

CRIANDO O PROGRAMA

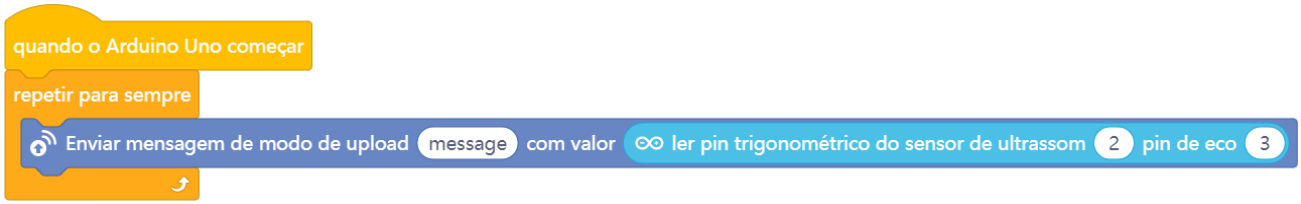
Etapa 07: Para criar o programa você deve utilizar uma função para ler o sensor de distância, essa função se encontra no grupo de comandos "Sensor", a função "ler pin trigonométrico do sensor de ultrassom (1) pin de eco (1). Arraste esse bloco para a área de programação e mude os valor dos pinos para 2 e 3, conforme o circuito montado.

COMUNICAÇÃO SERIAL

Etapa 08: Nesse projeto você precisará que o Arduino leia a distância medida pelo sensor e nos avise a distância medida. O bloco anterior já realiza a medida da distância, porém, precisamos nos comunicar com o Arduino, ele precisa nos "dizer" qual o valor que ele leu, para fazer isso você deve utilizar um bloco da extensão "transmissão do modo de upload" que envia uma mensagem do Arduino para o mBlock. O bloco "Enviar mensagem de modo de upload (message) com valor (1)". Com esse bloco o Arduino envia uma mensagem, chamada "message" com o valor 1 para o mBlock.

Etapa 09: Agora você pode juntar os dois blocos para que o Arduino envie uma mensagem para o mBlock com o valor igual ao valor lido pelo pino do sensor.

Etapa 10: Dessa forma, o Arduino já envia a mensagem com o valor lido no sensor de distância, porém, envia apenas uma vez. Para fazer com que ele fique lendo e enviando o valor de distância repetidas vezes, insira um bloco de repetição e o bloco para que o código seja executado quando o Arduino começar.



UTILIZANDO ATORES

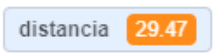
Etapa 11: Até esse momento o Arduino já está lendo a distância medida pelo sensor e enviando para o mBlock, porém o mBlock ainda não consegue ler a mensagem recebida. Para fazer isso, precisaremos utilizar os atores do mBlock, nesse caso o Panda. Na aba Atores, selecione o ator Panda, perceba que os blocos disponíveis mudaram, já que agora esses blocos estão relacionados ao ator e não ao Arduino.

Para que o ator entenda a mensagem enviada pelo Arduino, será necessário utilizar a mesma extensão utilizada anteriormente, dessa vez para o ator. Siga o mesmo procedimento anterior, clique no botão "extensão" e adicione a extensão "transmissão do modo de upload".

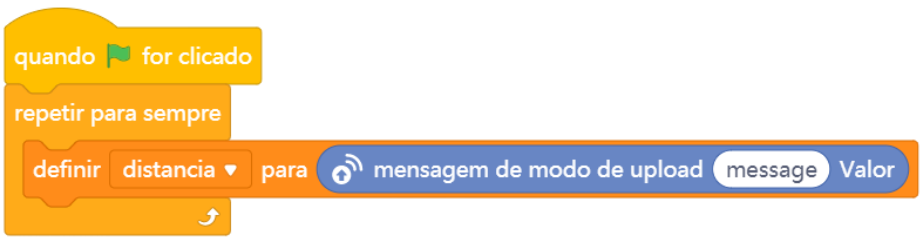
Etapa 12: Para apresentar o valor da distância, recebido na mensagem enviada pelo Arduino, crie uma variável chamada "distancia". Deixando selecionada a opção "Para todos os atores".

Etapa 13: Agora, você está trabalhando com dois códigos funcionando em paralelo. Um no Arduino, enviando os dados, e o do ator, recebendo esses dados. Para começar o código do ator, com o Panda selecionado, arraste o bloco de Evento, "quando a bandeira verde for clicado".

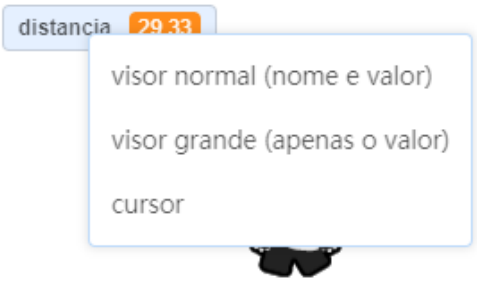
Etapa 14: Com o bloco anterior, sempre que você clicar no botão da bandeira verde, logo abaixo do panda, do lado esquerdo, o código será executado. Nesse código, sua intensão é que o valor da variável "distancia" seja definido como sendo o valor recebido do Arduino. O valor dessa variável sempre deve ser atualizado com o valor recebido, para tanto, você pode utilizar o bloco "repetir para sempre".



Etapa 15: Para definir qual o valor recebido na mensagem enviada pelo Arduino, basta utilizar o bloco "mensagem de modo de upload (message) Valor". Esse bloco é responsável por ler o valor recebido na mensagem. Agora, basta apertar o botão com a bandeira verde e observar o valor da variável distância, ao lado do Panda. Esse valor deve estar variando de acordo com a distância medida, em centímetros.



Caso esteja muito pequeno o quadrado com o valor, você pode clicar com o segundo botão do mouse, o botão da direita, e escolher a opção "visor grande (apenas o valor)".



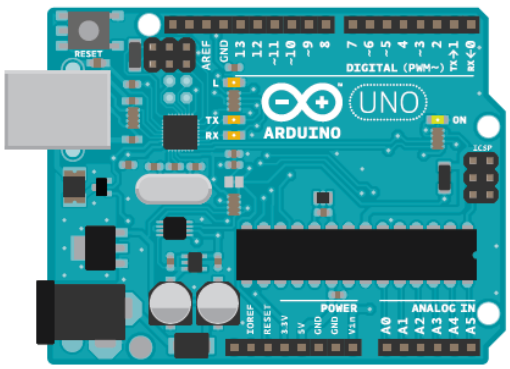
AGORA É A SUA VEZ

Que tal fazer um alerta de proximidade?, utilize 3 LEDs para montar um circuito que funcione da seguinte forma:

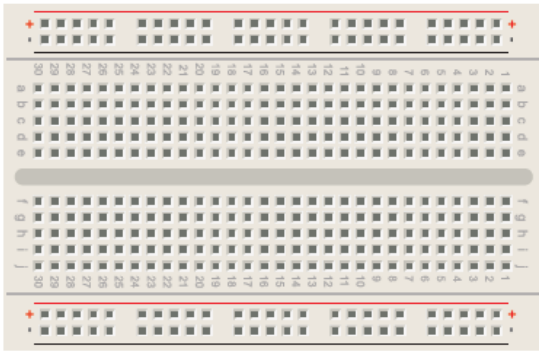
- Quando algum obstáculo está a 30 cm ou mais de distância do sensor nenhum LED será acesso;
- Quando estiver entre 20 e 30 cm de distância apenas um LED será acesso;
- Quando estiver entre 10 e 20 cm de distância dois LEDs serão acessos;
- Quando estiver a uma distância menor que 10 cm, os três LEDs serão acessos.

Não funcionou? Confira novamente todas as ligações, além disso, confira o valor dos pinos digitais no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar". Além disso, confira se as extensões foram incluídas no mBlock, tanto para o Arduino, como para o ator.

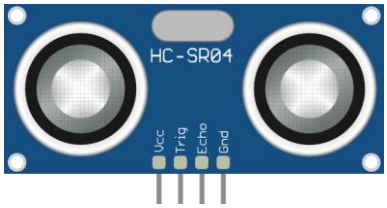
12. SENSOR DE ESTACIONAMENTO



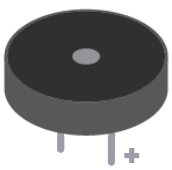
Arduino + Cabo USB (x1)



Placa de testes (x1)



Sensor Ultrassônico de Distância (HCSR04) (x1)



Buzzer (x1)

Jumpers:

- Vermelho (x2)
- Preto (x3)
- Azul (x1)
- Verde (x1)
- Amarelo (x1)

COMPONENTES

ME AJUDA A MANOBRAR O CARRO?

VOCÊ JÁ SABE COMO MEDIR A DISTÂNCIA E COMO CRIAR UM ALARME COM UM BUZZER, CHEGOU A HORA DE MONTAR UM SENSOR DE ESTACIONAMENTO.

Descobertas: uso de sensores, leitura de sinal em pinos digitais, condicionais.

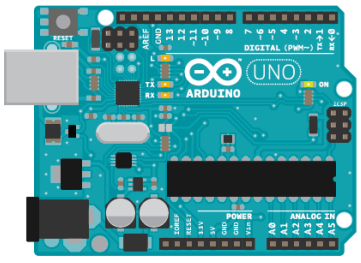
Tempo: **45 MINUTOS**

Dificuldade:

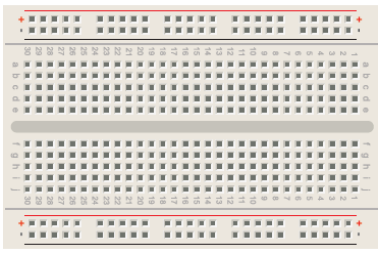
Nos projetos anteriores você utilizou o Arduino para acionar um buzzer e criar um alarme sonoro, além disso, com um sensor ultrassônico conseguiu medir a distância de um objeto até o sensor. Agora é a hora de você criar um sensor de estacionamento que aumenta a frequência do alarme, de acordo com a distância do obstáculo até o sensor.

MONTANDO O CIRCUITO

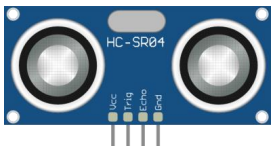
Etapa 01: Separe os componentes necessários para a montagem do circuito.



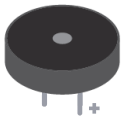
Arduino + Cabo USB (x1)



Placa de testes (x1)



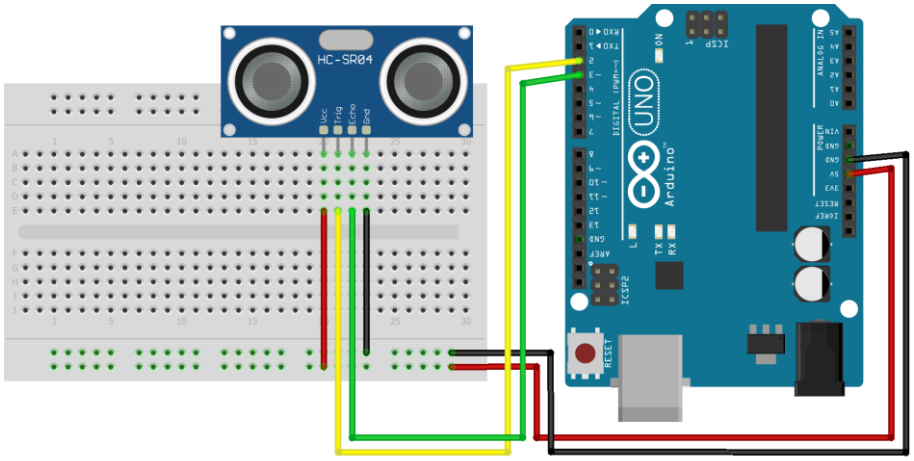
Sensor Ultrassônico de Distância (HCSR04) (x1)



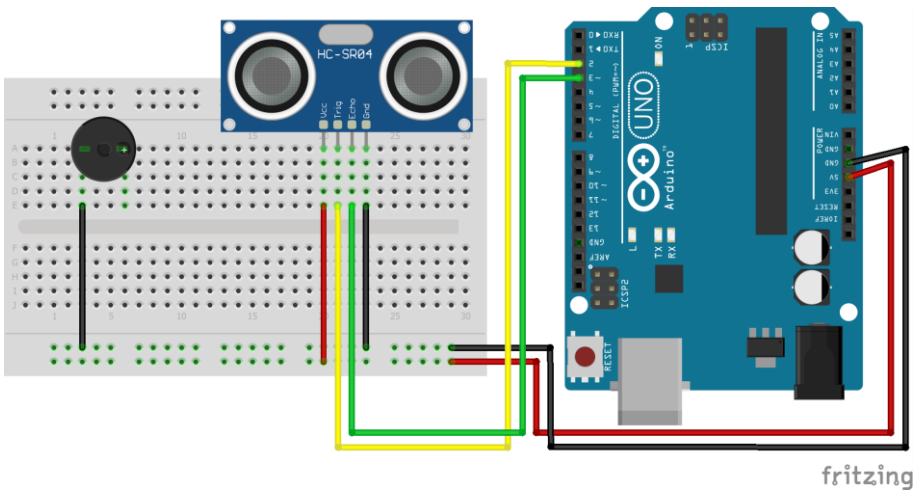
Buzzer (x1)

Jumpers:
- Vermelho (x2)
- Preto (x3)
- Azul (x1)
- Verde (x1)
- Amarelo (x1)

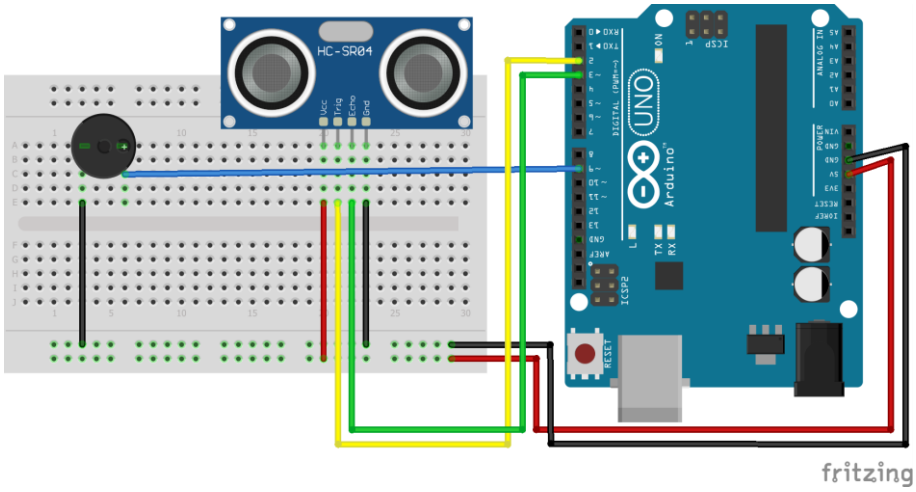
Etapa 02: Nesse projeto você deve, tal como no anterior, conectar o sensor ultrassônico de distância, HC-SR04, ao Arduino Uno, com os pinos Trigger e Echo nas portas 2 e 3, respectivamente, como na imagem abaixo.



Etapa 03: Em seguida, conecte o pino terra do buzzer, o pino menor, à coluna conectada ao pino GND do Arduino, utilizando um **jumper preto**.



Etapa 04: Por fim, conecte o pino de alimentação, o pino maior do buzzer, ao pino 9 do Arduino Uno, utilizando um jumper azul.



Etapa 05: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito:

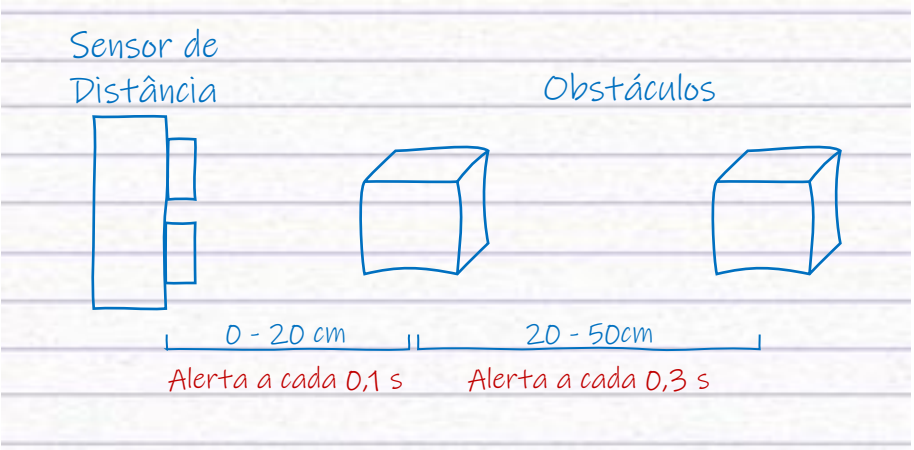
- Para o sensor de distância o pino VCC é conectado, por um jumper vermelho, em uma coluna conectada, por outro jumper vermelho no pino 5V do Arduino. Já o pino é conectado, por um **jumper preto**, na coluna conectada, por outro **jumper preto**, no pino GND do Arduino. Por fim, os pino Echo é conectado, por um jumper verde, ao pino 3 do Arduino e o pino Trig conectado, por um jumper amarelo, ao pino 2 do Arduino.

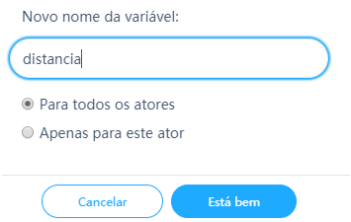
- Já o Buzzer tem seu pino menor, pino GND, conectado, por um **jumper preto**, à coluna conectada ao pino GND do Arduino, por outro **jumper preto**. O pino maior do Buzzer, pino VCC, por sua vez, é conectado, por um jumper azul, ao pino 9 do Arduino Uno.

Etapa 06: Caso esteja tudo correto, conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB, e inicie a conexão com o mBlock, no modo "Carregar".

CRIANDO O PROGRAMA

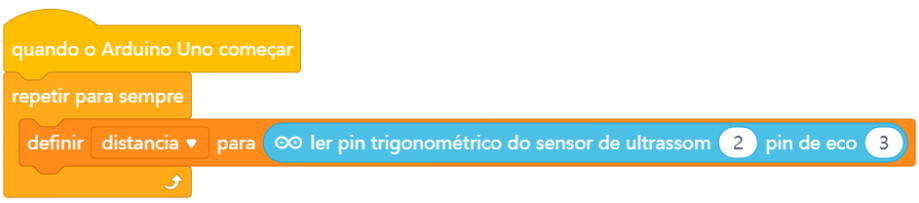
Antes de iniciar a programação pense em como o programa deve funcionar e faça um esboço em seu caderno das etapas envolvidas, considerando que, quando o sensor detectar um obstáculo à uma distância entre 20 e 30 centímetros, ele deve alarmar em intervalos de tempo de 0,3 segundos. Caso essa distância seja menor que 20 centímetros ele deve alarmar em intervalos de 0,1 segundo.





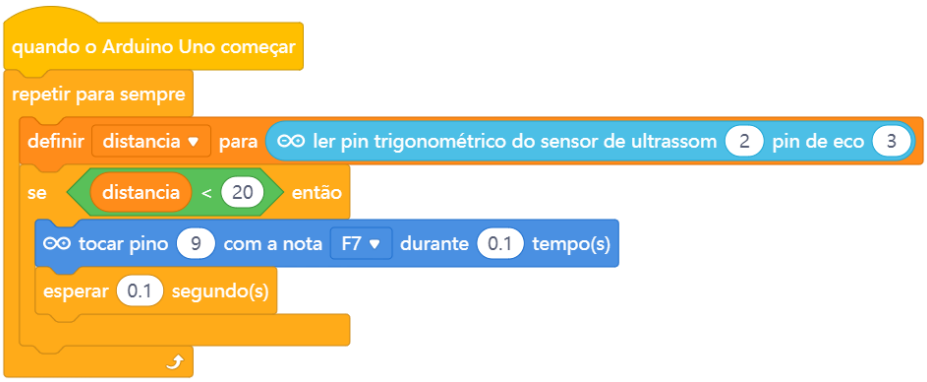
Etapa 07: Inicie o programa criando uma variável, chamada "distancia", essa variável deve ter seu valor definido como sendo o valor da distância medido pelo sensor ultrassônico, para tanto configure o bloco "ler pin trigonométrico do sensor de ultrassom (1) pin de eco (1), considerando as conexões nos pinos 2 e 3, conforme o circuito montado.

Etapa 08: Em seguida, arraste os blocos para que o programa seja iniciado quando o Arduino Uno começar e o bloco de repetição, e encaixe o bloco acima neles, para que o valor da variável distância seja atualizado continuamente.



Etapa 09: Dessa forma, o Arduino já consegue obter o valor da distância medida pelo sensor, agora temos que avaliar se a distância é menor que 20 cm. Para isso arraste um bloco de condicional "se". E como condição utilize um bloco "operador" para comparar se a variável "distância" é menor que 20.

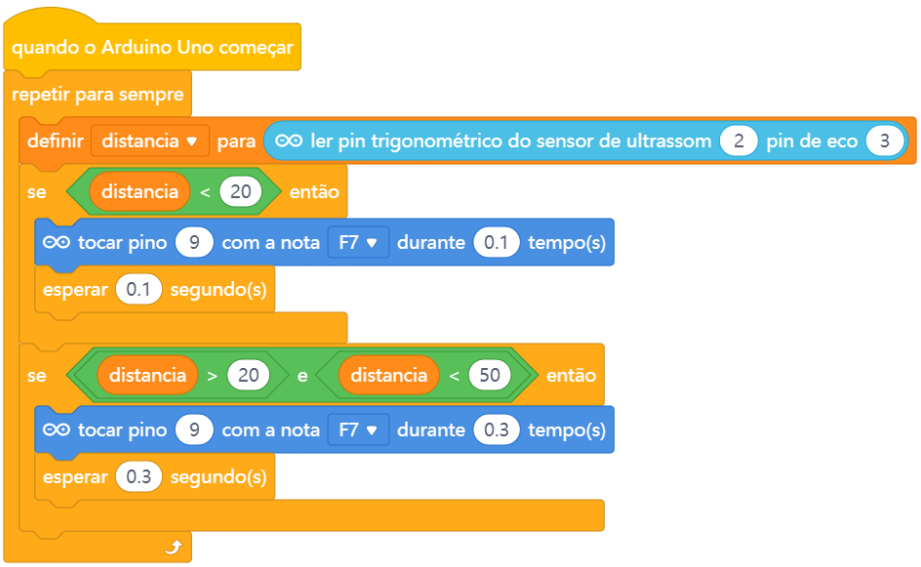
Etapa 10: Com essa condição seu programa já é capaz de identificar se a distância é menor que 20. Resta agora tocar o buzzer, conectado no pino 9, em intervalos de 0,1 segundos. Para isso coloque, dentro da condicional "se" o bloco para tocar o pino 9, com a nota F7 durante o tempo de 0,1 segundo. Em seguida adicione um bloco para esperar 0,1 segundo, para que o buzzer possa tocar como uma sirene, ligando e deligando a cada 0,1 segundo.



Carregue o programa para o Arduino Uno e observe se, ao aproximar um objeto a uma distância menor que 20 cm, o buzzer emite um alerta sonoro em intervalos de 0,1 segundo.

Etapa 11: Resta agora adicionar uma outra condicional, onde você criará o alerta quando a distância estiver entre 20 e 50 centímetros. Para fazer isso, o programa terá que analisar se a distância é maior que 20 e depois se a distância é menor que 50, já que se for maior que 50 ele não precisará emitir alerta sonoro. Dessa forma você deve arrastar o bloco "< > e < >", para a área de programação e colocar outros dois operadores, um maior que e outro menor que, para comparar os valores da distância.

Etapa 12: Por fim, adicione os blocos para o buzzer tocar durante 0,3 segundos e transfira o programa para o Arduino. O buzzer deve começar a alarmar quando um objeto estiver a menos que 50 cm do sensor e esse alarme deve ficar mais intenso quando a distância for menor que 20 cm.



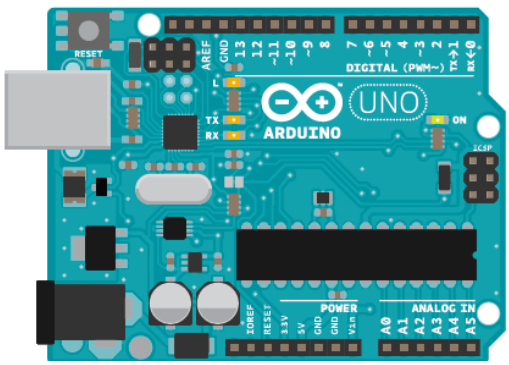
AGORA É A SUA VEZ

Implemente no seu alarme de distância outras escalas de distância, como sugerido abaixo:

- Quando algum obstáculo está a 50 cm ou mais de distância do sensor, nenhum alerta sonoro será emitido;
- Quando estiver entre 40 e 50 cm de distância o buzzer irá emitir som a cada 1 segundo;
- Quando estiver entre 30 e 40 cm de distância o buzzer irá emitir som a cada 0,7 segundo;
- Quando estiver entre 20 e 30 cm de distância o buzzer irá emitir som a cada 0,5 segundo;
- Quando estiver entre 10 e 20 cm de distância o buzzer irá emitir som a cada 0,3 segundo;
- Quando estiver a uma distância menor que 10 cm, o buzzer irá emitir som a cada 0,1 segundo;

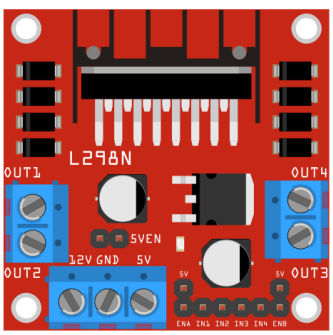
Não funcionou? Confira novamente todas as ligações, além disso, confira o valor dos pinos digitais no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar".

13. CONTROLANDO UM MOTOR DC



Arduino + Cabo USB (x1)

Motor DC (x1)



Módulo Ponte H (L298N) (x1)

Jumpers:

Vermelho (x4) Marrom (x1)
Preto (x1) Laranja (x1)

COMPONENTES

ACELERANDO COM MOTOR DC

CHEGOU A HORA DE ACELERAR UM MOTOR DC UTILIZANDO O ARDUINO E UM MÓDULO PARA CONTROLAR O MOTOR.

Descobertas: uso de módulos e motores.

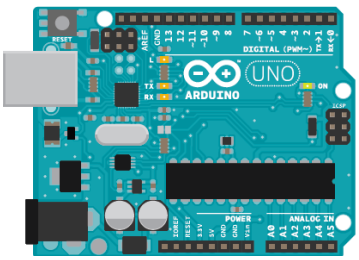
Tempo: **45 MINUTOS**

Dificuldade:

Nos projetos anteriores você utilizou o Arduino para controlar LEDs, buzzer, ler o sinal de sensores. Chegou a hora de controlar um motor DC utilizando, além do Arduino, um módulo ponte H.

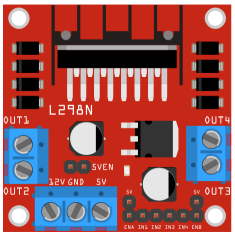
MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.



Arduino + Cabo USB (x1)

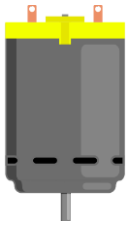
Motor DC (x1)



Módulo Ponte H (L298N) (x1)

Jumpers:
Vermelho (x4) Marrom (x1)
Preto (x1) Laranja (x1)

MOTOR DC

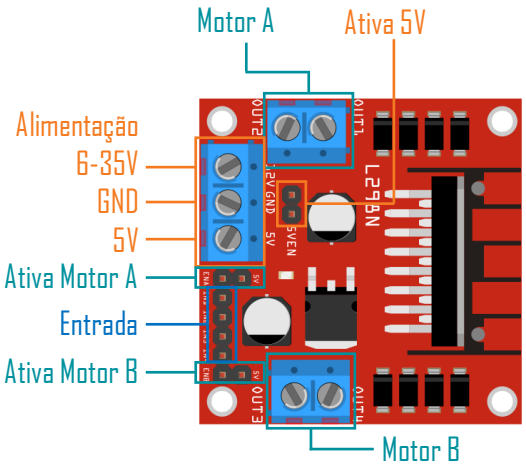


Motores elétricos são dispositivos eletromecânicos que convertem energia elétrica em energia mecânica na forma de rotação. Motores DC são uma classe de motores mais simples, que recebem uma tensão com corrente contínua que gera um campo magnético e pela atração e repulsão entre o campo magnético gerado e um imã presente no motor o eixo do motor rotaciona. Você utilizará um motor DC com faixa de alimentação de 3 a 6V e com caixa de redução de eixo duplo. A parte plástica, na qual o motor está encaixado, permite e redução da rotação do eixo do motor, tornando-o adequado para projetos que envolvem a construção de robôs, já que permite o encaixe de todas, além de um melhor controle de velocidade.

O acionamento e controle de motores DC não deve ser realizado diretamente no Arduino, já que, a maior parte desses motores consomem corrente maior do que a fornecida pelas portas digitais da placa, o que pode queimá-la.

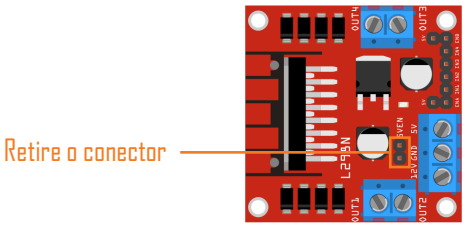
MÓDULO PONTE H (L298N)

Para esse projeto pode ser utilizado um Módulo Ponte H (L298N). Com esse módulo podem ser controlados até dois motores DC (Motor A e B). A alimentação desse módulo pode ser realizada utilizando fontes de 6 a 35V ou apenas 5V, para esse fim deve-se utilizar um jumper para ativar os 5V. Para controle, o módulo ponte H possui pinos para ativar os motores, além de dois pinos de entrada. Com esses pinos é possível controlar a velocidade do motor e o sentido da rotação.

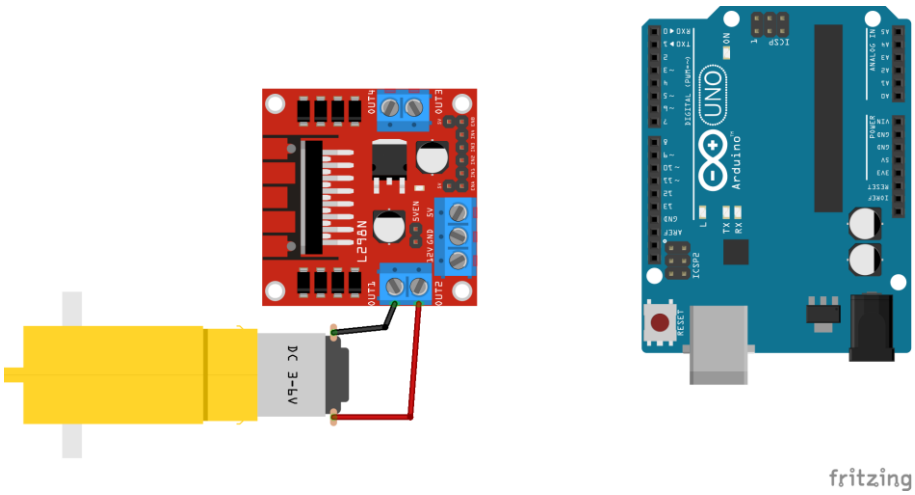


MONTANDO O CIRCUITO

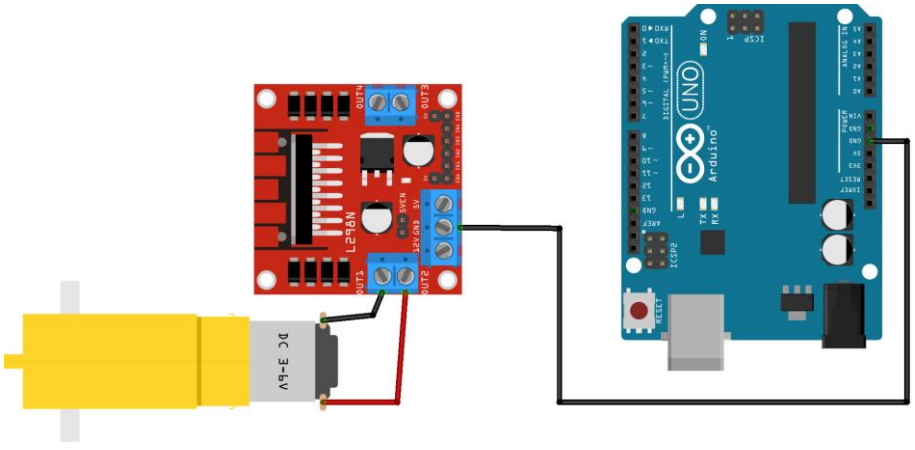
Etapa 02: Para iniciar a montagem do nosso circuito, como vamos utilizar 4 pilhas para alimentação elétrica dos motores e do Arduino, comece retirando, caso ainda não tenha sido retirado, dos pinos Ativa 5V. Esses dois pinos devem ficar "soltos".



Etapa 03: Agora, separe os componentes necessários e inicie conectando os pinos do motor no Módulo Ponte H (L298N). O motor é um dispositivo que não possui polaridade, dessa forma não importa a posição do fio vermelho e do preto. Essa posição pode interferir apenas na direção de rotação do motor, de acordo com o seu acionamento.

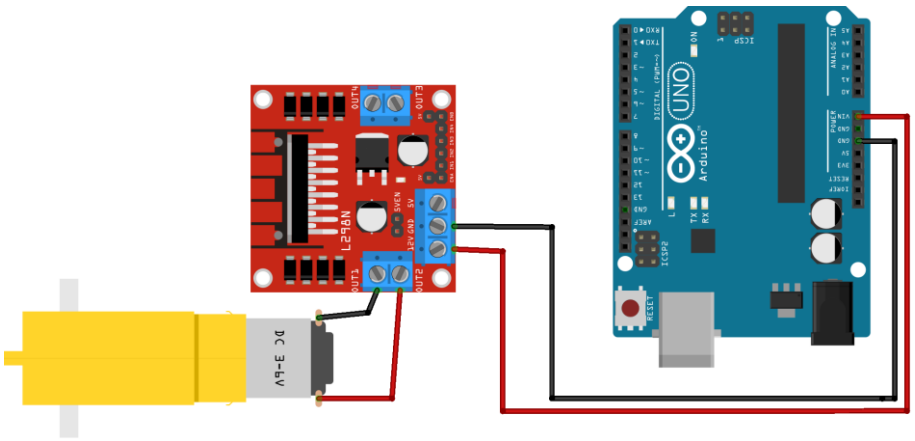


Etapa 04: Em seguida, vamos começar a alimentar o nosso circuito, começando pelo GND, conecte com um **jumper preto** o conector GND do Módulo Ponte H ao pino GND do Arduino.



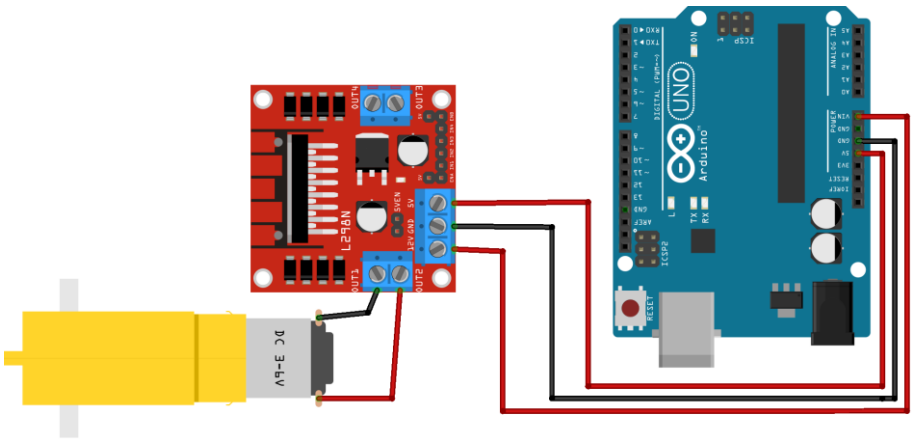
MONTANDO O CIRCUITO

Etapa 05: Vamos agora para a alimentação positiva. Conecte com um jumper vermelho o conector 12V do Módulo Ponte H ao pino Vin do Arduino.



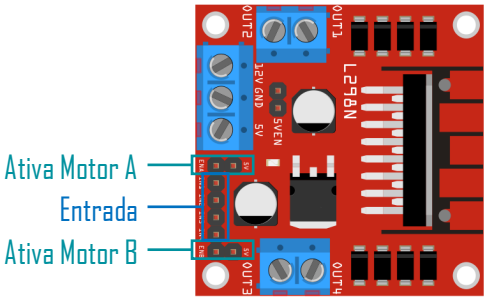
fritzing

Etapa 06: Em seguida, utilize um jumper vermelho para conectar o pino 5V do Arduino ao conector 5V do Drive de Ponte H.



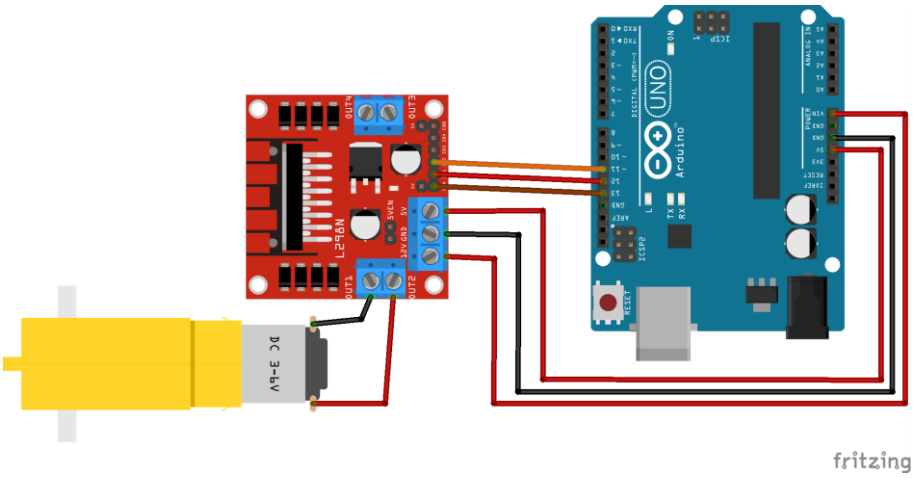
fritzing

Etapa 07: Nesse momento, todas as conexões relacionadas à alimentação elétrica do sistema já foram realizadas, restando apenas as conexões para ativar e controlar os motores. Confira se o pino que "Ativa Motor A" está sem o seu conector, ou seja, está "livre". Caso exista algum componente conectado a esse pino, retire-o.



MONTANDO O CIRCUITO

Etapa 08: Agora, utilizando três jumpers conecte, na sequência, os pinos ENA, IN1 e IN2, do Módulo Ponte H, aos pinos 13, 12 e 11 do Arduino. Conforme a figura abaixo.



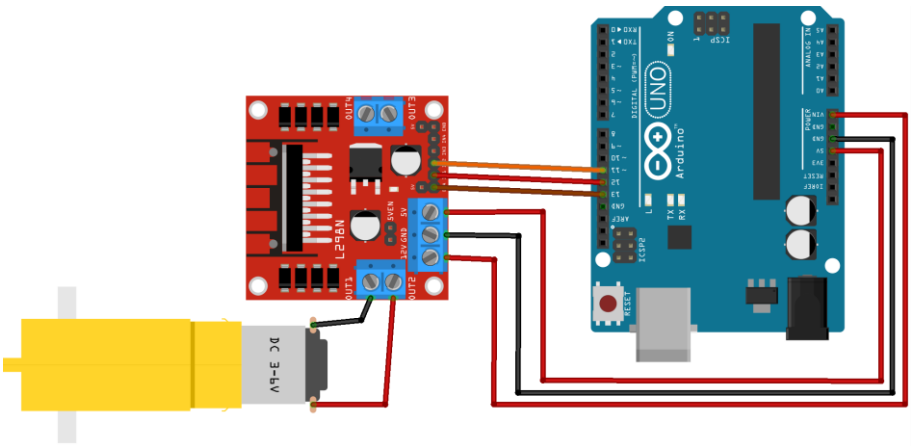
Etapa 09: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito para controlar um motor DC:

- Pino vermelho do motor DC conectado ao pino OUT1 do Módulo Ponte H.
- **Pino preto** do motor DC conectado ao pino OUT2 do Módulo Ponte H.
- Pino GND do Arduino conectado por um **jumper preto** ao GND do Módulo Ponte H.
- Pino Vin do Arduino conectado por um jumper vermelho ao 12V do Módulo Ponte H.
- Pino 5V do Arduino conectado por um jumper vermelho ao 5V do Módulo Ponte H.
- Pino 13 do Arduino conectado por um jumper marrom ao ENA do Módulo Ponte H.
- Pino 12 do Arduino conectado por um jumper vermelho ao IN1 do Módulo Ponte H.
- Pino 11 do Arduino conectado por um jumper laranja ao IN2 do Módulo Ponte H.
- Conector que Ativa 5V da placa retirado do Módulo Ponte H.

Etapa 09: Conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB, e inicie a conexão com o mBlock, no modo "Carregar".

CRIANDO O PROGRAMA

Antes de iniciar a programação vamos entender como a Módulo de Ponte H funciona de acordo com o nosso circuito.



O Pino 10 do Arduino está conectado ao Pino ENA do Módulo Ponte H, esse pino é responsável por ligar e desligar o motor. Dessa forma, quando o Pino 13 do Arduino estiver com o modo de saída "alto", o motor será ligado, quando estiver "baixo", o motor será desligado.

Já o Pino 12 e 11 do Arduino estão ligados aos Pinos IN1 e IN2 do Módulo Ponte H. Esses dois pinos são utilizados para definir o sentido que o motor irá girar, como ilustrado abaixo.

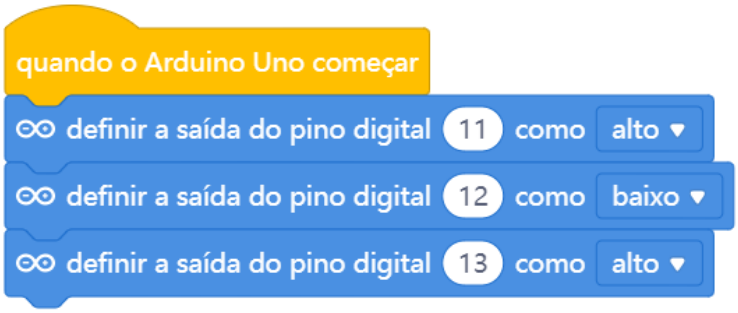
MOTOR A	IN1	IN2
SENTIDO DIRETO	ALTO	BAIXO
SENTIDO REVERSO	BAIXO	ALTO
FREIO	BAIXO	BAIXO
FREIO	ALTO	ALTO

Dessa forma, para que o motor gire em um sentido temos que colocar o pino ENA como "alto", para ativar o motor, e os pinos IN1 e IN2 como "alto" e "baixo". Para mudar o sentido da rotação basta colocar os pinos IN1 e IN2 como "baixo" e "alto". Caso os pinos IN1 e IN2 estejam ambos como "alto" ou como "baixo", o motor ficará parado. Porém deve-se evitar esse tipo de "freio" optando por desativar o motor definindo o pino ENA como "baixo".

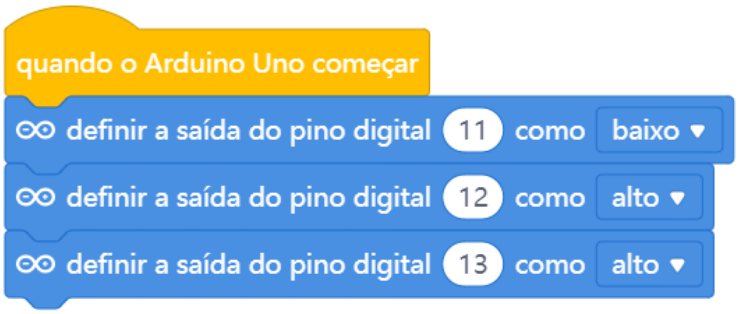
CRIANDO O PROGRAMA

ARDUINO	PONTE H
PINO 13	ENA
PINO 12	IN1
PINO 11	IN2

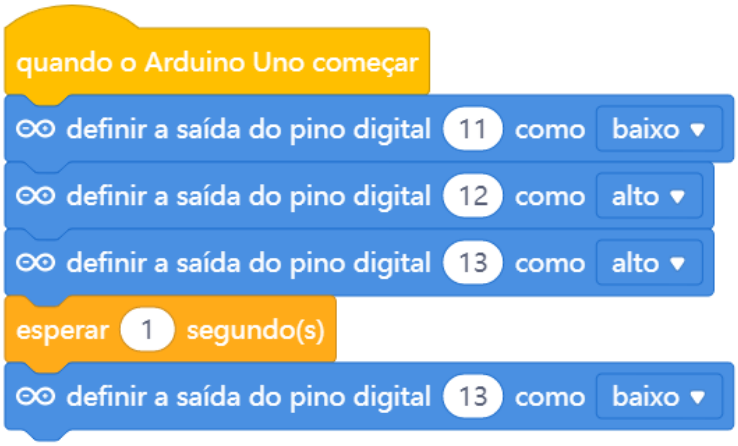
Etapa 10: Vamos criar um programa bem simples, apenas para ligar o motor e fazê-lo girar em um sentido. Antes de ligar o motor, vamos definir em qual sentido ele vai girar. Para que ele gire em um sentido defina os pinos 11 e 12, pinos IN1 e IN2, como "alto" e "baixo", respectivamente. Para ligar o motor temos que definir a saída do pino digital 13, pino INA, como "alto". Carregue o programa para o Arduino, observando se, ao término do processo de carregamento, o motor foi acionado e está girando.



Etapa 11: Para mudar o sentido do motor basta definir a saída do pino 9 como "baixo" e a saída do pino 8 como "alto". Faça essas mudanças e teste se o motor passou a girar no sentido oposto.

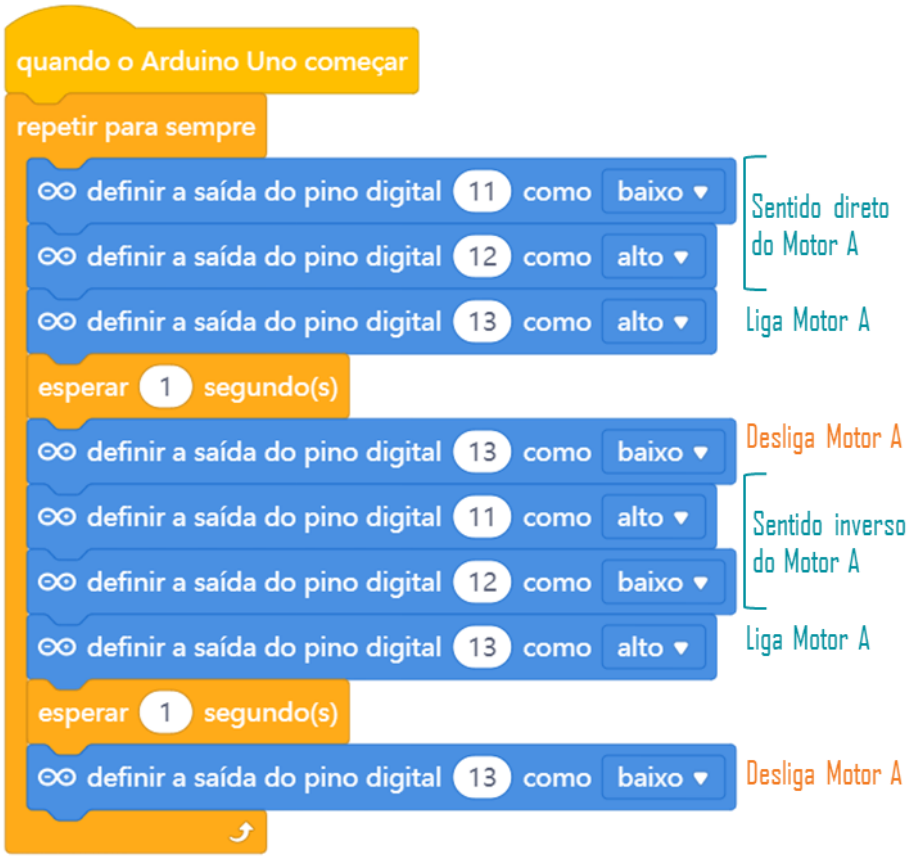


Etapa 12: Perceba que o motor fica girando sem parar. Para desligar o motor adicione um bloco esperar por 1 segundo e em seguida outro para definir a saída do pino 10 como "baixo", desativando o motor.



CRIANDO O PROGRAMA

Etapa 13: Vamos então fazer com que o motor gire em um sentido, desligue por 1 segundo, então gire em outro sentido e desligue por mais 1 segundo. Coloque os blocos dentre de um bloco para repetir para sempre e carregue para o Arduino.



Conforme pode ser observado no programa, é de se esperar que o motor gire no sentido direto por 1 segundo, então mude de sentido e gire no sentido inverso por mais 1 segundo.

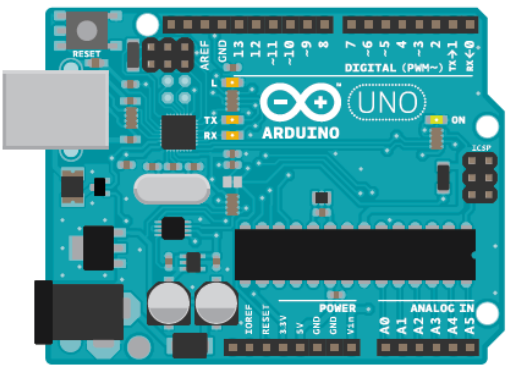
#Importante: Como estamos utilizando a mesma fonte de alimentação elétrica para o Arduino e para os motores, caso você tente mudar a direção do Arduino sem desligar o motor e esperar um intervalor de tempo, o motor pode funcionar como um gerador, isso faz com que o Arduino reinicie e pode até queimar a placa. Dessa forma, evite fazer a mudança do sentido do motor com esse ativado.

AGORA É A SUA VEZ

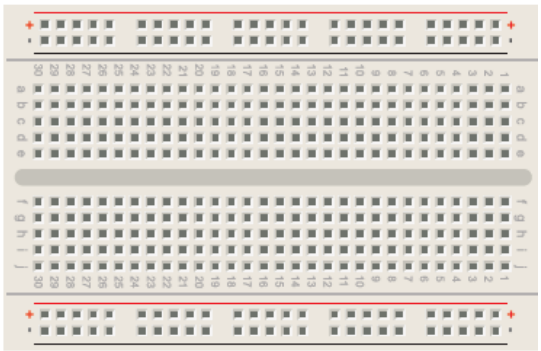
Faça com que o seu motor gire no sentido direto por 5 segundos, em seguida fique 1 segundo parado, depois gire no sentido inverso por 5 segundos e fique 1 segundo parado.

Não funcionou? Confira novamente todas as ligações, além disso, confira o valor dos pinos digitais no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar". Caso esteja tendo alguma dificuldade na gravação, desconecte o cabo USB e conecte novamente, gravando assim que inserir o cabo no computador.

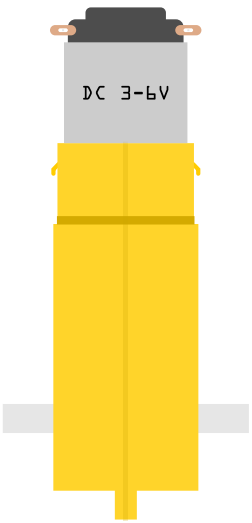
14. SENSOR DE COLISÃO



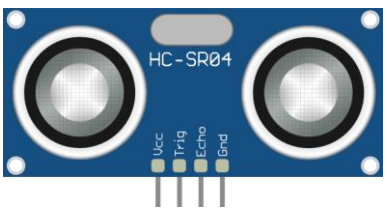
Arduino + Cabo USB (x1)



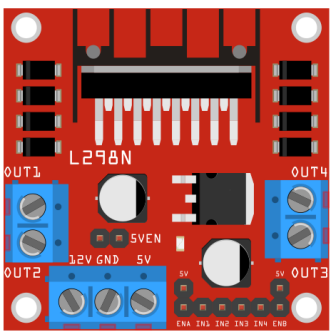
Placa de testes (x1)



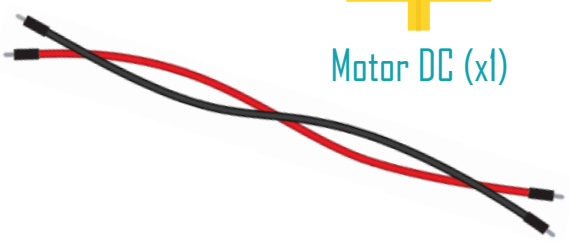
Motor DC (x1)



Sensor Ultrassônico de Distância (HCSR04) (x1)



Módulo Ponte H (L298N) (x1)



Jumpers:

- Vermelho (x4)
- Preto (x2)
- Laranja (x1)
- Verde (x1)
- Amarelo (x1)
- Marrom (x1)

COMPONENTES

EVITANDO ACIDENTES

ALGUNS CARROS MAIS NOVOS POSSUEM UM SENSOR DE COLISÃO. QUANDO ESSE SENSOR DETECTA UM OBSTÁCULO A UMA DETERMINADA DISTÂNCIA ELE FREIA O CARRO, EVITANDO QUE OCORRA ALGUM ACIDENTE.

Descobertas: associação de sensores e atuadores.

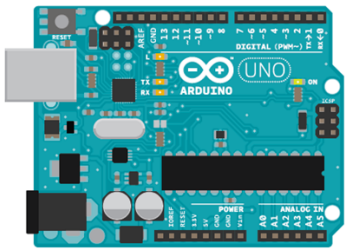
Tempo: **60 MINUTOS**

Dificuldade:

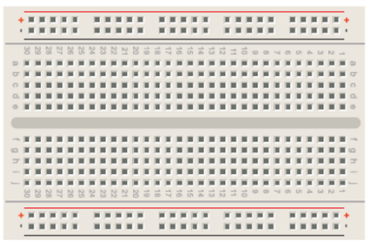
Nesse projeto, construam um sensor de obstáculo com um sensor de distância que desliga um motor ao detectar um obstáculo a menos de 20 cm.

MONTANDO O CIRCUITO

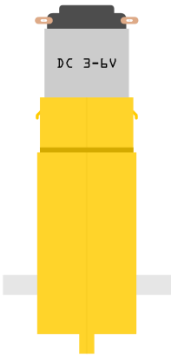
Etapa 01: Separe os componentes necessários para a montagem do circuito.



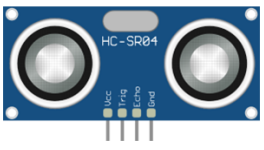
Arduino + Cabo USB (x1)



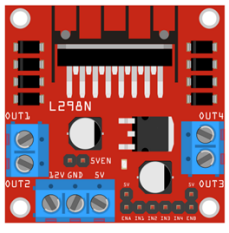
Placa de testes (x1)



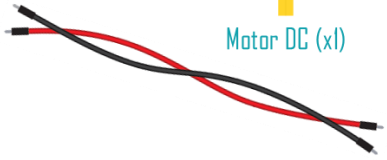
Motor DC (x1)



Sensor Ultrassônico de Distância (HCSR04) (x1)



Módulo Ponte H (L298N) (x1)

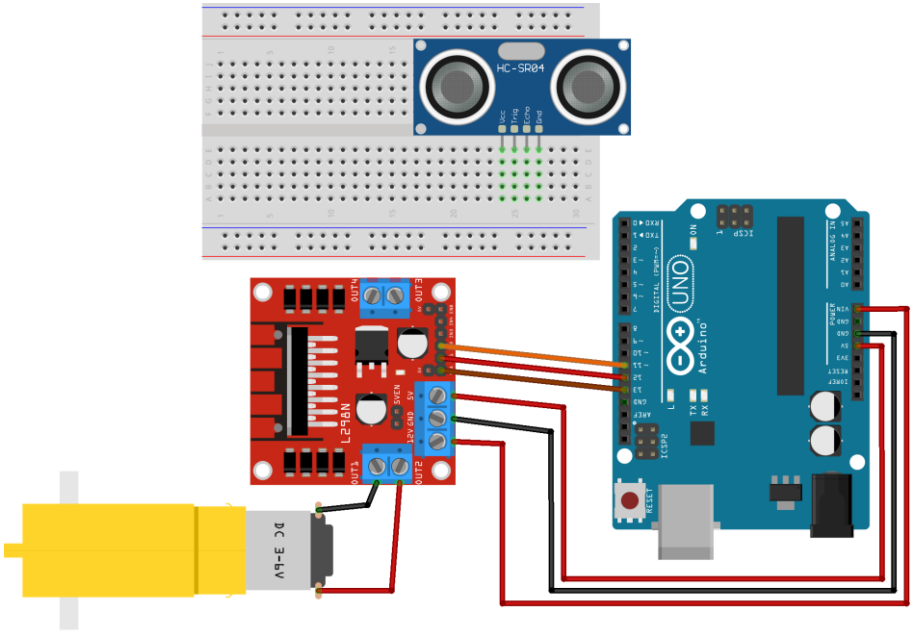


Jumpers:

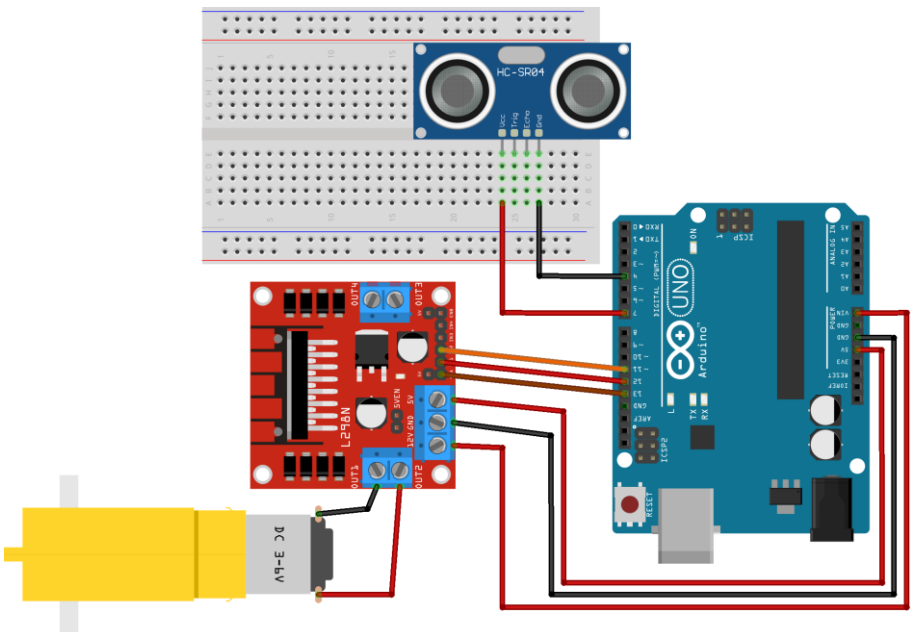
- Vermelho (x4)
- Preto (x2)
- Laranja (x1)
- Verde (x1)
- Amarelo (x1)
- Marrom (x1)

MONTANDO O CIRCUITO

Etapa 02: Para iniciar, monte um circuito exatamente igual aquele montado no projeto anterior. Em seguida, insira junto com uma placa de testes, o sensor ultrassônico de distância.

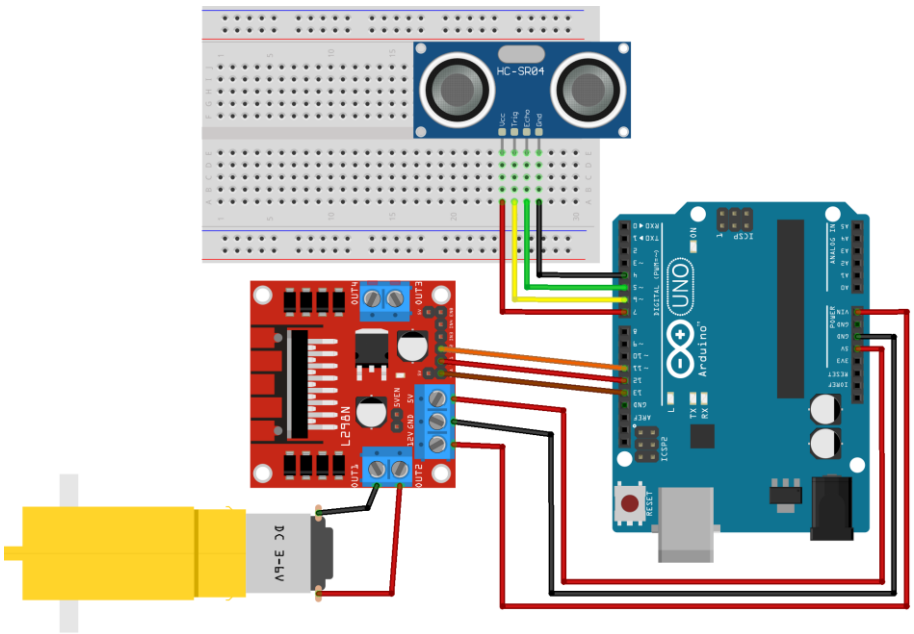


Etapa 03: Utilizando um **jumper preto**, conecte o pino GND do sensor ultrassônico ao pino 4 do Arduino e com um jumper vermelho, conecte o pino Vcc do sensor de distância ao pino 7 do Arduino.



MONTANDO O CIRCUITO

Etapa 04: Com jumpers amarelo e verde, conecte os pinos Trig e Echo do sensor ultrassônico aos pinos 5 e 6 do Arduino.



Etapa 05: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes.

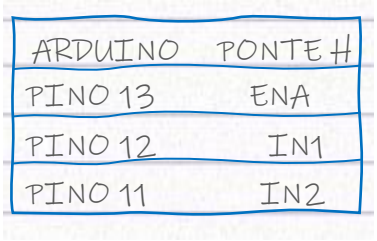
- Pino vermelho do motor DC conectado ao pino OUT1 do Módulo Ponte H.
- **Pino preto** do motor DC conectado ao pino OUT2 do Módulo Ponte H.
- Pino GND do Arduino conectado por um **jumper preto** ao GND do Módulo Ponte H.
- Pino Vin do Arduino conectado por um jumper vermelho ao 12V do Módulo Ponte H.
- Pino 5V do Arduino conectado por um jumper vermelho ao 5V do Módulo Ponte H.
- Pino 13 do Arduino conectado por um jumper marrom ao ENA do Módulo Ponte H.
- Pino 12 do Arduino conectado por um jumper vermelho ao IN1 do Módulo Ponte H.
- Pino 11 do Arduino conectado por um jumper laranja ao IN2 do Módulo Ponte H.
- Conector que Ativa 5V da placa retirado do Módulo Ponte H.
- Pino Vin do Sensor conectado por um jumper vermelho ao pino 7 do Arduino.
- Pino Trig do Sensor conectado por um jumper amarelo ao pino 6 do Arduino.
- Pino Echo do Sensor conectado por um jumper verde ao pino 5 do Arduino.
- Pino GND do Sensor conectado por um **jumper preto** ao pino 4 do Arduino.

CRIANDO O PROGRAMA

Etapa 06: Conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB, e inicie a conexão com o mBlock, no modo "Carregar".

Nesse projeto você deve criar um programa para ligar o motor e quando for detectado um obstáculo a menos de 20 cm do sensor desligar o motor. Voltando a liga-lo no momento em que não tiver mais obstáculo a menos de 20 cm.

Dica: Os pinos 7 e 4 do Arduino serão utilizados para alimentação de +5V e GND do Sensor de obstáculo.



Etapa 07: Vamos iniciar a nossa programação apenas ligando o nosso motor e fazendo com que ele gire no sentido direto. Para isso vamos definir os pinos 9 e 8 como saída "baixo" e "alto", para girar no sentido direto, e o pino 10 como "alto" para ligar o motor.

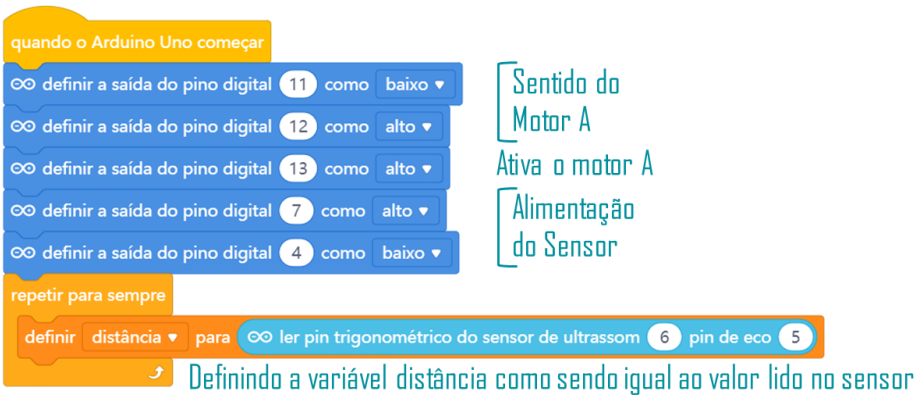
Etapa 08: Para saber qual a distância lida pelo sensor ultrassônico vamos criar uma variável "distância" e defini-la como sendo a distancia medida pelo sensor.

Como colocamos a alimentação do sensor, pinos Vcc (5V) e GND, conectado aos pinos 7 e 4, vamos defini-los então como pinos de saída "alta" e "baixa"

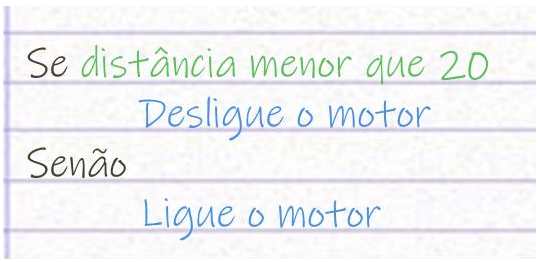
Definindo a variável distância como sendo igual ao valor lido no sensor

CRIANDO O PROGRAMA

Etapa 09: Podemos juntar esses blocos no nosso programa. Para que o valor da distância seja atualizado sempre, vamos colocar o bloco que define a distância dentro de um bloco de repetição.

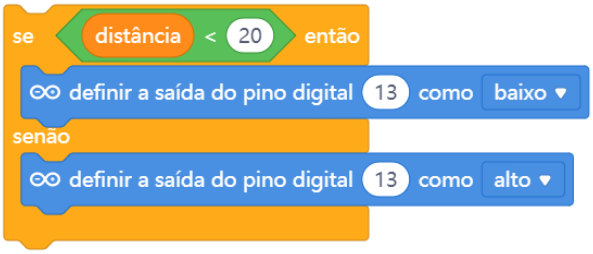


Resta apenas desligar o motor quando a distância for menor que 20 cm.



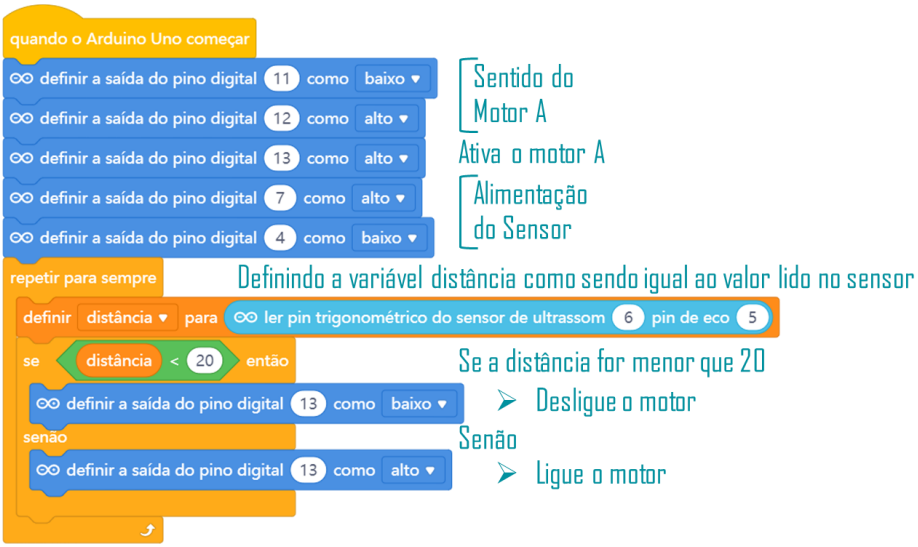
Etapa 10: Como vamos trabalhar com dois possíveis estados, se a distância for maior que 20 e senão for maior, vamos utilizar o bloco "SE SENÃO", com a condição do valor da variável "distância" ser menor (<) que 20.

Etapa 11: Para ligar o motor basta definir o pino 10 como "alto" e para desligar ele basta definir esse mesmo pino como "baixo".



CRIANDO O PROGRAMA

Etapa 12: Juntando todo o código:



Agora carregue seu código para o Arduino e teste se a programação funciona. Confira se todos os comandos estão sendo realizados de forma correta, olhe para o seu circuito e confirme se:

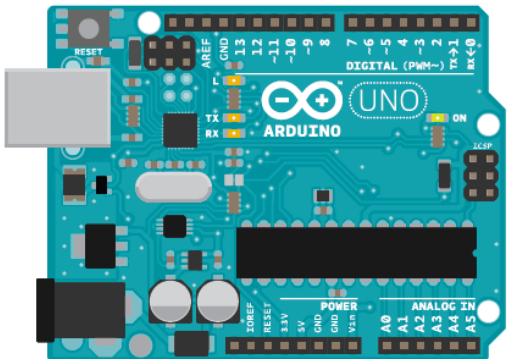
O motor está ativado e ao passar a mão ou algum objeto na frente do sensor o motor desliga.

AGORA É A SUA VEZ

Adicione um buzzer ao seu circuito e programe para que, além de frear, o buzzer emitir um sinal sonoro quando a distância for menor que 20 cm.

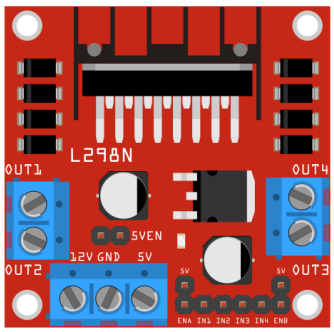
Não funcionou? Confira novamente todas as ligações, além disso, confira o valor dos pinos digitais no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar". Caso esteja tendo alguma dificuldade na gravação, desconecte o cabo USB e conecte novamente, gravando assim que inserir o cabo no computador.

15. CONTROLANDO DOIS MOTORES DC



Arduino + Cabo USB (x1)

Motor DC (x2)



Módulo Ponte H (L298N) (x1)

Jumpers:

Vermelho (x4) Marrom (x1)
Preto (x1) Laranja (x1)

COMPONENTES

CRIANDO MEUS PRÓPRIOS BLOCOS

AGORA QUE VOCÊ JÁ CONSEGUE CONTROLAR UM MOTOR, CHEGOU A HORA DE ACELERAR DOIS MOTORES DC UTILIZANDO O ARDUINO E CRIANDOS SEUS PRÓPRIOS BLOCOS.

Descobertas: criação de blocos no mBlock.

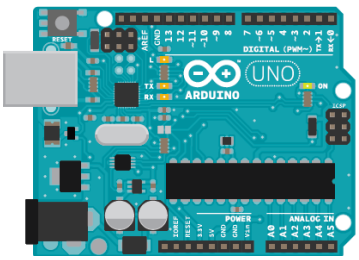
Tempo: **60 MINUTOS**

Dificuldade:

No projeto anterior você já conseguiu controlar um motor DC utilizando o Arduino e o Módulo de Ponte H, nesse módulo você irá controlar dois motores DC, de forma independente.

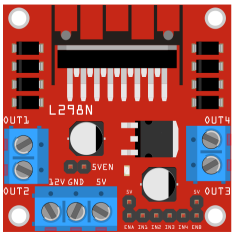
MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.



Arduino + Cabo USB (x1)

Motor DC (x2)

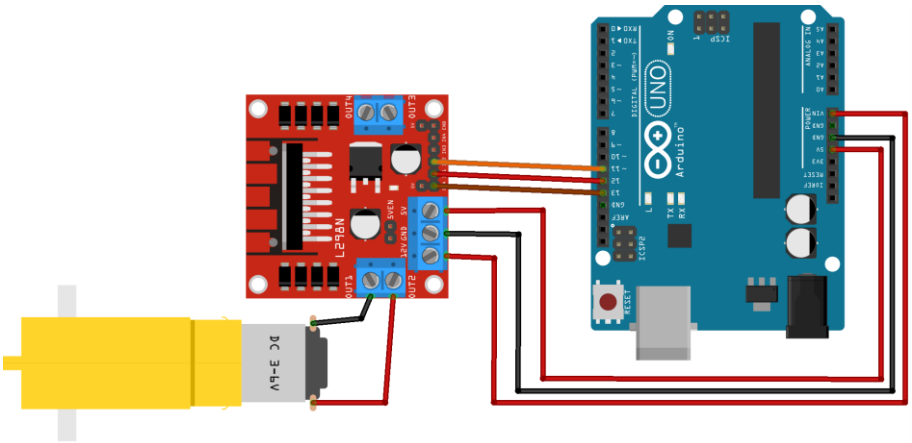


Módulo Ponte H (L298N) (x1)

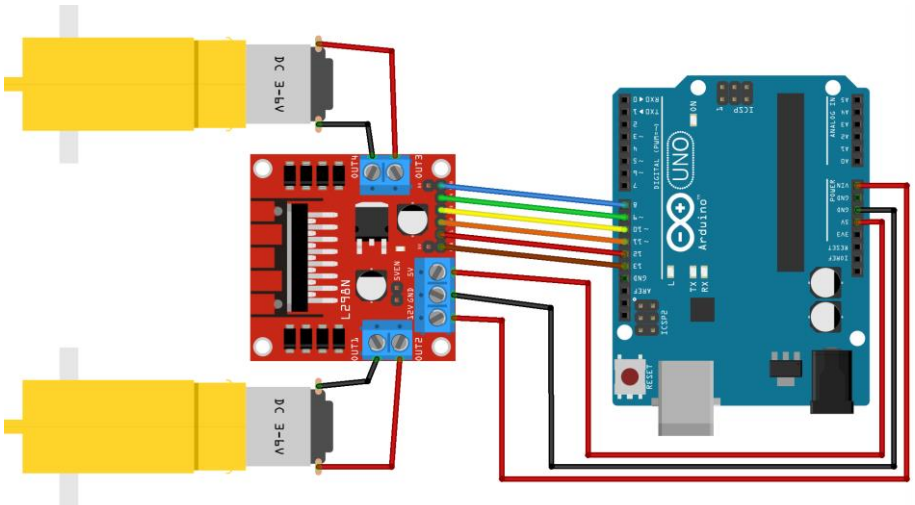
Jumpers:
Vermelho (x4) Marrom (x1)
Preto (x1) Laranja (x1)

MONTANDO O CIRCUITO

Etapa 02: Para iniciar, monte um circuito exatamente igual aquele montado no projeto anterior, porém adicione um segundo motor DC, com seus fios vermelho e preto, ligados aos conectores OUT3 e OUT4 do Módulo de Ponte H. Lembre de conferir se o conector do pinos para Ativar 5V foi retirado.



Etapa 03: Agora, realize a conexão dos pinos IN3, IN4 e ENB do Módulo de Ponte H aos pinos 10, 9 e 8 do Arduino, respectivamente, utilizando jumpers amarelo, verde a azul.



MONTANDO O CIRCUITO

Etapa 04: Antes de alimentar o circuito, confira todas as conexões realizadas. Atenção para os detalhes. Vamos conferir o circuito para controlar um motor DC:

- **Pino preto** do motor DC A conectado ao pino OUT1 do Módulo Ponte H.
- Pino vermelho do motor DC A conectado ao pino OUT2 do Módulo Ponte H.
- Pino vermelho do motor DC B conectado ao pino OUT3 do Módulo Ponte H.
- **Pino preto** do motor DC B conectado ao pino OUT4 do Módulo Ponte H.
- Pino GND do Arduino conectado por um **jumper preto** ao GND do Módulo Ponte H.
- Pino Vin do Arduino conectado por um jumper vermelho ao 12V do Módulo Ponte H.
- Pino 5V do Arduino conectado por um jumper vermelho ao 5V do Módulo Ponte H.
- Pino 13 do Arduino conectado por um jumper marrom ao ENA do Módulo Ponte H.
- Pino 12 do Arduino conectado por um jumper vermelho ao IN1 do Módulo Ponte H.
- Pino 11 do Arduino conectado por um jumper laranja ao IN2 do Módulo Ponte H.
- Pino 10 do Arduino conectado por um jumper amarelo ao IN3 do Módulo Ponte H.
- Pino 9 do Arduino conectado por um jumper verde ao IN4 do Módulo Ponte H.
- Pino 8 do Arduino conectado por um jumper azul ao ENB do Módulo Ponte H.
- Conector que Ativa 5V da placa retirado.

Etapa 05: Conecte a porta USB à placa Arduino Uno e ao computador, alimentando assim o circuito. Em seguida, abra o programa mBlock, ligue o Arduino ao computador, utilizando a porta USB, e inicie a conexão com o mBlock, no modo "Carregar".

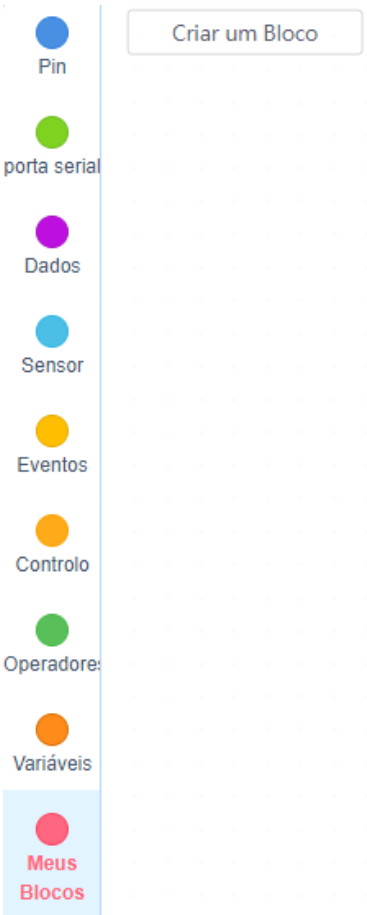
CRIANDO O PROGRAMA

Nesse projeto você deve criar um programa que controle os dois motores e os faça se movimentar da forma descrita na ilustração abaixo:

ETAPAS	SENTIDO: MOTOR A	MOTOR B
01	DIRETO	DIRETO
02	INVERSO	INVERSO
03	DIRETO	INVERSO
04	INVERSO	DIRETO

Você já deve imaginar como deve ser criado o programa, considerando o projeto anterior. Por outro lado, já deve imaginar também o tamanho do programa que será criado, já que serão controlados dois motores, mudando de sentido quatro vezes, de acordo com as etapas acima.

MEUS BLOCOS

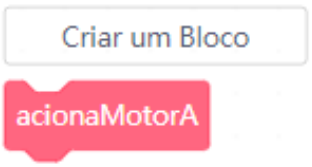


Em casos como esse, no qual a programação será basicamente a mesma, movimentar o motor em um sentido ou outro, porém será repetida várias vezes, nós podemos criar nossos próprios blocos no mBlock. Ao criar um bloco você pode inserir nele uma série de outros blocos, para realizar uma determinada tarefa, e utilizar esse bloco quando necessitar realizar a mesma tarefa.

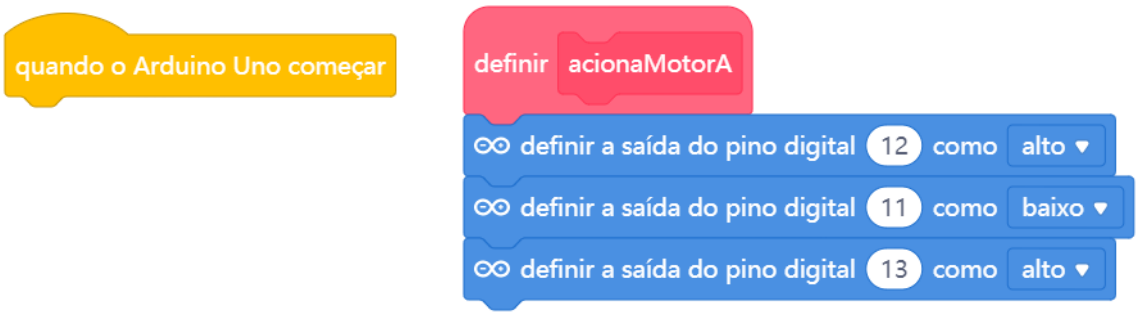
Etapa 06: Na parte inferior da lista de blocos, clique na opção "Meus Blocos" e em seguida o botão "Criar um Bloco". Na janela para criar um novo bloco você irá escolher o nome do bloco, além disso pode criar blocos que recebem entrada de número, texto ou um valor booleano (verdadeiro ou falso). Você pode ainda criar uma etiqueta de texto. No nosso primeiro exemplo vamos criar um bloco sem informação de entrada e chama-lo de "acionaMotor1".

CRIANDO O PROGRAMA

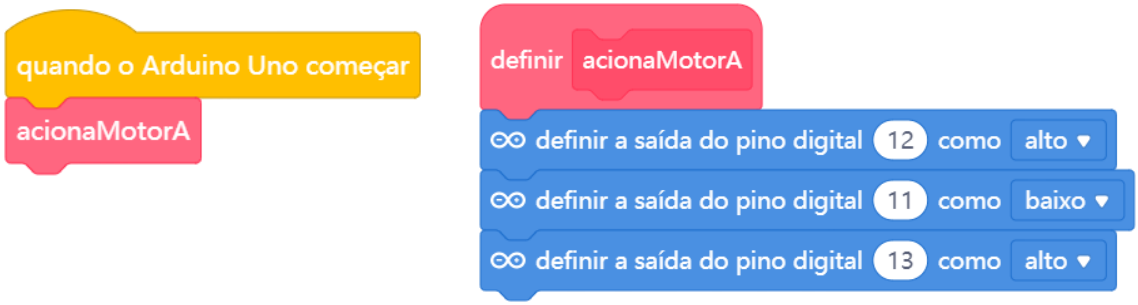
Perceba que, ao criar o novo bloco, ele aparece no menu de blocos, abaixo do botão para Criar um Blocos. Na área de programação esse bloco surge para ser definido, dessa forma, os blocos que são encaixados no "definir acionaMotorI", serão executados sempre que o bloco "acionaMotorI" for executado "quanto o Arduino Uno começar".

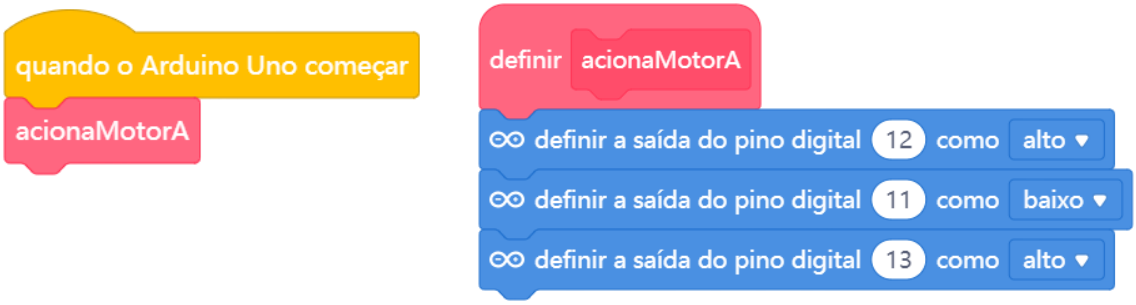


Etapa 07: Para compor nosso bloco, vamos começar por adicionar, abaixo do bloco para "definir acionaMotorI" os comandos que serão executados pelo nosso bloco. Como esse bloco foi criado para controlar o motor A, vamos criar um bloco apenas para ligar o Motor A e fazer ele rotacionar no sentido direto.



Etapa 08: Para testar se o seu bloco personalizado funcionou, adicione agora, abaixo da função "quando o Arduino Uno começar", o bloco "acionaMotorA", que você acabou de criar. Carregue o programa para o Arduino e observe se o motor A foi acionado.





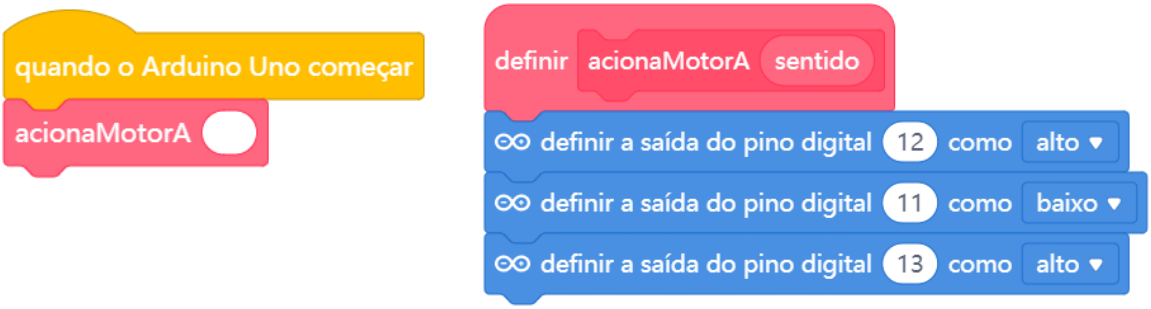
Lembre que, nos projetos anteriores, você utilizava apenas blocos abaixo do "quando o Arduino Uno começar". A partir desse momento você pode criar seus próprios blocos personalizados. Você pode, por exemplo, criar outro bloco para desligar o Motor A e fazer ele girar no sentido oposto. Porém, você pode fazer tudo isso utilizando apenas uma única função, porém definindo uma informação de entrada.

Etapa 09: Vamos então, editar o nosso bloco, clique no bloco com o botão direito do mouse e escolha a opção "editar". Em seguida escolha a opção "Adicionar uma entrada texto" e nomeie essa entrada de texto como "**sentido**".

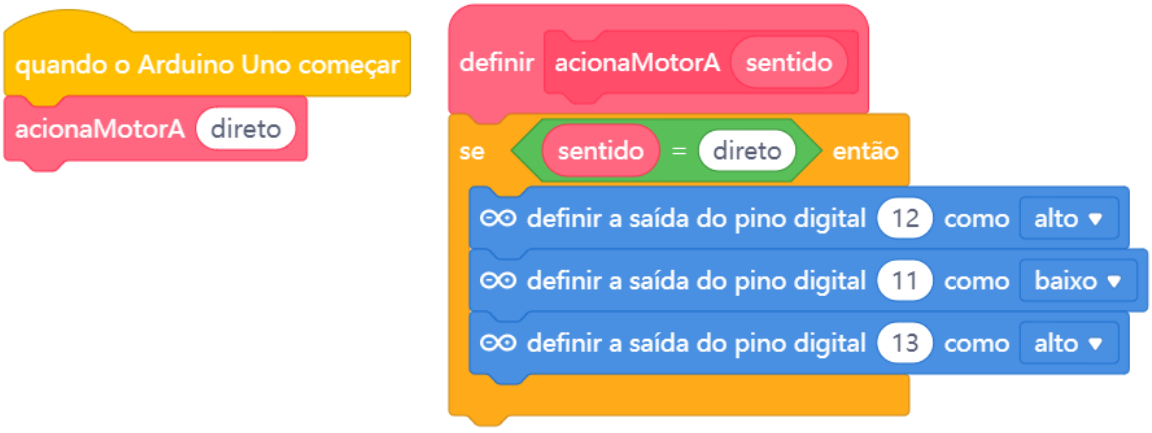
Agora podemos, utilizando nosso bloco, escolher para qual sentido o motor deve girar e ele definir, automaticamente, qual o estado dos pinos.

CRIANDO O PROGRAMA

Perceba que seu bloco, abaixo do bloco "quando o Arduino Uno começar", possui um espaço para que você possa informar uma informação no formato de texto. Na definição do nosso bloco "acionaMotorA", você pode utilizar as informações recebidas que serão armazenadas na variável "sentido".



Etapa 10: Vamos começar utilizando uma função "se < > então", presente no menu de blocos "Controlo", dentro do nosso bloco. Vamos definir que, se a "direção" for igual a "direta", utilizando o bloco operador de igualdade, então execute os três comandos anteriores. Em seguida, já especifique o valor de entrada do bloco "acionaMotorA" como sendo "direto", carregue para o Arduino Uno e teste se o motor é acionado.



Você pode estar pensando que esse código funciona da mesma forma que o anterior, só que de forma mais complicada. Mas o uso da variável "sentido" permite que você realize diferentes comandos no seu bloco, de acordo com o valor fornecido de entrada, quando o bloco "acionaMototA" é definido.

CRIANDO O PROGRAMA

Etapa 11: Agora, vamos definir na nossa função quando o motor gira no sentido inverso e quando ele deve ficar parado. Para girar no sentido inverso basta que você repita o mesmo código utilizado para o sentido direto, mudando apenas o estado dos pinos 9 e 8 para "baixo" e "alto".

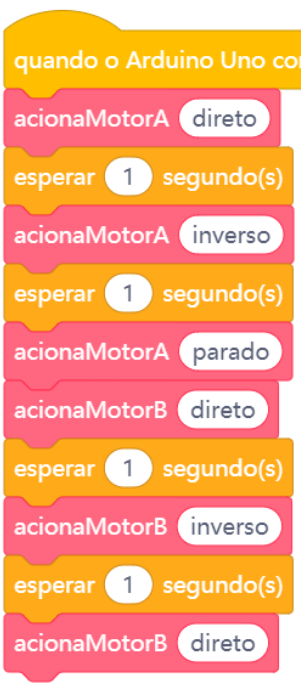
Para definir quando o motor fica parado vamos utilizar a seguinte estratégia, sabemos que sempre que temos que mudar o sentido da rotação do motor deve-se desativar esse motor, ou seja, definir o pino 10 como baixo. Podemos então definir o pino 10 como "baixo" no início da execução do nosso bloco. Dessa forma, sempre que definirmos alguma mudança de sentido o bloco já começa desativando o motor, até que o sentido seja modificado e, em seguida, o motor é religado. Além disso, ao fornecermos como entrada a palavra "parado" o motor será desativado e não haverá mudança de sentido.

Seu código deve ficar semelhante ao apresentado no exemplo abaixo, teste acionar o motor A no sentido "direto" por 1 segundo, então acionar o motor A no sentido "inverso" por 1 segundo e, por fim, definir o sentido do motor A como parado.

Se tudo deu certo o seu motor A girou no sentido direto, por 1 segundo, em seguida girou no sentido inverso, por 1 segundo e, então, ficou parado. Vamos relembrar agora qual o desafio desse projeto.

CRIANDO O PROGRAMA

Etapa 12: Crie um novo bloco, chamado de "acionaMotorB" que receba de entrada uma variável "sentido", e adicione nesse bloco os mesmos comandos que foram adicionados no bloco "acionaMotorA", modificando apenas os pinos de acionamento do motor para pino 5 e os pinos de direção para os pinos 6 e 7. Os blocos devem ficar como os apresentados na figura abaixo.



Etapa 13: Para testar, adicione abaixo dos blocos presentes no bloco "quando o Arduino Uno começar" os comandos para acionar o motor B no sentido "direto", e aguardar 1 segundo, em seguida o bloco para acionar o motor B no sentido "inverso", aguardar 1 segundo e, por fim, defina o sentido do motor B como parado.

Observe se o motor A gira em um sentido, por 1 segundo, muda de sentido e gira por 1 segundo, então é desligado e no mesmo instante o motor B começa a girar por 1 segundo, quando muda de sentido e após mais 1 segundo é então desativado.

CRIANDO O PROGRAMA

Vamos relembrar agora qual o desafio desse projeto.

ETAPAS	SENTIDO: MOTOR A	MOTOR B
01	DIRETO	DIRETO
02	INVERSO	INVERSO
03	DIRETO	INVERSO
04	INVERSO	DIRETO

Etapa 14: Agora que você já possui os dois blocos criados, para acionar os motores A e B, de acordo com o sentido desejado, basta adicionar os blocos abaixo do "quando o Arduino Uno começar". Deixe um intervalo de 1 segundo entre as etapas, lembrando de desligar os motores ao fim das etapas. A organização dos seus blocos deve ser semelhante aquela apresentada na figura.

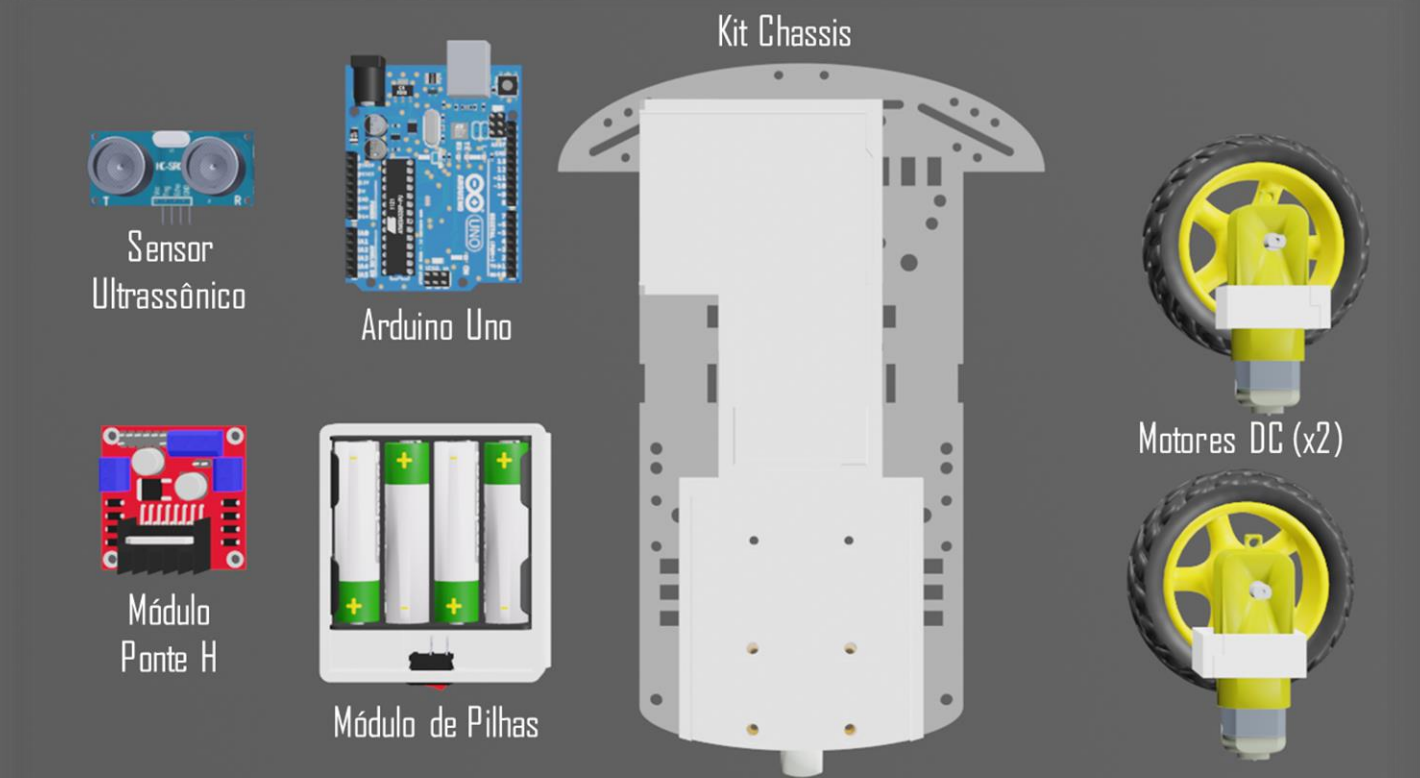
Observe a rotação dos motores, se eles se movem, inicialmente, para o mesmo sentido, depois no sentido inverso e em seguida se movem em sentidos opostos.

AGORA É A SUA VEZ

Mude a direção e o tempo de acionamento dos motores, imagine como seriam os comandos de direção para que, ao utilizar esses dois motores em um carro, o carro pudesse ir para frente, para trás, virar para um lado e para o outro.

Não funcionou? Confira novamente todas as ligações, além disso, confira o valor dos pinos digitais no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar". Caso esteja tendo alguma dificuldade na gravação, desconecte o cabo USB e conecte novamente, gravando assim que inserir o cabo no computador.

16. MONTANDO UM ROBÔ COM DOIS MOTORES



COMPONENTES

PERCEBENDO O MUNDO

ESTÁ CHEGANDO A HORA DE JUNTARMOS TUDO QUE ESTUDAMOS PARA CONTROLAR UM KIT DE ROBÓTICA COM DOIS MOTORES E UM SENSOR ULTRASSÔNICO, MAS ANTES VAMOS MONTAR O KIT.

Descobertas: uso de sensores, leitura de sinal em pinos digitais, condicional dupla.

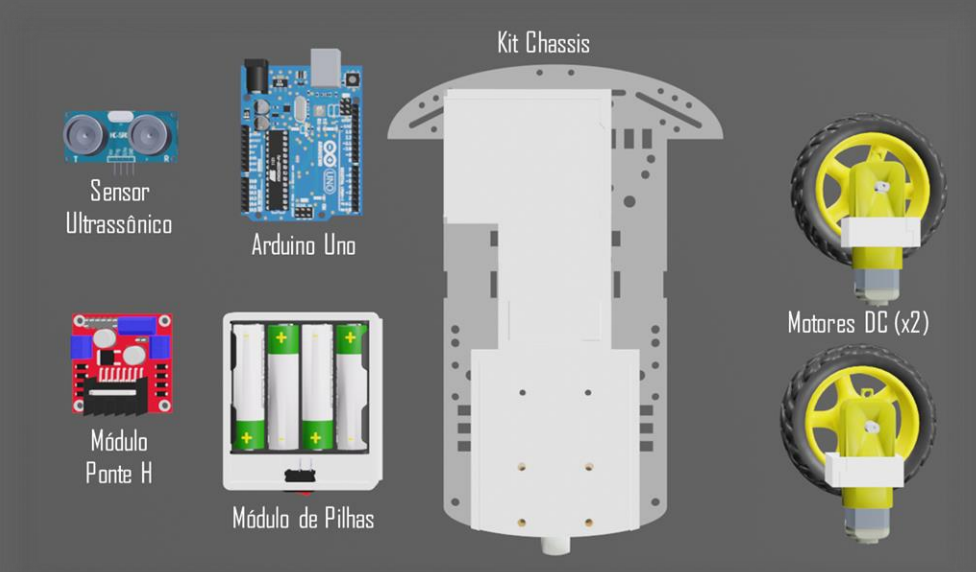
Tempo: **45 MINUTOS**

Dificuldade:

Nos projetos anteriores você utilizou o Arduino para controlar LEDs e Buzzer, nesses projetos as portas digitais foram sempre utilizadas no modo de "saída", nesse projeto, ao utilizar um sensor de proximidade, você vai aprender como funciona a "entrada" de dados em pinos digitais

MONTANDO O CIRCUITO

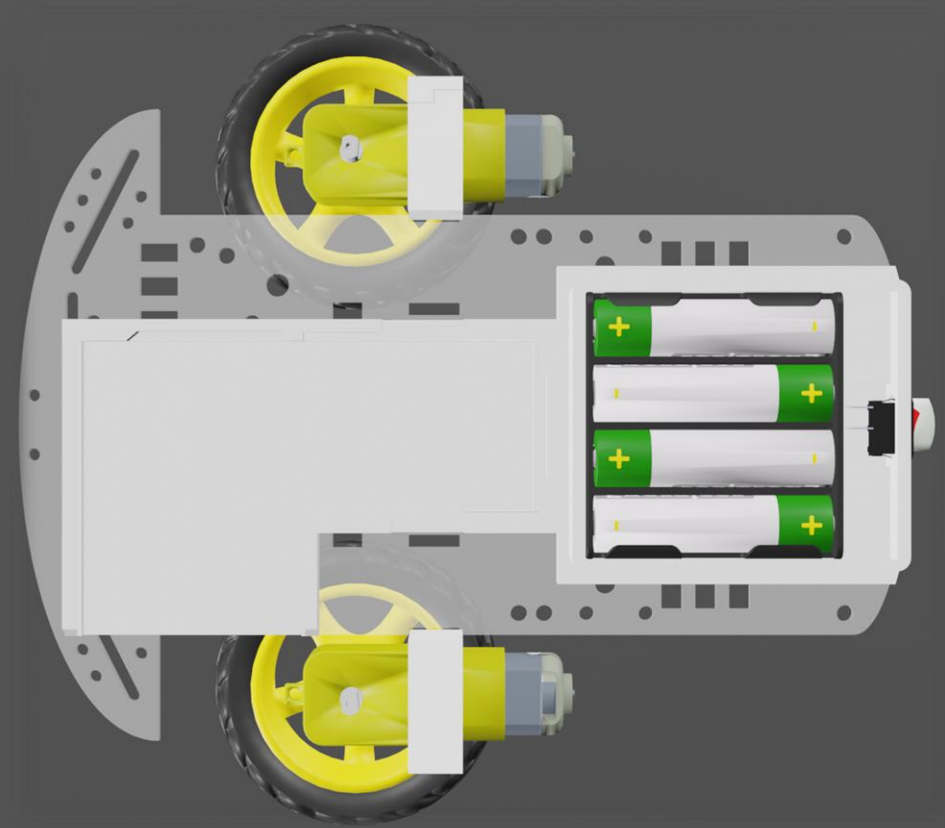
Etapa 01: Separe os componentes necessários para a montagem do circuito.



MONTANDO O KIT DE ROBÓTICA

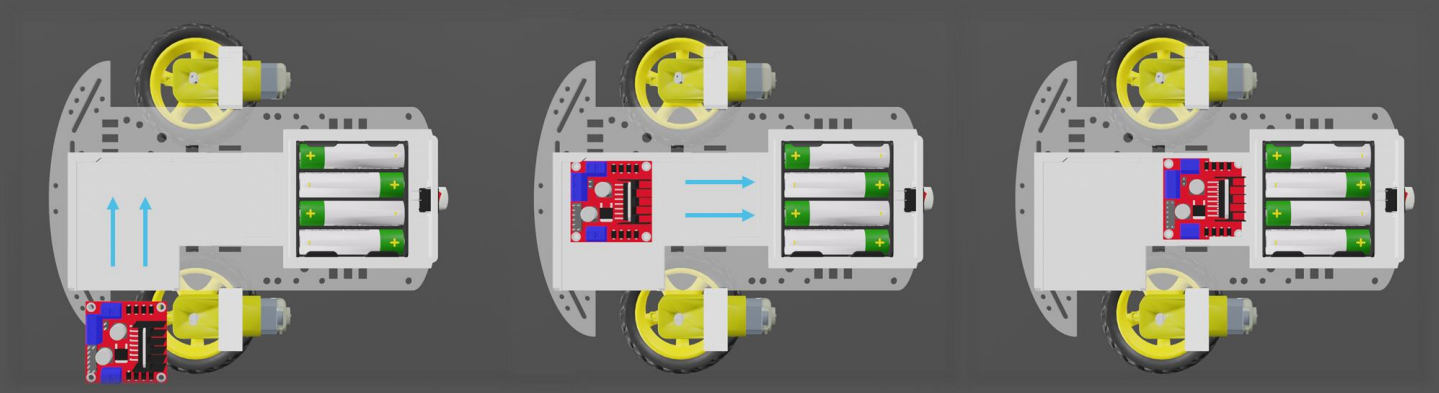
Etapa 02: Comece encaixando as rodas nos motores DC, observando o sentido do encaixe e com cuidado para não forçar o pino de plástico.

Etapa 03: Encaixe o módulo de pilhas no chassis e coloque o chassis de acrílico encostado nos motores, como na imagem..

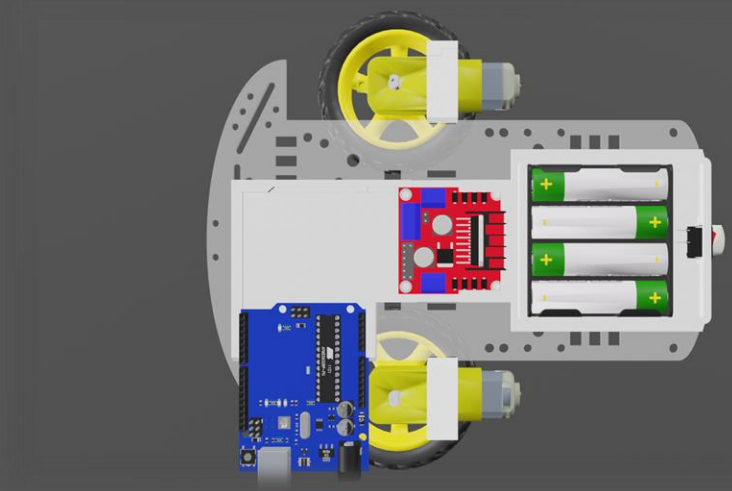


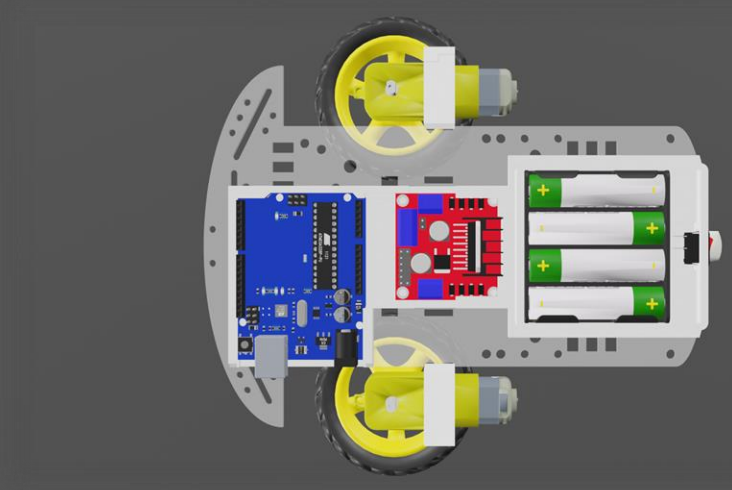
MONTANDO O KIT DE ROBÓTICA

Etapa 04: Encaixe o módulo de ponte H no chassis, conforme na imagem:



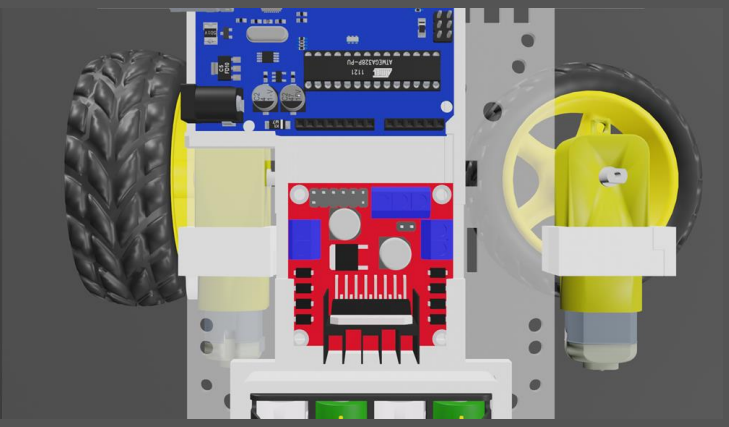
Etapa 05: Encaixe Arduino no chassis, conforme na imagem:



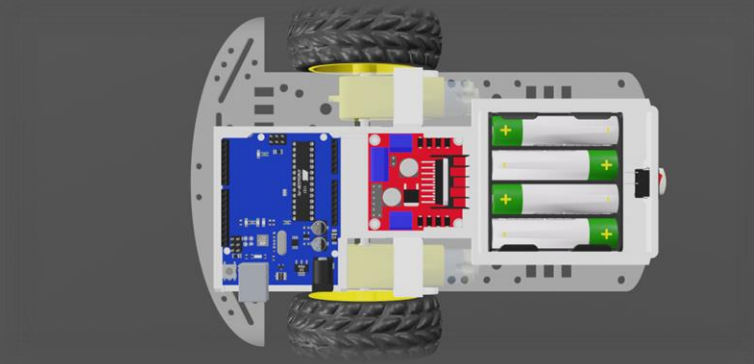


MONTANDO O KIT DE ROBÓTICA

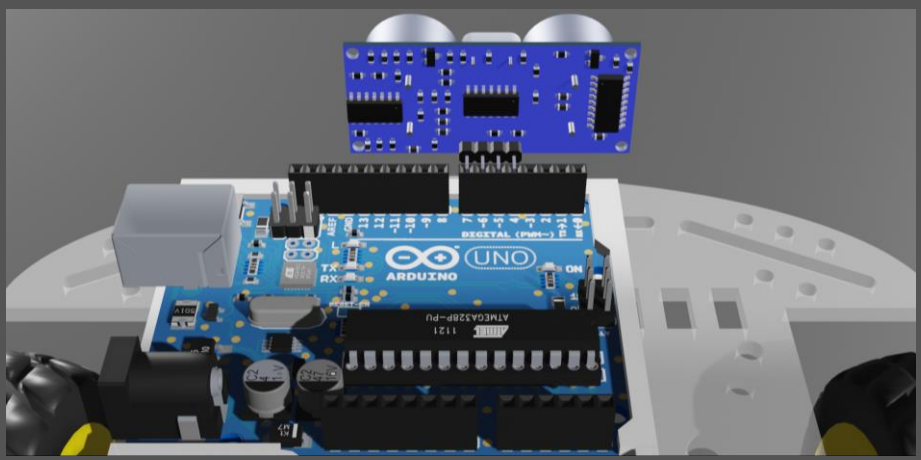
Etapa 06: Abra o encaixe plástico o motor e insira até o espaço ao lado do drive de ponte H, como na imagem:



Etapa 07: Abra o encaixe plástico do segundo motor e insira até o espaço ao lado do drive de ponte H, como na imagem:

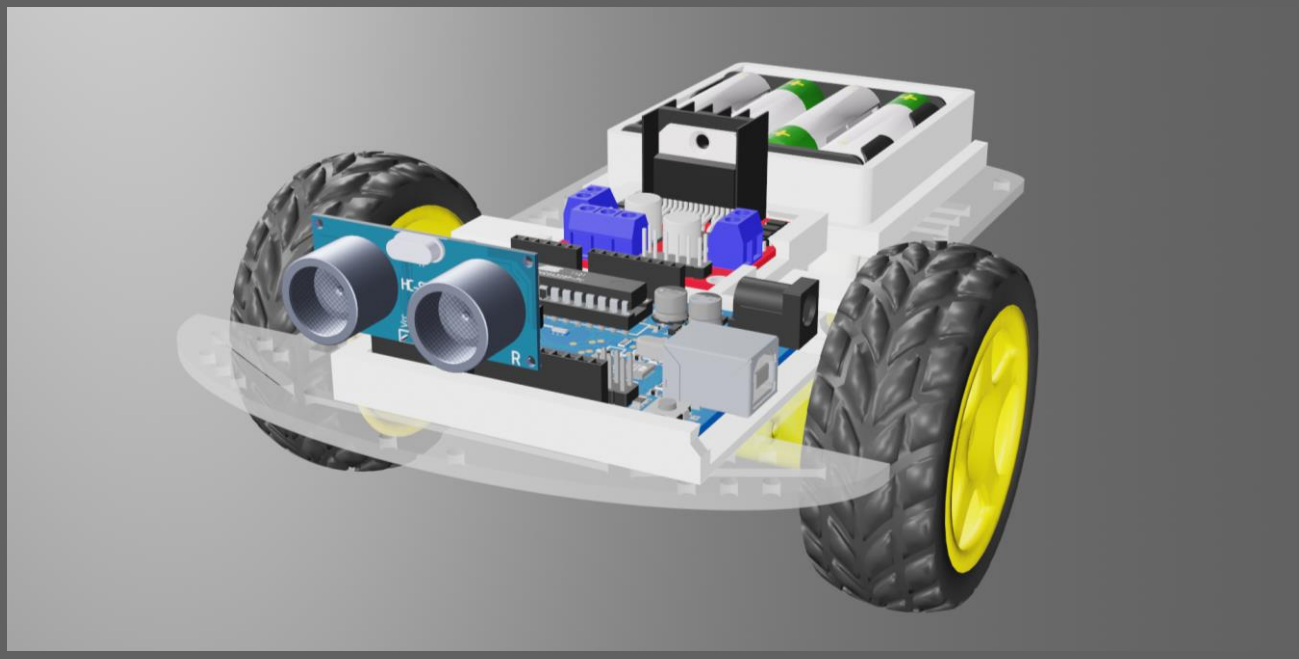


Etapa 07: Abra o encaixe plástico do segundo motor e insira até o espaço ao lado do drive de ponte H, como na imagem:



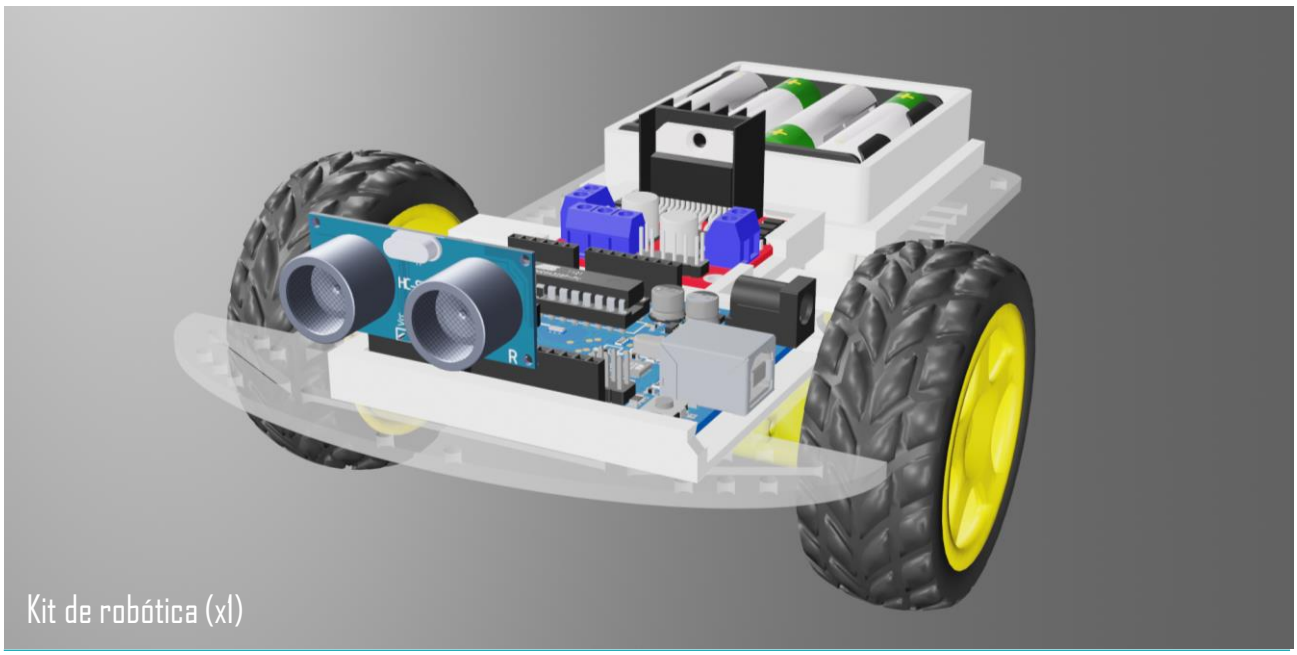
MONTANDO O KIT DE ROBÓTICA

Etapa 08: O kit de robótica deve ficar montado como na imagem:



Não funcionou? Confira novamente todas as partes, volte as páginas e reinicie a montagem do kit de robótica.

17. CONTROLANDO UM ROBÔ AUTÔNOMO



Kit de robótica (xl)

COMPONENTES

CRIANDO UM ROBÔ AUTÔNOMO

CHEGOU A HORA DO PROJETO FINAL! CRIAR UM ROBÔ AUTÔNOMO, OU SEJA, QUE DIRIJA SOZINHO E SAIBA DESVIAR DE OBSTÁCULOS.

Descobertas: controlar um robô autônomo.

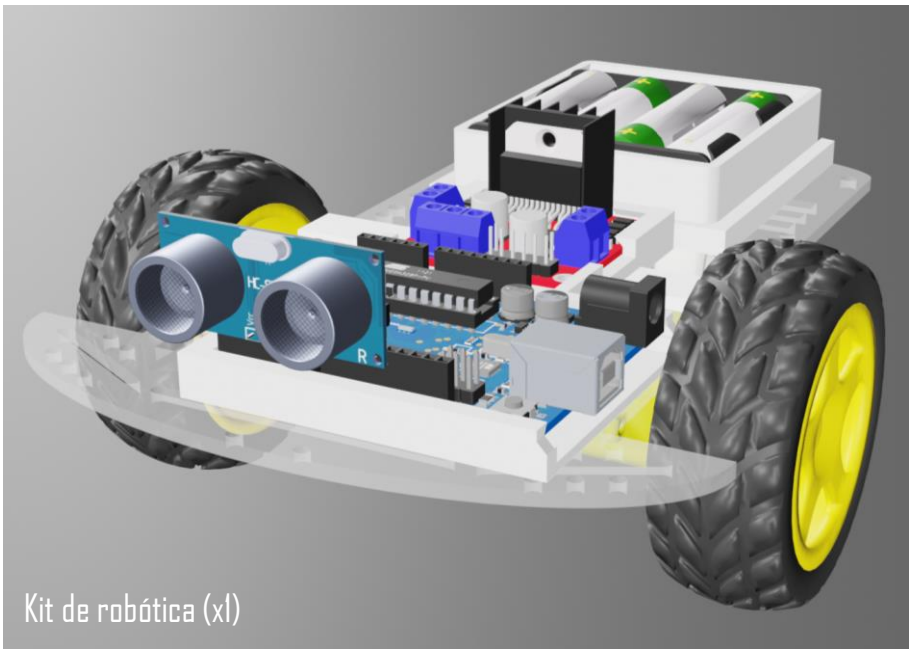
Tempo: **60 MINUTOS**

Dificuldade:

Nos projetos anteriores você montou o robô autônomo e já sabe como ler o sensor ultrassônico e controlar dois motores DC, chegou a hora de juntar tudo que estudamos para criar um robô autônomo!

MONTANDO O CIRCUITO

Etapa 01: Separe os componentes necessários para a montagem do circuito.

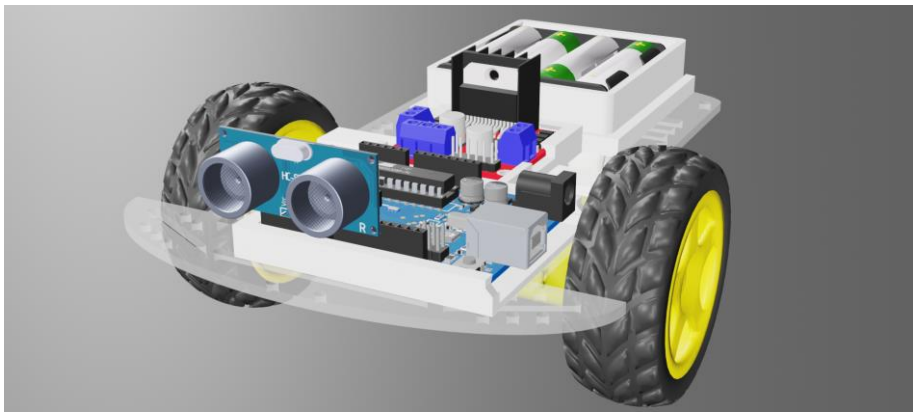


Kit de robótica (xl)

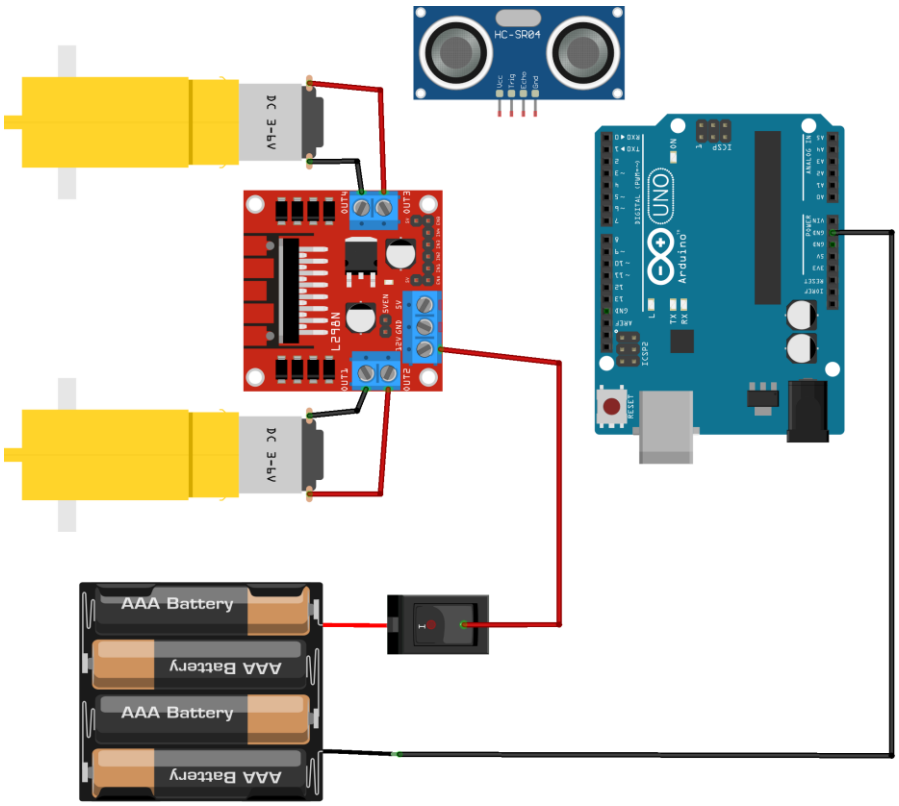
MONTANDO O CIRCUITO

Para controlar a movimentação do robô autônomo você deve programar seu robô para se mover em frente e utilizar o sensor ultrassônico para detectar algum obstáculos a menos de 20 cm e, ao encontrar algum obstáculos, programar o robô para girar 90° antes de continuar indo em frente.

Etapa 02: Monte o kit de robótica como realizado no projeto anterior.

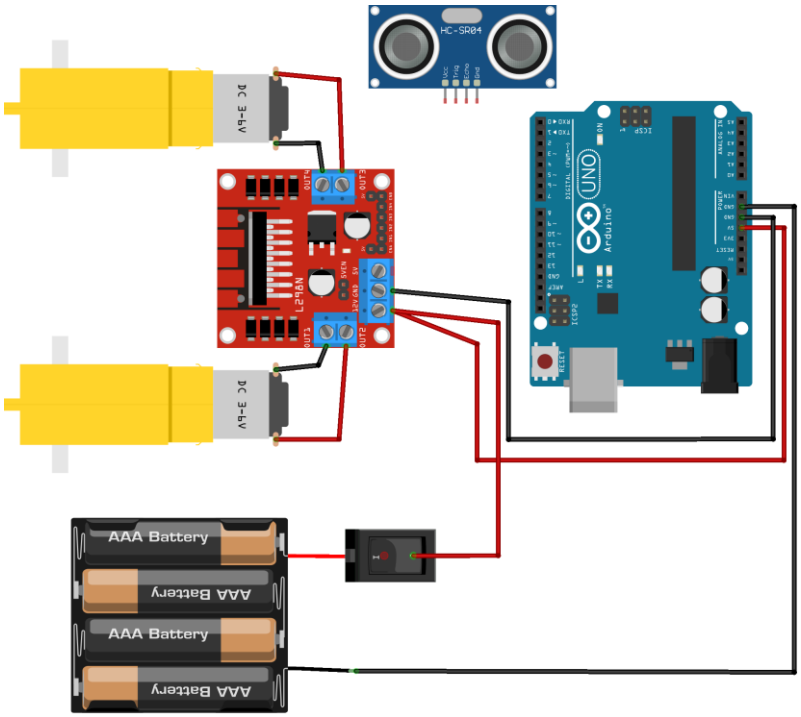


Etapa 03: Inicie a montagem do circuito conectando o módulo de pilhas. Fio preto no GND e o vermelho ao 12V do módulo Ponte H. Além disso, conecte os motores ao drive de ponte H, utilizando a chave para abrir o conector do drive de ponte H e fechá-lo depois.

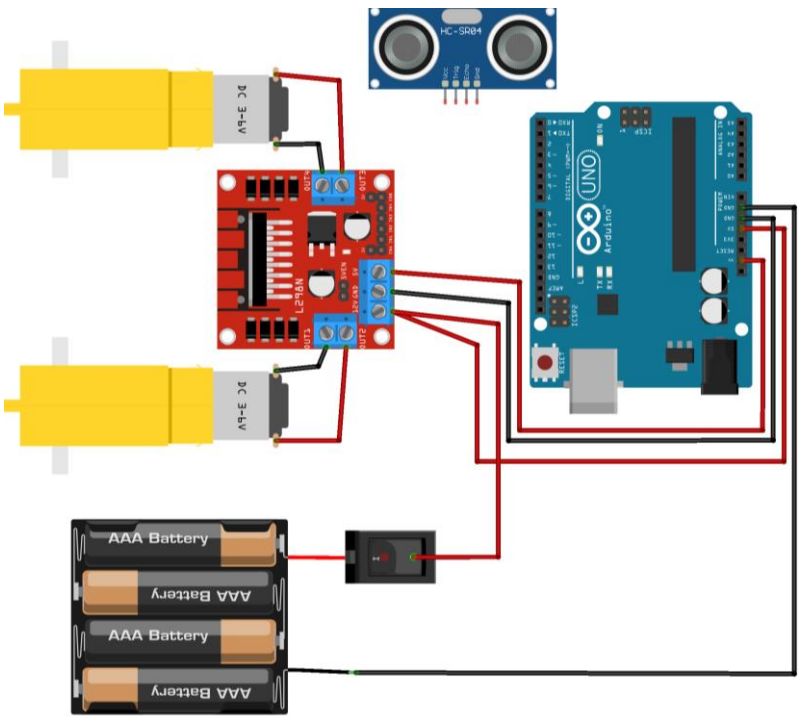


MONTANDO O CIRCUITO

Etapa 04: Em seguida, conecte um jumper preto o GND do drive ponte H ao GND do Arduino e um segundo jumper vermelho no pino 12V ao pino 5V do Arduino.

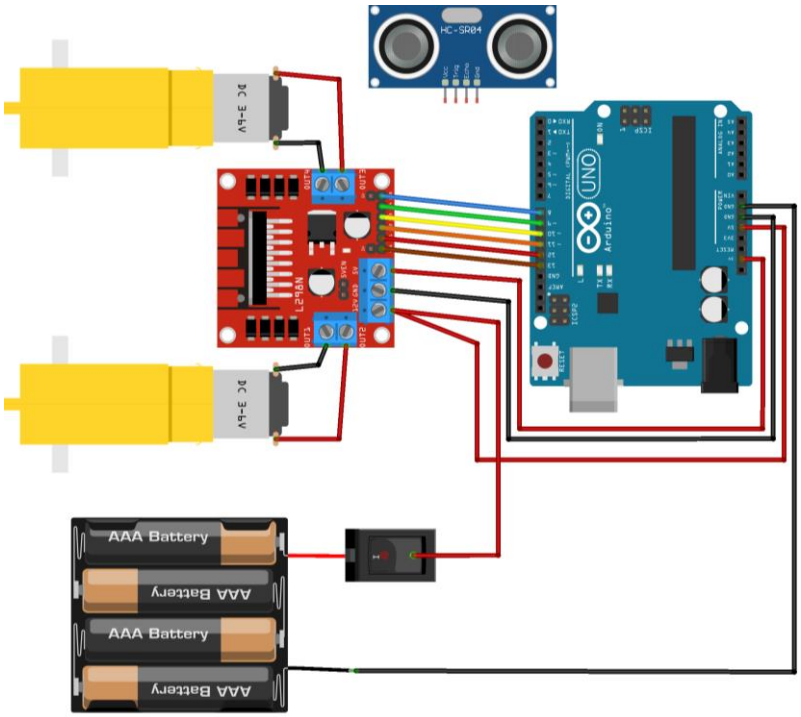


Etapa 05: Da mesma forma que o anterior, conecte agora, com um jumper vermelho, o conector 5V do drive de ponte H ao pino 5V do Arduino.

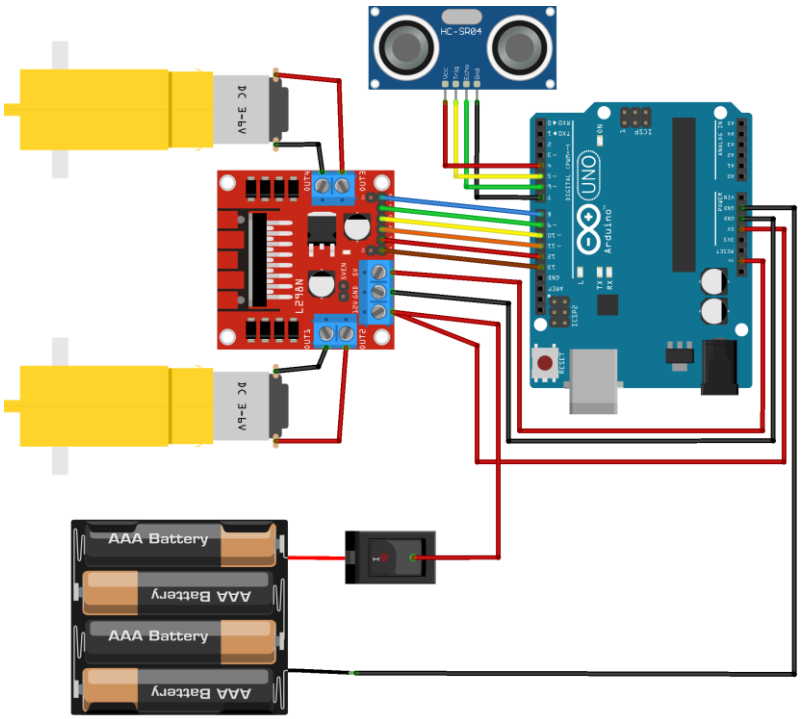


MONTANDO O CIRCUITO

Etapa 06: Conecte agora os pinos para controle do motor A do drive de ponte H, EN1, IN1 e IN2, aos pinos 13, 12 e 11 do Arduino. Em seguida, conecte agora os pinos para controle do motor B do drive de ponte H, IN3, IN4 e EN2, aos pinos 10, 9 e 8 do Arduino.



Etapa 07: Por fim, conecte o sensor ultrassônico ao Arduino nos pinos 7,6,5 e 4.



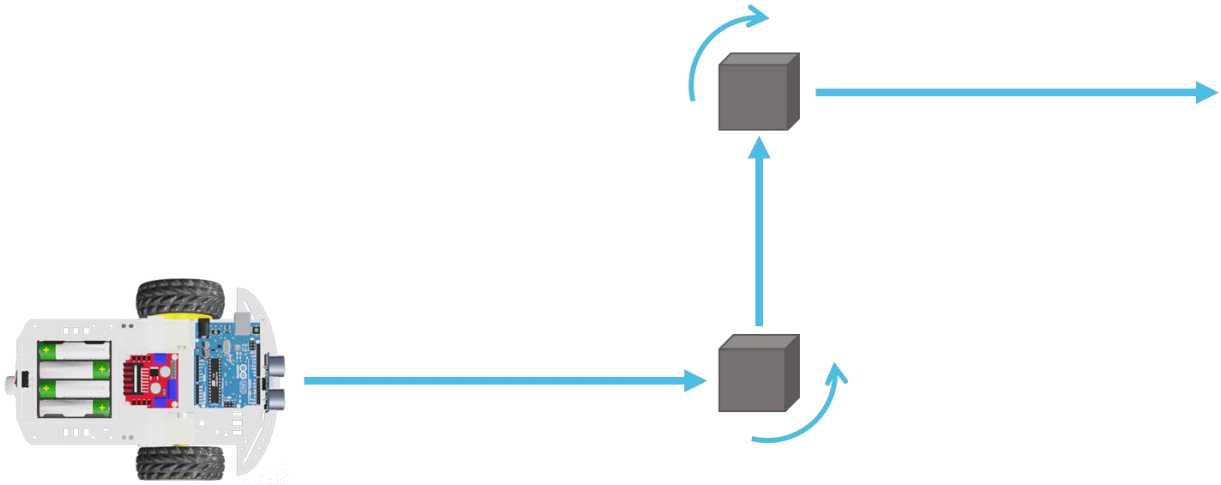
MONTANDO O CIRCUITO

Etapa 08: Antes de alimentar o circuito, confira todas as conexões realizadas.

- **Pino preto** e vermelho do motor DC A conectado ao pino OUT1 e OUT2 do Módulo Ponte H.
- **Pino preto** e vermelho do motor DC B conectado ao pino OUT3 e OUT4 do Módulo Ponte H.
- Pino vermelho de saída do suporte de pilhas conectado ao pino 12V do Módulo Ponte H.
- **Pino preto** de saída do suporte de pilhas conectado ao pino GND do Arduino.
- Pino GND do Arduino conectado por um **jumper preto** ao GND do Módulo Ponte H.
- Pino 5V do Arduino conectado por um jumper vermelho ao 5V do Módulo Ponte H.
- Pino 5Vdo Arduino conectado por um jumper vermelho ao 12V do Módulo Ponte H.
- Pino 13 do Arduino conectado por um jumper marrom ao ENA do Módulo Ponte H.
- Pino 12 do Arduino conectado por um jumper vermelho ao IN1 do Módulo Ponte H.
- Pino 11 do Arduino conectado por um jumper laranja ao IN2 do Módulo Ponte H.
- Pino 10 do Arduino conectado por um jumper amarelo ao IN3 do Módulo Ponte H.
- Pino 9 do Arduino conectado por um jumper verde ao IN4 do Módulo Ponte H.
- Pino 8 do Arduino conectado por um jumper azul ao ENB do Módulo Ponte H.
- Pino Vin do Sensor conectado por um jumper vermelho ao pino 7 do Arduino.
- Pino Trig do Sensor conectado por um jumper amarelo ao pino 6 do Arduino.
- Pino Echo do Sensor conectado por um jumper verde ao pino 5 do Arduino.
- Pino GND do Sensor conectado por um **jumper preto** ao pino 4 do Arduino.
- Conector que Ativa 5V da placa retirado.

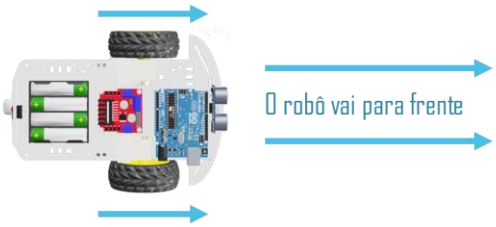
CRIANDO O PROGRAMA

Antes de iniciar a programação pense vamos entender como o kit robótico deve funcionar de forma autônoma. Na prática, ele deve se mover em frente e, ao detectar algum obstáculo, desviar desse obstáculo e seguir em frente.

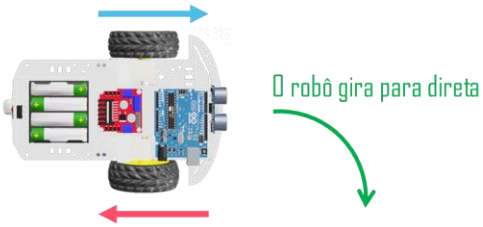


Toda a movimentação do kit robótico depende do sentido de rotação dos motores, de acordo com a rotação o kit pode ir em frente, para trás, ou girar para o lado direito ou esquerdo, como na imagem:

Dois motores no sentido direto

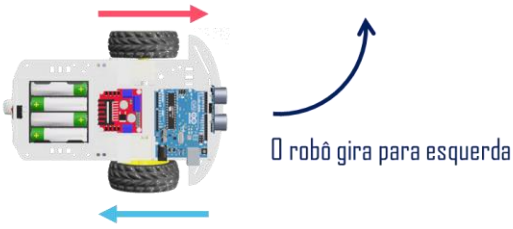


Motor A no sentido direto e Motor B inverso



Dois motores no sentido inverso

Motor A no sentido inverso e Motor B direto

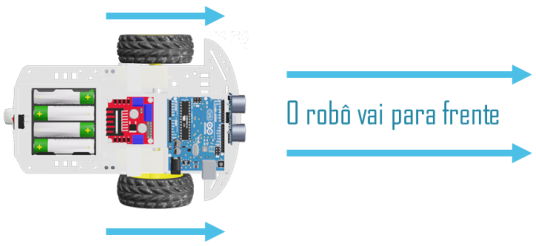


CRIANDO O PROGRAMA

Etapa 09: Vamos iniciar a nossa programação aproveitando as funções que criamos no Projeto 14, para controlar os dois motores.

Etapa 10: Nosso desafio é fazer o robô andar para frente e em caso de obstáculo a menos de 20 cm desviar. Vamos começar fazendo com que ele ande para frente, para isso temos que definir os dois motores como indo no sentido direto. Carregue a programação para o Arduino e teste se os dois motores estão girando para o mesmo lado.

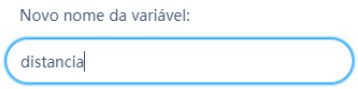
Dois motores no sentido direto



Caso os motores estejam girando para lados opostos, mude o valor dos pinos de controle de direção no motor que está indo para trás. Troque apenas o nome na condicional, onde está "direta" troque por "inversa" e no "inversa" troque por "direta". Realize essa mudança apenas para o motor que está indo no sentido errado e teste novamente seu código.

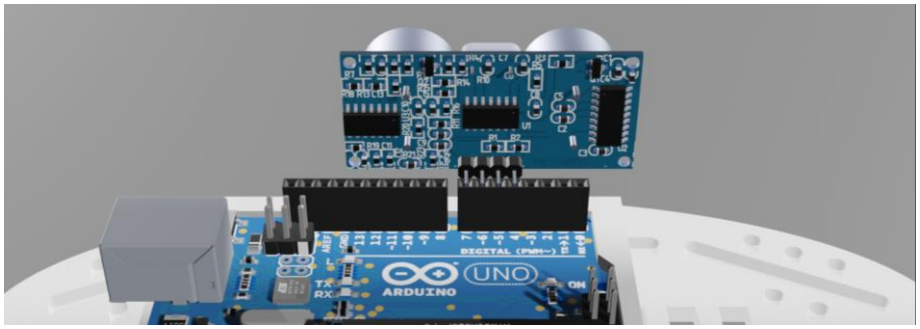
CRIANDO O PROGRAMA

Etapa 11: Como nosso robô já anda para frente nosso desafio agora é fazer o robô detectar obstáculos a menos de 10 cm e desviar. Para isso vamos utilizar uma programação semelhante a que usamos no Projeto 11, criando uma variável distância e definindo seu valor como sendo a distância medida pelo sensor ultrassônico.

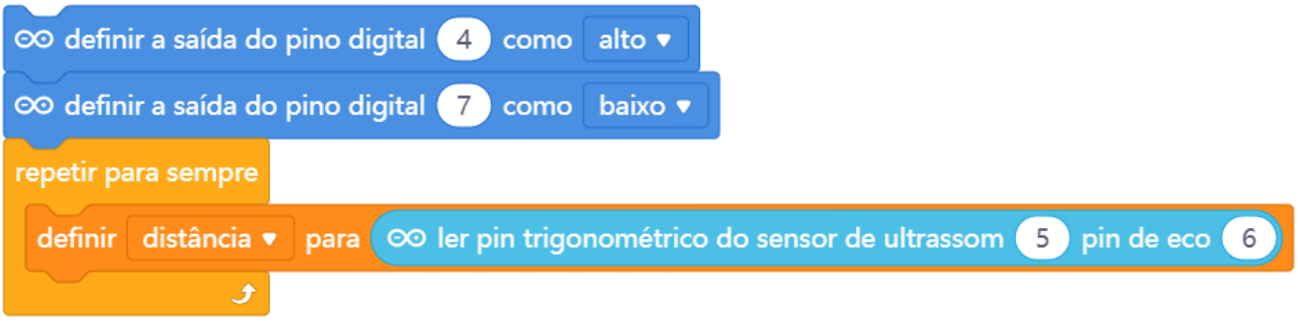


Para que esse valor seja atualizado, temos que inserir esse dentro de uma estrutura de repetição.

Etapa 12: Como o sensor ultrassônico foi conectado diretamente na placa Arduino UNO, os pinos de alimentação Vcc e GND, estão conectados a pinos digitais da placa Arduino, nos pinos 4 e 7, respectivamente.



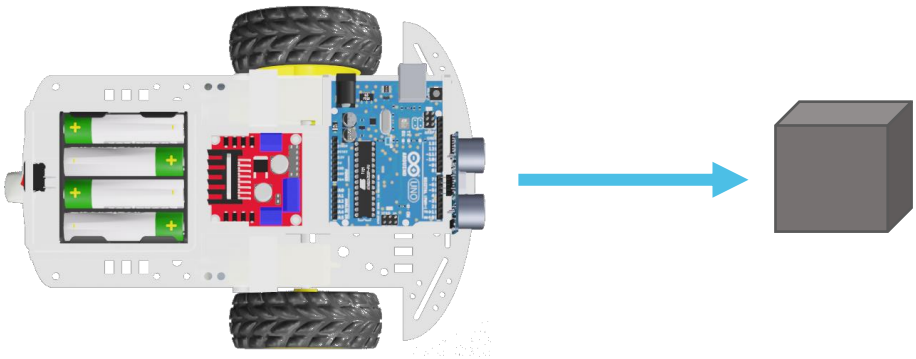
Para que seja garantida a alimentação do sensor, temos então que definir a saída digital do pino 4 como "alta" e do pino 7 como "baixa".



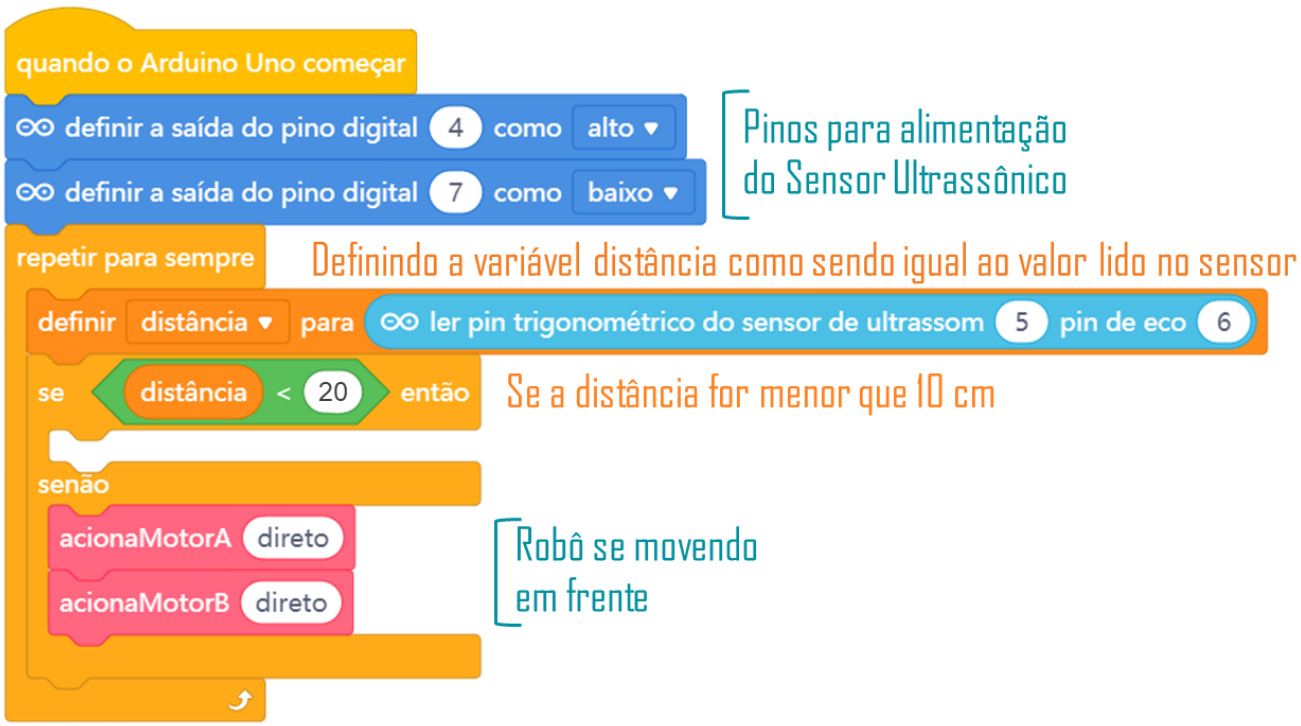
Vamos juntar ao código para controle do motor.

CRIANDO O PROGRAMA

Etapa 13: Resta-nos, agora, desviar dos obstáculos, quando o robô estiver a uma distância menor que 20 cm. Para isso, vamos começar avaliando se a distância é menor que 20 cm, caso a distância seja menor vamos programar para o nosso robô fazer o giro, caso não seja vamos manter o robô indo em frente.



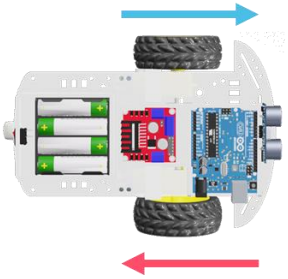
Juntando tudo o código deve ficar como o abaixo:



CRIANDO O PROGRAMA

Etapa 14: Agora que já conseguimos detectar quando há um obstáculo a frente do robô, falta apenas ensiná-lo a desviar do objeto, fazendo uma curva.

Motor A no sentido direto e Motor B inverso



O robô gira para direta

O código deve ficar como o abaixo:

Confira se todos os comandos estão sendo realizados de forma correta, olhe para o seu circuito e confirme se os motores estão ativados e o robô está se movendo no mesmo sentido, ao detectar um obstáculo, giram em sentidos opostos.

AGORA É A SUA VEZ

Faça com que o kit robótico, ao detectar um obstáculo, gire 90° e caso continue encontrando um obstáculo, gire 180° no sentido oposto. O ângulo pode ser definido de acordo com um tempo de espera após começar a rotação.

Não funcionou? Confira novamente todas as ligações, além disso, confira o valor dos pinos digitais no bloco, e se o Arduino está conectado com o mBlock e no modo "Carregar". Caso esteja tendo alguma dificuldade na gravação, desconecte o cabo USB e conecte novamente, gravando assim que inserir o cabo no computador.

MINIBIOGRAFIA DO ORGANIZADOR

Professor de Química, mestre e doutor em Química Analítica, atua principalmente na área de instrumentação analítica. Gambiarreiro, ou melhor, Maker de carteirinha, construiu sua primeira impressora 3D há mais de 10 anos e desde então atua como divulgador e incentivador da cultura maker, principalmente no ambiente escolar.

INSTITUTO FEDERAL
Sertão Pernambucano

EDITORA
IFSertãoPE