



# APRENDA ARDUINO DO ZERO

E SE DESAFIE COM 10  
PROJETOS DIFERENTES

BY FLÁVIO BABOS

# SUMÁRIO

PROJETO	COMPLEXIDADE	PÁGINA
1. PRIMEIROS COMPONENTES	■ ■ ■ ■ ■	13
2. ACENDENDO 3 LED'S	■ ■ ■ ■ ■	32
3. ACIONANDO O SERVO MOTOR	■ ■ ■ ■ ■	50
4. LED RGB E SENSOR DE LUZ	■ ■ ■ ■ ■	70
5. MEDIDOR DE TEMPERATURA	■ ■ ■ ■ ■	81
6. SPEAKER COM BUZZER	■ ■ ■ ■ ■	89
7. MINI TECLADO MUSICAL	■ ■ ■ ■ ■	101
8. CONTROLE DE UM MOTOR DC	■ ■ ■ ■ ■	111
9. LIGANDO UM DISPLAY LCD	■ ■ ■ ■ ■	121
10. FECHADURA ELÉTRICA	■ ■ ■ ■ ■	135



A **plataforma Arduino** facilita ao máximo a programação de pequenos computadores que na nomenclatura são os chamados de microcontroladores aos quais fazem a interação com os objetos no mundo exterior ao da programação, neste caso, C++ que você irá aprender muito sobre neste Arduino PDF.

Você está cercado de dezenas de microcontroladores todos os dias: estão embutidos em temporizadores, termostatos, brinquedos, controles remotos, micro-ondas, e até em escovas elétricas. Sendo assim, o objetivo de sua atuação na maioria dos dispositivos é sentir e controlar as atividades usando sensores e atuadores.

Basicamente, **sensores** escutam o mundo físico. Eles convertem energia que lhes é fornecida através do acionamento de um botão ou da identificação de energia solar no ambiente em sinais elétricos. **Botões**, neste caso, são **interruptores** que possuem como funcionalidade comandar um circuito elétrico a partir da modificação da sua situação de comutação (ON/OFF). No caso de nossos Projetos com Arduino aqui do PDF, esses botões são usados para ligar e desligar um LED, acionar um motor de 5V, ligar um buzzer, entre outras funções.

Já os **atuadores**, como o próprio nome já diz, atuam no mundo físico. Eles convertem a energia elétrica em energia física, como luz, temperatura e movimento.

Enquanto isso, os **microcontroladores** representam a intermediação entre os sensores e os atuadores. São eles que decidem o que fazer baseado no programa que você escrever.

Os microcontroladores e os componentes eletrônicos são o esqueleto dos seus projetos. Eles são o **hardware** que darão forma aos seus projetos enquanto que o **software** será os códigos escritos por você e processados pelos microcontroladores.

## SOBRE O AUTOR



### FALE COMIGO

 [contato@flaviobabos.com.br](mailto:contato@flaviobabos.com.br)

 [flaviobabos.com.br](http://flaviobabos.com.br)

 [Flávio Resende Babos](https://www.linkedin.com/in/Flávio_Resende_Babos)

 [@babosengenharia](https://www.instagram.com/babosengenharia)

### HABILIDADES

- Gerência de Projetos
- Vendas e Marketing
- Modelagem 3D
- Proc. Empresariais
- Arduino
- Negócios
- Lean Six Sigma

# FLÁVIO BABOS

## PERFIL PROFISSIONAL

Sou um apaixonado por tecnologia e desde pequeno faço projetos de robótica com ênfase em Arduino. Além disso, fui Gerente de Vendas e de Projetos na Empresa Júnior Meta Consultoria da Engenharia Mecatrônica da Universidade Federal de Uberlândia - UFU.

## EXPERIÊNCIA DE TRABALHO

### Gerência de vendas

Meta Consultoria | ABR/2018 - JAN/2019

- Gerente comercial e consultor de vendas da empresa por durante 4 meses;
- Gerente de marketing por durante 5 meses.

### Gerência de projetos

Meta Consultoria | FEV 2019 - DEZ 2019

- 4 projetos como Gerente de Projetos na área de Acústica, modelagem tridimensional com SolidWorks e Análise de Material;
- 3 projetos como Consultor de Projetos na área de Eletrônica/Arduino e modelagem tridimensional com SolidWorks.

## SOBRE ESTA APOSTILA

Ao final dessa apostila em formato Arduino PDF com **10 projetos**, você irá aprender finalmente como construir seus próprios circuitos com a plataforma e a escrever sua própria programação, mas com um detalhe, sabendo o que está copiando e entendendo a lógica dos algoritmos.

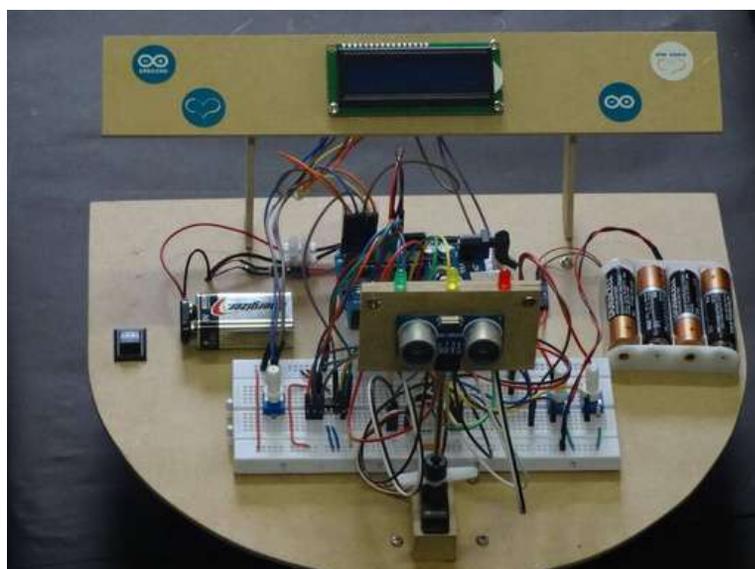
Se você ainda não me conhece, eu atuo com o Arduino desde 2014, quando descobri a plataforma no Laboratório de Automação e Robótica da UFV-CRP.



Essa foto representa um projeto de um **carro autônomo** desenvolvido em equipe durante o colegial que ganhou destaque na Universidade da minha cidade: a UFV-CRP.

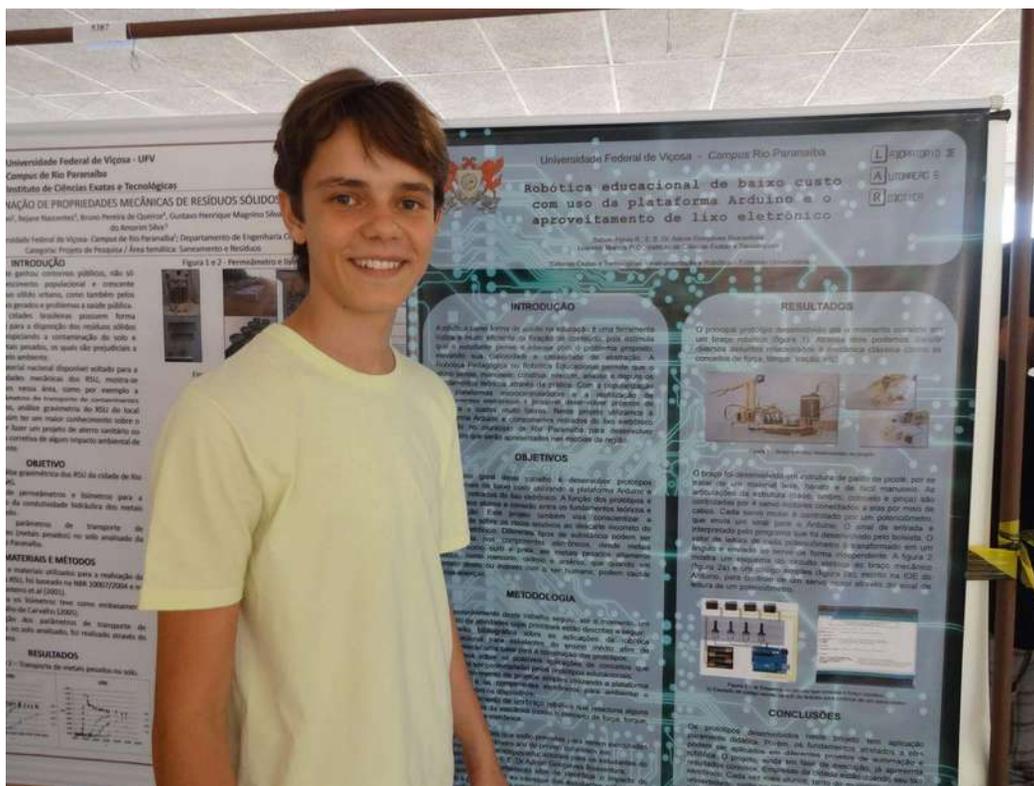


Projeto final do carro desenvolvido com objetivo de evitar os obstáculos com uso do sensor ultrassônico, servo motor, 2 motores DC, LED's e LCD para informarem a proximidade.



## SOBRE ESTA APOSTILA

Já essa segunda foto, representa um **braço robótico** de baixo custo com Arduino. Também ganhou destaque na Universidade e isso me ajudou a conseguir uma bolsa para continuar o desenvolvimento do projeto para ser usado na sala de aula no ensino à robótica.



Desde essa época, eu não parei de fazer pequenos projetos com Arduino pois fiquei apaixonado pela área de eletrônica e de robótica.

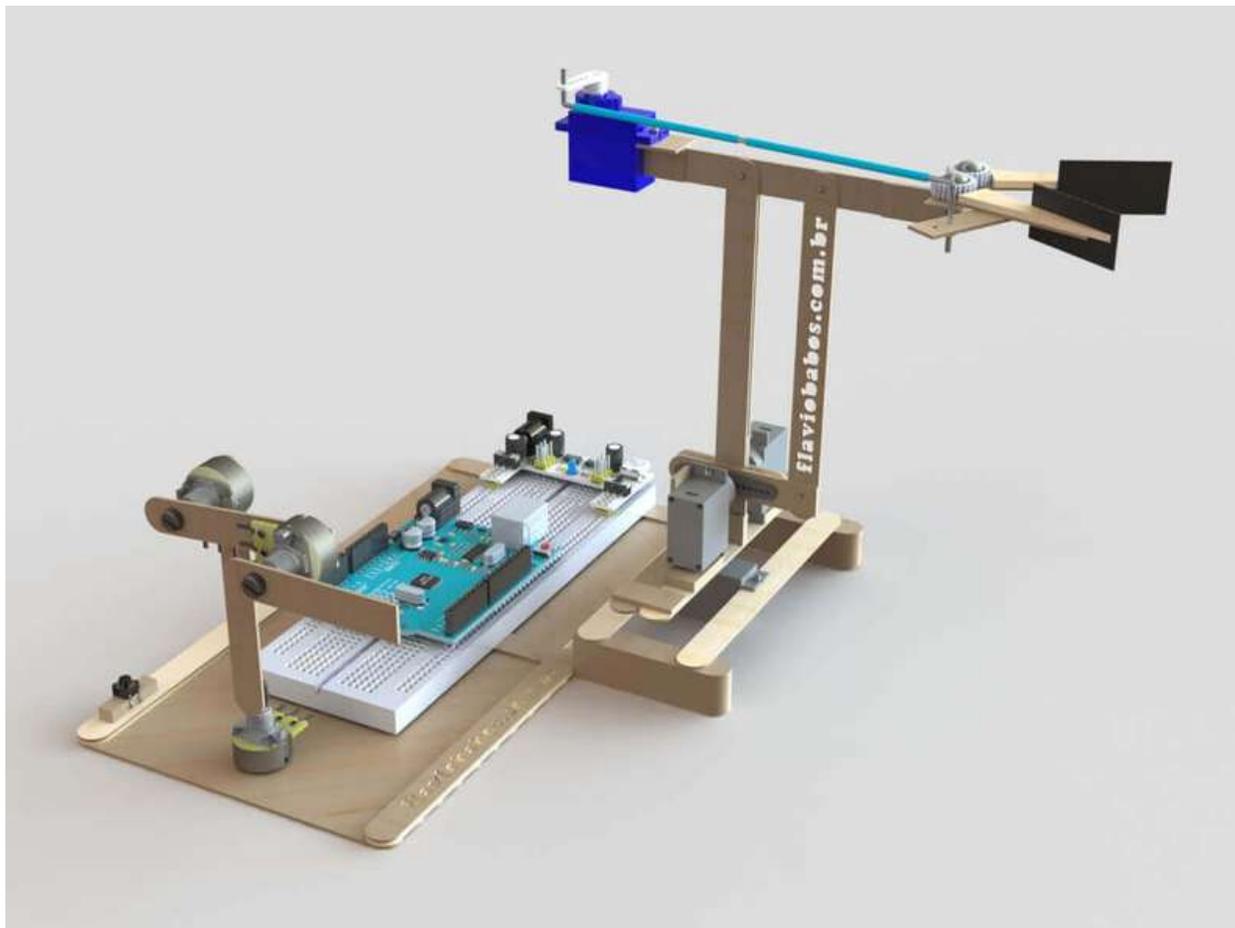
Inclusive é engraçado eu te contar isso pois minha paixão era tamanha que passava tardes inteiras no Laboratório de Robótica dando vida a projetos que sem o Arduino talvez não seria possível de criar...

Um dos projetos que mais gostava era o braço robótico.

Minha ideia com esse projeto era construir um mecanismo que fosse diferente e que fosse utilizado no ensino de robótica em minha escola.

## SOBRE ESTA APOSTILA

Veja só:



Ficou curioso(a)?

Então acesse: [como construir um braço robótico do zero](#) para saber mais sobre esse invento!

A parte estrutural desse projeto eu construí sem muitos problemas na época. O que complicava era a programação na época, não sabia quase nada.

Então, sabe o que eu fiz?

Fui copiando projetos que via na internet e seus códigos para ver se funcionava no meu braço robótico..., mas, não conseguia por nada colocar meus servos motores, o joystick e o Arduino funcionarem em conjunto.

Até que me dei por vencido e resolvi aprender a lógica por trás de cada linha dos projetos que estava copiando...

As coisas começaram a fazer sentido conforme iria aprendendo sobre **funções condicionais**, sobre **funções de conversão de valores**, sobre **bibliotecas**...

O que antes parecia muitas linhas sem sentido nos projetos que simplesmente só copiava, agora eu já entendia e sabia qual lógica escrever para fazer meu projeto funcionar.

E, pode parecer algo simples, mas quem já passou por tentar achar um erro de compilação na IDE do Arduino e não descobrir sabe do que estou falando...

E foi a partir de muitos outros projetos e estudos que fiz, que aprendi, de fato, como desenvolver meu próprio projeto com Arduino ao invés de ficar copiando de outras pessoas sem entender a lógica da programação.

Dessa forma, esse livro digital é o reflexo de toda a experiência que adquiri durante esses anos com o Arduino e que estou te dando de graça e de forma exclusiva para você não precisar passar pelas mesmas dificuldades que eu passei.

Sei que isso pode parecer bom demais para ser verdade. **Mas**, se eu tivesse esse conteúdo que estou te passando lá em 2014, com certeza teria sido muito mais fácil.

Ademais, não precisa se preocupar pois não estou falando de você copiar projetos e códigos que você não entende a lógica.

Ao invés disso, o que vou te mostrar é algo que realmente funciona e que vai te ajudar a chegar em um nível em que você vai construir seus próprios projetos sem precisar de cópias.

### **Você finalmente irá:**

- ✔ Aprender de vez sobre programação C++;
- ✔ Se aprofundar em eletrônica e ampliar seus conhecimentos em circuitos com Arduino;
- ✔ Criar seus próprios projetos.

### **Sem precisar de:**

- ✘ Ficar horas copiando códigos que não sabe programar ou que não entende a lógica;
- ✘ Imitar projetos das outras pessoas;
- ✘ Pagar por cursos sem qualidade que não vão resolver seus problemas.

### **E não importa se:**

- Você é um completo iniciante;
- Você tem poucas horas por dia para aplicar;
- Você tentou de tudo e não conseguiu.

Não precisa se preocupar, o método que está dentro desta apostila está estruturada de uma forma bem didática.

**Basta seguir o plano para ter 100% de sucesso!**

**PS: Para aprender tudo sobre esse Universo Open Source que é o Arduino e acabar de uma vez por todas sua aflição com programação, [clique aqui!](#)**

# PLACA ARDUINO UNO

**Conector de energia** aceita 7-12V.

## Porta USB

energizar o arduino, dar upload nos códigos e fazer comunicação serial (via `Serial.println()` etc).

**Botão de Reset** do microcontrolador.

**LED's TX e RX** indicam comunicação entre o Arduino e seu computador. Eles ascendem quando há upload de código e durante a comunicação serial.

**LED pino 13** é a única led construída no seu Arduino Uno.

**Pinos Digitais**, use esse pinos com `digitalRead()`, `digitalWrite()` e `analogWrite()`. Esse último se aplica apenas às portas de símbolo PWM.

**Microcontrolador ATMEGA.**

**Pinos GND, 5V e 3.3V** energize seus circuitos através desses pinos.

**Power LED**, indica que o Arduino está ligado.

**Analog in**, use esse pinos com `analogRead()`.

# ESTÁ PREPARADO (A) PARA POR A MÃO NA MASSA?

Então, chegou a hora de aprender a programar e a montar seus próprios projetos com os **10 exemplos didáticos e ilustrativos** aqui no PDF como prometido!

Se divirta com o que você irá desvendar nas próximas páginas!

**APROVEITE!**

# TIRE SUAS IDEIAS DO PAPEL!

E CRIE SEUS PRÓPRIOS  
PROJETOS COM  
ARDUINO TAMBÉM!



CURSO DE ARDUINO

CLIQUE AQUI



LET'S  
DO  
IT

# PROJETO 1



SWITCH



LED



RESISTOR DE 220 OHM

COMPONENTES UTILIZADOS

# SEUS PRIMEIROS COMPONENTES

Você irá fazer um circuito simples com alguns botões, um led e um resistor!

Descubra neste projeto: teoria básica de eletrônica, como funciona uma protoboard, componentes em série e em paralelo.

TEMPO: 30 minutos

NÍVEL: 

**Eletricidade** é um tipo de energia, como o calor, a gravidade ou a luz. A energia elétrica flui através de condutores como fios. Você pode converter a energia elétrica em outras formas de energia para fazer algo interessante como ligar uma luz/LED ou fazer barulho com um auto-falante.

Os componentes que você deverá usar para fazer isso, como auto-falantes ou lâmpadas, são **transdutores elétricos**. Transdutores transformam outros tipos de energia em energia elétrica e vice-versa. Componentes que convertem outras formas de energia em energia elétrica são chamados de **sensores**, e componentes que convertem energia elétrica em outras formas de energia são chamados de **atuadores**.

Em um circuito, eletricidade flui de um ponto de uma grande quantidade de energia potencial (geralmente é denotado como power ou +) para um ponto de energia potencial menor.

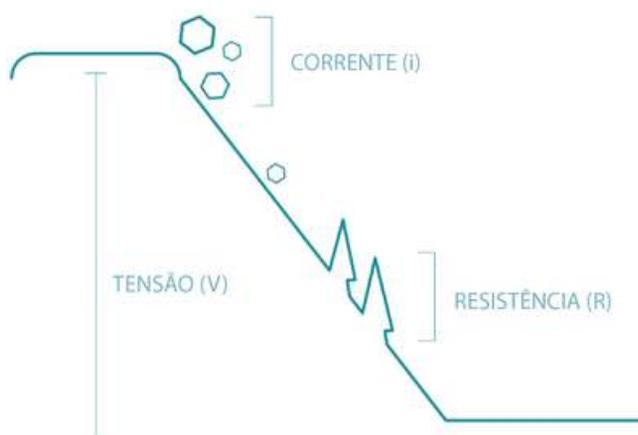
## PROJETO 1

**Ground** (geralmente representado por GND ou -) é geralmente um ponto com a menor energia potencial em um circuito. Nos circuitos que você irá montar, a eletricidade fluirá somente em uma direção. Esse tipo de circuito é chamado de **corrente contínua** ou **DC**.

Nos circuitos de **corrente alternada** (**CA**), a eletricidade muda de direção 50 ou 60 vezes por segundo (dependendo de onde você mora). Isso é o tipo de eletricidade que vem de uma tomada de parede.

Existem alguns termos com os quais você deve se familiarizar ao trabalhar com circuitos. **Corrente** (medida em ampères; com o símbolo A) é a quantidade de carga elétrica que flui para além de um ponto específico do seu circuito. **Tensão** (medida em volts; com o símbolo V) é a diferença de energia entre um ponto em um circuito e outro. E, finalmente, a **resistância** (medida em ohms; com o símbolo  $\Omega$ ) é o quanto um componente resiste ao fluxo de corrente elétrica.

Uma maneira de imaginar isso é pensar em um deslizamento de rocha indo ladeira abaixo em um penhasco, como mostrado ao lado. Quanto mais alto o penhasco, mais energia as rochas terão quando atingirem o fundo.



## PROJETO 1

A **altura** do penhasco é como a tensão em um circuito: quanto maior a tensão na energia da fonte, mais energia você tem que usar.

Quanto mais pedras você tem, mais energia será transportada pelo penhasco. O **número de rochas** é como a corrente em um circuito elétrico. As pedras vão através de arbustos ao lado do penhasco, perdendo um pouco de energia no processo; a energia é usada para esmagar os arbustos.

Os **arbustos** são como resistores em um circuito, oferecendo resistência à eletricidade fluir e convertê-la em outras formas de energia.

### Algumas coisas sobre circuitos

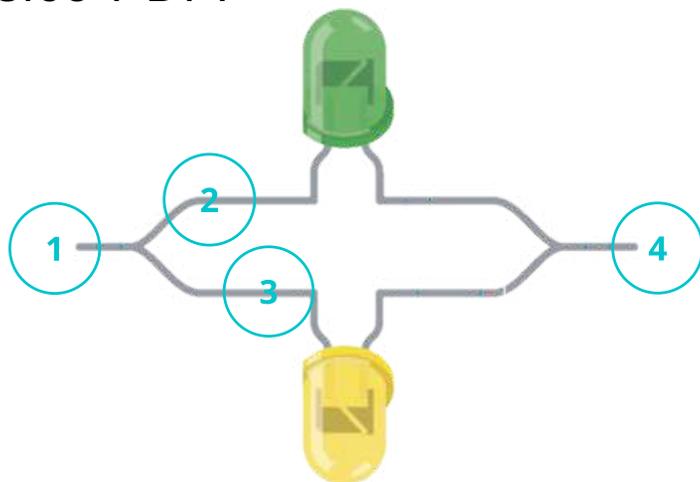
**1)** Precisa haver um caminho completo a partir da fonte de energia (potência) até o ponto de menor energia (terra) para fazer um circuito. Se não houver caminho para a energia viajar, o circuito não funcionará.

**2)** Toda a energia elétrica é consumida em um circuito pelos componentes nele. Cada componente converte parte da energia em outra forma de energia. Em qualquer circuito, toda a tensão é convertida para outra forma de energia você irá notar isso aqui no Livro Arduino PDF.

**3)** O fluxo de corrente em um ponto específico de um circuito sempre será o mesmo entrando e saindo.

## PROJETO 1

4) A corrente elétrica buscará o caminho de menor resistência até o ground. Se você tem uma ligação que conecta o power e o GND sem resistência, você causará um curto-circuito. Em um curto-circuito, a fonte de alimentação e os fios convertem a energia elétrica em luz e calor, geralmente como faíscas. Se você já deu um curto-circuito na bateria e viu faíscas, você sabe o quão perigoso um curto-circuito pode ser. Não se preocupe pois você irá aprender definitivamente a construir seus circuitos de forma segura neste Arduino Básico PDF.



corrente 1 = corrente 2 + corrente 3 = corrente 4

## O que é uma protoboard?

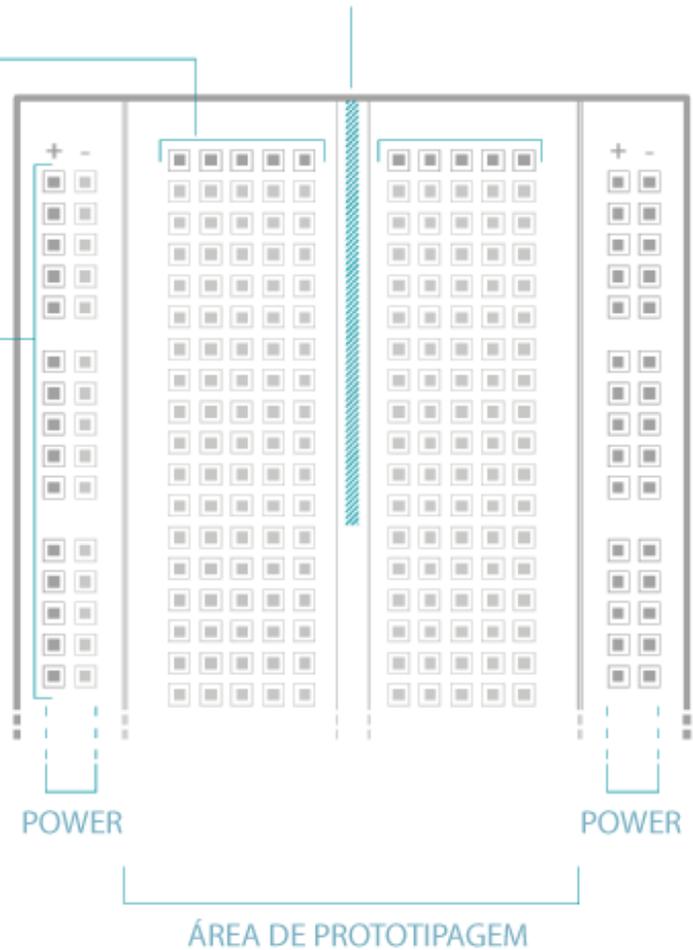
Uma protoboard é a ferramenta principal onde você irá construir os circuitos. Geralmente a que compramos em kits de arduino não precisa soldar os componentes. As linhas verticais e horizontais da protoboard, como mostrado na próxima página, são interligadas e a eletricidade pode fluir em conectores de metais bem finos sob o plástico com furos. Veja a imagem na página seguinte do Arduino PDF.

# PROJETO 1

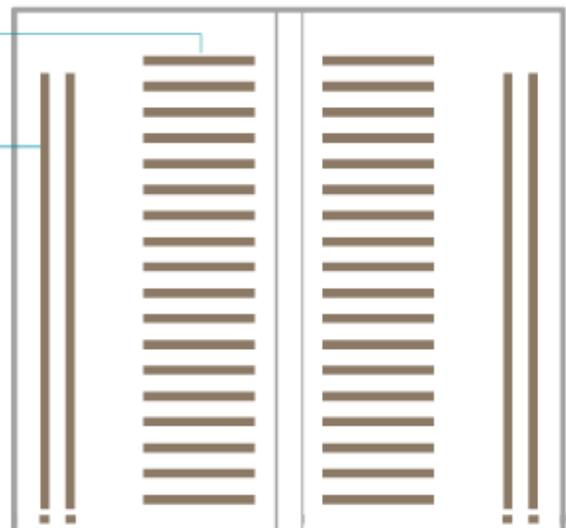
Os 5 furos em cada linha horizontal são conectados eletricamente através de tiras de metal dentro da protoboard.

A linha do meio interrompe a conexão entre os dois lados da placa.

As faixas verticais que percorrem o comprimento da protoboard estão conectadas eletricamente. As tiras são geralmente usadas para conexões de energia e terra.



Tiras de metal condutivo



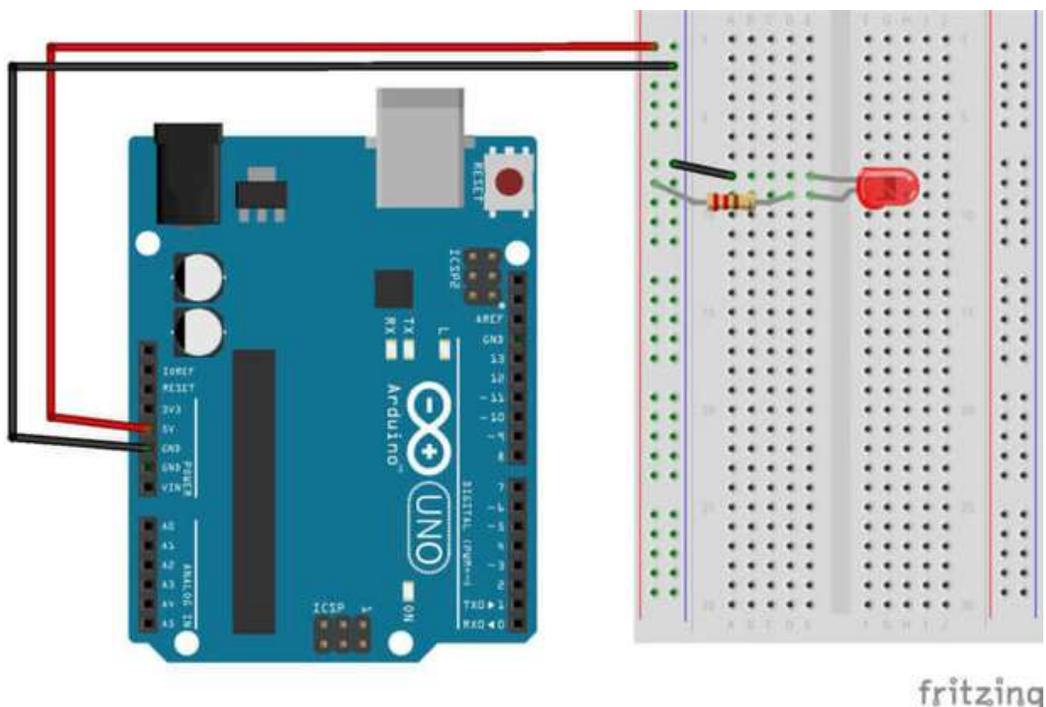
# Esquema do circuito

Ao longo deste guia você irá se deparar com 2 imagens dos Projetos Prontos PDF com Arduino: a **montagem dos componentes do projeto** e o **esquema elétrico da montagem** que é usado para representar as conexões entre os componentes de um circuito.

A vista esquemática não mostra sempre onde os componentes são inseridos na protoboard, mas ela mostra como eles são conectados entre si.

As imagens esquemáticas e do projeto montado são elaboradas com a ajuda do software Fritzing, um software de iniciativa open-source.

Você pode fazer o download dele [aqui](#):



## Montagem do projeto

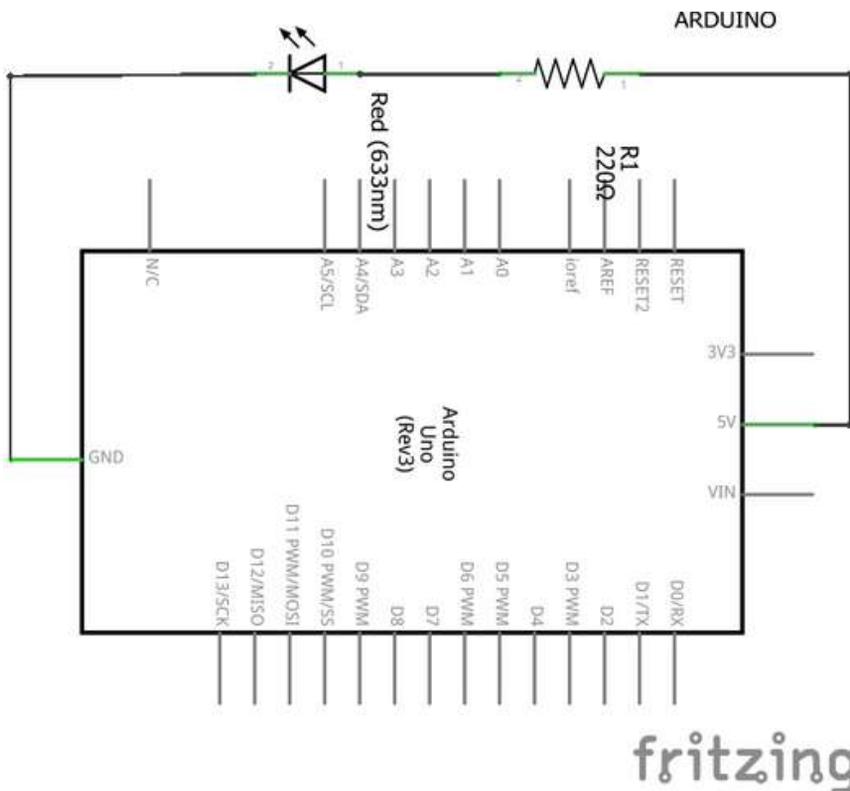
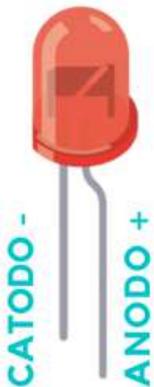


Ilustração esquemática do projeto

## Seus primeiros componentes



O **LED** ou **light-emiting diode**, sigla em inglês, é um componente que **converte** energia elétrica em luz. Eles são polarizados como mostra a imagem ao lado. A perna menor é o catodo e conecta ao ground e a maior é o anodo que conecta ao power.



Esse é um **resistor** de 220 OHMs. Seu papel é **dificultar** a passagem de corrente elétrica (veja a lista de componentes explicando as listras coloridas e seus valores do **Projeto 2**). Se você ligar esse componente em série com um componente como o LED, ele irá consumir uma quantidade de energia elétrica e dessa forma, você evita que o LED receba uma grande quantidade de energia e, conseqüentemente, o **queime**. Sem o resistor o LED irá brilhar por alguns momentos mas, logo após, irá queimar.

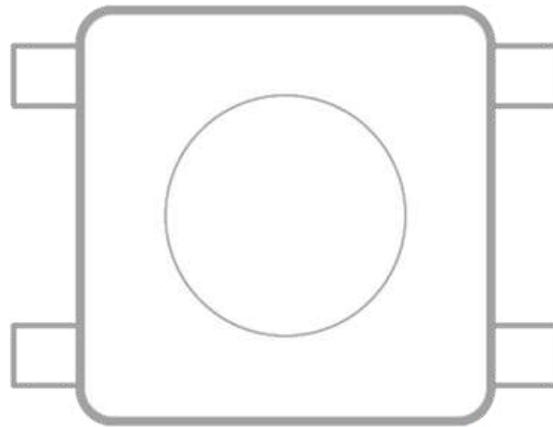


Um **botão** ou **switch** é um componente elétrico que aciona um circuito e o completa quando pressionado. A imagem ao lado representa o modelo mais utilizado em eletrônica e em projetos com Arduino.

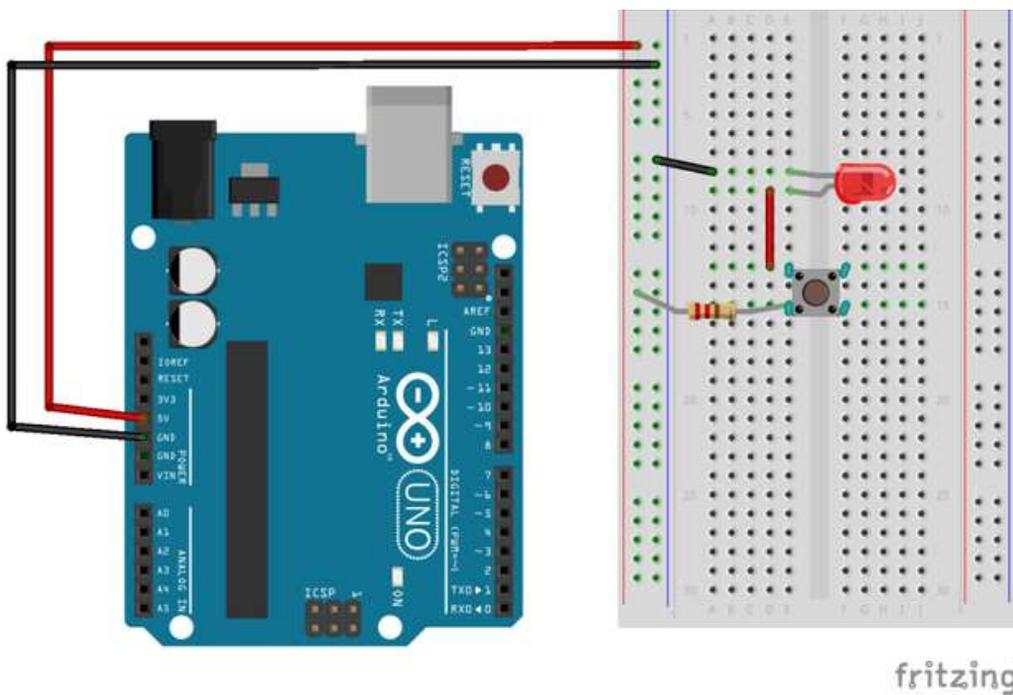
## Conexões do botão

Esses dois pinos estão conectados.

Já esses dois não estão.



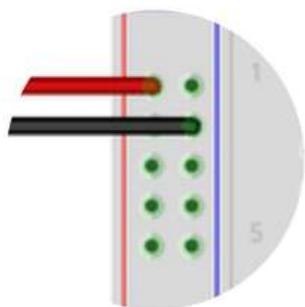
## Montando o circuito



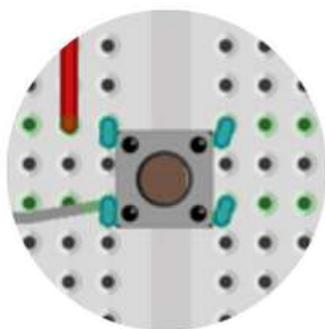
Seu primeiro circuito interativo do seu Arduino PDF, usando um botão, um resistor e um LED

## PROJETO 1

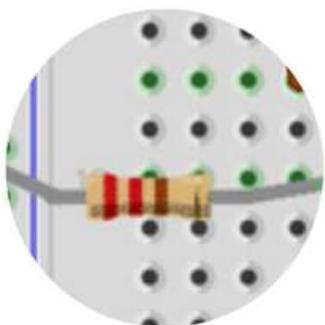
O Arduino está apenas fornecendo energia para esse circuito; em projetos futuros, você irá usar os pinos **input** e **output** do Arduino para controlar circuitos mais complexos.



1) Conecte o fio vermelho ao pino de 5V do Arduino e coloque a outra extremidade em uma das longas linhas da sua protoboard. Conecte o terra do Arduino com um fio preto na outra linha adjacente. É útil manter a cor do fio consistente (vermelho para power, preto para GND) em todo o seu circuito.



2) Agora que está tudo montado, coloque o botão atravessando a sua protoboard. O botão tem um lado para ser inserido, atente-se: a curva nas pernas do botão para o centro da protoboard.



3) Use um resistor de 220 ohms para conectar os 5V de um lado do interruptor. Aqui iremos usar resistores de 4 bandas. Já você deve possuir um mix de 4 a 5 bandas de resistores. Vá para o final do [Projeto 2](#) onde eu faço uma explicação detalhada sobre resistores e seus códigos.

# DIVIRTA-SE!

Basta pressionar o botão que você deverá ver o LED acender.

**Parabéns**, seu primeiro circuito foi montado! O próximo passo é inserir outro botão.

**SE liga!** Agora você irá aprender a colocar componentes em série e em paralelo na protoboard.

## Circuitos em série!

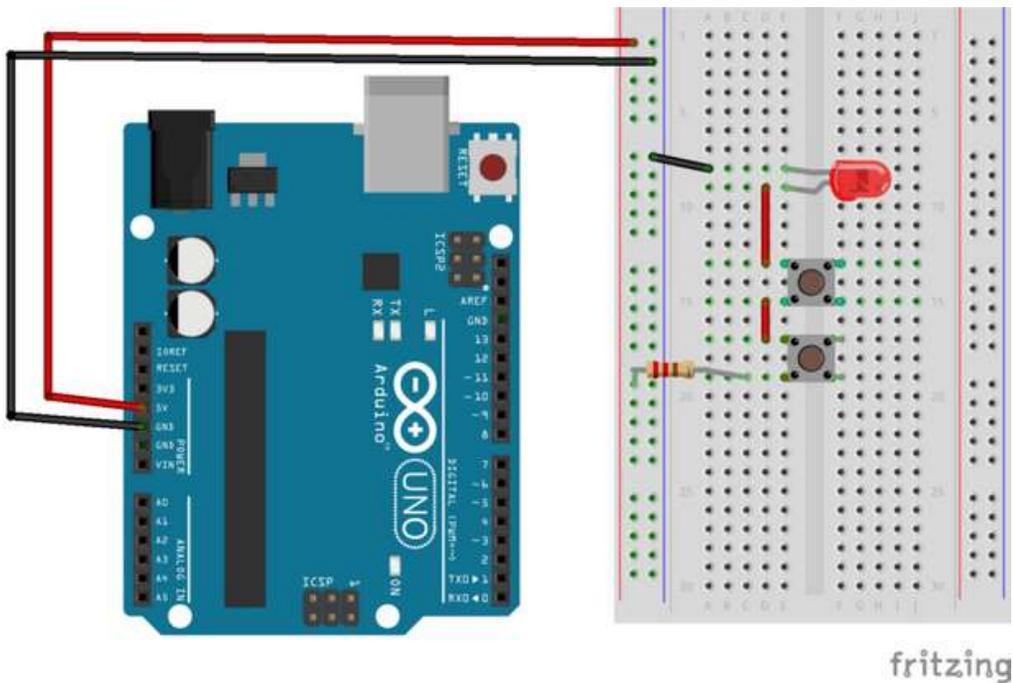
*Circuito em série, os elementos se posicionam um após o outro.*

Após remover o cabo USB ou a fonte do seu circuito, adicione outro botão na montagem feita anteriormente.

Conecte o LED ao catodo (gnd) e ligue o arduino. Agora para ligar o LED você precisa pressionar ambos botões ao mesmo tempo. Isso acontece pois, quando estão em série, ambos precisam fechar o circuito para a eletricidade fluir até o LED e acendê-lo.

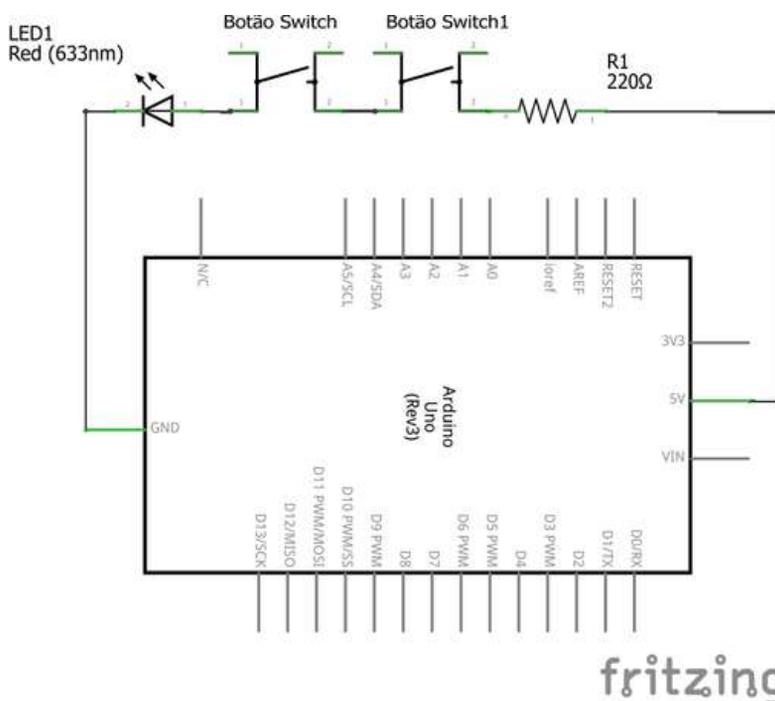
Passa para a próxima página e veja a montagem do circuito.

# PROJETO 1



## Montagem dos componentes do circuito em série

Os dois botões estão em série. Isso significa que a mesma corrente circula através deles. Portanto ambos precisam ser pressionados ao mesmo tempo para acender o LED.

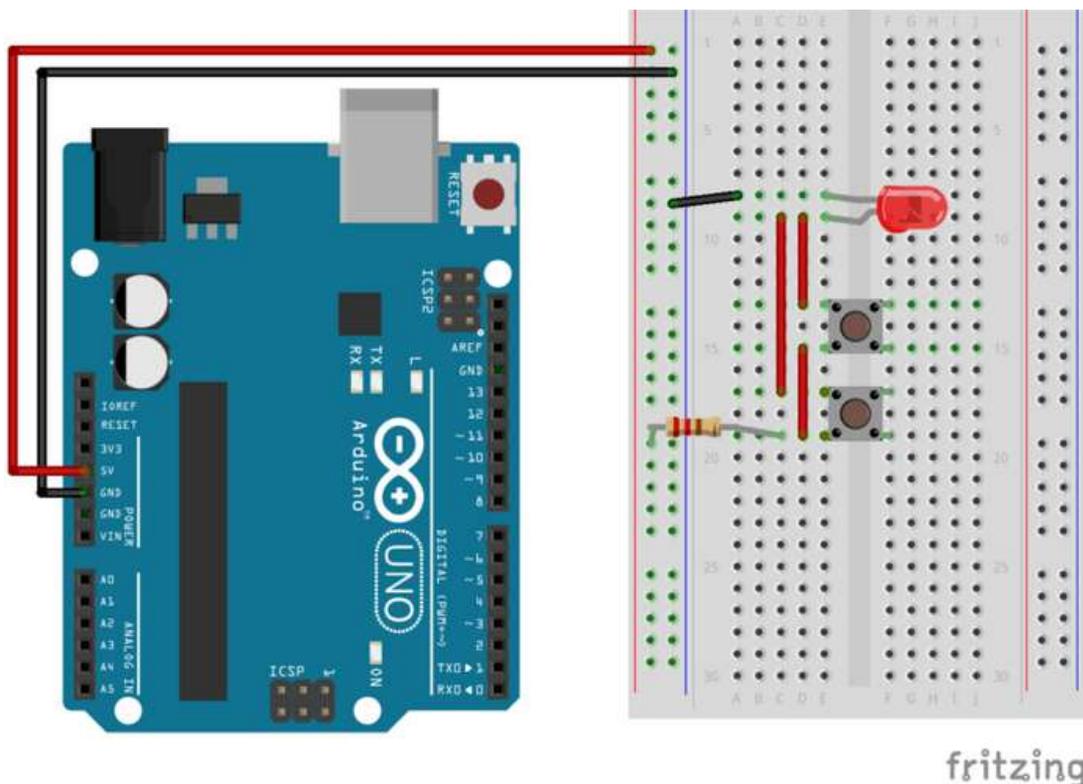


## Montagem esquemática do circuito em série

## Circuitos em paralelo!

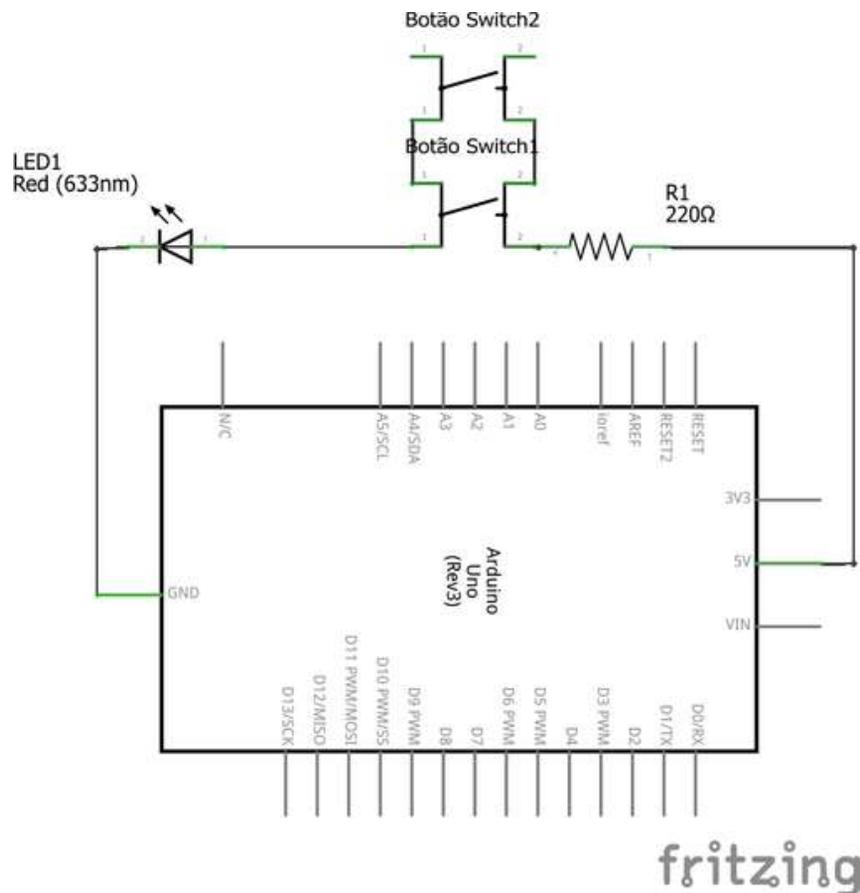
Agora que você já aprendeu o conceito por trás dos circuitos em série, é hora de aprender como montar um circuito em paralelo usando os mesmos componentes anteriores.

Mantenha o botão switch e o LED no mesmo lugar pois, o que irá fazer, será apenas acrescentar um fio que ligue o botão de cima ao LED. Veja figura abaixo:



fritzing

**Montagem dos componentes do circuito em paralelo**

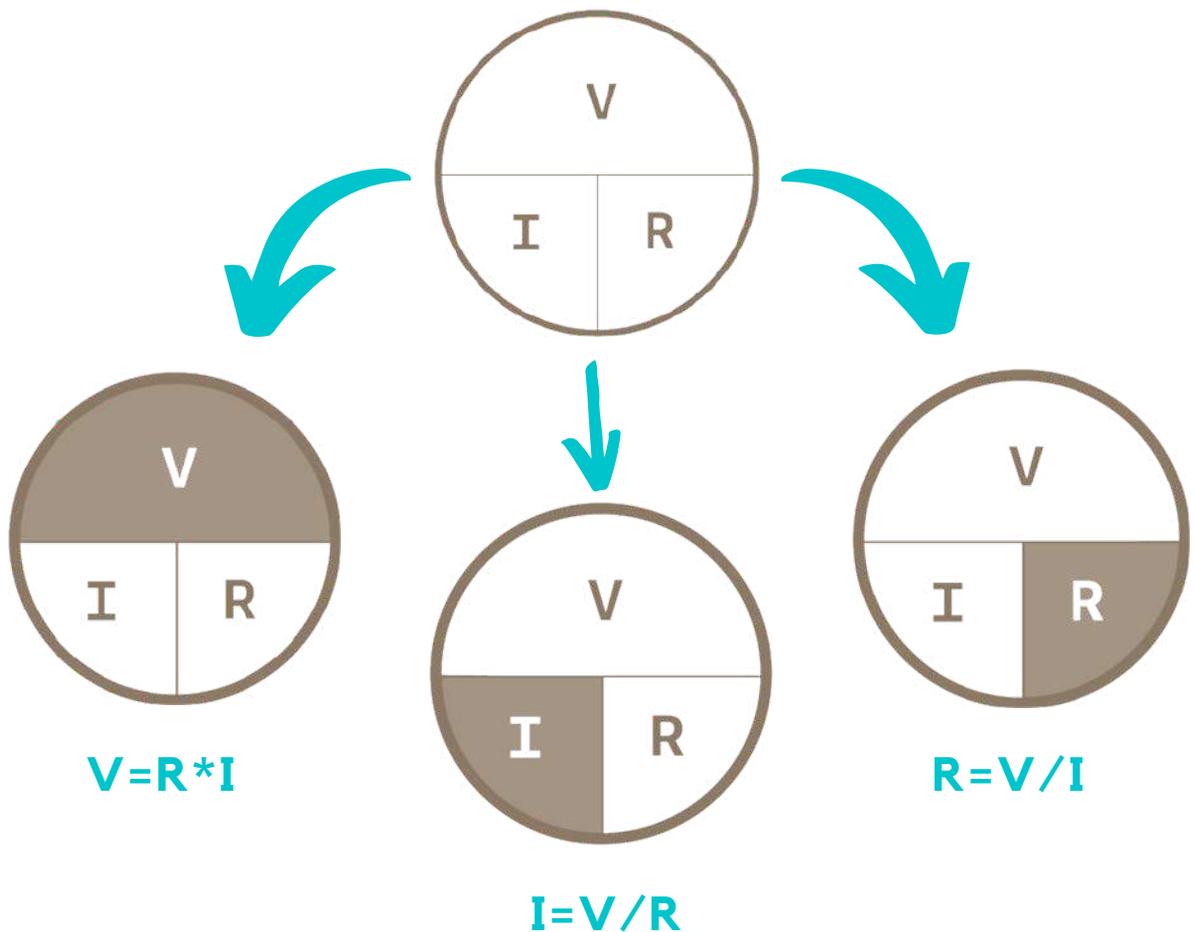


## Montagem esquemática do circuito em paralelo

Já que os botões estão em paralelo, basta pressionar **um dos dois** botões que o LED irá acender. Isso acontece pois, a corrente está dividida e, ao pressionar qualquer botão, ela irá passar e seguir até o LED acendendo-o.

## Entendendo a LEI de OHM

Observe os círculos abaixo para te ajudar a associar as relações entre tensão, corrente e resistência caso ainda não conheça essa Lei.



Observe que corrente, tensão e resistência estão diretamente relacionados. Quando você muda uma dessas variáveis em um circuito, irá afetar o restante. A relação entre eles é conhecida como **Lei de Ohm**, nomeada assim graças aos estudos e descobertas de Geog Simon Ohm.

**TENSÃO (V) = CORRENTE (I) \* RESISTÊNCIA (R)**

## PROJETO 1

No primeiro circuito mostrado nesse primeiro projeto (um LED e um resistor), você está fornecendo 5 volts enquanto que, o resistor oferece 220 ohms de resistência. Para achar a corrente que passa pelo LED, basta substituir os valores na equação:

$$\begin{aligned}5 &= I \cdot 220 \\ I &= 5/220 \\ I &= 0.023 \text{ A}\end{aligned}$$

Ou seja, isso é 23 miliamperes (23mA) usado pelo LED. Esse valor que descobrimos é o **máximo** de corrente que o LED pode usar sem que ele queime, e é por causa disso que usamos um resistor de 220-ohms neste Projeto do PDF com Arduino.

## Desafio

Você pode expandir esse projeto de diversas formas, através da criação de seu próprio botão (dois pedaços de papel alumínio com arame funcionam bem), ou criar uma combinação de interruptores e LED's em paralelo e em série.

- O que acontece quando você insere 3 ou 4 LED's em série?
- O que acontece quando eles estão em paralelo?
- Por que eles se comportam assim?

### Recaptulando...

- Você aprendeu até agora sobre as propriedades de **tensão**, **corrente** e **resistência** enquanto montava os circuitos propostos.
- Com alguns componentes como LED's, resistores e interruptores/botões, você criou um sistema interativo simples: um usuário pressiona o botão e a luz acende.
- Essas diretrizes básicas de trabalho na eletrônica, serão melhor trabalhadas e explicadas nos próximos projetos que estão por vir.

Os próximos 9 projetos que você está prestes a desenvolver, irão abordar melhor a programação do Arduino e você irá aprender bastante coisa interessante que poderá replicar nos seus inventos.

Portanto, se você não possui conhecimento de programação ou já tentou copiar códigos prontos mas não tem sucesso desenvolvendo seu próprio projeto, leia com muita atenção os códigos e as explicações dos mesmos para que você consiga minimizar seus problemas e avançar cada vez mais na programação do Arduino.

Para você saber do que estou falando, acesse agora o [\*\*Projeto 4 LED RGB e Sensor de Luz...\*\*](#)



**TIRE SUAS  
IDEIAS DO  
PAPEL!**

E CRIE SEUS PRÓPRIOS  
PROJETOS COM  
ARDUINO TAMBÉM!

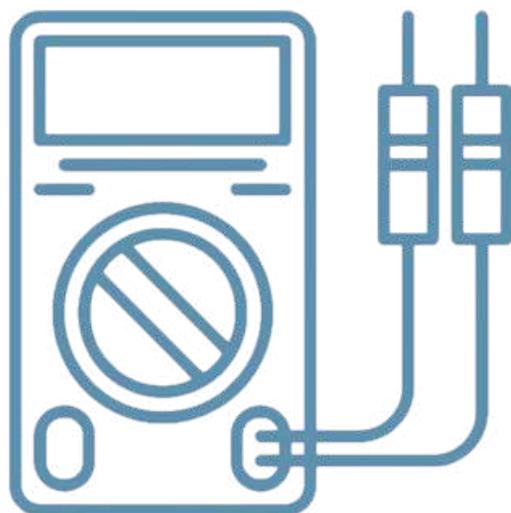
**CURSO DE ARDUINO**

**CLIQUE AQUI**

## A saber

Um **multímetro** é uma ferramenta que você consegue verificar diversas variáveis como resistência, corrente e tensão em um circuito. Não é requisito essencial possuir essa ferramenta em mãos para desenvolver os projetos prontos com Arduino desta apostila PDF.

**MULTÍMETRO**



# PROJETO 2



SWITCH



LED



RESISTOR DE 220 OHM



RESISTOR DE 10 KILOHM

COMPONENTES UTILIZADOS

# PROJETO: ACENDENDO 3 LED's

Descubra neste projeto: pinos digitais inputs e outputs, seu primeiro programa e variáveis.

TEMPO: 45 minutos

NÍVEL: 

Agora que você já tem uma certa base de eletricidade, é hora de usar seu Arduino para começar a controlar as coisas.

Neste projeto, você irá montar algo que poderá ter sido parte de uma interface de espaçonave de filme de ficção científica da década de 70.

Você irá montar um **painel de controle** com um botão e luzes que acendem quando você pressiona o interruptor. Um LED verde irá acender se você apertar o botão.

Quando o Arduino receber um sinal do botão, a luz verde irá desligar e outras 2 luzes começarão a piscar.

Os pinos digitais do Arduino podem ler apenas dois estados: quando há tensão em um pino de entrada e quando não há. Esse tipo de entrada é normalmente chamado de digital (ou, às vezes, binário, para dois estados).

Esses estados são comumente referidos como **HIGH e LOW**.

**HIGH** é o mesmo que dizer "há tensão aqui!" e **LOW** significa "não há tensão neste pino!". Quando você usa um pino **OUTPUT HIGH** com um comando chamado **digitalWrite ()**, está ativando-o.

Além disso, se você medir a tensão entre o pino e o GND você obterá 5 volts. Quando você atribui a um pino **OUTPUT LOW**, significa que você está desligando-o.

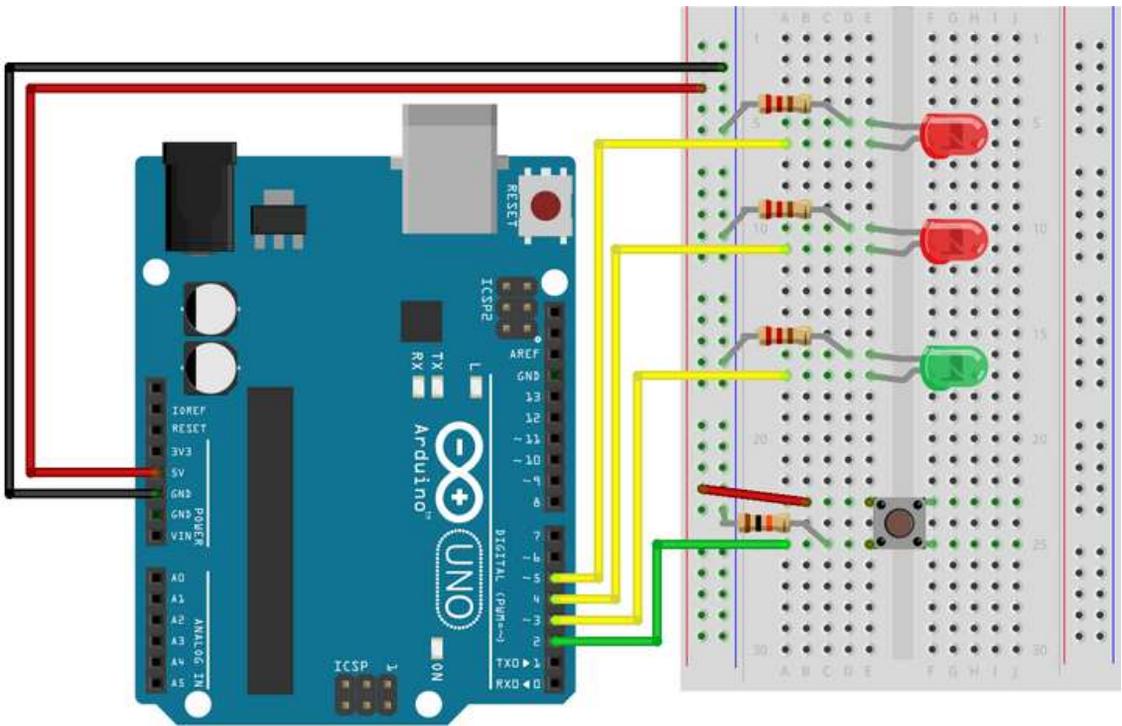
Os pinos digitais do Arduino podem atuar como entradas (**INPUTS**) e saídas (**OUTPUTS**). No seu código, você os configurará dependendo do que você deseja que a função deles seja.

Quando os pinos são **OUTPUTS**, você pode ligar componentes como LEDs. Se você configurar os pinos como **INPUTS**, poderá verificar se um interruptor está sendo pressionado ou não.

Para essa configuração de **INPUT** você irá usar o pino 2. Nós não usamos os pinos 0 e 1 pois eles são usados pelo Arduino para se comunicar com o computador.

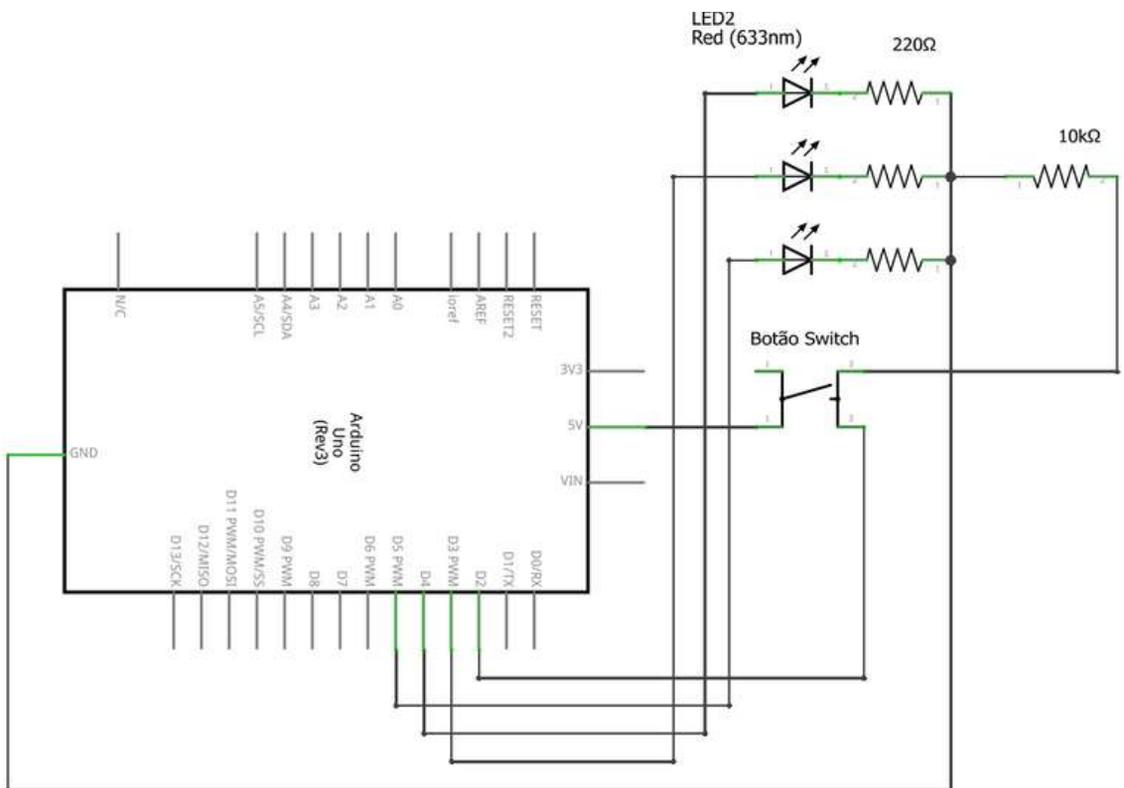
## Montando o circuito

## PROJETO 2



fritzing

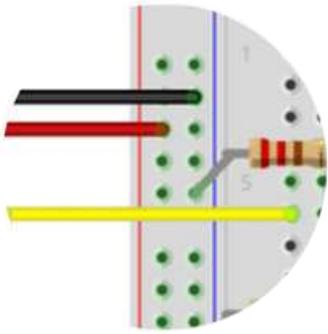
### Montagem do projeto



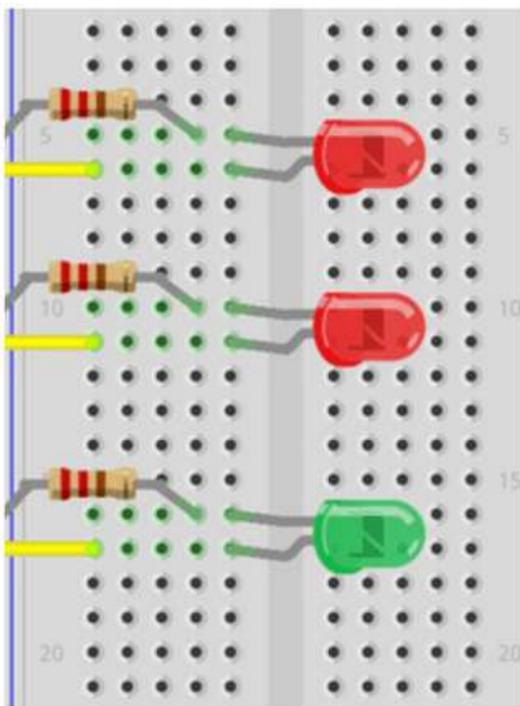
fritzing

### Montagem esquemática do projeto

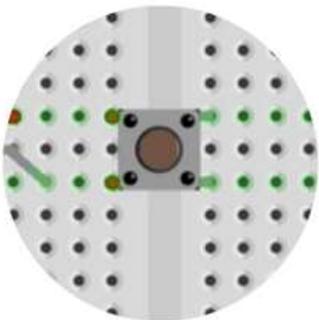
## PROJETO 2



**1)** Conecte sua protoboard aos 5V do Arduino e ao pino GND, assim como no projeto anterior.



**2)** Insira dois LEDs vermelhos e um LED verde na protoboard. Conecte o catodo (perna curta) de cada LED ao GND através de um resistor de 220 ohms. Conecte o anodo (perna longa) do LED verde ao pino 3. Conecte os anodos dos LEDs vermelhos aos pinos 4 e 5, respectivamente.



**3)** Insira o interruptor na protoboard, como fez no exemplo do projeto anterior. Conecte um lado ao power e o outro lado ao pino digital 2 do Arduino. Você também precisará adicionar um resistor de 10k ohm do GND ao pino do switch que se conecta no Arduino.

## O CÓDIGO

### Antes de começar...

Todo programa do Arduino tem duas funções principais. As funções são partes de um programa de computador que executa comandos específicos. As funções têm nomes exclusivos e são "chamadas" quando necessário. As principais e essenciais em qualquer código em um programa Arduino são chamadas de ***setup ()*** e ***loop ()***.

Essas funções precisam ser declaradas, o que significa que você precisa dizer ao Arduino o que essas funções *setup ()* e *loop ()* farão.



```
sketch_may27a | Arduino 1.8.12
Arquivo Editar Sketch Ferramentas Ajuda
sketch_may27a
void setup() {
  // put your setup code here, to run once:
}

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

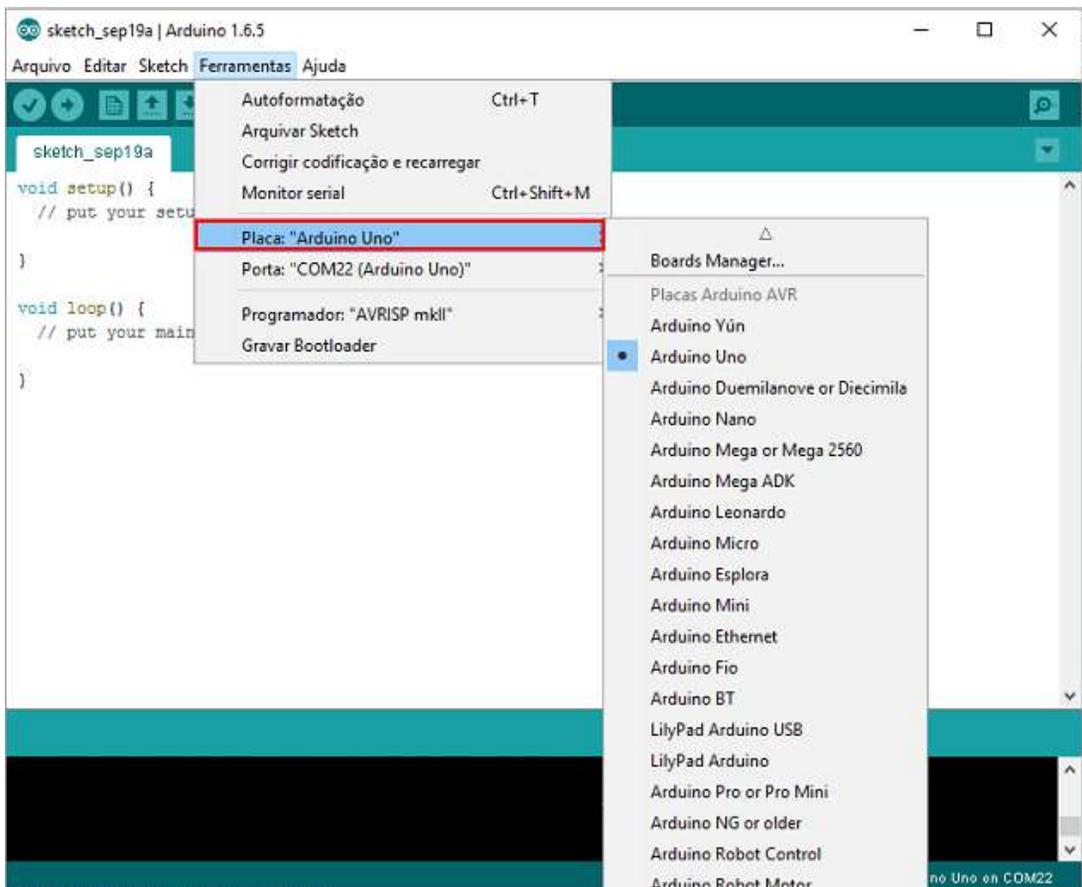
1 Arduino Uno em COM3
```

## PROJETO 2

Nele, você criará uma variável antes de ir para a parte principal do programa. Variáveis são nomes que você fornece para serem armazenadas na memória do Arduino, para que você possa acompanhar o que está acontecendo. Esses valores podem mudar dependendo das instruções que der ao seu programa.

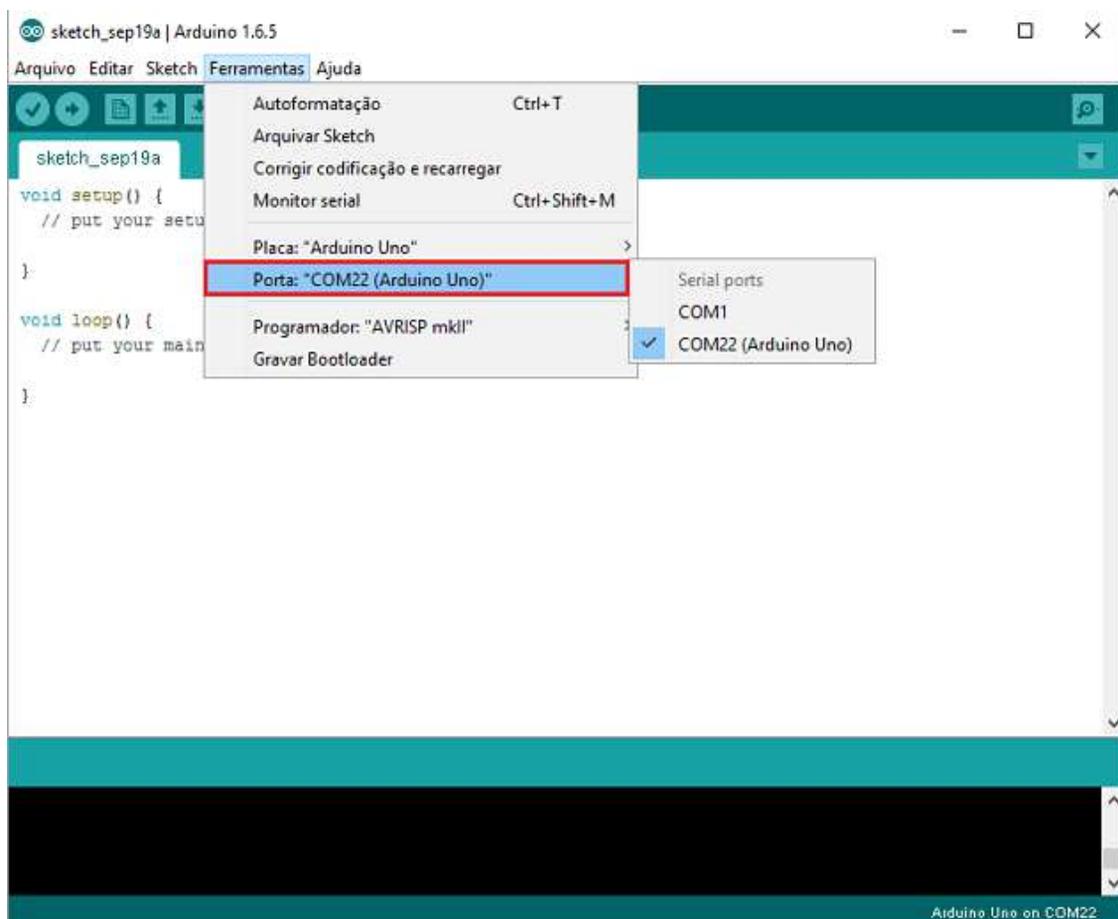
Os nomes de variáveis devem ser descritivos de qualquer valor que estejam armazenando. Por exemplo, uma variável chamada **estadobotao** informa que armazena o estado de um switch.

Feito isso, selecione a placa correta e a porta USB correta com as duas próximas instruções:



**Selecione a placa correta**

## PROJETO 2



Selecione porta USB correta

## ENFIM, BORA PROGRAMAR?

```
int estadobotao = 0;    //declare a variável inteira de leitura do botão

void setup() {

    pinMode(3,OUTPUT);    //diga ao arduino que os pinos das LEDs são de saída:
    'OUTPUT'
    pinMode(4, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(2, INPUT);    //agora para o botão, será de entrada: 'INPUT'
                          //será 'INPUT' para que o arduino faça a leitura do estado do
    botão
}

void loop() {
    estadobotao = digitalRead(2); //a variável 'estadobotao' recebe o estado da porta
    digital 2
```

```
if (estadobotao == LOW){ // se o botão não estiver pressionado

digitalWrite(3, HIGH); // acende o LED da porta 3
digitalWrite(4, LOW); // apaga o LED da porta 4
digitalWrite(5, LOW); // apaga o LED da porta 5
}

else{ // se estiver pressionado

digitalWrite(3, LOW); // apaga o LED da porta 3
digitalWrite(4, LOW); // apaga o LED da porta 4
digitalWrite(5, HIGH); // acende o LED da porta 5

delay(250); // espera 0.250 segundos
digitalWrite(4, HIGH); // acende o led da porta 4
digitalWrite(5, LOW); //apaga o led da porta 5
delay(250); //espera mais 0.250 segundos
}
}
```

## O código explicado passo a passo

```
int estadobotao = 0;
```

↳ Para criar uma variável, você precisa declarar de que tipo ela será. A variável que declarar como ***int*** irá guardar um número (também chamado de inteiro); isso significa ser qualquer valor sem um ponto decimal. A declaração da variável como em toda instrução deve terminar com um **ponto** e uma **vírgula (;)**.

## PROJETO 2

```
void setup() {  
  
  pinMode(3,OUTPUT);  
  pinMode(4, OUTPUT);  
  pinMode(5, OUTPUT);  
  pinMode(2, INPUT);  
}
```

→ CONFIGURE A FUNCIONALIDADE DOS PINOS! O **setup()** roda uma vez, quando o Arduino é iniciado/ligado. É aqui que você configura os pinos digitais para ou serem *inputs* ou *outputs* usando a função **pinMode()**. Os pinos onde LED's são conectados são chamados **OUTPUT** e o botão/switch será declarado como **INPUT**.

```
void loop() {  
  estadobotao = digitalRead(2);
```

→ CRIE UMA FUNÇÃO DE LOOP! O **loop()** roda repetitivamente após o **setup()** tiver sido executado. É na função **loop()** onde você irá verificar a tensão nos inputs, e ligar ou desligar os outputs. Para verificar a tensão em um input digital, usa-se a função **digitalRead()** que verifica a voltagem do pino escolhido. Para saber qual pino verificar, **digitalRead()** aguarda por um argumento.

Os argumentos são informações que você passa para funções anteriormente declaradas, dizendo a elas como elas devem

executar o código. Por exemplo, ***digitalRead*** () precisa de um argumento: qual pino checar.

No seu programa, ***digitalRead*** () irá verificar o estado do pino 2 e armazenar o valor na variável *estadobotao*. Caso não haja tensão no pino quando a função ***digitalRead*** () é chamada, será atribuída à variável *estadobotao* ***HIGH*** (ou 1). Se não há tensão no pino, será atribuído ***LOW*** ao *estadobotao* (ou 0).

```
if (estadobotao == LOW){
```

 A CONDIÇÃO IF: no código, você utiliza a palavra ***if*** (se, em inglês) para verificar o estado de algo. Essa declaração ***if*** em programação faz uma comparação entre dois fatores, e determina se a comparação é ***verdadeira*** ou se é ***falsa***. Feito isso, será tomada uma ação de acordo com o que você pedir para fazer após a condição.

Além disso, é usado o sinal ***==*** em programação para quesito de comparação. Se você usar apenas um sinal, será igualado um valor ao invés de compará-lo.

## PROJETO 2

```
digitalWrite(3, HIGH);  
  digitalWrite(4, LOW);  
  digitalWrite(5, LOW);  
}
```

↳ CONSTRUA SUA NAVE ESPACIAL: ***digitalWrite()*** é o comando que permite que você mande 5V ou 0V para um pino output. Essa função leva em consideração dois argumentos: qual pino controlar, e qual valor determinar para aquele pino, ***HIGH*** ou ***LOW***. Se você quer ligar o LED vermelho e desligar o LED verde dentro da sua condição ***if()***, seu código se parecerá com o exemplo desse projeto 2.

```
else{  
  digitalWrite(3, LOW);  
  digitalWrite(4, LOW);  
  digitalWrite(5, HIGH);  
  
  delay(250);  
  digitalWrite(4, HIGH);  
  digitalWrite(5, LOW);  
  delay(250);  
}  
  
}
```

↳ Até o momento, o arduino já sabe o que fazer quando o botão não está pressionado. Portanto, defina agora o que deve acontecer quando o botão for pressionado. A condição ***if***

tem uma opção chamada **else** (senão, em inglês) que permite que algo aconteça se a condição original não for obedecida. Neste caso, escreva um código para uma condição **HIGH** (1) após o *else*.

Para que os LED's vermelhos pisquem quando o botão for pressionado, você terá que ligar e desligar as luzes no *else* que você acabou de escrever. Para isso, mude o código para ficar igual ao exemplo dado neste projeto.

*Agora seu programa irá acender apenas o LED verde quando o botão estiver pressionado!*

Após ajustar os LED's para um determinado estado **HIGH/LOW**, iremos fazer com que o Arduino pause por um momento. Se você não quiser esperar, as luzes irão acender e apagar tão rapidamente que você nem perceberá. Isso porque a função **loop ()** é processada muitas vezes por segundo, e os LED's irão apagar e ligar sem que nós percebêssemos.

Para resolver esse problema usamos a função **delay ()**. Ela permite que você dê uma pausa na execução de qualquer coisa por um período de tempo. **delay ()** tem seu argumento de tempo na escala de milisegundos, ou seja, 1000 milisegundos equivale a 1 segundo. Um *delay (250)* representa  $\frac{1}{4}$  de segundo.

### Observe!

Uma vez que seu Arduino estiver programado, você irá ver a luz verde que está nele ligar. Quando você pressionar o botão, as luzes vermelhas começarão a piscar, e a luz verde irá desligar.

Tente mudar o tempo das funções **delay()** e note o que acontece com as luzes e como a resposta do sistema muda dependendo da velocidade do piscar das luzes. Use sua criatividade, não fique preso(a) na programação Arduino PDF.

Quando você insere uma função **delay()** no seu programa, ele interrompe todas as outras funcionalidades e nenhuma leitura de sensor irá ocorrer antes que o período de tempo estipulado por você passe.

### Desafio

- Como você faria para que os LEDs vermelhos piscassem quando o programa iniciar?
- Como você pode criar uma interface maior ou mais complexa para suas aventuras interestelares com LEDs e switches?

### Recaptulando...

- Neste projeto, você criou seu primeiro programa Arduino com o código do PDF para controlar o comportamento de alguns LEDs com base em um botão/switch.
- Você usou variáveis, uma instrução *if () ... else* e funções para ler o estado de uma entrada e controlar saídas.

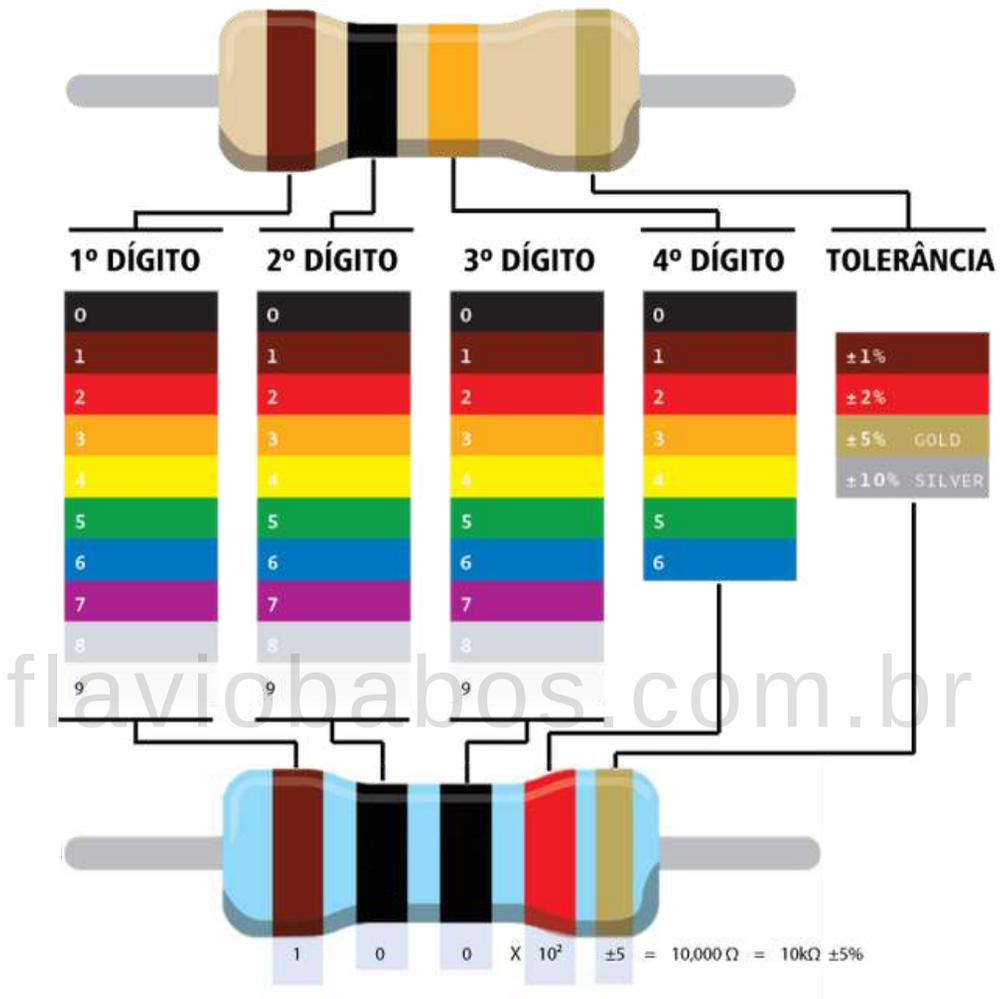
### Fazendo a leitura das cores dos resistores

Os valores dos resistores são marcados usando faixas coloridas, de acordo com um código desenvolvido na década de 1920, quando era muito difícil escrever números em objetos tão pequenos.

Cada cor corresponde a um número, como você vê na tabela da próxima página. Cada resistor possui 4 ou 5 bandas. No tipo de 4 bandas, as duas primeiras indicam os dois primeiros dígitos do valor enquanto o terceiro indica o número de zeros que se seguem (tecnicamente representa a potência de dez).

A última banda especifica a tolerância. No exemplo a seguir, o amarelo indica que o valor do resistor pode ser 10k ohm mais ou menos 5%.

# PROJETO 2



   **4 bandas**

   **5 bandas**

220 Ω      560 Ω      4.7kΩ

    **4 bandas**

    **5 bandas**

1kΩ      10kΩ      1MΩ      10MΩ

# ATENÇÃO!!

**Eu tenho certeza que isso vai te ajudar muito!**



Quero compartilhar um documento com você que acho ser de extrema relevância caso você queira aprimorar suas habilidades em programação com Arduino.

O fato é que esse arquivo é um mapa mental que é um apanhado geral das funções que mais utilizo na linguagem C++.

Ele é um atalho bem útil para você que quer ter praticidade enquanto programa seus projetos.

Consegui liberar esse mapa mental de graça. Então, sugiro que baixe o mais rápido possível antes que saia do ar.

[Clique aqui](#) ou na imagem abaixo para ter acesso a esse arquivo com as funções de programação que todo iniciante em Arduino deveria ter:

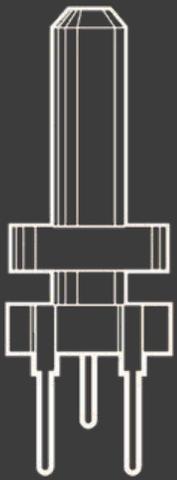


Lembrando que esse conteúdo pode te ajudar de uma vez por todas a entender como criar seus próprios projetos e a programar Arduino do zero, mesmo sem ter todo o conhecimento do mundo em programação.

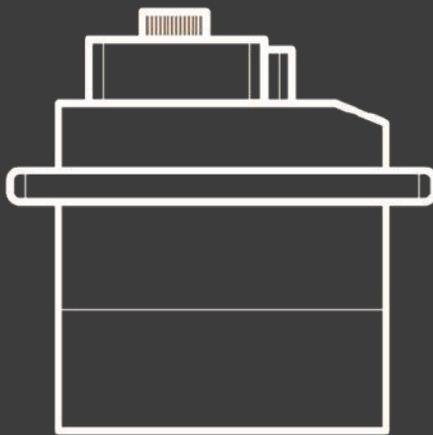
Espero que goste, fiz com muito carinho para te ajudar na sua jornada.

**PS:** Para acessar o nosso método completo que te faz ser aprender do básico ao avançado tudo sobre Arduino e em tempo recorde, [clique aqui](#).

# PROJETO 3



LED



SERVO MOTOR



ATUADOR SERVO



CAPACITOR DE 100µF

COMPONENTES UTILIZADOS

# PROJETO: ACIONANDO O SERVO MOTOR

Você irá aprender a como programar e controlar um servo motor.

Descubra neste projeto: mapeamento de valores, servomotores, utilização de bibliotecas internas.

TEMPO: 1 hora

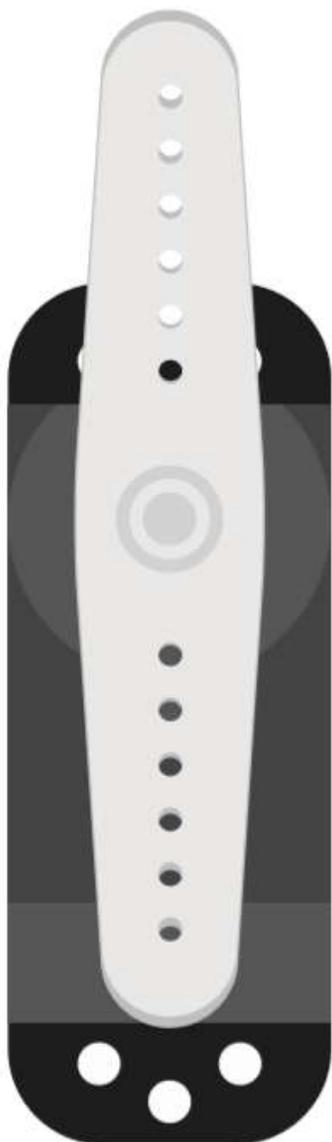
NÍVEL: 

Montado nos projetos 1, e 2.

Os **servomotores** são um tipo especial de motor que não gira em círculo, mas se move para uma posição específica. Eles geralmente rodam 180 graus (metade de um círculo).

Semelhante à maneira como utilizamos **pulsos para PWM** em um LED, os servomotores aguardam um **número de pulsos** que lhes dizem em que ângulo se mover.

Os pulsos sempre chegam nos mesmos intervalos de tempo, mas a largura varia entre 1000 e 2000 microssegundos. Embora seja possível escrever um código para gerar esses pulsos, o software Arduino vem com uma biblioteca que permite controlar facilmente o motor.



Como o servo gira apenas 180 graus e sua entrada analógica varia de 0 a 1023, você precisará usar uma função chamada [map\(\)](#) para alterar a escala dos valores provenientes do potenciômetro.

Uma das grandes contribuições da comunidade Arduino são pessoas talentosas que ampliam sua funcionalidade por meio de software adicional.

Por causa disso, existem [bibliotecas](#) para uma ampla variedade de sensores e atuadores e outros dispositivos que os usuários contribuíram para a comunidade.

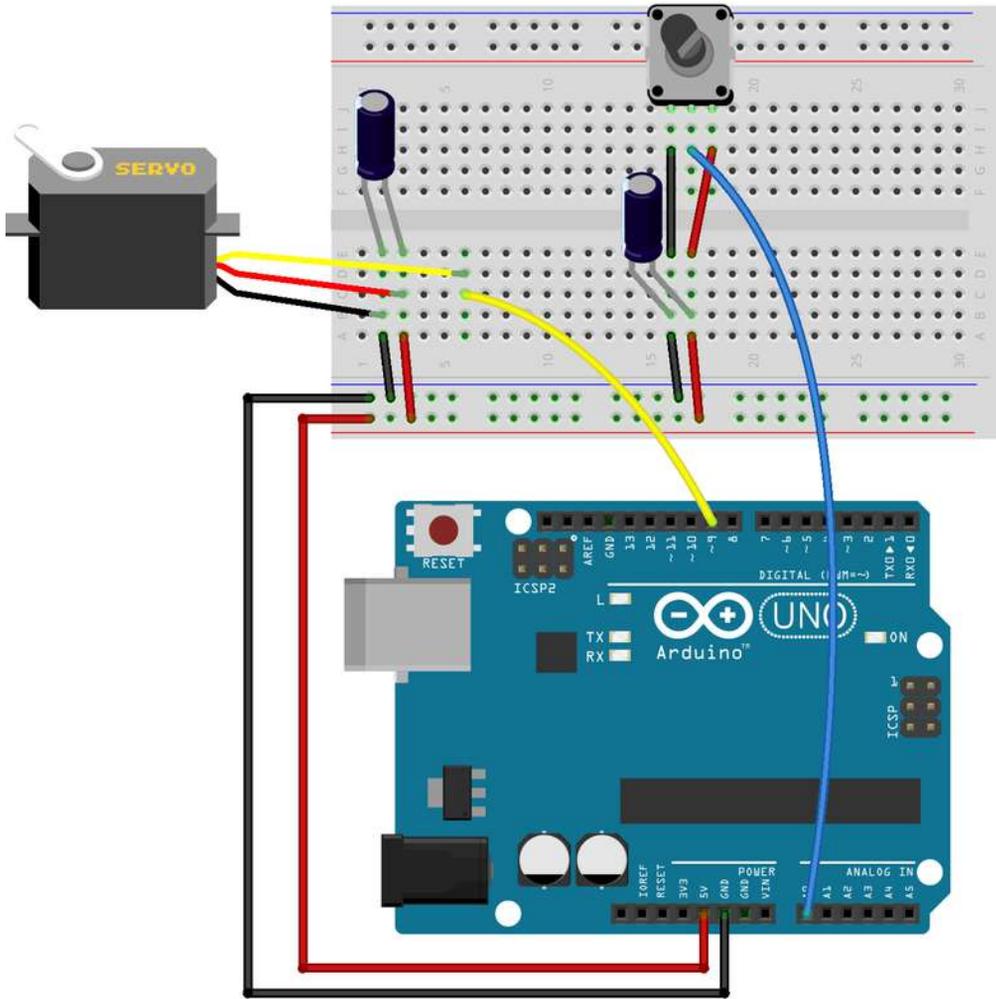
Uma biblioteca de software expande a funcionalidade de um ambiente de programação. Aliás, a IDE já vem com várias bibliotecas úteis para trabalhar com hardware ou dados.

Sendo assim, para este projeto usamos uma das bibliotecas que foi projetada para auxiliar o controle de servomotores.

Você irá importa-la para dentro da sua IDE e todas as suas funcionalidades já estarão disponíveis para você utilizar de maneira compactada. Veja o circuito na próxima página e em seguinte a Programação Arduino PDF.

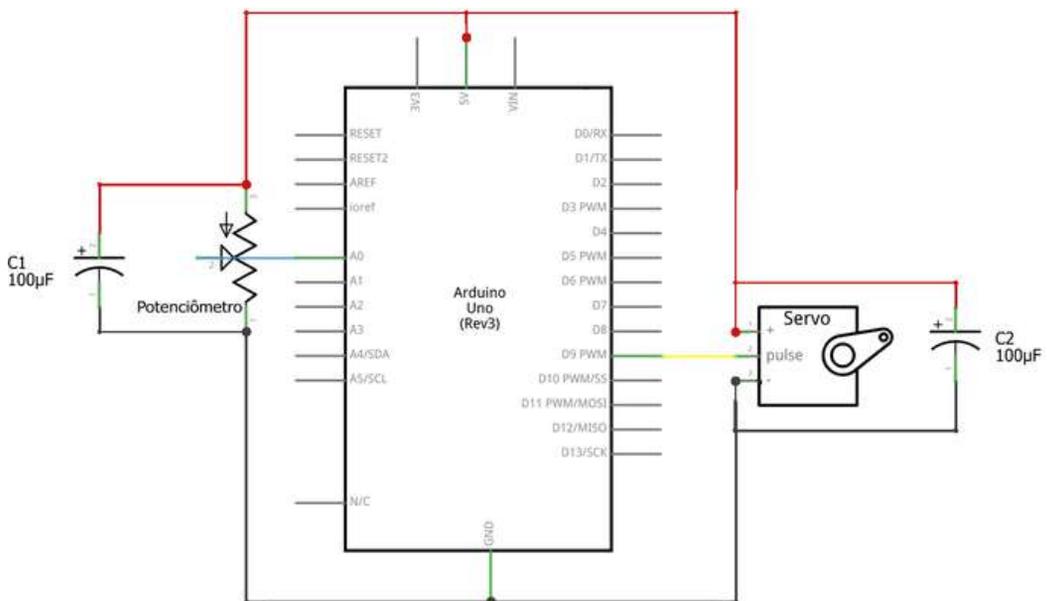
## Montagem do circuito

## PROJETO 3



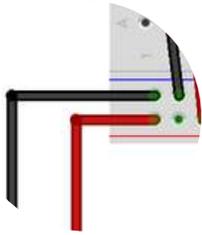
fritzing

### Montagem dos componentes do projeto

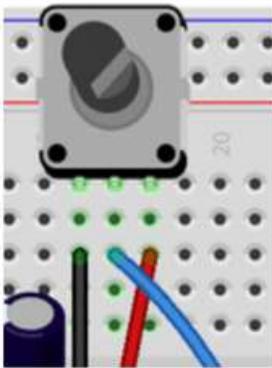


fritzing

### Montagem esquemática do projeto



1) Conecte Vin - 5V e GND na sua protoboard a partir do Arduino.



2) Coloque um potenciômetro na protoboard e conecte um lado nos 5V e o outro ao GND. Um potenciômetro é um tipo de **divisor de tensão**. Ao girar o botão, você altera a proporção da tensão entre o pino do meio e a potência. Você pode ler essa alteração em uma entrada analógica. Conecte o pino do meio ao pino analógico 0. Isso controlará a posição do seu servo motor.



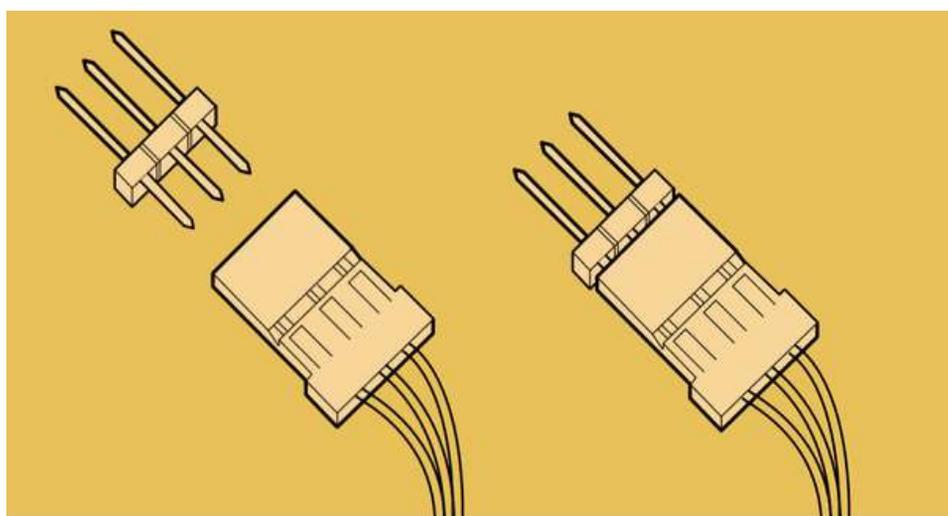
3) O servo tem três fios saindo dele. Um é o power (vermelho), um é GND (preto) e o terceiro (amarelo) é a linha de controle que receberá informações do Arduino. Conecte os 5V ao fio vermelho, aterre o fio preto e o fio amarelo conecte-o ao pino 9 do arduino.

**CONQUISTE NOVOS HORIZONTES E  
ALAVANQUE SEU CONHECIMENTO!**



**CURSO DE ARDUINO**

> CLIQUE AQUI



➔ Seu servo motor é fornecido com conectores fêmea, portanto, você precisará adicionar pinos de cabeçalho/header pins para conectá-lo à protoboard.



4) Quando um servo motor começa a se mover, ele consome mais corrente do que se já estivesse em movimento. Isso causa um **pico na tensão** da sua placa. Ao colocar um capacitor de 100uf no POWER e no GND, ao lado dos conectores macho, como mostrado na figura de montagem do projeto, você pode **suavizar** quaisquer alterações de tensão que possam ocorrer.

Você também pode colocar um capacitor no POWER e no GND, entrando no seu potenciômetro. Esses são chamados de **capacitores de desacoplamento**, porque reduzem ou desacoplam as alterações causadas pelos componentes no restante do circuito.

### NOTE

Depois de programar e ligar o seu Arduino, abra o monitor serial. Você deve ver um fluxo de valores semelhante a este:

***Valor do potenciometro: 1023, ângulo: 179***

Ao girar o potenciômetro, você verá os números mudarem. Mais importante, você deve ver seu servo motor mudar para uma nova posição.

## PROJETO 3

Observe a relação entre o valor **ValorPot** e o **ângulo** no monitor serial e a posição do servo. Você deve ver resultados consistentes ao girar o **pot**.

O uso de potenciômetros como entradas analógicas que você aprendeu na Programação do Arduino PDF, tem uma vantagem em fornecerem uma faixa completa de valores entre 0 e 1023. Isso os torna úteis para testar projetos que usam entrada analógica.

### ADOpte

O potenciômetro não é o único sensor que você pode usar para controlar o servo.

- Usando a mesma configuração física e um sensor diferente, que tipo de indicador você pode criar?
- Como isso funcionaria através de um sensor de temperatura?
- Você poderia dizer a hora do dia com um fotorresistor?
- Como os valores de mapeamento entram em jogo com esses tipos de sensores?

## SUMÁRIO

Este PDF Básico sobre Arduino é apenas uma amostra do que você encontrará na apostila completa e gratuita que conta com 155 páginas e 10 projetos, todos voltados para os iniciantes aprenderem na prática e sem enrolação a plataforma Arduino.

Para ter acesso total a apostila gratuita basta garantir seu exemplar completo [clcando aqui](#) ou no link abaixo:

[GARANTA AQUI SEU EXEMPLAR COMPLETO E GRATUITO](#)

