



O que é um arduino?

É uma plataforma de código e hardware livre para prototipação.

O que isto significa?

**OPEN SOURCE-**

"RECURSOS DE HARDWARE E SOFTWARE QUE PODEM SER USADOS, ADAPTADOS E DISTRIBUÍDOS SEM CUSTO"

**ELECTRONICS-**

"TECNOLOGIA QUE FAZ USO DO FLUXO DE ELÉTRONS EM DIVERSOS TIPOS DE MATERIAIS PARA OS MAIS DIVERSOS FINS"

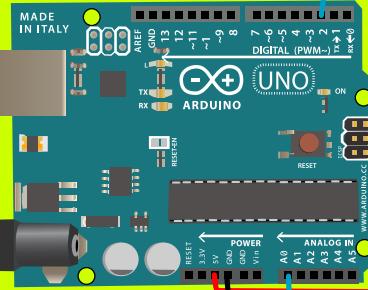
**PROTOTYPE-**

"UMA MONTAGEM NÃO DEFINITIVA DE UM PROJETO PARA COMPROVAR O FUNCIONAMENTO"

**PLATFORM-**

"UM CONJUNTO DE HARDWARE E SOFTWARE QUE POSSIBILITA REALIZAR PROJETOS DE ELETRÔNICA"

MICROCHIP



PHOTOCELL



LED



BREADBOARD

O Arduino é um microprocessador montado com alguns acessórios para poder funcionar como um minúsculo computador que pode ser programado. Ele pode responder aos estímulos (elettricidade) vindos de sensores (fotocélula) e usar estes dados para medir condições como a luz em um ambiente e, assim, controlar um interruptor para acender a luz quando o ambiente fica escuro.

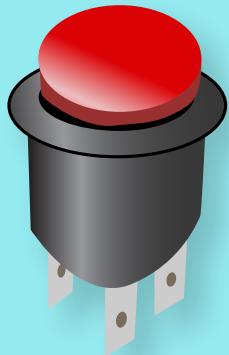


O arduino pode reagir ao se ligar um interruptor!

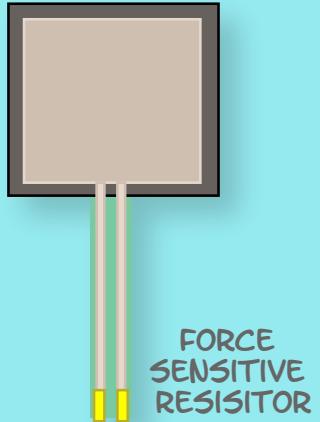
O mouse é um sensor ligado ao seu computador, com ele você interage com seu microprocessador.



Microprocessadores usam inputs e outputs como qualquer computador. Inputs capturam informação do usuário ou do ambiente enquanto os outputs fazem alguma coisa com a informação capturada.



MOMENTARY SWITCH



FORCE  
SENSITIVE  
RESISTOR

Um botão ou um sensor sofisticado podem ser os inputs do arduino



DC MOTOR



Qualquer objeto que se queira ligar e desligar pode ser um output do arduino, eles podem ser simples motores ou até um computador.



Qual é a diferença entre inputs e outputs analógicos e digitais?

Inputs e outputs podem ser analógicos ou digitais. Informação digital tem o formato binário e pode ser tanto verdadeira como falsa, zero ou 1. Informação analógica é por exemplo, a variação de uma corrente elétrica.

Informação digital pode ter dois estados. Ligado ou desligado, 0 ou 1. Quando a luz está ligada é 1, quando está desligada é zero.

Informação analógica é mais como um fluxo contínuo e pode ter infinitos valores.



O botão é um input digital, só pode ter dois valores. Um sensor é um input analógico que pode variar em muitos valores diferentes.



VOLTAGEM?  
CORRENTE?  
RESISTÊNCIA?  
LEI DE OHM?

Antes de ligar o Arduino  
vamos revisar alguns termos  
e princípios que tem a ver com  
a forma como a eletricidade ( e eletrônica)  
funcionam.

**VOLTAGEM (V)**

É A MEDIDA DO  
POTENCIAL  
ELÉTRICO  
  
SUA UNIDADE É O  
VOLT (V)

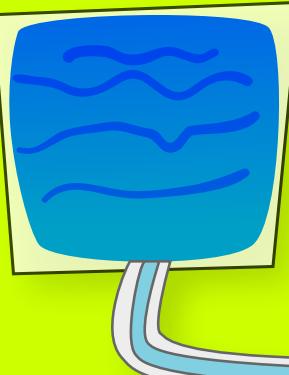
**CORRENTE (I)**

É O QUANTIDADE  
DO FLUXO EM UM  
MATERIAL  
CONDUTIVO  
  
SUA UNIDADE  
É O AMPÉRE OU  
AMP

**RESISTÊNCIA (R)**

É A CAPACIDADE  
QUE UM MATERIAL  
TEM DE RESISTIR  
A CORRENTE ELÉTRICA  
  
SUA UNIDADE É O OHM

ELETRICIDADE É O FLUXO DE ENERGIA POR UM MATERIAL CONDUTIVO.



A VELOCIDADE DO FLUXO É  
DETERMINADA PELA VOLTAGEM

A RESISTÊNCIA AUMENTA  
OU DIMINUI O FLUXO

A QUANTIDADE DO FLUXO CORRENDO  
PELO TUBO É A CORRENTE

A ANALOGIA COM A ÁGUA CORRENDO POR UM TUBO É UMA DAS FORMAS UTILIZADAS PARA ILUSTRAR ESTE PROCESSO

## LEI DE OHM

corrente = voltagem/resistência

$$(I = V/R)$$

ou

Resistência = voltagem/corrente

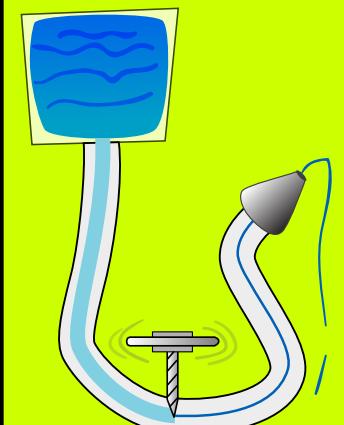
$$(R = V/I)$$

ou

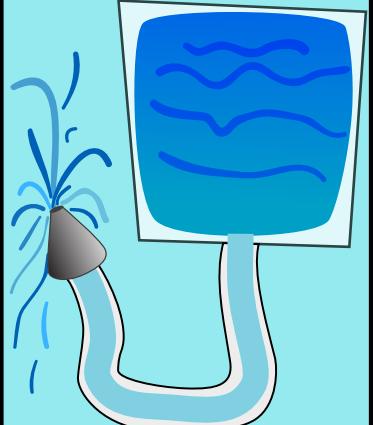
Voltagem = Resistência \* corrente

$$(V = R \cdot I)$$

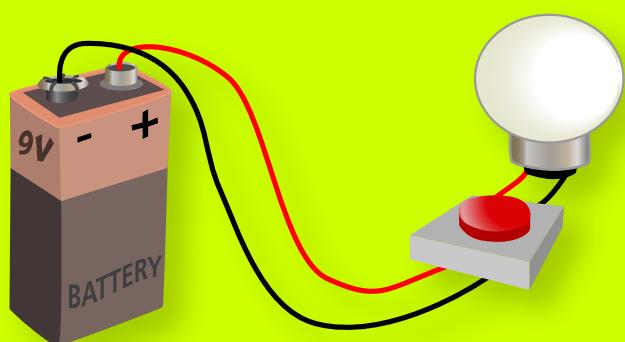
EXISTE UMA RELAÇÃO ENTRE VOLTAGEM, CORRENTE E RESISTÊNCIA. ESTA RELAÇÃO FOI DESCOBERTA PELO FÍSICO ALEMÃO GEORG OHM.



POR EXEMPLO, AO AUMENTAR A RESISTÊNCIA O FLUXO DIMINUI...



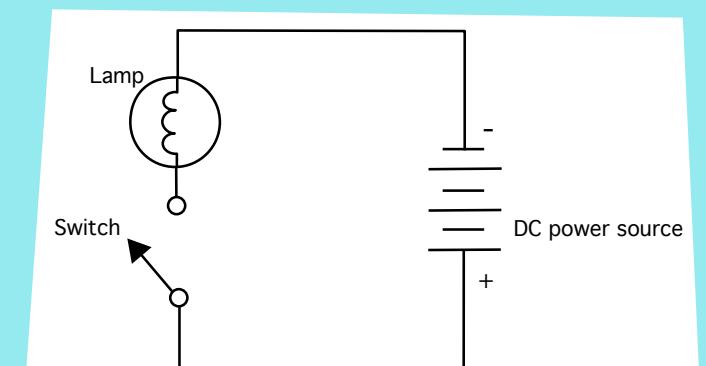
AO AUMENTAR O POTENCIAL O FLUXO AUMENTA



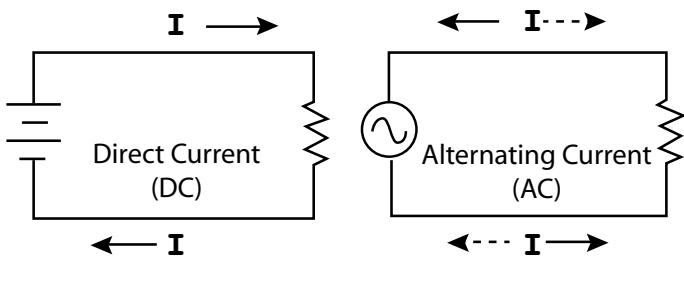
VAMOS OBSERVAR UM CIRCUITO SIMPLES.

Todo o circuito é um círculo fechado que tem uma fonte de energia (bateria) e uma carga (lâmpada).

A carga converte a energia elétrica e a utiliza (gera luz e calor). Este circuito também tem um interruptor.



ESTE É O ESQUEMA ELETRÔNICO DO MESMO CIRCUITO. É uma representação que utiliza símbolos para cada um dos componentes eletrônicos. Quando o interruptor é fechado a corrente flui da fonte (bateria) até a carga (lâmpada).



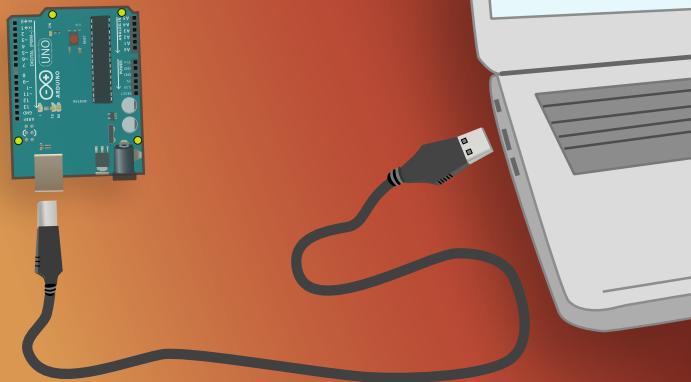
Existem dois tipos comuns de circuitos elétricos.

Corrente Contínua (CC) e Corrente Alternada (CA). A corrente contínua é a eletricidade que vem de uma bateria, é gerada quimicamente. A corrente alternada é a eletricidade que vem da tomada em sua casa, é gerada mecanicamente. Vamos usar a corrente contínua em nossos exemplos.

Agora que já revisamos os conceitos básicos de eletricidade, vamos voltar ao Arduino.....



O ARDUINO PRECISA DE ELETRICIDADE PARA FUNCIONAR, VAMOS CONECTÁ-LO AO COMPUTADOR.



CONECTAR O ARDUINO AO COMPUTADOR PELA PORTA USB VAI NOS DAR OS 5 VOLTS NECESSÁRIOS PARA SEU FUNCIONAMENTO E PARA PROGRAMÁ-LO.

**CARREGUE NO SITE DO ARDUINO:**

[HTTP://ARDUINO.CC/EN/MAIN/SOFTWARE](http://arduino.cc/en/Main/Software)

VOÇÊ PRECISA CARREGAR E INSTALAR O SOFTWARE PARA PODER PROGRAMAR O ARDUINO. O software é livre e existem versões para windows, mac e linux. O software é chamado de IDE (Integrated Development Environment), um ambiente integrado para desenvolvimento.

**PARA INSTRUÇÕES SOBRE A INSTALAÇÃO EM PORTUGUÊS VISITE:**

[http://www.labdegaragem.com.br/wiki/index.php?title=Sobre\\_Arduino](http://www.labdegaragem.com.br/wiki/index.php?title=Sobre_Arduino)

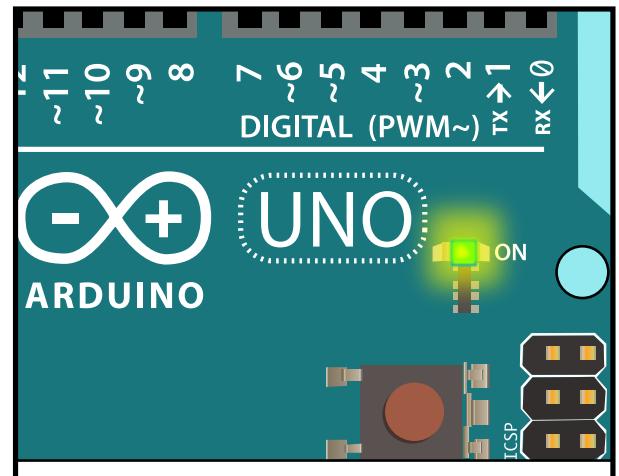
**PARA INSTRUÇÕES DE INSTALAÇÃO EM INGLÊS VISITE:**

Mac: <http://www.arduino.cc/en/Guide/MacOSX>

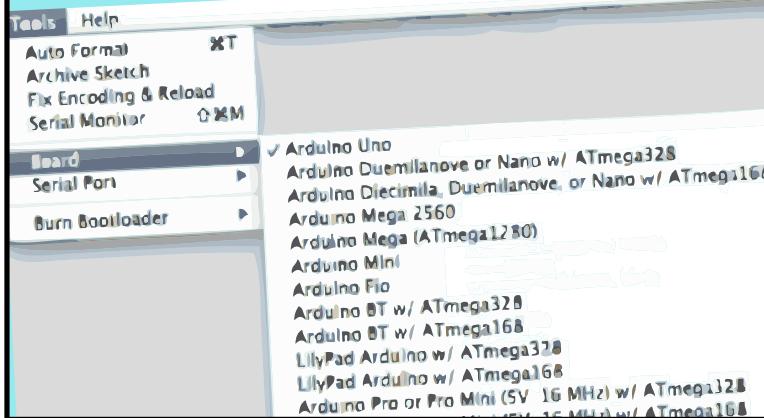
Windows: <http://www.arduino.cc/en/Guide/Windows>

Linux: <http://www.arduino.cc/playground/Learning/Linux>

**SE TIVER DIFICULDADES NA INSTALAÇÃO FREQUENTE AS NOITES DO ARDUINO NO GAROA HACKER CLUBE, AS LISTAS DE DISCUSSÃO OU O LAB DE GARAGEM.**

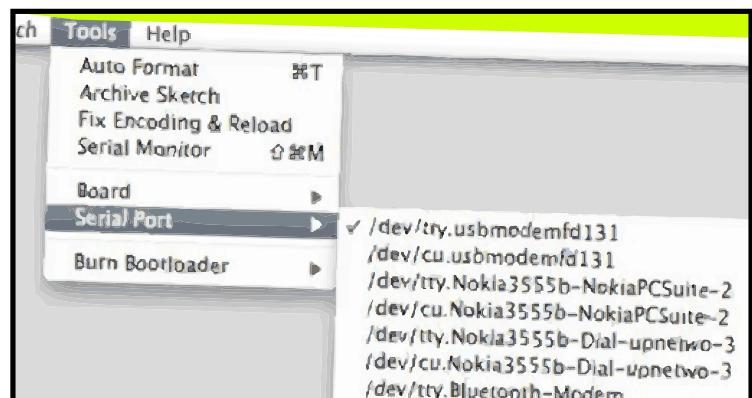


**DEPOIS DE INSTALAR O SOFTWARE E CONECTAR O ARDUINO AO COMPUTADOR O LED DE "POWER" DEVE ACENDER.**



**INICIE O SOFTWARE.**

**NO MENU "TOOLS" SELECIONE A PLACA QUE VOCÊ ESTÁ USANDO (TOOLS>BOARD) COMO POR EXEMPLO, ARDUINO UNO OU DUEMILENOVE.**



**DEPOIS É NECESSÁRIO SELECIONAR A PORTA SERIAL.**  
Use o menu (Tools> Serial port) no windows será algo como COM3 ou outra. No mac e linux algo parecido com "/dev/tty.usbmodem".  
Se não houver nada neste menu... revise sua conexão e instalação de drivers.



Arduino File Edit Sketch Tools Help

New ⌘N  
Open... ⌘O  
Sketchbook ►  
**Examples** ► 1.Basics ►

Close ⌘W  
Save ⌘S  
Save As... ⌘S  
Upload to I/O Board ⌘U  
Page Setup ⌘P  
Print ⌘P

AnalogRead  
BareMinimum  
**Blink** ►  
DigitalRead  
Fade

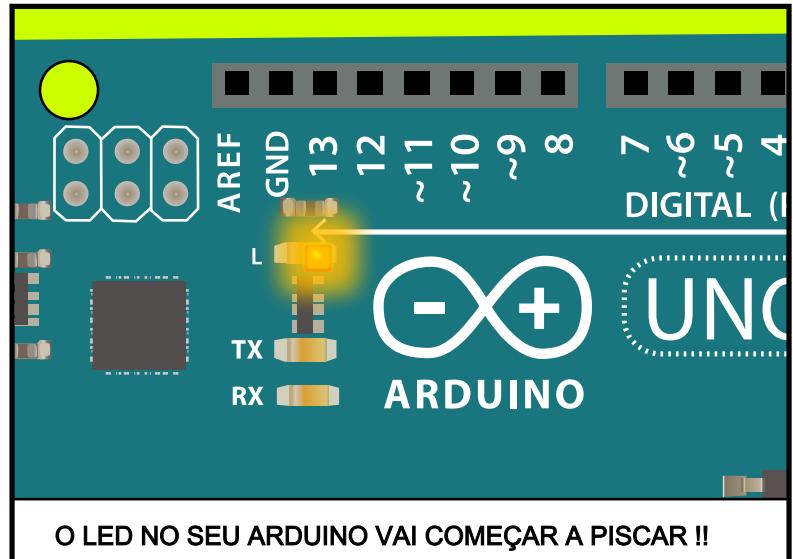
A IDE do Arduino permite que você crie os programas chamados de "SKETCH" e que eles sejam carregados no Arduino. Abra o exemplo Blink no menu: File > Examples > 1.Basics > Blink.

PLAY STOP FILE UPLOAD DOWNLOAD UPLOADER

```
int ledPin = 13;
void setup() {
  pinMode(ledPin, OUTPUT);
}
void loop() {
```

**UPLOAD BUTTON**

Depois de qabrir o exemplo blink, clique no botão UPLOAD para carregar este programa ou "SKETCH" no Arduino. Algumas mensagens vão aparecer na parte inferior da tela e caso tudo tenha corrido bem voce vai ver a mensagem "Done Uploading".



```

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has LED connected on most Arduino boards
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);    // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);     // set the LED off
  delay(1000);              // wait for a second
}

```

Um sketch, como um programa escrito em qualquer linguagem de programação, é um conjunto de instruções.

Se olharmos em detalhe no sketch blink veremos que ele tem duas partes principais: SETUP e LOOP

[HTTP://ARDUINO.CC/EN/REFERENCE/HOME PAGE](http://arduino.cc/en/Reference/Homepage)



Visite o site do Arduino na seção "reference" para aprender mais sobre a linguagem de programação. Use os exemplos que vem na IDE.

**SETUP:** ACONTECE UMA VEZ QUANDO O PROGRAMA COMEÇA A SER EXECUTADO

**LOOP:** REPETE INFINITAMENTE (ATÉ DESLIGAR OU REINICIAR O ARDUINO)

Estes dois blocos de código são chamados de funções e aparecem em todos os sketches. Note que as instruções são separadas por {}

```

void setup() {           //DECLARES BLOCK OF CODE
  pinMode(13, OUTPUT); //SETS PIN 13 TO OUTPUT
}                         //END BLOCK OF CODE

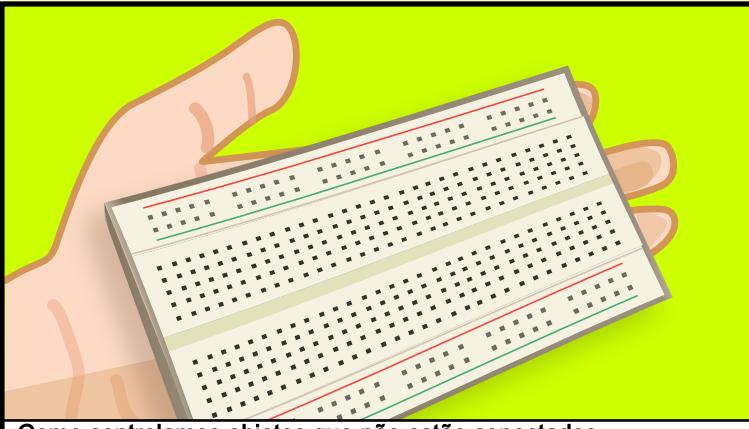
```

```

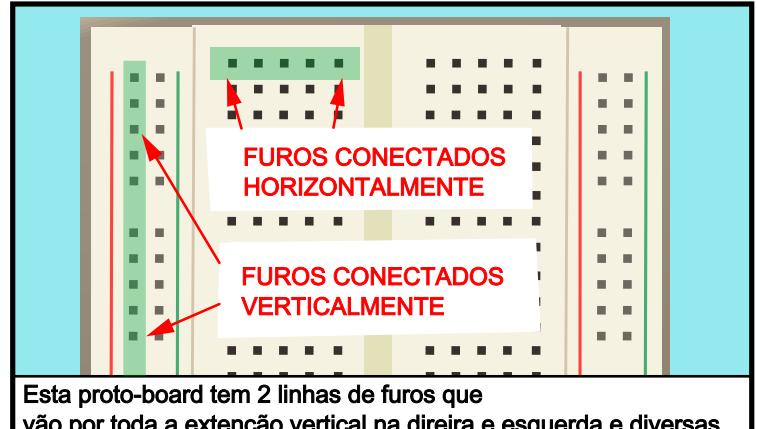
void loop() {           //DECLARES BLOCK OF CODE
  digitalWrite(13, HIGH); //SETS PIN 13 HIGH
  delay(1000);           //PAUSE 1 SECOND
  digitalWrite(13, LOW);  //SETS PIN 13 LOW
  delay(1000);           //PAUSE 1 SECOND
}                         //END BLOCK OF CODE

```

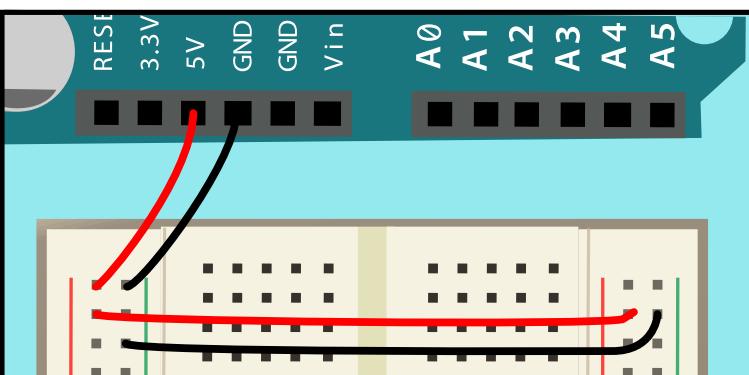
Por enquanto vamos ver este exemplo e entender o que cada linha faz.



Como controlamos objetos que não estão conectados no Arduino? Vamos ligar o Arduino em uma placa de protótipos chamada Proto-board ou Breadboard. Isto vai nos ajudar a ligar tudo rapidamente e testar nossos circuitos.



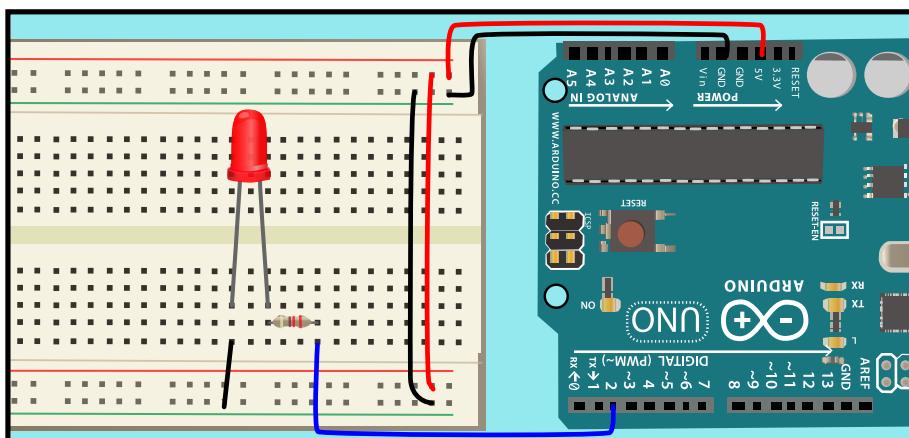
Esta proto-board tem 2 linhas de furos que vão por toda a extenção vertical na direira e esquerda e diversas linhas de 5 furos em toda a superfície na horizontal. As duas linhas verticais são conectadas eletricamente na vertical e as linhas horizontais são conectadas eletricamente na horizontal.



Nós vamos conectar a força (power) e o terra (ground) de nossa placa arduino aos furos verticais da proto-board. Desta forma todos os componentes que ligarmos nos furos horizontais poderão ser alimentados pela placa Arduino.



Quando a corrente flui por um LED ( Diodo Emissor de Luz) na direção correta ele se acende. Nós vamos conectar um led na proto-board e no Arduino para que possamos controlar o LED com nossa programação.



O anodo vai conectado ao pino 2 do Arduino com um resistor de 220 Ohm.  
O catodo vai conectado ao terra. Os pinos de 2 a 13 podem ser configurados como inputs ou outputs digitais.

Clique no botão "new" para começar seu próprio sketch.

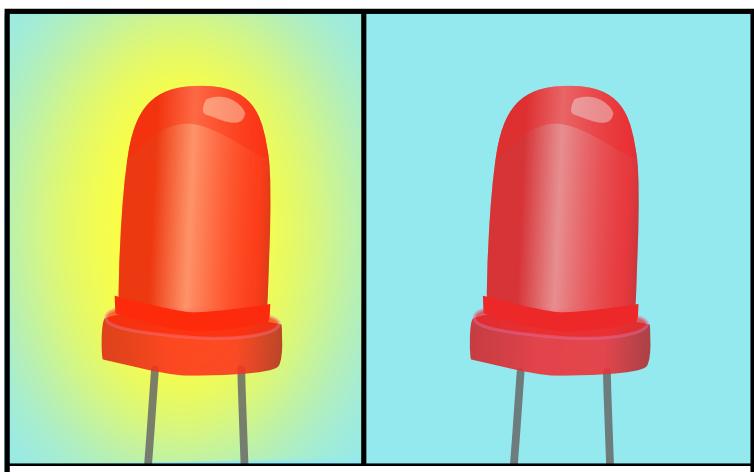
```
void setup() {
    pinMode(2, OUTPUT);
}

void loop() {
    digitalWrite(2, HIGH);
    delay(500);
    digitalWrite(2, LOW);
    delay(500);
}
```

Na função SETUP vamos definir o pino 2 como sendo um output. Na função LOOP vamos definir o pino 2 como "high" ou ligado para acender o LED. Logo abaixo vamos definir que o estado ligado dure 500 milissegundos (meio segundo) e então mude para "low" ou desligado. Pausamos mais meio segundo e repetimos.



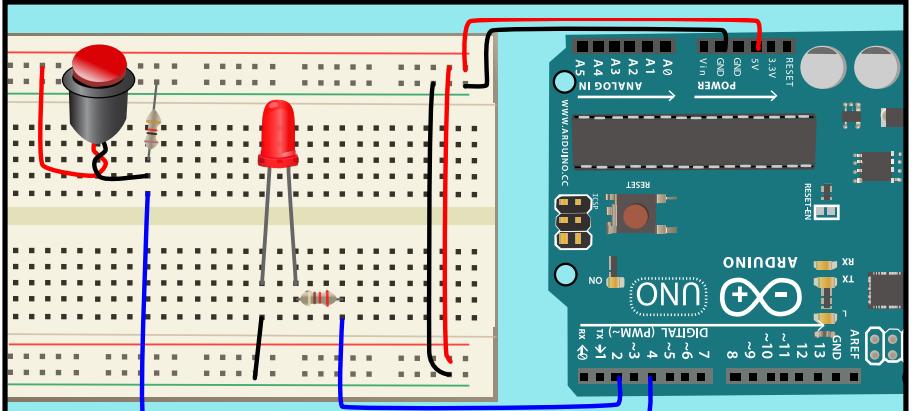
Clique no botão "verify" para verificar se seu código não tem erros.  
Se tudo estiver certo, clique no botão "upload" para carregar seu  
programa para a placa Arduino.



O LED vai piscar... aceso durante meio segundo e apagado durante meio segundo e apagado durante meio segundo... até você desligar ou mudar a programação.



Agora nós vamos usar um botão, um input digital, para controlar o led.

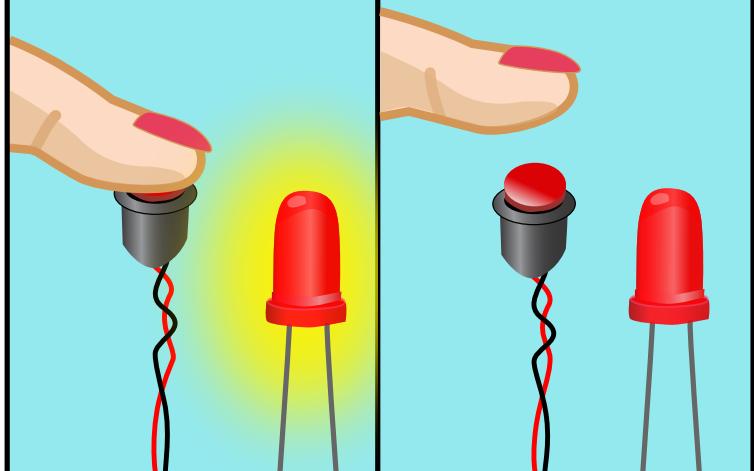


Conecte um terminal do botão ao pino 4 do arduino com um resistor de 10 k conectado ao terra e ligado ao mesmo terminal. Conecte agora o outro terminal na força. O led fica no mesmo pino do arduino.

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

void loop() {
    if(digitalRead(4)){
        digitalWrite(2, HIGH);
    }else{
        digitalWrite(2, LOW);
    }
}
```

Vamos escrever o código. No setup declaramos o pino 2 como output e o pino 4 como input. No loop vamos usar um IF (SE), se a leitura do pino 4 for High (ligado), nos ligamos o pino do led, em qualquer outro caso nós mantemos o pino do led LOW (desligado).

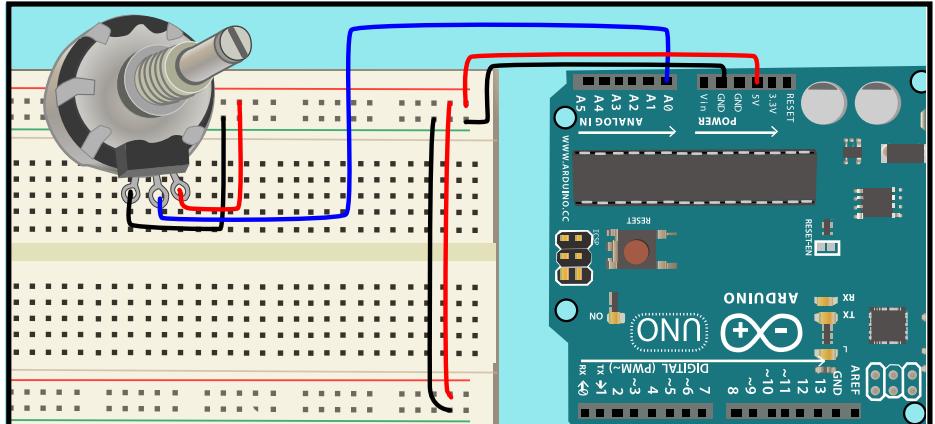


O LED ACENDE QUANDO APERTAMOS O BOTÃO

Um potenciômetro é um resistor variável. A resistência muda quando giramos o botão, aumentando ou diminuindo de acordo com a direção.



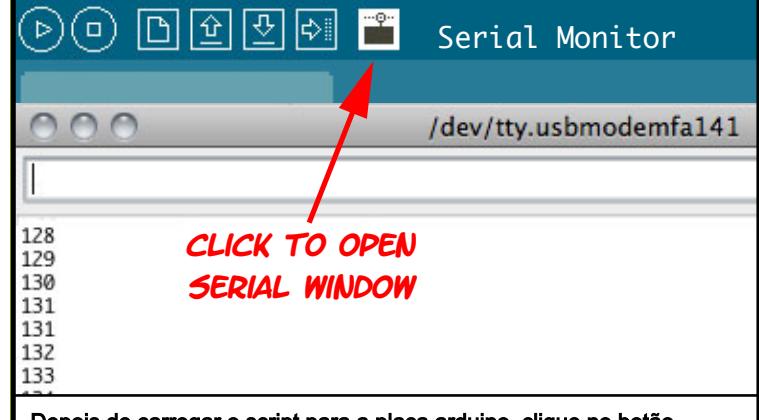
Agora vamos utilizar um input analógico, um potenciômetro.



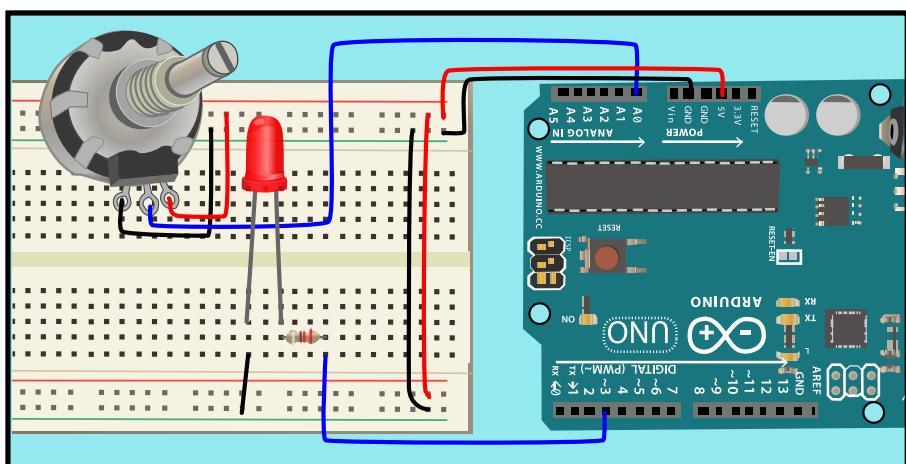
Conecte o terminal do centro do potenciômetro na porta analógica 0 do arduino (A0). Conecte um dos outros terminais ao terra e o outros a força.

```
void setup() {  
    Serial.begin(9600);  
}  
  
void loop() {  
    Serial.println(analogRead(A0));  
}
```

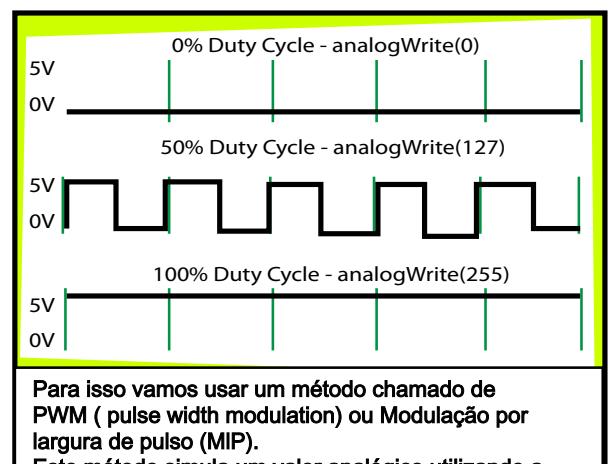
Primeiro vamos verificar os valores que obtemos girando o potenciômetro por meio do terminal serial. No código vamos inicializar a leitura serial e determinar uma velocidade de 9600 bps. No loop, vamos ler o valor obtido do pino A0 e escrever este valor no terminal serial utilizando a função `println`.



Depois de carregar o script para a placa arduino, clique no botão "Serial Monitor" para abrir a nova janela e ver os valores. Os valores exibidos terão uma gama entre 0 e 1024 enquanto giramos o potenciômetro



Agora vamos usar os valores variáveis do potenciômetro para aumentar e diminuir a luminosidade do led (dimmer). Coloque um led conectado ao pino 3 do arduino.



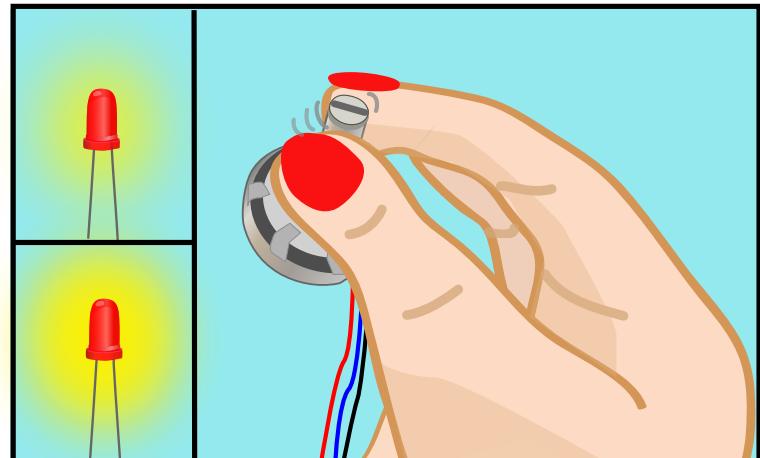
Para isso vamos usar um método chamado de PWM (pulse width modulation) ou Modulação por largura de pulso (MIP). Este método simula um valor analógico utilizando a variação de voltagem em ciclos regulares. As portas 3, 5, 6, 9, 10, e 11 do arduino podem ser usadas com PWM.

```
int sensorValue = 0;

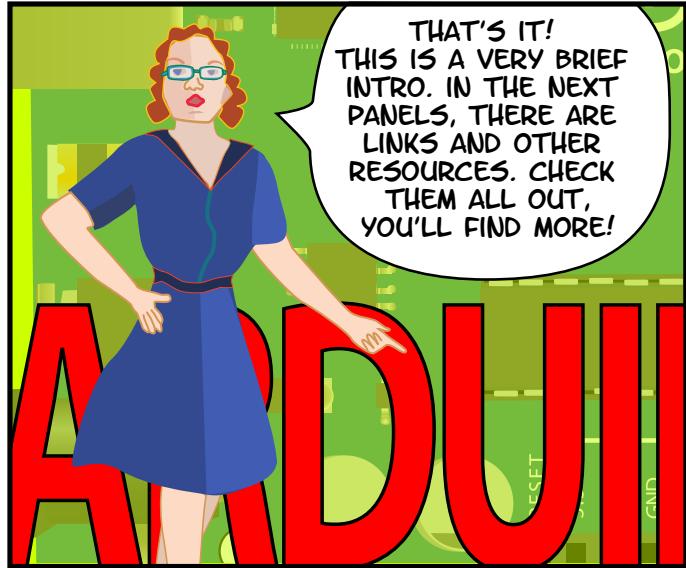
void setup() {
  pinMode(3,OUTPUT);
}

void loop() {
  sensorValue = analogRead(A0);
  analogWrite(3, sensorValue/4);
}
```

Primeiro vamos criar uma variável para estocar o valor do potenciômetro. No setup declaramos o pino 3 como output. No loop, estocamos o valor lido do potenciômetro (A0) em nossa variável. Então, escrevemos este valor para o pino 3 (pino do led). Temos que dividir o valor obtido por 4 para obter valores de 0 a 255 ou um byte.



Mudamos a intensidade da luz variando a posição do potenciômetro.



## TUTORIALS

### ARDUINO SITE TUTORIALS

[HTTP://WWW.ARDUINO.CC/EN/TUTORIAL/HOMEPAGE](http://www.arduino.cc/en/Tutorial/Homepage)

### LADY ADA

[HTTP://WWW.LADYADA.NET/LEARN/ARDUINO/](http://www.ladyada.net/learn/arduino/)

### INSTRUCTABLES

[HTTP://WWW.INSTRUCTABLES.COM/TAG/TYPE-ID/CATEGORY-TECHNOLOGY/CHANNEL-ARDUINO/](http://www.instructables.com/tag/type-id/category-technology/channel-arduino/)

## BOOKS

GETTING STARTED WITH ARDUINO BY MASSIMO BANZI  
MAKING THINGS TALK: USING SENSORS, NETWORKS, AND  
ARDUINO TO SEE, HEAR, AND FEEL YOUR WORLD BY  
TOM IGUE

PHYSICAL COMPUTING: SENSING AND CONTROLLING  
THE PHYSICAL WORLD WITH COMPUTERS BY DAN  
O'SULLIVAN & TOM IGUE

ARDUINO COOKBOOK BY MICHAEL MARGOLIS

## LINKS

### SOFTWARE

#### SOFTWARE DOWNLOAD

[HTTP://WWW.ARDUINO.CC/EN/MAIN/SOFTWARE](http://www.arduino.cc/en/Main/Software)

#### LANGUAGE REFERENCE

[HTTP://ARDUINO.CC/EN/REFERENCE/HOME PAGE](http://arduino.cc/en/Reference/Homepage)

### SUPPLIES

#### SPARKFUN ELECTRONICS

[HTTP://WWW.SPARKFUN.COM/](http://www.sparkfun.com/)

#### ADAFRUIT INDUSTRIES

[HTTP://ADAFRUIT.COM/](http://adafruit.com/)

#### MAKER SHED

[HTTP://WWW.MAKERSHED.COM/](http://www.makershed.com/)

#### JAMECO ELECTRONICS

[HTTP://WWW.JAMECO.COM/](http://www.jameco.com/)

TRADUZIDO PARCIALMENTE PARA O PORTUGUÊS BRASILEIRO POR CLÁUDIO MIKLÓS COM AUTORIZAÇÃO  
DO AUTOR PARA O WORKSHOP "ARDUINO 100 NOÇÃO" REALIZADO NO GAROA HACKER CLUBE - WWW.GAROA.NET.BR

ALL TEXT AND DRAWINGS BY **JODY CULKIN**  
FOR MORE, CHECK OUT [JODYCULKIN.COM](http://JODYCULKIN.COM)

SPECIAL THANKS TO TOM IGOE, MARIANNE PETIT, CALVIN REID, THE FACULTY AND STAFF OF THE INTERACTIVE TELECOMMUNICATIONS PROGRAM AT NYU, PARTICULARLY DAN O'SULLIVAN, DANNY ROZIN AND RED BURNS. THANKS TO CINDY KARASEK, CHRIS STEIN, SARAH TEITLER, KATHY GONCHAROV & ZANNAH MARSH.

MANY, MANY THANKS TO THE ARDUINO TEAM FOR BRINGING US THIS ROBUST AND FLEXIBLE OPEN SOURCE PLATFORM.

AND THANKS TO THE LIVELY, ACTIVE AND EVER GROWING ARDUINO COMMUNITY.

INTRODUCTION TO ARDUINO BY JODY CULKIN  
IS LICENSED UNDER A CREATIVE COMMONS ATTRIBUTION-NONCOMMERCIAL-SHAREALIKE 3.0 UNPORTED LICENSE.

