

**VER**

ver. beta 01

## **Indice**

- 3 - Intro
- 4 - Nota importante
- 5 - Links
- 6 - Software Utilizado
- 7 - Licença
- 8 - Conceito
- 13 - Utilização
- 15 - Monoculo
- 16 - Posicionamento
- 17 - Recomendação de Compra
- 18 - Arduino
- 19 - Ligacoes modelo por cabo
- 20 - Esp32
- 21 - Ligacoes modelo bluetooth
- 22 - Programacao Arduinod
- 31 - Codigo Arduino
- 32 - App Inventor
- 33 - Modo Offline
- 34 - Código Menu
- 41 - Código ligação por cabo
- 51 - Código ligação por bluetooth
- 66 - Lente
- 67 - Custo das Versões
- 68 - Versões de produção
- 69 - Objectivos futuros
- 71 -Em desenvolvimento
- 72 - Notas Finais
- 73 - Agradecimentos
- 74 - Créditos

## **Motivo**

Este projeto nasce da interação com uma colega surda, que com o uso das máscaras durante o período da pandemia se tornou mais isolada, porque a sua forma mais comum de perceber o que nós colegas lhe queríamos dizer, a leitura de lábios, foi bloqueada com o uso das máscaras. Isso chamou a minha atenção para a sua condição, não apenas nesta fase mas como algo que dura uma vida. Num mundo onde se pode ir viajar por lazer ao espaço, não temos tecnologia para resolver isto? Ou simplesmente ninguém ainda se debruçou sobre o assunto com a perspetiva correta .

## **Objectivo**

O objetivo do Projeto "Ouver" é facultar as pessoas com deficiência auditiva um modo de verem informações sonoras convertidas em texto. Isso significa não só converter fala em texto mas também sons de primeira importância como avisos sonoros.

## **Requisitos**

Ser acessível, simples e aberto.  
A informação deve ser apresentada em tempo real e deve estar sempre acessível.

## **Importante**

Eu gostaria de frisar que a minha formação é em Design do Produto. Os conhecimentos que utilizei de programação e de eletrónica adquiri por gosto através de cursos online e muitas, mas mesmo muitas horas a ver tutoriais e a ler fóruns online.

Mesmo assim consegui criar este projeto e por-lo a funcionar. Obviamente este pode ser muito otimizado e por isso abri o projeto para que quem possa contribuir com o seu conhecimento o faça e ajude a que este chegue a quem necessita.

Como poderão ver no manual, não há um único pormenor ou pensamento que ocultei, em prol dessa abertura.

A quem precisa deste projeto, espero que seja útil e ajude, a quem pode ajudar a desenvolver, a tua ajuda é sempre bem-vinda.

Como este é um projeto faça você mesmo não me responsabilizo por qualquer acidente.

Não necessita de recriar todos os passos, a app, os ficheiros do MIT AppInventor e do arduino estão todos disponíveis no Github.

ISTO É UMA VERSÃO  
BETA

## **Github**

[https://github.com/Developkings/Project\\_OUVER](https://github.com/Developkings/Project_OUVER)

## **Contactos**

Por favor contate-me pela página do facebook

<https://www.facebook.com/projectouver>

## **Página**

<https://www.facebook.com/projectouver>

## **Grupo Facebook**

<https://www.facebook.com/groups/857505315152820>

## Software utilizado



Desenvolvimento da app:  
<https://appinventor.mit.edu/>



Programação das boards  
<https://www.arduino.cc/en/software>



Produção do manual  
<https://www.designer.io/en/>



Edição de Imagens:  
<https://www.gimp.org/>



Criação de modelos 3D:  
[blender.org](https://blender.org)



Visualização e criação de integrados:  
[fritzing.org](https://fritzing.org)

## Licença

Attribution-NonCommercial 4.0 International (CC BY-NC 4.0)

Você tem o direito de:

Compartilhar — copiar e redistribuir o material em qualquer suporte ou formato

Adaptar — remixar, transformar, e criar a partir do material

De acordo com os termos seguintes:

Atribuição — Você deve atribuir o devido crédito, fornecer um link para a licença, e indicar se foram feitas alterações. Você pode fazê-lo de qualquer forma razoável, mas não de uma forma que sugira que o licenciante o apoia ou aprova o seu uso.

NãoComercial — Você não pode usar o material para fins comerciais.

Sem restrições adicionais — Você não pode aplicar termos jurídicos ou medidas de caráter tecnológico que restrinjam legalmente outros de fazerem algo que a licença permita.

<https://creativecommons.org/licenses/by-nc/4.0/deed.pt>

## Motivo

Um Projeto aberto .

Com esta licença podemos manter todos a par dos últimos desenvolvimentos e dar a comunidade a possibilidade de ajudar no mesmo.

Possibilidade de fazer você mesmo.

Com a divulgação da integra de todos os documentos do projeto e recorrendo ao uso de tecnologias modulares acessíveis e disponíveis em todo o lado, garantimos que qualquer pessoa possa construir o seu próprio dispositivo e que terá sempre acesso a ultima versão disponível.

Assumimos que existem passos mais complicado que alguma população possa ter dificuldade em fazer, mas contamos que no bom espírito de responsabilidade e interajuda social, as pessoas recorram a ajuda de makers, e das fablabs locais para a execução dos mesmos.

Estas comunidades já demonstraram que estão aqui prontas para ajudar nestes casos e contamos com elas para manter este projeto acessível, barato e em continuo desenvolvimento.

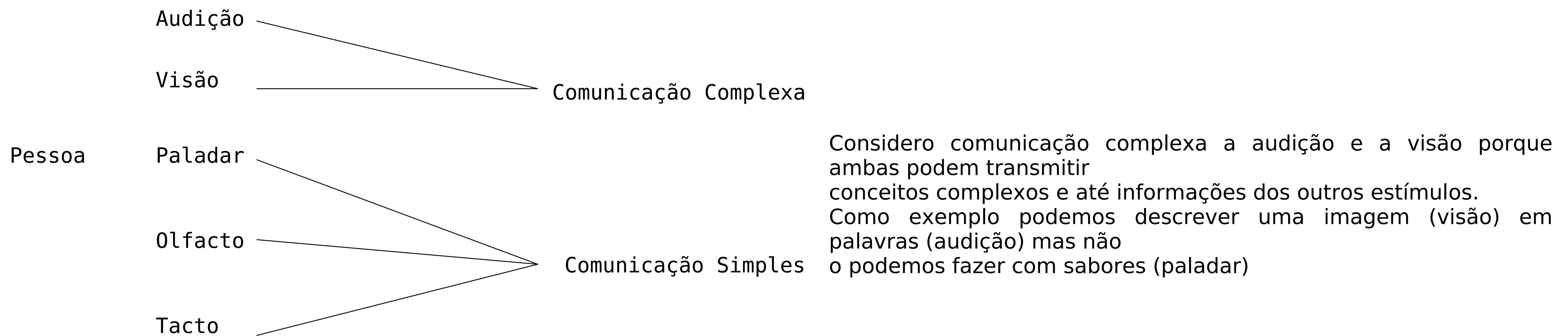
## Controlo do custo

Ao mantermos o projeto aberto garantimos que caso este produto venha a ser produzido industrialmente, este tenha um preço justo, porque senão o tiver, as pessoas poderão sempre fazer o seu.

Contamos com este facto e com a competitividade faca os preços baixarem ao ponto de ser mais barato comprar feito do que fazer.

## Uma sequencia lógica simples

Quero deixar aqui a sequencia lógica que me levou a solução.  
Compreender esta sequencia é o fator mais importante para que se compreenda o projeto de forma a poder não só construir, mas também desenvolver.  
Pode parecer infantil, mas é o uso de conceitos básicos que permite a construção de conceitos complexos.



## Uma sequencia lógica simples

Audição → Visão

A solução teria de passar por converter sons em imagens, e isso já é feito faz anos na TV e cinema, são as legendas com descrição áudio.

Agora teria de encontrar algo que me convertesse voz para texto, e a solução estava no meu bolso



Só que não há necessidade de utilizar um assistente, podemos utilizar uma das suas formas mais básicas que vem com todos os telemóveis, trata-se da opção de acessibilidade conversor de fala para texto.

## **Uma sequencia lógica simples**

Agora que temos uma forma de obter e converter a fala em texto faltava uma forma de ver o texto como se de uma legenda se tratasse, para isso só existem três soluções possíveis

### **Óculos Vr**

Permitem ver a informação diretamente do telemóvel mas impedem a visão do resto.

Funcionam em qualquer condição de luminosidade.

Volumosos

### **Óculos Ar**

Permitem ver a informação diretamente do telemóvel

Permitem ver o mundo em simultâneo

Funcionam em qualquer condição de luminosidade mas para isso requerem ou um telemóvel com muita luminosidade ou então uma lente escurecida

Muito volumosos

### **Google glasses (versões DIY)**

Não requer o telemóvel para se ver a informação

Permitem ver o mundo em simultâneo

Funcionam em qualquer condição de luminosidade quando feito com hardware extremamente caro

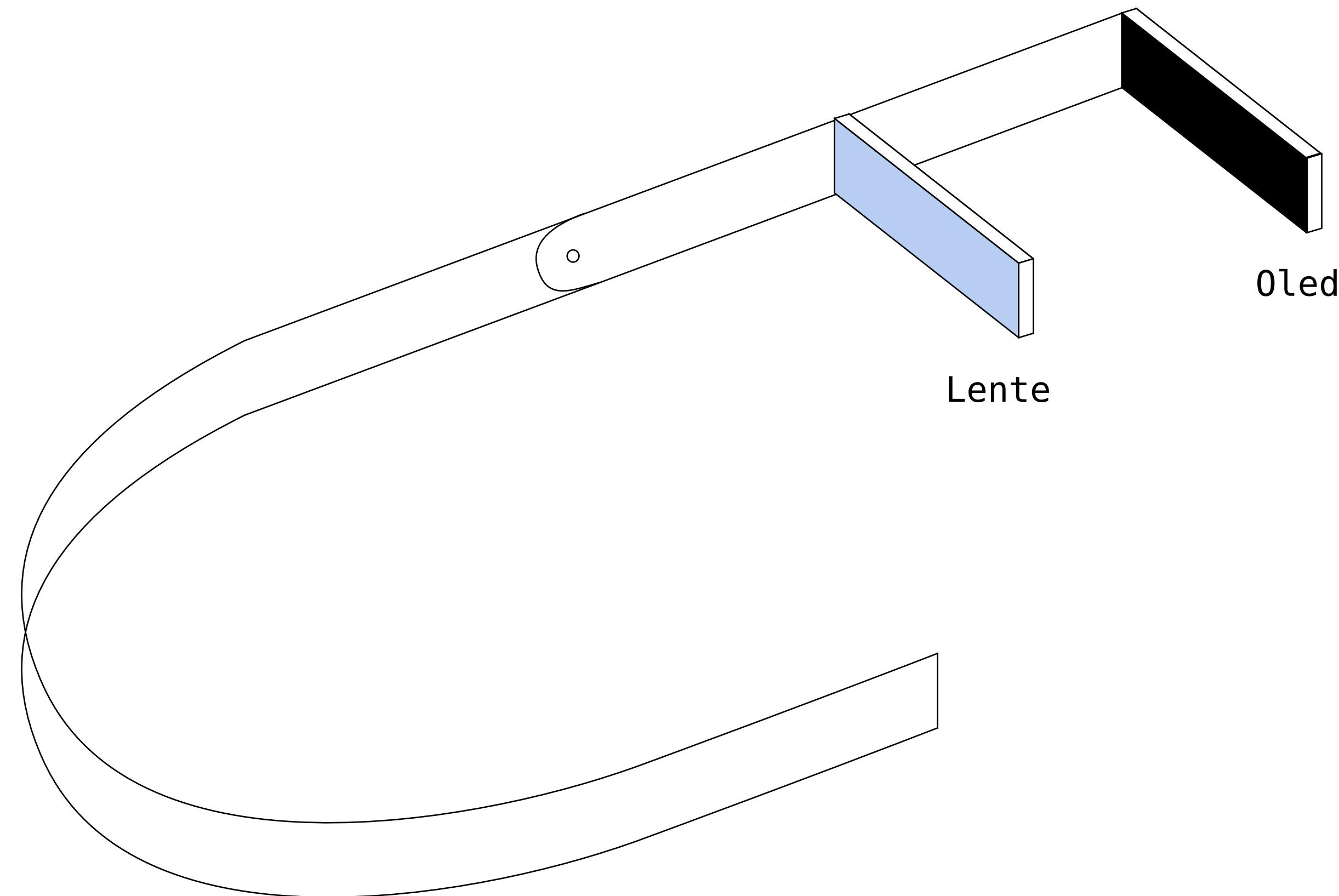
Volume mínimo

Eletrónica pode ser feita com recurso a ajardino, mas perde visibilidade em condições de luminosidade equivalentes a um dia de sol.

## Uma sequencia lógica simples

Para isso desenvolvi um híbrido com as melhores características de cada.

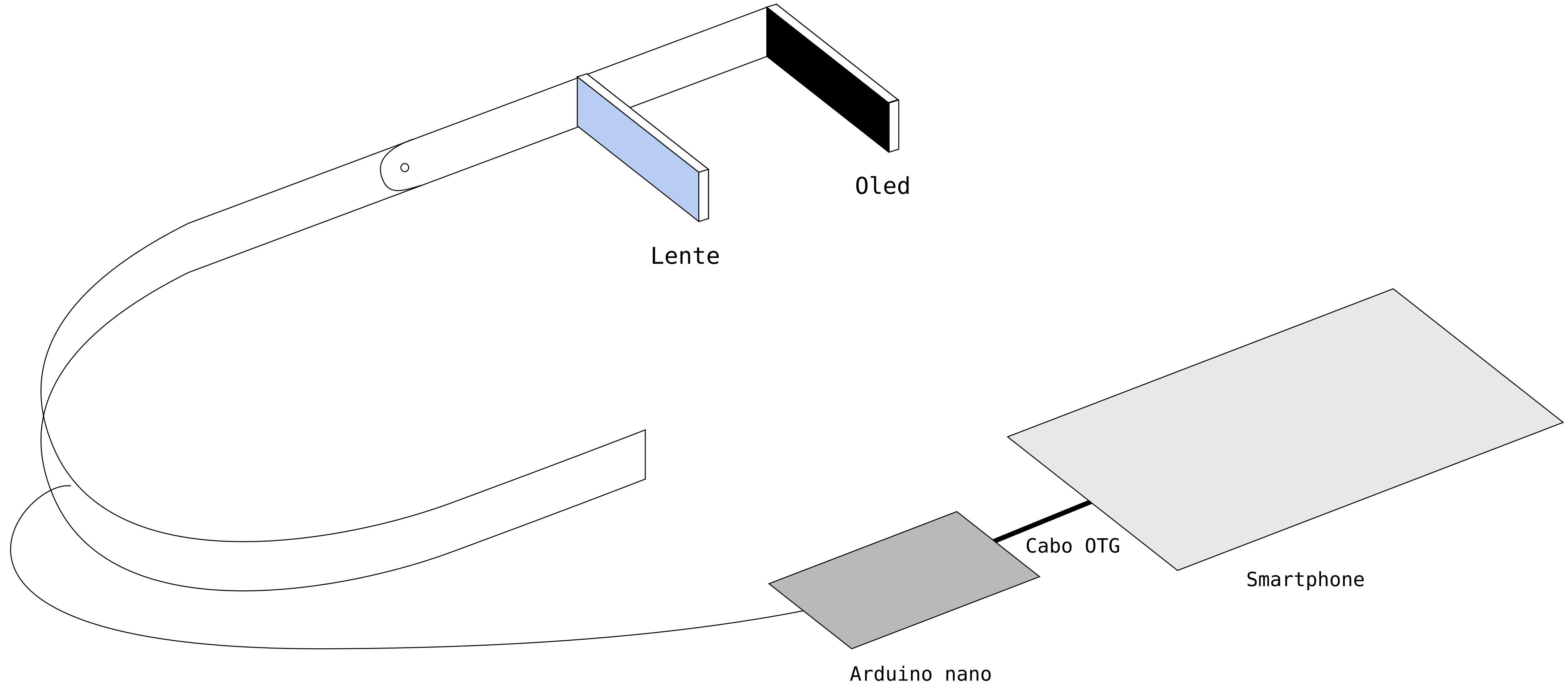
Utilizei o sistema dos óculos VR para garantir que é visível em qualquer condição de luminosidade e substitui o telemóvel por um OLED pequeno de forma a só ocupar uma área reduzida do campo de visão.



O olho humano só consegue focar a uma distância mínima de 26cm por isso utilizei a lente de uns óculos VR para permitir colocar o OLED mais perto dos olhos

## Uma sequencia lógica simples

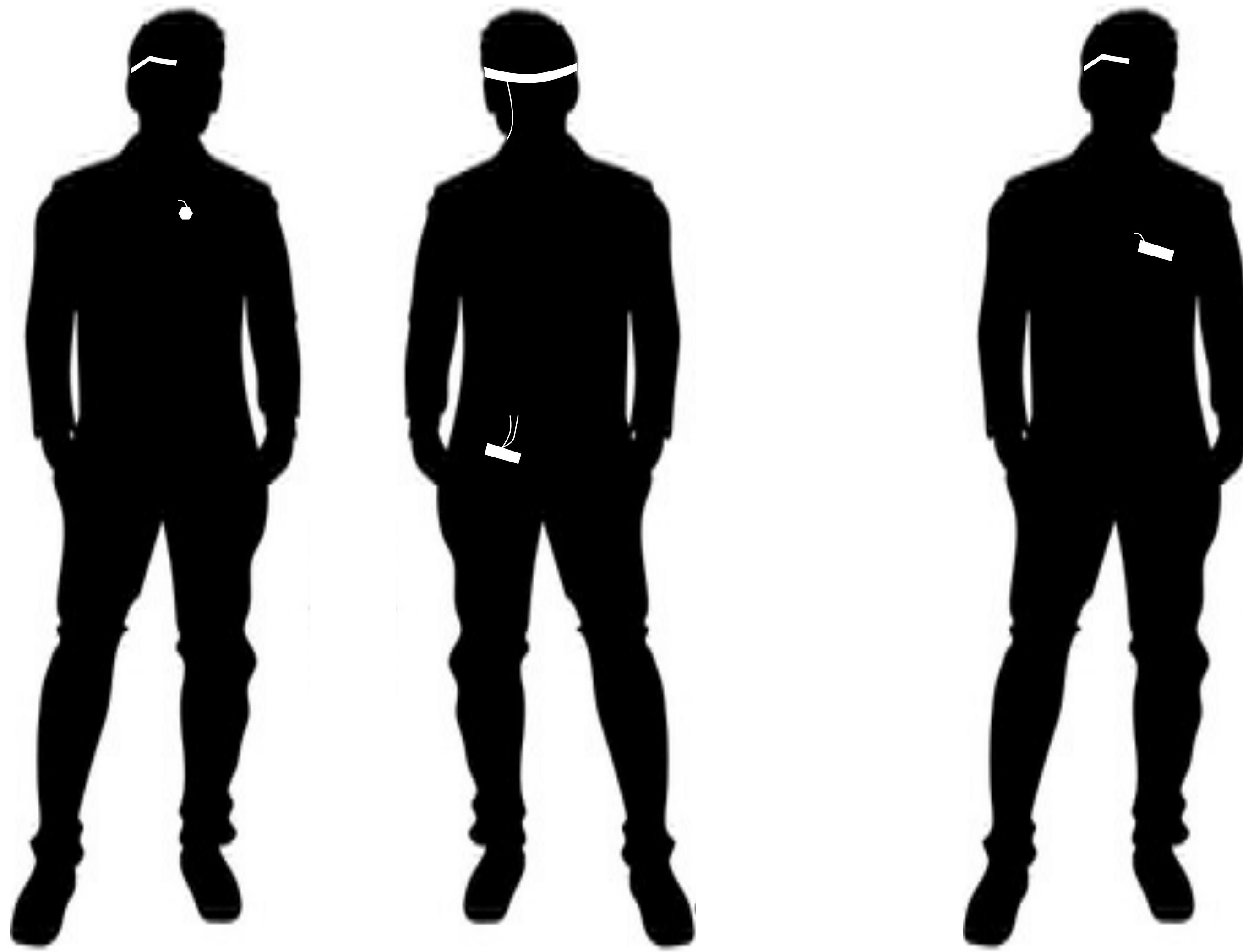
O resultado final é o seguinte



O telemóvel comunica com o arduino através da sua porta USB e um cabo OTG, o mesmo cabo é capaz de fornecer ao arduino e ao OLED energia para o seu funcionamento

## Uma sequencia lógica simples

Modo de utilização versão  
por cabo



Usado num bolso com  
auxilio de um microfone  
para captura de som

Usado no bolso da camisa  
com o microfone do  
telemovel destapado

## Uma sequencia logica simples

Modo de utilização versão por  
bluetooth



Usado no bolso da camisa  
com o microfone do  
telemovel destapado

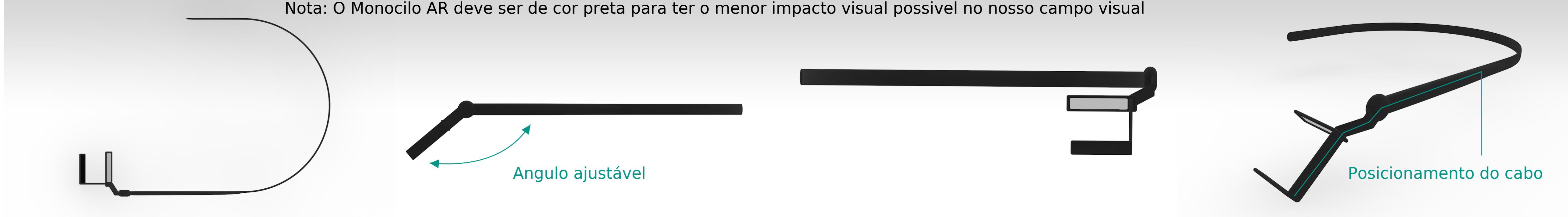
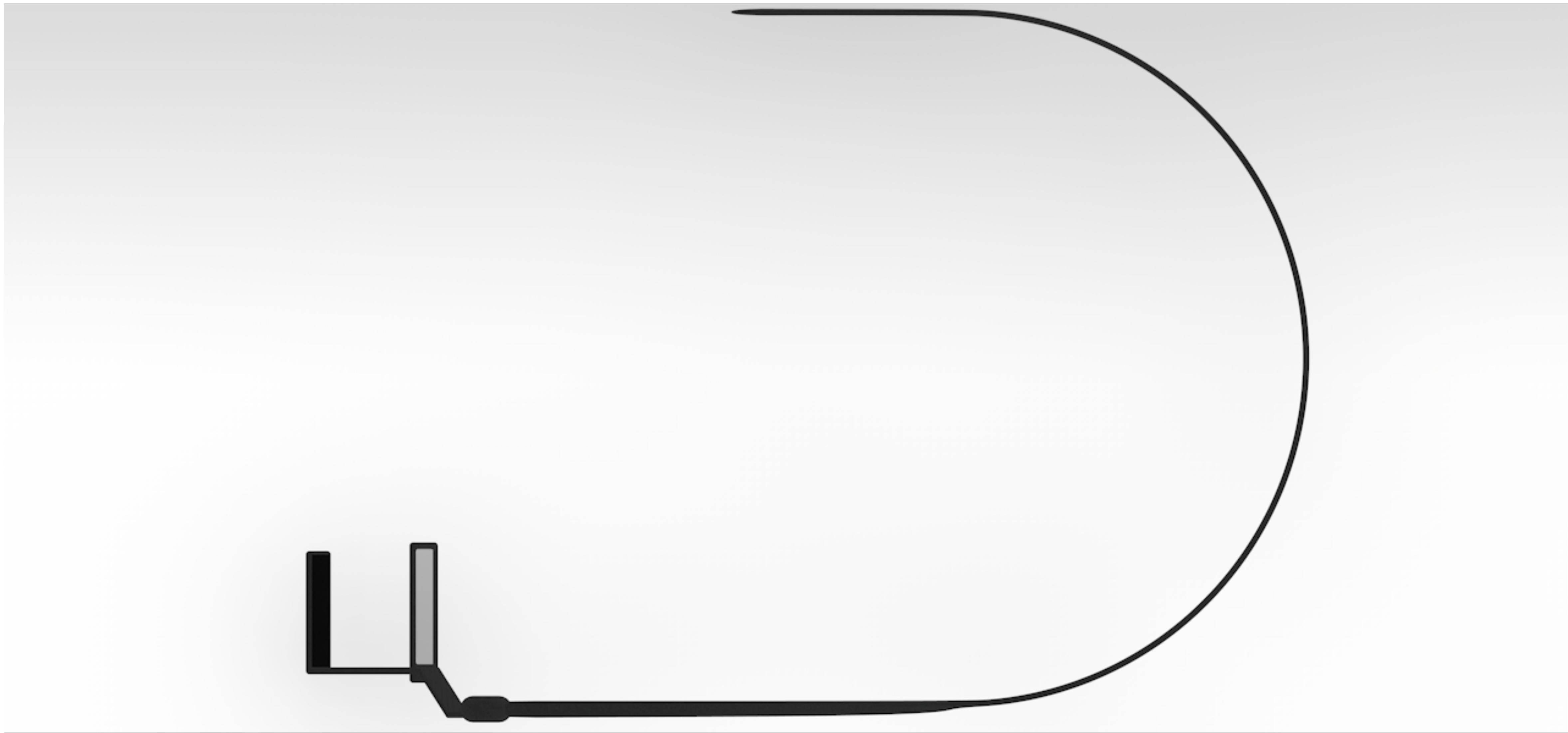


Usado num bolso com  
auxilio de um microfone  
para captura de som



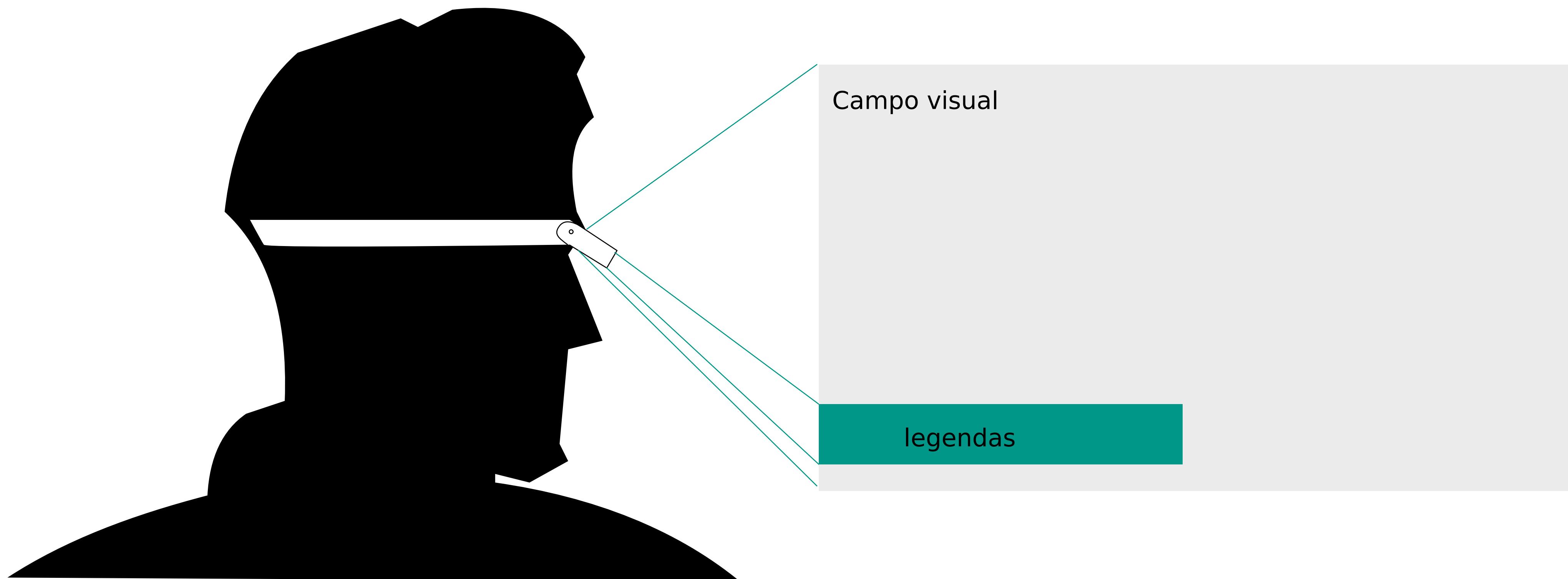
Futura versão com  
microfone incluido nos  
óculos

## Monóculo



## Uma sequencia lógica simples

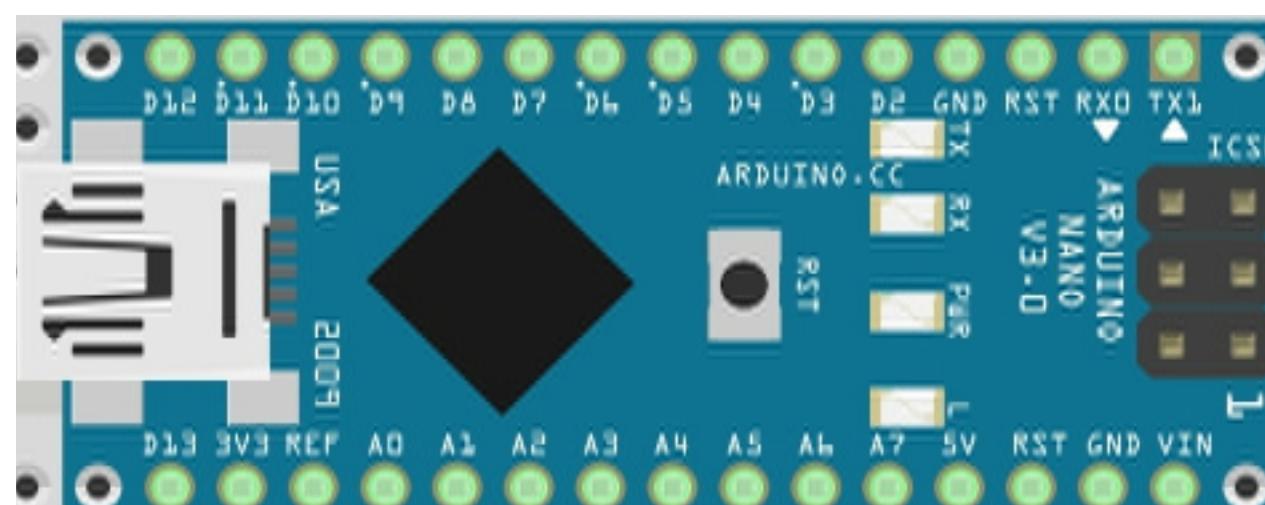
Posicionamento no campo visual



# Recomendação de Compra de componentes

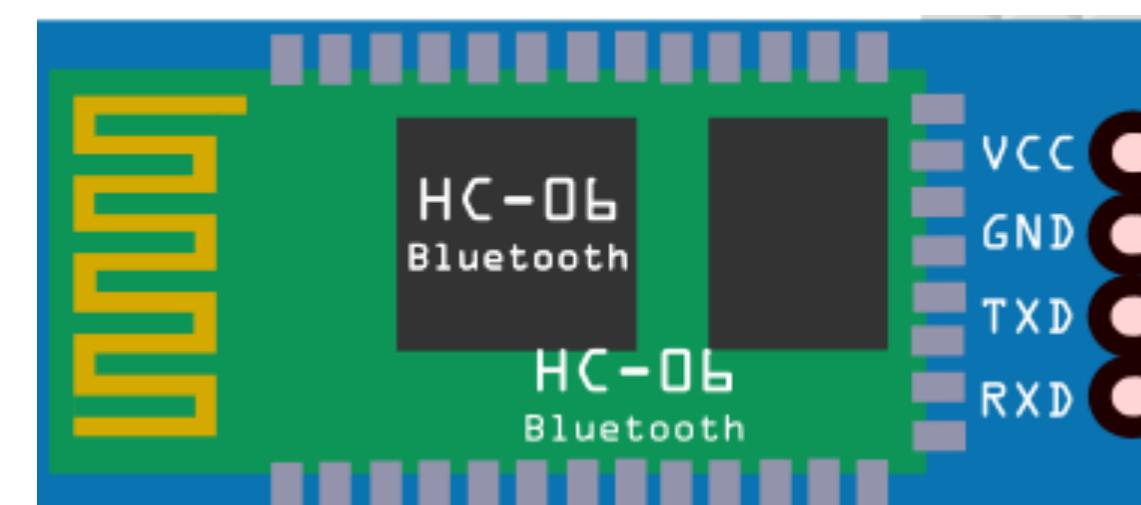
Apesar de existirem varios locais onde poderá adquirir os componentes necessários para executar o seu monoculo a baixo preço, muitas dessas opções são cópias que nem sempre seguem as especificações das boards e componentes originais. Por isso, caso o valor de peças originais não seja proibitivo para si, recomendamos que as obtenha na fonte ou em lojas oficiais.

Ao fazer isso estará igualmente a ajudar no desenvolvimento das mesmas e das suas bibliotecas.



# Arduino Nano

<https://store.arduino.cc/products/arduino-nano>

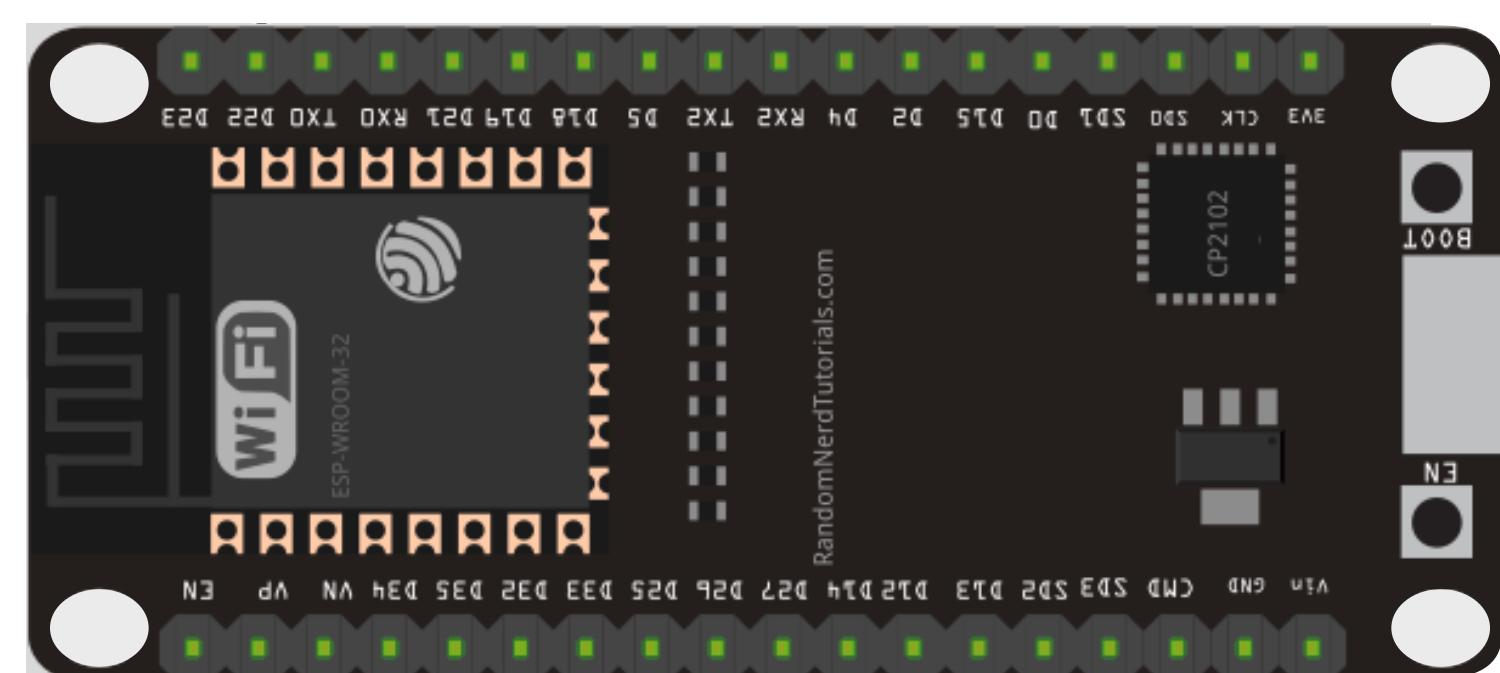


# Bluetooth HC-06

## Sem pagina oficial



oled "128x32" ssd1306 I2c  
Sem pagina oficial

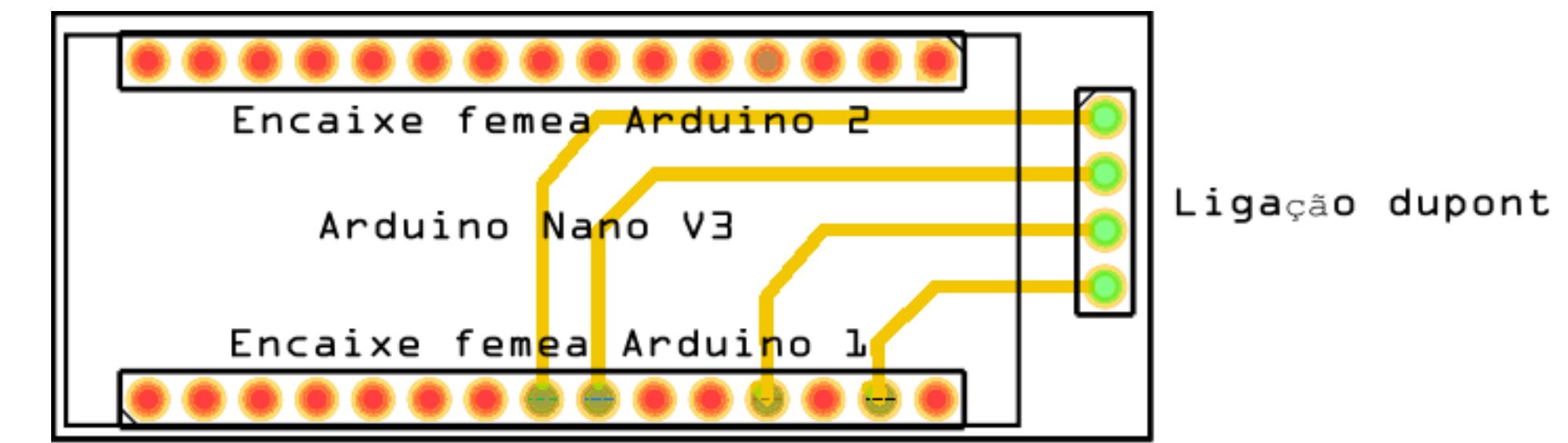
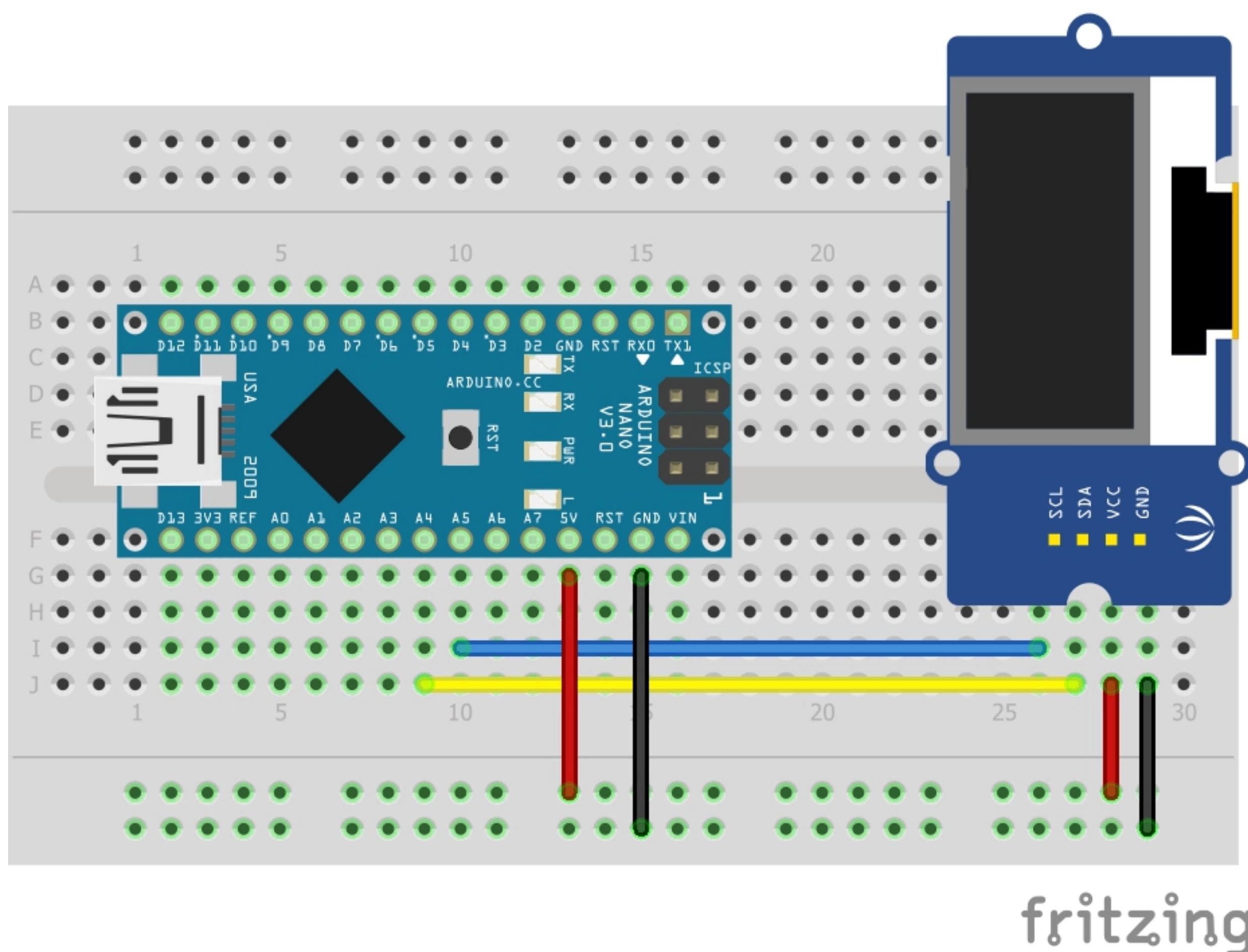


# ESP32

<https://www.espressif.com/en/products/socs/esp32>

## Ligações Arduino por cabo

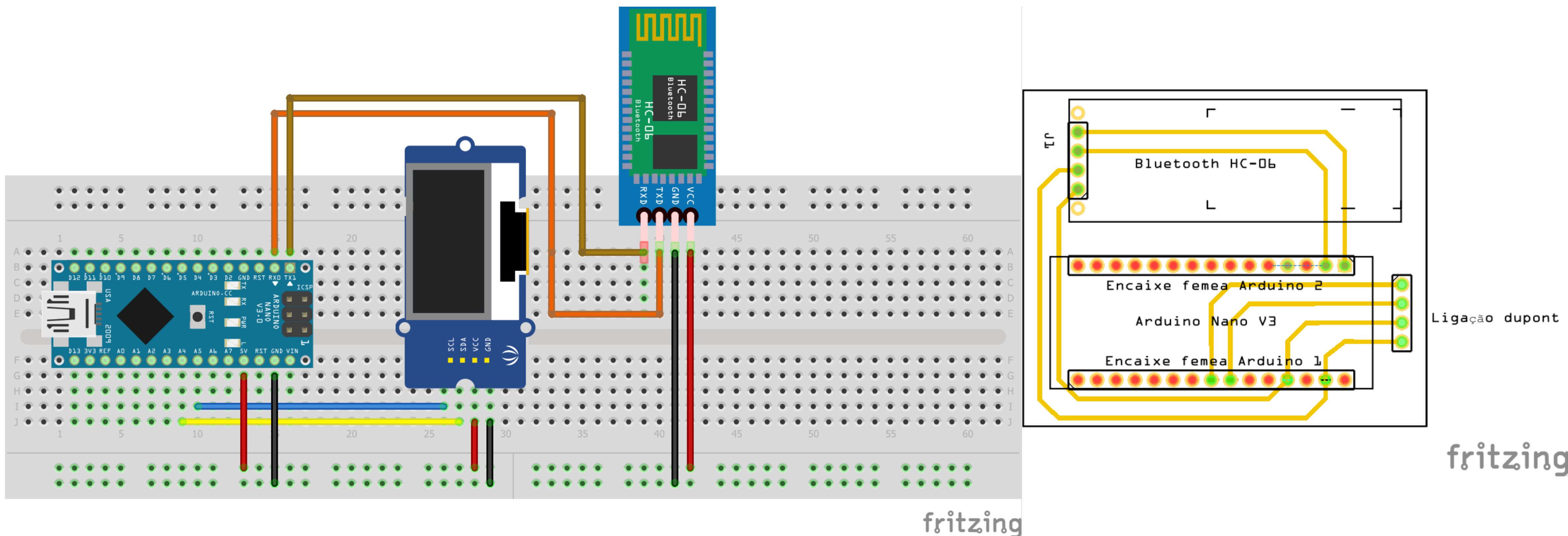
Testado, placa integrada em desenvolvimento



Este é o modelo mais económico, usa o próprio telemóvel como bateria para alimentar o monóculo.  
Em contrapartida necessita de ligações por cabo entre os componentes

Nota: O OLED na imagem não é o utilizado no projeto. O que optei por utilizar foi um  
OLED 128x32 i2c monochrome ssd1306

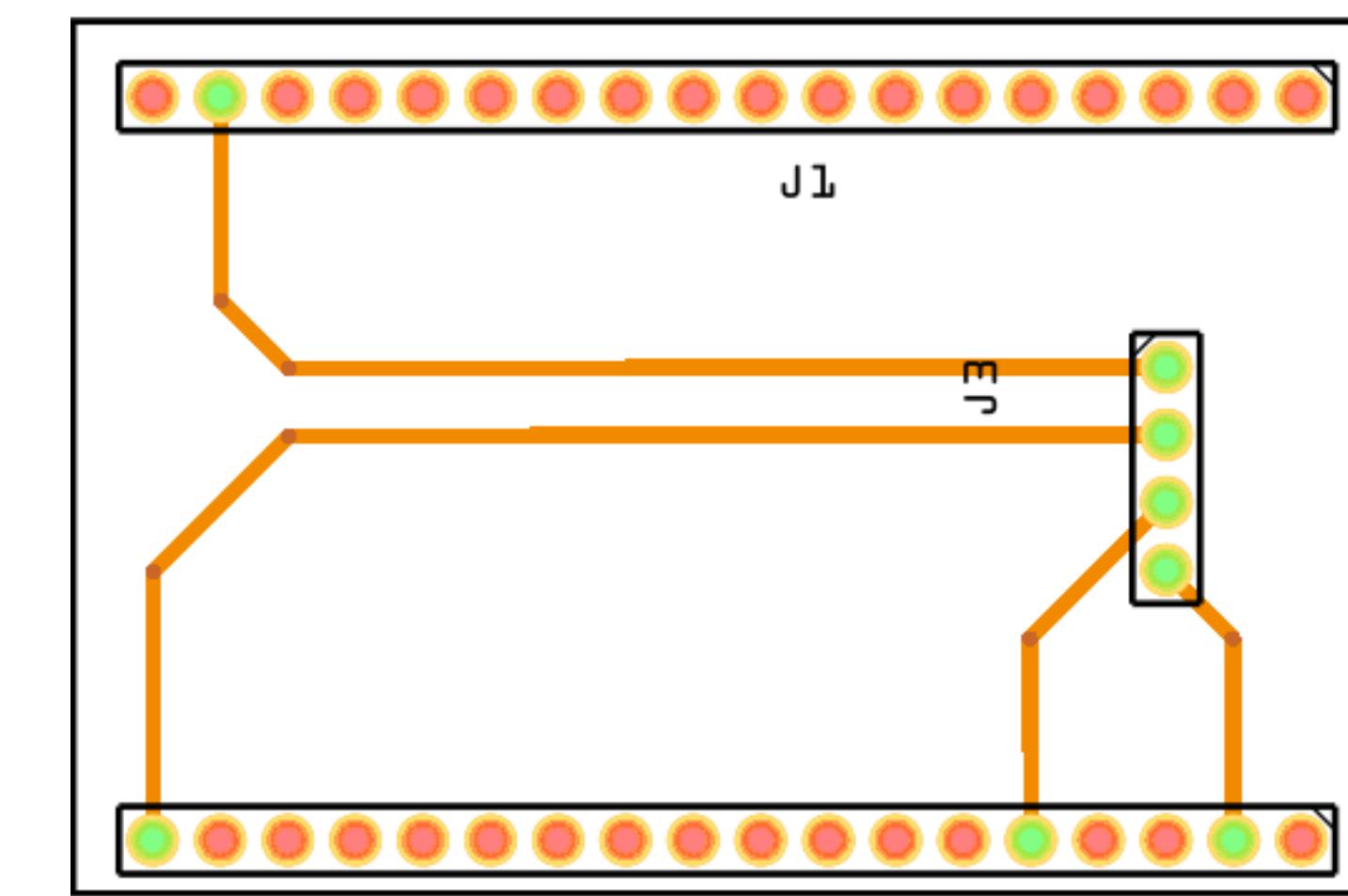
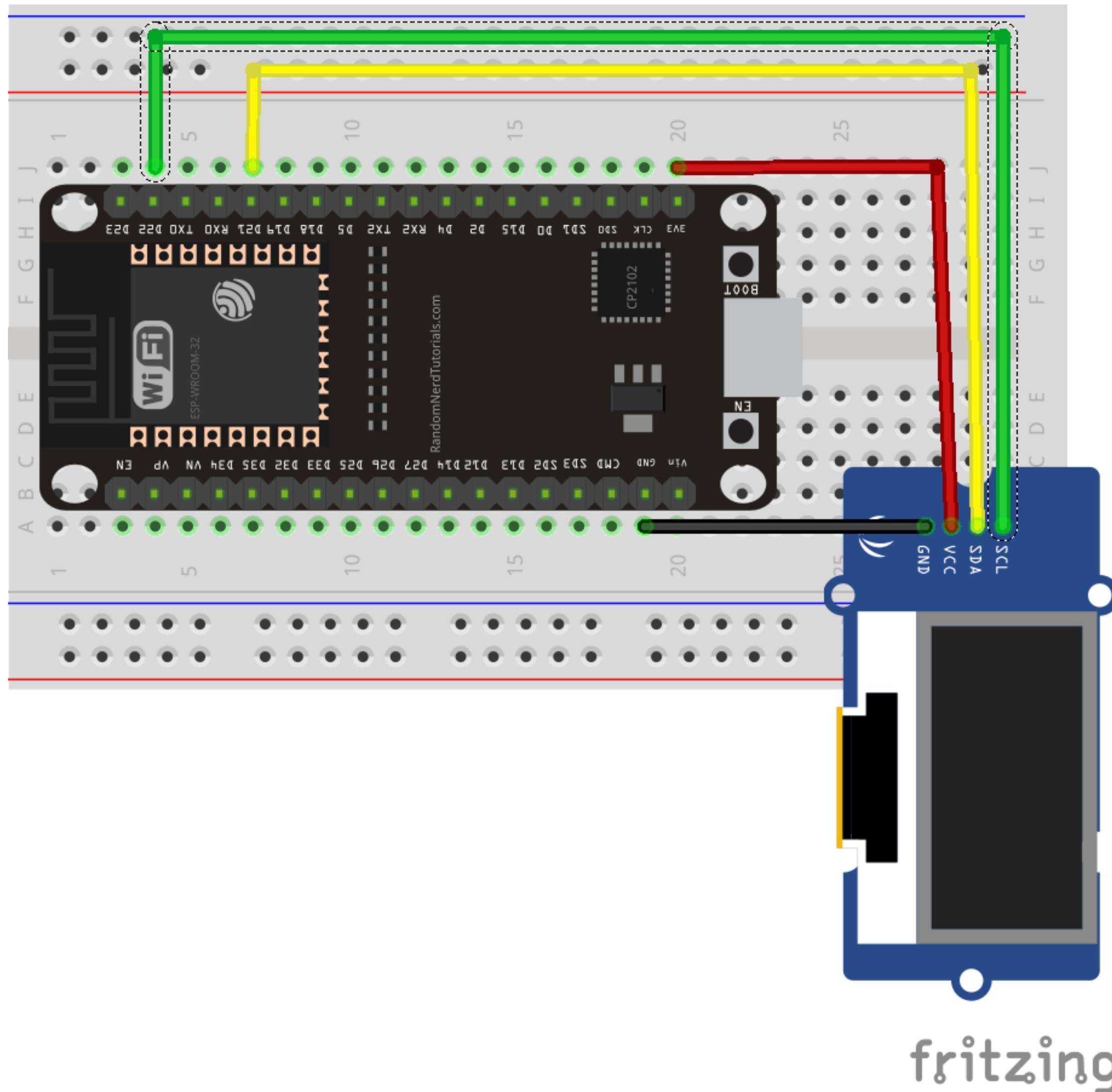
# Ligações Arduino bluetooth com cabo Testado, placa integrada em desenvolvimento



Este modelo é alimentado por um powerbank de telemovel, permite que este seja utilizado normalmente sem as restrincoes de ter um cabo constantemente ligado

Nota: O OLED na imagem não é o utilizado no projeto. O que optei por utilizar foi um OLED 128x32 i2c monochrome ssd1306

## Ligações Arduino bluetooth com cabo Testado, placa integrada em desenvolvimento

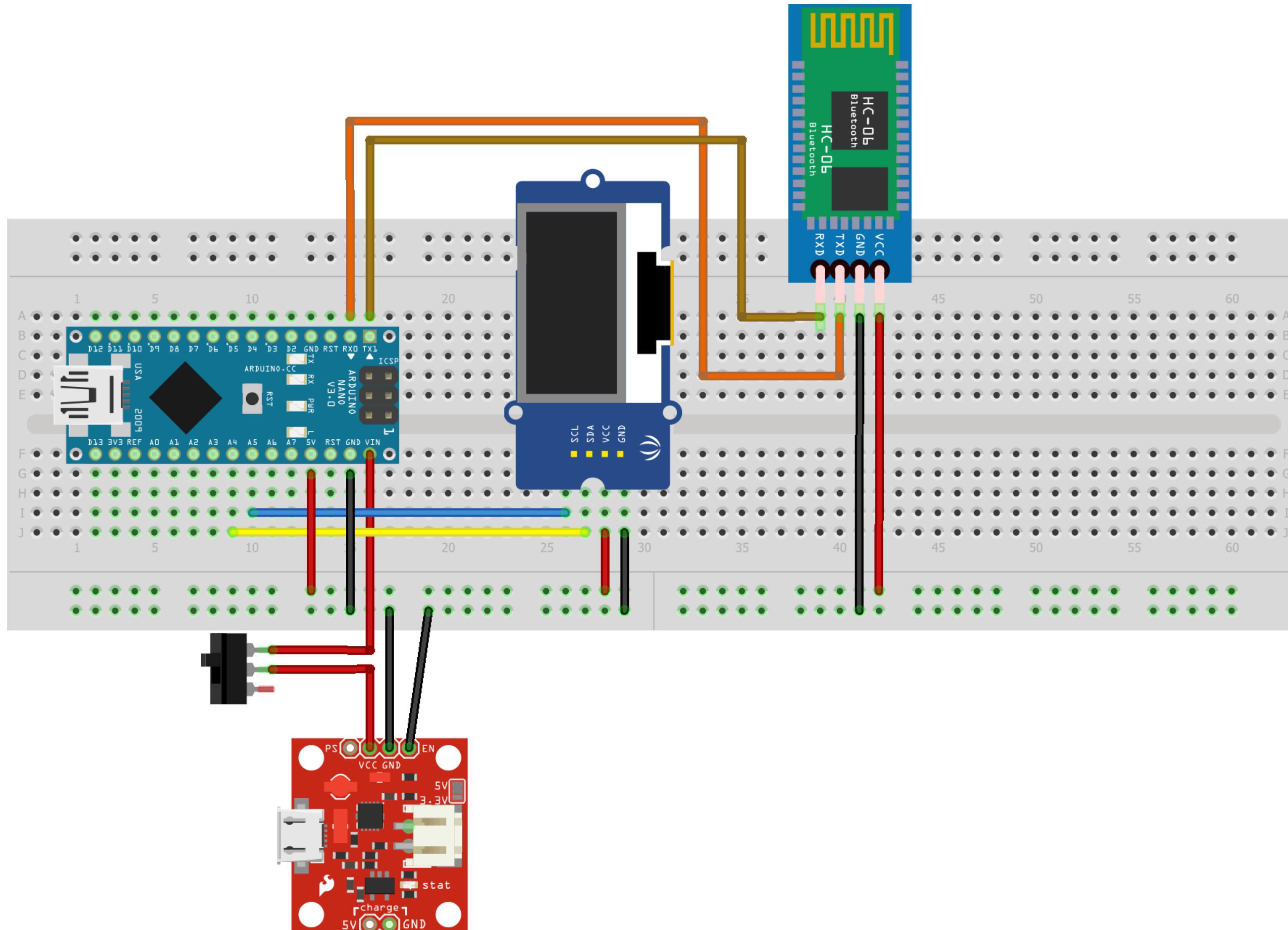


Este modelo é alimentado por um powerbank de telemovel, permite que este seja utilizado normalmente sem as restrincoes de ter um cabo constantemente ligado

Nota: O OLED na imagem não é o utilizado no projeto. O que optei por utilizar foi um OLED 128x32 i2c monochrome ssd1306

## Ligações Arduino Modelo Armação

Não testado



Neste modelo a eletrotécnica encontra-se na totalidade incorporada no monóculo AR.

Tem a vantagem de eliminar na totalidade os cabos.

O resultado é um monóculo mais volumoso e uma redução da autonomia em prol de uma maior mobilidade.

Nota: O OLED na imagem não é o utilizado no projeto. O que optei por utilizar foi um  
OLED 128x32 i2c monochrome ssd1306

fritzing

# Programação Arduino

The screenshot shows the Arduino IDE interface. The title bar reads "Soundview\_v2 | Arduino 1.8.15". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The "Tools" menu is open, showing options like "Auto Format", "Archive Sketch", and "Manage Libraries...". A sub-menu for "Board" is open, listing various Arduino boards. The "Arduino Nano" option is selected. The main code editor area contains C++ code for an SSD1306 display. The status bar at the bottom left says "Done compiling." and the bottom right shows system icons and the date/time "13.09.2021 14:37".

```
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

#define SCREEN_WIDTH 128 // OLED display width, i
#define SCREEN_HEIGHT 32 // OLED display height,
// Declaration for an SSD1306 display connected t
// The pins for I2C are defined by the Wire-libra
// On an arduino UNO:      A4 (SDA), A5 (SCL)
// On an arduino MEGA 2560: 20 (SDA), 21 (SCL)
// On an arduino LEONARDO:  2 (SDA),  3 (SCL), ...
#define OLED_RESET 4 // Reset pin # (or -1 if
#define SCREEN_ADDRESS 0x3C // See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

Sketch uses 19964 bytes (64%) of program storage space. Maximum is 30720 bytes.
Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes.
```

De seguida vamos aprender a copiar o "programa" para o arduino. Pode seguir os passos ou usar o ficheiro disponibilizado no nosso GitHub. Nesse caso, abra o ficheiro e use apenas os passos "bibliotecas" e "Upload"

## Programação Arduino - Board

```
Soundview_v2 | Arduino 1.8.15
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Nano"
Processor: "ATmega328P"
Port: "COM1"
Get Board Info
Programmer: "USBtinyISP"
Burn Bootloader
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

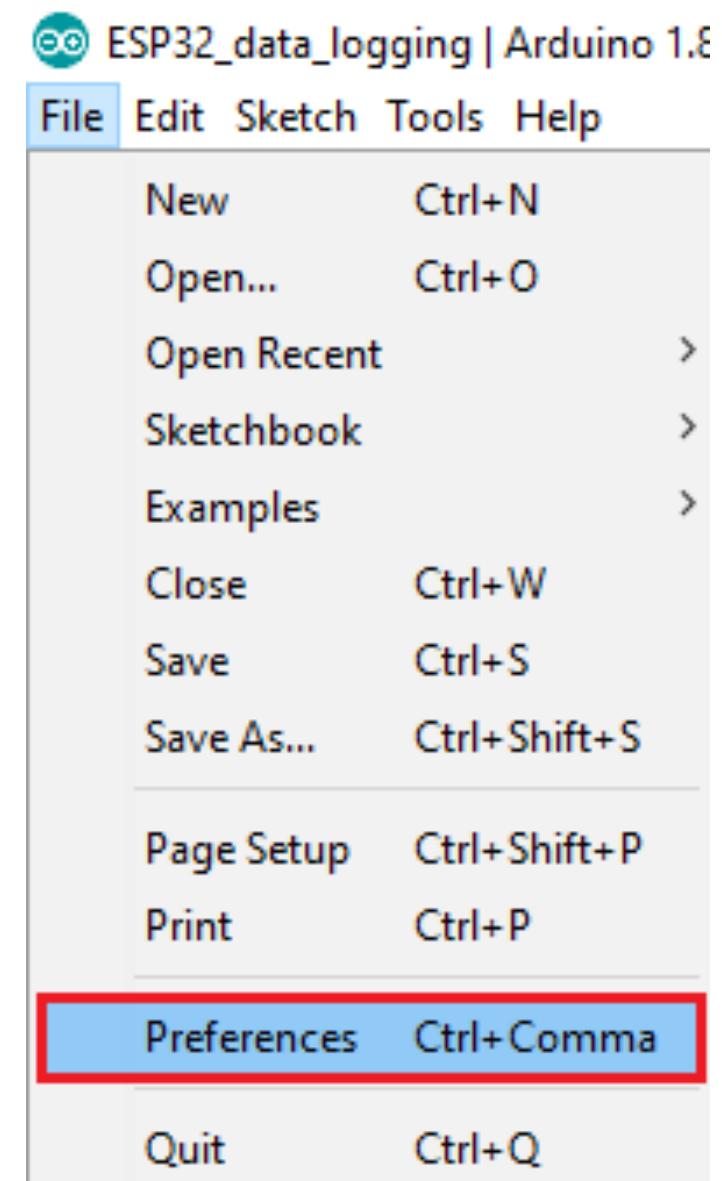
#define SCREEN_WIDTH 128 // OLED display width, i
#define SCREEN_HEIGHT 32 // OLED display height,
// Declaration for an SSD1306 display connected t
// The pins for I2C are defined by the Wire-libra
// On an arduino UNO: A4 (SDA), A5 (SCL)
// On an arduino MEGA 2560: 20 (SDA), 21 (SCL)
// On an arduino LEONARDO: 2 (SDA), 3 (SCL), ...
#define OLED_RESET 4 // Reset pin # (or -1 if
#define SCREEN_ADDRESS 0x3C // See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

Done compiling.

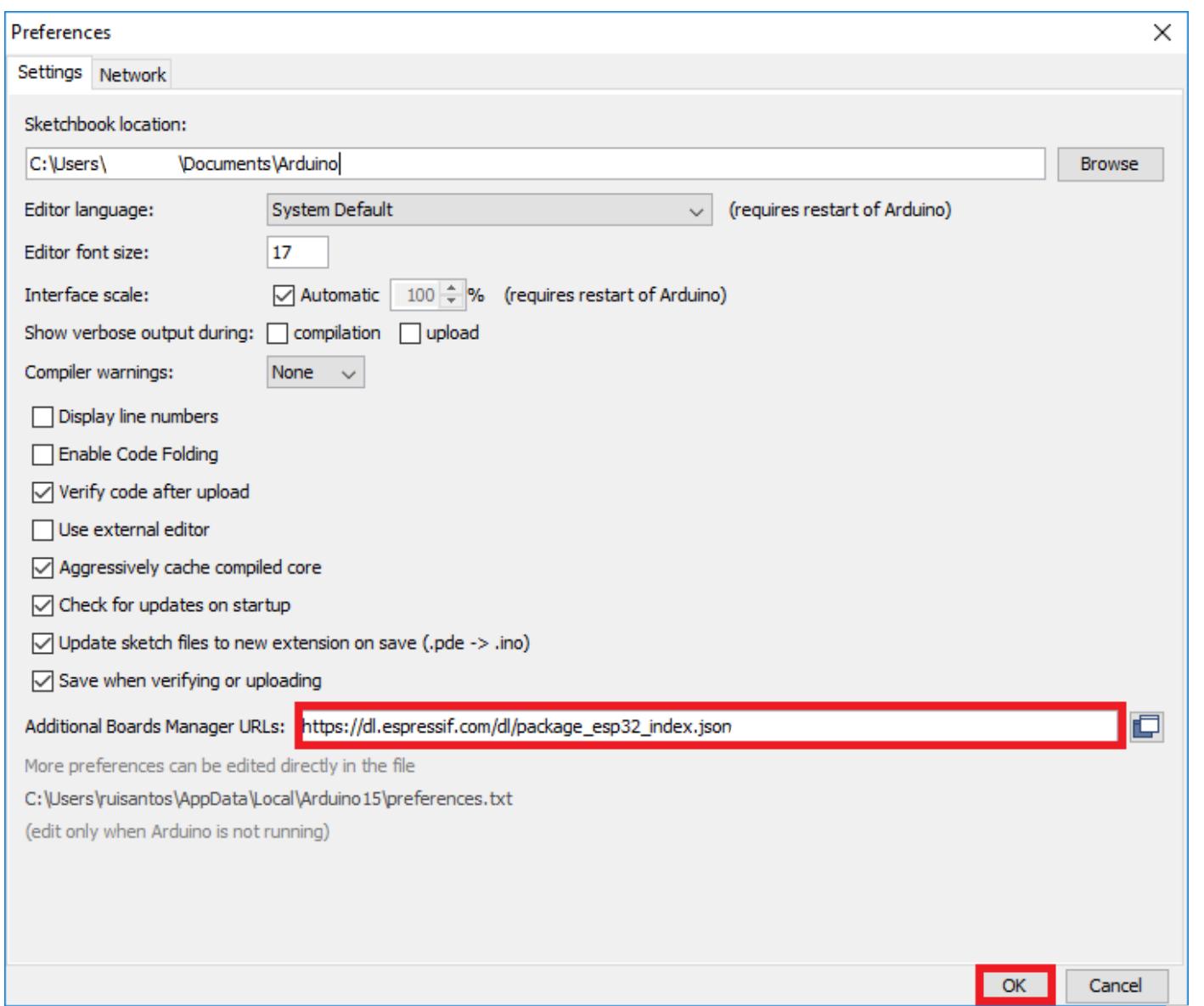
Sketch uses 19964 bytes (64%) of program storage space. Maximum is 30720 bytes.
Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes.
```

Conecte o seu arduino por USB, de seguida abra o Arduino IDE e selecione a "Board" - "Arduino Nano"

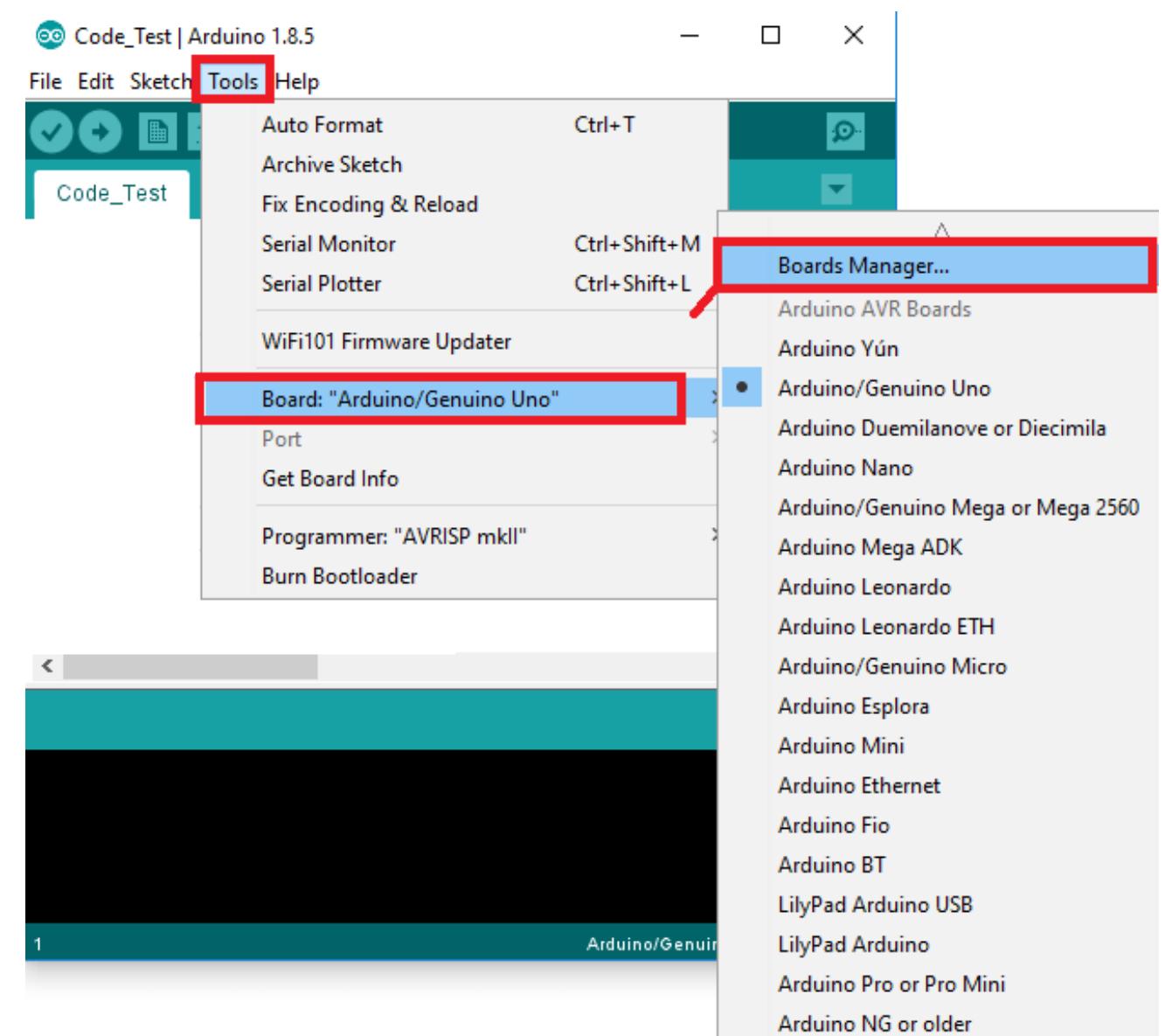
## Programação Arduino - Para ESP32



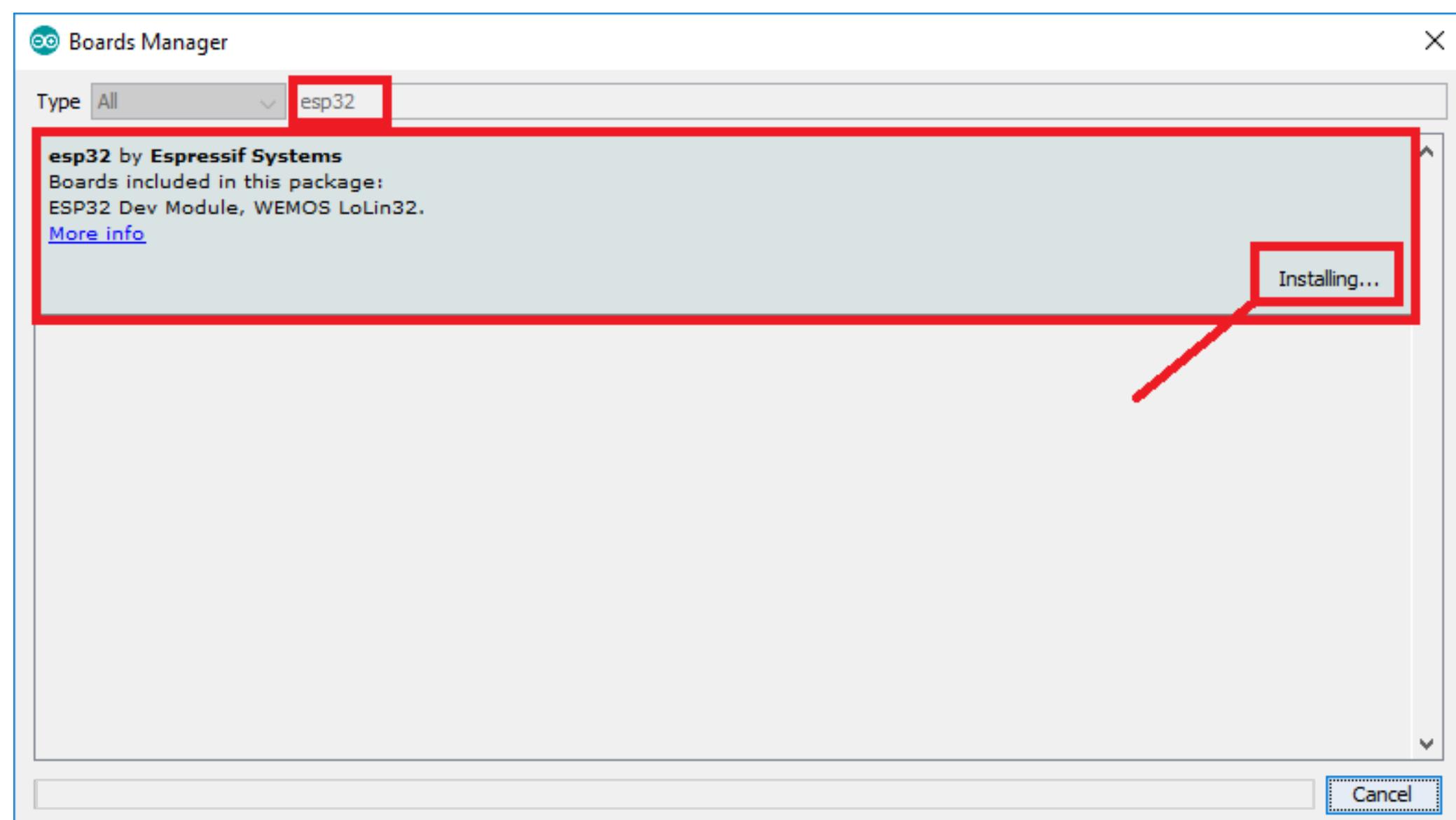
Abra as preferências do arduino



Adicione o seguinte link:  
[https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json)



Abra o seu "Boards Manager"



Procure por "ESP32" e instale, depois selecione nas boards como no exemplo da uno

## Programação Arduino - Programador

Soundview\_v2 | Arduino 1.8.15

File Edit Sketch Tools Help

Auto Format Ctrl+T

Archive Sketch

Fix Encoding & Reload

Manage Libraries... Ctrl+Shift+I

Serial Monitor Ctrl+Shift+M

Serial Plotter Ctrl+Shift+L

WiFi101 / WiFiNINA Firmware Updater

Board: "Arduino Nano" >

Processor: "ATmega328P" >

Port: "COM1" >

Get Board Info

#include "Programmer: "USBtinyISP"

#include "Burn Bootloader"

```
//#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

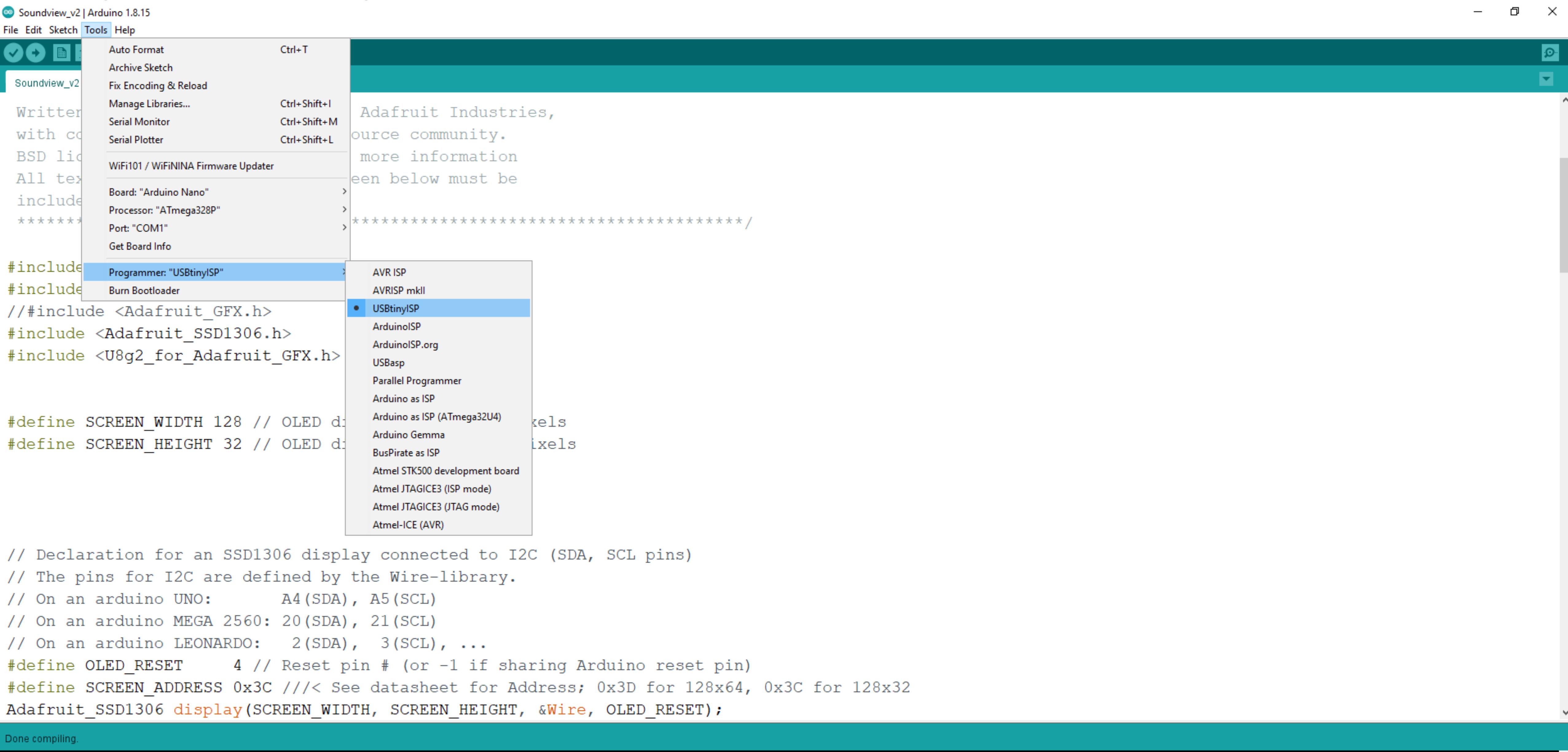
#define SCREEN_WIDTH 128 // OLED d...
#define SCREEN_HEIGHT 32 // OLED d...

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
// The pins for I2C are defined by the Wire-library.
// On an arduino UNO: A4 (SDA), A5 (SCL)
// On an arduino MEGA 2560: 20 (SDA), 21 (SCL)
// On an arduino LEONARDO: 2 (SDA), 3 (SCL), ...
#define OLED_RESET 4 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C // See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

Done compiling.

Sketch uses 19964 bytes (64%) of program storage space. Maximum is 30720 bytes.

Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes.



De seguida selecione o programador "USBtinyISP"



## Programação Arduino - Porta



The screenshot shows the Arduino IDE interface. The title bar reads "Soundview\_v2 | Arduino 1.8.15". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The "Tools" menu is open, showing options like "Auto Format", "Archive Sketch", and "Serial Ports". Under "Serial Ports", "COM1" is selected. The main code editor window displays the following sketch:

```
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
// The pins for I2C are defined by the Wire-library.
// On an arduino UNO:      A4 (SDA), A5 (SCL)
// On an arduino MEGA 2560: 20 (SDA), 21 (SCL)
// On an arduino LEONARDO:  2 (SDA),  3 (SCL), ...
#define OLED_RESET 4 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C // See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

At the bottom of the code editor, it says "Done compiling." Below the code editor, the status bar shows "Sketch uses 19964 bytes (64%) of program storage space. Maximum is 30720 bytes. Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes." The bottom right corner of the status bar also shows "Arduino Nano, ATmega328P on COM1".

Agora seleccione a "COM" onde o seu arduino está ligado. Geralmente é a "COM3"

## Programação Arduino - Bibliotecas

The screenshot shows the Arduino IDE interface. The title bar reads "Soundview\_v2 | Arduino 1.8.15". The menu bar includes "File", "Edit", "Sketch" (which is highlighted in blue), "Tools", and "Help". A status bar at the bottom right shows "Arduino Nano, ATmega328P on COM1", "14:44", "13.09.2021", and a battery icon.

The main code editor window displays the following C++ code:

```
#include <SPI.h>
#include <Wire.h>
//#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 32

// Declaration for an SSD1306
// The pins for I2C are defined
// On an arduino UNO:
// On an arduino MEGA 2560:
// On an arduino LEONARDO:
#define OLED_RESET 4
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(SPI, CS, DC, RST, I2C_ADDRESS, 128, 64);

Done compiling.

Sketch uses 19964 bytes (63%) of program space. Maximum is 30720 bytes.
Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes.
```

A context menu is open over the line "#include <Adafruit\_SSD1306.h>". The menu items are:

- Verify/Compile Ctrl+R
- Upload Ctrl+U
- Upload Using Programmer Ctrl+Shift+U
- Export compiled Binary Ctrl+Alt+S
- Show Sketch Folder Ctrl+K
- Include Library** (highlighted in blue)
- Add File...

Below the menu, a list of available libraries is shown:

- Manage Libraries... Ctrl+Shift+L
- Add .ZIP Library...
- Arduino libraries
- Arduino\_AVRSTL
- Bridge
- EEPROM
- Esplora
- Ethernet
- Firmata
- GSM
- HID
- Keyboard
- LiquidCrystal
- Mouse
- Robot Control
- Robot IR Remote
- Robot Motor
- SD
- SPI
- Servo
- SoftwareSerial
- SpacebrewYun
- Stepper
- TFT
- Temboo
- WiFi
- Wire
- ...
- Contributed libraries
- Adafruit BusIO
- Tiny4kOLED
- TinyOLED-Fonts
- U8g2\_for\_Adafruit\_GFX
- ssd1306

Agora vamos adicionar as Bibliotecas que permitirão utilizar o monitor OLED

## Programação Arduino - Bibliotecas

Soundview\_v2 | Arduino 1.8.15

File Edit Sketch Tools Help

Soundview\_v2

Written by Limor Fried/Ladyada for Adafruit Industries,  
with contributions from the open source community.  
BSD license, check license.txt for more information  
All text above, and the splash screen below must be  
included in any redistribution.

\*\*\*\*\*

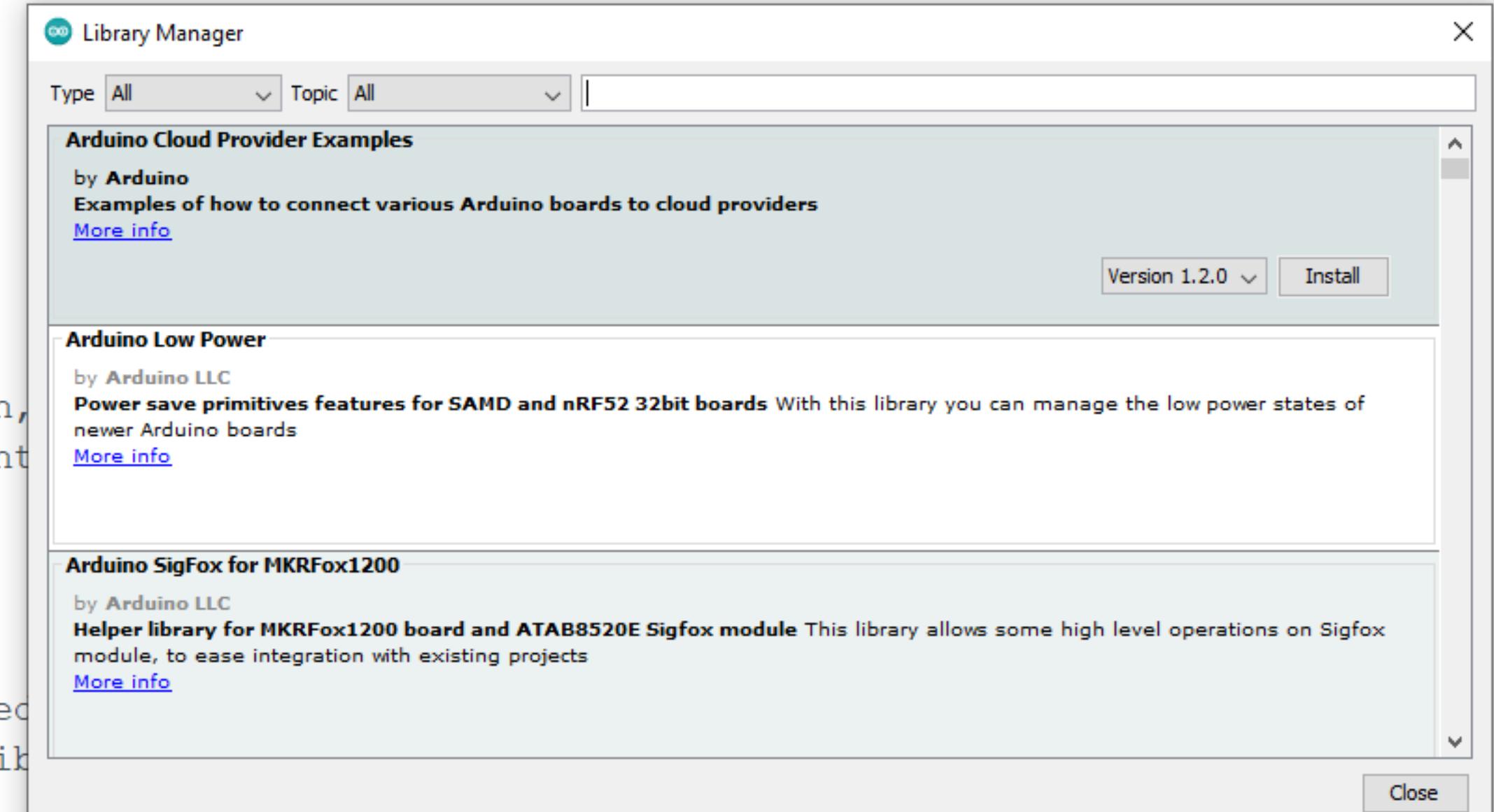
```
#include <SPI.h>
#include <Wire.h>
//#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

#define SCREEN_WIDTH 128 // OLED display width,
#define SCREEN_HEIGHT 32 // OLED display height

// Declaration for an SSD1306 display connected
// The pins for I2C are defined by the Wire-like
// On an arduino UNO: A4(SDA), A5(SCL)
// On an arduino MEGA 2560: 20(SDA), 21(SCL)
// On an arduino LEONARDO: 2(SDA), 3(SCL), ...
#define OLED_RESET 4 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C // See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

Done compiling.

Sketch uses 19964 bytes (64%) of program storage space. Maximum is 30720 bytes.  
Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes.



## Programação Arduino - Inserir o código

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** Soundview\_v2 | Arduino 1.8.15
- Menu Bar:** File Edit Sketch Tools Help
- Toolbar:** Includes icons for Save, Undo, Redo, Open, Save As, Print, and others.
- Code Editor:** The main area contains the Arduino sketch code for Soundview\_v2. The code includes headers for SPI, Wire, Adafruit SSD1306, and U8g2 libraries, defines SCREEN\_WIDTH and SCREEN\_HEIGHT, sets up pins for the OLED display, initializes the display, and handles setup and loop functions. It also includes a Serial.begin call and a loop that prints an error message if the display fails to initialize.
- Status Bar:** Shows "Done compiling."
- Bottom Status Bar:** Displays system information including the sketch size, memory usage, and connection to an Arduino Nano, ATmega328P on COM1.

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

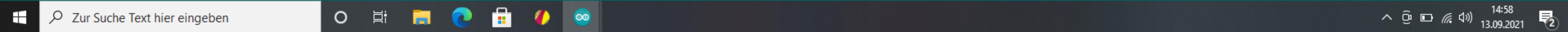
#define OLED_RESET 4
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
U8G2_FOR_ADAFRUIT_GFX u8g2_for_adafruit_gfx;

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

  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }

  u8g2_for_adafruit_gfx.begin(display);
}
```

Sketch uses 19964 bytes (64%) of program storage space. Maximum is 30720 bytes.  
Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes.



Copie o código disponível no manual ou no GitHub da OUVER e cole na pagina principal

14:58  
13.09.2021

## Programação Arduino - Upload

Soundview\_v2 | Arduino 1.8.15

File Edit Sketch Tools Help

Upload Using Programmer

Soundview\_v2 §

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

#define OLED_RESET     4
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
U8G2_FOR_ADAFRUIT_GFX u8g2_for_adafruit_gfx;

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

  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }

  u8g2_for_adafruit_gfx.begin(display);
}

Done compiling.
```

Sketch uses 19964 bytes (64%) of program storage space. Maximum is 30720 bytes.  
Global variables use 595 bytes (29%) of dynamic memory, leaving 1453 bytes for local variables. Maximum is 2048 bytes.



Arduino Nano, ATmega328P on COM1  
15:03 13.09.2021

Pressione o botão de upload e quando terminar está pronto a usar.

## Código Arduino

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_SSD1306.h>
#include <U8g2_for_Adafruit_GFX.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

#define OLED_RESET 4
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
U8G2_FOR_ADAFRUIT_GFX u8g2_for_adafruit_gfx;

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

  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
  // Show initial display buffer contents on the screen --
  // the library initializes this with an Adafruit splash screen.
  u8g2_for_adafruit_gfx.begin(display);

  display.clearDisplay();
}

void loop() {

  String character = "";

  while(Serial.available()) {
    character = Serial.readStringUntil('\n');
  }

  if (character != "") {
    Serial.println(character);

    int string_length = character.length();

    if (string_length <= 21){

      display.clearDisplay();
      u8g2_for_adafruit_gfx.setFontMode(1);
      u8g2_for_adafruit_gfx.setFontDirection(0);
      u8g2_for_adafruit_gfx.setForegroundColor(WHITE);
      u8g2_for_adafruit_gfx.setFont(u8g2_font_t0_12_mf);
      u8g2_for_adafruit_gfx.setCursor(0,10);
      u8g2_for_adafruit_gfx.print(character.substring(0,21));
      display.display();
    }
    else if (string_length > 21 && string_length <= 42){

      display.clearDisplay();
      u8g2_for_adafruit_gfx.setFontMode(1);
      u8g2_for_adafruit_gfx.setFontDirection(0);
      u8g2_for_adafruit_gfx.setForegroundColor(WHITE);
      u8g2_for_adafruit_gfx.setFont(u8g2_font_t0_12_mf);
      u8g2_for_adafruit_gfx.setCursor(0,10);
      u8g2_for_adafruit_gfx.print(character.substring(0,21));
      u8g2_for_adafruit_gfx.setCursor(0,21);
      u8g2_for_adafruit_gfx.print(character.substring(21,string_length));
      display.display();
    }
    else if (string_length > 42 && string_length <= 63){

      display.clearDisplay();
      u8g2_for_adafruit_gfx.setFontMode(1);
      u8g2_for_adafruit_gfx.setFontDirection(0);
      u8g2_for_adafruit_gfx.setForegroundColor(WHITE);
      u8g2_for_adafruit_gfx.setFont(u8g2_font_t0_12_mf);
      u8g2_for_adafruit_gfx.setCursor(0,10);
      u8g2_for_adafruit_gfx.print(character.substring(0,21));
      u8g2_for_adafruit_gfx.setCursor(0,21);
      u8g2_for_adafruit_gfx.print(character.substring(21,42));
      u8g2_for_adafruit_gfx.setCursor(0,32);
      u8g2_for_adafruit_gfx.print(character.substring(42,63));
      display.display();
    }
    else {
      display.clearDisplay();
      u8g2_for_adafruit_gfx.setFontMode(1);
      u8g2_for_adafruit_gfx.setFontDirection(0);
      u8g2_for_adafruit_gfx.setForegroundColor(WHITE);
      u8g2_for_adafruit_gfx.setFont(u8g2_font_t0_12_mf);
      u8g2_for_adafruit_gfx.setCursor(0,10);
      u8g2_for_adafruit_gfx.print(character.substring(0,21));
      u8g2_for_adafruit_gfx.setCursor(0,21);
      u8g2_for_adafruit_gfx.print(character.substring(21,42));
      u8g2_for_adafruit_gfx.setCursor(0,32);
      u8g2_for_adafruit_gfx.print(character.substring(42,60)+ "...");
      display.display();
    }
  }
}
```

## **App Inventor**

Para a criação da APP android vamos utilizar uma ferramenta chamada APP Inventor.

Esta ferramenta não é a ideal para a criação de uma App otimizada mas é perfeita para quem não percebe de programação.

Uma cópia da versão actualizada do programa irá ser fornecida no nosso GitHub, este é um exemplo simplificado para compreender como trabalhar com o programa.

Ao mostrarmos o processo estamos a abrir as portas a que altere a App a seu gosto, tentamos que a nossa seja a mais crua possível por esse motivo.

Futuramente as versões otimizadas criadas em Android studio serão disponibilizadas para download na App Store da Google, mas até lá poderá fazer a sua própria App seguindo o tutorial.

Para tal terá de ir ao seguinte site e criar um conta gratuita.

<https://appinventor.mit.edu/>

## **Modo Offline**

Para utilizarem a tradução em modo offline, têm de fazer download da língua, fazendo os seguintes passos:

- 1 - Abrir as definições do Android
- 2 - Ir a linguagem
- 3 - Selecionar a Opção de Voz Google
- 4 - Selecionar Reconhecimento Offline
- 5 - Fazer o download na linguagem desejada

Nota: Este menu varia nas versões de android, por favor pesquise online para a sua versão, enquanto não fizer um guia para todos, ou tente procurar a opção mais semelhante.

## Código Android Menú

The screenshot shows the MIT App Inventor Designer interface for a project titled "Ouver\_beta1". The central area displays a smartphone screen with two buttons labeled "Bluetooth" and "Serial". The "Properties" panel on the right is highlighted with a red border and contains the following configuration:

- Screen1**
  - AboutScreen
  - AccentColor: White
  - AlignHorizontal: Center : 3
  - AlignVertical: Center : 2
  - AppName: Ouver
  - BackgroundColor: Black
  - BackgroundImage: None...
  - BigDefaultText
  - BlocksToolkit: All
  - CloseScreenAnimation: None
  - DefaultFileScope: App
  - HighContrast
  - Icon: Ouver.png
  - OpenScreenAnimation: Default
  - PrimaryColor: Default
  - PrimaryColorDark: Default
  - ScreenOrientation: Landscape
  - Scrollable
  - ShowListsAsJson
  - ShowStatusBar
  - Sizing
- Media**
  - Ouver.png
  - Upload File ...

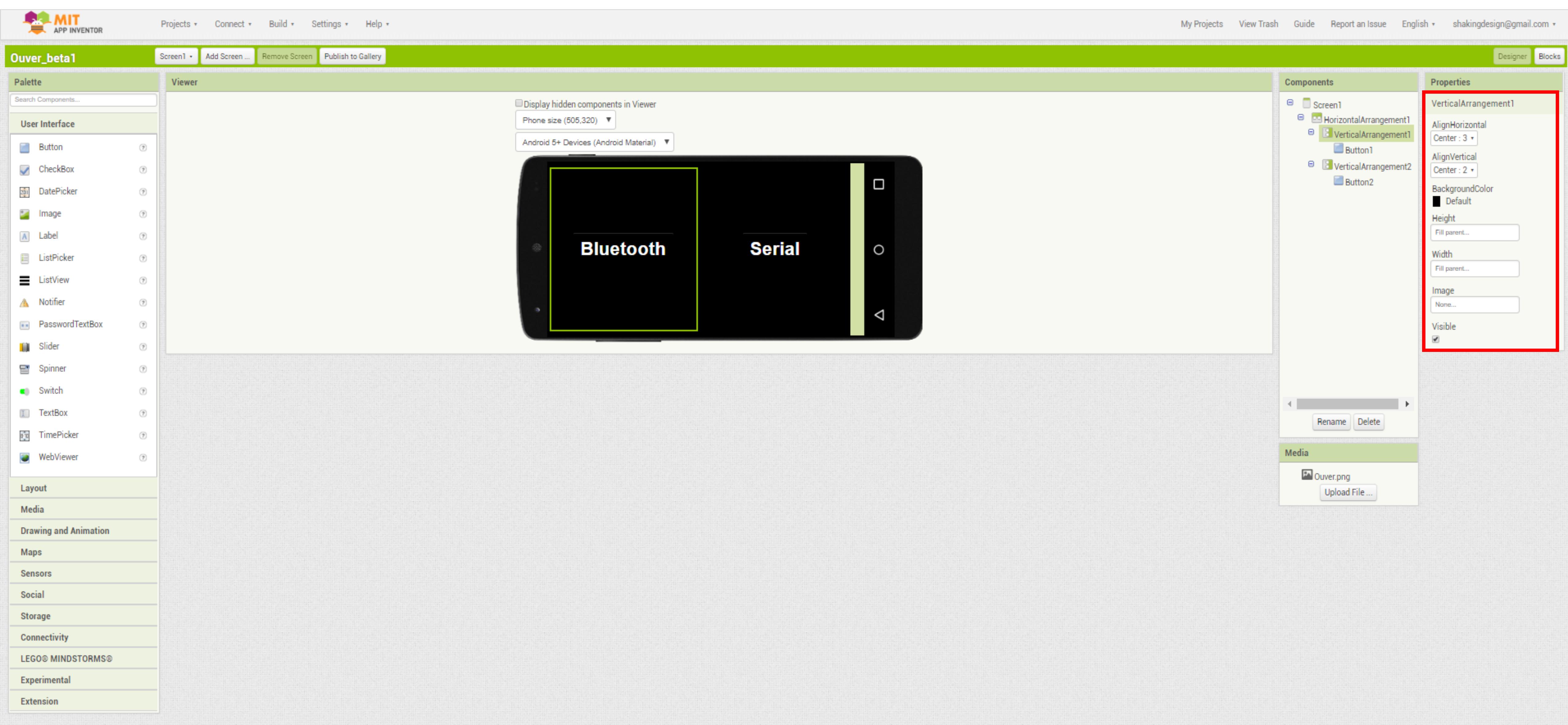
Copie as definições a direita para a sua pagina inicial, só necessita deste menu se quiser ter a opção de ligar por cabo ou por bluetooth. Se só quiser uma opção, recrie apenas a pagina necessária

## Código Android Menu - HorizontalArrangement1

The screenshot shows the MIT App Inventor Designer interface. The top navigation bar includes 'Projects', 'Connect', 'Build', 'Settings', 'Help', 'My Projects', 'View Trash', 'Guide', 'Report an Issue', 'English', and an email address 'shakingdesign@gmail.com'. The main workspace is titled 'Ouver\_beta1' and contains a mobile screen representation. The screen has a black background with white text. On the left side, there is a large button labeled 'Bluetooth'. To its right, separated by a vertical line, is another button labeled 'Serial'. In the bottom right corner of the screen, there is a small circular icon with a triangle symbol. The 'Components' panel on the right lists the screen's structure: 'Screen1' containing a 'HorizontalArrangement1' component, which holds a 'VerticalArrangement1' (containing 'Button1') and a 'VerticalArrangement2' (containing 'Button2'). The 'Properties' panel, which is the focus of the red box, shows settings for 'HorizontalArrangement1': AlignHorizontal (Left: 1), AlignVertical (Top: 1), BackgroundColor (Default), Height (Fill parent...), Width (Fill parent...), Image (None...), and Visible (checked). The 'Media' panel shows an uploaded file named 'Ouver.png'.

Arraste um "HorizontalArrangement" para o ecrã

## Código Android Menu - VerticalArrangement1



Arraste um "VerticalArrangement" para dentro do "HorizontalArrangement1"

## Código Android Menu - Button1

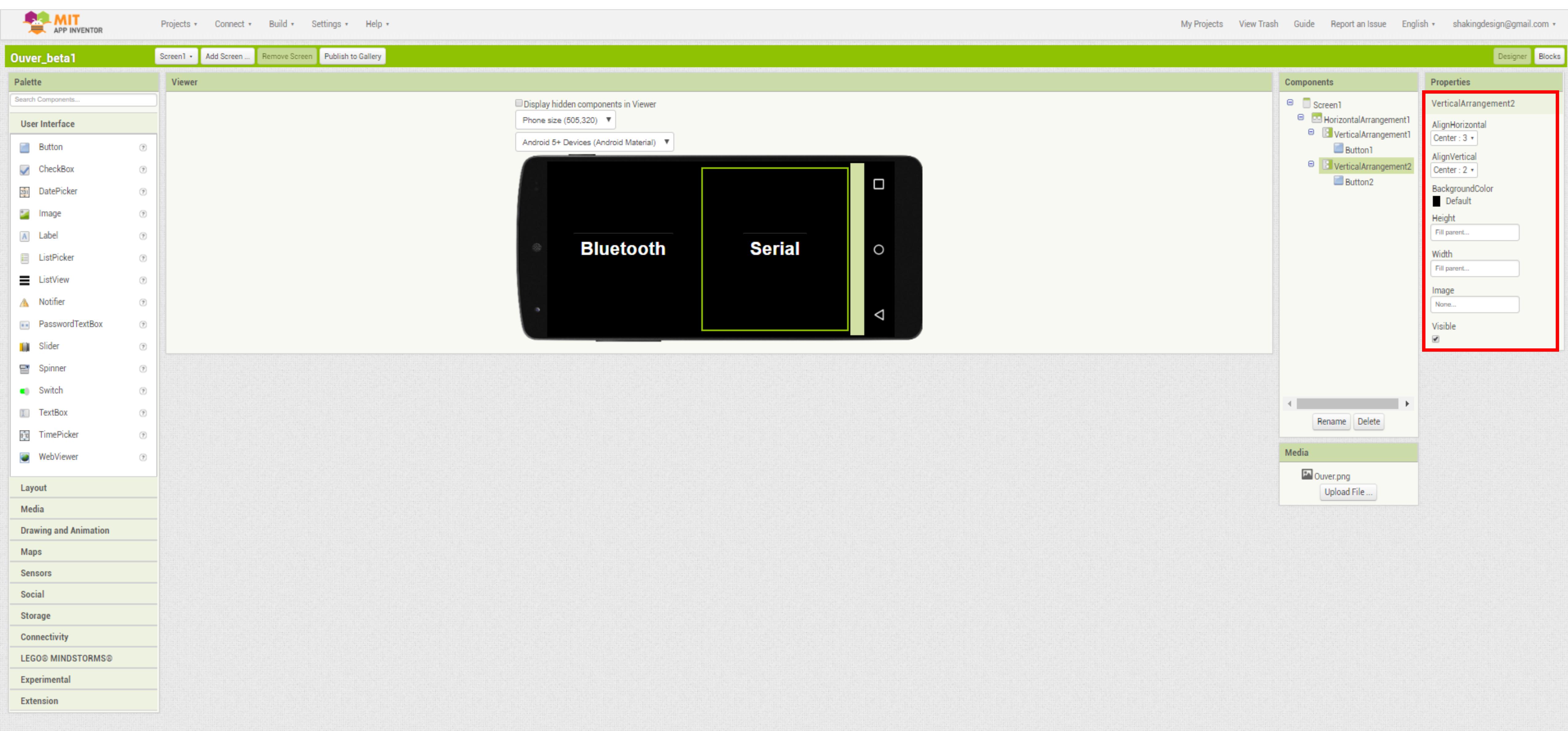
Screenshot of the MIT App Inventor Designer interface showing a mobile screen layout.

The screen layout includes:

- A title bar with "Ouver\_beta1" and tabs for "Screen1", "Add Screen...", "Remove Screen", and "Publish to Gallery".
- A "Components" panel on the right showing a tree structure:
  - Screen1
    - HorizontalArrangement1
      - VerticalArrangement1
        - Button1
      - VerticalArrangement2
        - Button2
- A "Properties" panel on the far right with a red border, showing properties for "Button1":
  - BackgroundColor: None
  - Enabled: checked
  - FontBold: checked
  - FontItalic: unchecked
  - FontSize: 32
  - FontTypeface: default
  - Height: Automatic...
  - Width: Automatic...
  - Image: None...
  - Shape: default
  - ShowFeedback: checked
  - Text: Bluetooth
  - TextAlignment: center
  - TextColor: Default
  - Visible: checked
- A central "Viewer" area showing a smartphone icon with a black background. Inside the phone, there is a white rectangular button labeled "Bluetooth" and the word "Serial" to its right.
- A "Palette" panel on the left containing sections for User Interface (e.g., Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, Switch, TextBox, TimePicker, WebViewer), Layout, Media, Drawing and Animation, Maps, Sensors, Social, Storage, Connectivity, LEGO® MINDSTORMS®, Experimental, and Extension components.

Arraste um "Button" para dentro da "VerticalArrangement"

## Código Android Menu - VerticalArrangement2



Arraste um "VerticalArrangement" para dentro do "HorizontalArrangement"

## Código Android Menu -

Screenshot of the MIT App Inventor Designer interface showing a mobile screen layout.

The screen layout includes:

- Viewer Area:** Displays a smartphone icon representing the screen design. The screen has two main sections: "Bluetooth" on the left and "Serial" on the right.
- Components Panel:** Shows the hierarchical structure of components used:
  - Screen1
    - HorizontalArrangement1
      - VerticalArrangement1
        - Button1
      - VerticalArrangement2
        - Button2
- Properties Panel (highlighted with a red box):** Details for Button2:
  - BackgroundColor: None
  - Enabled: checked
  - FontBold: checked
  - FontItalic: unchecked
  - FontSize: 32
  - FontTypeface: default
  - Height: Automatic...
  - Width: Automatic...
  - Image: None...
  - Shape: default
  - ShowFeedback: checked
  - Text: Serial
  - TextAlignment: center
  - TextColor: Default
  - Visible: checked
- Media Panel:** Shows the file "Ouer.png" uploaded to the screen.

Arraste um "Button" para dentro da "VerticalArrangement"

## Código Android Menu - Blocks

The screenshot shows the MIT App Inventor Blocks editor interface. The top navigation bar includes links for Projects, Connect, Build, Settings, Help, My Projects, View Trash, Guide, Report an Issue, English, and a user email (shakingdesign@gmail.com). The main workspace is titled "Ouver\_beta1". The "Blocks" palette on the left contains categories like Built-in (Control, Logic, Math, Text, Lists, Dictionaries, Colors, Variables, Procedures), Screen1 (HorizontalArrangement1, VerticalArrangement1, Button1, VerticalArrangement2, Button2), and Any component. The "Media" palette shows an uploaded image file "Ouver.png". The central workspace displays the following Scratch-style script:

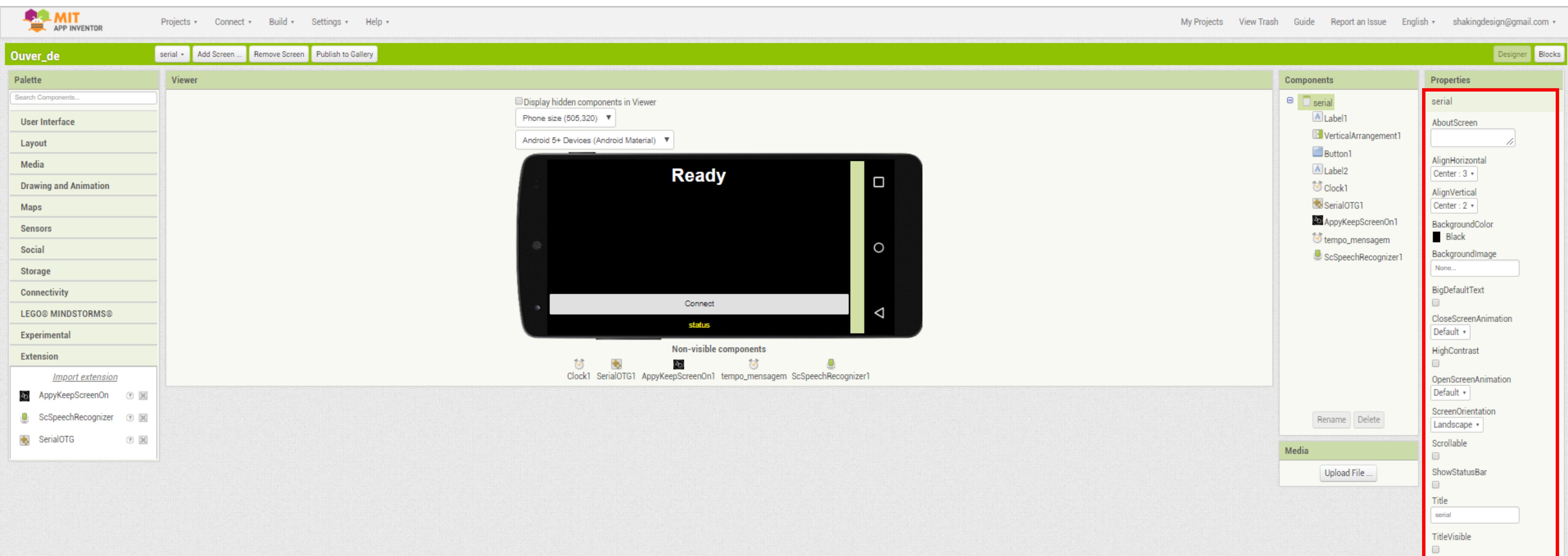
```
when [Button1] .Click
do [open another screen screenName [bluetooth] ]
```

```
when [Button2] .Click
do [open another screen screenName [serial] ]
```

At the bottom of the workspace, there are warning icons (triangle with exclamation, triangle with X) and a "Show Warnings" button. To the right of the workspace are three circular icons (+, -, o) and a trash can icon.

Recrie o programa com as peças oriundas do menu no lado esquerdo

## Código Android Serial - Página Inicial



Crie uma nova pagina chamada "serial" e copie as definições a direita para a sua pagina inicial

## Código Android Serial - Label

The screenshot shows the MIT App Inventor Designer interface. The top navigation bar includes the MIT logo, Projects, Connect, Build, Settings, Help, My Projects, View Trash, Guide, Report an Issue, English, and a user email (shakingdesign@gmail.com). The main workspace is titled "Ouver\_beta1". The left sidebar (Palette) lists categories like User Interface (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, Switch, TextBox, TimePicker, WebViewer), Layout, Media, Drawing and Animation, Maps, Sensors, Social, Storage, Connectivity, LEGO® MINDSTORMS®, Experimental, and Extension. The center (Viewer) shows a smartphone screen with the word "Ready" in a large green box at the top, a "Connect" button below it, and a "status" label at the bottom. Below the phone are "Non-visible components": Clock1, SerialOTG1, AppyKeepScreenOn1, tempo\_mensagem, and ScSpeechRecognizer1. The right side (Components and Properties) shows a tree view of components under "serial": VerticalArrangement1, Button1, Label2, Clock1, SerialOTG1, AppyKeepScreenOn1, tempo\_mensagem, and ScSpeechRecognizer1. A specific "Label1" component is selected and highlighted with a red border. The Properties panel for "Label1" shows the following settings:

label1	BackgroundColor	None
label1	FontBold	<input checked="" type="checkbox"/>
label1	FontItalic	<input type="checkbox"/>
label1	FontSize	32
label1	FontTypeface	default
label1	HTMLFormat	<input type="checkbox"/>
label1	HasMargins	<input checked="" type="checkbox"/>
label1	Height	Automatic...
label1	Width	Automatic...
label1	Text	Ready
label1	TextAlignment	center : 1
label1	TextColor	White
label1	Visible	<input checked="" type="checkbox"/>

Arraste uma "Label" para o ecrã

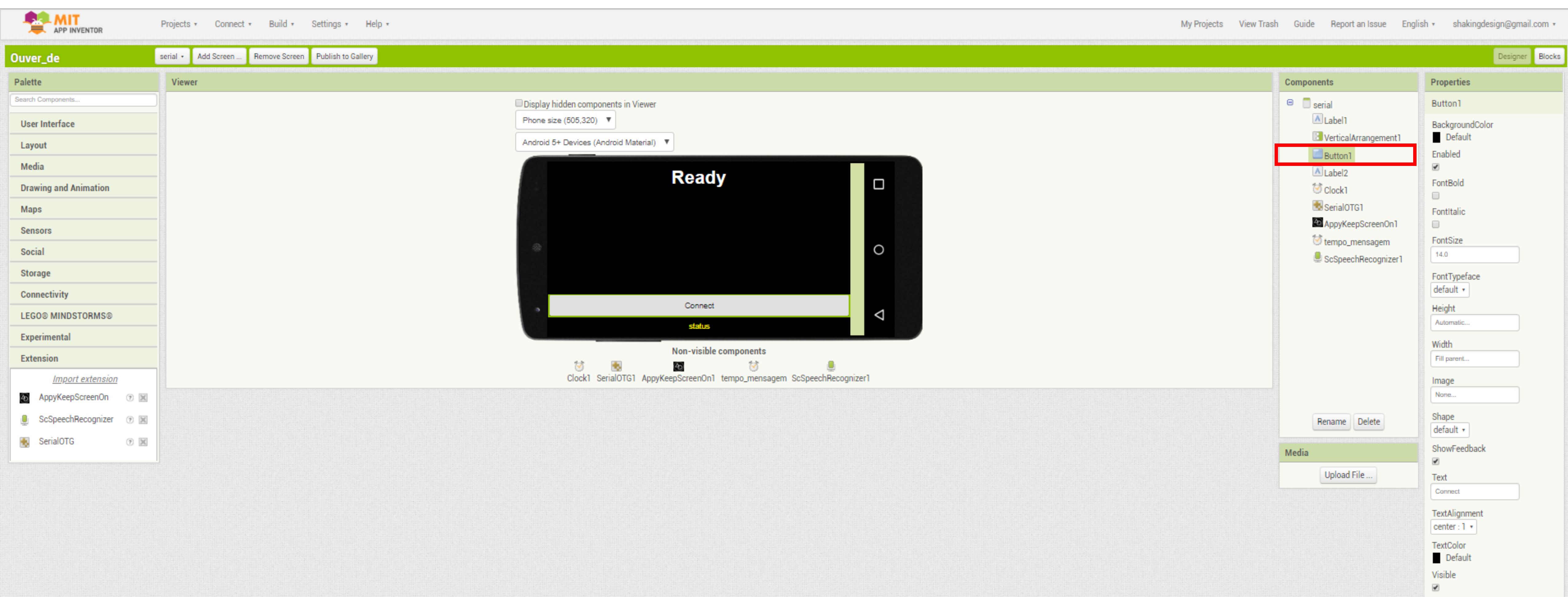
## Código Android Serial - Arrangement

The screenshot shows the MIT App Inventor Designer interface. The top navigation bar includes the MIT logo, user account information (My Projects, View Trash, Guide, Report an Issue, English, shakingdesign@gmail.com), and menu items (Projects, Connect, Build, Settings, Help). The main workspace is titled "Ouver\_de" and contains the following elements:

- Palette:** On the left, it lists categories like User Interface, Layout, Media, Drawing and Animation, Maps, Sensors, Social, Storage, Connectivity, LEGO® MINDSTORMS®, Experimental, and Extension. Under Extension, there are components: AppyKeepScreenOn, ScSpeechRecognizer, and SerialOTG.
- Viewer:** The central area displays a smartphone screen with the word "Ready" at the top. Below it is a large empty rectangular area, followed by a "Connect" button and a "status" label. A vertical green bar is visible on the right side of the screen.
- Components:** A tree view on the right shows the project structure:
  - serial
    - label1
    - VerticalArrangement1 (highlighted with a red border)
    - Button1
    - Label2
    - Clock1
    - SerialOTG1
    - AppyKeepScreenOn1
    - tempo\_mensagem
    - ScSpeechRecognizer1
- Properties:** A panel on the far right lists properties for the selected component (VerticalArrangement1):
  - VerticalArrangement1
  - AlignHorizontal: Center : 3
  - AlignVertical: Center : 2
  - BackgroundColor: Default
  - Height: Fill parent...
  - Width: Fill parent...
  - Image: None...
  - Visible: checked
- Media:** A section at the bottom right allows for file upload.

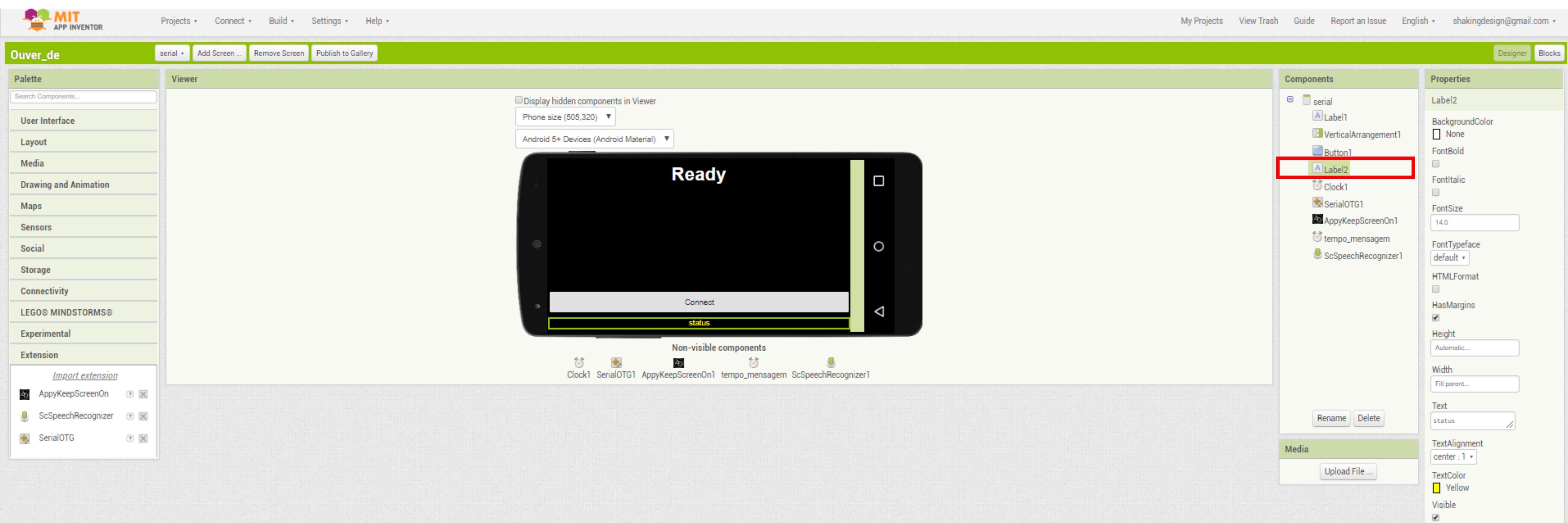
Arraste um "verticalArrangement" para baixo da "label"

## Código Android Serial - Clock



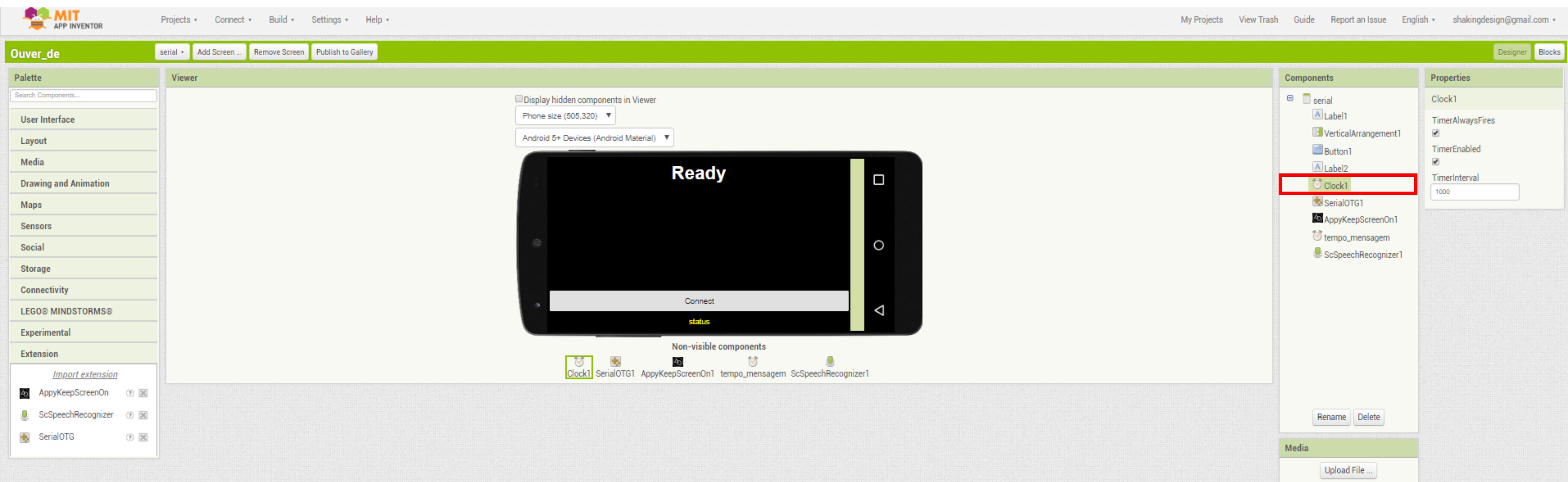
Adicione um butao para estabelecer a conexão

## Código Android Serial - Label 2



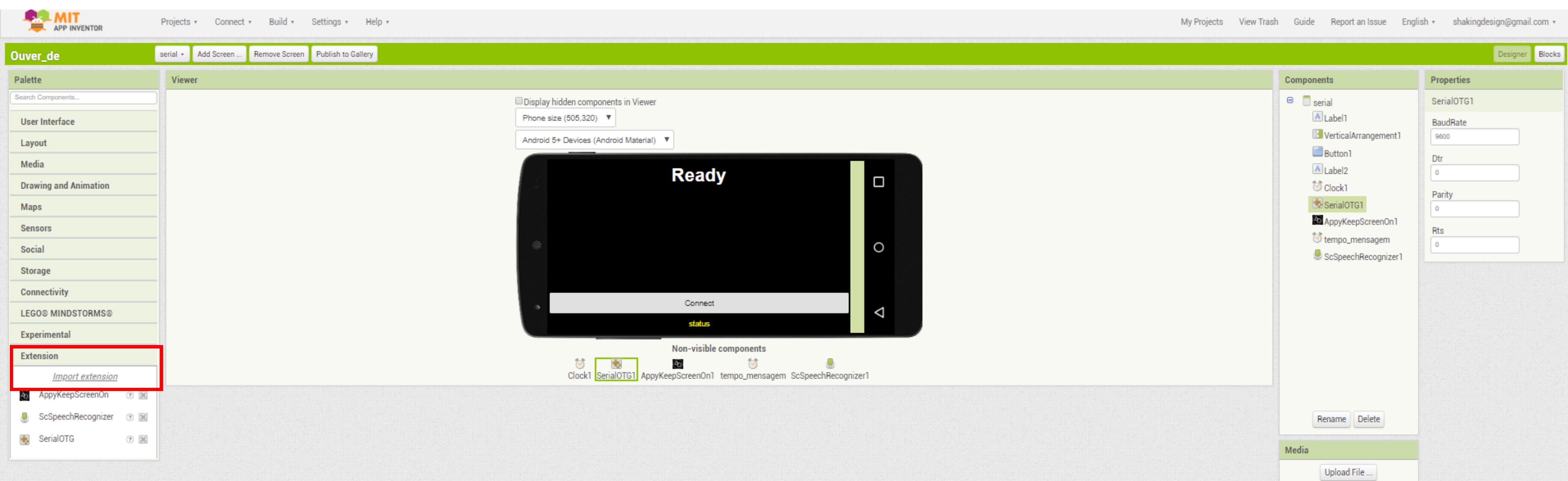
Adicione uma "label" para apresentar se a conexão ao Monoculo Ar foi efectuada

## Código Android Serial - Clock



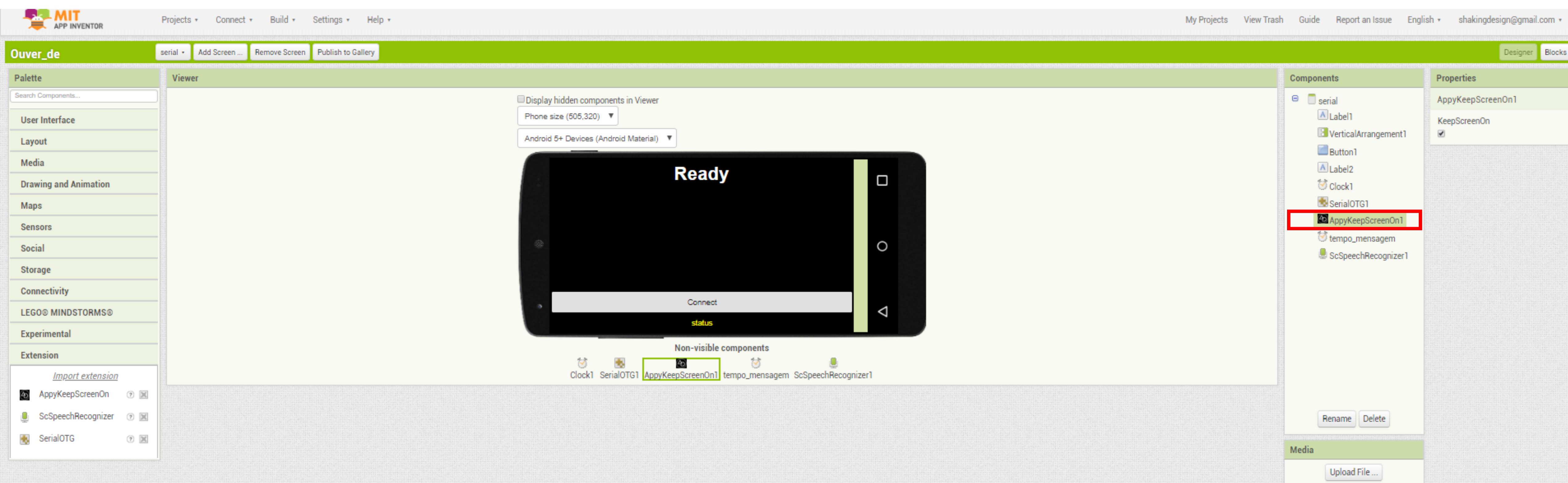
Adicione um "clock" para verificar se a conexão se mantém activa

## Código Android Serial - Serial OTG



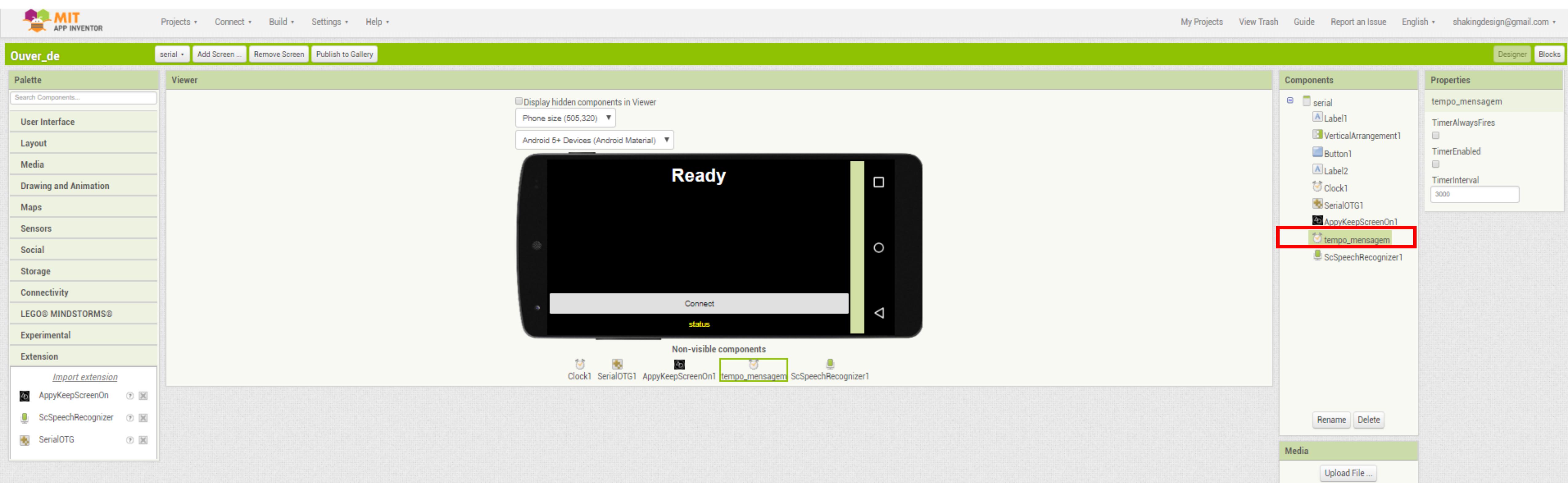
Faça download do modulo "Serial OTG" em <https://github.com/rkl099/AppInventor-SerialOTG> e importe no separador "extensions"  
Adicione o modulo de conexão OTG "SerialOTG"

## Código Android Serial - KeepScreenON



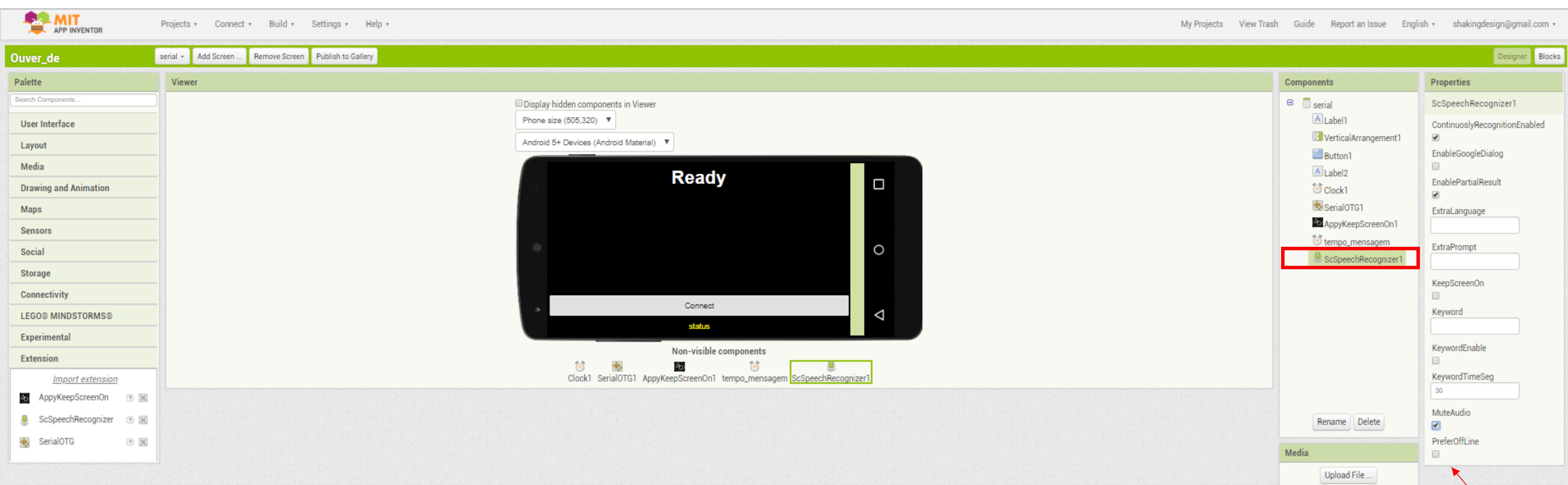
Faça download do modulo "Keep Screen On" em <https://community.thunkable.com/t/free-extension-keep-screen-on/67770>  
Adicione o modulo de manutenção de ecrã activo "ApplyKeepScreenOn" do menu "Extensions"

## Código Android Serial



Adicione mais um "Clock" para controlar a duração que o texto é mostrado

## Código Android Serial - SpeechRecognizer



Para utilizar em modo online tem de activar esta função  
Futuramente irei incluir um botão para activar e desactivar esta função  
Em modo offline a captação não é tão eficiente.

Adicione o modulo de reconhecimento de fala "SCSpeechRecognizer" do menu "Extensions"  
<https://community.kodular.io/t/free-voice-recognition-extension-without-google-dialogue/82347/2>

## Código Android Serial - Blocks

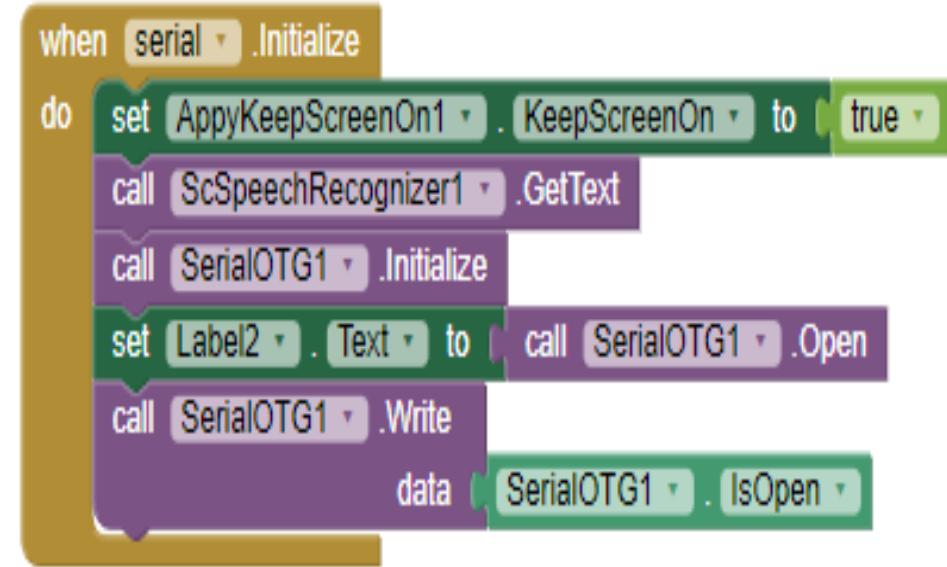
The screenshot shows the MIT App Inventor Blocks editor interface. The top navigation bar includes 'Projects', 'Connect', 'Build', 'Settings', 'Help', 'My Projects', 'View Trash', 'Guide', 'Report an Issue', 'English', and an email address 'shakingdesign@gmail.com'. The main workspace is titled 'Ouver\_de' and contains several blocks programs:

- Top Block:** A sequence of two 'initialize global' blocks: 'initialize global tester to " " ' and 'initialize global resultado to " " '.
- Serial Initialization:** A 'when serial.Initialize' event loop. It sets 'AppyKeepScreenOn1.KeepScreenOn' to true, calls 'ScSpeechRecognizer1.GetText', calls 'SerialOTG1.Initialize', sets 'Label2.Text' to 'call SerialOTG1.Open', and calls 'SerialOTG1.Write' with 'SerialOTG1.IsOpen' as data.
- Button Click:** A 'when Button1.Click' event loop. It calls 'SerialOTG1.Initialize', sets 'Label2.Text' to 'call SerialOTG1.Open', and calls 'SerialOTG1.Write' with 'SerialOTG1.IsOpen' as data.
- Clock Timer:** A 'when Clock1.Timer' event loop. It checks if 'SerialOTG1.IsOpen'. If true, it sets 'Label2.Text' to 'Connected' and 'Label2.TextColor' to green. If false, it sets 'Label2.Text' to 'Disconnected' and 'Label2.TextColor' to red.
- Speech Recognition:** A 'when ScSpeechRecognizer1.AfterGettingText [result]' event loop. It checks if 'ScSpeechRecognizer1.Result' is not empty. If true, it calls 'ScSpeechRecognizer1.GetText'. If false, it sets 'global resultado' to 'get result', sets 'Label1.Text' to 'get global resultado', and checks if 'Label1.Text' is not equal to 'get global tester'. If true, it sets 'global tester' to 'Label1.Text', calls 'SerialOTG1.Write' with 'Label1.Text' as data, and sets 'tempo\_mensagem.TimerEnabled' to true. Finally, it calls 'ScSpeechRecognizer1.GetText'.
- Tempo Mensagem Timer:** A 'when tempo\_mensagem.Timer' event loop. It checks if 'Label1.Text' is not equal to 'get global tester'. If true, it sets 'global tester' to 'Label1.Text', sets 'Label1.Text' to 'get global tester', calls 'SerialOTG1.Write' with 'Label1.Text' as data, and sets 'tempo\_mensagem.TimerEnabled' to true.

The left sidebar lists 'Blocks' categories: Built-in (Control, Logic, Math, Text, Lists, Dictionaries, Colors), and custom blocks for 'serial' (Label1, VerticalArrangement1, Button1, Label2, Clock1, SerialOTG1, AppyKeepScreenOn1). The bottom left shows warning icons and a 'Show Warnings' button. The bottom right features standard UI controls (circle, plus, minus, trash).

Recrie o programa com as peças oriundas do menu no lado esquerdo

## Código Android Serial - Explicação



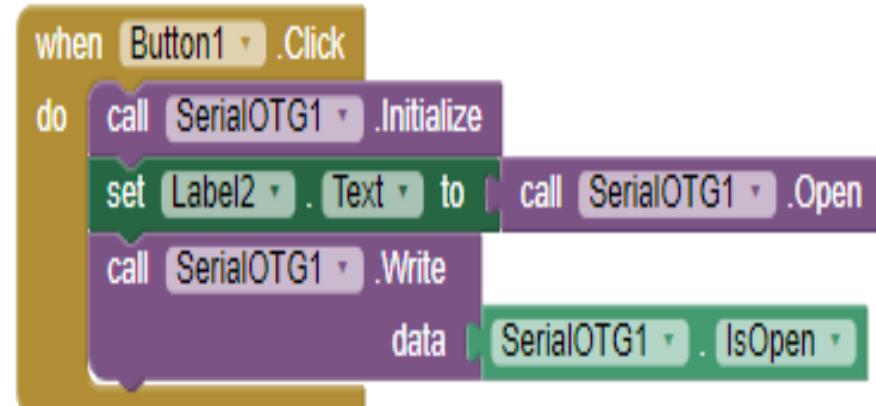
Um problema com o MIT APPinventor é que as suas apps geralmente não correm em segundo plano e alguns dos seus elementos deixam de funcionar se o telemóvel bloquear.

Para evitar isso coloquei o KeepScreenOn, para que o telemóvel não bloqueie.

Isto varia de versão de android para android e devem testar com a vossa. No meu, a versão 9, mesmo com o ecrã bloqueado ou se sair dela ela continua a funcionar e só para caso abra uma segunda aplicação.

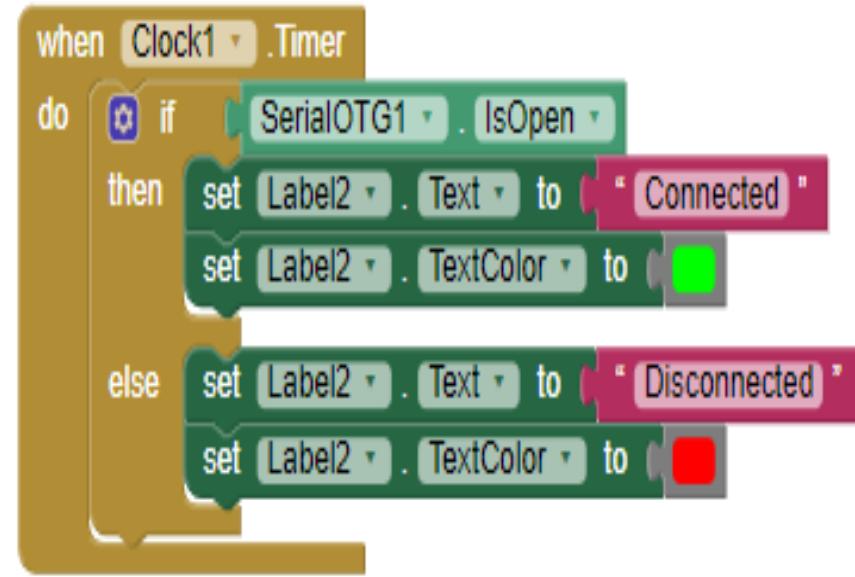
O código seguinte pede para que o utilizador dê permissão para ligar ao dispositivo por usb.

Apesar do MIT APPinventor ter a função de comunicação serial, este não é compatível com algumas boards de arduino, dai ter de ir buscar uma biblioteca extra para garantir esta compatibilidade

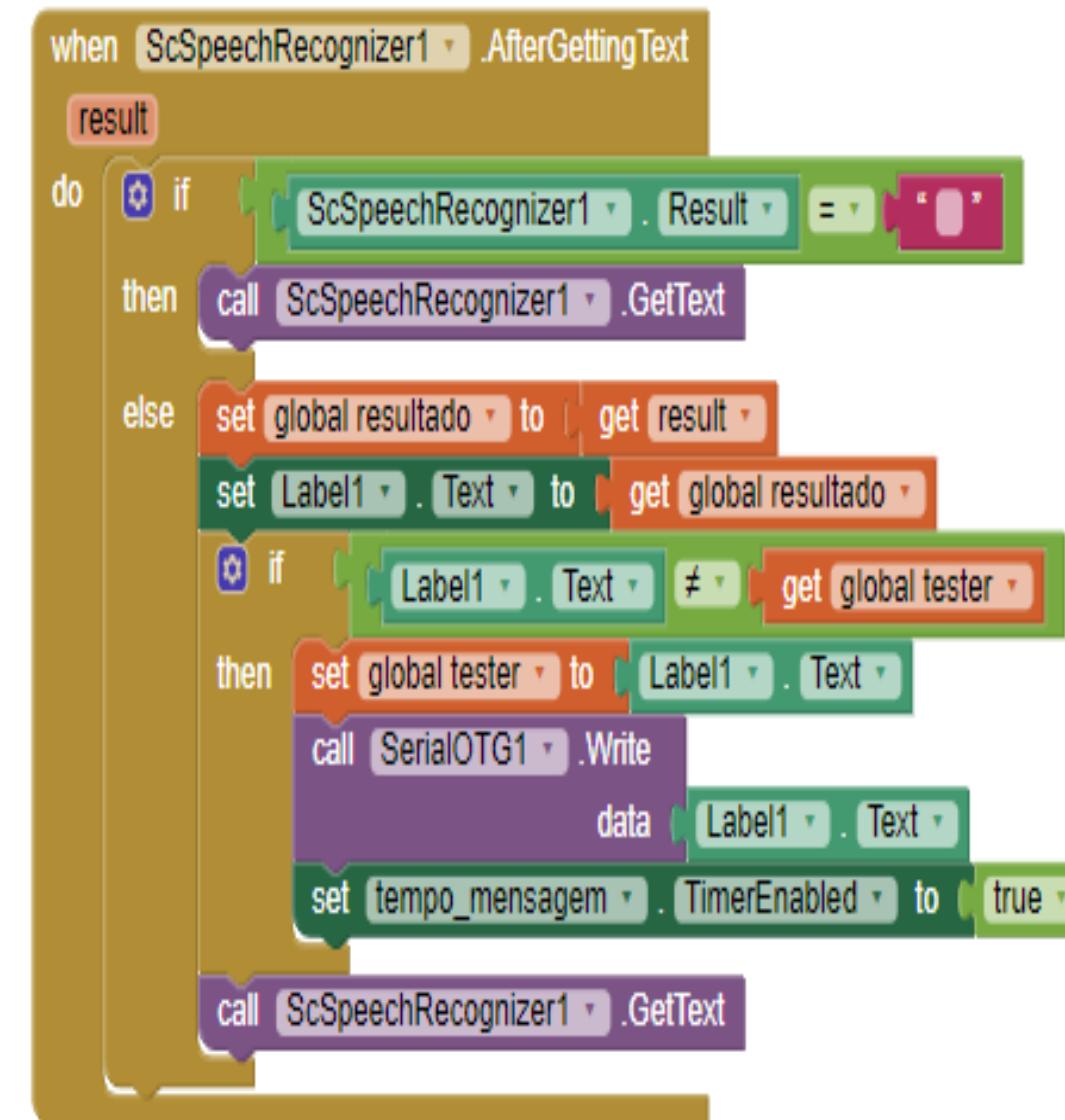


Adicionei um botão para fazer a ligação caso esta falhe e mostrar no monóculo que esta foi estabelecida

## Código Android Serial - Explicação



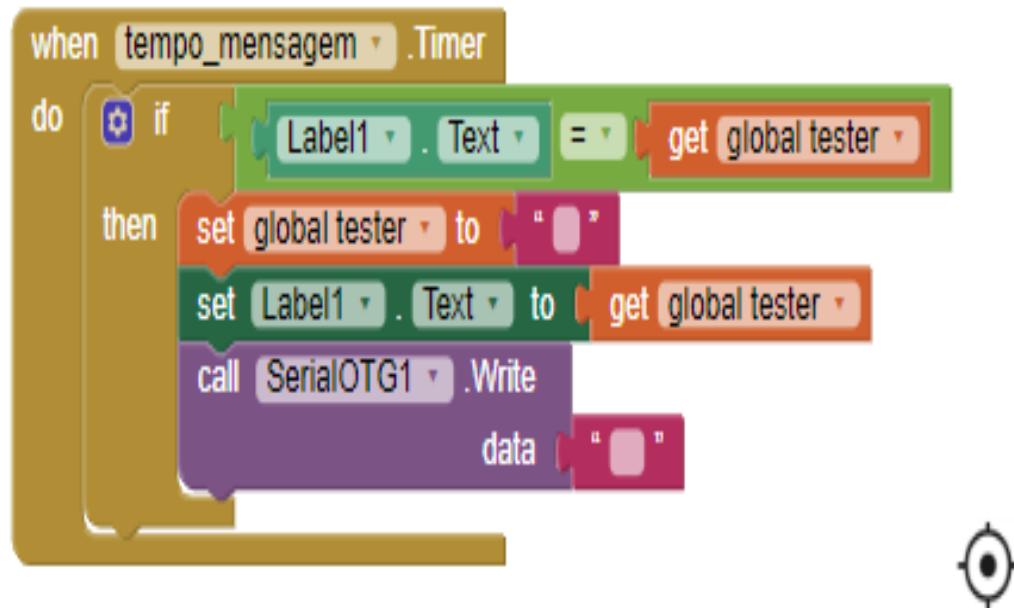
Para manter a ligação activa tive de colocar um timer a testar se está conectada com um curto intervalo de tempo, sem isto, a mesma tinha a tendência a deixar de funcionar



Neste bloco ocorre a principal função.

Eu utiliei uma biblioteca externa de captacao de voy porque a que vem com MAi produz um som sempre que inicia o processo, teoricamente com ela seria possivel ter a captação ininterrupta, mas infelizmente comigo não funcionou, por isso tive que criar uma função que se nao reconhecesse a voy, teria de reiniciar o processo, senão, iria mostrar o resultado no ecrã e enviar o mesmo para o monoculo.

## Código Android Serial - Explicação



Esta função apaga a mensagem do monoculo ao final de algum tempo de forma a que este poupe energia.

initialize global [tester] to [ ]

initialize global [resultado] to [ ]

Estas são as variaveis onde é testado se o texto não foi alterado nos ultimos segundo o que resulta no apagar do mesmo no monoculo

## Código Android Bluetooth - Página Inicial

Screenshot of the MIT App Inventor Designer interface showing the initial screen of a project titled "Ouver\_beta1".

The screen displays a smartphone-like viewer with the word "Ready!" at the top. Below it, there is a text input field containing "Text for ListPicker1" and a message "Not connected" below it.

The Components panel on the right shows the following components:

- bluetooth (selected):
  - Label
  - VerticalArrangement1
  - ListPicker1
  - Label1
  - BluetoothClient1
  - Clock1
  - AppyKeepScreenOn1
  - clear\_message
  - ScSpeechRecognizer1

The Properties panel on the far right is highlighted with a red box and shows the properties for the selected "bluetooth" component:

- bluetooth
- AboutScreen
- AlignHorizontal: Right : 2
- AlignVertical: Top : 1
- BackgroundColor: Black
- BackgroundImage: None...
- BigDefaultText
- CloseScreenAnimation: Default
- HighContrast
- OpenScreenAnimation: Default
- ScreenOrientation: Landscape
- Scrollerable
- ShowStatusBar
- Title: bluetooth
- TitleVisible

The palette on the left lists various components categorized by type (User Interface, Layout, Media, etc.) and provides search functionality.

Crie uma nova pagina chamada "bluetooth" e copie as definições a direita para a sua pagina inicial

## Código Android Bluetooth - Label 2

Screenshot of the MIT App Inventor Designer interface showing a mobile screen with components and their properties.

The screen displays the following components:

- A large **Label** at the top center containing the text **Ready!**.
- A **ListPicker** below it with the placeholder text **Text for ListPicker1**.
- A **Text** component below the ListPicker with the text **Not connected**.
- Non-visible components listed below the screen:
  - BluetoothClient1
  - Clock1
  - AppyKeepScreenOn1
  - clear\_message
  - ScSpeechRecognizer1

The Properties panel on the right shows the configuration for the **Label** component, which has the ID **Label2**. The properties are as follows:

- BackgroundColor**: None
- FontBold**: checked
- FontItalic**: unchecked
- FontSize**: 32
- FontTypeface**: default
- HTMLFormat**: unchecked
- HasMargins**: checked
- Height**: Automatic...
- Width**: Fill parent...
- Text**: Ready!
- TextAlignment**: center : 1
- TextColor**: White
- Visible**: checked

Arraste uma "Label" para o ecrã

## Código Android Bluetooth - VerticalArrangement1

The screenshot shows the MIT App Inventor Designer interface. The top navigation bar includes the MIT logo, project tabs (Projects, Connect, Build, Settings, Help), user account information (My Projects, View Trash, Guide, Report an Issue, English, shakingdesign@gmail.com), and toolbars for adding screens, removing screens, and publishing.

The main workspace is titled "Ouver\_beta1". It features a "Viewer" section displaying a smartphone screen with the text "Ready!" and "Text for ListPicker1" below it, followed by "Not connected". Below the phone are "Non-visible components" including BluetoothClient1, Clock1, AppyKeepScreenOn1, clear\_message, and ScSpeechRecognizer1.

The "Components" panel lists the following components:

- bluetooth
- Label
- VerticalArrangement1
- ListPicker1
- Label
- BluetoothClient1
- Clock1
- AppyKeepScreenOn1
- clear\_message
- ScSpeechRecognizer1

The "Properties" panel for the selected "VerticalArrangement1" component shows the following settings:

- VerticalArrangement1
- AlignHorizontal: Center : 3
- AlignVertical: Center : 2
- BackgroundColor: Default
- Height: Fill parent...
- Width: Fill parent...
- Image: None...
- Visible: checked

Buttons for "Rename" and "Delete" are also present in the Properties panel.

Arraste um "verticalArrangement" para baixo da "label"

## Código Android Bluetooth - ListPicker1

Screenshot of the MIT App Inventor Designer interface showing a mobile phone screen with a "Ready!" message and a "Text for ListPicker1" input field. The "Not connected" status is displayed below the input field. The Properties panel on the right is highlighted with a red border, focusing on the ListPicker1 component settings.

The Properties panel shows the following configuration for ListPicker1:

- BackgroundColor: Default
- Enabled: checked
- FontBold: unchecked
- FontItalic: unchecked
- FontSize: 14.0
- FontTypeface: default
- Height: Automatic...
- Width: Fill parent...
- Image: None...
- ItemBackgroundColor: Default
- ItemTextColor: Default
- Selection:
- Shape: default
- ShowFeedback: checked
- ShowFilterBar: unchecked
- Text: Text for ListPicker1
- TextAlignment: center : 1
- TextColor: Default
- Title:
- Visible: checked

Arraste um "ListPicker" para poder seleccionar o dispositivo bluetooth

## Código Android Bluetooth - Label1

The screenshot shows the MIT App Inventor Designer interface for a project titled "Ouver\_beta1". The project is using the "bluetooth" component. The phone screen displays the text "Ready!" at the top and "Not connected" at the bottom of a white bar. The "Properties" panel on the right is highlighted with a red box and shows the configuration for the "Label1" component, which has a red text color and is currently visible.

**Properties Panel (highlighted in red):**

- Label1
- BackgroundColor: None
- FontBold: False
- FontItalic: False
- FontSize: 14.0
- FontTypeface: default
- HTMLFormat: False
- HasMargins: True
- Height: Automatic...
- Width: Fill parent...
- Text: Not connected
- TextAlignment: center : 1
- TextColor: Red
- Visible: True

Adicione uma "label" para apresentar se a conexão ao Monoculo Ar foi efectuada

## Código Android Bluetooth - Bluetooth

Screenshot of the MIT App Inventor Designer interface showing a project titled "Ouver\_beta1".

The project contains a screen named "bluetooth" which displays a smartphone with the text "Ready!" on its screen. Below the phone, there is a message bar with "Text for ListPicker1" and "Not connected".

The Components panel shows the following components:

- bluetooth (Main screen)
- Label (Text for ListPicker1)
- VerticalArrangement1
- ListPicker1
- Label1
- BluetoothClient1 (highlighted with a red border)
- Clock1
- AppyKeepScreenOn1
- clear\_message
- ScSpeechRecognizer1

The Properties panel for BluetoothClient1 shows the following settings:

- CharacterEncoding: UTF-8
- DelimiterByte: 0
- DisconnectOnError: unchecked
- HighByteFirst: unchecked
- Secure: checked

The Media panel has an "Upload File..." button.

Adicione o modulo de conexão bluetooth

## Código Android Bluetooth - Clock1

The screenshot shows the MIT App Inventor Designer interface for a project titled "Ouver\_beta1". The main area displays a smartphone screen with the word "Ready!" centered. Below it is a white bar containing the text "Text for ListPicker1" and the message "Not connected". The "Components" panel on the right lists several components, including "bluetooth", "Label", "VerticalArrangement1", "ListPicker1", "Label1", "BluetoothClient1", "Clock1" (which is highlighted with a red border), "AppyKeepScreenOn1", "clear\_message", and "ScSpeechRecognizer1". The "Properties" sub-panel for "Clock1" shows the following settings: "TimerAlwaysFires" and "TimerEnabled" are checked, and "TimerInterval" is set to 1000. The "Media" panel at the bottom right contains a "Upload File..." button.

bluetooth Add Screen ... Remove Screen Publish to Gallery

My Projects View Trash Guide Report an Issue English shakingdesign@gmail.com

Ouver\_beta1

Palette

User Interface

- Button
- CheckBox
- DatePicker
- Image
- Label
- ListPicker
- ListView
- Notifier
- PasswordTextBox
- Slider
- Spinner
- Switch
- TextBox
- TimePicker
- WebViewer

Layout

Media

Drawing and Animation

Maps

Sensors

Social

Storage

Connectivity

LEGO® MINDSTORMS®

Experimental

Extension

Viewer

Display hidden components in Viewer

Phone size (505,320)

Android 5+ Devices (Android Material)

Ready!

Text for ListPicker1

Not connected

Non-visible components

BluetoothClient1 Clock1 AppyKeepScreenOn1 clear\_message ScSpeechRecognizer1

Components

- bluetooth
- Label
- VerticalArrangement1
- ListPicker1
- Label1
- BluetoothClient1
- Clock1**
- AppyKeepScreenOn1
- clear\_message
- ScSpeechRecognizer1

Properties

Clock1

TimerAlwaysFires

TimerEnabled

TimerInterval

1000

Rename Delete

Media

Upload File...

Adicione um "clock" para verificar se a conexão se mantém activa

## Código Android Bluetooth - KeepAlwaysOn

The screenshot shows the MIT App Inventor Designer interface for a project titled "Ouver\_beta1". The interface is divided into several panels:

- Top Bar:** Projects, Connect, Build, Settings, Help, My Projects, View Trash, Guide, Report an Issue, English, shakingdesign@gmail.com.
- Left Sidebar (Palette):** Search Components, User Interface (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, Switch, TextBox, TimePicker, WebViewer), Layout, Media, Drawing and Animation, Maps, Sensors, Social, Storage, Connectivity, LEGO® MINDSTORMS®, Experimental, Extension.
- Central Area (Viewer):** Shows a smartphone screen with the text "Ready!" and a placeholder "Text for ListPicker1" below it. Below the screen, a status bar shows "Not connected". A list of non-visible components is shown at the bottom: BluetoothClient1, Clock1, AppyKeepScreenOn1 (highlighted with a green border), clear\_message, ScSpeechRecognizer1.
- Right Side Panels:**
  - Components:** A tree view of components including bluetooth, Label, VerticalArrangement1, ListPicker1, Label1, BluetoothClient1, Clock1, AppyKeepScreenOn1 (highlighted with a green border), clear\_message, ScSpeechRecognizer1.
  - Properties:** Shows the properties for the AppyKeepScreenOn1 component, specifically the "KeepScreenOn" checkbox, which is checked (indicated by a red border).
  - Media:** A section for uploading files.

Faça download do modulo "Keep Screen On" em <https://community.thunkable.com/t/free-extension-keep-screen-on/67770>  
Adicione o modulo de manutenção de ecrã activo "ApplyKeepScreenOn" do menu "Extensions"

## Código Android Bluetooth - Clock Clear Message

The screenshot shows the MIT App Inventor Designer interface for a project titled "Ouver\_beta1". The main area displays a smartphone screen with the word "Ready!" centered. Below it is a white bar with the placeholder text "Text for ListPicker1". At the bottom of the screen, the status bar shows "Not connected". The Components panel on the right lists several components: bluetooth, Label, VerticalArrangement1, ListPicker1, Label1, BluetoothClient1, Clock1, AppyKeepScreenOn1, clear\_message, and ScSpeechRecognizer1. The "clear\_message" component is currently selected, as indicated by a red border around its entry in the Components list. In the Properties panel, the "TimerInterval" property for "clear\_message" is set to 3000.

Adicione mais um "Clock" para controlar a duração que o texto é mostrado

## Código Android Bluetooth - SpeechRecognizer

The screenshot shows the MIT App Inventor Designer interface. The project is titled "Ouver\_beta1". The top menu bar includes "Projects", "Connect", "Build", "Settings", "Help", "My Projects", "View Trash", "Guide", "Report an Issue", "English", and an email address "shakingdesign@gmail.com". The Designer tab is selected.

The left sidebar "Palette" lists various components under categories like User Interface (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, Switch, TextBox, TimePicker, WebViewer), Layout, Media, Drawing and Animation, Maps, Sensors, Social, Storage, Connectivity, LEGO® MINDSTORMS®, Experimental, and Extension.

The "Components" panel on the right lists the following components:

- bluetooth
- Label
- VerticalArrangement1
- ListPicker1
- Label1
- BluetoothClient1
- Clock1
- AppyKeepScreenOn1
- clear\_message
- ScSpeechRecognizer1

The "Properties" panel on the far right shows the properties for the ScSpeechRecognizer1 component, which are highlighted with a red border. These properties include:

- ContinuouslyRecognitionEnabled: checked
- EnableGoogleDialog: unchecked
- EnablePartialResult: checked
- ExtraLanguage: (empty)
- ExtraPrompt: (empty)
- KeepScreenOn: unchecked
- Keyword: (empty)
- KeywordEnable: unchecked
- KeywordTimeSeg: 30
- MuteAudio: checked
- PreferOffLine: unchecked

Adicione o modulo de reconhecimento de fala "SCSpeechRecognizer" do menu "Extensions"  
<https://community.kodular.io/t/free-voice-recognition-extension-without-google-dialogue/82347/2>

## Código Android Bluetooth - Blocks

MIT APP INVENTOR

Projects ▾ Connect ▾ Build ▾ Settings ▾ Help ▾ My Projects View Trash Guide Report an Issue English ▾ shakingdesign@gmail.com ▾

Ouver\_beta1

bluetooth ▾ Add Screen ... Remove Screen Publish to Gallery Designer Blocks

**Blocks**

- Built-in
  - Control
  - Logic
  - Math
  - Text
  - Lists
  - Dictionaries
  - Colors
  - Variables
  - Procedures
- bluetooth
  - Label2
  - VerticalArrangement1
  - ListPicker1
  - Label1
  - BluetoothClient1
  - Clock1
  - AppyKeepScreenOn1

Rename Delete

Media

Upload File ...

Viewer

```

script1:
    initialize global [resultado] to [""]
    initialize global [tester] to [false]

    when [bluetooth v].Initialize
        do
            set [AppyKeepScreenOn1 v].KeepScreenOn to [true v]
            call [ScSpeechRecognizer1 v].GetText

    when [ListPicker1 v].BeforePicking
        do
            set [ListPicker1 v].Elements to [BluetoothClient1 v].AddressesAndNames

    when [ListPicker1 v].AfterPicking
        do
            if [call [BluetoothClient1 v].Connect address [ListPicker1 v].Selection] then
                set [ListPicker1 v].Elements to [BluetoothClient1 v].AddressesAndNames
                call [BluetoothClient1 v].SendText text ["connected"]

    when [Clock1 v].Timer
        do
            if [BluetoothClient1 v].IsConnected then
                set [Label1 v].Text to ["Connected"]
                set [Label1 v].TextColor to [green]
            else
                set [Label1 v].Text to ["Disconnected"]
                set [Label1 v].TextColor to [red]

script2:
    when [ScSpeechRecognizer1 v].AfterGettingText [result]
        do
            if [ScSpeechRecognizer1 v].Result = [tester] then
                call [ScSpeechRecognizer1 v].GetText
            else
                set [global resultado] to [get result]
                set [Label2 v].Text to [get global resultado]
                if [Label2 v].Text ≠ [get global tester] then
                    set [global tester] to [Label2 v].Text
                    call [BluetoothClient1 v].SendText text [Label2 v].Text
                    set [clear_message v].TimerEnabled to [true v]
                    call [ScSpeechRecognizer1 v].GetText

    when [clear_message v].Timer
        do
            if [Label1 v].Text = [get global tester] then
                set [global tester] to [false]
                set [Label1 v].Text to [get global tester]
                call [BluetoothClient1 v].SendText text [false]

```

Show Warnings

Recrie o programa com as peças oriundas do menu no lado esquerdo - o principio é o mesmo explicado no Serial

## Lente

A lente usada foi retirada de uns óculos VR. Tanto os óculos como as lentes podem ser adquiridos online. Aconselhamos sites como o aliexpress ou banggod para as obter ao menor custo.  
Deve procurar usando os termos "Óculos VR" ou "Lentes VR"

Óculos de papel são ideais por serem mais fáceis de retirar as lentes.

Nota: Não tenho qualquer afiliação com estes sites.



Lente biconvex para google, lentes de 25mm x 45mm para google papelão óculos vr 3d de alta qualidade

★★★★★ 4.7 ✓ 28 avaliações 75 pedidos

Enjoy special discounts!

€ 1,62 / lote (2 itens)  
€ 2,16 -25%

Ofertas para Novos Usuários

Preço incluindo IVA  
Desconto Instantâneo : € 0,87 de desconto em € 25,11 ✓  
€ 1,73 Cupons para Você Pegue seu desconto

Quantidade:  
- 1 + Adicional 2% desc. (10 lots ou mais)  
9937 lots disponíveis

Frete Grátis  
para Germany via Cainiao Super Economy Global ✓  
Estimativa de Entrega: 11 Nov. ⓘ

Comprar agora Adicione ao carrinho

Proteção ao Consumidor de 75 Dias Garantia de Reembolso

Reembolso Garantido Comprou, mas não gostou? Devolva dentro de 15 dias

Exemplo lente VR



Exemplo Óculos VR

Óculos de realidade virtual para google, óculos vr 3d de alta configuração para celular iphone

★★★★★ 4.5 ✓ 13 avaliações 76 pedidos

€ 2,24 - 2,34 € 3,03 - 3,16 -26%

Preço incluindo IVA  
€ 1,73 Cupons para Você € 0,87 desc. a cada € 27,71 Pegue seu desconto

Envio de:  
CHINA A polónia Estados Unidos Espanha austrália  
França Itália

Quantidade:  
- 1 + Adicional 1% desc. (2 itens ou mais)  
342 itens disponíveis

Por favor selecione o país de origem do envio

Comprar agora Adicione ao carrinho

## Custo

Para calcular os custos vamos utilizar valores médios do aliexpress como referência.

Versão com cabo	Versão bluetooth ESP32	Versão bluetooth com cabo	Versão bluetooth (incompleta)
Arduino Nano	- 3,5€	ESP32	- 9,5€
Lentes pack de 2	- 2€	Lentes pack de 2	- 2€
Oled	- 2,5€	Oled	- 2,5€
1m cabo dupont 4 fios	- 1€	1m cabo dupont 4 fios	- 1€
Cabo OTG	- 2,5€	total	Bluetooth
total	- 11,5€	- 15,5€	- 4,5€
		total	Modulo de carregamento
		- 13,5€	- 16€
			Bluetooth
			- 4,5€
			total
			- 29,5€

## **Três versões de produção**

### **O objectivos:**

- Disponibilizar o monóculo terminado por um preço igual ou inferior ao custo da versão DIY, garantindo assim um custo ajustado.
- Criar um objeto mais confortável, de dimensões mais reduzidas e com maior autonomia, recorrendo para isso a criação de placas integradas específicas e a armações produzidas por métodos industriais.
- Caso possível oferecer mais funções quando a poupança na produção industrial permitir.

### **Versão Lite**

Oculos com ligação por cabo

### **Versão Bluetooth Standard**

Oculos com ligação por cabo ao modulo bluetooth

### **Versão BLuetooth Pro**

Oculos com ligação por bluetooth com microfone incorporado

## **Objectivos futuros**

### **Armação**

#### A curto prazo

Criação da armação para ambas as versões em formato STL para impressão 3D

#### Médio/Longo prazo

Alteração da lente para uma que reduza a distância necessária ao OLED

Criação de um modelo para produção por injeção

### **Versão Lite/ Bluetooth com cabo**

#### A curto prazo

Criar a caixa onde se encontrará a eletrónica

#### Médio prazo

Substituir a placa arduino nano por uma attiny85

#### Longo Prazo

Desenvolver integrado próprio já com ligação OTG

### **Versão Bluetooth Pro**

#### Longo Prazo

Definir eletrónica

## **Objectivos futuros**

### **APP Android**

#### Curto prazo

Otimizar o código em Android Studio

#### Médio prazo

Modo conferencia, poder dar a hipótese de por exemplo, numa sala de aulas um professor ter um telemóvel a receber o áudio e a converter o texto e de seguida a enviar para vários alunos.

Isso poderá ajudar com problemas criados pela captação de sons a longa distância ou em espaços com condições sonoras desfavoráveis

#### Longo prazo

Reconhecimento de sons de emergência (sirenes, buzinas, estrondos)

Adicionar a possibilidade de receber as notificações do telemóvel no ecrã de forma a dar mais utilidade

Adicionar a função de tradução

## **Código Arduino**

#### Curto prazo

Otimizar

#### Médio/longo prazo

Abandonar o uso de bibliotecas em prol de código escrito de raiz

## **Comunidade**

#### Curto prazo

Página do Facebook, Instagram e blog do projeto

GitHub

#### Médio prazo

Manual detalhado de construção

Apresentação do projeto a Associações de pessoas com deficiência Auditiva

#### Longo prazo

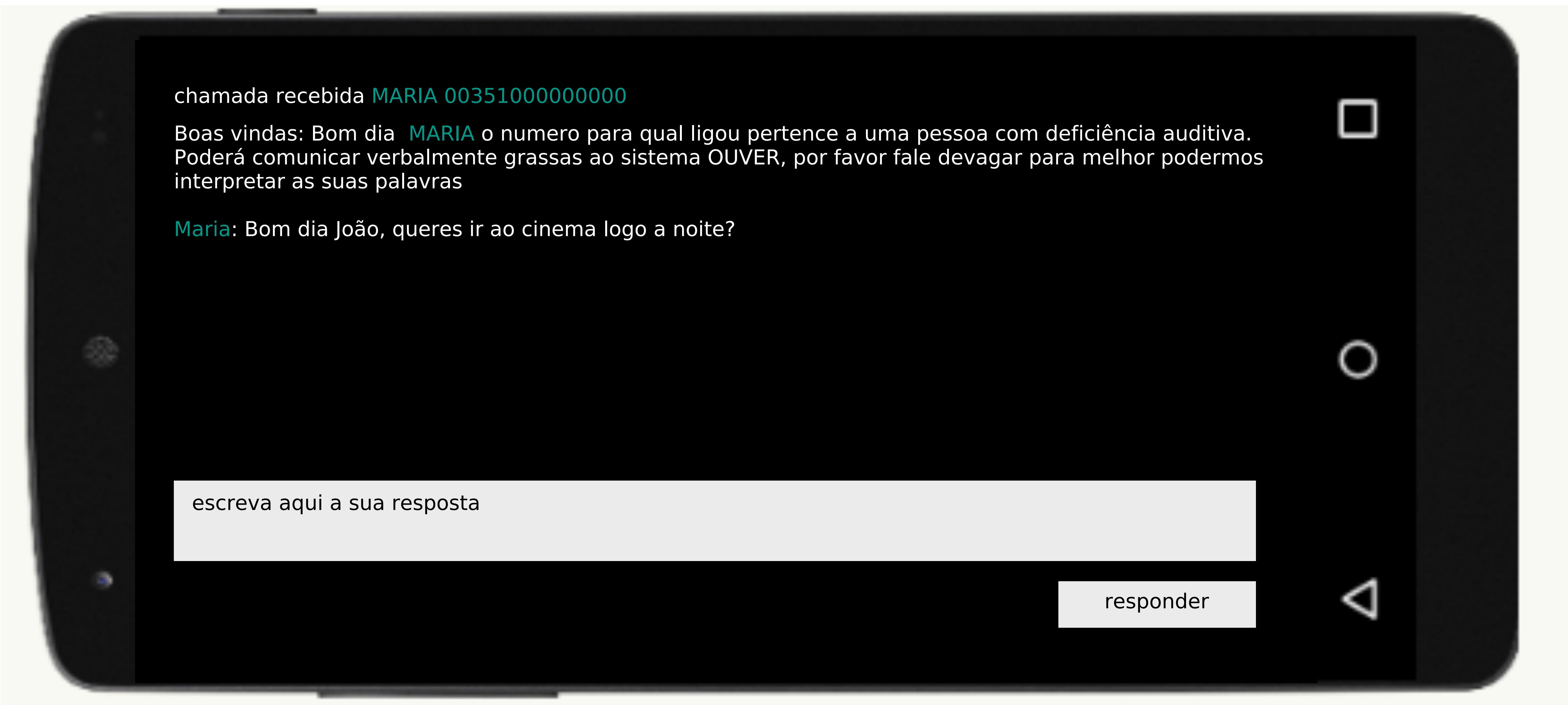
Contacto com instituições para desenvolvimento do projeto

## **Em desenvolvimento**

### **Novas Funções**

#### Medio prazo

Função Chamada (já em desenvolvimento neste momento permite efectuar chamadas mas não receber, pode escrever ou responder por voz)



## **Notas finais**

Este projeto assenta sobre as ferramentas de acessibilidade do Android e futuramente do iOS. Ambos os desenvolvedores destas plataformas oferecem assistentes por voz, as quais tem o seu motor de conversão voz para texto em constante melhoria.

Ao criarmos um projeto como este, que está meramente limitado pela qualidade das ferramentas de acessibilidade, estamos a forçar os desenvolvedores das plataformas a utilizarem a tecnologia das assistentes nas ferramentas, o que irá melhorar tanto este projeto como todo o leque de aplicações normais das mesmas..

É vitoria dupla na democratização da acessibilidade, que devia ser mais que um direito, devia ser uma prioridade.  
Enquanto não acontece, este projeto irá avançar com o aperfeiçoamento do hardware para receber essa melhoria.

## **Agradecimentos/Contribuidores:**

Aqui estão as pessoas que com o seu know-how, contribuiram para ajudar a desenvolver este projeto. Um muito obrigado a eles pela ajuda e pelo tempo dispendido.

Aqueles que futuramente se juntarão, terei muito prazer em vos adicionar a esta lista juntamente com links do vosso trabalho.

**TechFreak\_2003** - <https://www.instructables.com/Smart-Glasses-4/>

**Taifun** - <https://puravidaapps.com/>

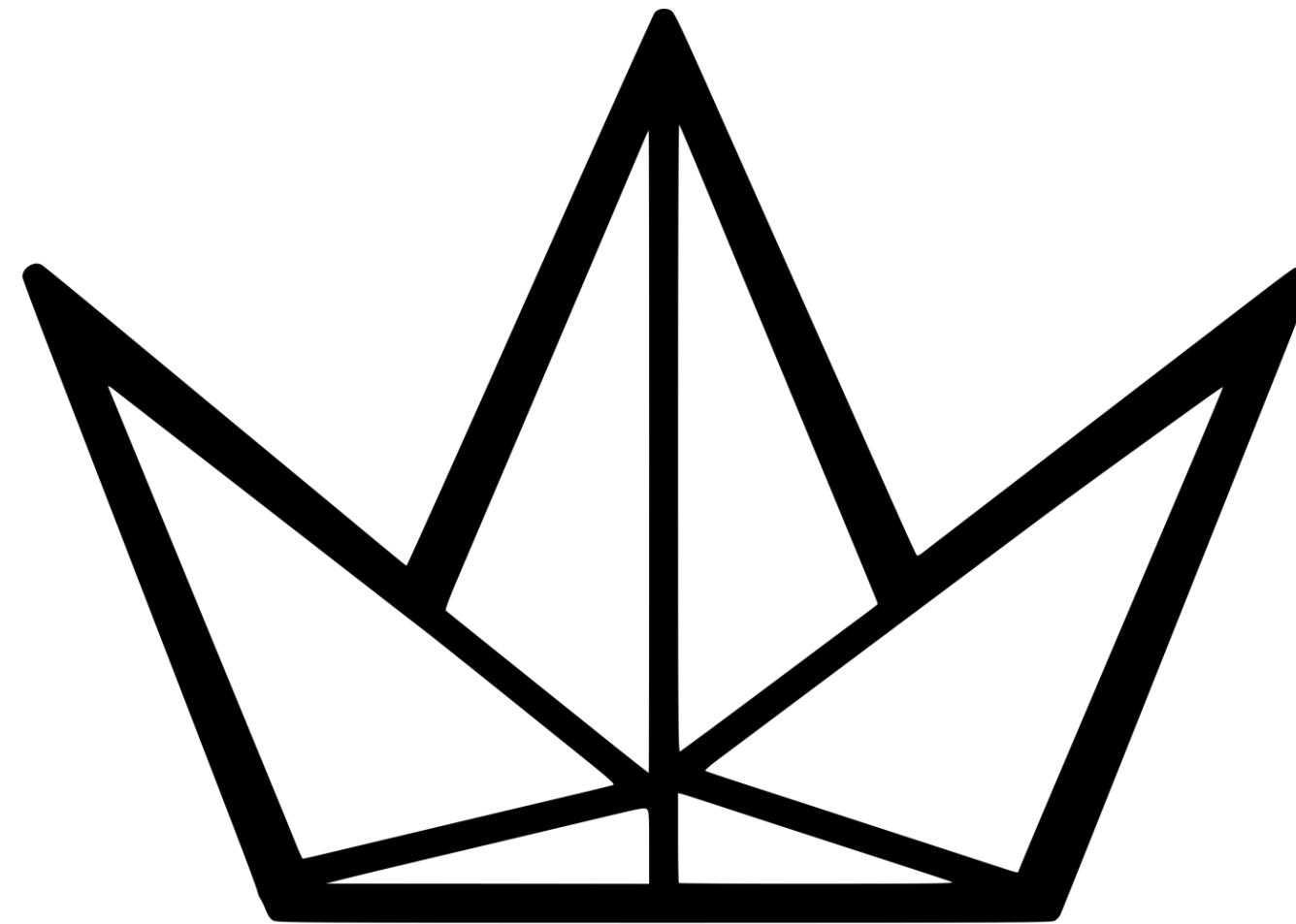
**Mr. Big**

**Herr YM**

**ScorpioNormal** - <https://community.kodular.io/t/free-voice-recognition-extension-without-google-dialogue/82347>

**O teu nome** - o teu projeto

Criado por:



DevelopKings

Apoios:



**OUVIR**  
ASSOCIAÇÃO PORTUGUESA  
DE PORTADORES DE PRÓTESES  
E IMPLANTES AUDITIVOS