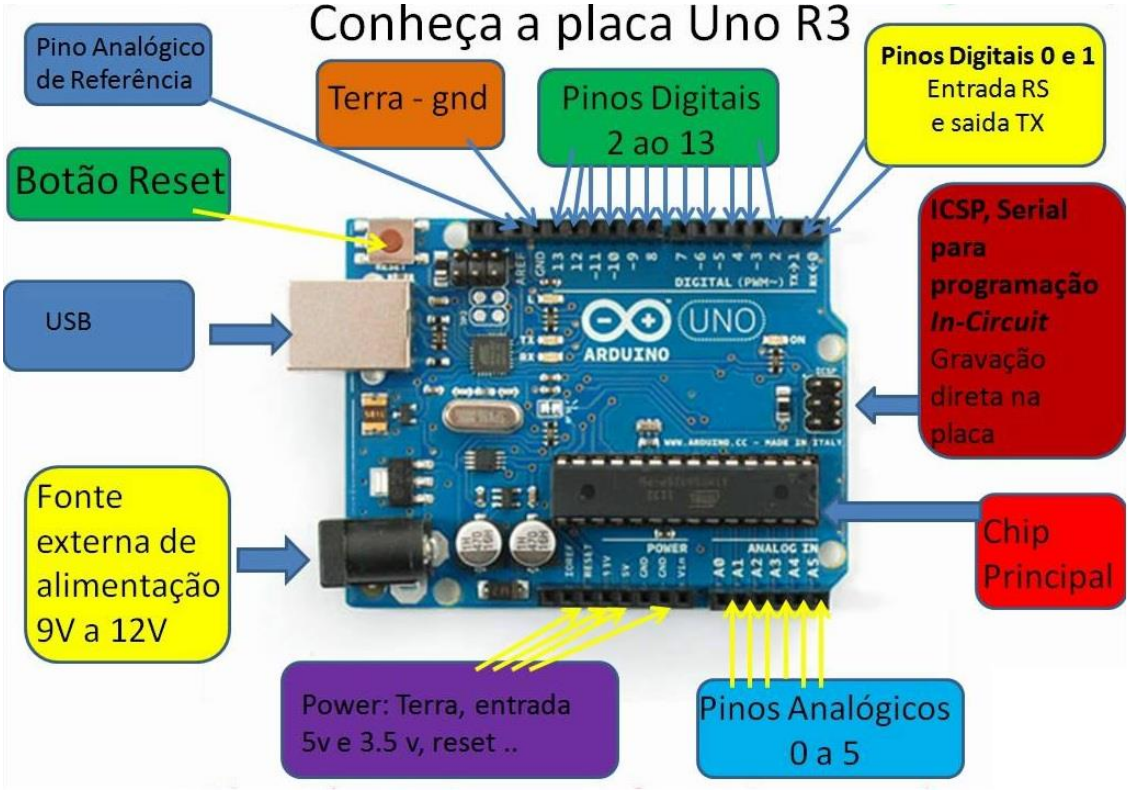
Curso Arduino

Módulo de treinamento

Índice:

1-Funções Básicas.



1.1-Como Funciona o Arduino?

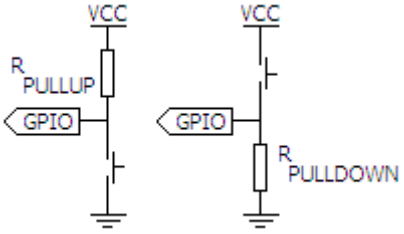
Vin possui a mesma tensão de entrada DC do Arduino, a saída 5V possui 5 volts independente da tensão de entrada de alimentação, pois vem do regulador de tensão.
As portas trabalham com tensões gerenciadas pela linguagem C.
Através da linguagem C, podemos controlar as diversas portas, o código inicial segue abaixo:

Aqui definimos as variáveis
void setup() {
Aqui iniciamos parâmetros e variáveis
}
void loop() {
Aqui colocamos o código que será monitorado através do loop de repetição.
}

1.2-A Função Input

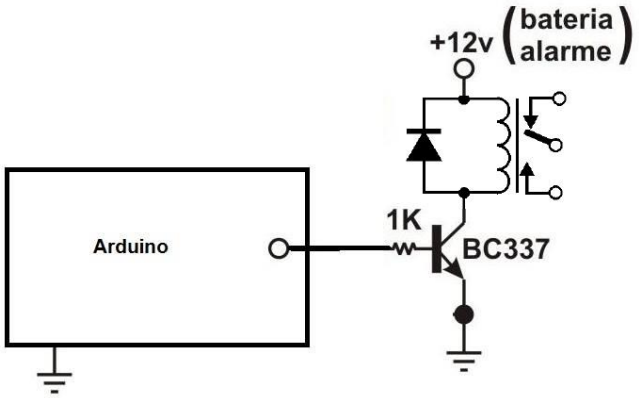
Utiliza as portas digitais do Arduino para entrada de estado Lógico 0 ou 5V, se preferir trabalhar com tensões diferentes de 0 ou 5 utilize as portas analógicas.
Utilize as portas de 2 a 13 para esta função, comando :
Define a porta 2
int led = 2;
A porta 2 será utilizada como entrada de dados:
pinMode(led, INPUT);

informa que a variável led será para saída de dados.
Um problema comum, se colocado uma chave devemos usar um resistor para negativar ou positivar a porta e evitar acionamento aleatório, segue figura abaixo:



1.3-A função Output
Utiliza as portas digitais do Arduino para saída de tensão, 0 ou 5v são gerados na saída do arduino, se quiser tensões diferente de 0 e 5v, utilize a função PWM nas portas especificas.

Um problema comum é querer alimentar cargas altas na saída destas portas, como motores e relés, o correto é utilizar transistor para chavear cargas maiores, como segue figura abaixo:



A Corrente máxima de saída do arduino Uno é de 50mA

1.4-Low e High
Estes comandos setam a saída e entrada do Arduino para nível alto ou baixo:
Envia sinal alto para a variavel led, se led for definida como porta 2 acenderá 5V.
digitalWrite(led, HIGH);

Envia sinal baixo para a variavel led, se led for definida como porta 2 apagará, 0V .
digitalWrite(led, LOW);

buttonState = digitalRead(buttonPin);

A Variável buttonState irá enviar seu status para a porta definida de acordo com o resistor de negativação ou positivação e o código descrito,.

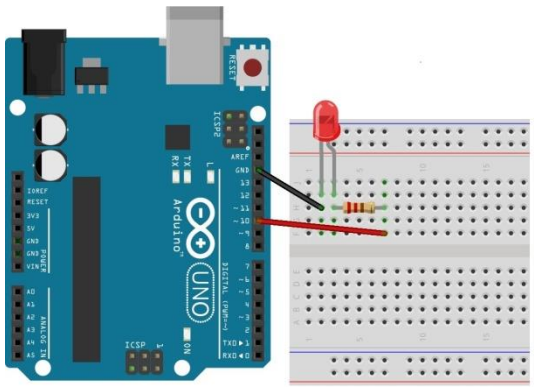
Exemplos de códigos :

1.5-Exercicios

1-Faça um led acender utilizando a alimentação do Arduino(USB)

2-Faça um led Piscar na porta 10

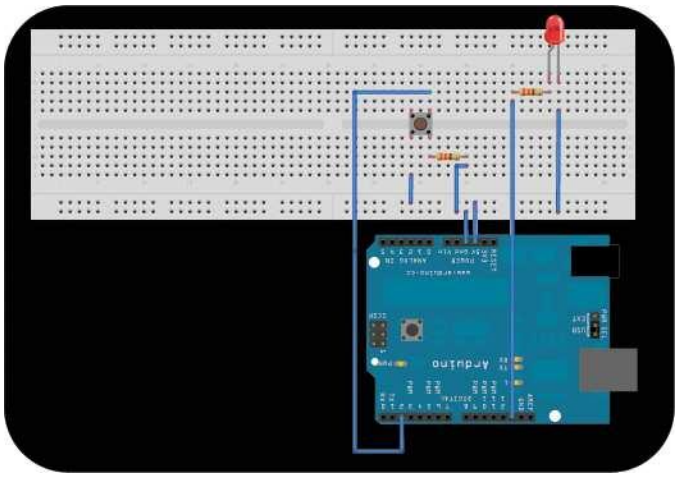
```
int led = 10;
void setup() {
  pinMode(led, OUTPUT);
}
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```



3-Faça um led acender na porta 3 quando pressionar um botão na porta 2

```
const int buttonPin = 2;
const int ledPin =  3;
int buttonState = 0;
void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}
void loop(){
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    digitalWrite(ledPin, HIGH);
  }
  else {
    digitalWrite(ledPin, LOW);
  }
```

}



4-Faça um led acender na porta 4,pressionando um botão na porta 2 e quando pressionar o botão novamente o led apaga.

```
const int buttonPin = 2;
const int ledPin =  4;
int st=0;
int buttonState = 0;
void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}
void loop(){
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
     if (st==0){
       delay(200);
        digitalWrite(ledPin, LOW);
     st=1;
     }else{
       delay(200);
       digitalWrite(ledPin, HIGH);
      st=0;
    }
  }
 }
```

1.6-O Problema do Delay na Programação.

O Delay para toda a sequência de programação durante o tempo determinado em seu comando, tal característica torna-se um problema quando temos que monitorar vários sensores ao mesmo tempo, pois se colocarmos 20 segundos em um dos sensores e o mesmo disparar, só poderemos interromper seu acionamento após o código retomar sua sequencia, imagine um alarme residencial programado para em caso de acionamento o alarme tocar por 5 minutos, se eu quiser desarmar o alarme não poderei até que os 5 minutos termine e o código retome a sua sequencia, a solução para este problema é utilizar o Millis no lugar do delay.

Exemplo

Faça um led piscar na porta 2 usando Millis.

```
const int ledPin =  2;
int ledState = LOW;
long previousMillis = 0;
long interval = 1000;
void setup() {
  pinMode(ledPin, OUTPUT);
}
void loop()
{
  unsigned long currentMillis = millis();
  if(currentMillis - previousMillis > interval) {
  previousMillis = currentMillis;
    if (ledState == LOW)
    ledState = HIGH;
    else
     ledState = LOW;
    digitalWrite(ledPin, ledState);
  }
}
```

1.7-Exercicio em Sala:

1-Faça um circuito monoestavel com arduino.
2-Faça um circuito biestável com Arduino.
3-Faça um circuito astavél com arduino.
4-Explique o problema que existe no delay para a programação com Arduino.
5-Faça um led piscar sem usar o millis e sem usar o Delay.

2-Funções avançadas e Monitoramento

2.1-PWM

Assistir Video Sobre PWM em
https://www.youtube.com/watch?v=Gxdrs1BAVoo

Trata-se de conseguir variar a tensão de saída digital do arduino utilizando **modulação por largura de pulso**, muito utilizado para controle de potência de motores, esta função está disponível apenas nas portas identificadas com um ~ (til), veja o funcionamento utilizando o código abaixo:

```
int led=3;
int  fadeValue;
void setup() {
}
void loop() {
for(int fadeValue=0;fadeValue<=255; fadeValue +=5){
analogWrite(led,fadeValue);
 delay(30);
}
for(int fadeValue=255;fadeValue>0; fadeValue -=5){
analogWrite(led,fadeValue);
delay(30);
}
}
```

2.2-Portas Analógicas
AnalogRead() e AnalogWrite()
Utiliza as portas de A0 a A4

Caracteristicas:
Tensão máxima de entrada na porta 5V, que corresponde ao numero 1023 do conversor AD interno do Arduino, se desejar colocar tensões maiores utilize **resistores divisores de tensão.**

Assistir video sobre portas analogicas em:
https://www.youtube.com/watch?v=IdF-1n90NZA

Exemplo:

1-Controle o brilho de um led na porta 3 através da variação de um potenciômetro na entrada analógica A0.

```
int Led=3;
int Valor;
void setup() {
}
```

```
void loop() {
  int recebeValor=analogRead(A0);
  delay(200);
   Valor=(recebeValor/5);
   analogWrite(Led,Valor);
}
```

2.3-Comunicação Serial

É utilizado para monitorar estados e valores, acionamentos via WEB, Bluetooth e outros, vamos monitorar o programa anterior utilizando o Monitor Serial.

Basta iniciar a porta Serial :
Serial.begin(9600);

Depois mandar a informação para a porta:
Serial.println(recebeValor);

Vejamos:

```
int Led=3;
int Valor;
void setup() {
Serial.begin(9600);
delay(1000);
}
void loop() {
  int recebeValor=analogRead(A0);
  Serial.println(recebeValor);
  delay(200);
   Valor=(recebeValor/5);
   analogWrite(Led,Valor);
}
```

2.4 - Exemplos

Interagindo com o Serial Monitor.

Vamos fazer com que ao digitar a letra “A” o arduino acenda o led na porta 3 e ao digitar a letra “B” apague o led.

```
void setup() {
Serial.begin(9600);
pinMode(3, OUTPUT);
```

```
}
void loop() {
char c = Serial.read();
if (c=='A') {
digitalWrite(3, HIGH);
}
if (c=='B') {
digitalWrite(3, LOW);
}
}
```

Acionando e apagando o led pela WEB usando a porta Serial.

Necessário:
Seu computador configurado como servidor WEB local e código PHP abaixo na página:

```
<?
if (getenv("REQUEST_METHOD") == "POST") {
$a=$_POST[luz];
if ($a=='1'){
$port = fopen('COM3', 'w');
fwrite($port, 'AAAA');
sleep(1);
fclose($port);
echo "<br><Br><center><b>Led ligado</b></center>";
}
if ($a=='2'){
$port = fopen('COM3', 'w');
fwrite($port, 'BBB');
sleep(1);
fclose($port);
echo "<br><Br><center><b>Led Desligado</b></center>";
}
}
?>
<br>
<center><table>
<form name="salvar" action="serial2.php" method="POST" onSubmit="return
validar(this)">
<tr><td  bgcolor="#36b3b3" ><FONT color="white"><b> Luz:
 </b></FONT></td><td  bgcolor="#36b3b3" >
<SELECT NAME="luz" class="fieldStyle" onkeypress="return tabE(this,event)">
<option value="1" >Liga
```

```
<option value="2" >Desliga
</select></td>
<tr><td  bgcolor="#36b3b3" ></td><td  bgcolor="#36b3b3" ><center><input
class="art-button" type="submit"  value="Registra" /></center></td>
</form>
</table></center>
<br><br>
```

Através do Bluetooth veremos em **Shields.**

2.5-Exercícios Propostos

1-faça 3 leds acenderem na sequencia e 1 led aumentando e diminuindo o brilho.
2-faça com que um led acenda ao aquecer um diodo.
3-Faça um jogo de desarme de bomba, onde 3 fios corta um ok, os outros dois acende um led, que o fio correto alterne automaticamente entre eles.

3-Buzzers e Speakers

Existem basicamente duas formas do Arduino emitir som diretamente no Speaker sem a utilização de Shields, Esta função é importante para saber se o comando foi econhecido, ou diagnosticar alertas e avisos.

3.1- Tone (Tom)

```
#include "pitches.h"
// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_G3,NOTE_G3, NOTE_A3, NOTE_G3,0, NOTE_B3, NOTE_C4};
// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4,4,4,4,4 };
void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0; thisNote < 8; thisNote++) {
    // to calculate the note duration, take one second
```

```
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000/noteDurations[thisNote];
    tone(8, melody[thisNote],noteDuration);
    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    // stop the tone playing:
    noTone(8);
  }
}
void loop() {
  // no need to repeat the melody.
}
```

3.2-Tom Múltiplo

```
void setup() {
}
void loop() {
   // turn off tone function for pin 11:
   noTone(11);
  // play a note on pin 6 for 200 ms:
  tone(6, 440, 200);
  delay(200);
  // turn off tone function for pin 6:
  noTone(6);
  // play a note on pin 7 for 500 ms:
  tone(7, 494, 500);
  delay(500);
    // turn off tone function for pin 7:
  noTone(7);
  // play a note on pin 11 for 500 ms:
  tone(11, 523, 300);
  delay(300);
}
```

3.3-Beep

```
void setup() {
beep(100);
}
```

```
void loop() {
}
 void beep(unsigned char delayms){
     analogWrite(11, 20);      // Almost any value can be used except 0 and 255
     delay(delayms);          // wait for a delayms ms
     analogWrite(11, 0);       // 0 turns it off
     delay(delayms);          // wait for a delayms ms
}
```

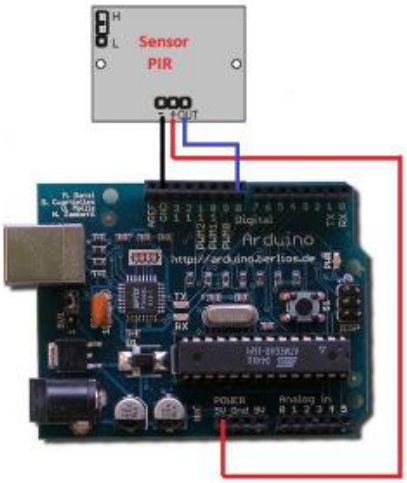
4-Shields

Shields, são acessórios para conexão ao arduino aumentando seu poder de controle e gerenciamento.

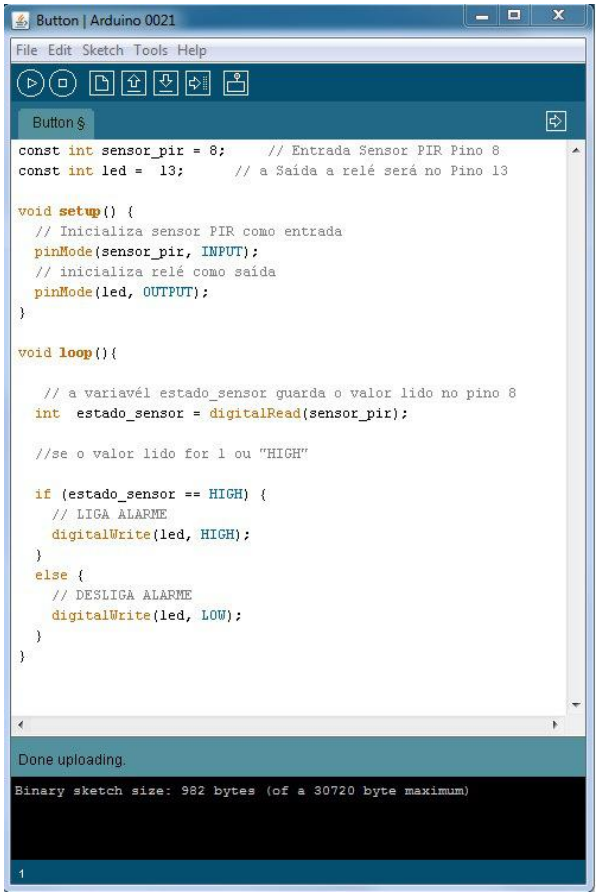
Para utilizar cada shield, faz-se necessário o download da biblioteca e cópia para a pasta **libraries** dentro da pasta do programa de sketch do arduino.

4.1-PIR:

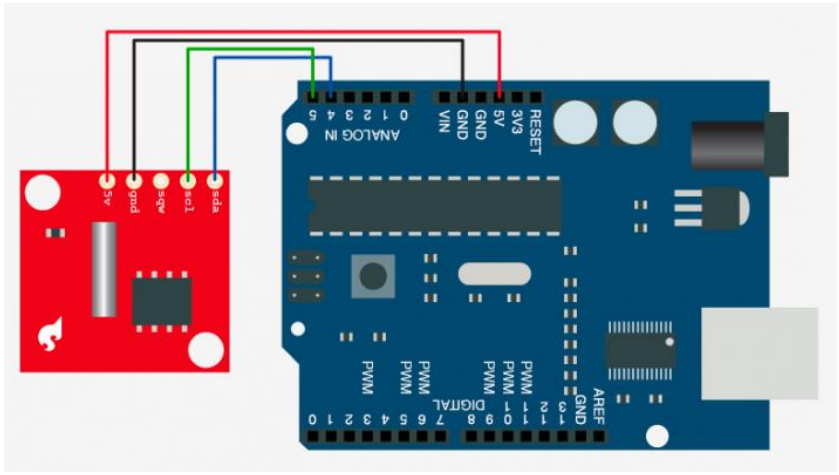
Esquema de ligação:



Sistema simples de uma entrada uma saída.



4.2-RTC:



RTC Module	Arduino
Vcc	+3.3V or +5V (As you need)
GND	GND
SDA	SDA (Analog Pin 4)
SCL	SCL (Analog Pin 5)

Código:

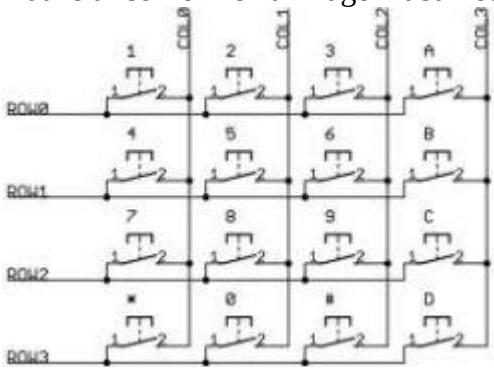
// Date and time functions using a DS1307 RTC connected via I2C and Wire lib

```
#include <Wire.h>
#include "RTClib.h"
RTC_DS1307 rtc;
void setup () {
  Serial.begin(9600);
#ifdef AVR
  Wire.begin();
#else
  Wire1.begin(); // Shield I2C pins connect to alt I2C bus on Arduino Due
#endif
  rtc.begin();
  //rtc.adjust(DateTime("JUL 28 2013","13:40:45"));
    if (! rtc.isrunning()) {
    Serial.println("RTC is NOT running!");
    // following line sets the RTC to the date & time this sketch was compiled
    rtc.adjust(DateTime(__DATE__, __TIME__));
     }
}
void loop () {
    DateTime now = rtc.now();
    Serial.print(now.day(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.year(), DEC);
    Serial.print(' ');
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.println();
    Serial.println();
    delay(3000);
}
```

4.3-Teclado

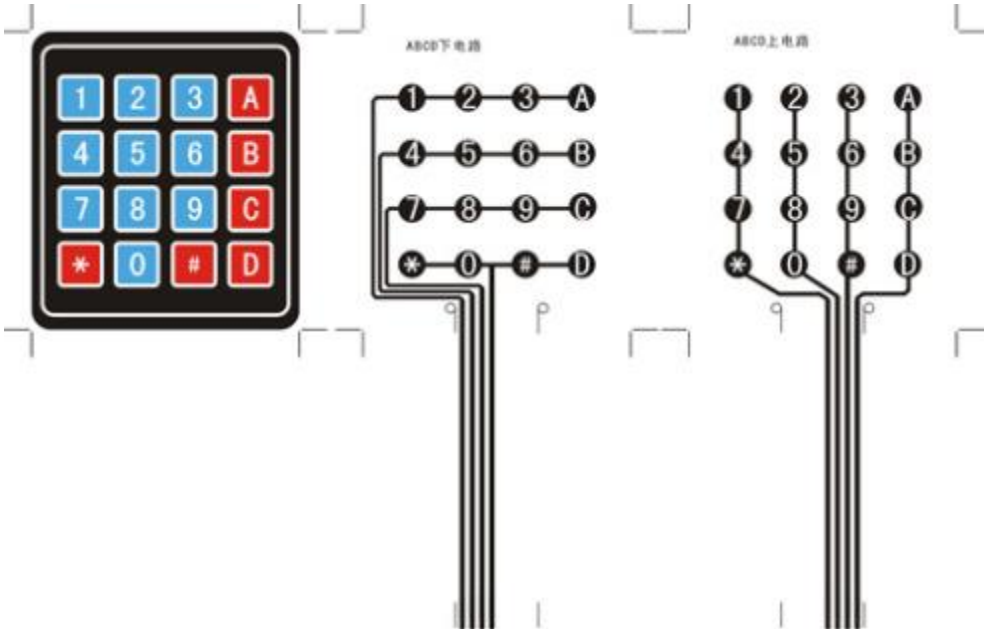
Teclado Matricial Membrana 4×4

O que é um teclado matricial?
O teclado matricial é um periférico de entrada de dados com os botões organizados em linhas e colunas, conforme você aperta uma tecla, você cria um curto entre o pino referente a sua linha e sua coluna, internamente são push-buttons organizados de forma matricial conforme na imagem abaixo:

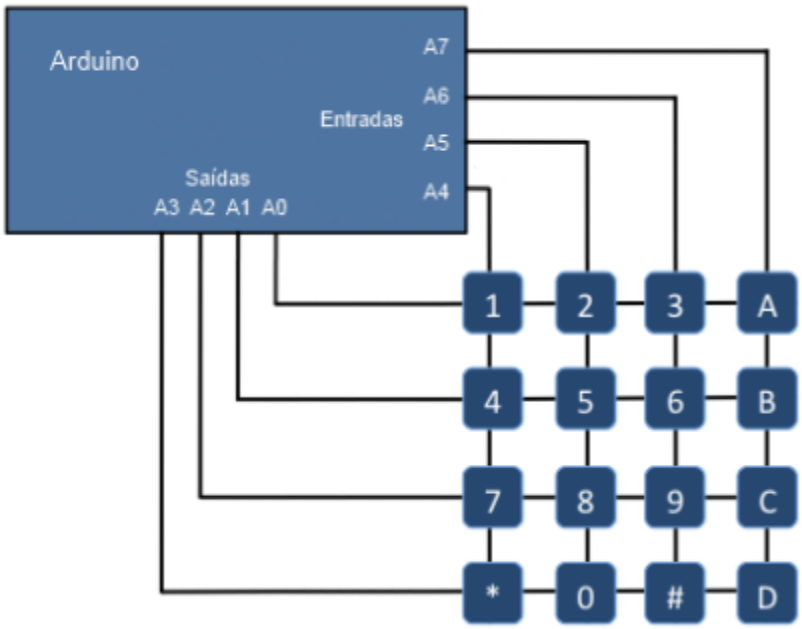


O nosso Teclado Matricial Membrana nada mais é do que um teclado matricial muito fino e com uma cola adesiva na parte posterior para fixar em alguma máquia, painel ou em qualquer lugar.

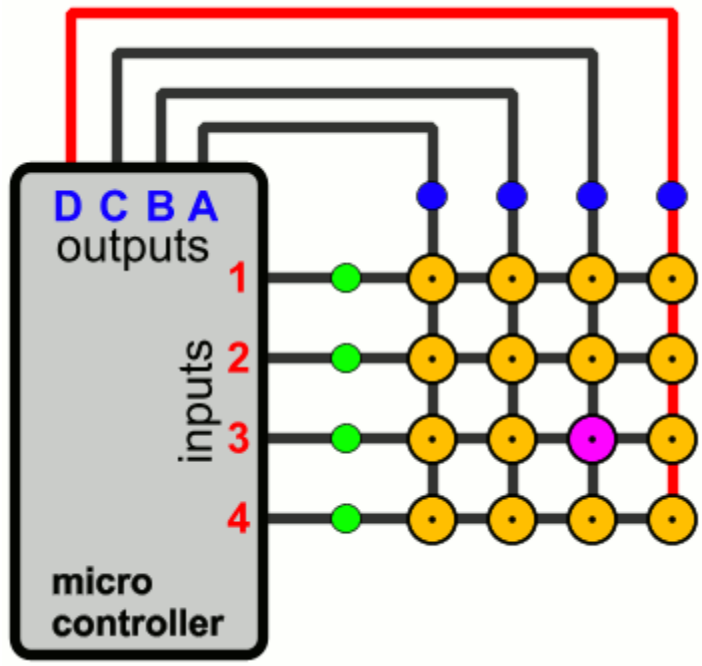
Como identificar as teclas?
É fácil, veja na imagem abaixo que o pino 1 do conector (pino mais a esquerda) refere-se a primeira linha e o oitavo pino do conector (mais a direita) refere-se a quarta coluna. Assim sendo se você apertar a tecla “A” haverá um curto entre os pinos 1 e 8 do conector. Se não houver nenhuma tecla apertada não há nenhum curto entre os terminais.



Para que o Arduino identifique as teclas existem algumas maneiras diferentes de conectar, uma delas é ligar todos os terminais das linhas em *saídas* do Arduino e ligar os terminais das colunas em*entradas*. Assim sendo teremos 4 saídas (A0, A1, A2 e A3) referentes às linhas e 4 entradas referentes às colunas (A4, A5, A6 e A7).



Desta forma, quando você apertar a tecla “A” a primeira saída do Arduino (A0) irá estar em curto com a quarta entrada do Arduino (A7), se você apertar a tecla ’8′ a terceira saída do Arduino (A2) estará conectada a segunda entrada do Arduino (A5).
Para efetuar a leitura da tecla primeiramente setamos individualmente as saídas referentes às linhas em nível alto intermitentemente até que alguma mudança ocorra. Se nenhuma tecla for apertada todas as entradas referentes às colunas lerão nível baixo pois não haverá nenhum curto jogando o nível alto das linhas nas colunas. A partir do momento que uma entrada referente a uma coluna ler nível alto sabemos que uma tecla foi apertada. A animação a seguir exemplifica isso:



A partir daí você vai trabalhar esta informação e pode, por exemplo, enviar este caractere ao display conforme nosso post específico: Controlando um LCD 16×2. Esperamos que tenhamos ajudado um pouco no entendimento do funcionamento do Teclado.
Não esqueça de visitar nossa loja e conferir nossa variedades de produtos dedicados a projetos eletrônicos e à família Arduino: www.filipeflop.com!

Pinos:

Teclado com o Cabo para baixo:

Teclado	/	Arduino
1		2
2		3
3		4
4		5
5		6
6		7
7		8

Programação

```
/* @file CustomKeypad.pde
|| @version 1.0
|| @author Alexander Brevig
|| @contact alexanderbrevig@gmail.com
||
|| @description
|| | Demonstrates changing the keypad size and key values.
|| #
*/
#include <Keypad.h>

const byte ROWS = 4; //four rows
```

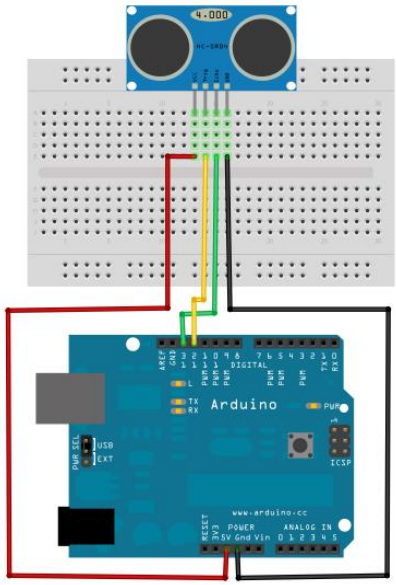
```
const byte COLS = 3; //four columns
//define the cymbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] = {
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'*','0','#'}
};
byte rowPins[ROWS] = {2, 3, 4, 5}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {6, 7, 8}; //connect to the column pinouts of the keypad

//initialize an instance of class NewKeypad
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS,
COLS);

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

void loop(){
  char customKey = customKeypad.getKey();

  if (customKey){
    Serial.println(customKey);
  }
}
```

4.4-Ultrassom

Hoje vamos utilizar em nosso projeto o sensor ultrasônico, ele funciona como um detector de objetos e pode chegar a medir distancias máximas de até 5 metros com uma precisão de 3 milímetros.
O que vamos utilizar no projeto:
- Arduino Uno
- Protoboard Mini
- Sensor Ultrasônico HC-SR04

Vamos entender um pouco mais sobre o Sensor Ultrasônico
O sinal de retorno é captado, permitindo-se deduzir a distância do objeto ao sensor tomando o tempo de trânsito do sinal.
A velocidade do sinal ultrasônico é de aproximadamente 340 m/s, assim, se o sensor estiver a uma distância d do objeto, o sinal percorrerá uma distância equivalente a 2d para sair e retornar ao sensor.

O sensor é composto por 4 pinos, sendo eles:
1.VCC
2.trig(T)
3.echo(R)
4.GND

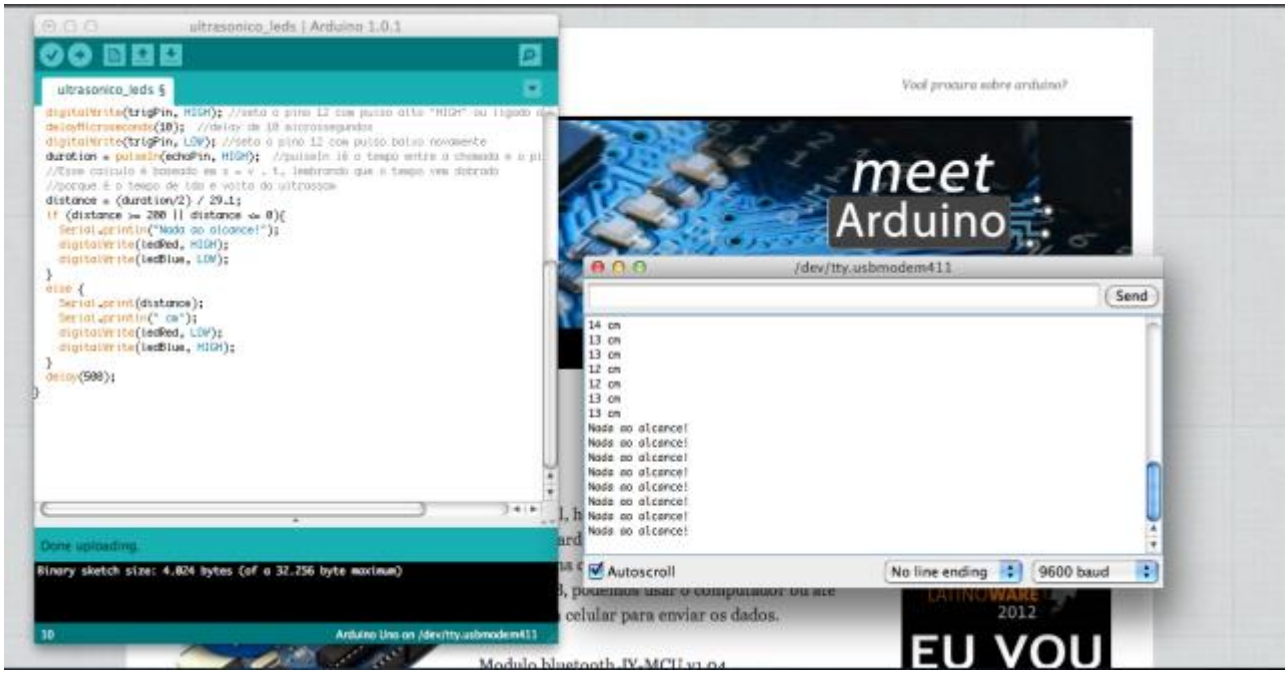


Código:

```
1       #include <Ultrasonic.h>
2
3       // funcão do ultrasonic nos pinos 12 e 13
4       Ultrasonic ultrasonic(12,13);
5
6       void setup()
7       {
8       Serial.begin(9600); //inicia a porta serial
9       pinMode(echoPin, INPUT); // seta pino como entrada
10      pinMode(trigPin, OUTPUT); // seta pino saída
11      }
12
13      void loop()
14      {
15      digitalWrite(trigPin, LOW);
16      delayMicroseconds(2);
17      digitalWrite(trigPin, HIGH);
18      delayMicroseconds(10);
19      digitalWrite(trigPin, LOW);
20      // A função Ranging converte o tempo de resposta em centímetros e seta na variável distancia
21      int distancia = (ultrasonic.Ranging(CM));
22
23      Serial.print("Distancia em CM: ");
24      Serial.println(distancia);
25      delay(1000); //espera 1 segundo para fazer a leitura novamente
26      }
```

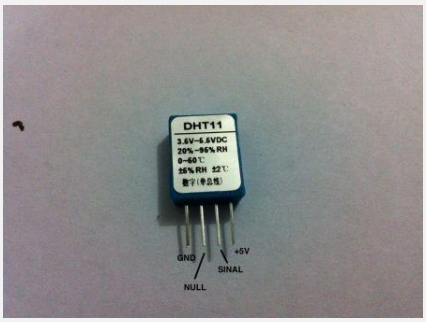
Outra forma sem utilizar a biblioteca Ultrasonic.h:

```
1   #define trigPin 13
2   #define echoPin 12
3   #define ledRed 8
4   #define ledBlue 7
5
6   void setup() {
7     Serial.begin (9600);
8     pinMode(trigPin, OUTPUT);
9     pinMode(echoPin, INPUT);
10    pinMode(ledRed, OUTPUT);
11    pinMode(ledBlue, OUTPUT);
12  }
13
14  void loop() {
15    long duration, distance; //http://arduino.cc/en/Reference/Long
16    digitalWrite(trigPin, LOW);  //seta o pino 12 com um pulso baixo "LOW" ou desligado ou ainda
17    delayMicroseconds(2); // delay de 2 microssegundos
18
19    digitalWrite(trigPin, HIGH); //seta o pino 12 com pulso alto "HIGH" ou ligado ou ainda 1
20    delayMicroseconds(10);  //delay de 10 microssegundos
21    digitalWrite(trigPin, LOW); //seta o pino 12 com pulso baixo novamente
22    duration = pulseIn(echoPin, HIGH);  //pulseIn lê o tempo entre a chamada e o pino entrar em hig
23    //Esse calculo é baseado em s = v . t, lembrando que o tempo vem dobrado
24    //porque é o tempo de ida e volta do ultrassom
25    distance = (duration/2) / 29.1;
26    if (distance >= 200 || distance <= 0){
27      Serial.println("Nada ao alcance!");
28      digitalWrite(ledRed, HIGH);
29      digitalWrite(ledBlue, LOW);
30    }
31    else {
32      Serial.print(distance);
33      Serial.println(" cm");
34      digitalWrite(ledRed, LOW);
35      digitalWrite(ledBlue, HIGH);
36    }
37    delay(500);
38  }
```

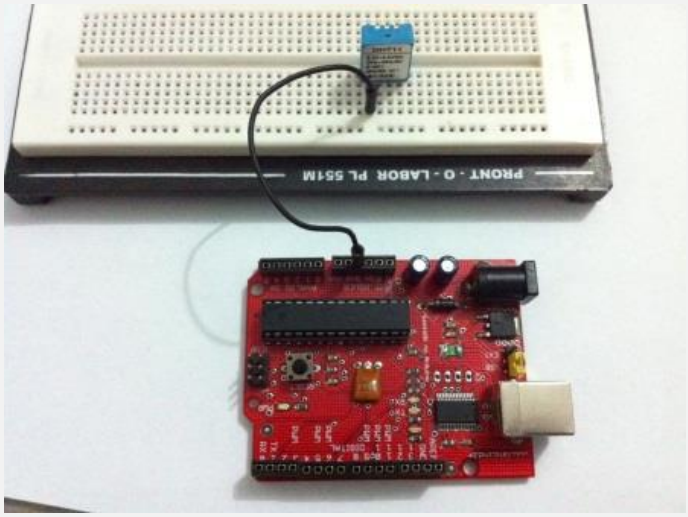


4.5-DHT

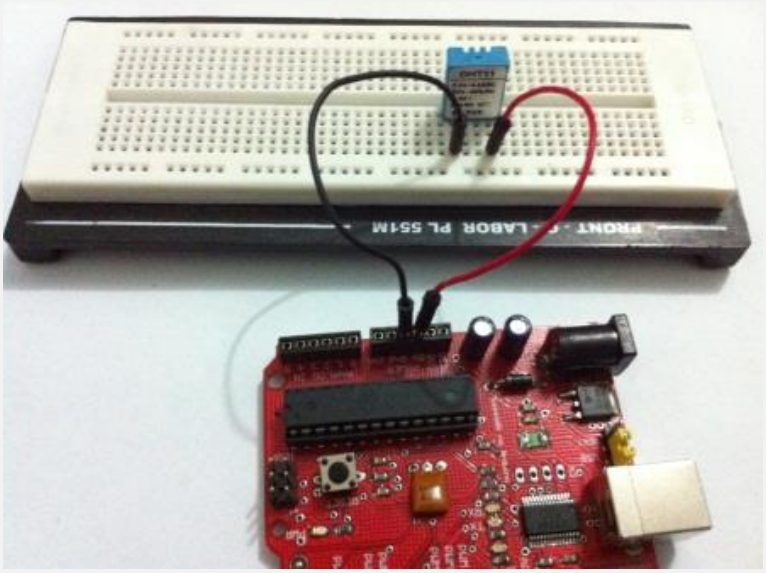
Basta o experimento será necessário, obviamente um Arduino, uma placa de contatos (prot-o-board), um resistor de 10K ohm e pelo menos 3 fios para ligações.



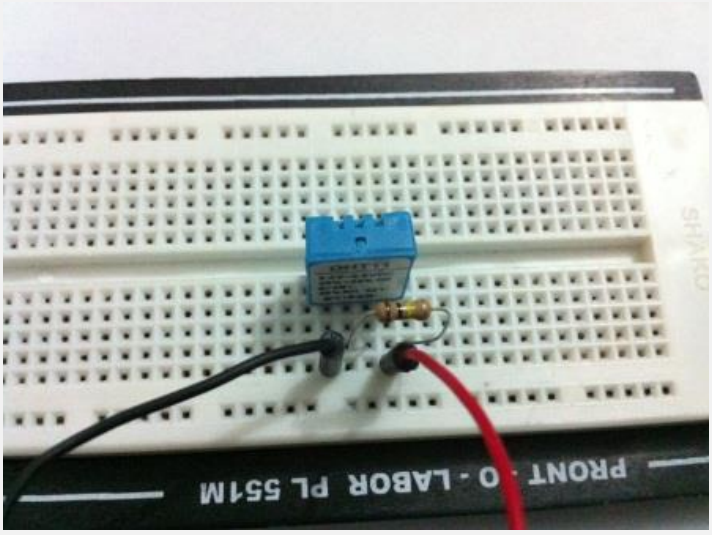
Ligue o pino GND do sensor ao GND do Arduino:



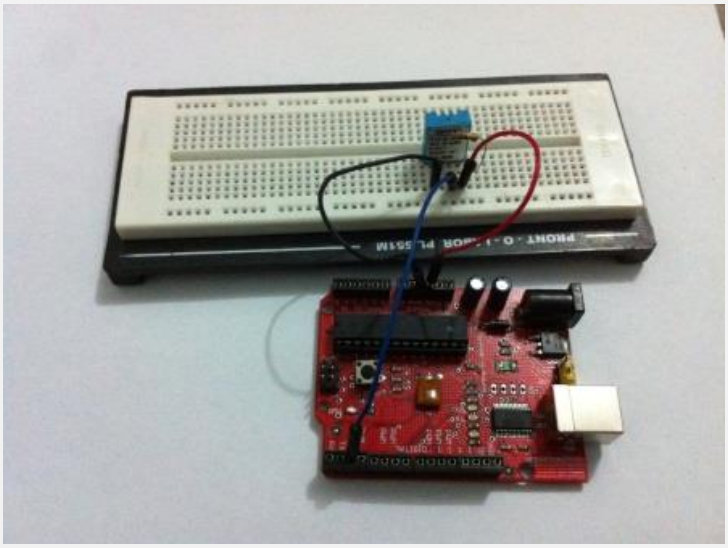
Ligue o pino +5V do sensor ao 5V do Arduino:



Ligue o resistor de 10K ohm entre os pinos +5V e SINAL do sensor:



Agora ligue o pino SINAL do sensor ao pino digital 2 do Arduino:



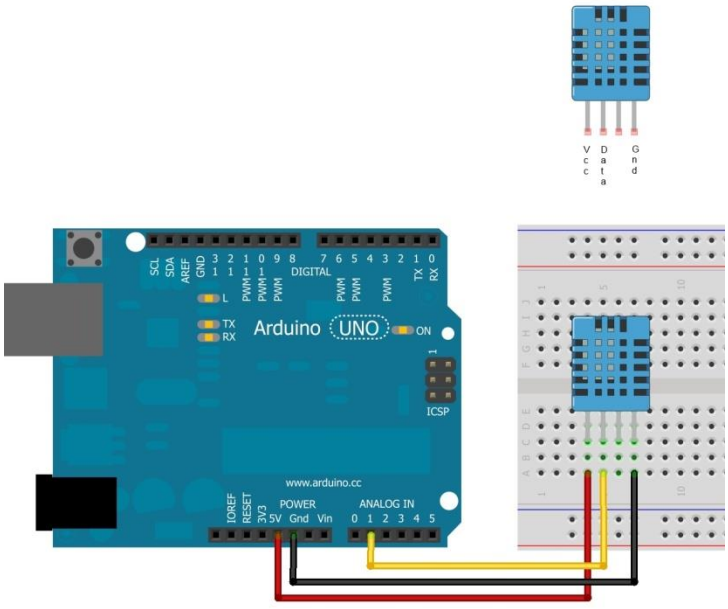
Ligações feitas e conferidas, vamos para a programação.

Código porta digital:

```
#include<dht11.h>
dht11 sensor;
void setup() {
```

```
Serial.begin(9600);
}
void loop() {
Serial.print("Lendo sensor: ");
int chk = sensor.read(2);
switch(chk) {
case DHTLIB_OK:
Serial.println("OK");
break;
case DHTLIB_ERROR_CHECKSUM:
Serial.println("Erro no checksum");
break;
case DHTLIB_ERROR_TIMEOUT:
Serial.println("Tempo esgotado");
break;
default:
Serial.println("Erro desconhecido");
}
Serial.print("Umidade (%): ");
Serial.println((float)sensor.humidity, 2);
Serial.print("Temperatura (graus Celsius): ");
Serial.println((float)sensor.temperature, 2);
delay(2000);
}
```

Código Porta Analógica:




```
/Programa : Sensor de umidade e temperatura DHT11
//Autor : Arduino e Cia

#include <dht.h>
#define dht_dpin A1 //Pino DATA do Sensor ligado na porta Analogica A1
```

```
dht DHT; //Inicializa o sensor
void setup()
{
  Serial.begin(9600);
  delay(1000);//Aguarda 1 seg antes de acessar as informações do sensor
}
void loop()
{
  DHT.read11(dht_dpin); //Lê as informações do sensor
    Serial.print("Umidade = ");
    Serial.print(DHT.humidity);
    Serial.print(" %  ");
    Serial.print("Temperatura = ");
    Serial.print(DHT.temperature);
    Serial.println(" Celsius  ");
  delay(2000);  //Não diminuir muito este valor. O ideal é a leitura a cada 2 segundos
}
```

4.6-LCD

sketch on a 2x16 LCD

Hardware Required

- Arduino Board
- LCD Screen (compatible with Hitachi HD44780 driver)
- pin headers to solder to the LCD display pins
- 10k Potentiometer
- breadboard
- hook-up wire

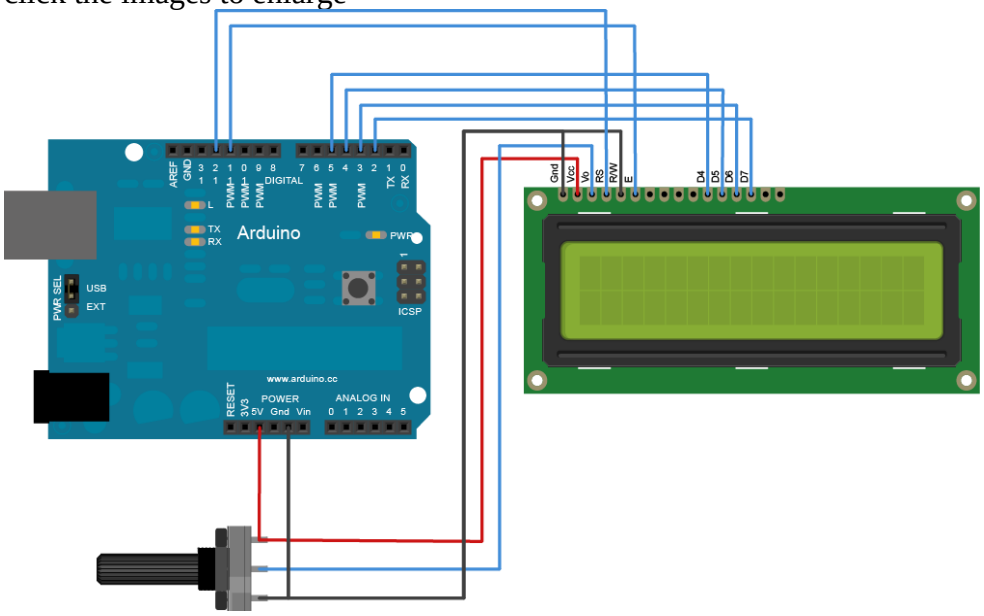
Circuit

Before wiring the LCD screen to your Arduino we suggest to solder a pin header strip to the 14 (or 16) pin count connector of the LCD screen, as you can see in the image above.

To wire your LCD screen to your Arduino, connect the following pins:

- LCD RS pin to digital pin 12
- LCD Enable(E) pin to digital pin 11
- LCD D4 pin to digital pin 5
- LCD D5 pin to digital pin 4
- LCD D6 pin to digital pin 3
- LCD D7 pin to digital pin 2
- VSS = GND
- VDD=+5V
- V0 ligado a Resistor de 5,6k para o Terra
- RW=GND
- D0, D1, D2, D3 sem conexão
- A=+5V
- K=GND ou resistor de brilho de fundo de tela

Additionally, wire a 10K pot to +5V and GND, with it's wiper (output) to LCD screens VO pin (pin3).
click the images to enlarge



Código para relógio :

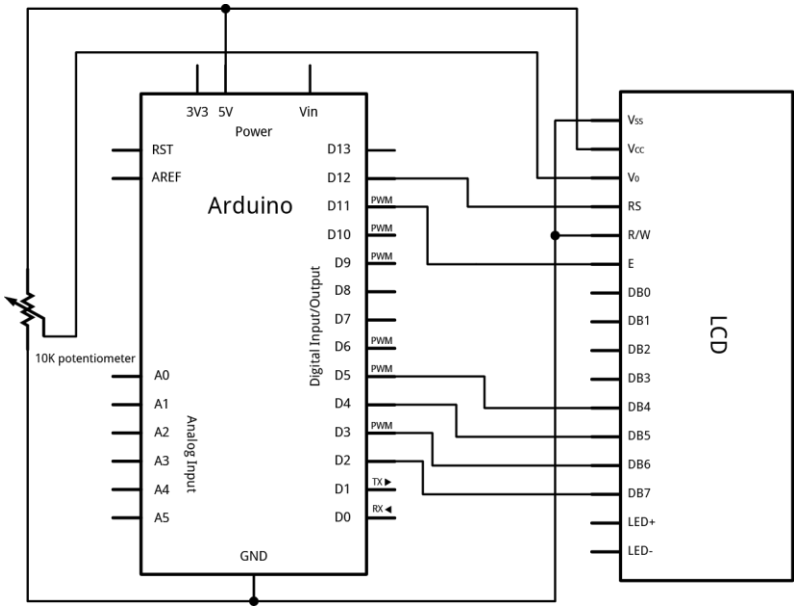
```
#include <LiquidCrystal.h> //Inclui a biblioteca do LCD

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int segundo,minuto, hora, dia, mes,ano;
unsigned long UtlTime;

void setup()
{

UtlTime=0;
minuto=0;
```

```
hora=0;
dia=0;
mes=0;
ano=0;
Serial.begin(9600);
lcd.begin(16, 2);

lcd.setCursor(0,0);
lcd.print(" Data e hora ");
lcd.setCursor(0,1);
lcd.print(" com Arduino");
delay (2000);

//Configura o minuto
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Minuto: ");
Serial.print("\nEntre Minuto:");
while(minuto==0)   {
if (Serial.available() > 0)
{
minuto= Serial.parseInt();
}
}
lcd.print(minuto);
delay(1000);

//Configura a hora
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Hora: ");
Serial.print("\nEntre Hora:");
while(hora==0)
{
if (Serial.available() > 0)
{
hora= Serial.parseInt();
}
}
lcd.print(hora);
delay(1000);

//Configura o Dia
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Dia: ");
Serial.print("\nEntre Dia:");
while(dia==0)
{
if (Serial.available() > 0)
{
```

```
dia= Serial.parseInt();
}
}
lcd.print(dia);
delay(1000);

//Configura o mês
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Mes: ");
Serial.print("\nEntre Mes:");
while(mes==0)
{
if (Serial.available() > 0)
{
mes= Serial.parseInt();
}
}
lcd.print(mes);
delay(1000);

//Configura o Ano
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Ano: ");
Serial.print("\nEntre ano:");
while(ano==0)
{
if (Serial.available() > 0)
{
ano= Serial.parseInt();
}
}
lcd.print(ano);
delay(1000);

lcd.clear();

}

void loop()
{

if(millis()-UtlTime<0)
{
UtlTime=millis();
}
else
{
segundo=int((millis()-UtlTime)/1000);
}
```

```
if(segundo>59)
{
segundo=0;
minuto++;
UtlTime=millis();
if(minuto>59)
{
hora++;
minuto=0;
if(hora>23)
{
dia++;
hora=0;
if(mes==1||mes==3||mes==5||mes==7||mes==8||mes==10||mes==12)
{
if(dia>31)
{
dia=1;
mes++;
if(mes>12)
{
ano++;
mes=1;
}
}
}
else if(mes==2)
{
if(ano%400==0)
{
if(dia>29)
{
dia=1;
mes++;
}
}
else if((ano%4==0)&&(ano%100!=0))
{
if(dia>29)
{
dia=1;
mes++;
}
}
else
{
if(dia>28)
{
dia=1;
mes++;
}
```

```
}
}
else
{
if(dia>30)
{
dia=1;
mes++;
}
}
}
}
}
Serial.print(dia);
Serial.print("/");
Serial.print(mes);
Serial.print("/");
Serial.print(ano);
Serial.println();

lcd.setCursor(0,0);
lcd.print("Data ");
lcd.print(dia);
lcd.print("/");
lcd.print(mes);
lcd.print("/");
lcd.print(ano);

Serial.print(hora);
Serial.print(":");
Serial.print(minuto);
Serial.print(":");
Serial.print(segundo);
Serial.print("\n");
Serial.println();

lcd.setCursor(0,1);
lcd.print("Hora ");
lcd.print(hora);
lcd.print(":");
lcd.print(minuto);
lcd.print(":");
lcd.print(segundo);

}
```



Code

```
/*
LiquidCrystal Library - Hello World
Demonstrates the use a 16x2 LCD display.
The LiquidCrystal
 library works with all LCD displays that are compatible with the
 Hitachi HD44780 driver. There are many of them out there, and you
 can usually tell them by the 16-pin interface.

 This sketch prints "Hello World!" to the LCD
 and shows the time.

  The circuit:
 * LCD RS pin to digital pin 12
 * LCD Enable pin to digital pin 11
 * LCD D4 pin to digital pin 5
 * LCD D5 pin to digital pin 4
 * LCD D6 pin to digital pin 3
 * LCD D7 pin to digital pin 2
 * LCD R/W pin to ground
 * 10K resistor:
 * ends to +5V and ground
 * wiper to LCD VO pin (pin 3)

 Library originally added 18 Apr 2008
 by David A. Mellis
 library modified 5 Jul 2009
 by Limor Fried (http://www.ladyada.net)
 example added 9 Jul 2009
 by Tom Igoe
 modified 22 Nov 2010
 by Tom Igoe

 This example code is in the public domain.
```

```
 http://www.arduino.cc/en/Tutorial/LiquidCrystal
 */

// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);
}
```

See Also:

lcd.begin()
lcd.print()
lcd.setCursor()
Liquid Crystal Library
Blink: control of the block-style cursor.
Cursor: control of the underscore-style cursor.
Display: quickly blank the display without losing what's on it.
TextDirection: control which way text flows from the cursor.
Scroll: scroll text left and right.
Serial input: accepts serial input, displays it.
SetCursor: set the cursor position.
Autoscroll: shift text right and left.

Comandos:

include <LiquidCrystal.h>

Lcd.begin(16,2) = inicializa display de 16 caracteres por duas colunas

LCD cursor()=exibe o cursor

LCD.SetCursos(0,0)=Coloca o texto na primeira Linha

LCD.SetCursos(0,1)=Coloca o texto na Segunda Linha

LCD.autoscroll()=deslocamento automático

LCD.write(texto ou variável aqui); exibe a variável ou texto

LCD.rightToLeft();=vai para a esquerda da próxima letra

LCD.leftToRight()=vai para a direita da próxima letra

LCD.home()=vai para 0,0
LCD.blink()=pisca o cursor
LCD.noblink()=não pisca o cursor
LCD.nodisplay()=oculta os caracteres
LCD.display()=exibe os caracteres

4.7-SD Card

Gravando um arquivo .doc no SD Card

Olá pessoal, agora vamos ver como gravar e visualizar um arquivo .doc no cartão SD, gravando com o Arduino e visualizando o arquivo no notebook, desktop, tablet, ou ate mesmo no seu smartphone. Neste projeto foi utilizado um adaptador de microSD da Samsung, não tive problemas com este cartão, portanto se houver problemas com o que você utilizar deixe um comentário.

Obs. importante: Antes de compilar, coloque o cartão SD.

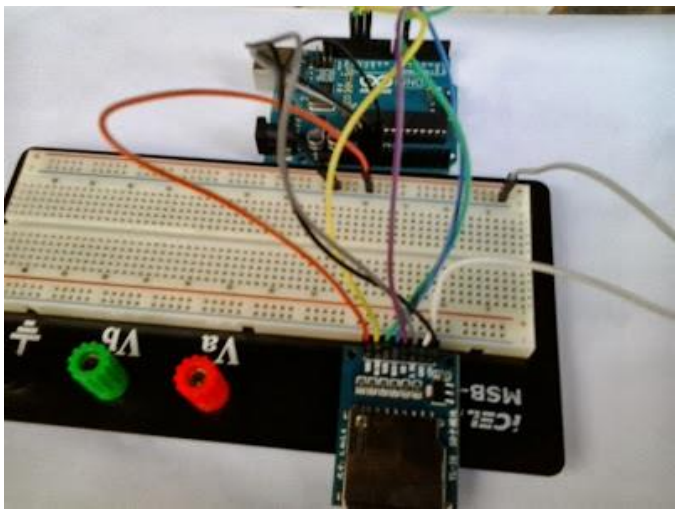
Os códigos serão todos comentados para melhor entendimento, podendo ser retirados após a compreensão de cada linha. Bom trabalho !!!

Componentes necessários

1 Protobord
1 Modulo SD
Fios jumpers

Conectando os componentes

Primeiro, certifique-se de que seu Arduino esteja desligado, desconectando-o do cabo USB. Agora, pegue o módulo e os fios e conecte-os como mostra a figura.



Para melhor entendimento conecte os jumpers assim

MOSI - pino 11 do Arduino
MISO - pino 12 do Arduino
CLK ou SCK - pino 13 do Arduino
CS - pino 6 do Arduino

Obs.: Neste modulo tem que conectar também o 2 GND, 1 Vcc de 3V e 1 Vcc 5V todos no Arduino.

Não importa se você utiliza fios de cores diferentes ou furos diferentes na protoboard, desde que os componentes e os fios estejam conectados na mesma ordem da figura.
Tenha cuidado ao inserir os componentes na protoboard. Caso sua protoboard seja nova, a superfície dos furos ainda estará rígida. A não inserção cuidadosa dos componentes pode resultar em danos.
Certifique-se de que todos os componentes estejam conectados corretamente. Quando você estiver seguro de que tudo foi conectado corretamente, ligue seu Arduino e conecte o cabo USB.

Agora vamos ao código

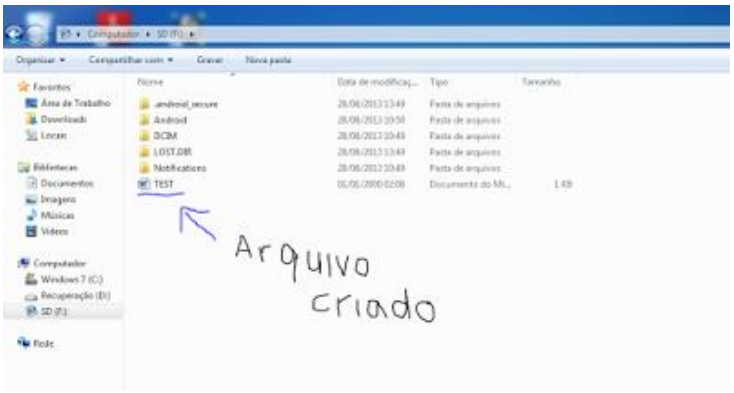
```
/*
Projeto 17 - Gravando um arquivo .doc no SD Card
Este exemplo mostra como ler e escrever dados em um cartão SD
Conexão dos pinos do SD Card no Arduino

 MOSI  - pino 11
 MISO -  pino 12
 CLK ou SCK - pino 13
 CS - pino 6

 */

#include <SD.h>  // Biblioteca para manipular os dados no Cartão
File meuArquivo;  // Criação de uma variável para guardar os dados inseridos
void setup()
{
 // Abre a comunicação serial liberando a porta
  Serial.begin(9600);
   while (!Serial) {
    ; // Aguarda a porta serial conectar.
  }

  Serial.print("Inicializando SD card...");  // Inicializa o cartão SD
  if (!SD.begin(6)) {  // Esta condição analisa se o cartão está inserido
corretamente no shield
    Serial.println("A inicializacao falhou! ");
    return;
  }
```

```
  Serial.println("Inicializacao concluida! ");  // Imprime no Serial Monitor que foi bem
sucedida a inserção do SD card

  // Abre o arquivo test.doc
   meuArquivo = SD.open("test.doc", FILE_WRITE);

  // Se o arquivo esta ok, então será escrito isso
  if (meuArquivo) {
    Serial.print("Escrevendo test.doc...");
    meuArquivo.println("Parabens voce conseguiu!
www.facacomarduino.blogspot.com");
          // Fecha o arquivo:
    meuArquivo.close();
    Serial.println("Concluido! ");
  } else {
    // Se o arquivo não abriu, um menssagem de erro será mostrada
    Serial.println("Erro ao abrir arquivo test.doc");
  }

  // Abre novamente o arquivo para lê-lo

  meuArquivo = SD.open("test.doc");
  if (meuArquivo) {
    Serial.println("test.doc:");

    // Le o arquivo até que não há mais nada nele

    while (meuArquivo.available()) {
          Serial.write(meuArquivo.read());
    }
    // Fecha o arquivo

    meuArquivo.close();
  } else {
          // Se o arquivo não abriu, será mostrado uma menssagem de erro

    Serial.println("Erro ao abrir arquivo test.doc");
  }
}

void loop()
{
          // Nada acontece aqui
}
```

Para certificar se o código está correto pressione o botão Verify/Compile.
Se tudo estiver correto pressione o botão Upload para fazer o upload do código para seu Arduino. Pronto, abra o Serial Monitor e logo você verá os passos para gravação do arquivo, assim que gravar retire o cartão e insira-o no computador lá terá um arquivo chamado TEST, clique nele e ele será aberto.

Captura do Serial Monitor

Captura dentro da raiz do SD Card



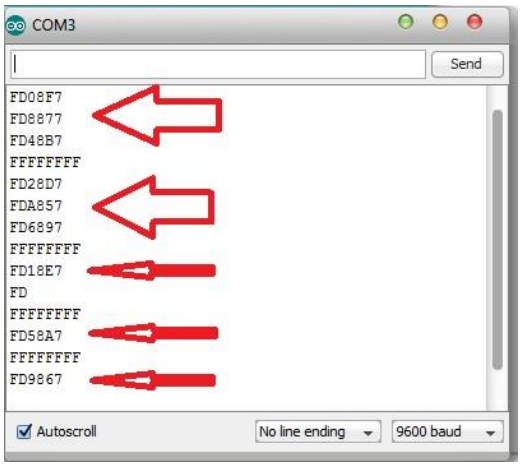
4.8-IR

Divide-se em duas partes, a primeira mapeia o controle e a segunda filtra as funções, vejamos:

Mapeando o Controle Remoto

#include <IRremote.h>

```
int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
}
void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value, HEX);
    irrecv.resume(); // Receive the next value
  }
}
```



Comando de detecção de Controle

```
#include <IRremote.h>
int RECV_PIN = 8;
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup() {
 irrecv.enableIRIn(); // Start the receiver
  beep(50);
}
void loop() {
 if (irrecv.decode(&results))
  {
  translateIR();
  irrecv.resume();
  }
}
void beep(unsigned char delayms){
```

```
  analogWrite(9, 20);      // Almost any value can be used except 0 and 255
  delay(delayms);          // wait for a delayms ms
  analogWrite(9, 0);       // 0 turns it off
  delay(delayms);          // wait for a delayms ms
}
  void translateIR() // takes action based on IR code received
{
switch(results.value){
case 0xFD30CF:
    // tecla 0
    beep(50);
    break;
case 0xFD08F7:
    //tecla 1
    beep(50);
    break;
}
}
```

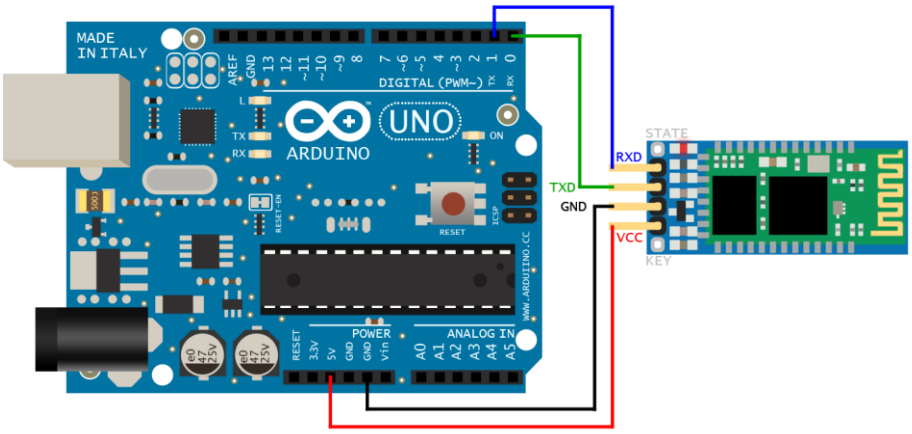
4.9-Bluetooth

Assistir:
https://www.youtube.com/watch?v=IwnofqvGKow
https://www.youtube.com/watch?v=R6o1UlC-ztg
https://www.youtube.com/watch?v=hWUSloIi6gg

Basta pegar o código usado na aula de comunicação Serial:

```
void setup() {
Serial.begin(9600);
pinMode(3, OUTPUT);
}
void loop() {
char c = Serial.read();
if (c=='A') {
digitalWrite(3, HIGH);
}
if (c=='B') {
digitalWrite(3, LOW);
}
}
```

Conecte o Bluetooth na alimentação (5v e GND), pinos 1 e 2 digitais(RX e TX), utilize alimentação de uma fonte, pois o cabo USB irá conflitar com os pinos 1 e 2.



Para testar baixe o BT Term no Google Play e envie através deste programa a Letra A ou B, para acender ou apagar o LED.

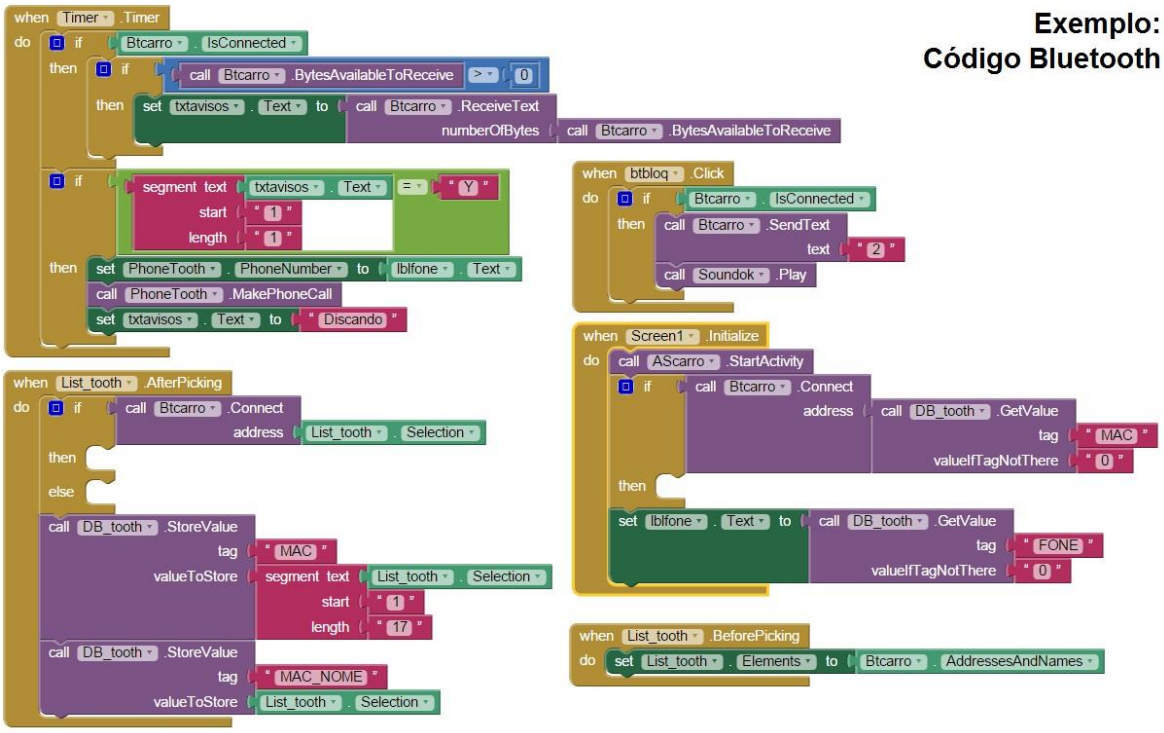
4.10-Exercicios Propostos:

1-Faça o display exibir data e Hora
2-Faça o display exibir umidade e temperatura
3-Faça o display exibir distância do sensor Ultra som
4-Faça apertar no teclado e aparecer o numero no Display
5-Faça um alarme Residencial com sensor PIR
6-Faça apertar no Controle Remoto e ela exibir no display a tecla pressionada

5-Appinventor

Interface para desenvolvimento de aplicações via WEB
Acesse: **http://ai2.appinventor.mit.edu/**

Exemplo:
Código Bluetooth



6-Teste Final:

1-Faça um despertador para tocar todos os dias ás 06:00, para teste pode-se utilizar outro horário,gravados no SD Card.

2- faça um app que temporize um relé ligado a um arduino, o tempo é configurado no Celular

3-Faça uma página web que acione o led ás 09:00hs, para teste pode-se utilizar outros horários

4-Faça um Sensor para a Sala, onde o Sensor PIR sirva como alarme ou ligar auto a lâmpada,ligar a lâmpada pelo celular, utilizando um app para escolher a função.