

**DESENVOLVIMENTO PARA ANDROID COM
ARDUINO**

INTRODUÇÃO

INTRODUÇÃO

EMENTA

Introdução à plataforma de desenvolvimento Arduino. Linguagem de programação Arduino. Introdução a eletrônica digital, analógica e modulação por pulso (PWM). Construção de circuitos eletrônicos e prototipação. Sensores e atuadores. Comunicação Serial/USB. Armazenamento na EEPROM. Controle de relógio. Comunicação sem fio. Interação com a plataforma android. Integração via Internet.

INTRODUÇÃO

BIBLIOGRAFIA

Google android: aprenda a criar aplicações para dispositivos móveis com Android SDK. 3.ed. São Paulo: Novatec, 2013. ISBN 978-85-7522-344-4.

Elementos de eletronica digital. 40.ed. São Paulo: Érica, 2007. 524 p. ISBN 978-85-7194-019-2.

Arduino Básico. 2^a Edição. Michael McRoberts. Novatec, 2015. ISBN: 978-85-7522-404-5

Arduino em Ação. Martin Evans, Joshua Noble, Jordan Hochenbaum. Novatec, 2013. ISBN: 978-85-7522-373-4.

Arduino Home Automation Projects: Automate your home using the powerful Arduino platform. Marco Schwartz. ISBN 978-1-78398-606-4. Packt Publishing Ltd., 2014.

NEWS

- ▶ Carros terão sistema de rede para intercomunicação
- ▶ Hacker invade sistema de carro e aciona freio.
- ▶ Robôs são usados para ensinar inglês na Coreia do Sul
- ▶ Popularização de casas automatizadas é crescente



SMARTCITY - PROJETO CROATÁ LAGUNA ECOPARK

- ▶ transporte alternativo, compartilhamento de bicicletas e motos;
- ▶ praças dotadas de equipamentos esportivos que geram energia;
- ▶ controle computadorizado da iluminação pública;
- ▶ sistema social totalmente integrado;
- ▶ pagamentos via smartphone;



MUNDO PROGRAMÁVEL

- ▶ “hoje: estamos passando a viver em tempos onde temos acesso [e queremos ter acesso] à programabilidade do mundo e dos dispositivos ao nosso redor.” Silvio Meira (UFPE)

TV

...

Ar-condicionado

Lâmpadas

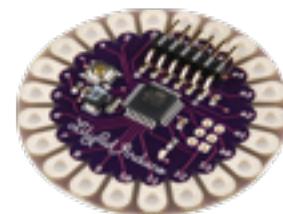
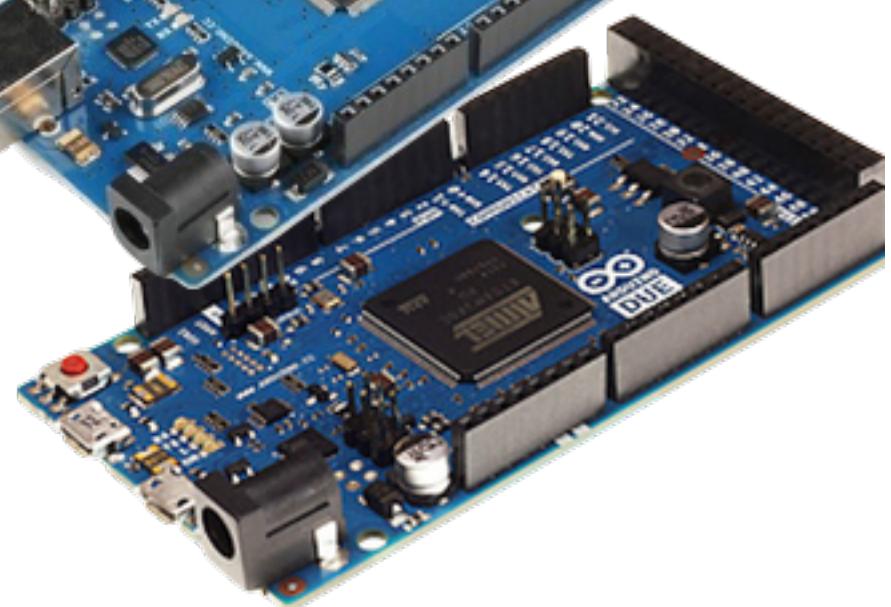
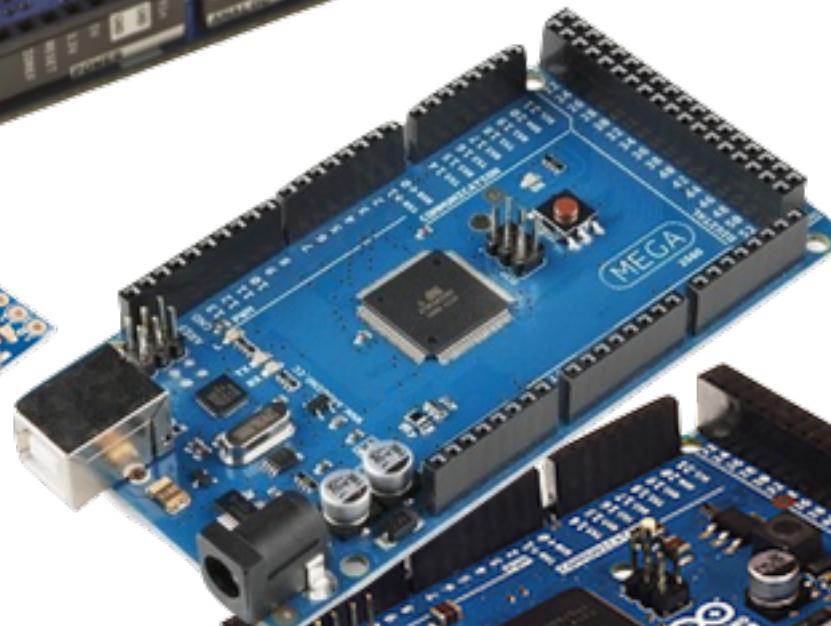
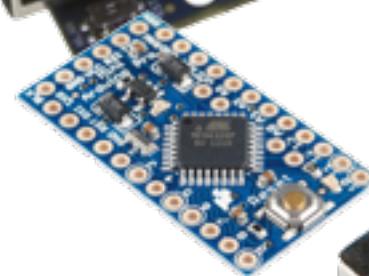
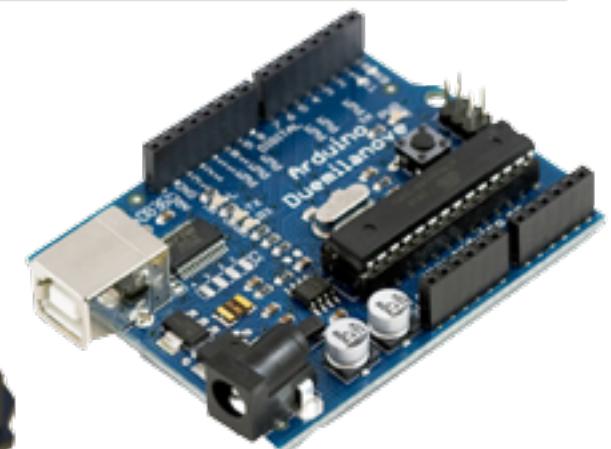
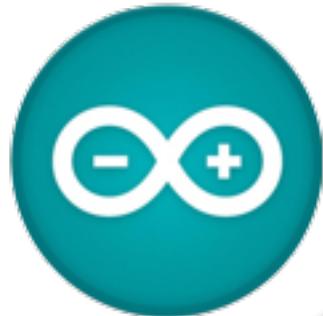
Geladeira

Smartphone

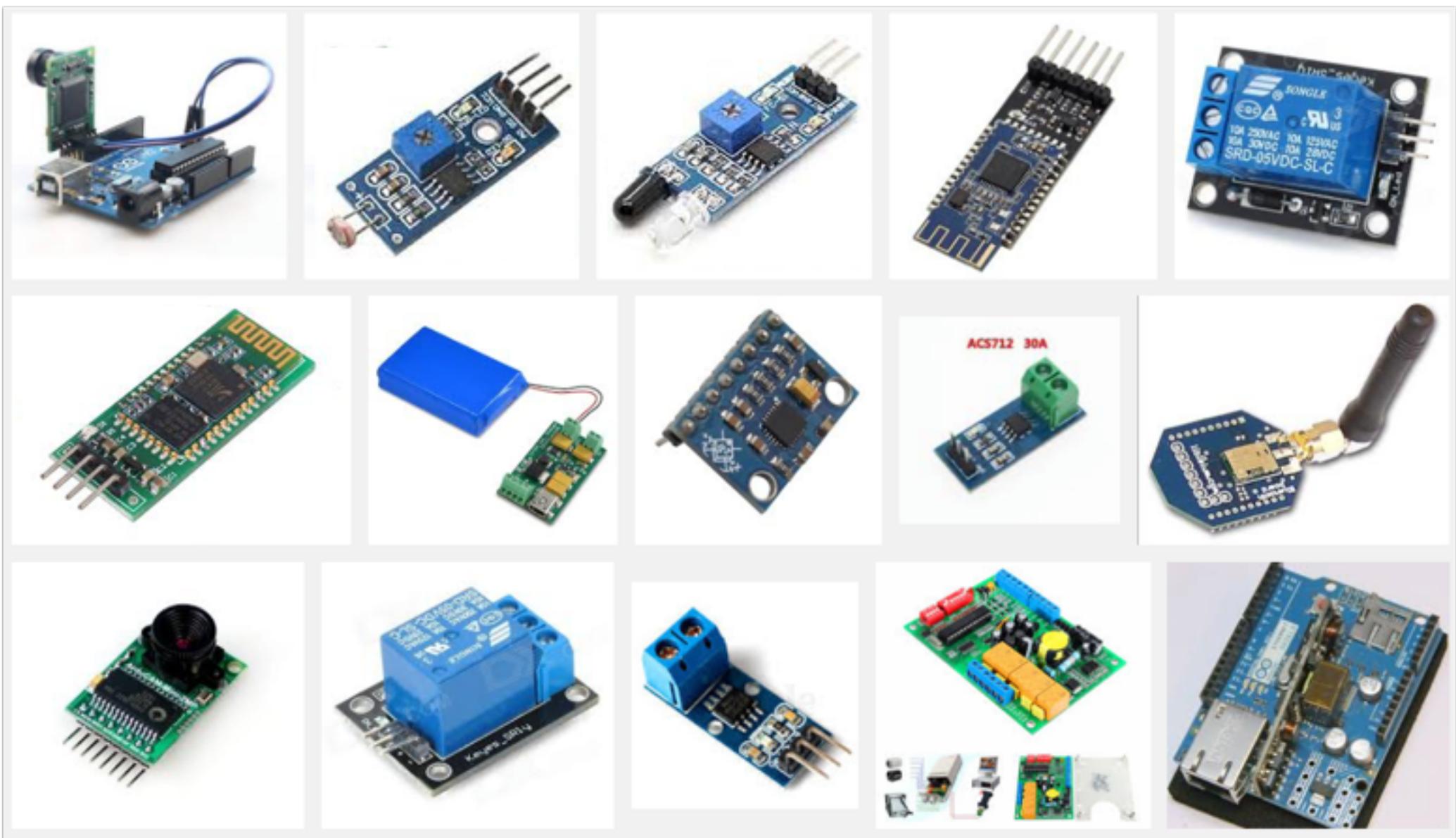
Carros

INTRODUÇÃO

PLATAFORMA ARDUINO







ENTRY LEVEL	ARDUINO UNO	ARDUINO 101	ARDUINO PRO	ARDUINO PRO MINI	ARDUINO MICRO
	ARDUINO STARTER KIT	ARDUINO BASIC KIT	ARDUINO MKR2UNO ADAPTER		
ENHANCED FEATURES	ARDUINO MEGA	ARDUINO ZERO	PROTO SHIELD	MKR PROTO SHIELD	
	MKR PROTO LARGE SHIELD				
INTERNET OF THINGS	ARDUINO MKR1000	MKR1000 BUNDLE	ARDUINO WIFI SHIELD 101	ARDUINO YÚN SHIELD	
WEARABLE	ARDUINO GEMMA	LILYPAD ARDUINO USB	LILYPAD ARDUINO MAIN BOARD		
	LILYPAD ARDUINO SIMPLE	LILYPAD ARDUINO SIMPLE SNAP			
3D PRINTING	MATERIA 101				

BOARDS

MODULES

SHIELDS

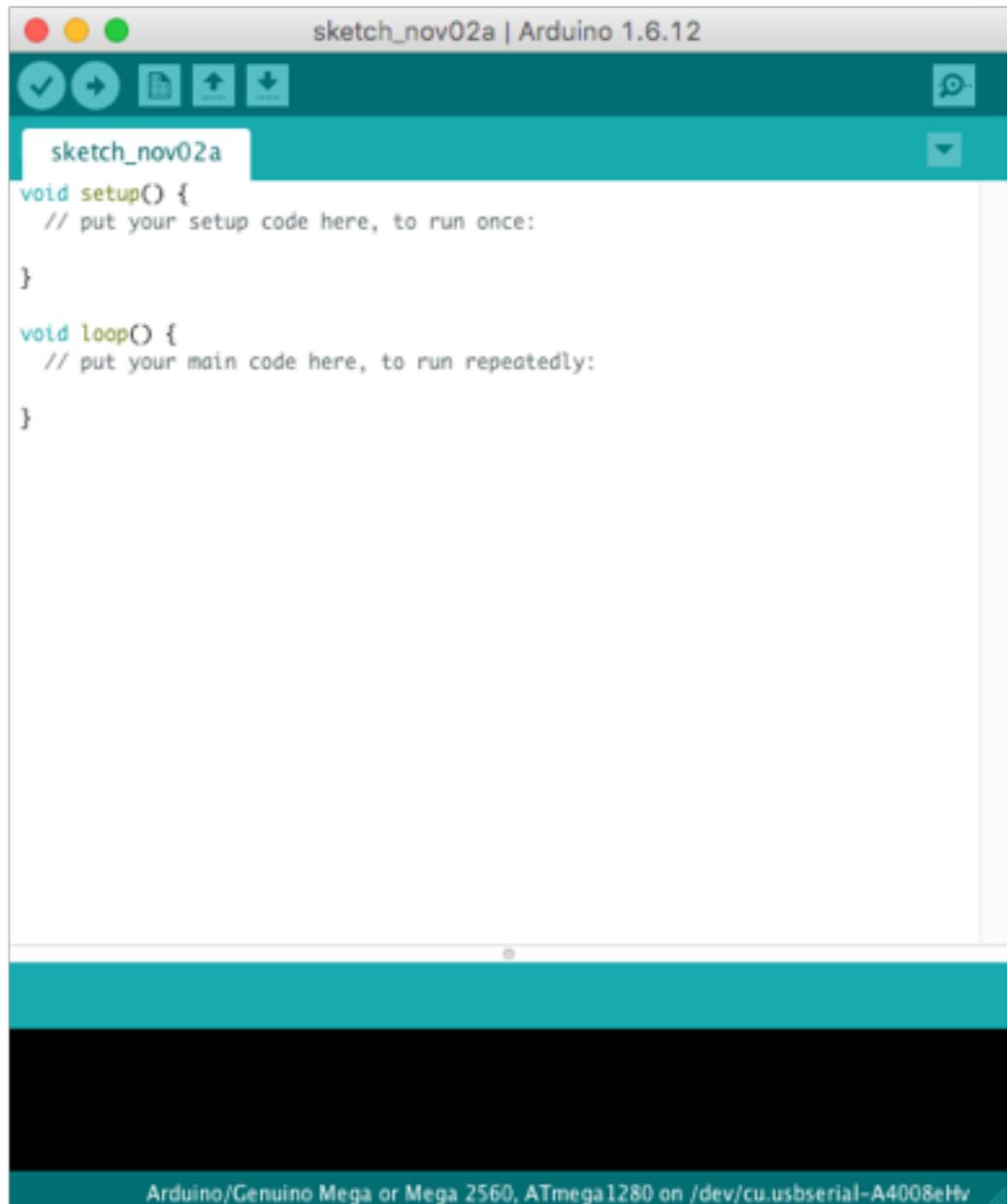
KITS

ACCESSORIES

COMING NEXT

PLATAFORMA ARDUINO

- ▶ Bootloader;
- ▶ Codificação:
 - ▶ IDE Arduino;
 - ▶ AVR Studio;
 - ▶ Fritzing;
 - ▶ Programino;
 - ▶ Eclipse, Sublime...



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_nov02a | Arduino 1.6.12". The main window displays the following code:

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

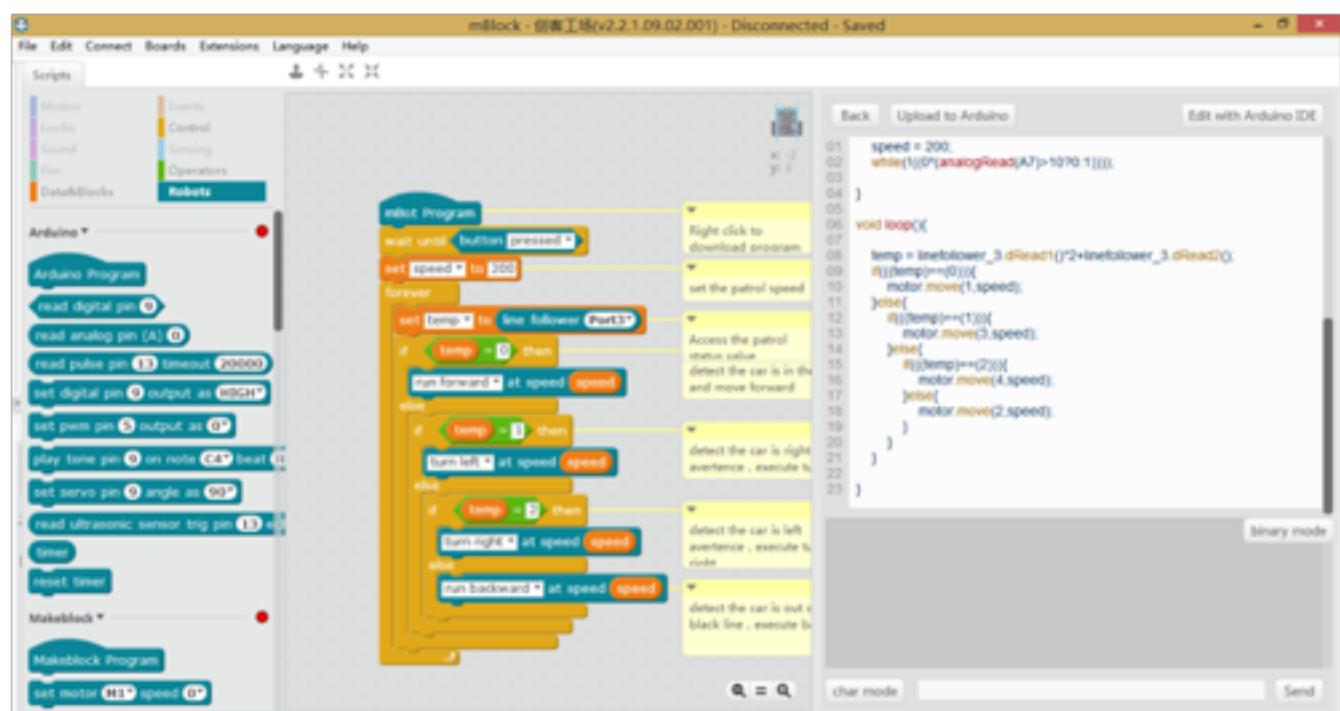
void loop() {
  // put your main code here, to run repeatedly:
}
```

At the bottom of the screen, a status bar indicates: "Arduino/Genuino Mega or Mega 2560, ATmega1280 on /dev/cu.usbserial-A4008eHv".

INTRODUÇÃO

PLATAFORMA ARDUINO

- ▶ Linguagens oficiais suportadas:
 - ▶ C/C++ e todo o pacote AVRLibc/AVR-GCC.
- ▶ Outras linguagens:
 - ▶ Assembler;
 - ▶ Basic;
 - ▶ Python.

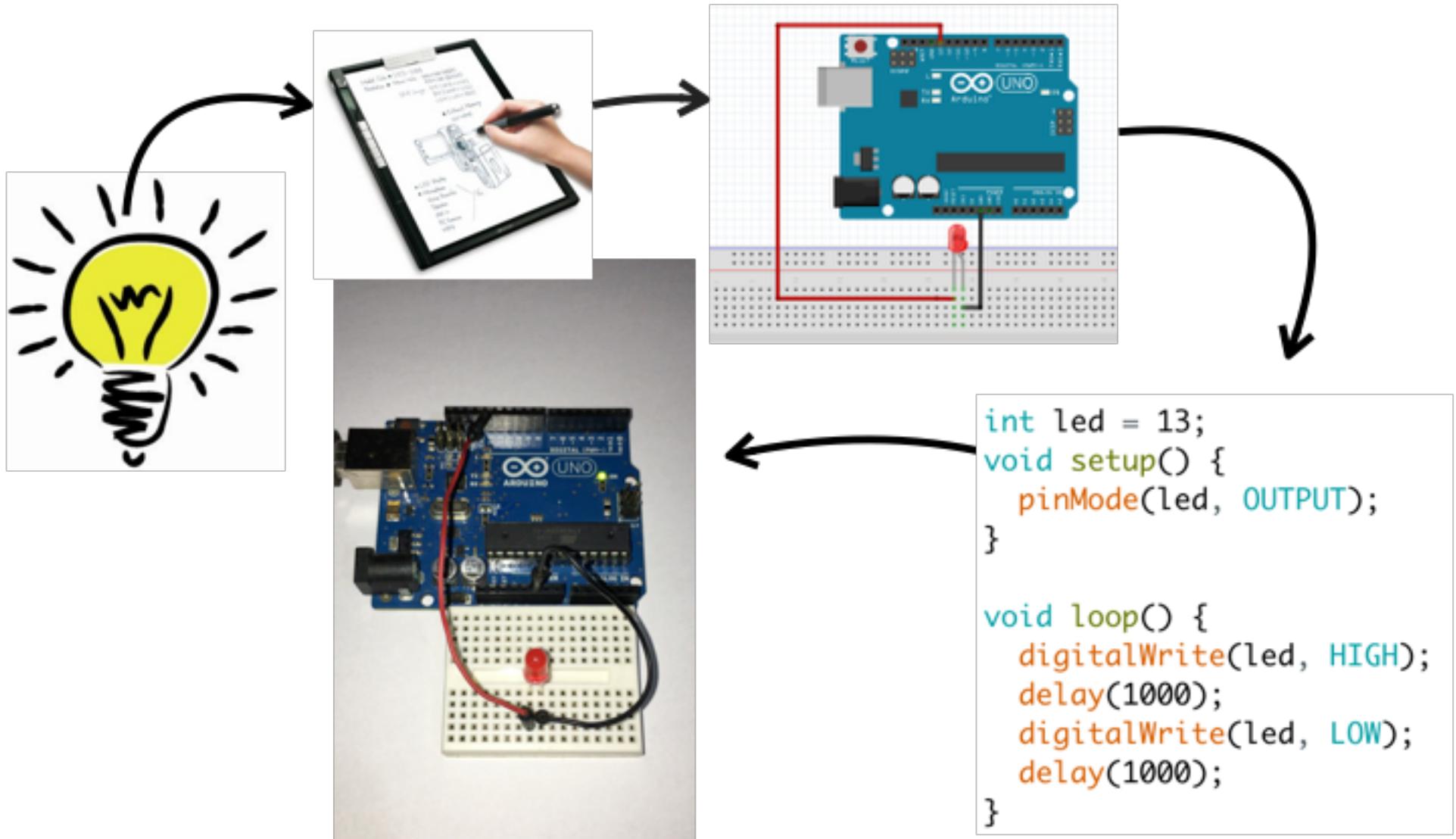


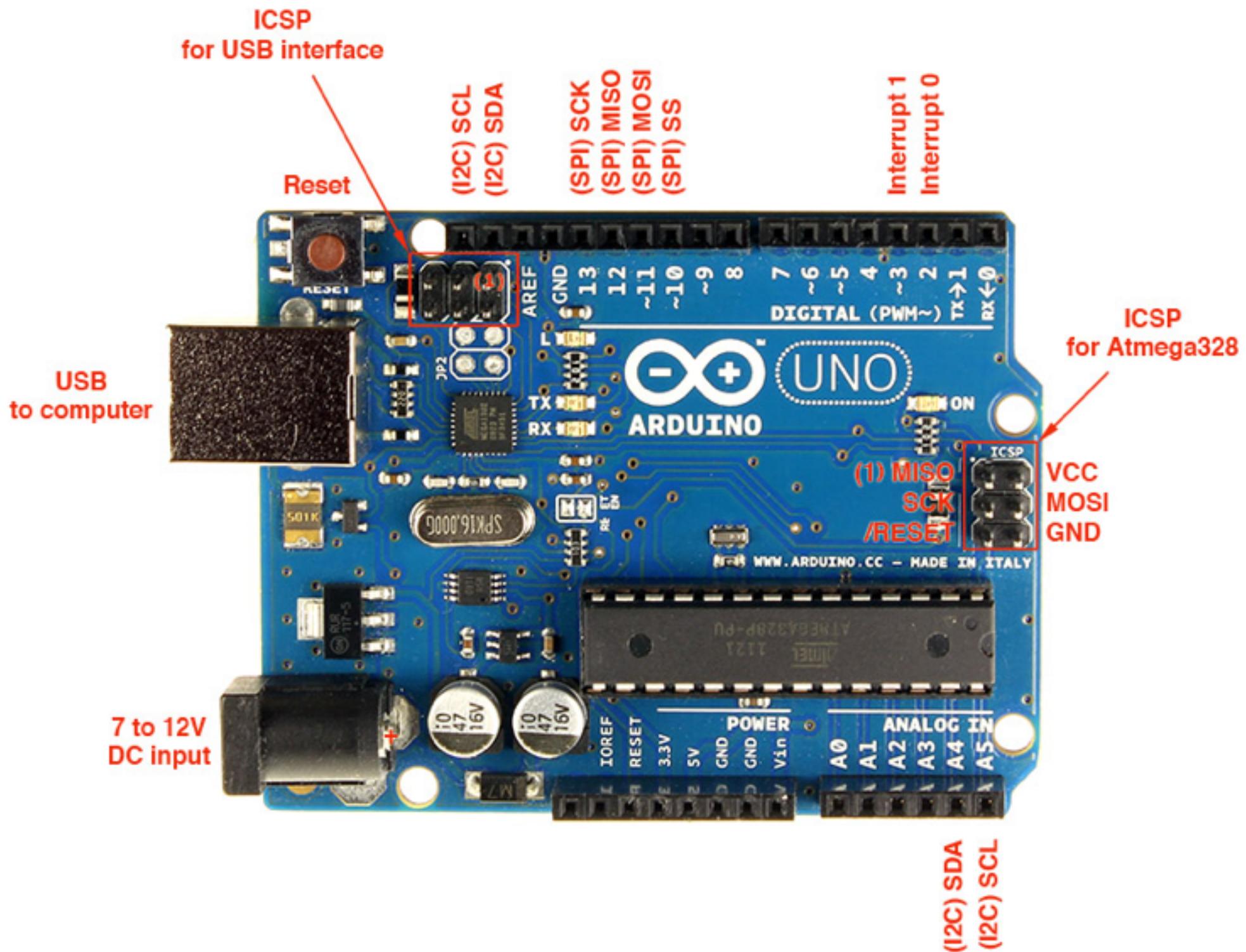
<https://www.arduino.cc/en/Reference/HomePage>

ARDUINO

PROGRAMAÇÃO BÁSICA

PROCESSO DE TRABALHO





Arduino File Edit Sketch Tools Help

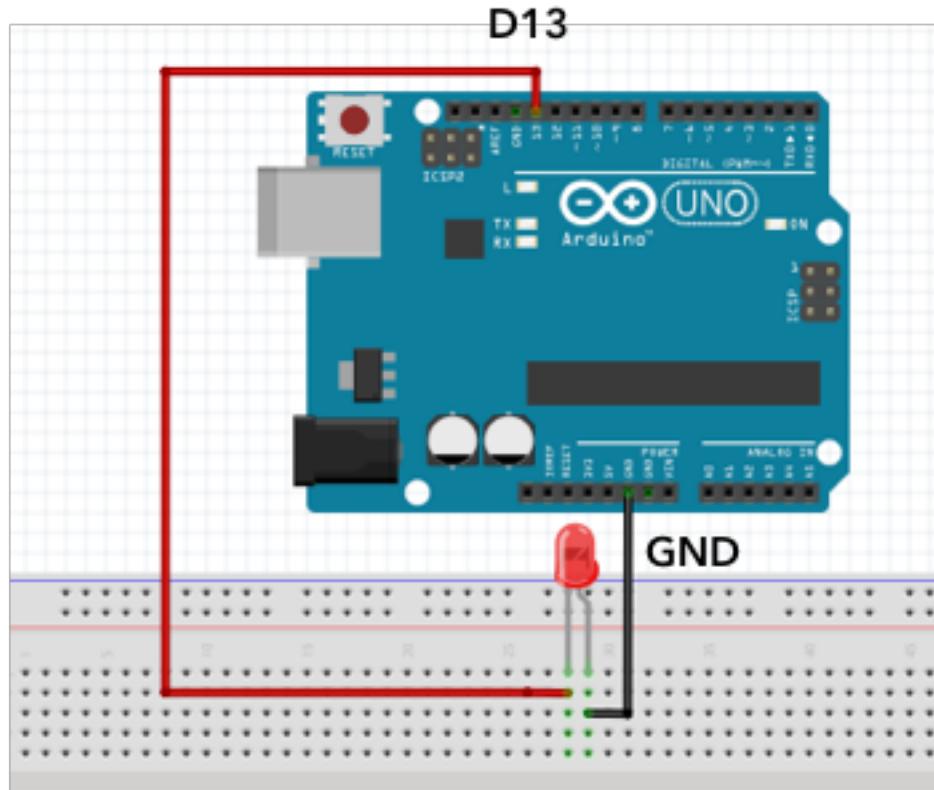
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Serial Monitor ⌘M
Serial Plotter ⌘L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno" ▶
Port ▶
Get Board Info
Programmer: "AVRISP mkII" ▶
Burn Bootloader

```
int led = 13;  
void setup() {  
  pinMode(led, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(led, HIGH);  
  delay(1000);  
  digitalWrite(led, LOW);  
  delay(1000);  
}
```

Done uploading.

Sketch uses 940 bytes (2%) of program storage space. Maximum is 32,256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2,039 bytes for local variables. Maximum is 2,048 bytes.

PROCESSO DE TRABALHO

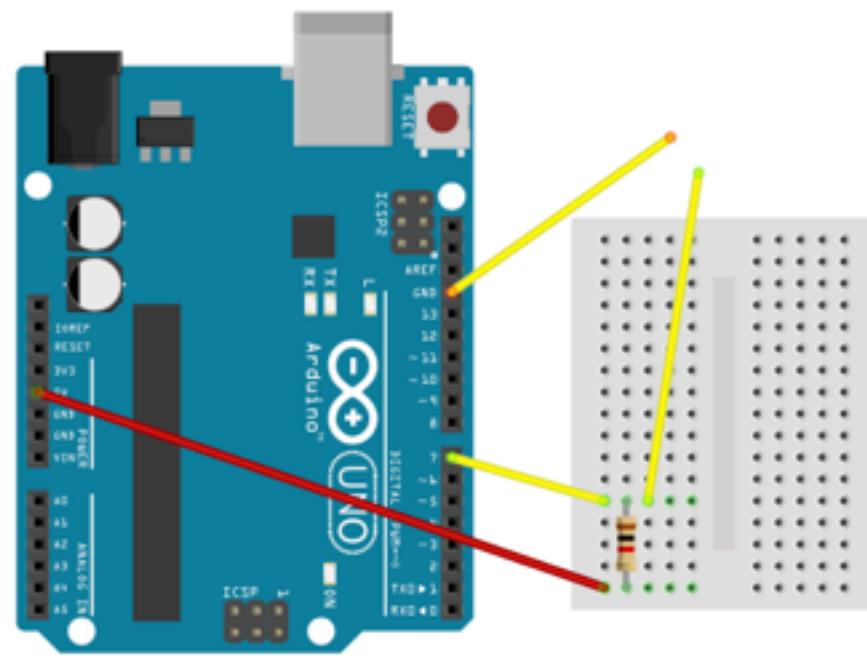


```
int led = 13;  
void setup() {  
    pinMode(led, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(led, HIGH);  
    delay(1000);  
    digitalWrite(led, LOW);  
    delay(1000);  
}
```

EXPERIMENTE ALTERAR A VELOCIDADE DA PISCADA.

ENTRADA PINO DIGITAL

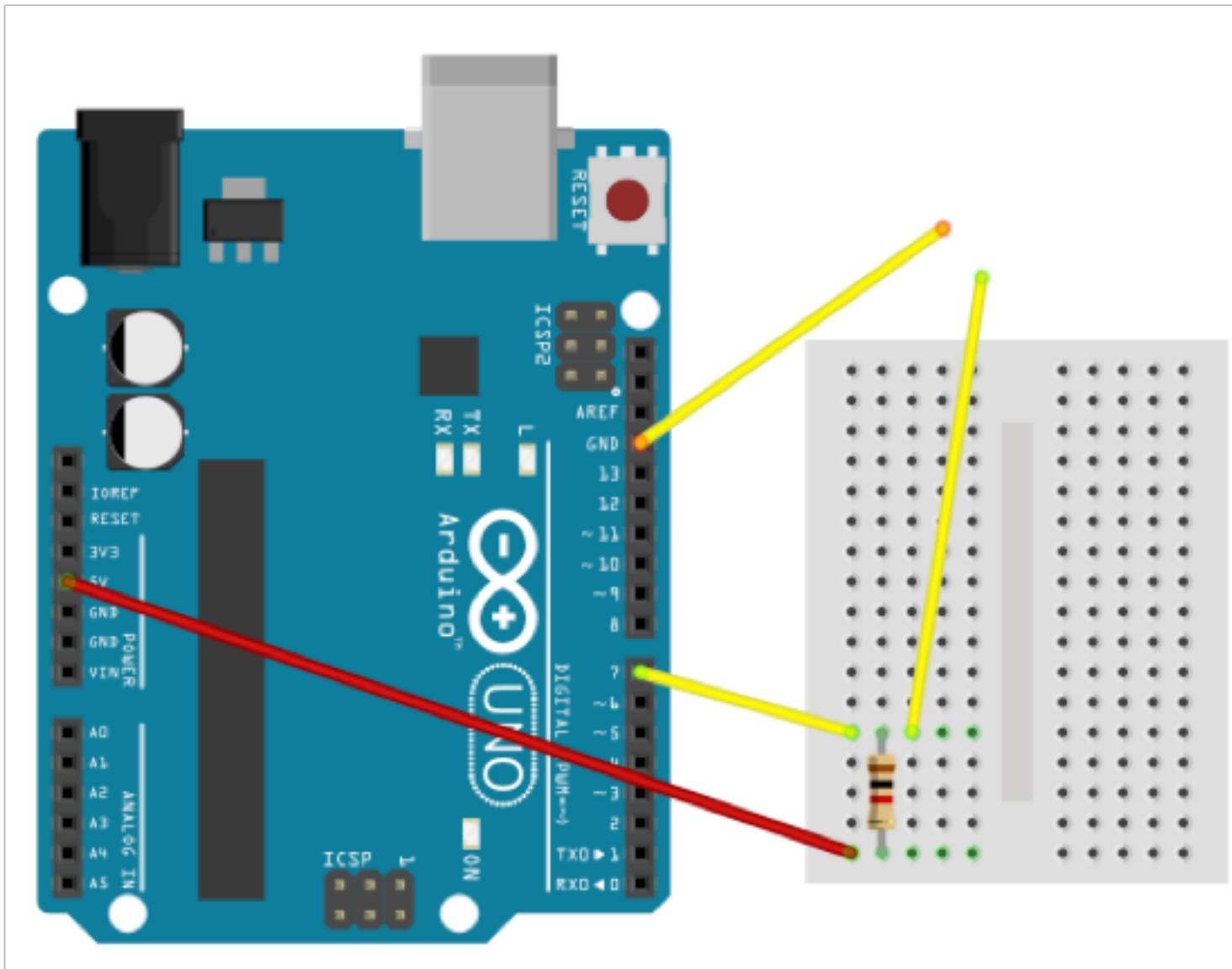
INPUT
INPUT_PULLUP



```
int ledPin = 13;  
int botao = 7;  
  
void setup()  
{  
    pinMode(ledPin, OUTPUT);  
    pinMode(botao, INPUT);  
    //pinMode(botao, INPUT_PULLUP);  
}
```

SE `DIGITALWRITE()` ALTERA O VALOR DA PORTA, `DIGITALREAD()` LÊ?

ENTRADA PINO DIGITAL



FUNÇÃO

```
void piscar(int tempoDelay)
{
    digitalWrite(ledPin, HIGH);
    delay(tempoDelay);
    digitalWrite(ledPin, LOW);
    delay(tempoDelay);
}
```

```
void loop()
{
    if(digitalRead(botao) == LOW)
    {
        piscar(100);
    } else
    {
        piscar(500);
    }
}
```

FAÇA A FUNÇÃO FUNCIONAR NO LOOP DE FORMA CONDICIONADA
AO VALOR EXISTENTE NO “BOTÃO”: TRUE OU FALSE.

SERIAL

**COMO SABER REALMENTE
O VALOR EXISTENTE NAS
VARIÁVEIS E PORTAS?
COMUNICAÇÃO SERIAL.**

```
Serial.begin();
Serial.parseFloat();
Serial.parseInt();
Serial.read();
Serial.print();
Serial.println();
Serial.write();
Serial.serialEvent()**;
```

```
void setup() {
    Serial.begin(9600);
}

void loop() {
    Serial.println("Arduino x Android");
}
```

**VAMOS EXIBIR EM TELA O
VALOR CAPTURADO NA
PORTAL DIGITAL 7?**

ENTRADAS ANALÓGICAS

QUAIS SÃO OS
INTERVALOS MÍNIMO E
MÁXIMO SUPORTADOS
PELA PORTA
ANALÓGICA?

PLUG UM JUMPER NA
ENTRADA A0 E A OUTRA
PONTA EXPERIMENTE:
GND, 5V E 3.3V.

```
int pinoAnalogico = A0;  
int vPino = 0;  
void setup()  
{  
    Serial.begin(9600);  
}  
  
void loop()  
{  
    vPino = analogRead(pinoAnalogico);  
    Serial.print(vPino);  
    delay(500);  
}
```

SAÍDAS PWM - “ANALÓGICAS”

QUAIS SÃO OS INTERVALOS
MÍNIMO E MÁXIMO SUPORTADOS
PELA PORTA PWM?

0-255

PLUG UM LED EM UMA
PORTA PWM (~).
VIA SERIAL MONITOR DIGITE
VALORES ENTRE 0 E 255 PARA
ALTERAR A INTENSIDADE DO
LED.

LEMBRAR DA RESISTÊNCIA.

```
int pinoPWM = 11;  
int vCiclo = 0;  
void setup()  
{  
    pinMode(pinoPWM, OUTPUT);  
    Serial.begin(9600);  
}  
  
void loop()  
{  
    if(Serial.available())  
    {  
        vCiclo = Serial.parseInt();  
        analogWrite(pinoPWM, vCiclo);  
    }  
}
```

ATIVIDADE

**EXPERIMENTE ALTERAR A INTENSIDADE
DO LED ATRAVÉS DO VALOR LIDO NA
PORTAL ANALÓGICA AO.**

**PENSE:
LIMITES DA PORTA DIGITAL: 0-255
LIMITES DA PORTA ANALÓGICA: 0-1023**

SOLUÇÃO FAZ USO DA FUNÇÃO MAP();

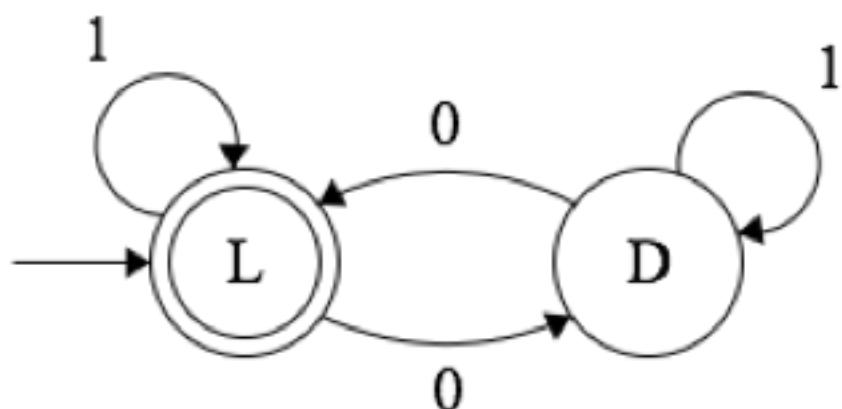
**PROCURE NAS REFERENCIAS NO PORTAL ARDUINO.CC.
LEMBRAR DA RESISTÊNCIA.**

ATIVIDADE

```
int pinoPWM = 11;
int pinoA = A0;
int vPinoA = 0;
int vCiclo = 0;
void setup()
{
    pinMode(pinoPWM, OUTPUT);
    Serial.begin(9600);
}
void loop()
{
    vPinoA = analogRead(pinoA);
    vCiclo = map(vPinoA, 0, 1023, 0, 255);
    analogWrite(pinoPWM, vCiclo);
}
```

ATIVIDADE

CRIE UMA MÁQUINA DE ESTADO CAPAZ DE
ALTERAR O VALOR DO LED SOMENTE
QUANDO UM “BOTÃO” FOR PRESSIONADO.



DICA!
PARA MANTER O BOTÃO EM NÍVEL
ALTO USE: INPUT_PULLUP.

PARA BAIXAR O NÍVEL, CONECTE
O JUMPER DO “BOTÃO” EM GND.

ATIVIDADE

```
#define D 0
#define L 1
int estado;
int botao = 7;
int vBotao = 1;
int led = 13;

void setup()
{
    pinMode(botao, INPUT_PULLUP);
    pinMode(led, OUTPUT);
    Serial.begin(9600);
    estado = L;
}
```

```
void loop()
{
    vBotao = digitalRead(botao);
    Serial.print("Estado: ");Serial.print(estado);
    Serial.print("\tBotao: ");Serial.println(vBotao);
    switch(estado) {
        case L:
            if(vBotao == LOW)
                estado = D;
            break;
        case D:
            if(vBotao == LOW)
                estado = L;
            break;
        default:
            break;
    }
    digitalWrite(led, estado);
    delay(2000);
}
```

PROBLEMA DO DELAY

SUPONHA QUE CONSTRUIU UM SEMÁFORO DE PEDESTRE!

SEMPRE QUE UM PEDESTRE PRESSIONA UM BOTÃO O SINAL DARÁ UM TEMPO E FECHARÁ OS VEÍCULOS E ABRIRÁ PARA A FAIXA DE PEDESTRE.

QUAL É A SOLUÇÃO?

COMO SERÁ POSSÍVEL REALIZAR A LEITURA CONSTANTE DO BOTÃO SE O SINAL ESTARÁ EM "DELAY" PARA MANTER OS LEDS ACESOS.

SEU SEMÁFORO DEVE RESPEITAR O CICLO DE FUNCIONAMENTO, ORDEM DAS CORES E AO MESMO TEMPO ESCUTAR CONSTANTEMENTE O BOTÃO DE PEDESTRE.

ARDUINO

INTERRUPÇÃO

INTERRUPÇÃO

INTERRUPÇÕES PERMITEM QUE O MICROCONTROLADOR RESPONDA A EVENTOS SEM QUE SEJA PRECISO CONSULTÁ-LOS A TODO O MOMENTO PARA VERIFICAR SE HOUVE ALGUMA ALTERAÇÃO.

GERALMENTE TEMOS AS INTERRUPÇÕES POR HARDWARE E POR TEMPO.

CUIDADOS COM AS INTERRUPÇÕES.

INTERRUPÇÃO POR HARDWARE

DISPARADA QUANDO ALGUM PINO SOFRE
ALTERAÇÃO DE VALOR DE ACORDO COM O
TRATAMENTO ESPECIFICADO NA INTERRUPÇÃO.

Número da interrupção

	0	1	2	3	4	5
UNO/Nano	D2	D3	-	-	-	-
Leonardo	D3	D2	D0	D1	-	-
Mega2560	D2	D3	D21	D20	D19	D18
Due	Todos pinos digitais					

INTERRUPÇÃO POR HARDWARE

LOW

Dispara a interrupção sempre que estiver em nível baixo (LOW).

RISING

Dispara quando o pino passa de baixo (LOW) para alto (HIGH).

FALLING

Dispara quando o pino passa de alto (HIGH) para baixo (LOW).

CHANGE

Dispara sempre que o pino muda de nível em qualquer sentido.

HIGH

Dispara sempre que estiver em nível alto (HIGH). Apenas Due.

INTERRUPÇÃO POR HARDWARE

EXEMPLO: QUANDO O PINO PASSAR DE LOW PARA HIGH A FUNÇÃO ALGOACONTEceu SERÁ INVOCADA.

VOLATILE

```
void setup()
{
    pinMode(led, OUTPUT);
    pinMode(2, INPUT_PULLUP);
    attachInterrupt(0, algoAconteceu, RISING);
}
```

DIGITALPINTOINTERRUPT

ATIVIDADE

VAMOS ALTERAR A ATIVIDADE DO LED COM A
MÁQUINA DE ESTADO?

```
void setup()
{
    pinMode(led, OUTPUT);
    pinMode(2, INPUT_PULLUP);
    attachInterrupt(0, algoAconteceu, RISING);
}
```

INFORMAÇÕES RELEVANTES

PROCURAR POR:
ARDUINO + ISR
(INTERRUPT SERVICE ROUTINES)

CUIDADOS:
FUNÇÕES RÁPIDAS;
DADOS DA FUNÇÃO E O RESTO DO CÓDIGO DO TIPO VOLATILE;
NÃO UTILIZAR DELAY, EM ÚLTIMO CASO USAR: DELAYMICROSECONDS;
NÃO CONFIAR CEGAMENTE NA COMUNICAÇÃO, LEITURA E ESCRITA.

INTERRUPÇÃO POR TEMPO

UMA BIBLIOTECA QUE
PERMITE ACESSO AO TIMER2.
(SIM, TEMOS BIBLIOTECAS).

INVOCAR UMA FUNÇÃO EM UMA
JANELA DE TEMPO FIXO.

```
#include <MsTimer2.h>
const int led = 13;
volatile boolean valor = HIGH;
void pisca() {
    digitalWrite(led, valor);
    valor = !valor;
}
void setup() {
    pinMode(led, OUTPUT);
    MsTimer2::set(1000, pisca);
    MsTimer2::start();
}
void loop() { }
```

BIBLIOTECAS

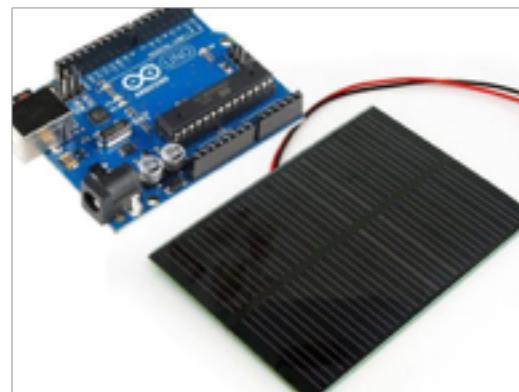
- ▶ EPROM
- ▶ ETHERNET
- ▶ FIRMATA
- ▶ LIQUIDCRYSTAL
- ▶ SD
- ▶ SERVO
- ▶ SPI
- ▶ SOFTWARE SERIAL
- ▶ STEPPER
- ▶ WIFI
- ▶ WIRE
- ▶ KEYBOARD
- ▶ MOUSE
- ▶ AUDIO
- ▶ SCHEDULER
- ▶ USBHOST
- ▶ ONEWIRE
- ▶ XBEE
- ▶ GFX
- ▶ CAPSENCE

ARDUINO

ENERGIA

FORMAS DE POUPAR ENERGIA PARA USO COM BATERIAS

- ▶ COLOCAR O MICROCONTROLADOR PARA DORMIR QUANDO NÃO ESTIVER DESEMPENHANDO UMA TAREFA;
- ▶ FAZER O ARDUINO FUNCIONAR COM UMA TENSÃO MENOR;
- ▶ FAZER O ARDUINO FUNCIONAR COM UMA FREQUÊNCIA DE RELÓGIO MENOR.



DORMIR

- ▶ DESLIGAR COISAS! É POSSÍVEL DESLIGAR PARTE DOS RECURSOS, COMO ENTRADAS ANALÓGICAS, INTERFACES SPI, TWI, USART E OS TEMPORIZADORES 0, 1 E 2.
- ▶ PARA REALIZAR ESTA AÇÃO É PRECISO UTILIZAR A BIBLIOTECA: AVR/POWER.H.

POWER_ADC_DISABLE();
POWER_SPI_DISABLE();
POWER_TWI_DISABLE();
POWER_USART0_DISABLE();
POWER_TIMER0_DISABLE();
POWER_TIMER1_DISABLE();
POWER_TIMER2_DISABLE();
POWER_ALL_DISABLE();



DORMIR

- ▶ TAMBÉM É POSSÍVEL DESLIGAR COMPLETAMENTE A PLACA E, CLARO, RELIGAR QUANDO NECESSÁRIO.
- ▶ BIBLIOTECA: NARCOLEPTIC.
- ▶ [HTTPS://GITHUB.COM/BRABL2/NARCOLEPTIC](https://github.com/BRABL2/Narcoleptic)



```
void loop() {  
    digitalWrite(LED_BUILTIN, HIGH);  
    Narcoleptic.delay(1000);  
    digitalWrite(LED_BUILTIN, LOW);  
    Narcoleptic.delay(1000);  
}
```

DORMIR

- ▶ ALGO DEVE ACONTECER SE UM BOTÃO FOR PRESSIONADO? VERIFIQUE MENOS.

```
void loop() {  
    if(digitalRead(pino) == LOW) {  
        facaAlgo();  
    }  
    Narcoleptic.delay(100);  
}
```



DORMIR

- ▶ SE NECESSITAR
DE ACORDAR
IMEDIATAMENTE
APÓS UM EVENTO

```
void dormir() {  
    set_sleep_mode(SLEEP_MODE_PWR_DOWN);  
    sleep_enable();  
    attachInterrupt(0, setAcao, HIGH);  
    sleep_mode();  
    sleep_disable();  
    detachInterrupt(0);  
}
```

CÓDIGO COMPLETO: DORMIR.INO

VELOCIDADE DO CLOCK

- ▶ NÃO PRECISA DE DESEMPENHO? PRA QUE ANDAR ACELERADO?

```
#include<prescaler.h>

const int led = 13;

void setup(){
    pinMode(led, OUTPUT);
    setClockPrescaler(CLOCK_PRESCALER_1);
    //setClockPrescaler(CLOCK_PRESCALER_256);
}

void loop() {
    digitalWrite(led, HIGH);
    trueDelay(1000);
    digitalWrite(led, LOW);
    trueDelay(1000);
}
```

ALIMENTAÇÃO CONSTANTE

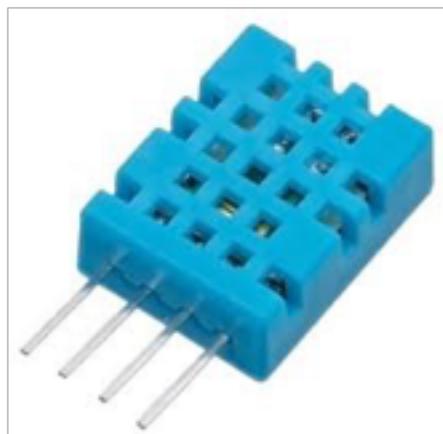
- ▶ EXEMPLOS:
 - ▶ SENSORES DE LUZ;
 - ▶ TEMPERATURA;
 - ▶ UMIDADE...
- ▶ PRECISO MANTER RECURSOS DIRETAMENTE ALIMENTADOS SE OS UTILIZO EM EVENTOS CURTOS.

ARDUINO

SENSORES

DHT11

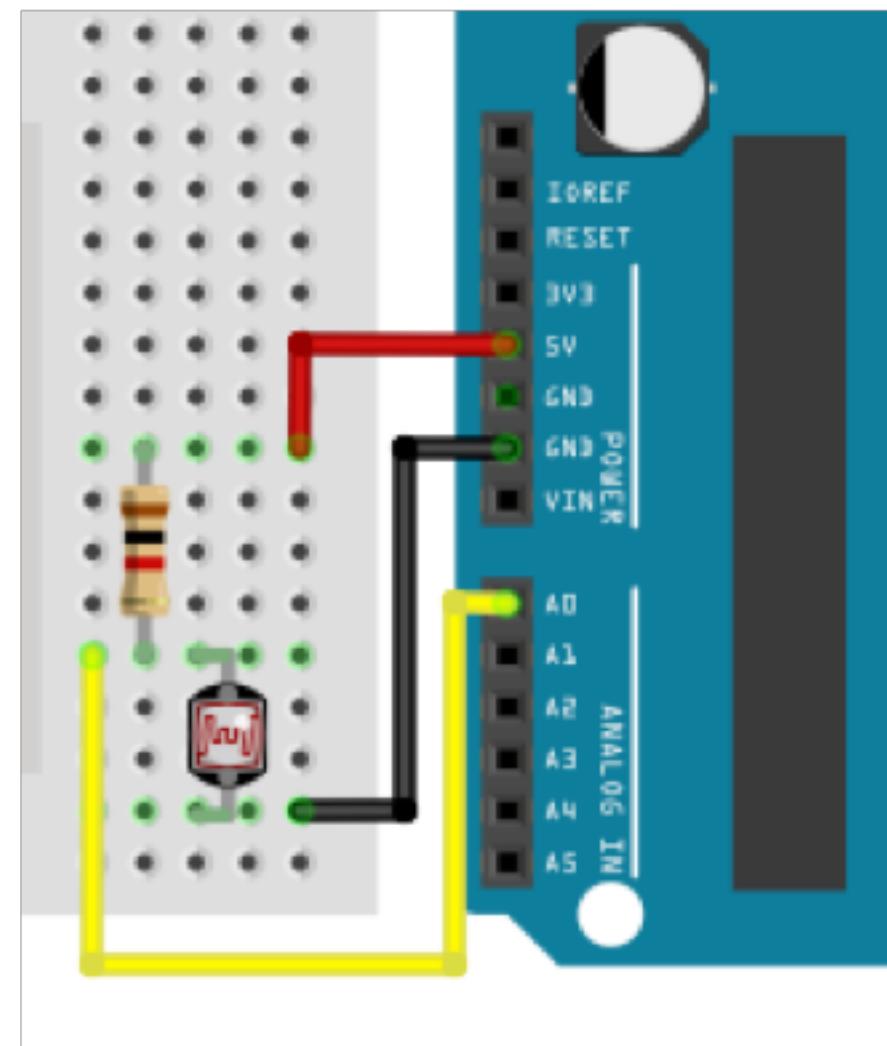
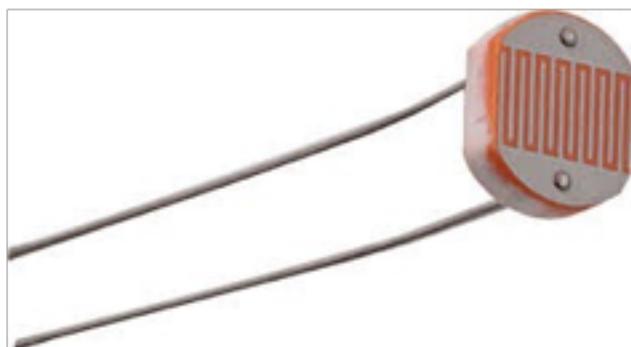
- ▶ DHT11 FORNECE LEITURA AMBIENTE DA TEMPERATURA E UMIDADE.
- ▶ BAIXE A BIBLIOTECA: DHT.H
- ▶ REQUER UMA RESISTÊNCIA DE 10KOHM



```
#include "dht.h"
void loop() {
    float t = dht.readTemperature();
    float h = dht.readHumidity();
}
```

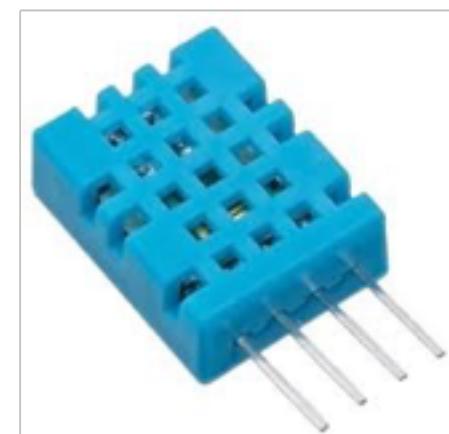
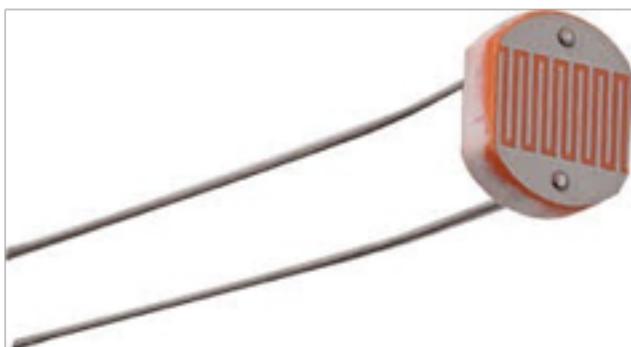
LDR

- ▶ LDR FORNECE LEITURA AMBIENTE DA INTENSIDADE DE LUZ.
- ▶ REQUER UMA RESISTÊNCIA DE 1KOHM



ATIVIDADE

- ▶ DESENVOLVA UM CÓDIGO QUE EFETUA A LEITURA DA TEMPERATURA E UMIDADE SOMENTE A CADA "X" SEGUNDOS.
- ▶ A LEITURA DA INTENSIDADE DE ILUMINAÇÃO NÃO NECESSITA SER REALIZADA MAIS DO QUE 10 VEZES POR SEGUNDO.
- ▶ USE RECURSOS DE ECONOMIA DE ENERGIA.

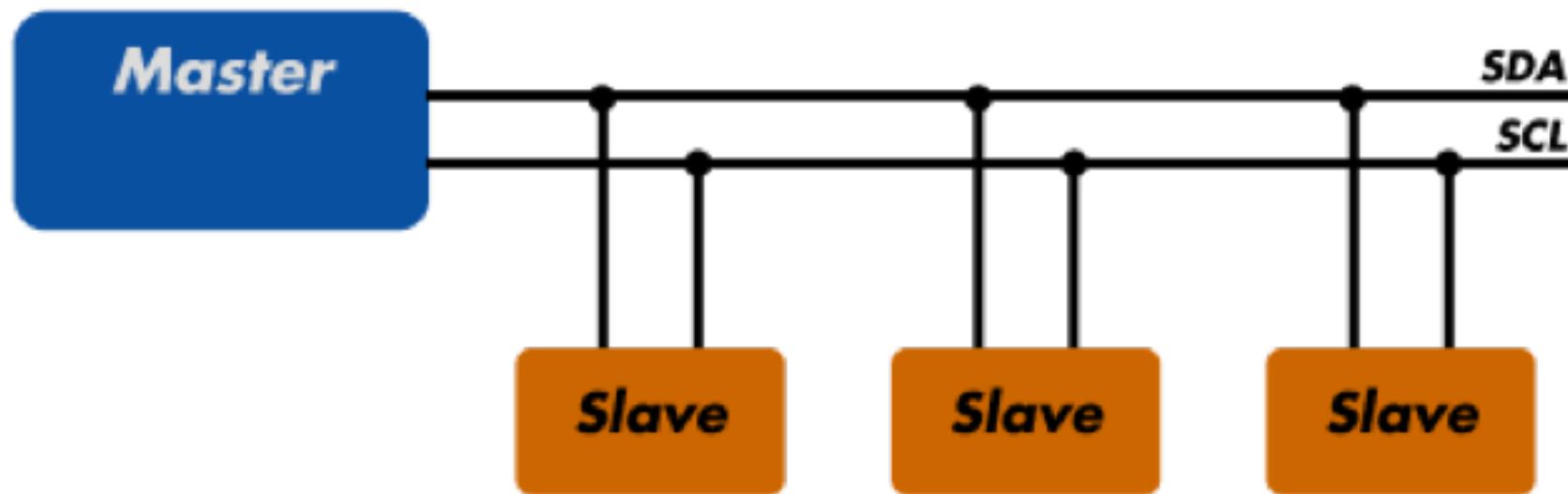


ARDUINO

BARRAMENTO E INTERFACE

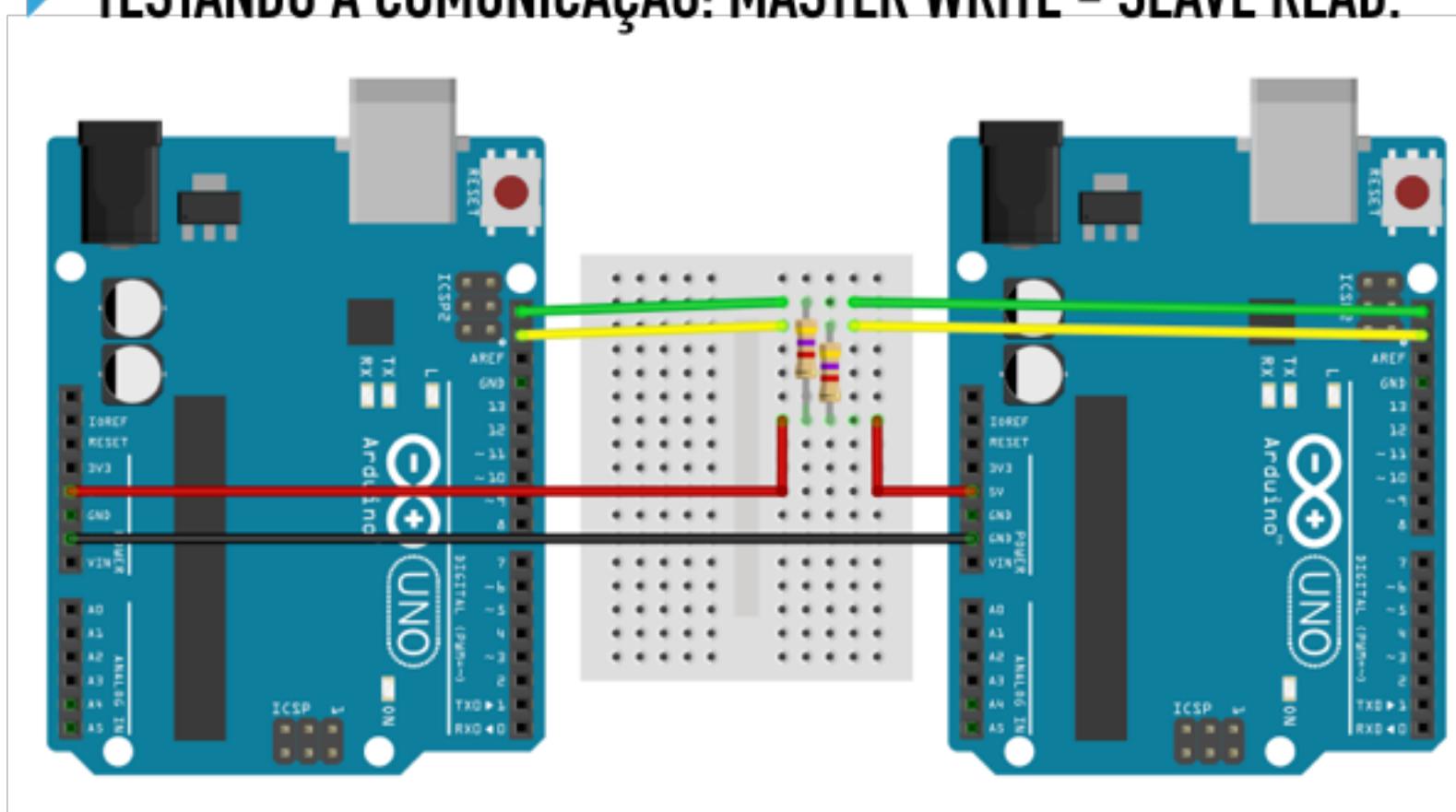
I2C

- ▶ PROTOCOLO I2C CONTROLA UM BARRAMENTO COM 2 FIOS PARA PERIFÉRICOS DE BAIXA VELOCIDADE.
- ▶ A ARDUINO UNO UTILIZA 7 BITS PARA ENDEREÇAR OS PERIFÉRICOS.



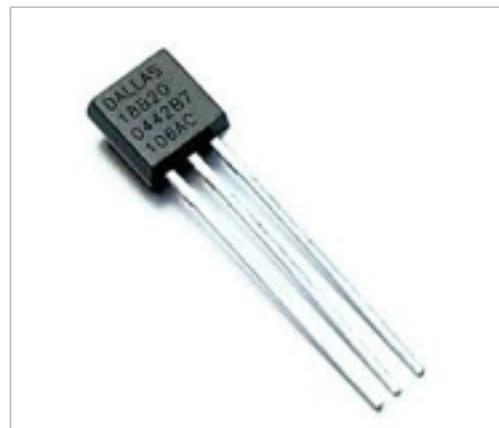
I²C

- ▶ BIBLIOTECA WIRE;
- ▶ TESTANDO A COMUNICAÇÃO: MASTER WRITE - SLAVE READ.



1-WIRE

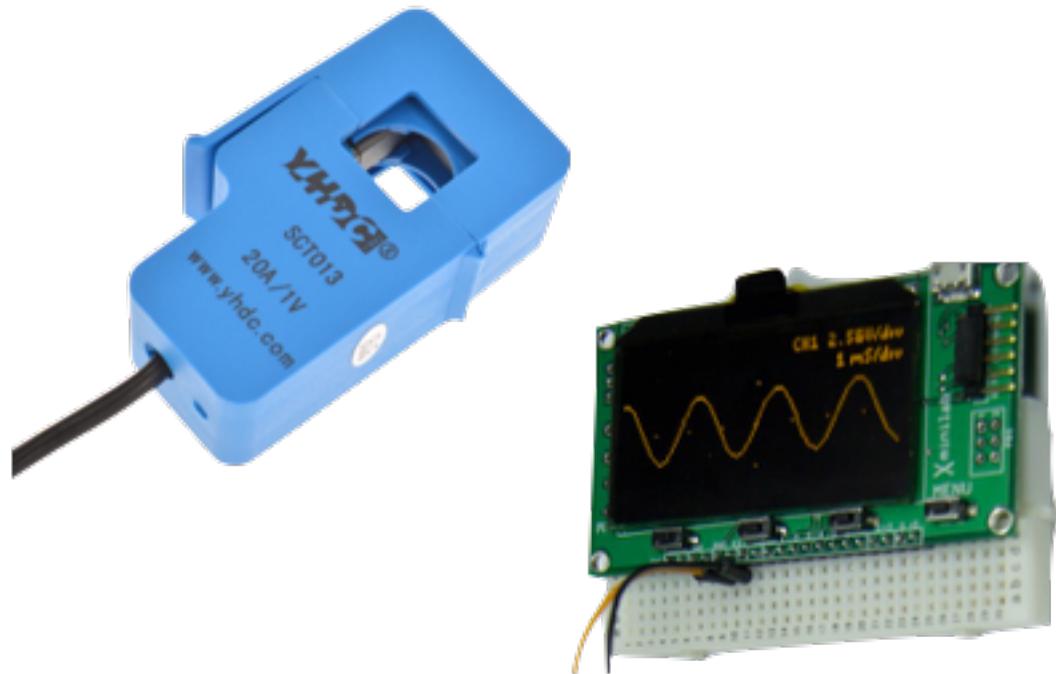
- ▶ BIBLIOTECA ONEWIRE;
- ▶ BARRAMENTO DE 1 FIO;
- ▶ DALLAS INSTRUMENT



- ▶ SENSORES DE PRECISÃO;
- ▶ EXPANSÃO DE MEMÓRIA EEPROM.

SPI

- ▶ BIBLIOTECA SPI:
 - ▶ BARRAMENTO DE 4* FIOS;
 - ▶ SCLK, MOSI, MISO, SS
 - ▶ VEJA ICSP NA PLACA.



- ▶ A CADA NOVO DISPOSITIVO NO 'BARRAMENTO' UM NOVO PINO SS SERÁ NECESSÁRIO.

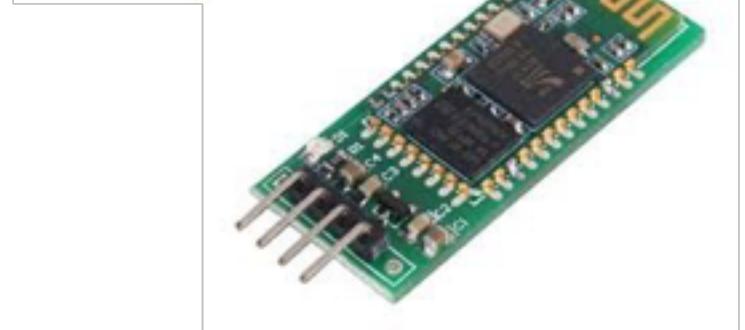
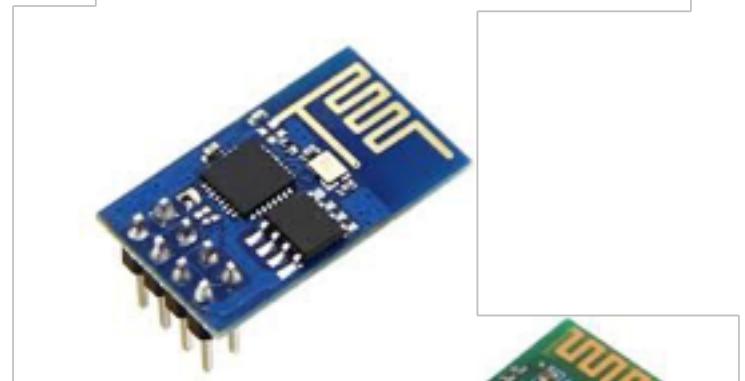
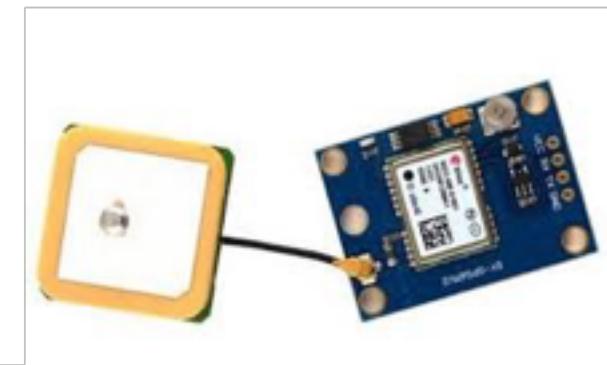
UART

- ▶ BIBLIOTECA SERIAL;
- ▶ INTERFACE PONTO A PONTO.

	Qtd portas	Portas (RX, TX)
UNO	"1"	D0, D1
Leonardo	2*	D0, D1;
Mega2560	4	D0, D1; D19, D18;
Due	4	D17, D16; D15, D14

UART

- ▶ DISPOSITIVOS QUE UTILIZAM ESTE CANAL:
 - ▶ GPS;
 - ▶ GPRS;
 - ▶ BLUETOOTH;
 - ▶ WIFI;
 - ▶ ETHERNET;
 - ▶ ARDUINO...

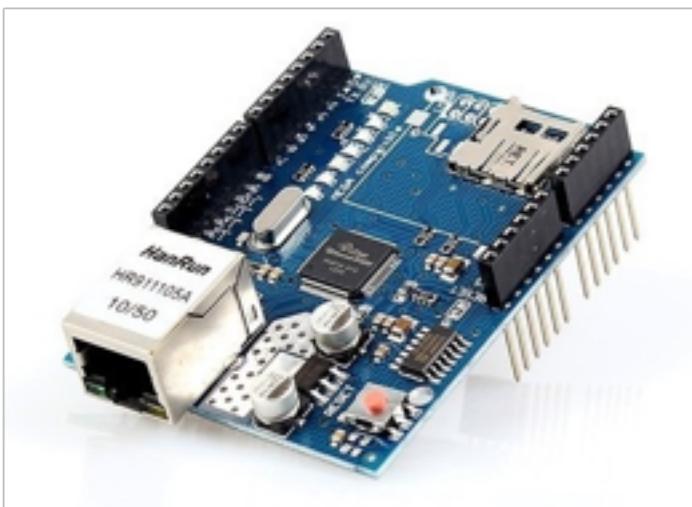


ARDUINO

ETHERNET

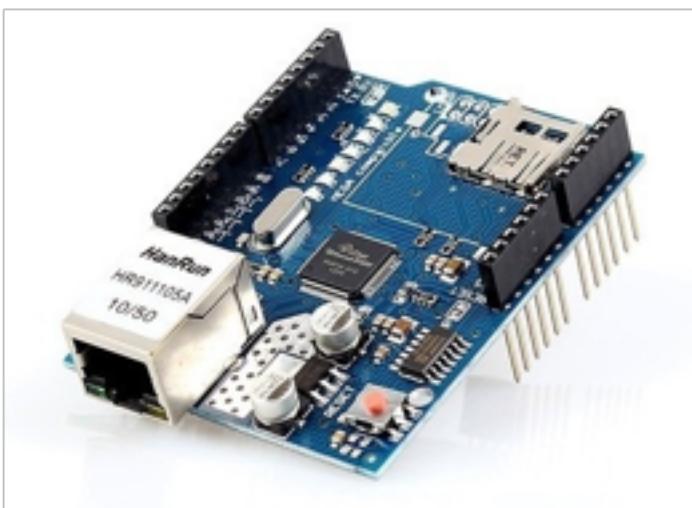
SHIELD | MÓDULO

- ▶ ACESSO A REDE LOCAL*
- ▶ SUPORTE AOS PROTOCOLOS: HTTP, UDP, NTP, DHCP
- ▶ UTILIZA O BARRAMENTO SPI: PORTAS 10, 11, 12, 13
- ▶ SD USA A PORTA 4.



SHIELD | MÓDULO

- ▶ EXPERIMENTE OS EXEMPLOS DE IP AUTOMÁTICO (DHCP).
- ▶ VEJA O QUE OCORRE SE TENTAR ACESSAR A PÁGINA DO GOOGLE (CLIENT WEB).

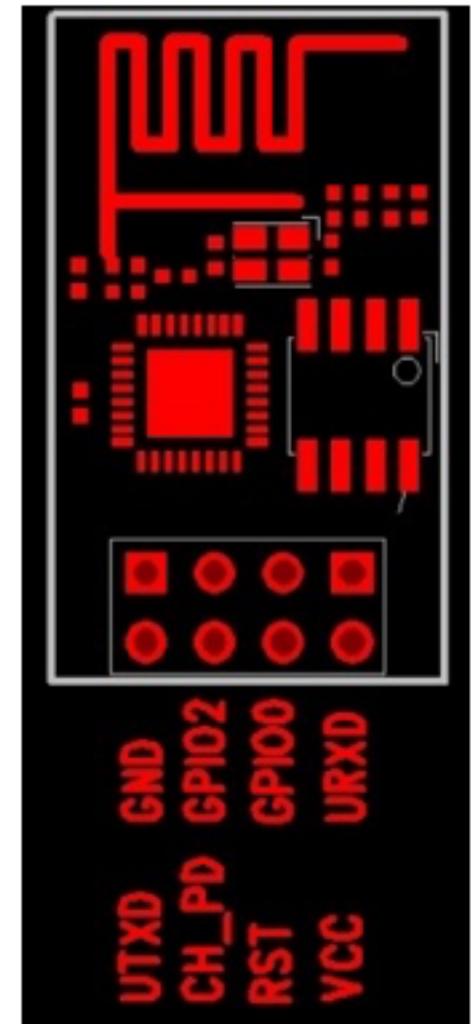
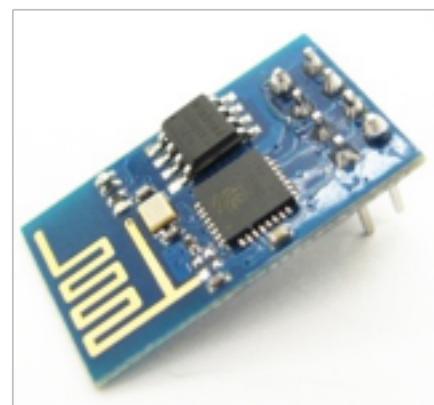


ARDUINO

ESP8266

ESP8266

- ▶ ACESSO A REDE LOCAL* WIFI
 - ▶ SUPORTE AOS PROTOCOLOS: HTTP, UDP, DHCP, AT
 - ▶ UTILIZA O BARRAMENTO UART
 - ▶ POSSUI INDEPENDÊNCIA: GPIO



https://www.itead.cc/wiki/ESP8266_Serial_WIFI_Module

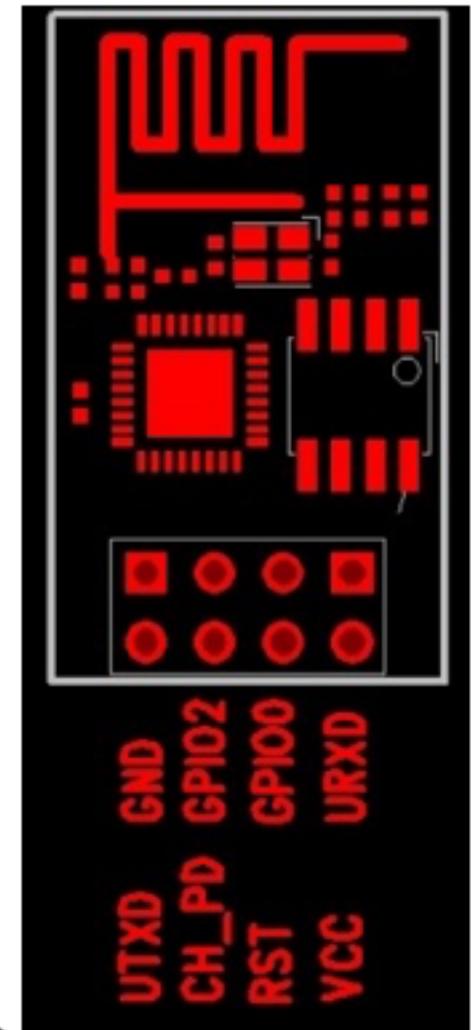
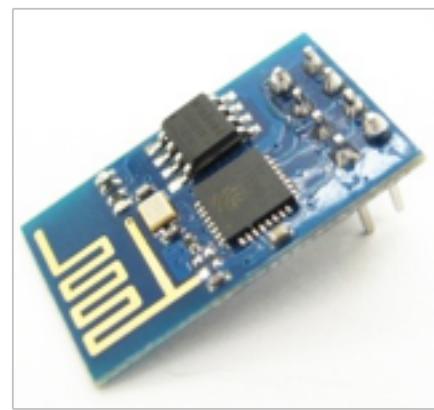
ESP8266

- ▶ PARA INSTALAR BUSQUE NO GOOGLE:
 - ▶ ESP8266 ARDUINO INSTALL GITHUB
 - ▶ INSTALAR AS PROPRIEDADES E AS BIBLIOTECAS.



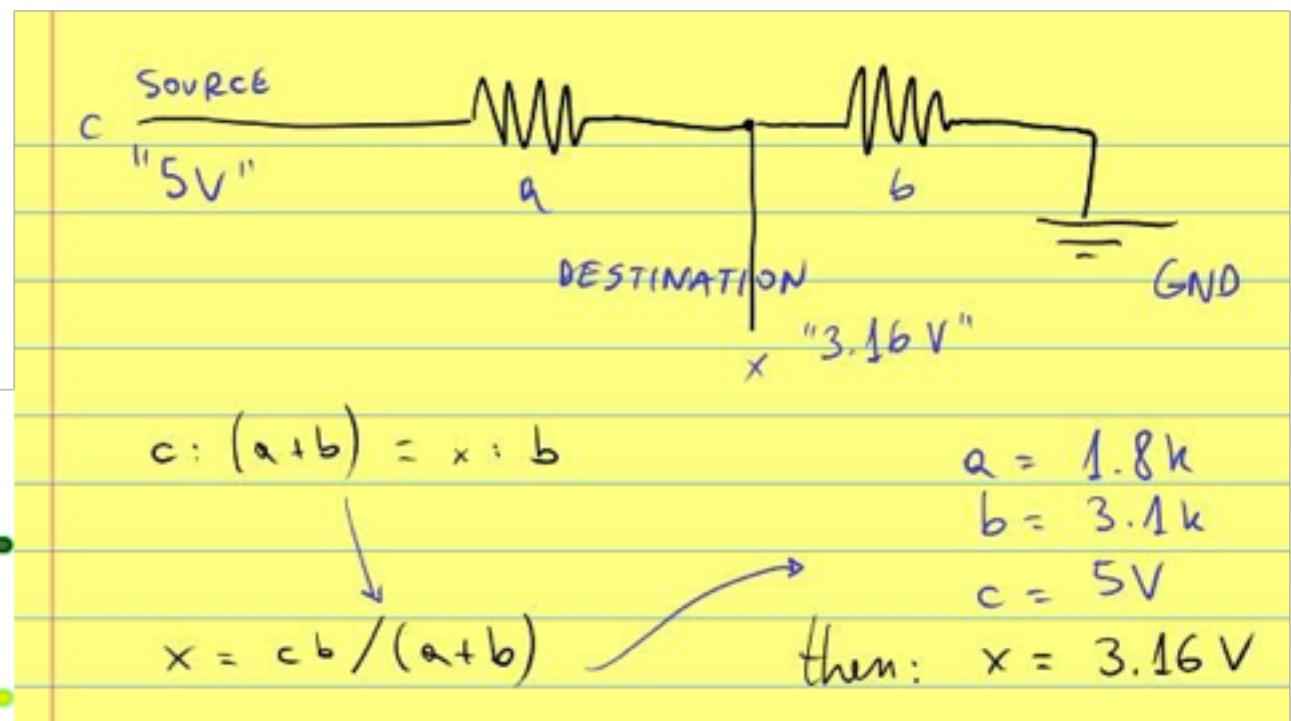
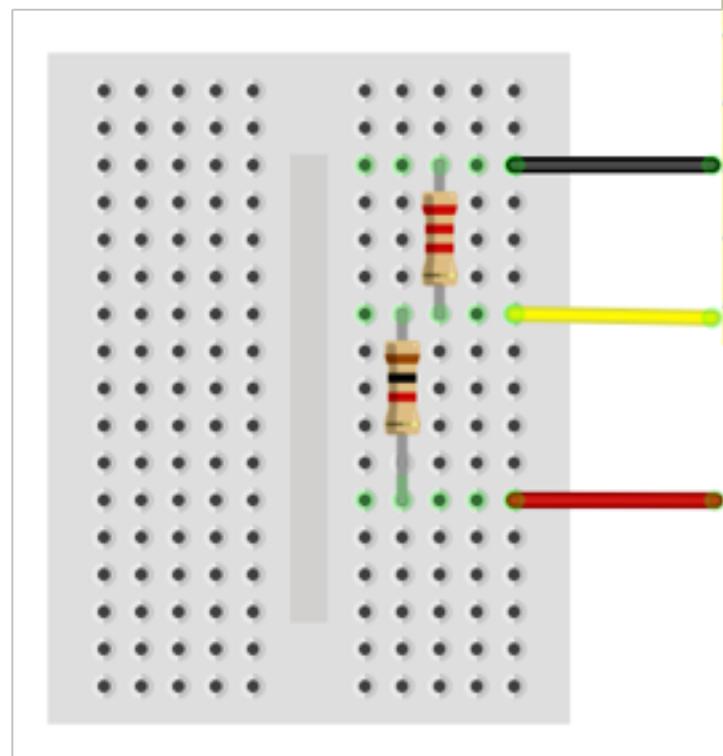
ESP8266

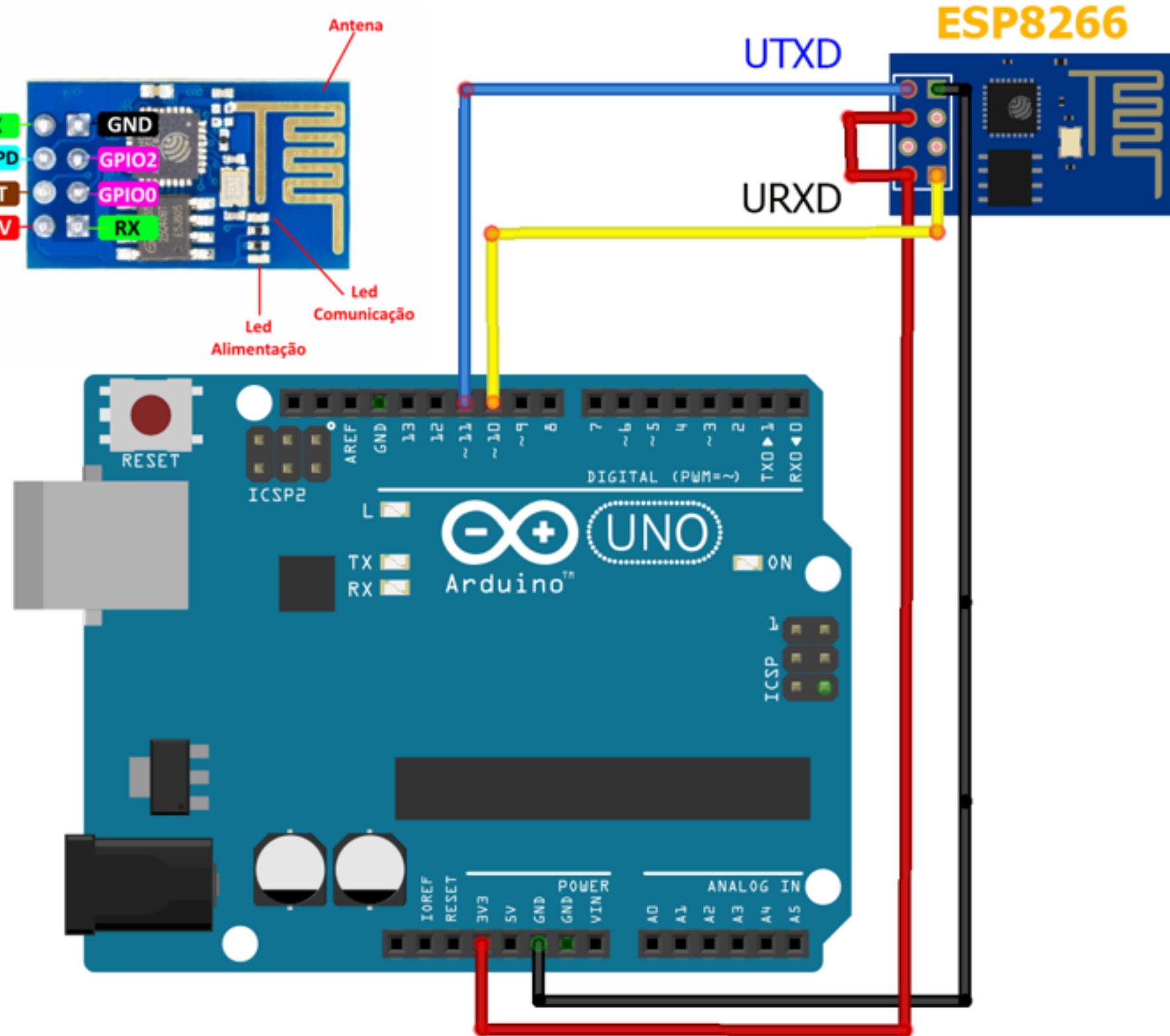
- ▶ CUIDADOS:
 - ▶ REDUTOR DE TENSÃO;
 - ▶ SOFTWARE SERIAL.



https://www.itead.cc/wiki/ESP8266_Serial_WIFI_Module

ESP8266





WiFiEsp by bportaluri Version 2.1.2 **INSTALLED**

Arduino WiFi library for ESP8266 Arduino WiFi library for ESP8266. Works only with SDK version 2.4.1 or later.

[More info](#)

WiFiManager by tzapu Version 0.12.0 **INSTALLED**

ESP8266 WiFi Connection manager with fallback web configuration portal Library for managing WiFi connections and providing a web-based configuration interface.

[More info](#)

PubSubClient by Nick O'Leary Version 2.6.0 **INSTALLED**

A client library for MQTT messaging. MQTT is a lightweight messaging protocol ideal for small devices to receive MQTT messages. It supports the latest MQTT 3.1.1 protocol and can be configured to use the Arduino Ethernet Client compatible hardware, including the Intel Galileo/Edison, ESP8266 and TI CC3200.

[More info](#)

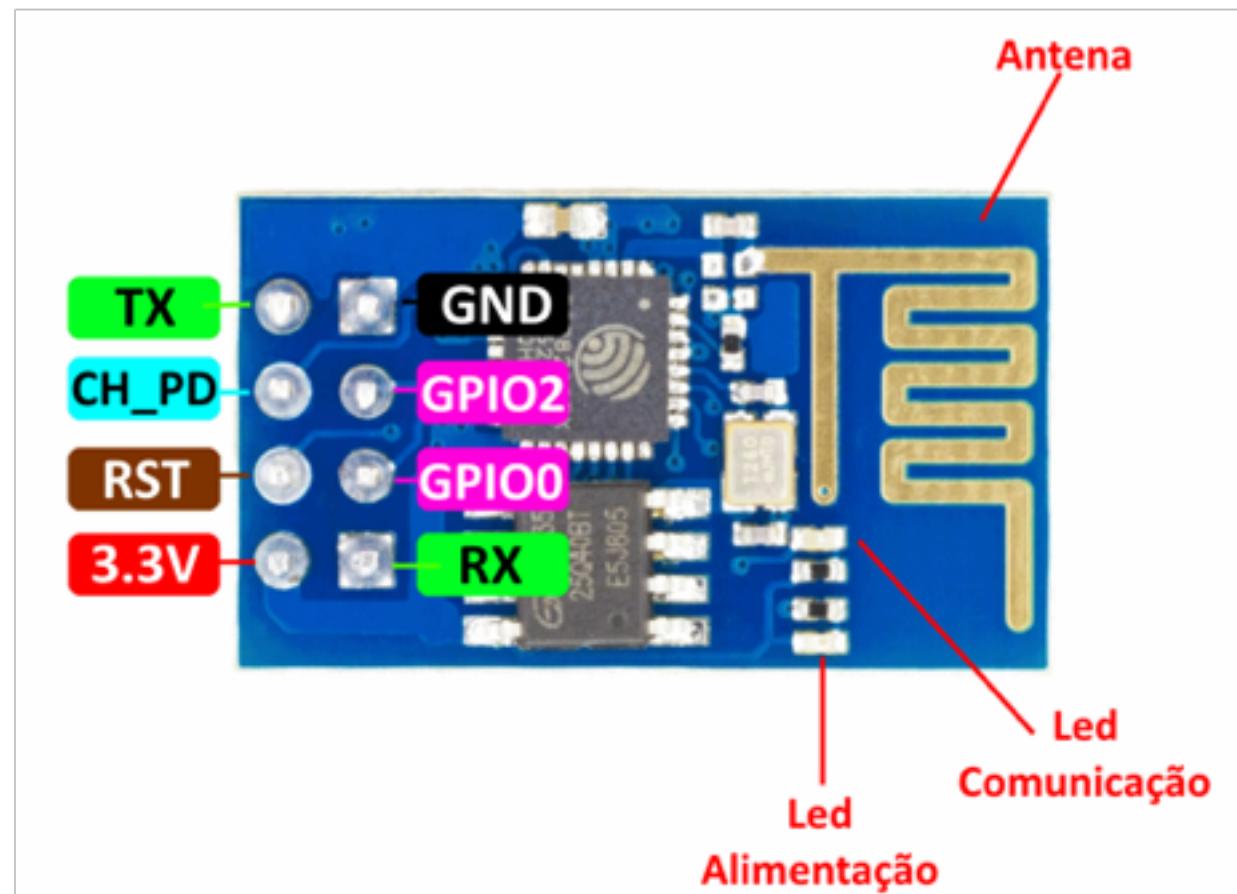
ConfigManager by Nick Wiersma Version 0.6.0 **INSTALLED**

ESP8266 WiFi connection manager Library for configuring ESP8266 modules WiFi credentials.

[More info](#)

ESP8266

- ▶ EXECUTAR EXEMPLOS:
 - ▶ CONNECT WPA
 - ▶ CLIENT WEB



https://www.itead.cc/wiki/ESP8266_Serial_WIFI_Module

ESP8266

- ▶ CONFIGURAÇÃO INICIAL:
 - ▶ CONEXÃO UTILIZANDO FTDI PARA ALTERAÇÃO DO MODO DE EXECUÇÃO: 1 - CLIENT; 2 - AP; 3 - AMBOS.
 - ▶ COMANDOS AT:
 - ▶ AT+GMR -> VERSÃO
 - ▶ AT+MODE? -> CONSULTA MODO DE OPERAÇÃO
 - ▶ AT+MODE=1 -> ALTERA MODO DE OPERAÇÃO
 - ▶ AT+CWLAP -> EM MODO CLIENT EXIBE REDES DISPONÍVEIS
 - ▶ AT+CWJAP -> CONECTA A UMA REDE
 - ▶ AT+CIUPDATE -> ATUALIZA VERSÃO
 - ▶ PROCURE POR OUTROS COMANDOS INTERESSANTES

CARACTERIZAÇÃO DO ESP8266

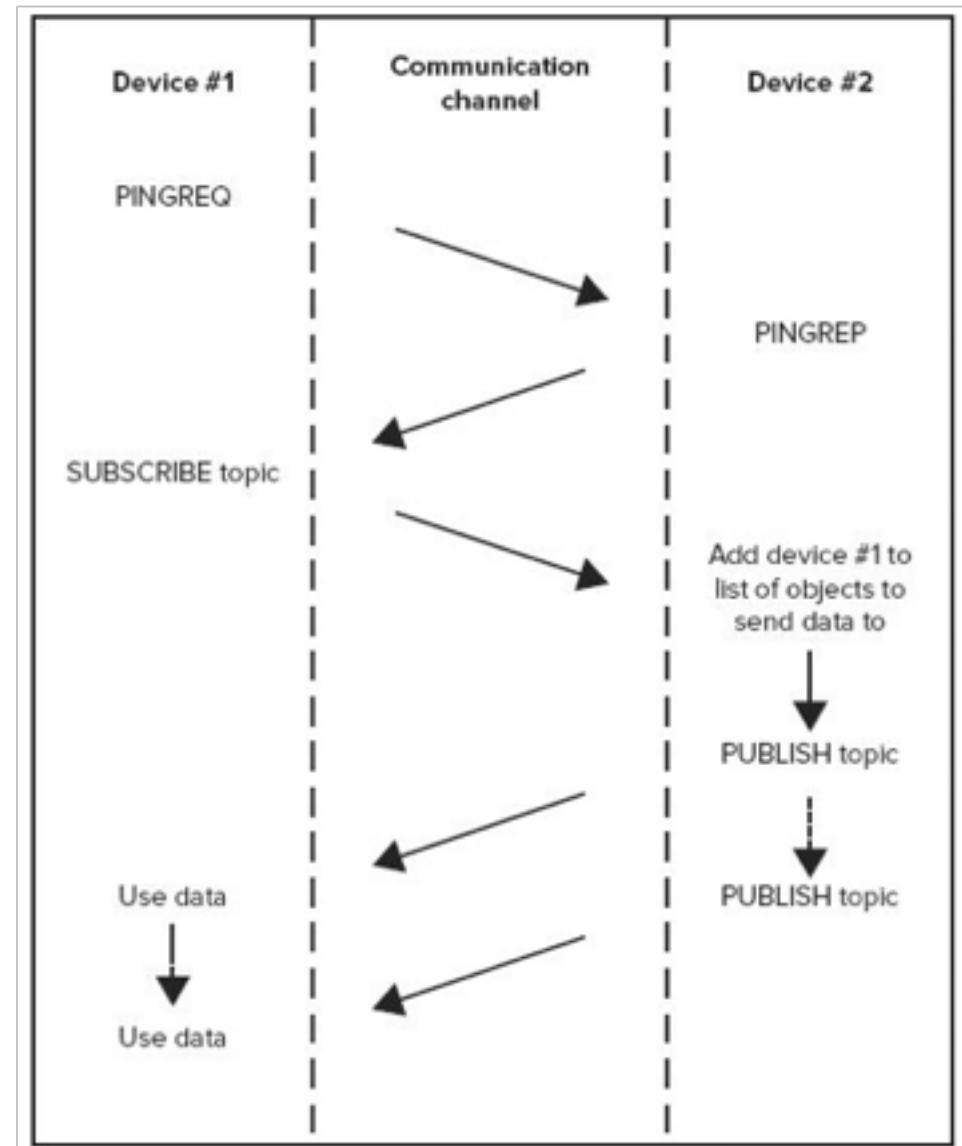
- ▶ <http://blog.filipeflop.com/wireless/esp8266-arduino-tutorial.html>
- ▶ <http://bbs.espressif.com/viewtopic.php?f=57&t=433>
- ▶ <http://blog.filipeflop.com/wireless/upgrade-de-firmware-do-modulo-esp8266.html>
- ▶ <https://github.com/esp8266/Arduino>
- ▶ <http://www.whatimade.today/esp8266-easiest-way-to-program-so-far/>
- ▶ <http://www.embarcados.com.br/modulo-esp8266/>

ARDUINO

MQTT

MQTT

- ▶ MQ TELEMETRY TRANSPORT
- ▶ M2M/IOT
- ▶ COMUNICAÇÃO
 - ▶ INFORMAÇÃO DE TRANSMISSOR
 - ▶ ENDEREÇO DO DESTINATÁRIO
 - ▶ MENSAGEM
 - ▶ BYTES DE PARIDADE



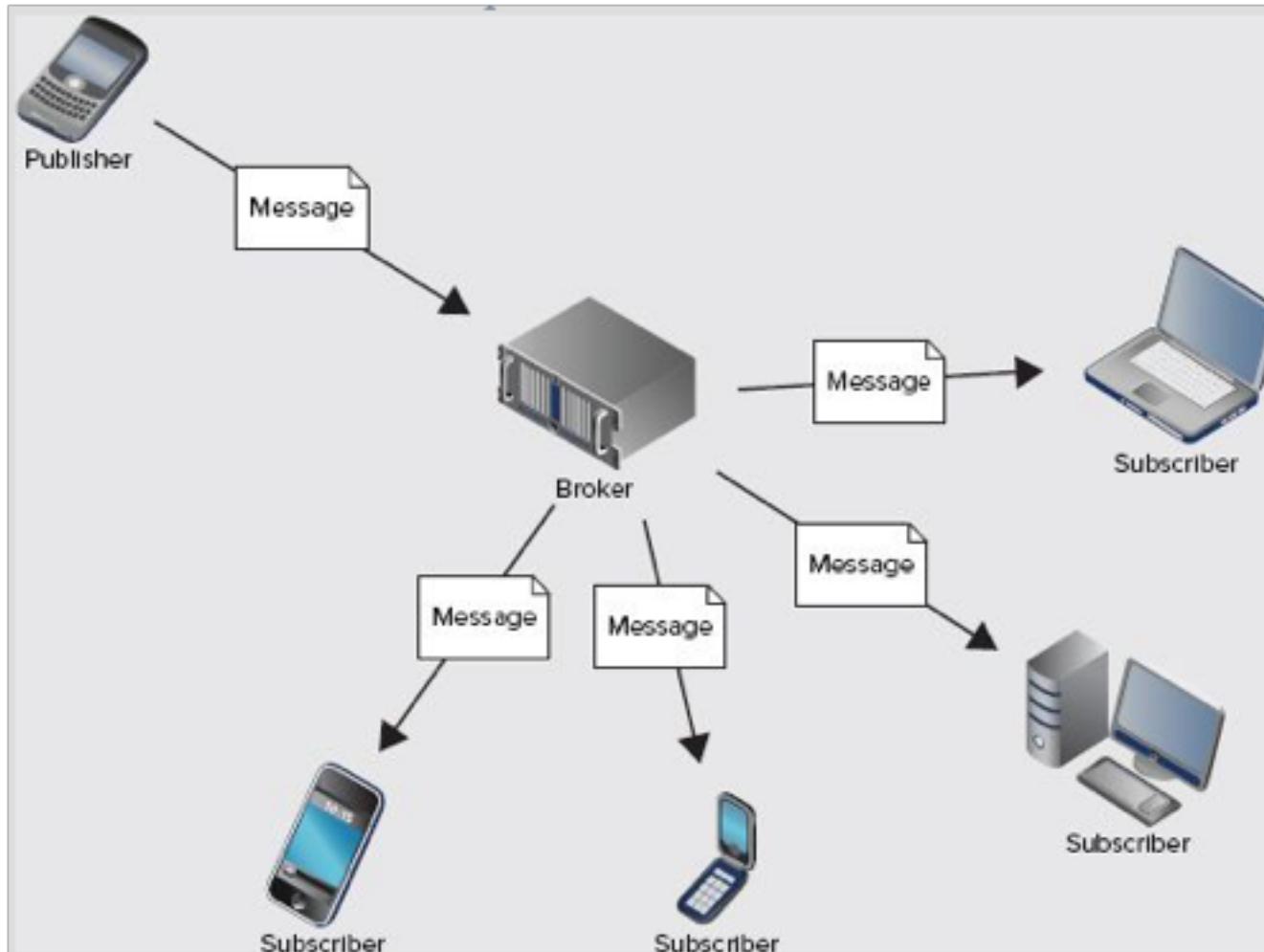
MQTT

- ▶ NOT HUMAN-READABLE
- ▶ MUITO EFICIENTE PARA COMUNICAÇÃO ENTRE MÁQUINAS
- ▶ EFICIENTE PARA PEQUENAS COMUNICAÇÕES
- ▶ PORTÁVEL PARA UM NUMEROZO TIPO DE DISPOSITIVOS

MQTT

- ▶ DATA
- ▶ HEADER
- ▶ PAYLOAD
- ▶ PACKAGE
- ▶ ACKNOWLEDGMENT
- ▶ PING
- ▶ MSB VS LSB
- ▶ FIXO VS VARIADO
- ▶ CRC
- ▶ ENCRYPTION

MQTT



- ▶ **PUBLISH/SUBSCRIBE**
- ▶ **ONE-TO-MANY**
- ▶ **BROKER**
- ▶ **TRANSMISSOR/
RECEPTOR**
- ▶ **ESCALÁVEL**
- ▶ **BASEADO EM TÓPICOS**

MQTT

- ▶ QUALITY OF SERVICE - QOS
- ▶ 0 - ENVIA UMA VEZ, SEM CONFIRMAÇÃO DE ENTREGA: AT MOST ONCE
- ▶ 1 - ENVIO ATÉ A CONFIRMAÇÃO DE ENTREGA: AT LAST ONCE
- ▶ 2 - ENVIO APENAS UMA VEZ, NEM MAIS, NEM MENOS: EXACTLY ONCE

MQTT

- ▶ **BROKERS**
 - ▶ [MQTT.ORG](#) -> SOFTWARE -> BROKERS/SERVERS
 - ▶ VEJA TAMBÉM CLIENT LIBRARIES

ARDUINO

MOSQUITTO

MOSQUITTO

- ▶ BROKER COM SUPORTE AO MQTT;
- ▶ SIMPLES INSTALAÇÃO;
- ▶ SSL;
- ▶ MULTIPLATAFORMA;
- ▶ PERSISTÊNCIA.



Mosquitto

An Open Source MQTT v3.1/v3.1.1 Broker

MOSQUITTO

- ▶ USO DE EXPRESSÕES REGULARES:

- ▶ #
- ▶ +
- ▶ \$SYS



MOSQUITTO

- ▶ SUBSCREVER PARA:
 - ▶ CASA/LAMPADA/+/STATUS
- ▶ PUBLICAÇÕES COMO:
 - ▶ CASA/LAMPADA/1
 - ▶ CASA/LAMPADA/2



Mosquitto

An Open Source MQTT v3.1/v3.1.1 Broker

BROKER

MOSQUITTO



Mosquitto

An Open Source MQTT v3.1/v3.1.1 Broker

- ▶ SUBSCREVER PARA:
 - ▶ CASA/#
- ▶ PUBLICAÇÕES COMO:
 - ▶ CASA/LAMPADA/1
 - ▶ CASA/LAMPADA/2
 - ▶ CASA/LAMPARA/2/STATUS
 - ▶ CASA/VENTILADOR/1

BROKER

MOSQUITTO



Mosquitto

An Open Source MQTT v3.1/v3.1.1 Broker

- ▶ \$SYS
 - ▶ GERENCIAMENTO
 - ▶ ESTATÍSTICA
 - ▶ MENSAGENS
 - ▶ BYTES
 - ▶ CLIENTES
 - ▶ NRO MENSAGENS DESCARTADAS
 - ▶ ...

SÃO CHAVES ÚNICAS:

MAC

IP

CLIENTID

ATENÇÃO

**PLUGIN CHROME:
MUTTLENS**

IP: 10.30.2.56 PORTA: 8883

ARDUINO

ANDROID

GERAL

- ▶ PUBLISH
 - ▶ OPERAÇÕES RÁPIDAS
 - ▶ BROADCAST
- ▶ SUBSCRIBE
 - ▶ OPERAÇÕES RÁPIDAS
 - ▶ BROADCAST
 - ▶ SERVIÇOS

PRINCÍPIOS

- ▶ SIMPLICIDADE VISANDO INTEGRAÇÃO;
- ▶ MENSAGENS QUE PERMITAM AGREGAR NOVOS COMPONENTES;
- ▶ MENOR GERENCIAMENTO POSSÍVEL;
- ▶ POUPAR REDE, MENSAGENS CURTAS;
- ▶ PREVER PROBLEMAS DE REDES;
- ▶ CONTINUIDADE DE SESSÃO;
- ▶ DISPOSITIVOS LIMITADOS;
- ▶ USAR QOS SEMPRE QUE POSSÍVEL;
- ▶ FLEXIBILIDADE NOS DADOS.

ARDUINO

GRANDE

```
defaultConfig {  
    //...  
    multiDexEnabled true  
    jackOptions {  
        enabled true  
    }  
}  
lintOptions {  
    abortOnError false  
}  
repositories {  
    maven {  
        url "https://repo.eclipse.org/content/repositories/paho-releases/"  
    }  
}  
dependencies {  
    //...  
    compile 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.0.2'  
    compile 'org.eclipse.paho:org.eclipse.paho.android.service:1.0.2'  
    //...  
}
```

ARDUINO

PUBLISH

PUBLISH

```
private static final String TAG = "MAIN";
String topic          = "casa/lampada/1";
String content        = "1";
int qos              = 1;
String broker         = "tcp://192.168.0.102:8883";
String clientId       = "ClientAndroidBasico";
String username        = "m";
String userPass        = "m";
MemoryPersistence persistence = new MemoryPersistence();
boolean lampadaStatus = false;
```

PUBLISH

```
try {
    MqttClient sampleClient = new MqttClient(broker,
        clientId, persistence);
    sampleClient.connect();
    MqttMessage message = new MqttMessage(content.getBytes());
    sampleClient.publish(topic, message);
} catch(MqttException me) {
    Log.d(TAG, "excep "+me);
    me.printStackTrace();
}
```

CONFIGURAÇÃO MÍNIMA

PUBLISH

```
MqttConnectOptions connOpts = new MqttConnectOptions();
connOpts.setUserName(username);
connOpts.setPassword(userPass.toCharArray());
connOpts.setCleanSession(true);
sampleClient.connect(connOpts);
```

AUTENTICAÇÃO

PUBLISH

```
<USES-PERMISSION ANDROID:NAME="ANDROID.PERMISSION.WAKE_LOCK" />
<USES-PERMISSION ANDROID:NAME="ANDROID.PERMISSION.INTERNET" />
<USES-PERMISSION ANDROID:NAME="ANDROID.PERMISSION.WRITE_EXTERNAL_STORAGE" />
<USES-PERMISSION ANDROID:NAME="ANDROID.PERMISSION.ACCESS_NETWORK_STATE" />

<USES-PERMISSION ANDROID:NAME="ANDROID.PERMISSION.READ_PHONE_STATE" />
<USES-PERMISSION ANDROID:NAME="ANDROID.PERMISSION.READ_EXTERNAL_STORAGE" />
```

PERMISSÕES

ARDUINO

SUBSCRIBE

SUBSCRIBE

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    new ClientSubscribe().Subscribe("casa/lampada/+status", 1);  
}
```

SUBSCRIBE

implements MqttCallback

```
@Override  
public void connectionLost(Throwable cause) {  
    Log.d(TAG, "Conexao perdida")  
}  
  
@Override  
public void messageArrived(String topic, MqttMessage message)  
    throws Exception {  
    Log.d(TAG, message.toString());  
}  
  
@Override  
public void deliveryComplete(IMqttDeliveryToken token) {  
    Log.d(TAG, "Mensagem completa");  
}
```

SUBSCRIBE

```
private static final String TAG = "Subscribe";
MqttClient client;
String broker      = "tcp://192.168.0.102:8883";
String clientId    = "ClientAndroidSubscribe";
String username     = "m";
String userPass     = "m";
MemoryPersistence persistence = new MemoryPersistence();
String topic = "";
int QoS = 1;

public ClientSubscribe() {}
```

```
public void Subscribe(String topic, int QoS)
{
    this.topic = topic;
    this.QoS = QoS;
    try {
        client = new MqttClient(broker, clientId, persistence);
        MqttConnectOptions connOpts = new MqttConnectOptions();
        connOpts.setUserName(username);
        connOpts.setPassword(userPass.toCharArray());
        connOpts.setCleanSession(true);

        client.connect(connOpts);
        Log.d(TAG, "Conectou");

        client.setCallback(this);
        Log.d(TAG, "Call back");

        client.subscribe(topic, QoS);
        Log.d(TAG, "Subscreveu");

    } catch (MqttException e) {
        e.printStackTrace();
    }
}
```

INTRODUÇÃO AO MOSQUITTO

- ▶ <https://mosquitto.org>
- ▶ <http://stackoverflow.com/questions/24556160/mosquitto-client-obtain-refused-connection>
- ▶ <http://jpmens.net/2014/07/03/the-mosquitto-mqtt-broker-gets-websockets-support/>
- ▶ <http://www.switchdoc.com/2016/02/tutorial-installing-and-testing-mosquitto-mqtt-on-raspberry-pi/>
- ▶ <https://www.digitalocean.com/community/questions/how-to-setup-a-mosquitto-mqtt-server-and-receive-data-from-owntracks>