Arduino completo

Tiago Barros | tiago@tiagobarros.org

Apresentação

Tiago Barros - @tgfb

Mestre em Ciência da Computação, UFPE / 2007
B.Sc. Ciência da Computação, UFPE / 2003
Tec. Eletrônica, ETFPE / 1998

Engenheiro de Sistemas Sênior do C.E.S.A.R
Especialista em tecnologia, Grupo de Inovação

Professor de pós-graduação e especialização em diversos cursos:
C.E.S.A.R(Recife), Cin/UFPE/Motorola(Recife), Universidade Positivo (Curitiba), Instituto FaberLudens/FISAM/UnC (Curitiba).

Pré-requisitos

- conhecimentos básicos de programação

Conteúdo

- computação física
- conceitos básicos de eletricidade e eletrônica
- plataforma arduino
- sensores e atuadores analógicos e digitais
- bibliotecas do arduino
  - servo library
  - nunchuk library
  - capsense library
  - atuadores sonoros
  - GLCD library (LCD)

computação física

computação física

- uso de computação e eletrônica [sensores e atuadores] na prototipação de objetos físicos para interação com seres humanos
- comportamento implementado por software
- utilização de microcontroladores

computação física

- o objetivo é **interligar** o mundo físico com o mundo virtual
- usar a computação e a interação com a tecnologia para o desenvolvimento das suas atividades
- meio para **comunicação** e **interação** entre **pessoas**

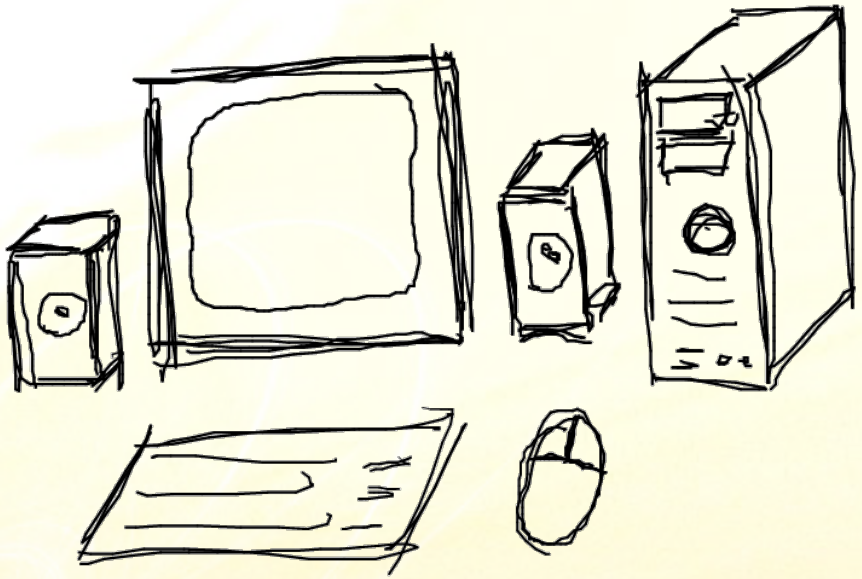
computação física

como vemos os computadores?

computação física

como vemos os computadores?

- teclado
- mouse
- monitor
- CPU
- caixas de som



computação física

como os computadores nos veem?

computação física

como os computadores nos veem?

- dedos
  [teclado/mouse]
- olho
  [monitor]
- duas orelhas
  [caixas de som]

reflexo das entradas e saídas do computador

computação física

“mudar a forma que os computadores nos veem mudará como eles interagem conosco”

Tom Igoe – Physical Computing

computação física

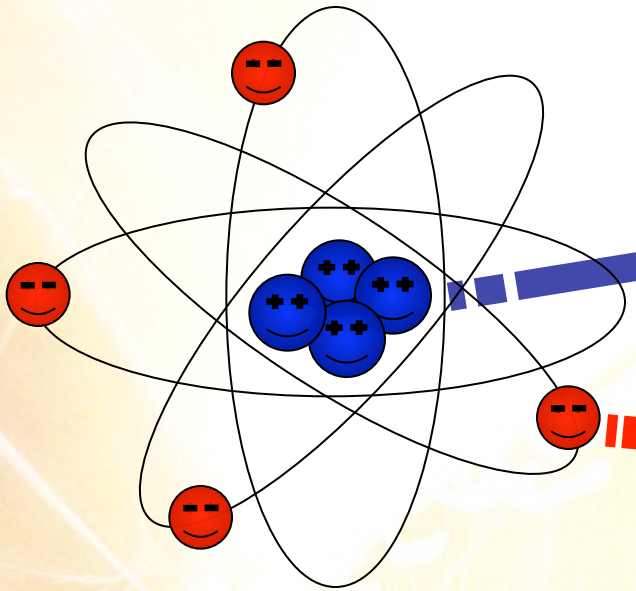
...através de elementos físicos de interação mais adequados às interfaces humanas

Perguntas

mas antes da computação...

conceitos básicos de eletricidade

eletricidade



universo formado de átomos

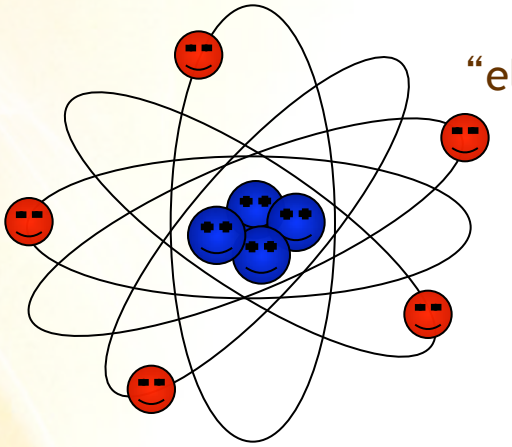
partículas atômicas:

prótons: cargas positivas

elétrons: cargas negativas

eletricidade - interação entre partículas atômicas

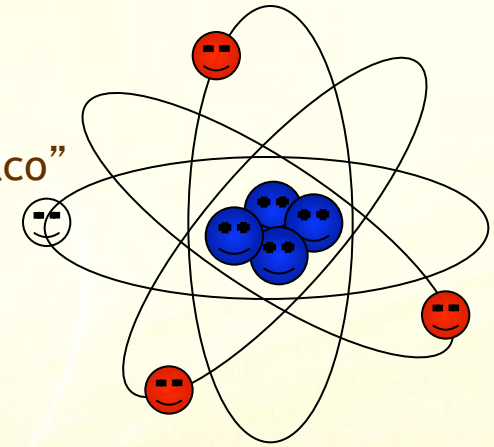
eletricidade



“elétron extra”

Atomos com **mais elétrons** que prótons estão carregados **negativamente** (íon negativo)

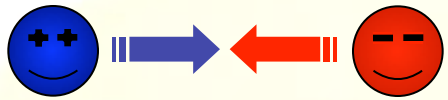
“buraco”



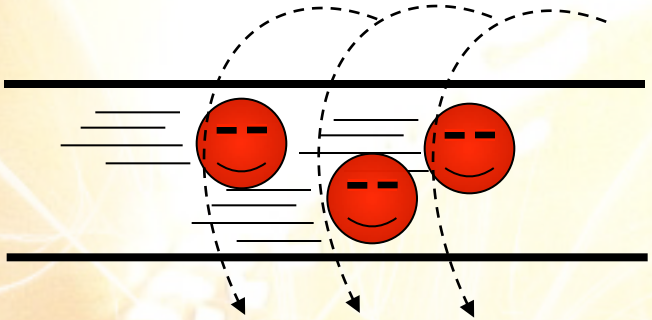
Atomos com **menos elétrons** que prótons estão carregados **positivamente** (íon positivo)

eletricidade

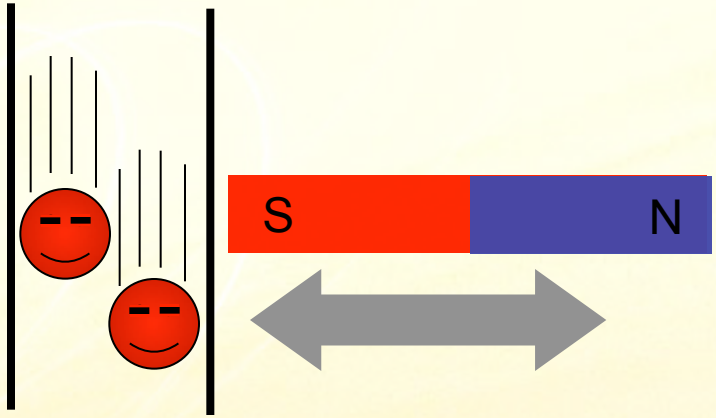
cargas iguais se repelem



cargas opostas se atraem

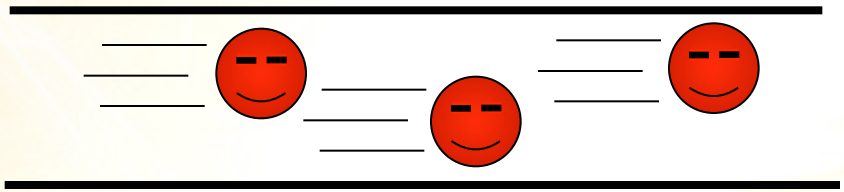


cargas em movimento
geram campo magnético

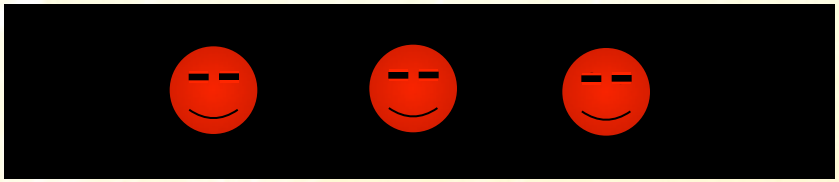


campo magnético em movimento
gera corrente elétrica

eletricidade - condutores e isolantes



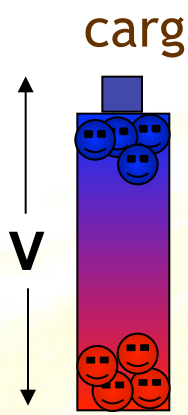
condutor – permite o fluxo de elétrons



isolante – evita a passagem de elétrons

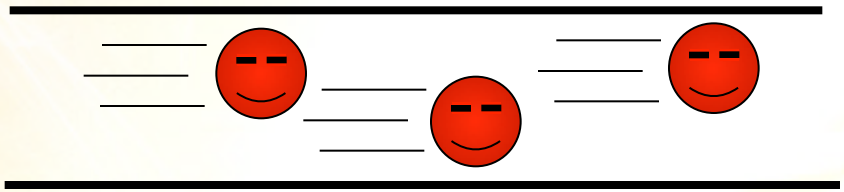
eletricidade - diferença de potencial (v)

diferença de potencial
ou tensão.

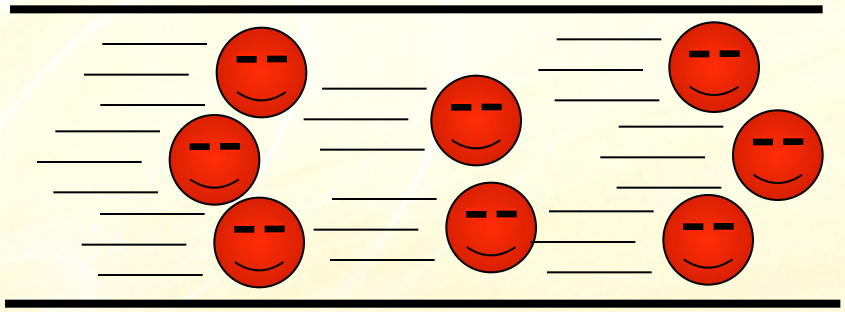


quanto maior a tensão, mais “força” teem os elétrons

eletricidade - corrente elétrica (i)

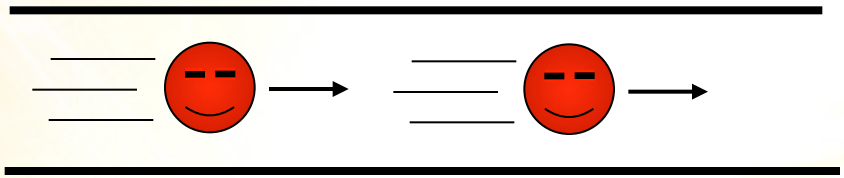


fluxo de elétrons em um condutor

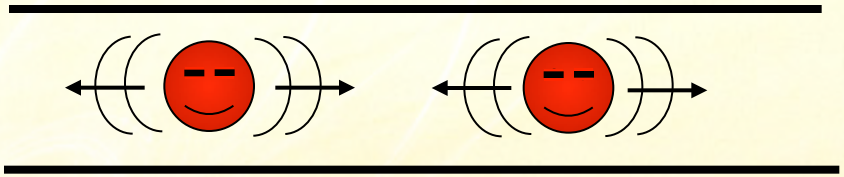


quanto maior a corrente,
maior a “quantidade” de elétrons

eletricidade - tipos de corrente elétrica

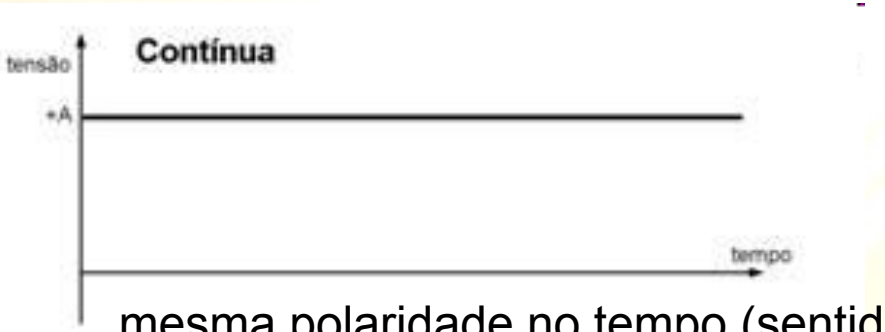


corrente contínua

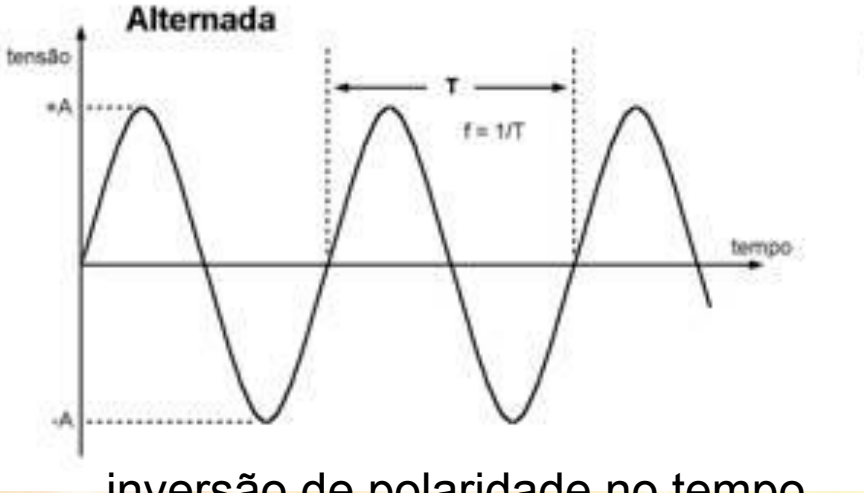


corrente alternada

eletricidade - tipos de corrente elétrica

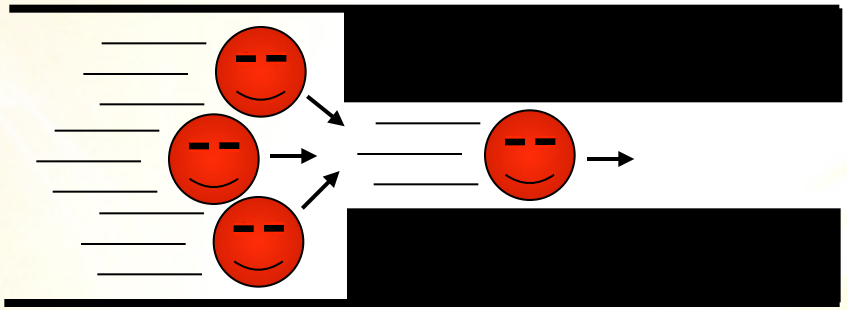


mesma polaridade no tempo (sentido continuo)

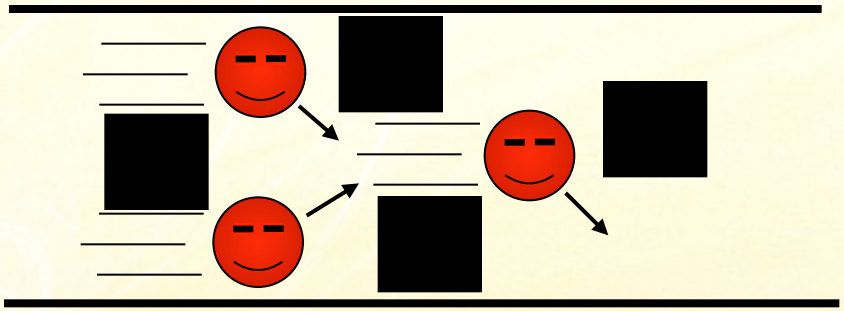


inversão de polaridade no tempo

eletricidade - resistência elétrica (r)



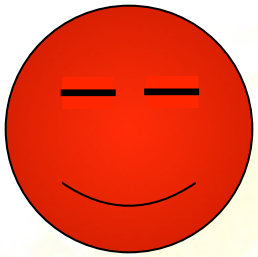
propriedade do material condutor em reduzir
a passagem dos elétrons

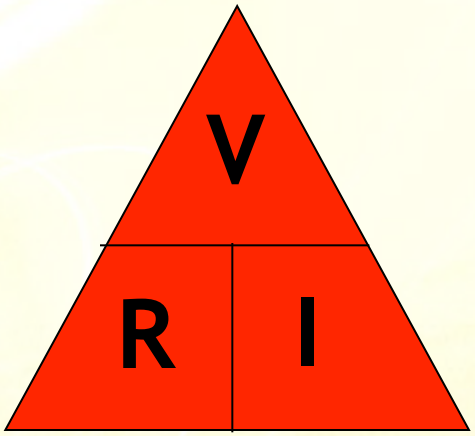


elétrons “se acumulam e batem”
no condutor, “dissipando” sua energia
(gerando calor)

eletricidade - lei de ohm

$$V = R \times I$$

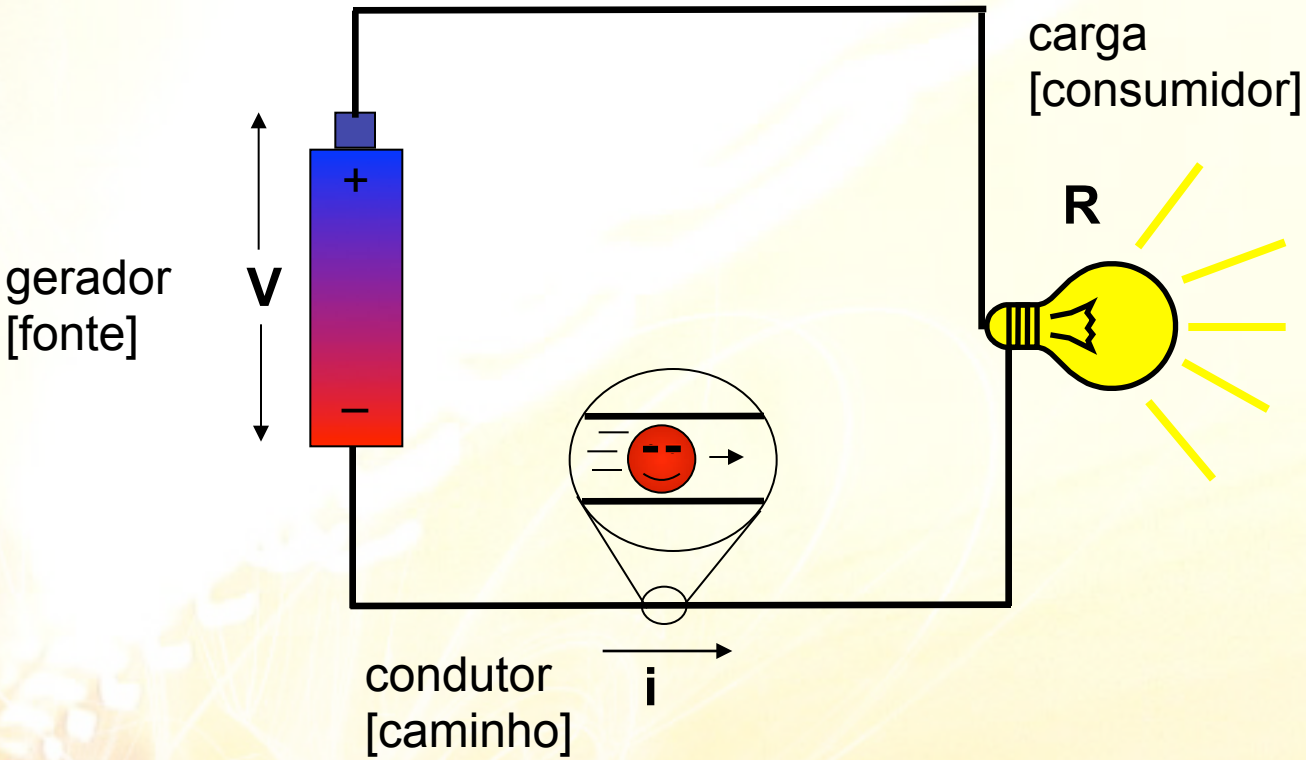




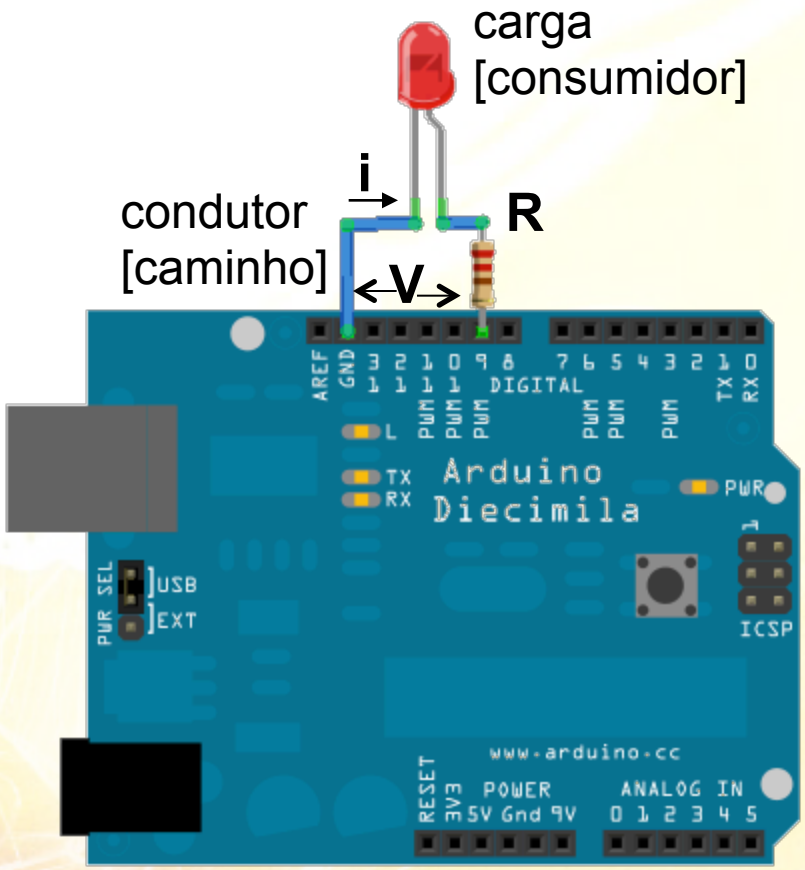
$R = V/I$
$I = V/R$

a diferença de potencial (V) entre dois pontos de um condutor é proporcional à corrente elétrica (I) que o percorre e à sua resistência (R)

eletricidade - circuito elétrico



eletricidade - circuito elétrico



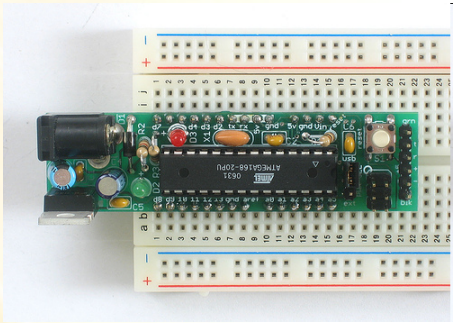
e agora, computação...

plataforma Arduino

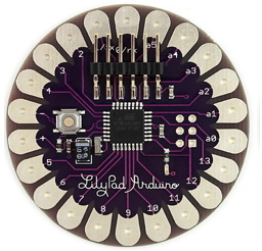
plataforma arduino

- microcontrolador Atmel
- programação usando Wiring (subconjunto de processing, baseado em C/C++)
- open-source: evolução da plataforma através de contribuições dos usuários

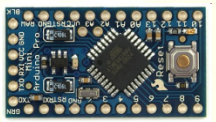
plataforma arduino - hardware



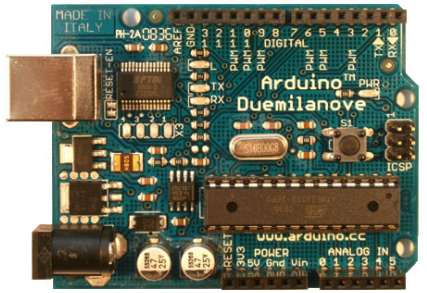
boarduino



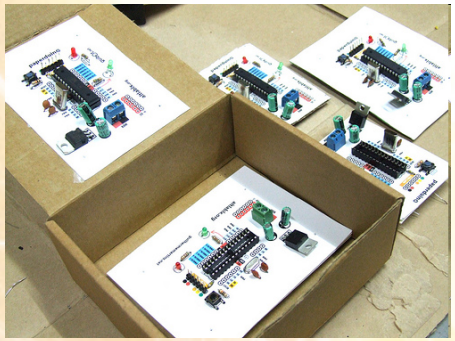
lilypad



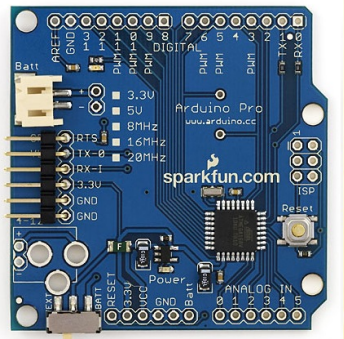
mini



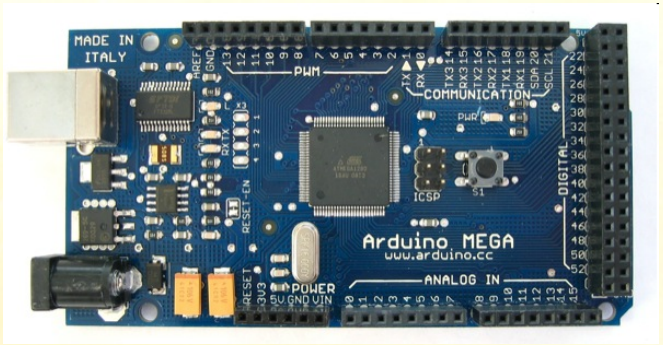
Duemilanove



paperduino

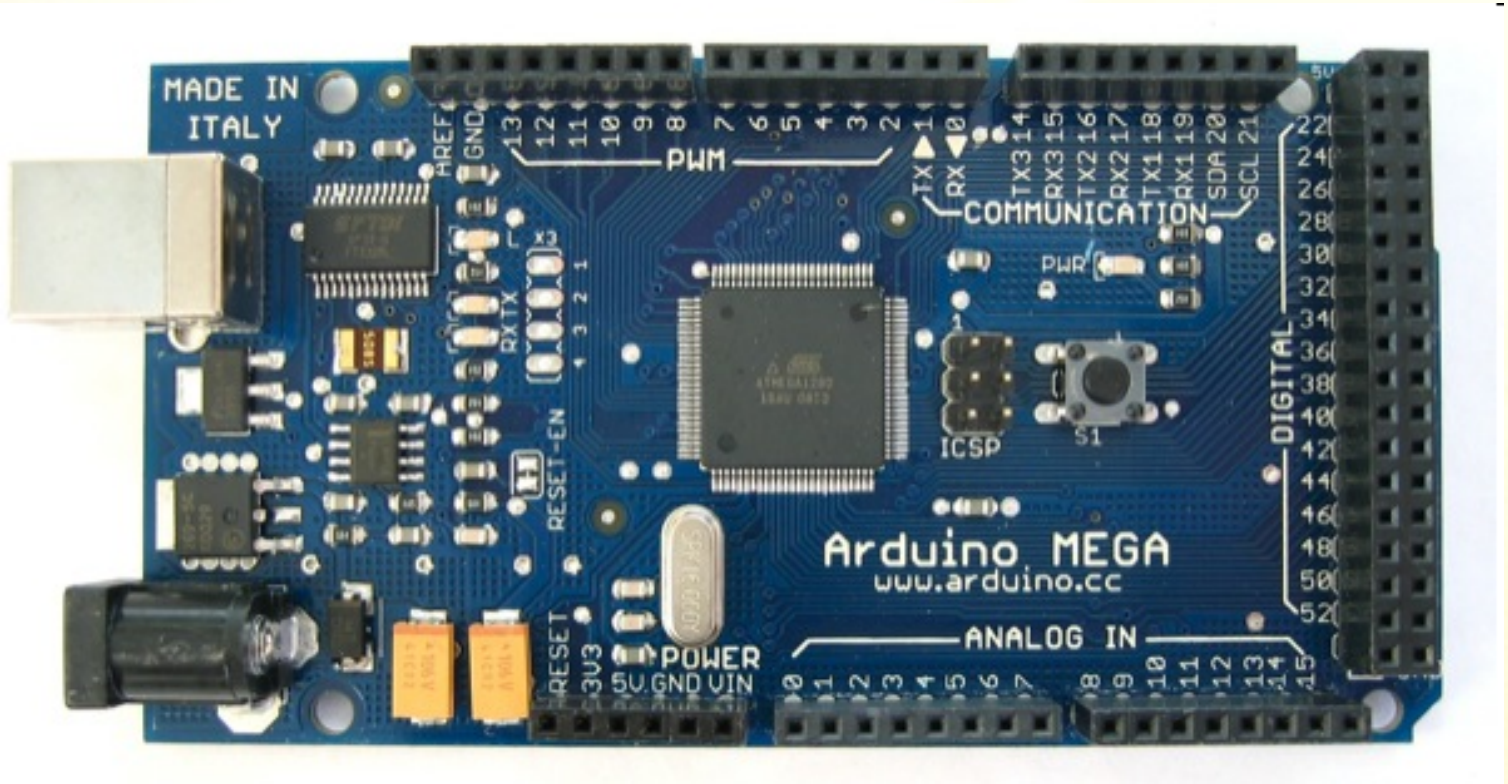


pro



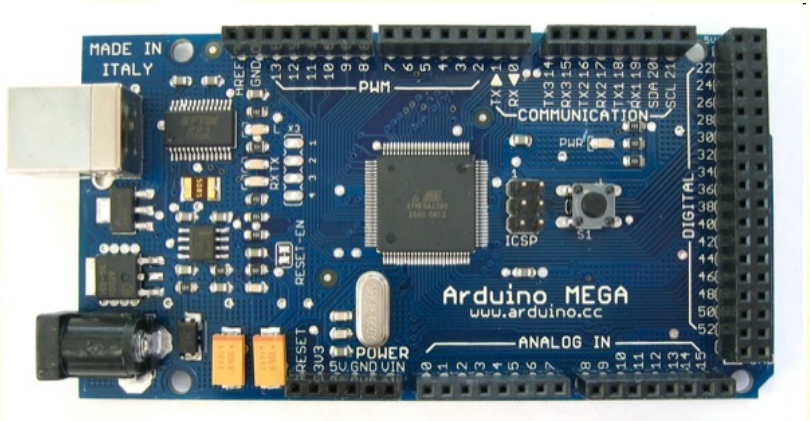
mega

arduino mega - hardware



arduino mega - hardware (Atmega 1280)

- portas
  - 54 entradas/saídas digitais
  - 16 entradas analógicas

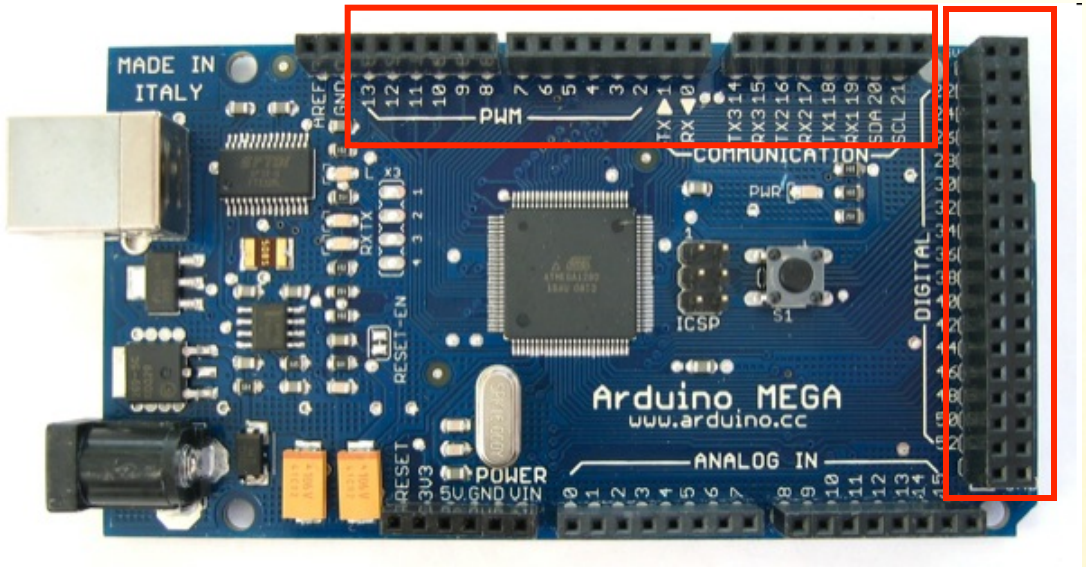


- memória
  - RAM: 8K
  - Flash (programa): 128k – 4k (bootloader)

- velocidade de processamento: 16MHz

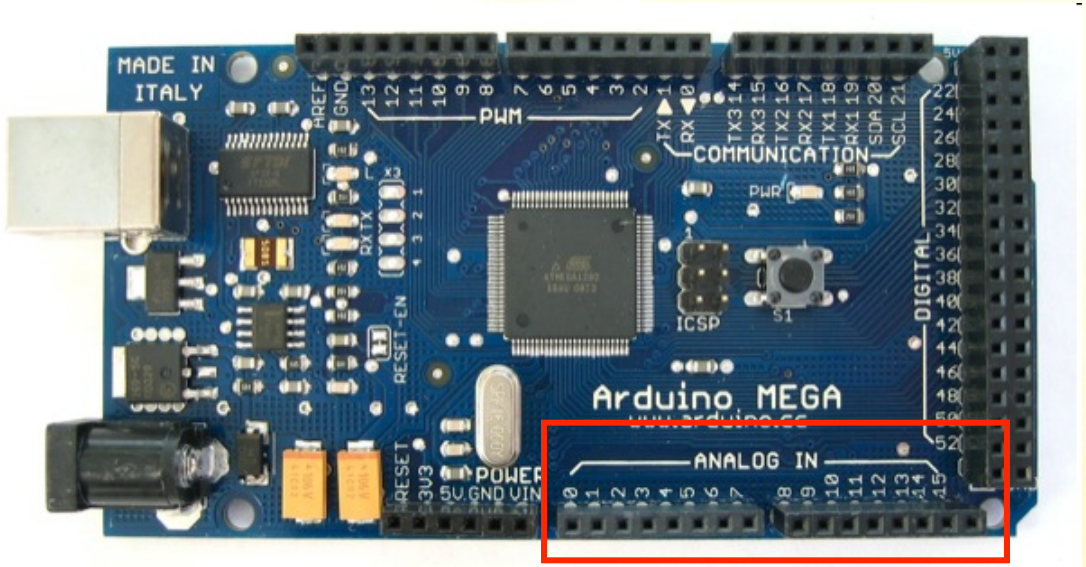
entradas e saídas digitais

- 54 pinos de entradas e saídas digitais (0 – 54)



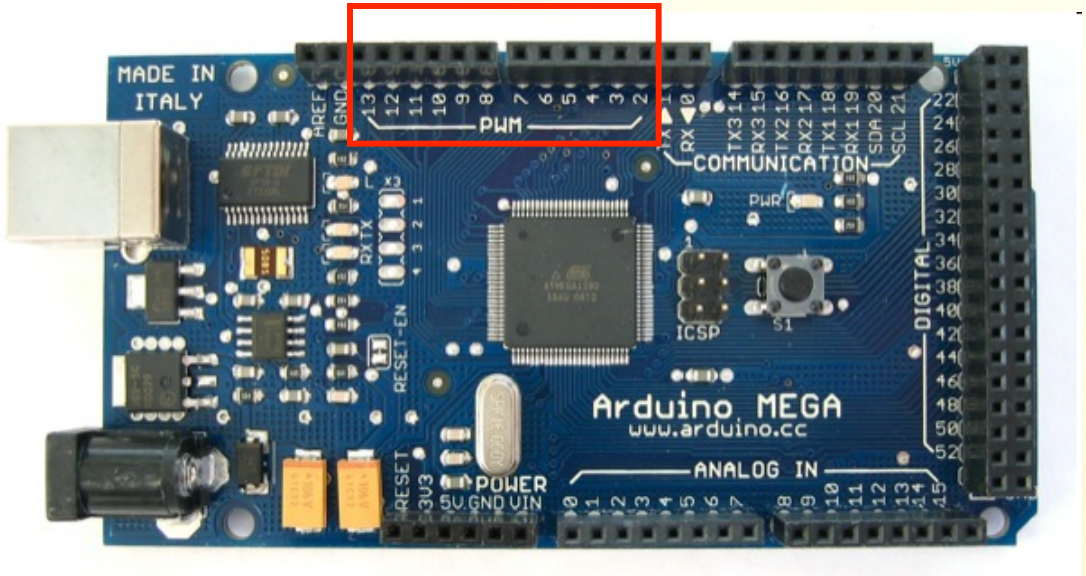
entradas analógicas

- 16 pinos de entrada analógica com resolução de 10 bits (0 – 1023)



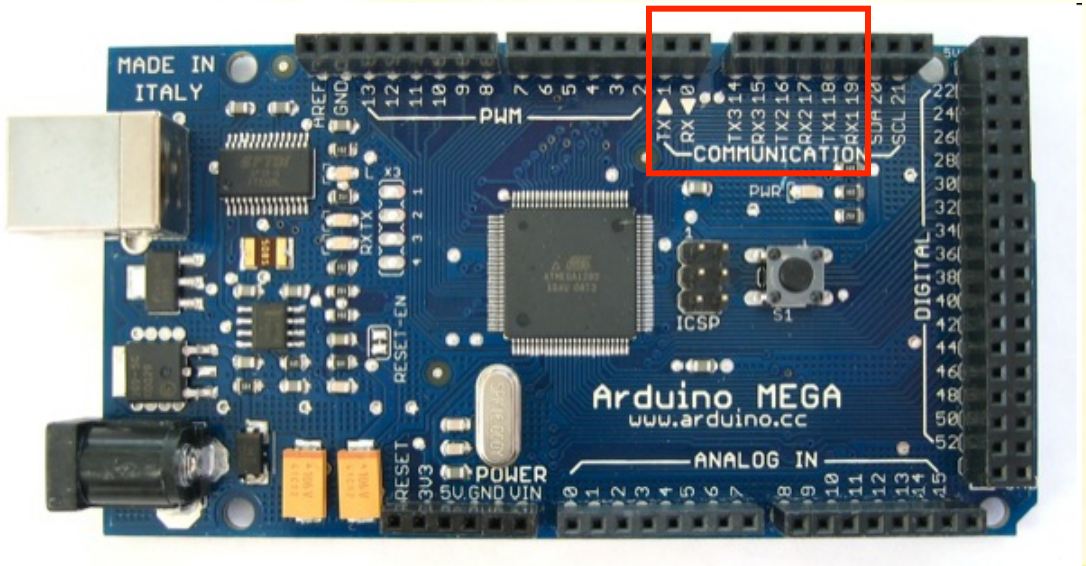
saídas analógicas

- 12 pinos de saída analógica (PWM) com resolução de 8 bits (0 – 255)



portas seriais

- 4 portas seriais
- os objetos `Serial`(conectado à USB), `Serial1`, `Serial2 e Serial3` são utilizados para acessar essas portas.



plataforma arduino - instalação

- driver

  windows: FTDI Serial USB

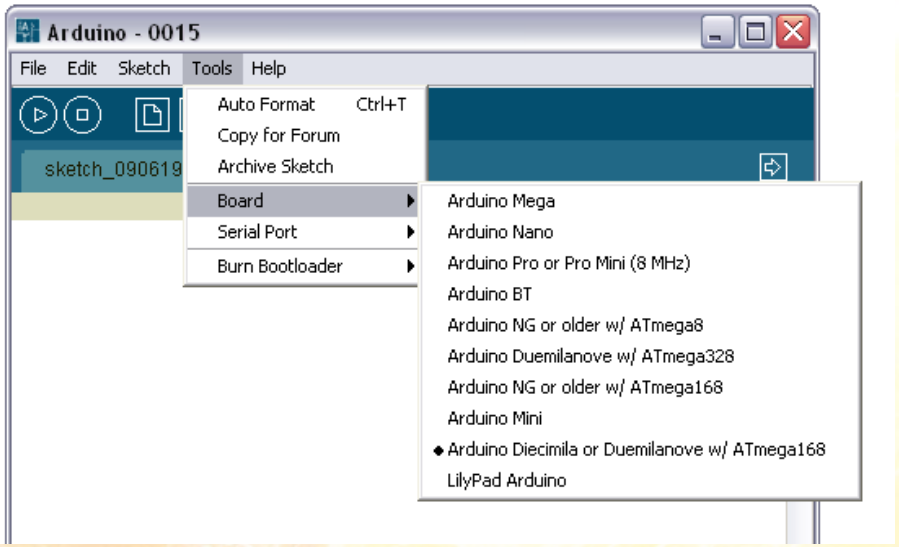
  linux: não precisa instalar nada :-)

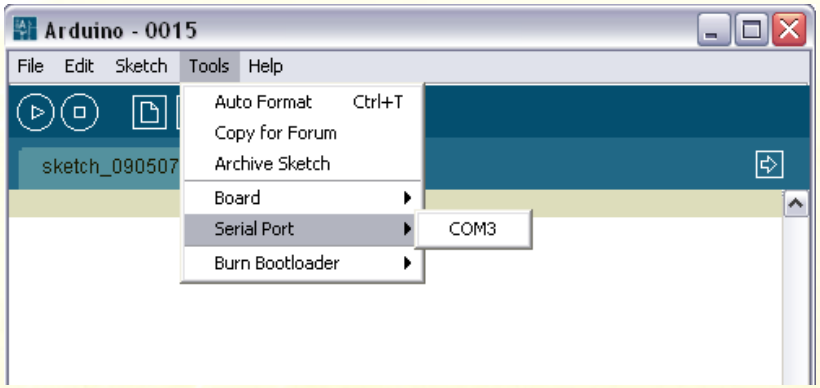
- software

  é só descompactar e executar

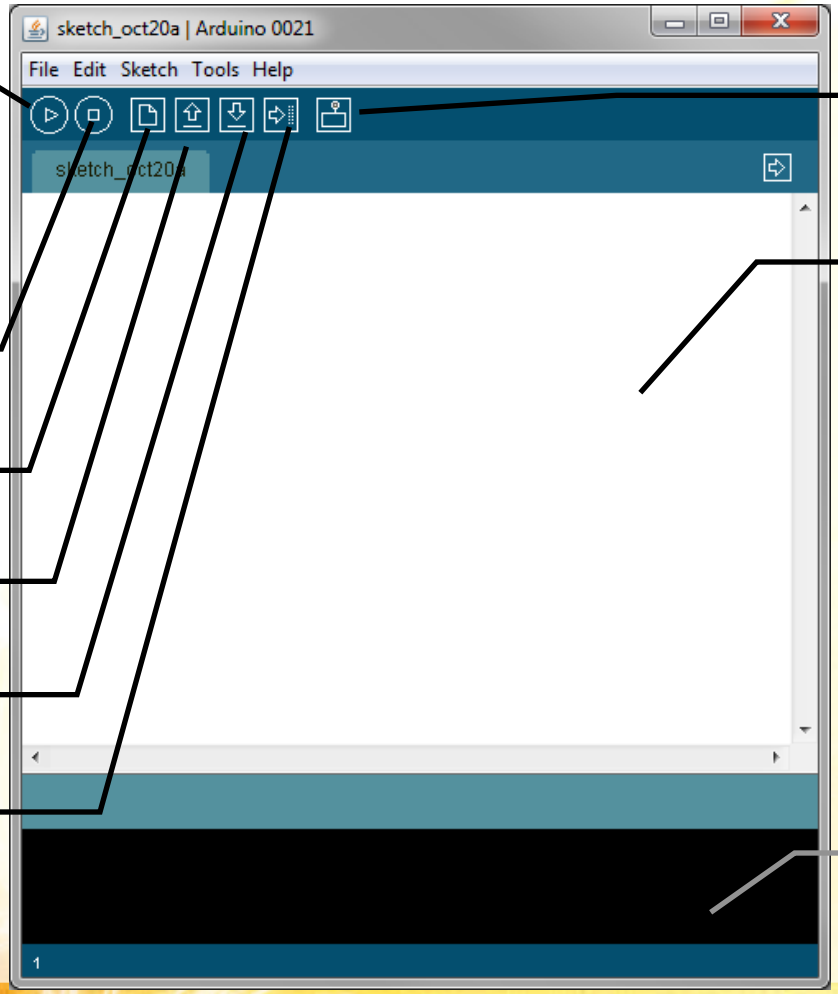
plataforma arduino - instalação

- Selecionando a placa e a porta serial





plataforma arduino - ambiente



compilar (verif. programa)

parar execução

novo

abrir

salvar

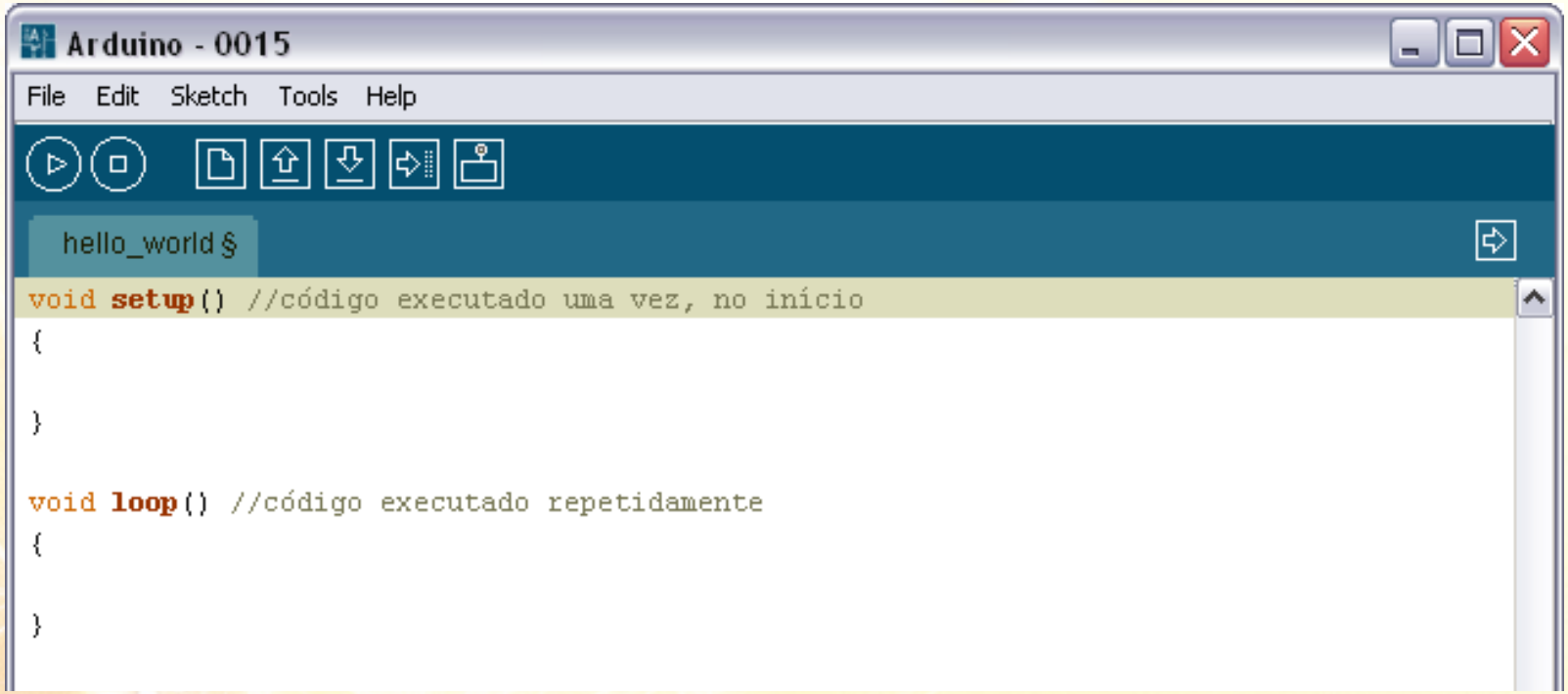
enviar programa para placa

exibir serial

área de código

área de status

plataforma arduino - estrutura do sketch



plataforma arduino - linguagem

- linguagem baseada em C (mas bem mais fácil)
- comandos básicos
  - pinMode() – define um pino com entrada ou saída
  - digitalWrite() – liga ou desliga uma saída digital
  - delay() – “espera” um determinado tempo

plataforma arduino - linguagem

- Exemplos
  - `pinMode(num_do_pino, OUTPUT);`
  - `digitalWrite(num_do_pino, valor);`
    valor é LOW ou HIGH (0 ou 1, 0V ou 5V)
  - `delay(milisegundos);`

plataforma arduino - linguagem

- constantes

`LOW | HIGH` – indica nível baixo (0V) e alto (5V) nos pinos

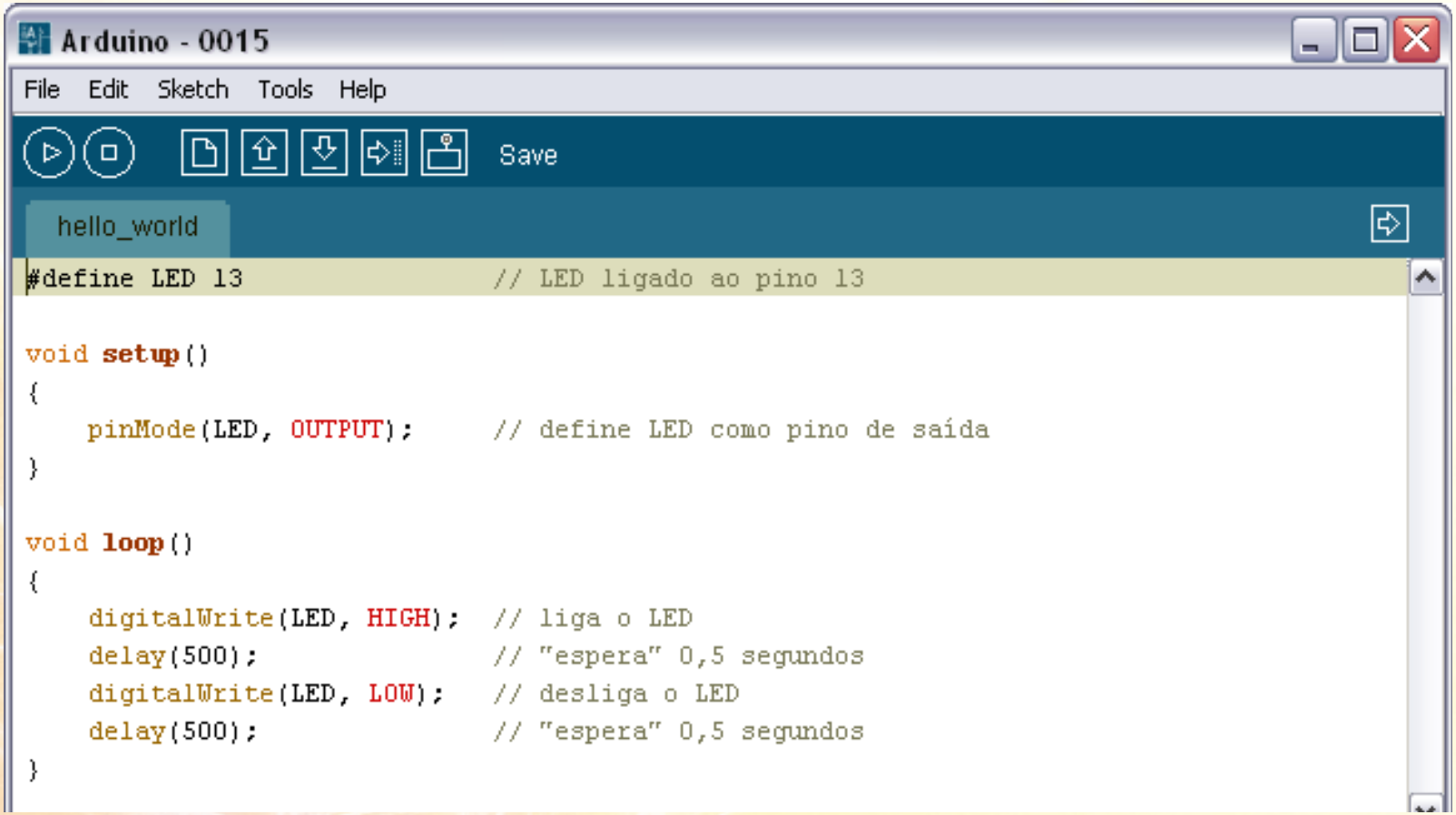
`INPUT | OUTPUT` – define se um pino vai ser pino de entrada ou de saída

atividade prática!

prática

- fazer o programa hello arduino, que pisca um led
- use o pino 13 de saída digital, a placa já possui um led ligado a ele :-)

plataforma arduino - hello arduino

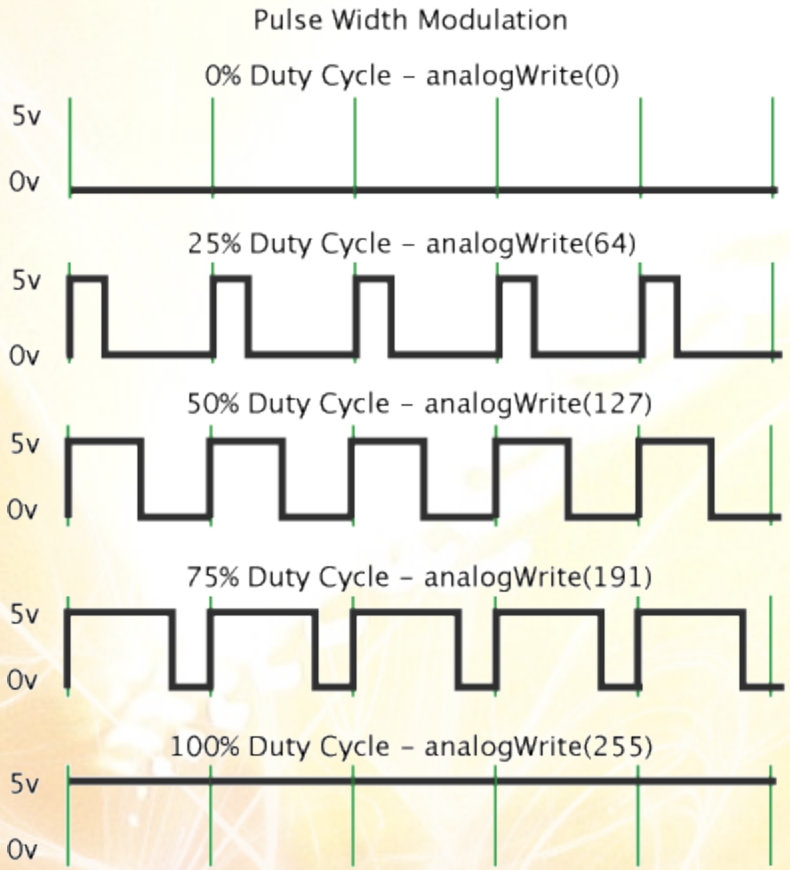


Perguntas

plataforma arduino - linguagem

- comandos básicos
  - analogWrite() – escreve um valor analógico no pino
  - `analogWrite(num_pino, valor);`
    valor entre 0 e 255

eletrônica - modulação PWM



a função analogWrite() escreve “pulsos” muito rápidos no pino digital (só funciona nos pinos marcados com PWM).

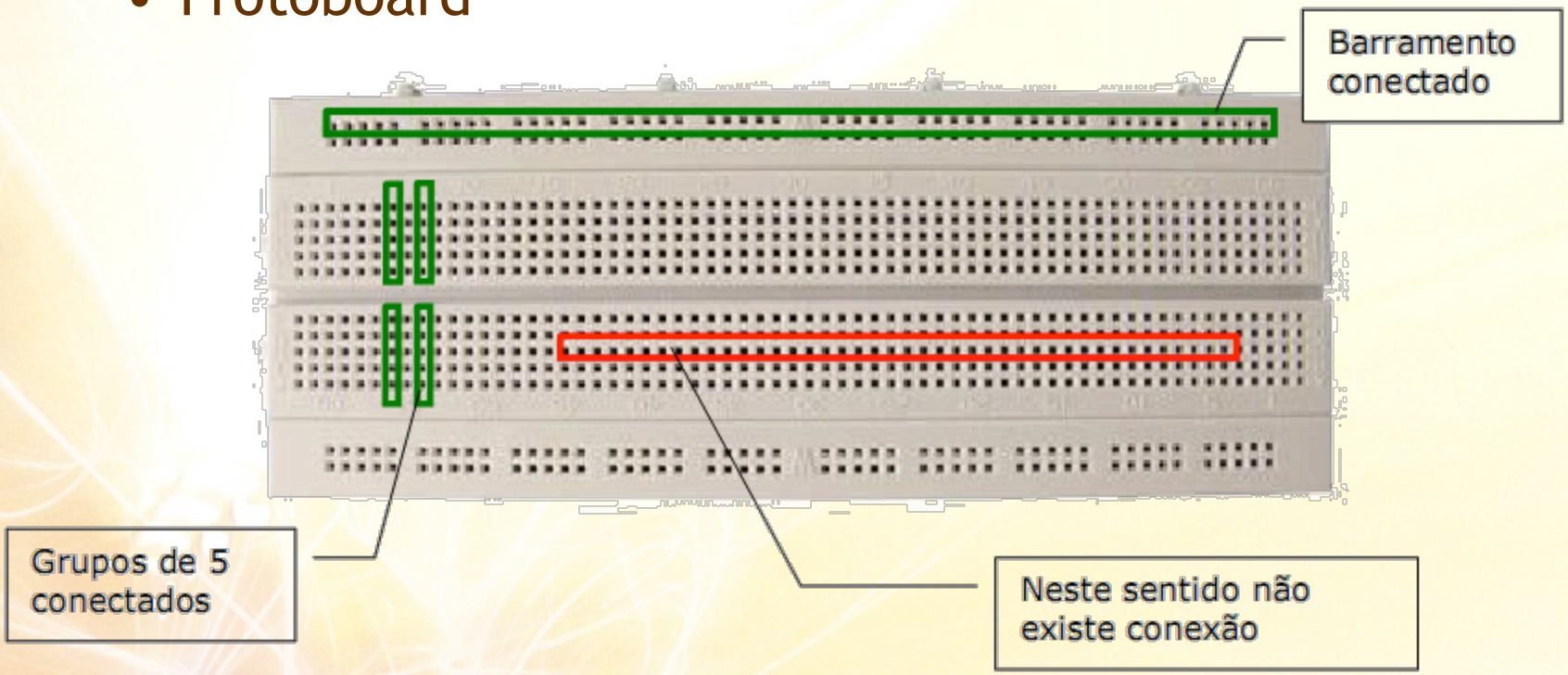
o valor a ser escrito representa o tempo que o pulso fica em nível alto e varia de 0 a 255.

quanto mais tempo o pulto permanecer em nível alto, maior é a “tensão média” da saída

mais prática!

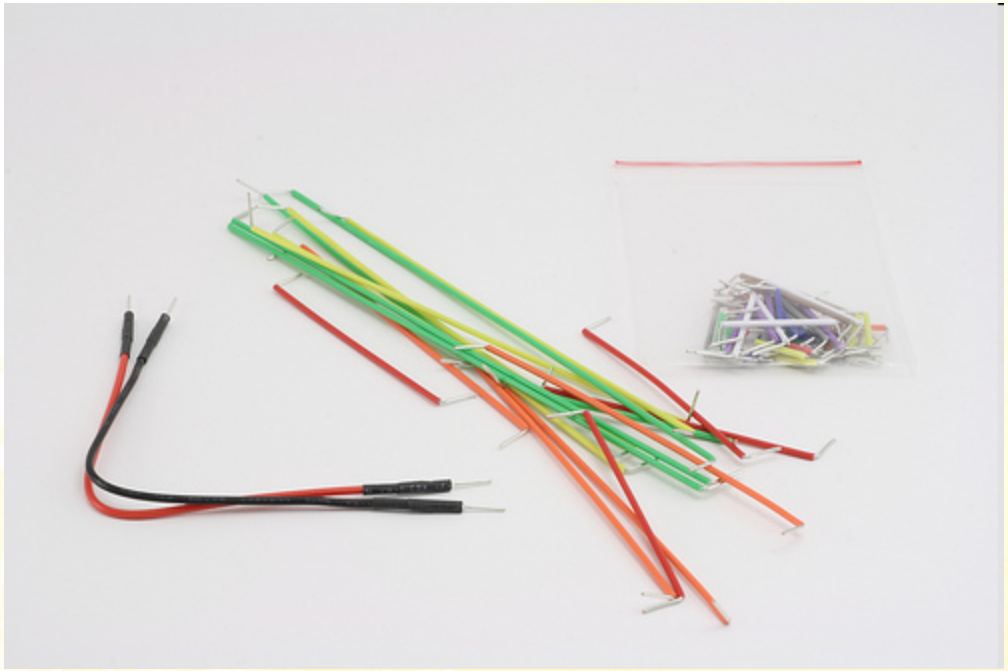
eletrônica - protoboard

- antes disso:
  - Protoboard



eletrônica - protoboard

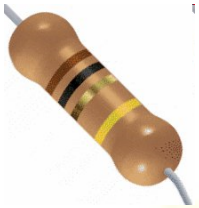
- jumpers



antes disso mais um pouco de eletrônica...

eletrônica - resistores

oferecem resistência à passagem da corrente elétrica

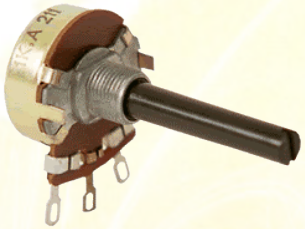


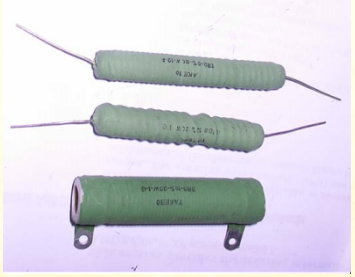
resistência:

fixo
variável

tipos:

carvão [carbono]
filme
fio



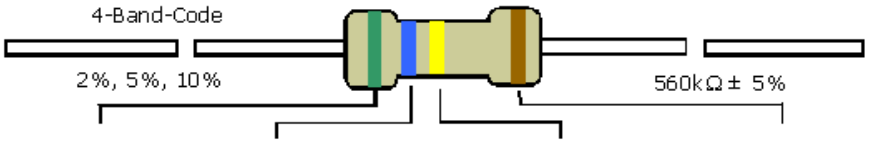


transformam energia elétrica em energia térmica
[pode ser usado como atuador]

eletrônica - resistores

valores expressos em ohms

o corpo dos resistores possui um código de cores para identificar o valor



COLOR	1st BAND	2nd BAND	3rd BAND	MULTIPLIER	TOLERANCE	
Black	0	0	0	1Ω		
Brown	1	1	1	10Ω	± 1%	(F)
Red	2	2	2	100Ω	± 2%	(G)
Orange	3	3	3	1KΩ		
Yellow	4	4	4	10KΩ		
Green	5	5	5	100KΩ	±0.5%	(D)
Blue	6	6	6	1MΩ	±0.25%	(C)
Violet	7	7	7	10MΩ	±0.10%	(B)
Grey	8	8	8		±0.05%	
White	9	9	9			
Gold				0.1	± 5%	(J)
Silver				0.01	± 10%	(K)

agora sim, prática!

prática

- modificar o programa hello arduino para acender o led com efeito de “fading” (acender gradativamente)

- dica: use analogWrite() em vez de digitalWrite(), variando os valores escritos, de 0 a 255

prática

- circuito

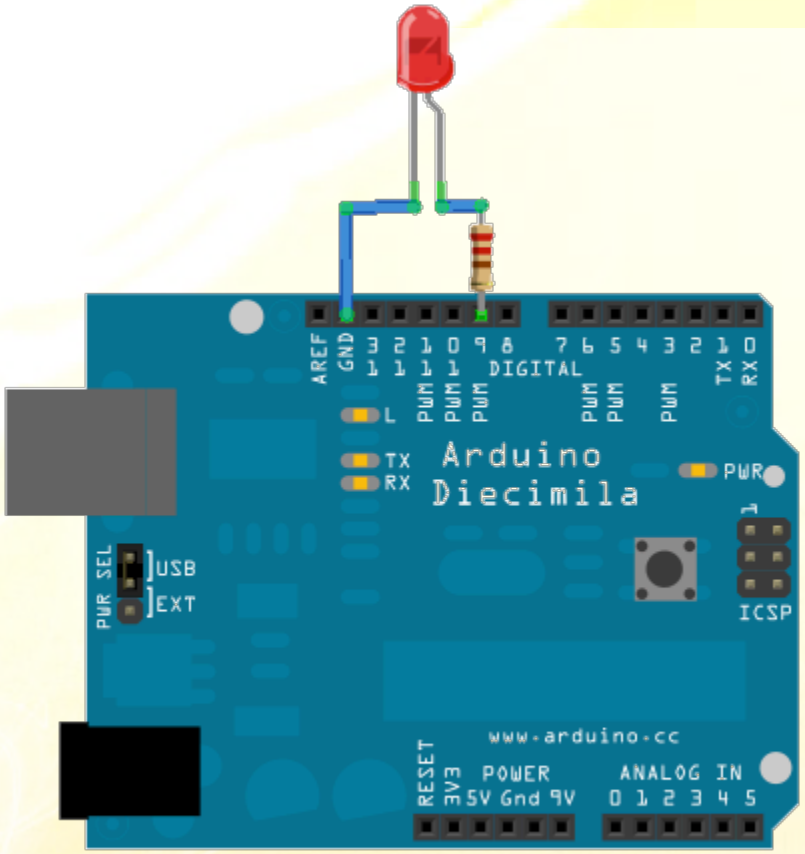


Figura retirada de http://arduino.cc/

prática

- esquemático

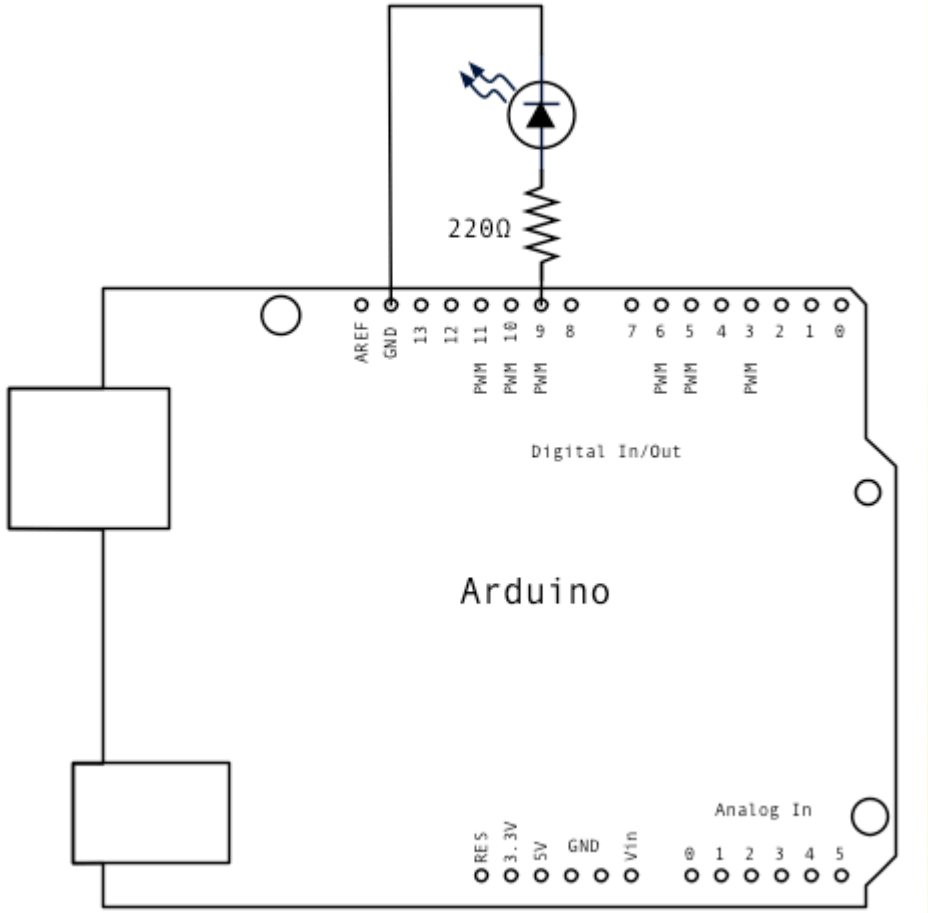


Figura retirada de http://arduino.cc/

prática

- protoboard

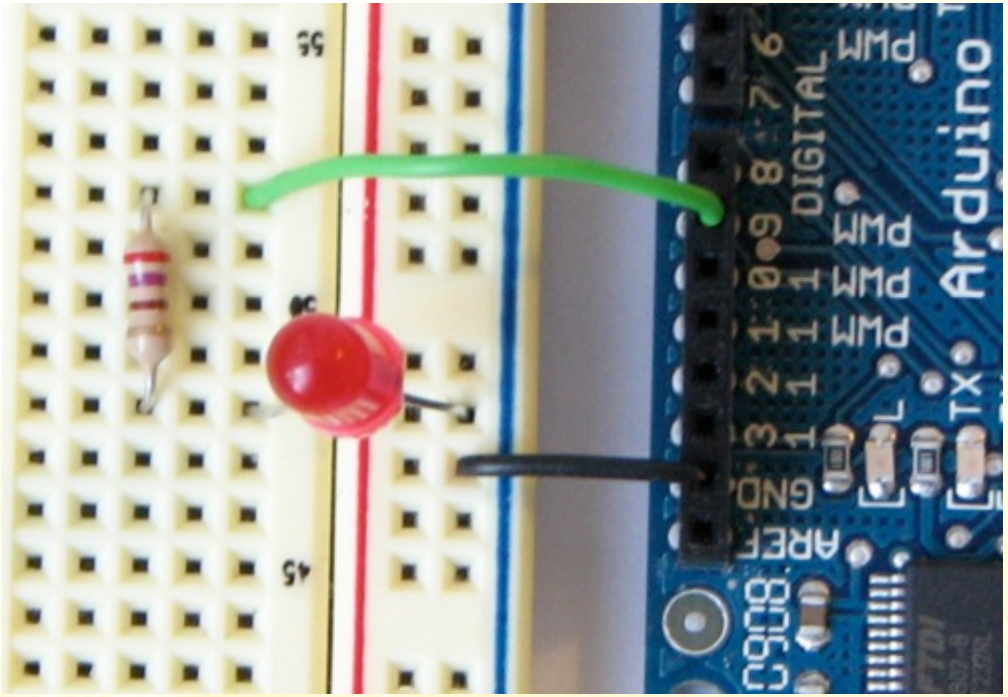


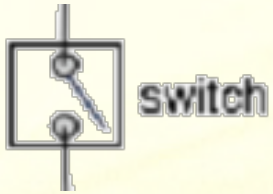
Figura retirada de
http://www.multilogica-shop.com/Aprendendo/Exemplos/Fading

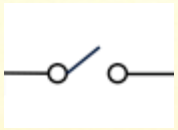
Perguntas

sensores

sensores - chave (switch/button)

- interrompe a passagem da corrente elétrica
- liga/desliga o circuito
- sensor de toque





esquemático

plataforma arduino - linguagem

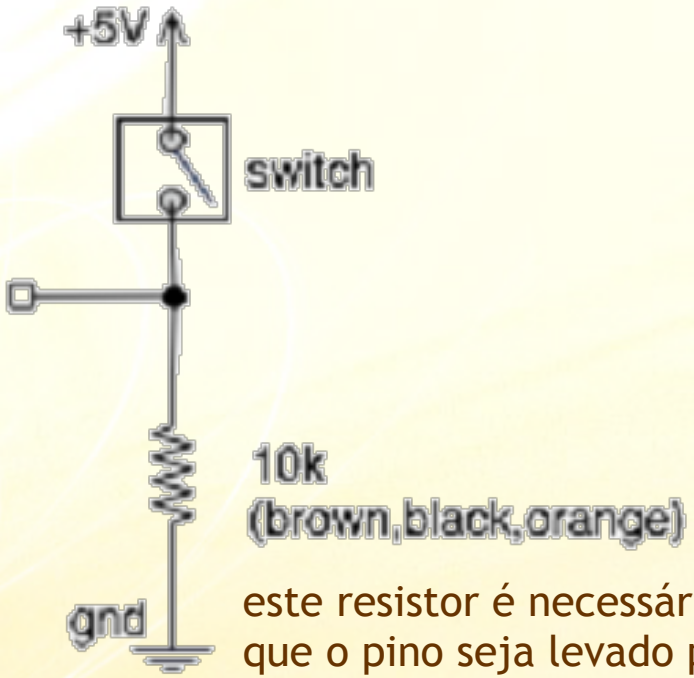
- Comandos
  - digitalRead() – le um pino de entrada
- Exemplo:
  - `int chave = 0;`
  - `chave = digitalRead(num_do_pino);`

sensores

arduino lê tensões de entrada (e não valores 0 e 1)

5 volts == HIGH (1)
0 volts == LOW (0)

sem conexão em um pino, a entrada flutua entre 0 e 5 volts (HIGH e LOW)

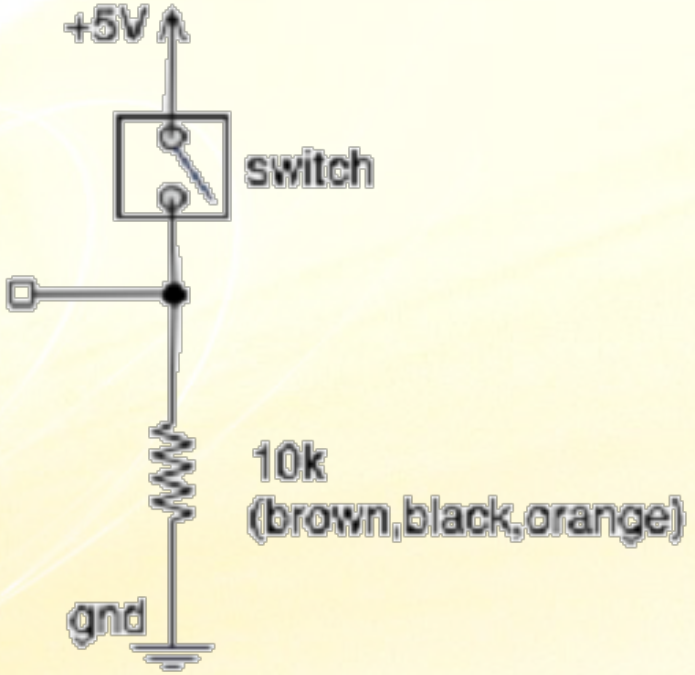


este resistor é necessário para que o pino seja levado para 0 quando não estiver conectado (chave aberta)

atividade prática!

sensores - prática

- fazer o circuito e o programa para acender o led 13 de acordo com sinal de entrada do pino 2



sensores - prática

- esquemático

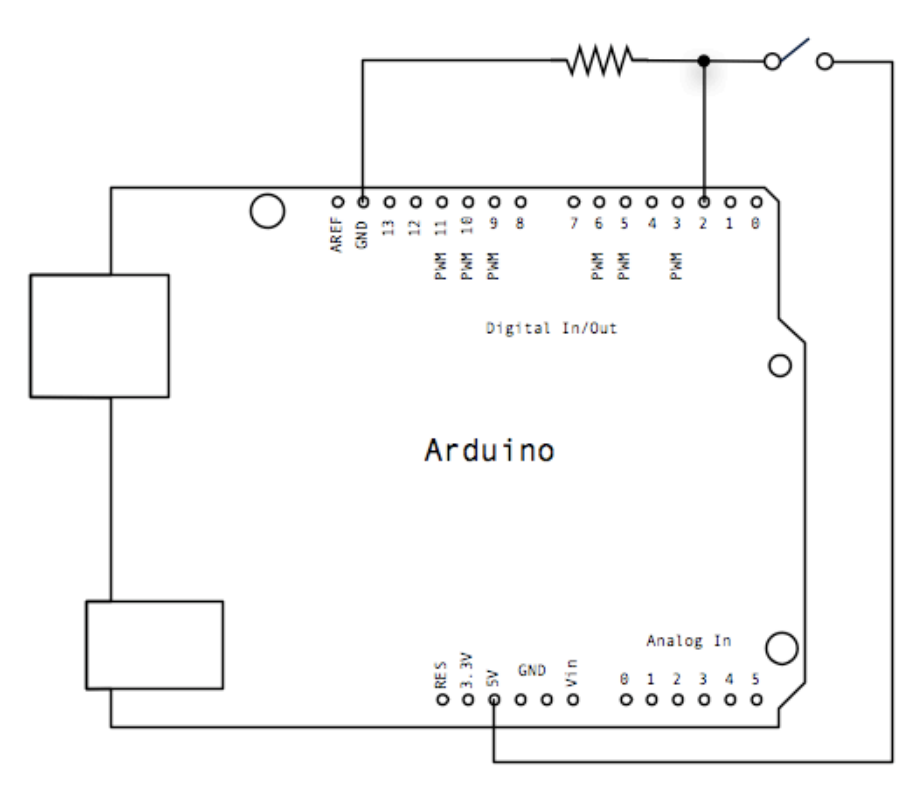
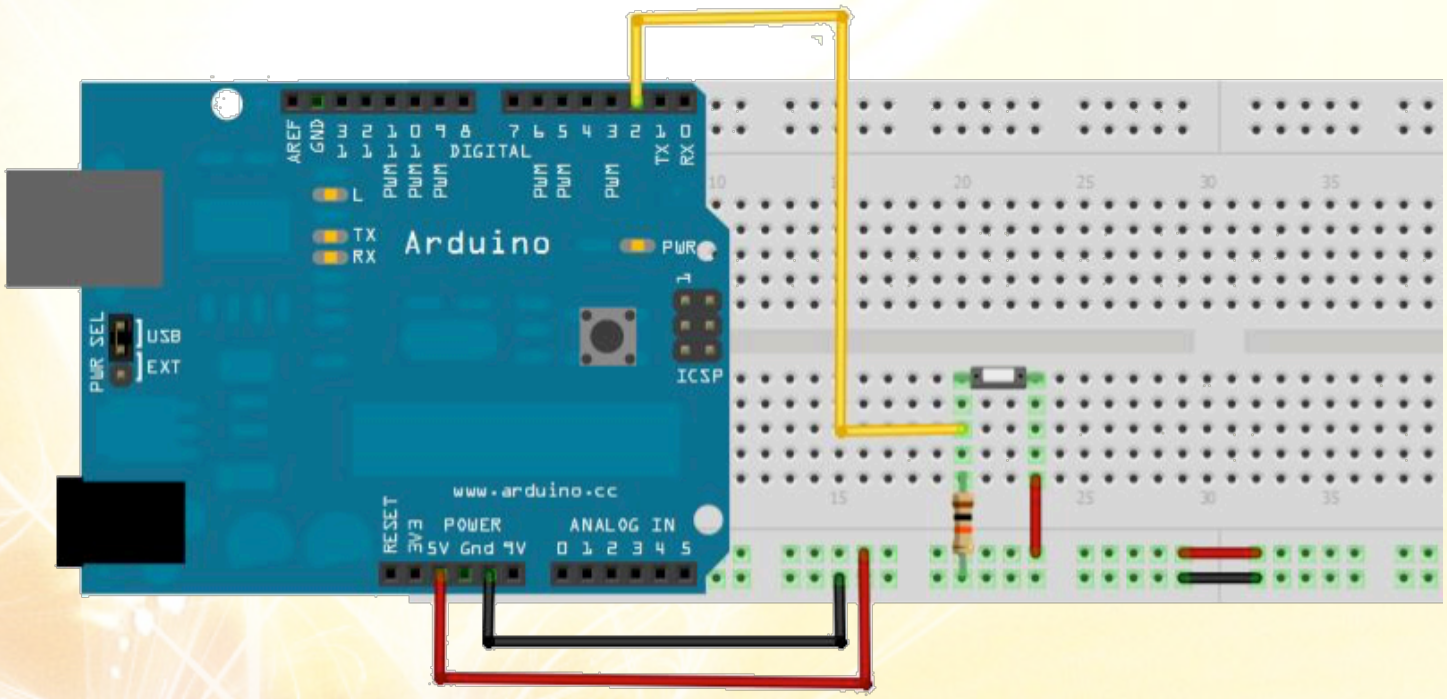


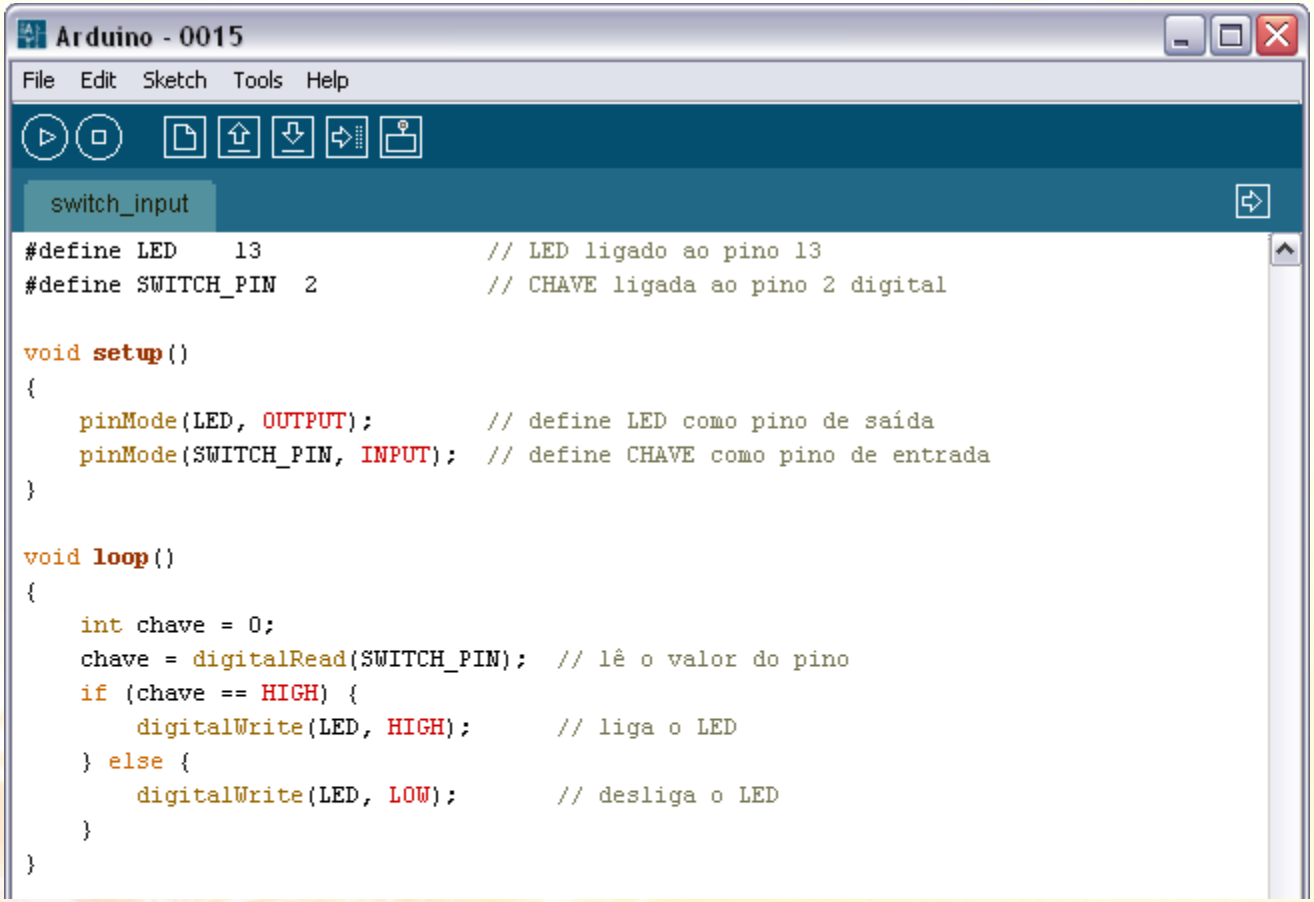
Figura retirada de http://arduino.cc/

sensores - prática

- protoboard



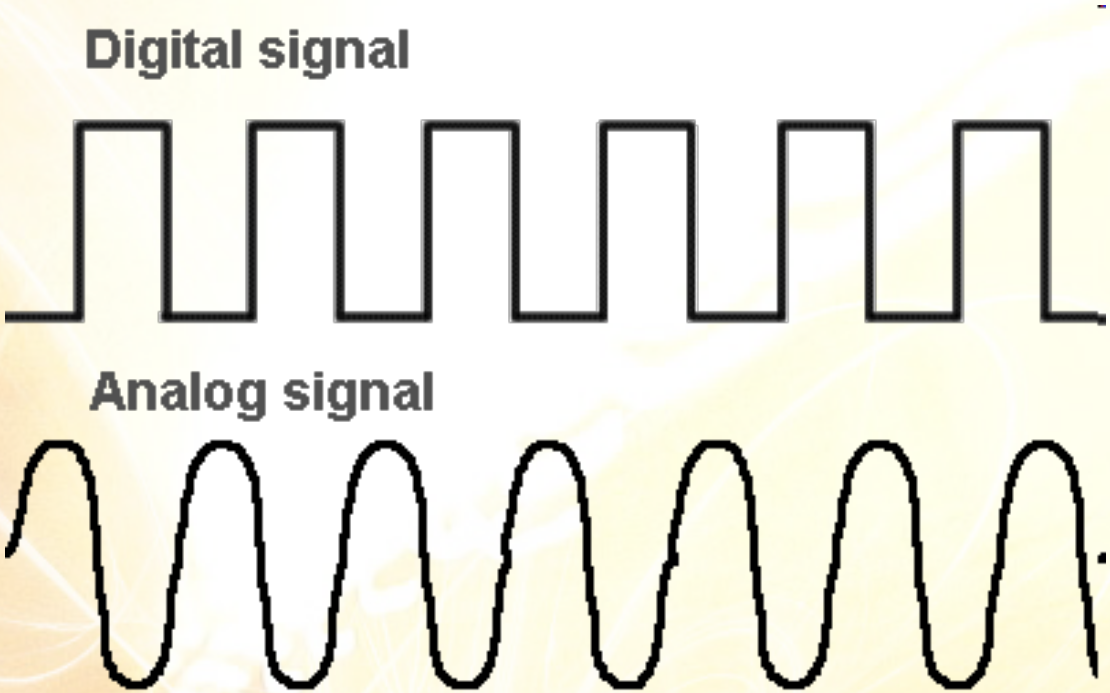
sensores - prática




```
#define LED    13                // LED ligado ao pino 13
#define SWITCH_PIN  2            // CHAVE ligada ao pino 2 digital

void setup()
{
    pinMode(LED, OUTPUT);        // define LED como pino de saída
    pinMode(SWITCH_PIN, INPUT);  // define CHAVE como pino de entrada
}

void loop()
{
    int chave = 0;
    chave = digitalRead(SWITCH_PIN);  // lê o valor do pino
    if (chave == HIGH) {
        digitalWrite(LED, HIGH);      // liga o LED
    } else {
        digitalWrite(LED, LOW);       // desliga o LED
    }
}
```


voltando à eletrônica...

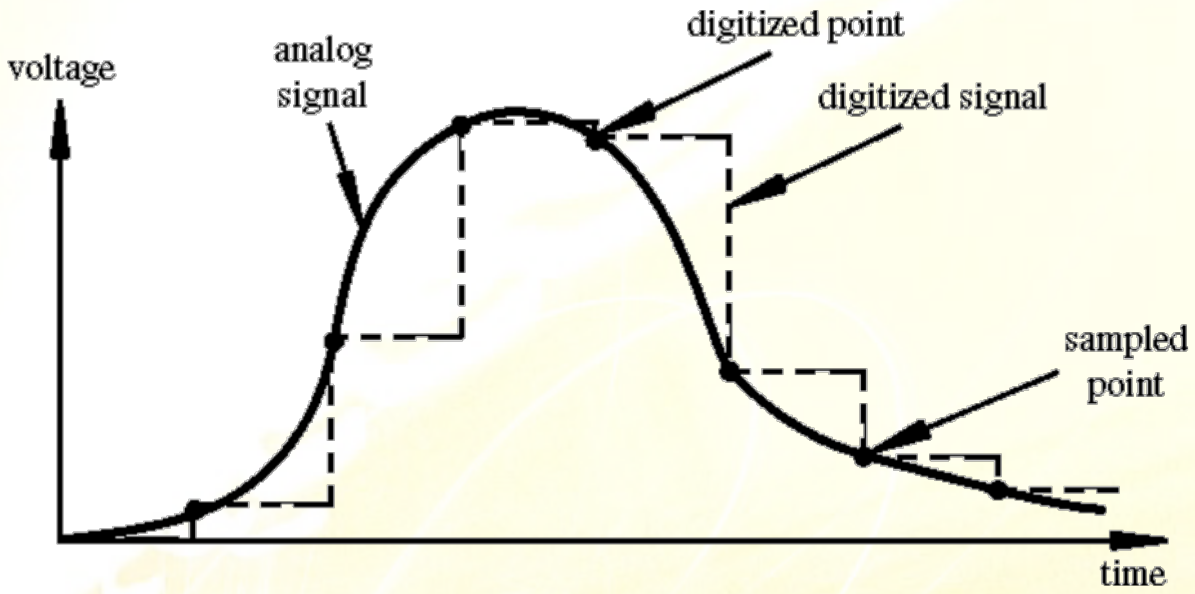
eletrônica - sinais analógicos e digitais



sinal com variação discreta (valores pré-definidos)

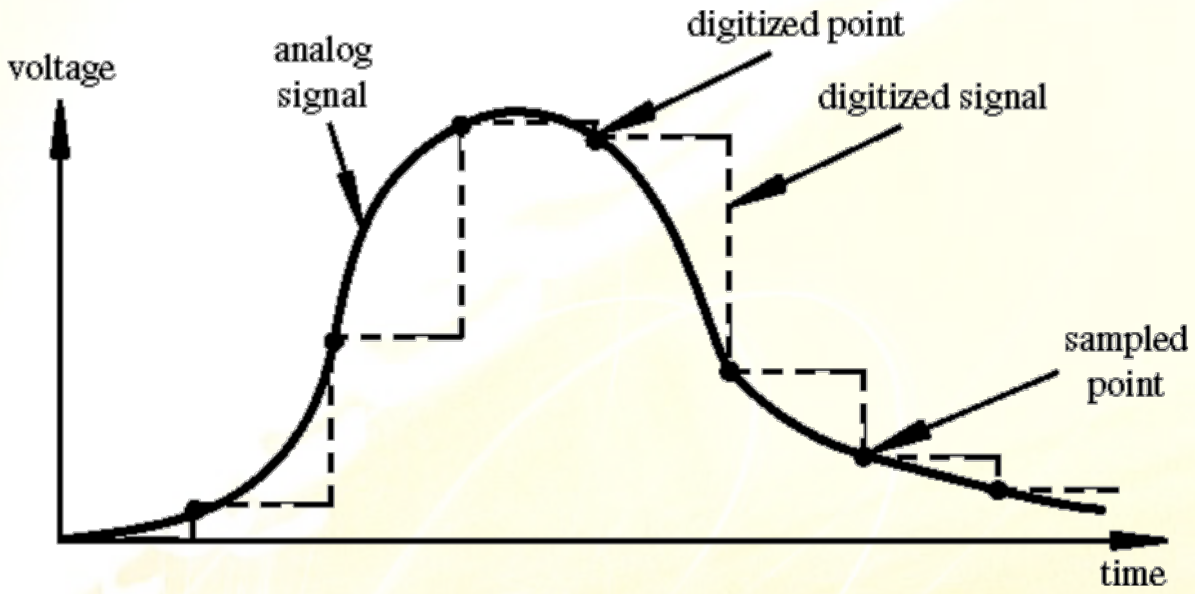
sinal com variação contínua no tempo

eletrônica - conversão de sinais



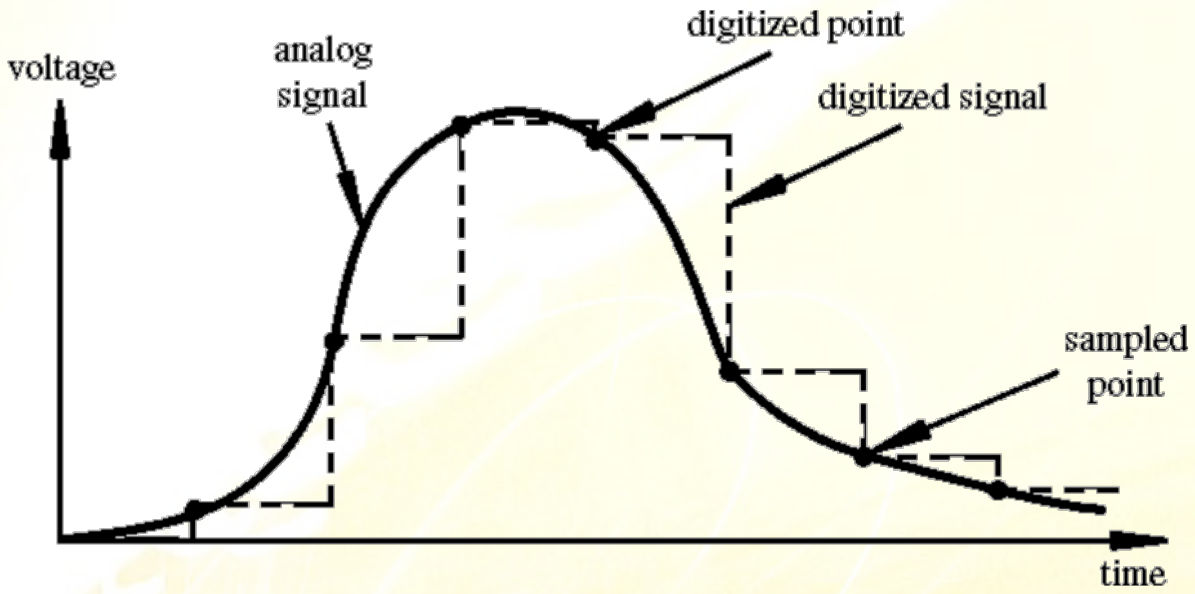
valor é lido em intervalos regulares de tempo e transformado em um número digital

eletrônica - conversão de sinais



vários valores, não só HIGH e LOW. quantiade de valores é a resolução.

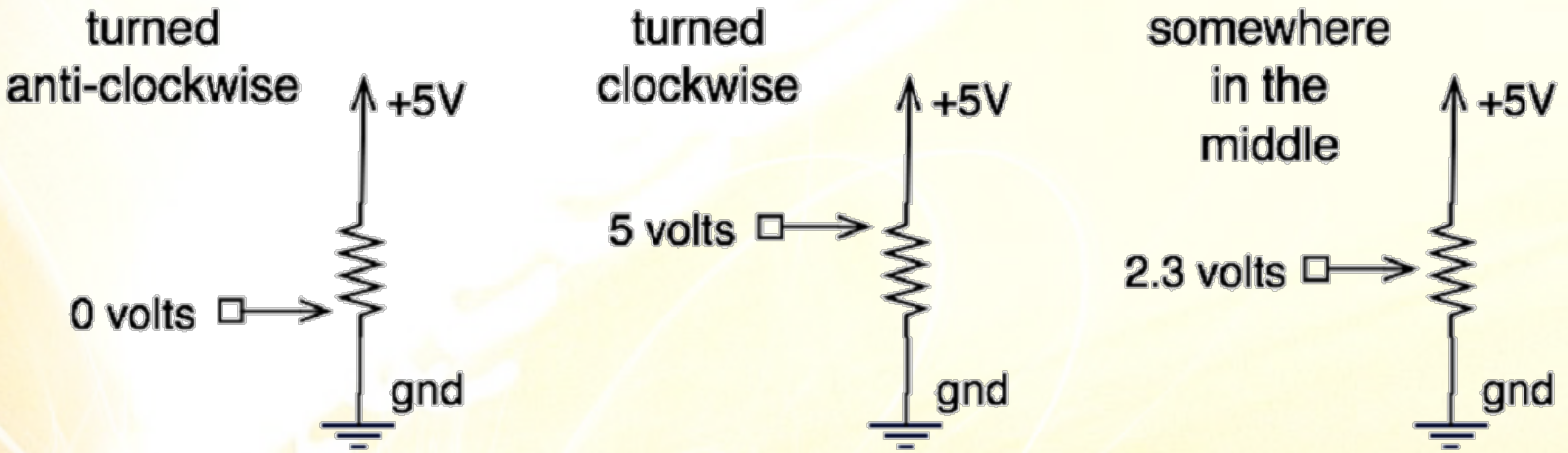
eletrônica - conversão de sinais



resolução de 8 bits = 256 valores
resolução de 16 bits = 65536 valores

eletrônica - resistores

- Como funciona um resistor variável?

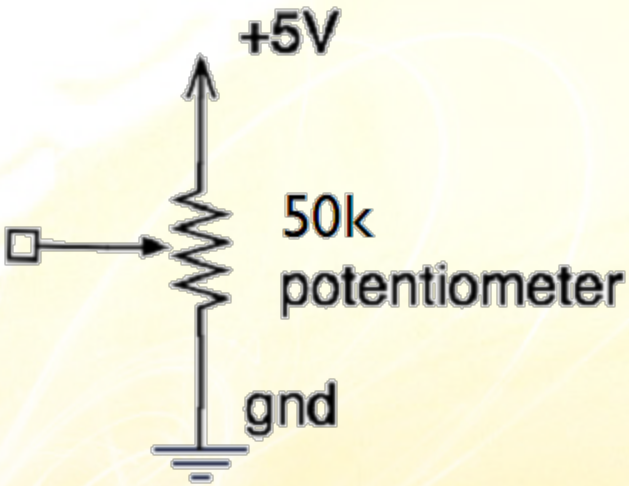


- no arduino, o valor da tensão é transformado em um valor digital entre 0 e 1023

atividade prática!

sensores analógicos - prática

- ler o valor do resistor variável e ligar um LED se esse valor passar de um determinado limite.



sensores analógicos - prática

- esquemático

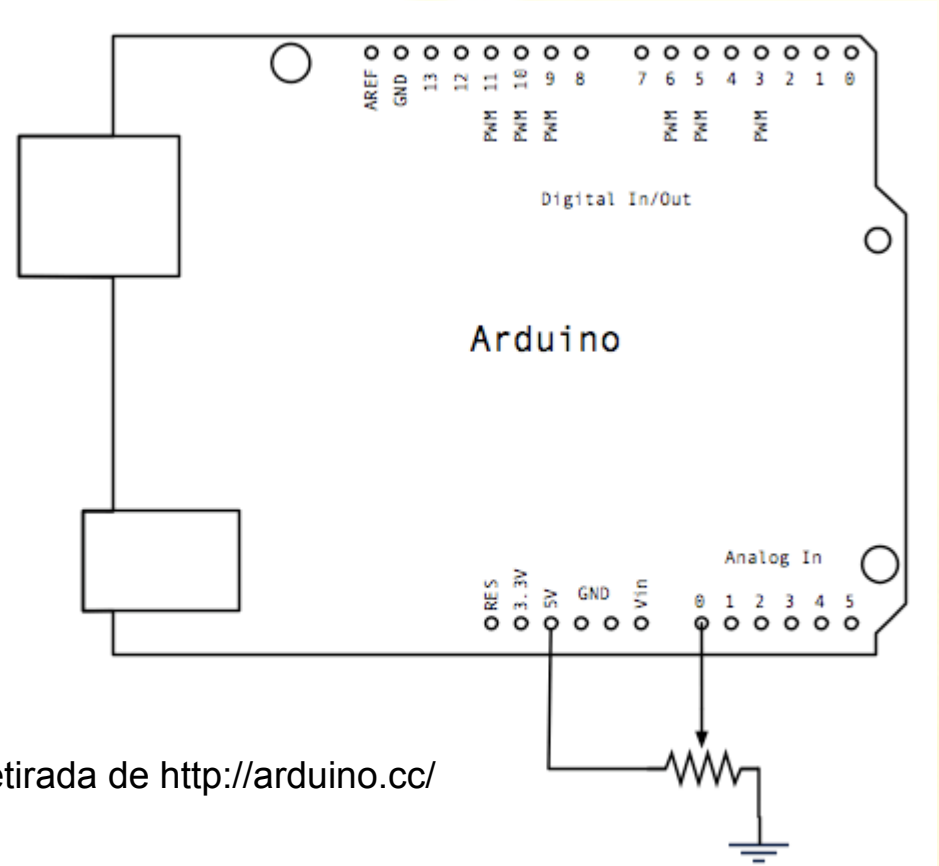


Figura retirada de http://arduino.cc/

sensores analógicos - prática

- circuito

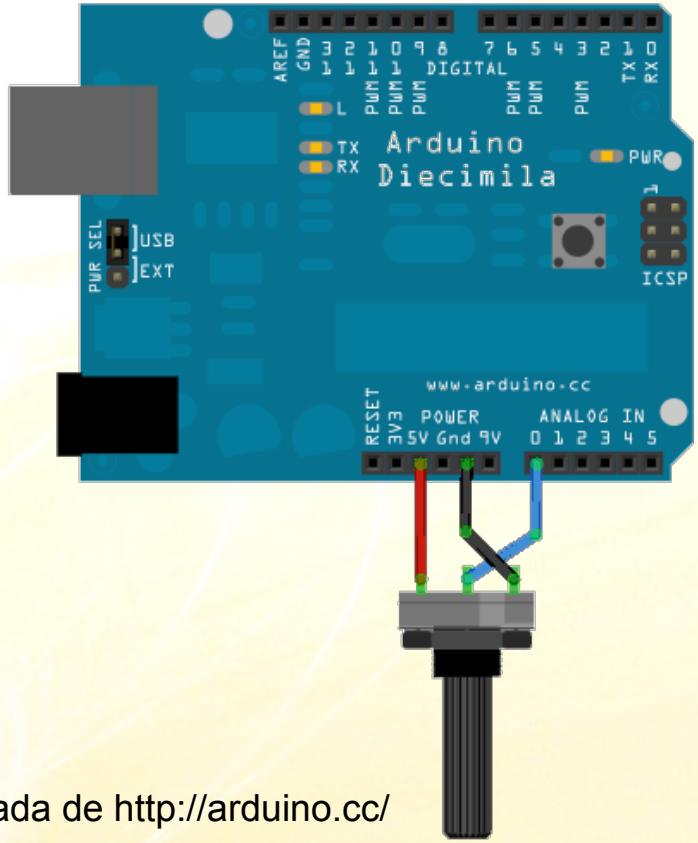
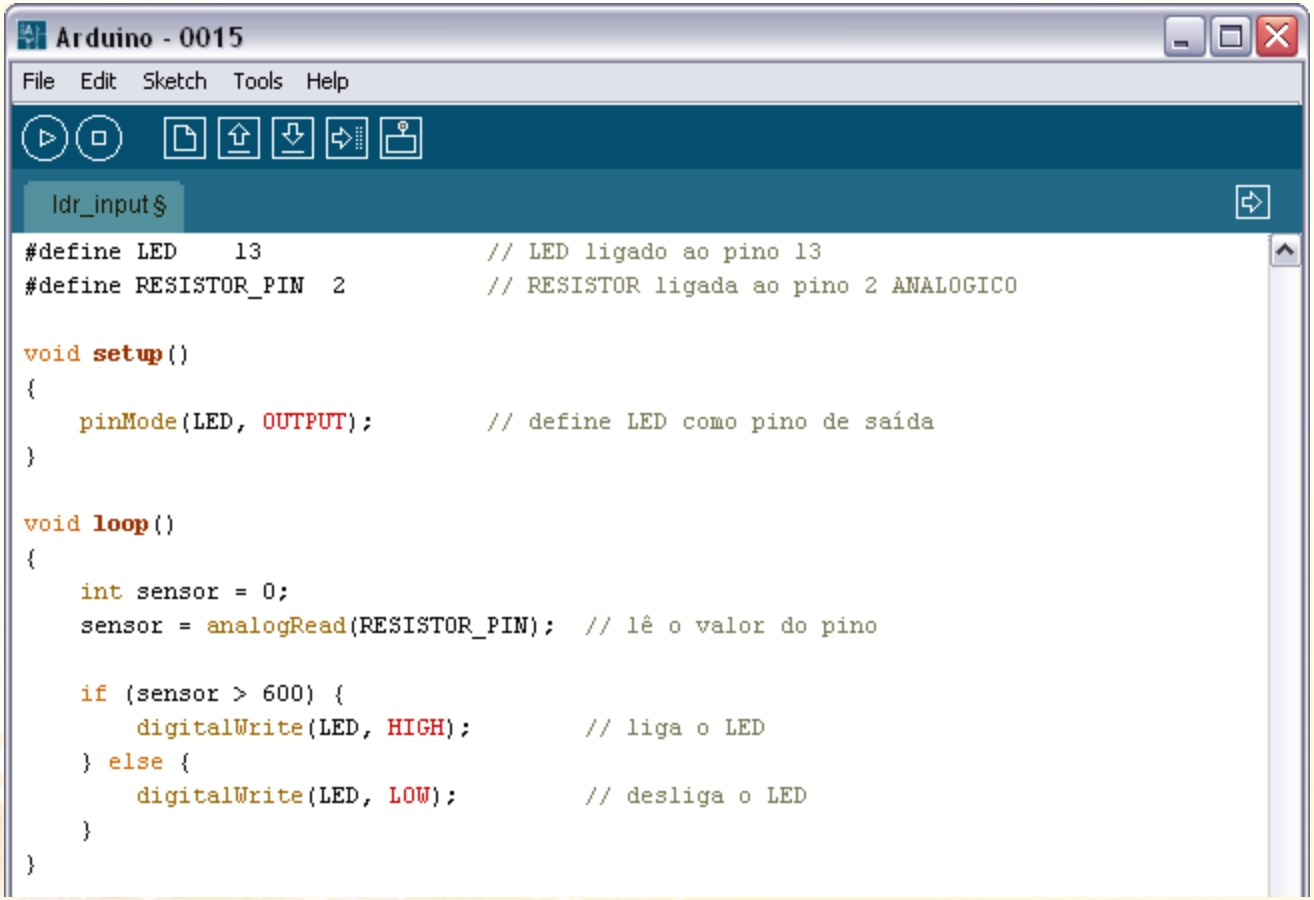


Figura retirada de http://arduino.cc/

entrada analógica - prática




```
#define LED     13                // LED ligado ao pino 13
#define RESISTOR_PIN  2           // RESISTOR ligada ao pino 2 ANALOGICO

void setup()
{
    pinMode(LED, OUTPUT);         // define LED como pino de saída
}

void loop()
{
    int sensor = 0;
    sensor = analogRead(RESISTOR_PIN);  // lê o valor do pino

    if (sensor > 600) {
        digitalWrite(LED, HIGH);        // liga o LED
    } else {
        digitalWrite(LED, LOW);         // desliga o LED
    }
}
```


Perguntas

protocolos de comunicação

comunicação serial - RS232

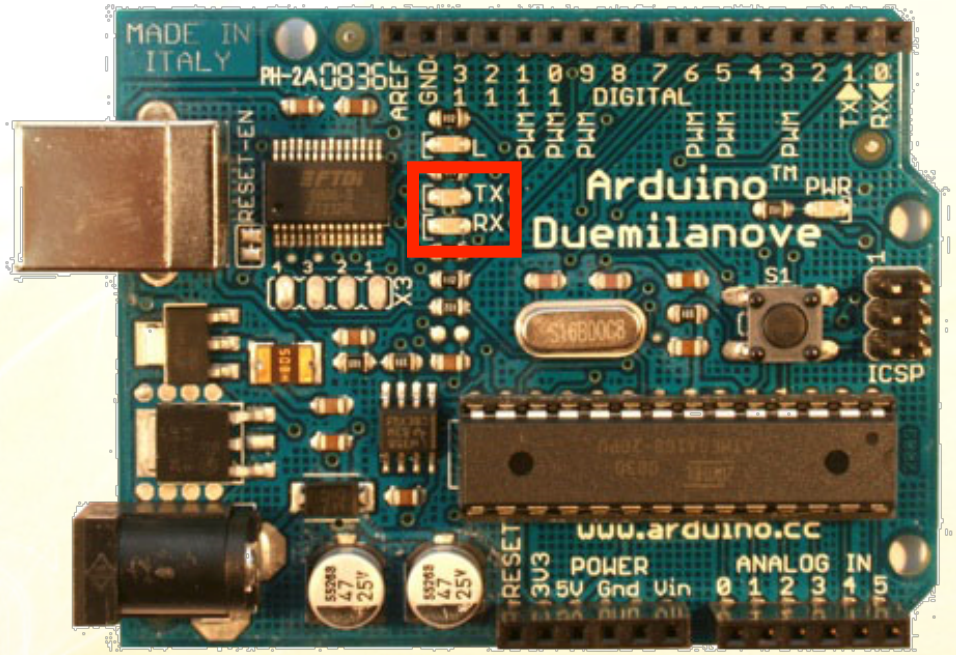
- chip ATMEGA 168 só tem interface serial, não tem USB
- nossa placa arduino possui um chip que converte Serial para USB
- usamos o mesmo cabo USB pra enviar dados pro PC via serial

comunicação serial - RS232

- o arduino possui uma biblioteca que implementa comunicação serial
- Serial.begin();
- Serial.print();
- Serial.read();

comunicação serial - RS232

- Leds
  - TX: dados enviados para o PC
  - RX: dados recebidos do PC

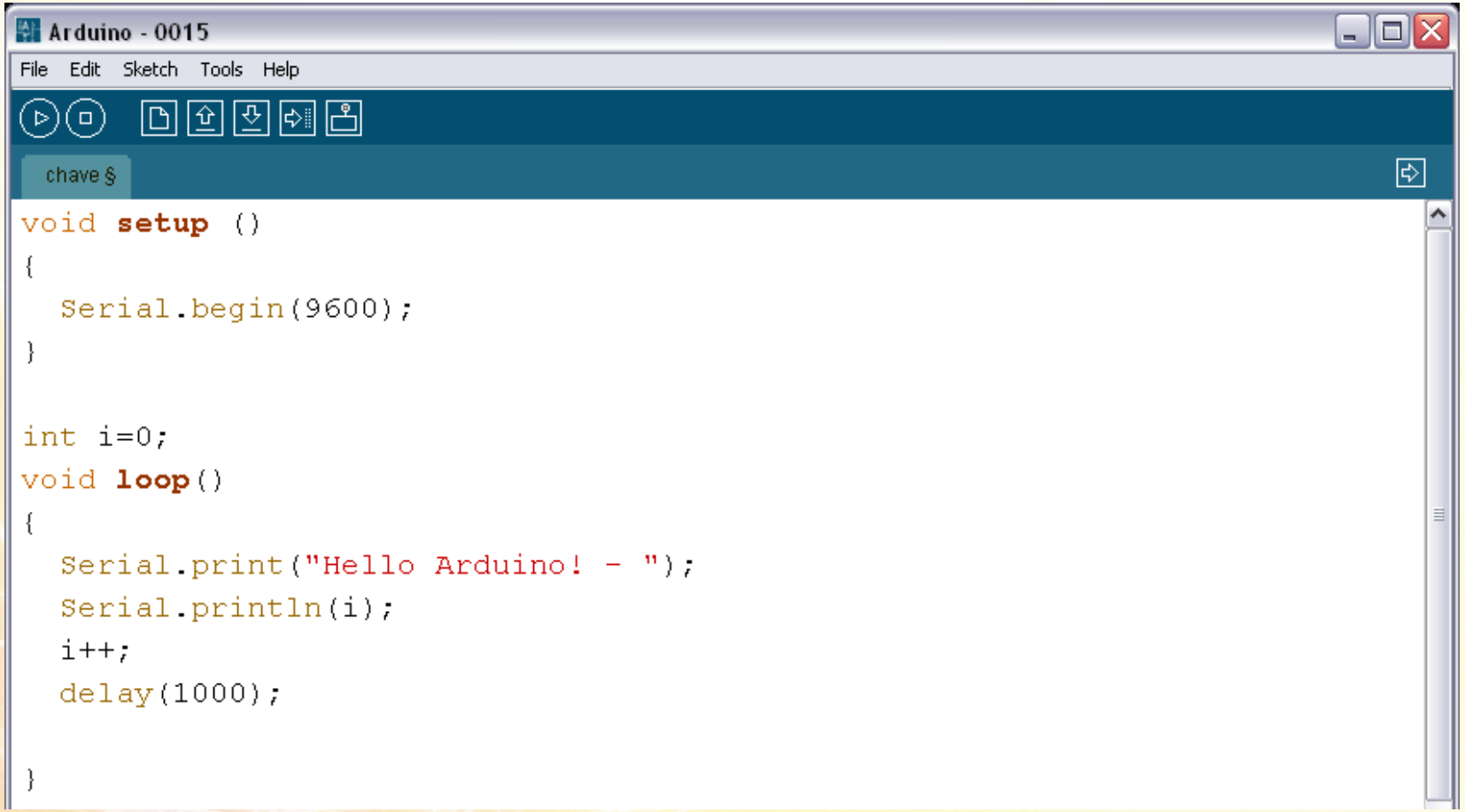


atividade prática!

comunicação serial - prática

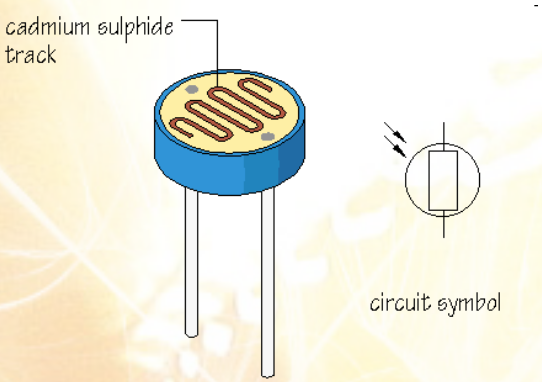
"Hello Arduino" via serial

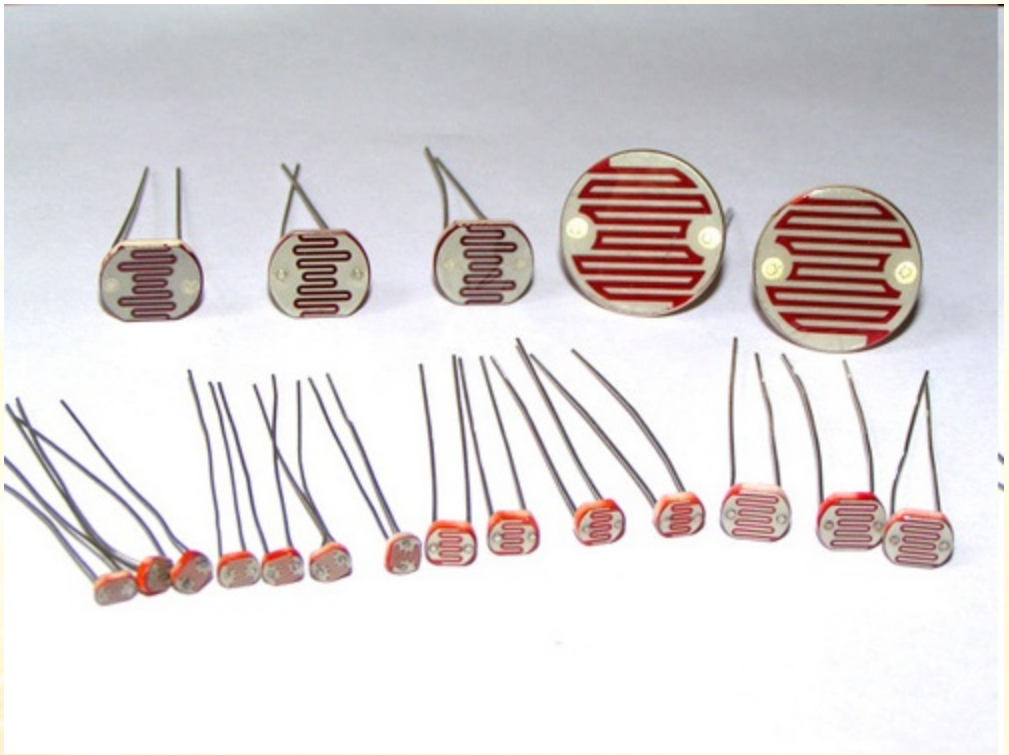
comunicação serial - prática



eletrônica - resistores LDR

- resistor variável sensível à luz



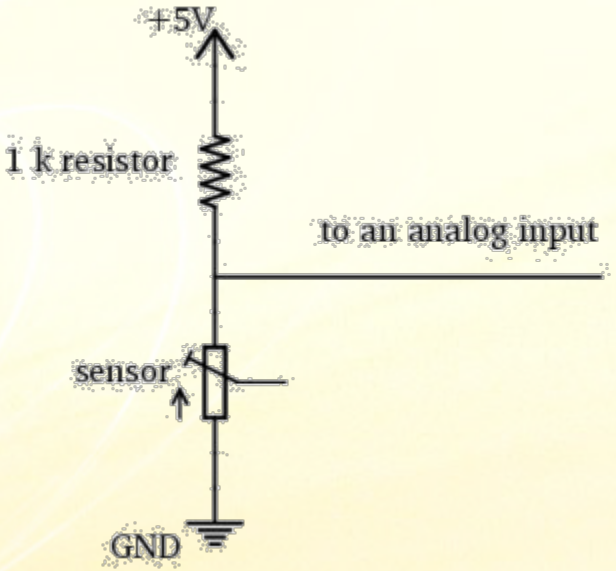


eletrônica - resistores LDR

- resistor variável sensível à luz

  circuito para arduino

  porque o resistor de 1k?
  - para limitar a corrente se o LDR assumir valores muito baixos



atividade prática!

comunicação serial - prática

ler valores do LDR e enviar via serial

perguntas?

interrupções do arduino mega

interrupções

- o arduino mega possui 6 pinos que podem ser ligados a interrupções de entrada e saída (IO).
- uma interrupção de IO é uma função que é executada quando existe uma mudança de estado no pino correspondente, independente do ponto em que se encontra a execução do programa.

interrupções

- a função `attachInterrupt();` permite configurar uma função para ser executada caso haja uma mudança no pino de IO correspondente:

  - `Interrupção 0 - pino 2`
  - `Interrupção 1 - pino 3`
  - `Interrupção 2 - pino 21`
  - `Interrupção 3 - pino 20`
  - `Interrupção 4 - pino 19`
  - `Interrupção 5 - pino 18`

interrupções

- exemplo:

```
attachInterrupt(num, function, mode);
```

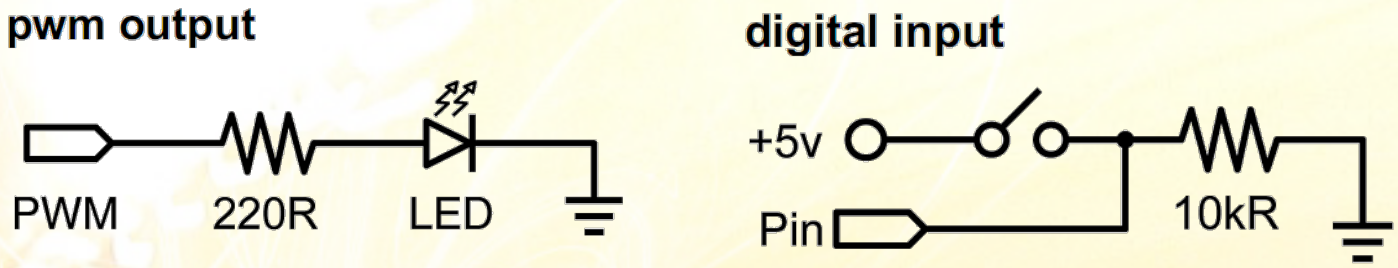
Mode: LOW, CHANGE, RISING, FALLING

```
int state = LOW;
void setup () {
    attachInterrupt(0, changeState, RISING);
}

void changeState() {
    state = !state;
}
```


atividade prática!

prática

- montar circuito de leitura de sinal digital usando interrupções.
- botão pressionado muda o estado do LED

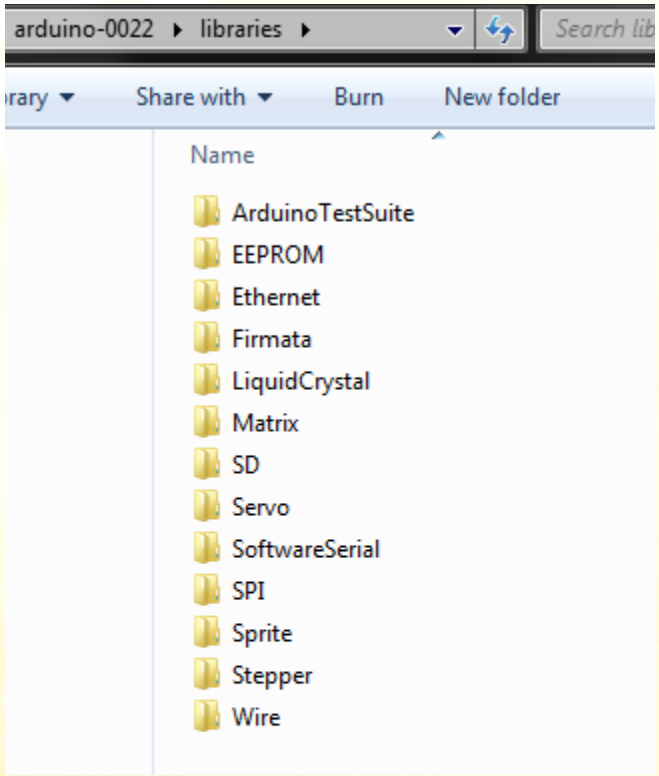


perguntas?

bibliotecas do arduino

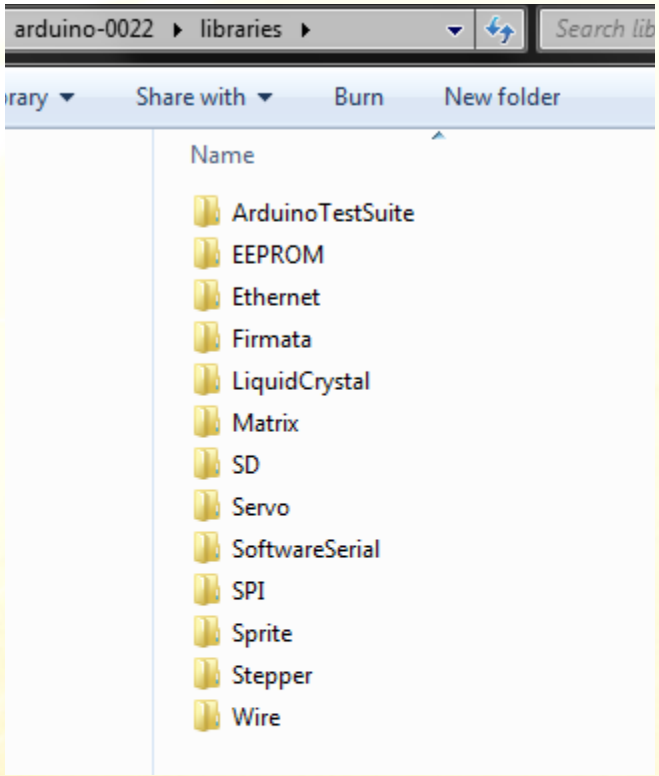
bibliotecas do arduino

- é possível estender a plataforma Arduino com adição de componentes de código, para controlar sensores e atuadores específicos.
- estes componentes são chamados de bibliotecas (libraries)



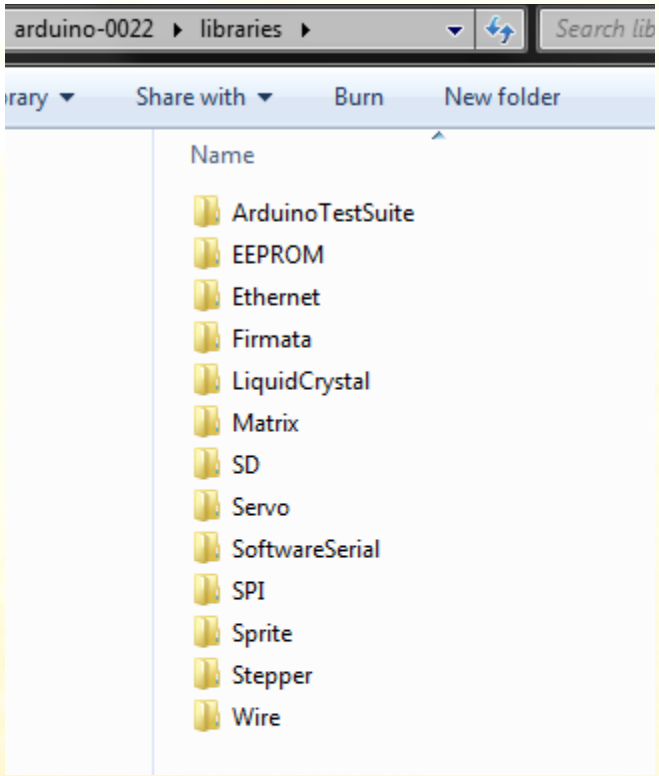
bibliotecas do arduino

- as bibliotecas são geralmente disponibilizadas como um zip que deve ser descompactado dentro da pasta libraries do Arduino.



bibliotecas do arduino

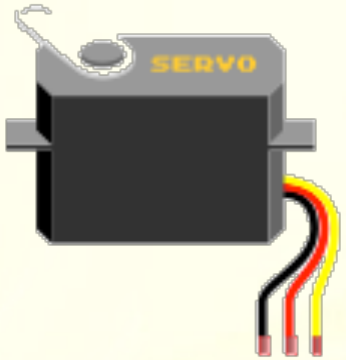
- após reiniciar o Arduino, a biblioteca estará disponível no menu `Sketch->Import Library`
- a maioria das bibliotecas para o Arduino pode ser encontrada em http://arduino.cc/en/Reference/Libraries



servo library

biblioteca servo

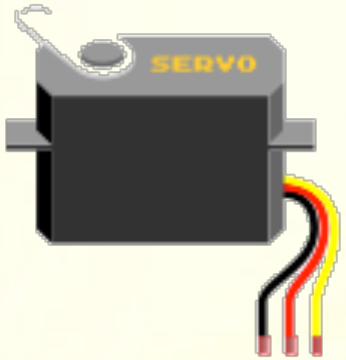
- servo motores são um tipo especial de motor que possui controle sobre a sua posição



- eles não são feitos para girar livremente, em vez disto, movimentam-se para a posição escolhida dentro de um limite, que é geralmente 180 graus.

biblioteca servo

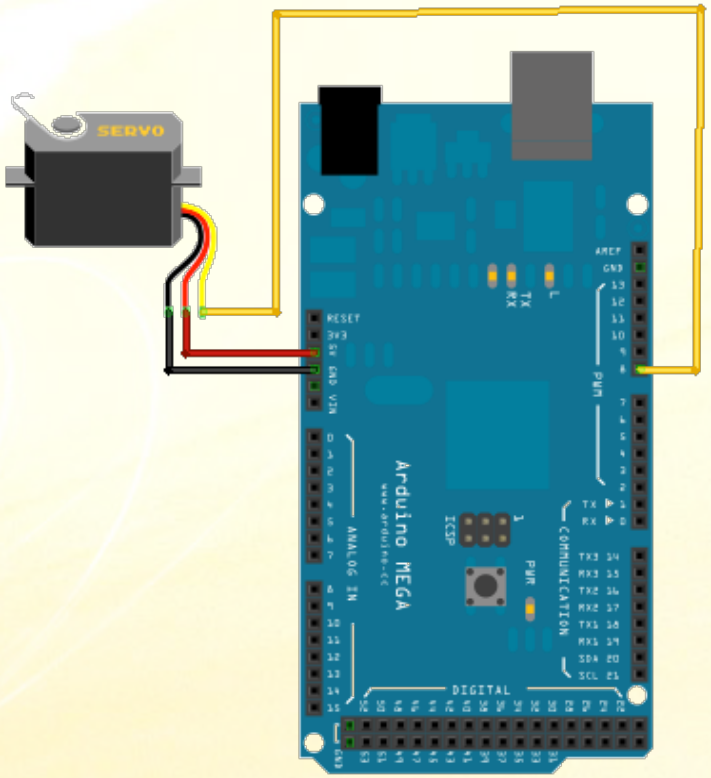
- um servo motor possui 3 fios
  dois para alimentação e
  um terceiro para controlar sua posição.



- este controle é feito através da modulação por largura de pulso (PWM).

biblioteca servo

- A conexão com os servos é feita da seguinte forma:
  - Fio preto ou marrom: GND
  - Fio vermelho: 5V
  - Fio amarelo ou laranja: pino de controle (saída digital)



biblioteca servo

- usando a biblioteca servo:
  - para usar esta biblioteca, é necessário importá-la para o nosso programa, através do menu:
    `Sketch->Import Library->Servo`
  - aparecerá a seguinte linha no seu programa indicando que a biblioteca foi importada
    `#include <Servo.h>`
  - depois, é só criar uma variável do tipo Servo:
    `Servo motor1;`

biblioteca servo

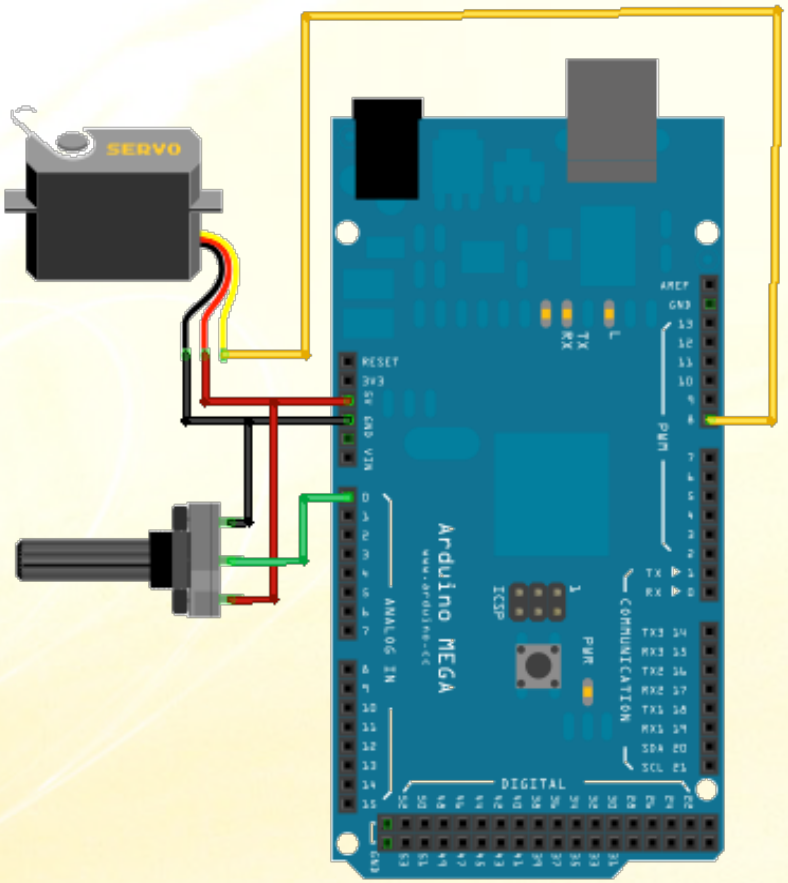
- usando a biblioteca servo:

```
motor1.attach(pino);   // associa a variável ao
                       // pino em que o servo
                       // motor está ligado

motor1.write(angulo); // angulo para o qual o
                       // servo deverá girar, entre
                       // 0 e 180 graus

ang = motor1.read();   // retorna o angulo em que o
                       // servo se encontra
```


atividade prática!

biblioteca servo - prática

- Fazer um programa que movimente o servo de acordo com o valor lido do potenciômetro.



perguntas?

nunchuck library

biblioteca nunchuck

- é possível utilizar o nunchuck do WII como dispositivo de entrada.
- o nunchuck é um dispositivo que utiliza a interface I^2C para comunicação. Esta interface possui 2 fios de comunicação: SDA(dados) e SCL(clock).
- o arduino possui uma biblioteca chamada Wire que implementa o I^2C.
- a biblioteca Nunchuck foi desenvolvida utilizando a biblioteca Wire, por isso as duas precisam ser importadas no sketch.

biblioteca nunchuck

- o nunchuck possui um acelerômetro de 3 eixos, um joystick analógico de 2 eixos e 2 botões.

- cada um destes sensores pode ser lido através da biblioteca Nunchuck.

biblioteca nunchuck

- conexões com o Arduino Mega:
  - Fio Vermelho: 3.3V
  - Fio Preto: GND
  - Fio Amarelo(SDA): Pino 20
  - Fio Azul(SCL): Pino 21
- Objeto para acessar os dados deve ser criado

```
Nunchuck nunchuck;
```

biblioteca nunchuck

- usando a biblioteca nunchuck:

```
nunchuck.begin();     // inicializa a biblioteca,
                      // deve ser chamada no
                      // setup();

nunchuck.update();    // lê novos valores dos
                      // sensores do nunchuck, deve
                      // ser chamada continuamente
                      // dentro do loop();

nunchuck.calibrateJoy(); // define os valores
                         // atuais do joystick
                         // como os valores
                         // centrais
```


biblioteca nunchuck

- usando a biblioteca nunchuck:

```
x = nunchuck.readAccelX(); // lê o valor do
                           // acelerômetro no eixo
                           // X
```

  - Mesmo para `readAccelY();` e `readAccelZ();`

```
x = nunchuck.readAngleX(); // lê o valor do angulo
                           // (entre 0 e 180) no
                           // eixo X
```

  - Mesmo para `readAngleY();` e `readAngleZ();`

biblioteca nunchuck

- usando a biblioteca nunchuck:

```
z = nunchuck.readZ(); // lê o valor atual do botão
                      // Z (0 liberado, 1
                      // pressionado)
```

  - Mesmo para readC();

```
z = nunchuck.zPressed(); // retorna se o botão Z
                         // foi pressionado
```

  - Mesmo para cPressed();

biblioteca nunchuck

- usando a biblioteca nunchuck:

```
x = nunchuck.readJoyX(); // lê o valor atual do
                         // joystick no eixo X
                         // (0 - 255)
```

  - Mesmo para readJoyY();

```
x = nunchuck.leftJoy();  // retorna se o joystick
                         // foi para a esquerda
                         // (informação digital)
```

  - Mesmo para rightJoy(); upJoy(); e downJoy();

atividade prática!

biblioteca nunchuck - prática

- Ler os valores do nunchuck e enviá-los pela serial para o PC

: x, y, z – btnC, btnZ – joyX, joyY

atividade prática!

biblioteca nunchuck - prática

- controlar servo motores através do acelerômetro e/ou do joystick
- utilizar o joystick, os servo motores e o material de apoio (papel, fita, etc) para criar um robô

perguntas?

capsense library

biblioteca capacitive sense

- esta biblioteca permite receber dados de sensores capacitivos utilizando 2 pinos do Arduino.

- um dos pinos do arduino (send pin) envia um sinal para o outro pino (receive pin), através de um resistor. O atraso entre enviar e receber o sinal depende dos valores de R*C, onde C é a capacitância que é alterada de acordo com a proximidade do corpo humano.

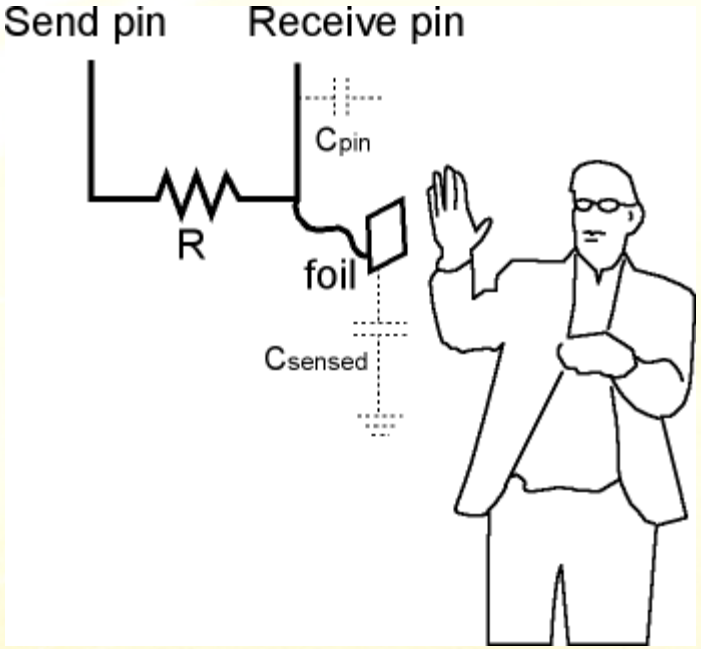
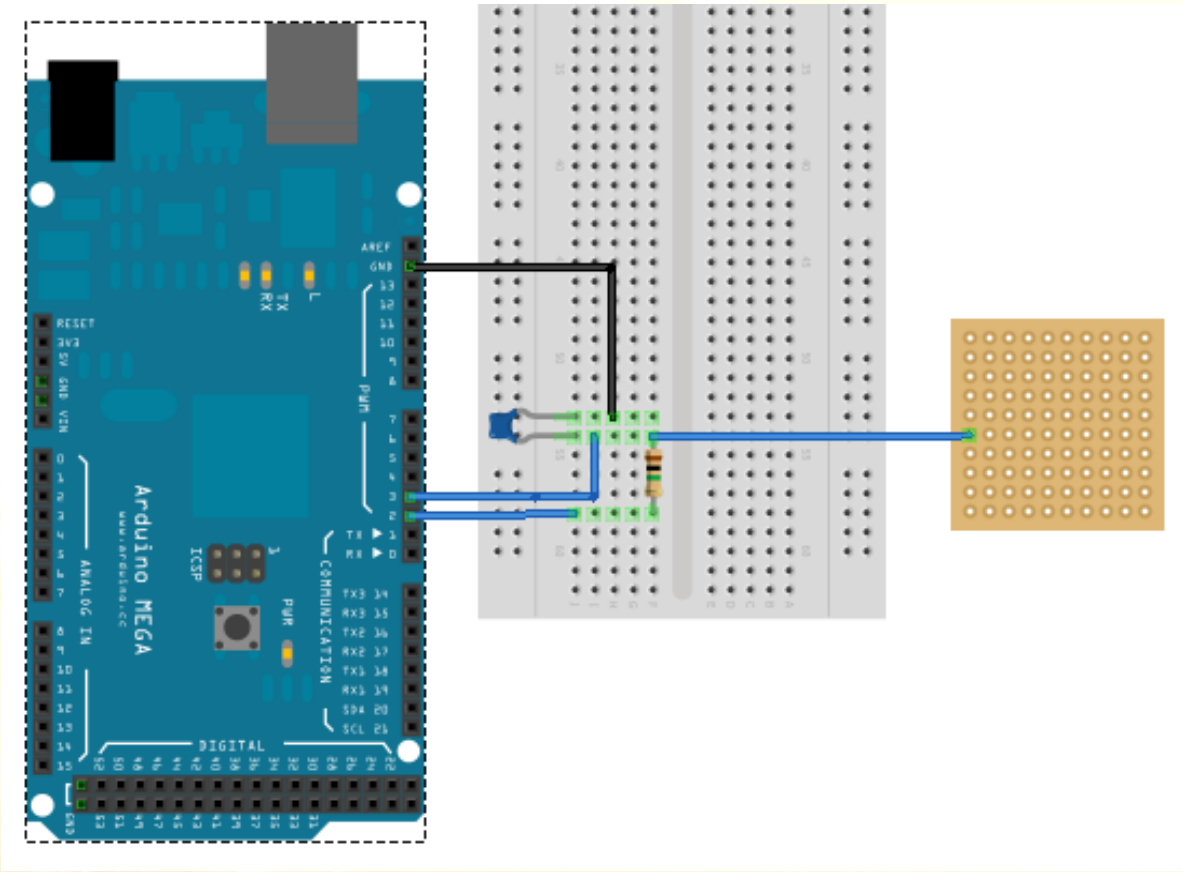


Figura retirada de
http://www.arduino.cc/playground/Main/CapSense

biblioteca capacitive sense

- circuito



biblioteca capacitive sense

- usando a biblioteca cap sense:

```
CapSense btn1 = CapSense(sendPin, receivePin);
// cria um sensor capacitivo entre os pinos
// sendPin e receivePin

btn1.capSense(numSamples);
// retorna a capacitância média, de acordo com a
// quantidade de amostras.
// esta capacitância possui um valor baixo se
// não houver toque e um valor alto ao toque.
```


biblioteca capacitive sense

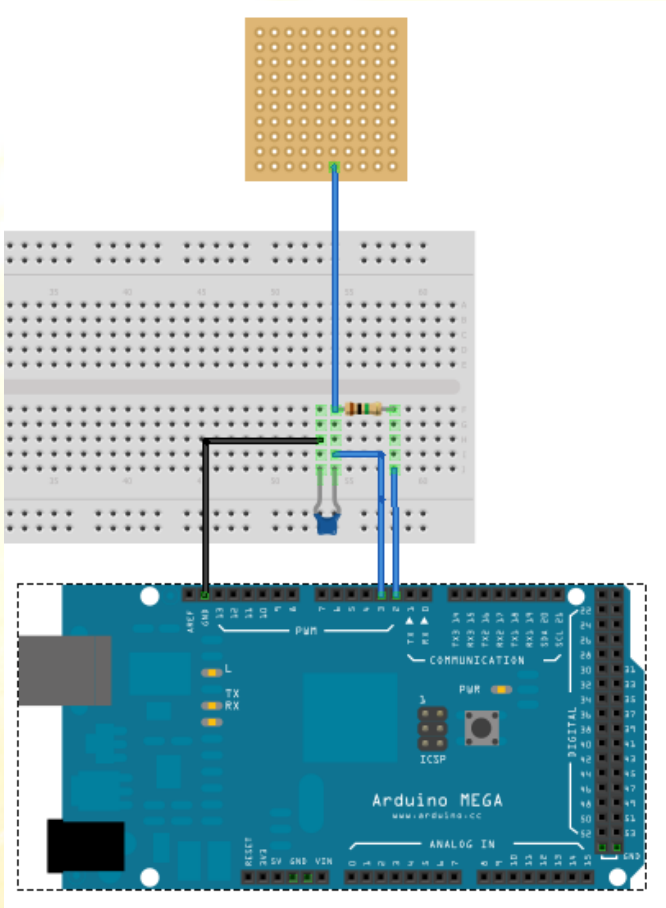
- usando a biblioteca cap sense:

```
set_CS_Timeout_Millis(timeout_millis);
// define o valor de timeout para o sensor
// caso o mesmo não consiga ler o valor correto

set_CS_AutocaL_Millis(autoCal_millis);
// define o tempo de auto calibragem do sensor
```


atividade prática!

biblioteca capacitive sense - prática

- acender leds ou movimentar motores de acordo com o sensor capacitivo

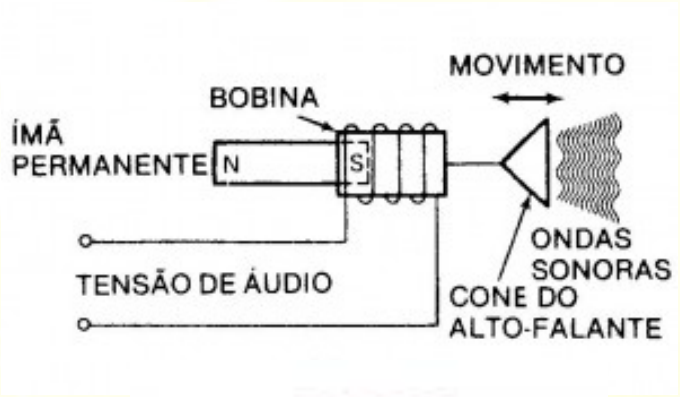


perguntas?

atuadores sonoros

atuadores sonoros

- auto falantes
  - bobina em volta de um imã
  - corrente elétrica na bobina produz campo magnético
  - campo magnético variável faz a membrana se deslocar, produzindo som

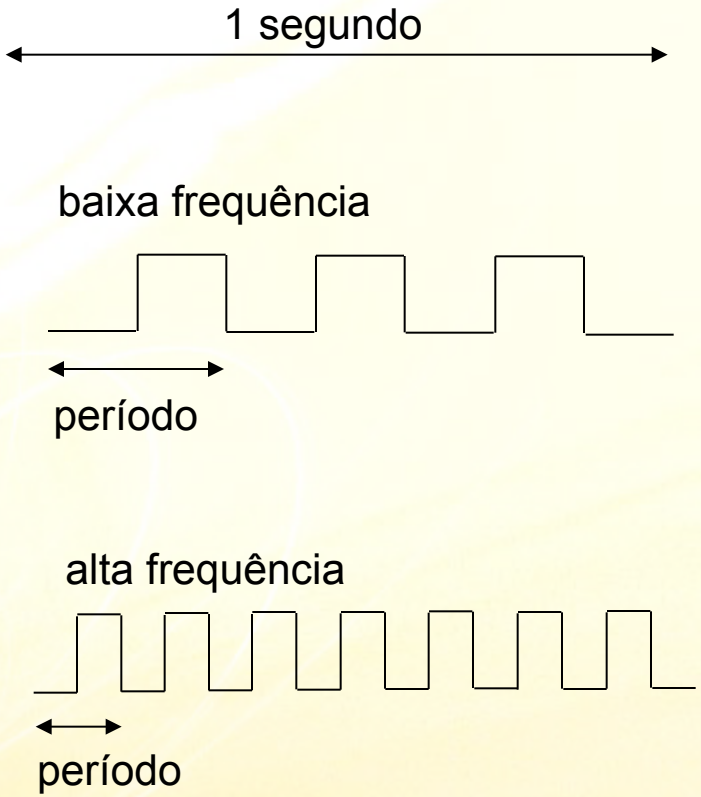


atuadores sonoros

- como programar o arduino para tocar uma nota musical?
  - uma nota musical é um som em uma determinada frequência
  - a frequência de uma nota significa quantas vezes o atuador sonoro vibra em 1 segundo

atuadores sonoros

- para fazer o atuador vibrar, escrevemos no pino uma sequência de valores HIGH e LOW, tantas vezes por segundo quanto for a frequência da nota
- o tempo de cada variação HIGH e LOW é chamada de período e é o inverso da frequência



atuadores sonoros

- programar o arduino para tocar uma nota musical

```
void playTone(int period, int duration)
{
    for (long i = 0; i < duration * 1000L; i += period* 2)
    {
        digitalWrite(speakerPin, HIGH);
        delayMicroseconds(period);
        digitalWrite(speakerPin, LOW);
        delayMicroseconds(period);
    }
}
```


atuadores sonoros

- como tocar uma nota musical?

```
timeHigh = periodo / 2 = 1 / (2 * frequência)

* nota          frequência       periodo  tempo em nivel alto
* c (dó)        261 Hz           3830     1915
* d (ré)        294 Hz           3400     1700
* e (mi)        329 Hz           3038     1519
* f (fá)        349 Hz           2864     1432
* g (sol)        392 Hz          2550     1275
* a (lá)        440 Hz           2272     1136
* b (si)        493 Hz           2028     1014
* C (dó)        523 Hz           1912     956
```

Não é necessário escrever essas frequências, podemos incluir o arquivo **notes.h**

atuadores sonoros

- como tocar uma nota musical?

notes.h

```
#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
...
```


função tone

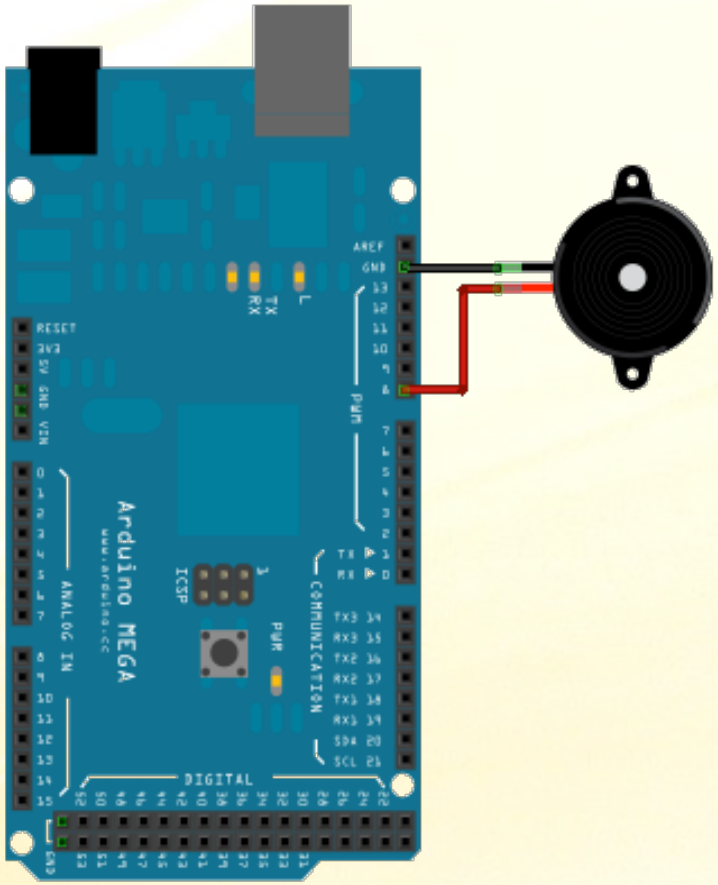
- Arduino já possui uma função para tocar notas

```
tone(pin, frequency);
// emite uma determinada nota (representada pela
// frequência) no pino ocrrespondente

noTone(pin);
// para de emitir a frequência definida por tone()
// no pino correspondente

tone(pin, frequency, duration);
// emite uma determinada nota (representada pela
// frequência) no pino ocrrespondente durante uma
// determinada duração
```


atividade prática!

atuadores sonoros - prática

- Tocar uma melodia (“dó ré mi fá”, por exemplo) usando a função tone();
- Usar alguns switches ou sensores capacitivos para tocar uma nota quando pressionados, criando um teclado.



perguntas?

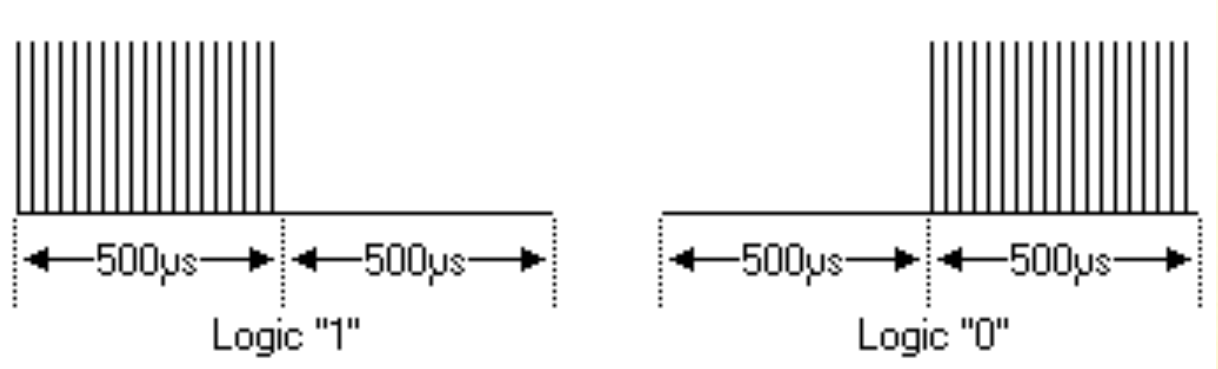
IR Remote library

biblioteca IR Remote

- esta biblioteca permite enviar e receber dados através de atuadores e sensores infravermelhos.
- a luz infravermelha não é visível ao olho humano, entretanto está presente em vários dispositivos emissores de luz, como lâmpadas incandescentes e o próprio sol.
- assim, a comunicação infravermelho sofre interferência destes fatores.

biblioteca IR Remote

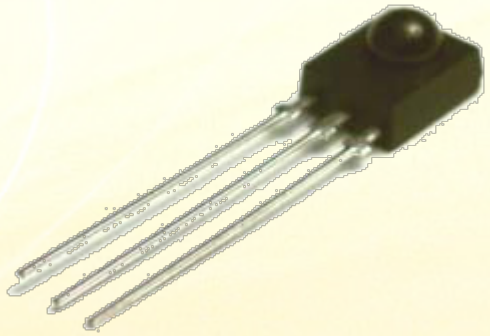
- para reduzir a interferência de outras fontes de luz infravermelha, os dispositivos utilizam a modulação de sinal.



biblioteca IR Remote

- um receptor sintonizado de infravermelho é um dispositivo capaz de detectar pulsos de infravermelho a frequências de 36kHz a 40kHz, sendo 38kHz a mais comumente utilizada.

- ao detectar esta frequência o receptor põe o nivel 0 em sua saída, caso a frequência não seja detectada, a saída tem nível lógico 1.



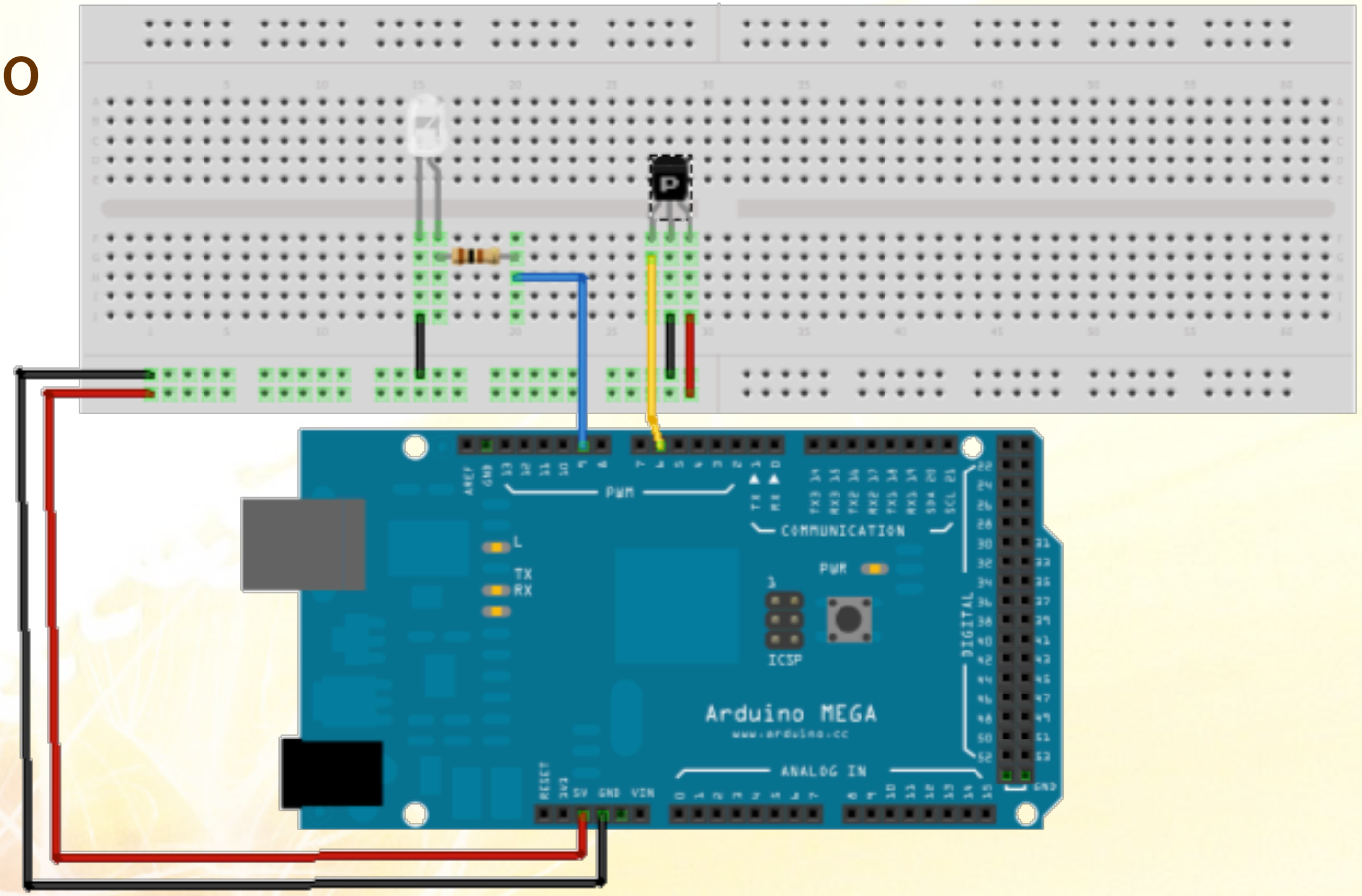
biblioteca IR Remote

- o transmissor utilizado para comunicação infravermelho é um LED, parecido com um LED comum mas que emite luz infravermelha.

- não é possível verificar se o LED infravermelho está aceso a olho nu, entretanto olhá-lo através de uma câmera (de celular, pode exemplo) vai mostrar se ele está aceso ou não.

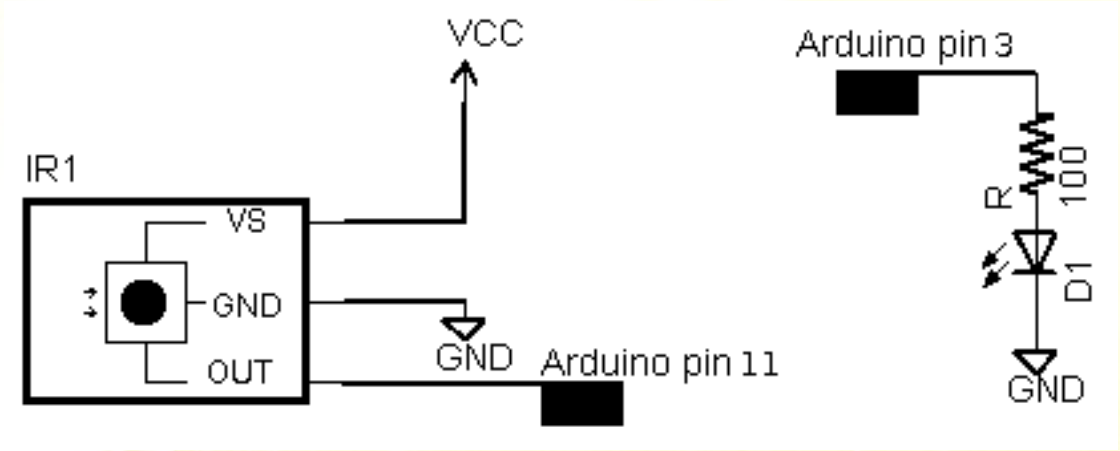
biblioteca IR Remote

- circuito



biblioteca IR Remote

- circuito



*No Arduino Mega, a biblioteca IRRemote envia dados através do **pino 9**
Também é necessário setar o `pinMode(9, OUTPUT);`

biblioteca IR Remote

- recepção de infravermelho

```
IRrecv irrecv(pin);
// cria um objeto para recepção de infravermelho
// (receiver), com o sensor conectado ao pin.

decode_results results;
// cria um objeto para armazenar o resultado
// da decodificação do sinal recebido

irrecv.enableIRIn(); // Inicia o receiver
```


biblioteca IR Remote

- recepção de infravermelho

```
X = irrecv.decode(&results);
// decodifica o sinal caso haja algum dado
// recebido. Neste caso, retorna valor diferente
// de zero.

Y = results.value;
// results.value contém o valor do comando
// recebido

irrecv.resume();
// prepara o objeto para receber o próximo valor
```


biblioteca IR Remote

- recepção de infravermelho - exemplo de código

```
int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // inicia o receiver
}

void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value, HEX);
    irrecv.resume(); // Recebe o próximo valor
  }
}
```


biblioteca IR Remote

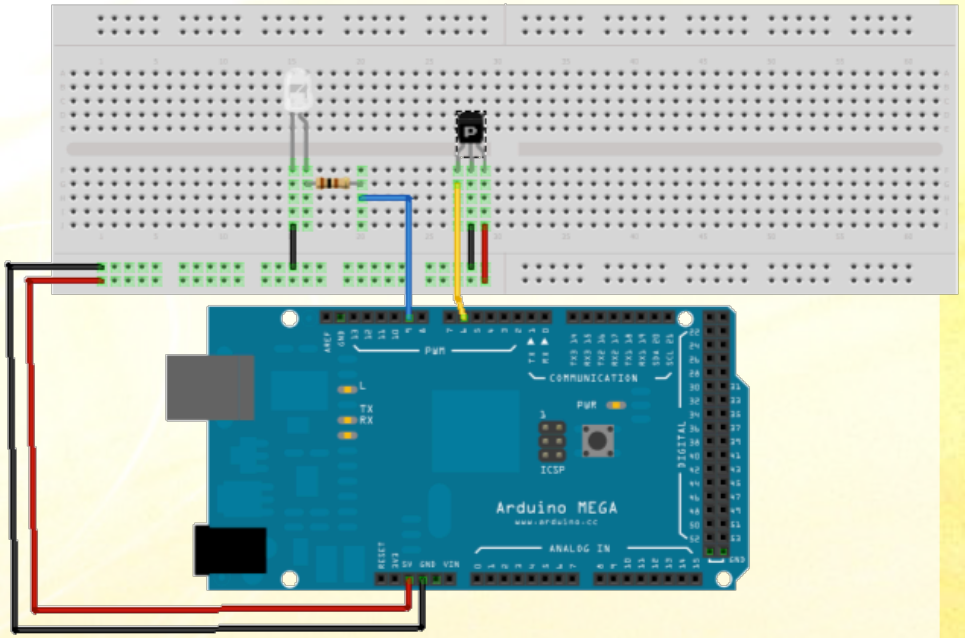
- transmissão de infravermelho

```
IRsend irsend;
// cria um objeto para transmissão de
// infravermelho. No arduino mega, este objeto
// transmite no pino 9.

irsend.sendSony(dado, nBits);
irsend.sendNEC(dado, nBits);
irsend.sendRC5(dado, nBits);
irsend.sendRC6(dado, nBits);
// envia o dado com o determinado numero de bits
// utilizando o protocolo especificado
```


atividade prática!

biblioteca IR Remote - prática

- ler os valores recebidos pelo controle remoto e mostrá-los na serial
- comandar leds e motores através do controle remoto

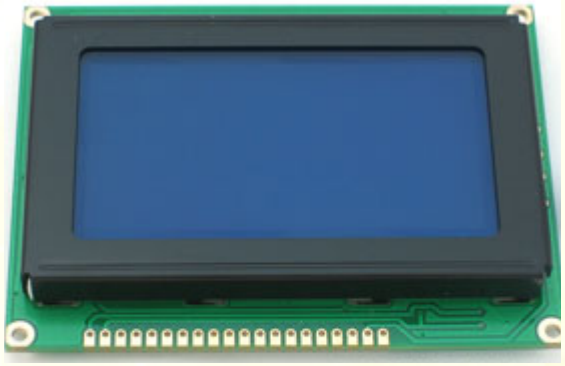


perguntas?

GLCD library

biblioteca GLCD

- a biblioteca GLCD permite controlar displays Gráficos LCD. Foi feita inicialmente para o display gráfico com controlador KS0108, mas atualmente suporta outros controladores.



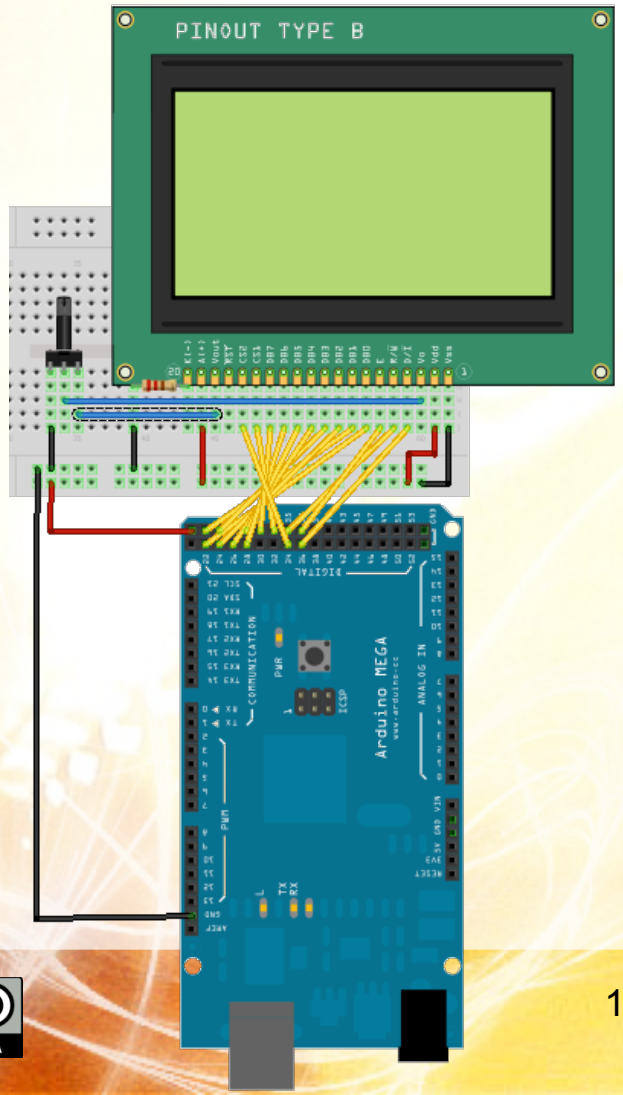
biblioteca GLCD

- o display gráfico que estamos utilizando possui o controlador KS0108B. Este display possui 20 pinos.

- a conexão deste display com o arduino mega se dá de acordo com a tabela ao lado.

Pino LCD	Arduino	Pot
1	GND	1
2	5V	---
3	---	2
4	36	---
5	35	---
6	37	---
7	22	---
8	23	---
9	24	---
10	25	---
11	26	---
12	27	---
13	28	---
14	29	---
15	33	---
16	34	---
17	---	---
18	---	3
19	5V	---
20	---	220R

biblioteca GLCD - circuito



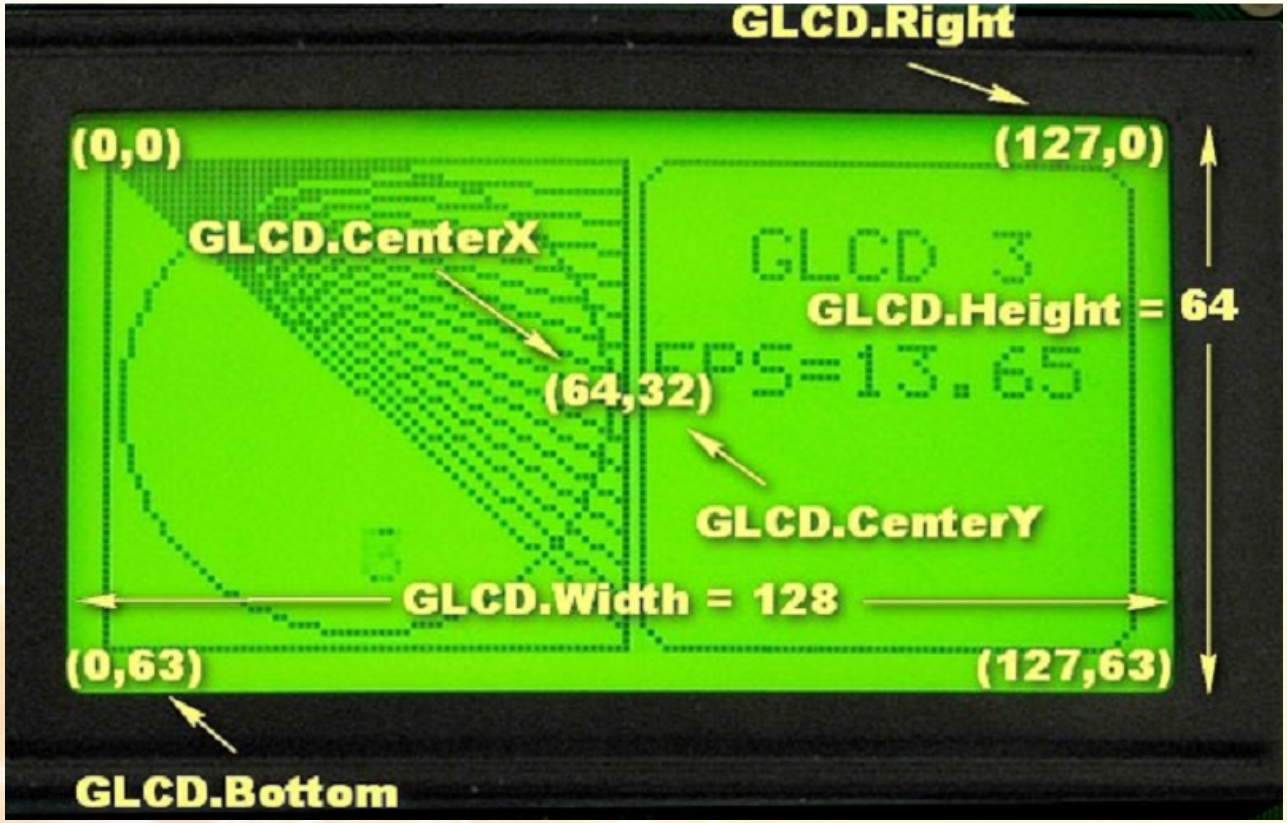
Pino LCD	Arduino	Pot
1	GND	1
2	5V	---
3	---	2
4	36	---
5	35	---
6	37	---
7	22	---
8	23	---
9	24	---
10	25	---
11	26	---
12	27	---
13	28	---
14	29	---
15	33	---
16	34	---
17	---	---
18	---	3
19	5V	---
20	---	220R

biblioteca GLCD

```
GLCD.Init(mode);
// inicializa a biblioteca GLCD de acordo com o
// modo que pode ser: INVERTED ou NON-INVERTED

GLCD.SetDisplayMode(mode);
// define o estado do display de acordo com o
// modo que pode ser: INVERTED ou NON_INVERTED

GLCD.ClearScreen(color);
// apaga a tela inteira com a cor desejada, que
// pode ser: WHITE ou BLACK
```


biblioteca GLCD

Sistema de coordenadas



biblioteca GLCD

```
GLCD.SetDot(x, y, color);
// desenha um ponto nas coordenadas x,y com a cor
// desejada (BLACK ou WHITE)

GLCD.DrawVLine(x, y, height, color);
// desenha uma linha vertical, começando em x,y
// com altura height e com cor definida em color

GLCD.DrawHLine(x, y, width, color);
// desenha uma linha horizontal, começando em x,y
// com altura width e com cor definida em color

GLCD.DrawLine(x1, y1, x2, y2, color);
// desenha uma linha começando em x1,y1
// e terminando em x2,y2 com a cor desejada
```


atividade prática!

biblioteca GLCD - prática

- montar o circuito para ligar o display LCD e desenhar formas básicas

Pino LCD	Arduino	Pot
1	GND	1
2	5V	---
3	---	2
4	36	---
5	35	---
6	37	---
7	22	---
8	23	---
9	24	---
10	25	---
11	26	---
12	27	---
13	28	---
14	29	---
15	33	---
16	34	---
17	---	---
18	---	3
19	5V	---
20	---	220R

perguntas?

biblioteca GLCD

```
GLCD.DrawRect(x, y, width, height, color);
// desenha um retangulo com canto superior
// esquerdo em x,y e com largura e altura
// definidas por width e height

GLCD.FillRect(x, y, width, height, color);
// desenha um retangulo preenchido, da mesma forma
// que a função anterior

GLCD.InvertRect(x, y, width, height);
// inverte as cores dos pontos (pixels) de uma
// área começando em x,y e com área width,height
```

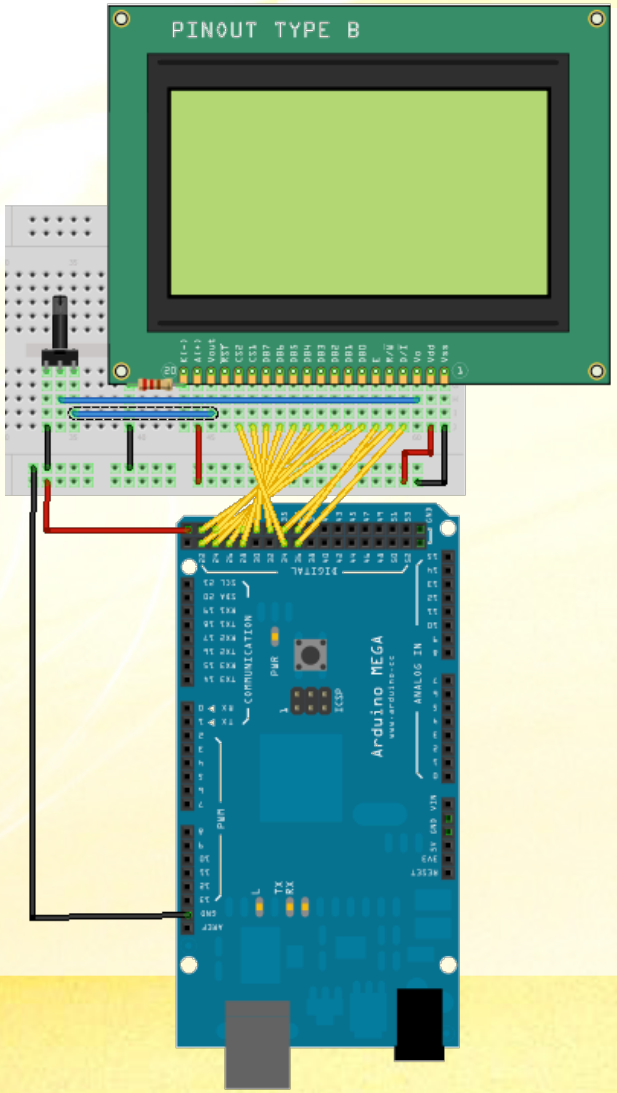

biblioteca GLCD

```
GLCD.DrawRoundRect(x, y, w, h, radius, color);
// desenha um retangulo com cantos arredondados
// definidos por radius, que pode ter valor entre
// 1 e metadade da altura ou largura do retangulo

GLCD.DrawCircle(x, y, r, color);
// desenha um circulo centralizado em x,y e com
// raio r

GLCD.FillCircle(x, y, r, color);
// desenha um circulo preenchido, centralizado em
// x,y e com raio r
```


atividade prática!

biblioteca GLCD - prática

- definir e implementar um jogo usando o LCD e os sensores e atuadores aprendidos
- após a definição do jogo, o mesmo deverá ser apresentado para a turma



perguntas?

biblioteca GLCD - bitmaps (figuras)

- a biblioteca GLCD possui um utilitário para importar figuras no ambiente do arduino
- este utilitário encontra-se em arduino-0022\libraries\glcd\bitmaps\utils\Java\glcdMakeBitmap.jar
- basta clicar duas vezes no arquivo jar e este será executado, abrindo uma janela para conversão. Arrastar um bitmap sobre esta janela irá converter o bitmap em código que pode ser carregado no Arduino. Este código é gerado no arquivo **`nome_da_imagem.h`**

biblioteca GLCD - bitmaps

- Para usar uma imagem, basta incluí-la no seu sketch
  `#include "bitmaps\nome_da_imagem.h"`

- Para incluir todos os bitmaps, podemos usar
  `#include "bitmaps\allBitmaps.h"`

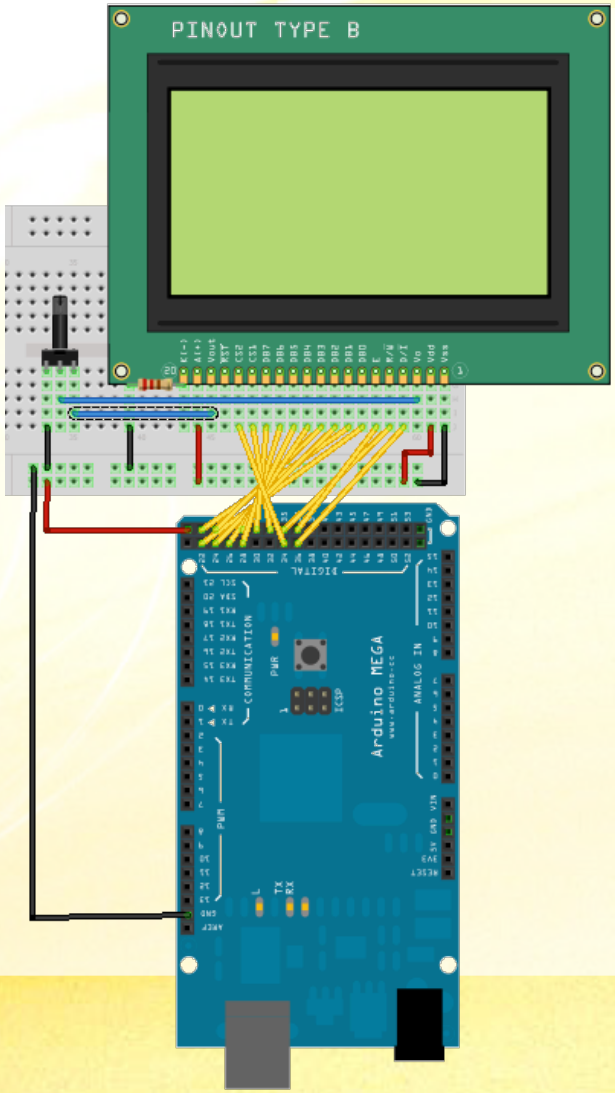
- Função para desenhar o bitmap

```
GLCD.DrawBitmap(bitmap, x, y);
// desenha o bitmap especificado na posição x,y
```


atividade prática!

biblioteca GLCD - prática

- incrementar o jogo com imagens bitmap



perguntas?

biblioteca GLCD - fontes

- Existe um utilitário grátis que permite converter fontes do windows em arquivos que podem ser importados no seu sketch, chamado FontCreator2

- Para usar uma fonte, basta incluí-la no seu sketch

```
#include "fonts\nome_da_fonte.h"
```

- Para incluir todas as fontes, podemos usar

```
#include "fonts\allFonts.h"
```

- Função para selecionar a fonte

```
GLCD.SelectFont(nome_da_fonte);
// seleciona a fonte desejada para ser utilizada
// nas próximas operações de escrita
```


biblioteca GLCD - escrita de texto

```
GLCD.SetFontColor(color);
// define a cor a ser usada para escrever com a
// fonte atual

GLCD.SetTextMode(mode);
// define o modo de rolagem de texto, que pode ser
// SCROLL_UP ou SCROLL_DOWN

gText myTxtArea = gText(x1, y1, x2, y2);
// cria uma área de texto de tamanho começando em
// x1,y1 e terminando em x2,y2
```

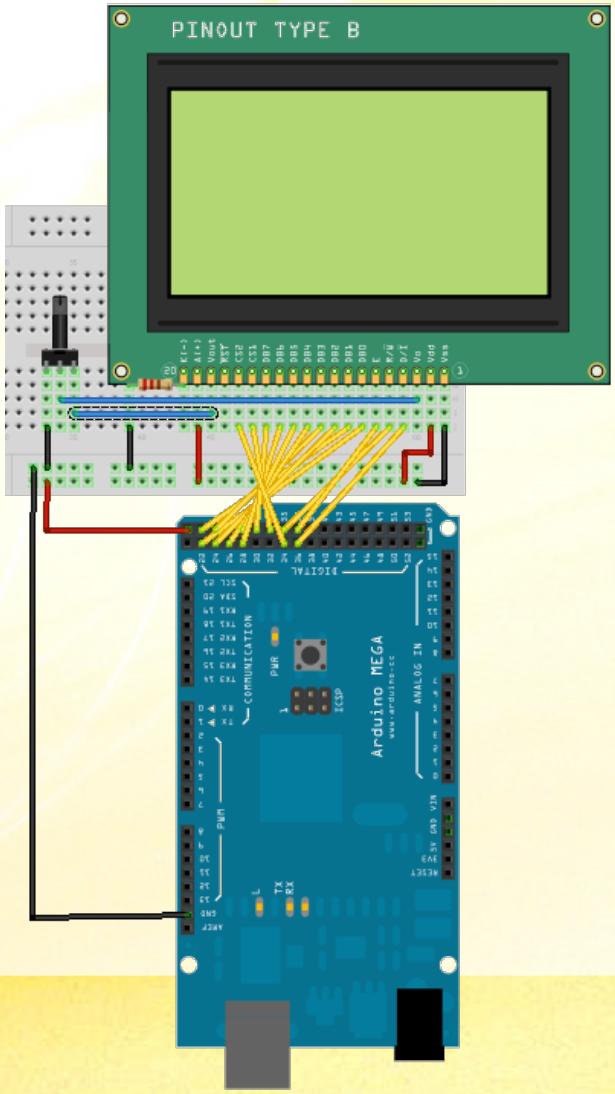

biblioteca GLCD - escrita de texto

```
gText myTxtArea = gText(predefArea);
// cria uma área predefinida, que pode ser:
// textAreaFULL, textAreaTOP, textAreaBOTTOM,
// textAreaLEFT ou textAreaRIGHT

myTxtArea.ClearArea();
// apaga a área de texto

myTxtArea.print(argumento);
// imprime o argumento na área de texto. Este
// argumento pode ser uma variável ou uma string
```


atividade prática!

biblioteca GLCD - prática

- finalizar o jogo com os conceitos aprendidos



perguntas?

arduino - referencias

- Lista dos comandos da linguagem em:

http://arduino.cc/en/Reference/HomePage

- Lista dos tutoriais em:

http://www.arduino.cc/en/Tutorial/HomePage

Obrigado!

Tiago Barros | tiago@tiagobarros.org