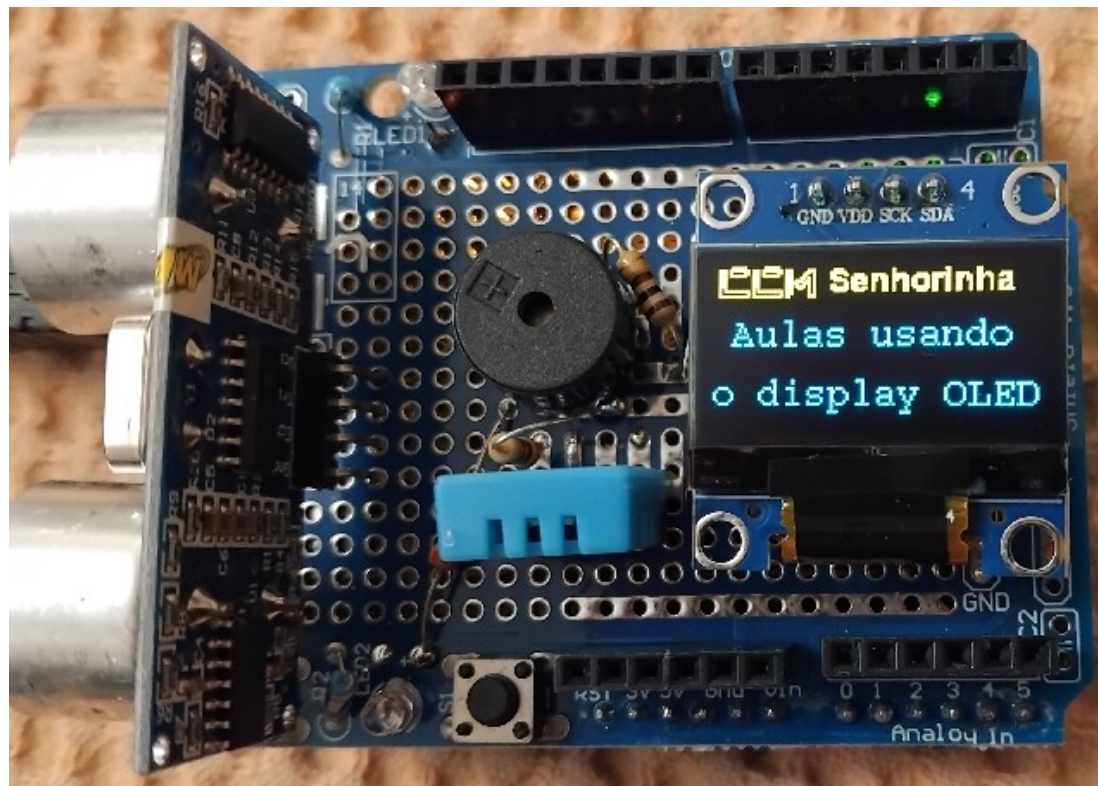


# Utilizando o *display* OLED

OLED = Diodo Orgânico Emissor de Luz



# Cores



Faixa amarela e o restante azul

Todo amarelo

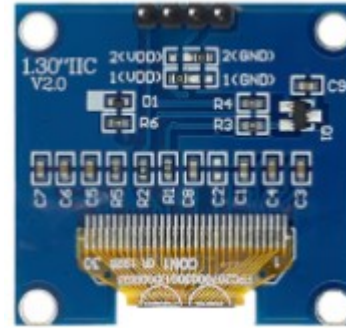
Todo branco

Todo azul

É fabricado assim, não dá para mudar durante a programação.

# Conexão

O *display* vem montado em um pequeno módulo, ao qual são adicionados um *chip* controlador e alguns componentes de apoio.



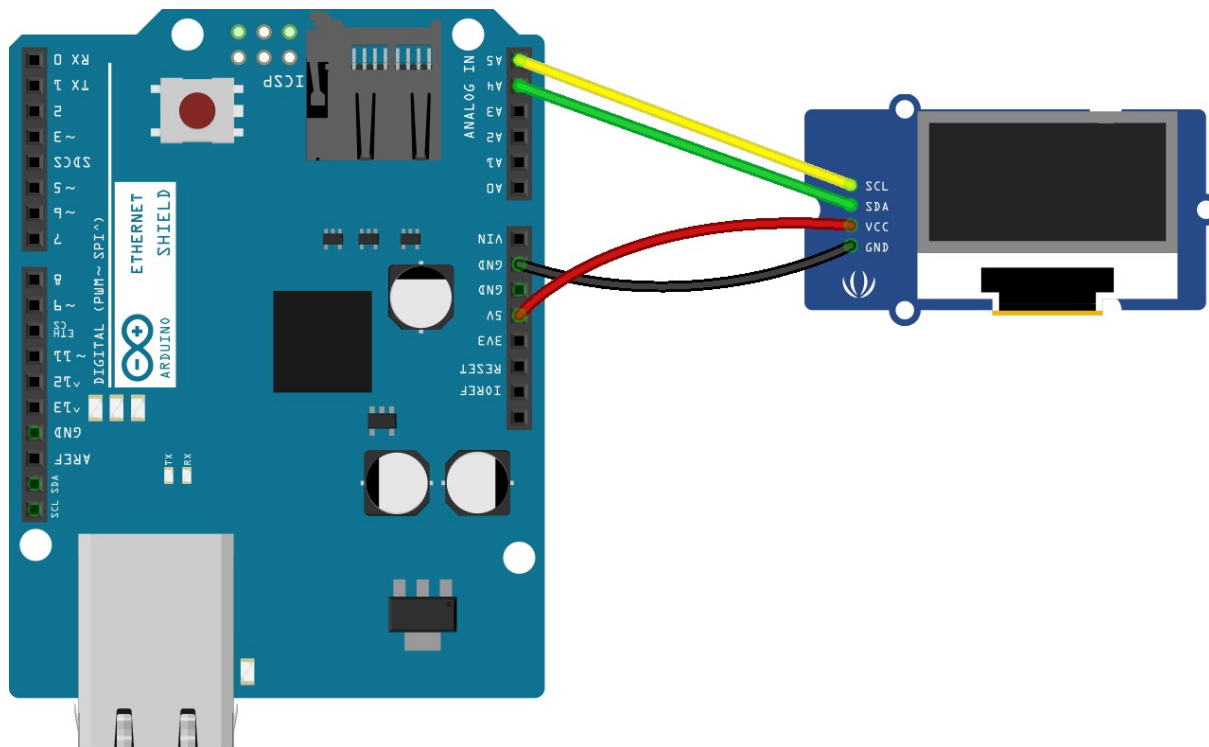
O módulo do *display* será conectado por meio de 4 fios: dois para a energia (VCC ou positivo e GND ou negativo) e dois para comunicação (SDA para os dados, e SCL para o sincronismo, ou *clock*).

# Conexão

Para o Arduino UNO, os sinais ão padronizados na interface de conexão, sendo:

SDA – no pino A4

SCL no pino A5



# Biblioteca

- Para que o Arduino possa se comunicar com e controlar o *display*, ele usará comandos de uma biblioteca de códigos
- Existem várias bibliotecas, de acordo com o chip utilizado para controlar o *display*
  - Fisicamente *displays* iguais, porém podem exigir bibliotecas de controle diferentes (chato...)

# Biblioteca

- O controle é o mesmo, independente da cor do *display*
  - ou seja, um *display* azul e um amarelo possuem o mesmo código de controle, a mesma biblioteca – só muda a cor da tela



# *Display* gráfico

- O *display* é do tipo gráfico, ou seja, podemos acessar todos os seus *pixels*:
  - 128 na horizontal
  - 64 na vertical
    - $\Rightarrow 128 \times 64 = 8192$  bits
- Pixel é um ponto de imagem, um ponto luminoso que forma imagens

# Exemplos



# Referência

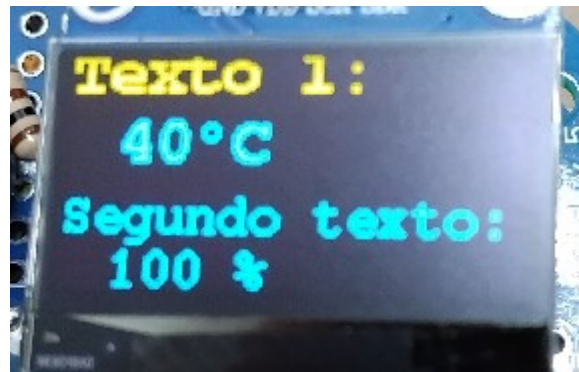
<https://github.com/olikraus/u8glib/wiki/userreference>

# Código

```
1 #include "U8glib.h" //biblioteca padrão do Google
2
3 U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE); // conexão do display OLED - note o 128 e o 64
4
5 void setup()
6 {
7     //nada em especial a fazer
8 }
9
10 void loop()
11 {
12     u8g.firstPage();
13     do {
14         mostraTexto1();
15         mostraTexto2();
16     } |
17     while( u8g.nextPage() );
18     delay(1000);
19 }
20
21 void mostraTexto1() {
22     u8g.setFont(u8g_font_courB14);
23     u8g.drawStr( 0, 10, "Texto 1:"); //coluna, linha
24     u8g.setPrintPos(15, 28);
25     u8g.print("40");
26     u8g.print(char(176));
27     u8g.print("C");
28 }
29
30 void mostraTexto2() {
31     u8g.setFont(u8g_font_courR10); //fonte diferente
32     u8g.drawStr( 0, 48, "Segundo texto:");
33     u8g.setPrintPos(15, 63);
34     u8g.print("100 %");
35 }
```

← Este código

Está neste arquivo



# Código

```
1 #include "U8glib.h" //biblioteca padrão do Google
2
3 U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE); // conexão do display OLED - note o 128 e o 64
4
5 void setup()
6 {
7     //nada em especial a fazer
8 }
```

Linha 1 - inclusão da biblioteca de controle (do Google): "U8glib.h"

Linha 3 - declaração do objeto u8g

Linha 5 - função **setup()** do Arduino

# Código

---

```
#include "U8glib.h" //biblioteca padrão do Google
```

```
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE);
```

**U8GLIB\_SSD1306\_128X64** biblioteca, tipo de controlador (**SSD1306**) e tamanho (**128** colunas, **64** linhas – lembre-se que, em 'C', vetores começam em zero; portanto, os endereços vão de **0-127** para colunas e **0-63** para linhas)

# Código

Modelos diferentes de *display* podem utilizar controladores diferentes, o que exigirá declarações (e bibliotecas) diferentes!

# Código

```
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE);
```

**u8g** nome do objeto (você pode trocar para qualquer outro, por exemplo 'meuDisplay' – o código padrão do Google, em geral, virá com u8g). Este objeto vai se comportar como uma cópia do *display* na memória.

# Código

```
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE);
```

**U8G\_I2C\_OPT\_NONE** é uma constante para o tipo de conexão utilizado; neste caso, é para **I2C** (que utiliza os pinos **SDA** e **SCL** para dados e sincronismo, respectivamente – que são os pinos A4 e A5 na placa do Arduino UNO)

# Código

```
10 void loop()  
11 {  
12     u8g.firstPage();  
13     do {  
14         mostraTexto1();  
15         mostraTexto2();  
16     }  
17     while( u8g.nextPage() );  
18     delay(1000);  
19 }
```

Estrutura  
padrão  
da  
biblioteca



# Código

```
10 void loop()  
11 {  
12     u8g.firstPage();  
13     do {  
14         mostraTexto1();  
15         mostraTexto2();  
16     }  
17     while( u8g.nextPage() );  
18     delay(1000);  
19 }
```

Linha 10 - função **loop()** do Arduino (termina na linha 19)

Linha 12 - exibição de 'páginas' de memória

Linha 13 - execute as linhas 14 e 15 enquanto houver páginas (linha 17)

Linha 18 - aguarde 1 segundo (1000 milissegundos)

# Código

```
21 void mostraTexto1() {  
22     u8g.setFont(u8g_font_courB14);  
23     u8g.drawStr( 0, 10, "Texto 1:"); //coluna, linha  
24     u8g.setPrintPos(15, 28);  
25     u8g.print("40");  
26     u8g.print(char(176));  
27     u8g.print("C");  
28 }
```

- Linha 21 - função mostraTexto1 (termina na linha 28 do código)
- Linha 22 - estabelece a fonte de caracteres
- Linha 23 - **desenha** (*draw*) um texto, na coluna e linha especificadas (0 e 10)
- Linha 24 - estabelece a próxima posição de escrita/ desenho (coluna 15, linha 28 do *display*)
- Linha 25 - **escreve** o valor '40'
- Linha 26 - escreve um caracter especial de código 176 (que é o °)
- Linha 27 - escreve o valor 'C'

# Código

```
30 void mostraTexto2() {  
31     u8g.setFont(u8g_font_courR10); //fonte diferente  
32     u8g.drawStr( 0, 48, "Segundo texto:");  
33     u8g.setPrintPos(15, 63);  
34     u8g.print("100 %");  
35 }
```

Linha 30 - função mostraTexto2 (termina na linha 35 do código)

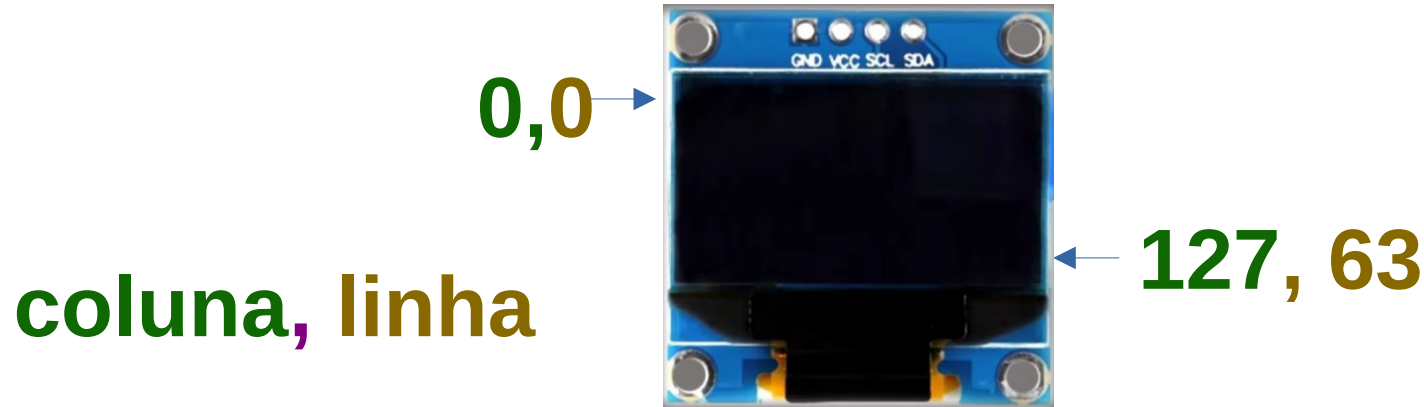
Linha 31 - estabelece a fonte de caracteres (diferente da primeira função)

Linha 32 - **desenha** (*draw*) um texto, na coluna e linha especificadas (0 e 48)

Linha 33 - estabelece a próxima posição de escrita/ desenho (coluna 15, linha 63 do *display*)

Linha 34 - **escreve** o valor '100 %'

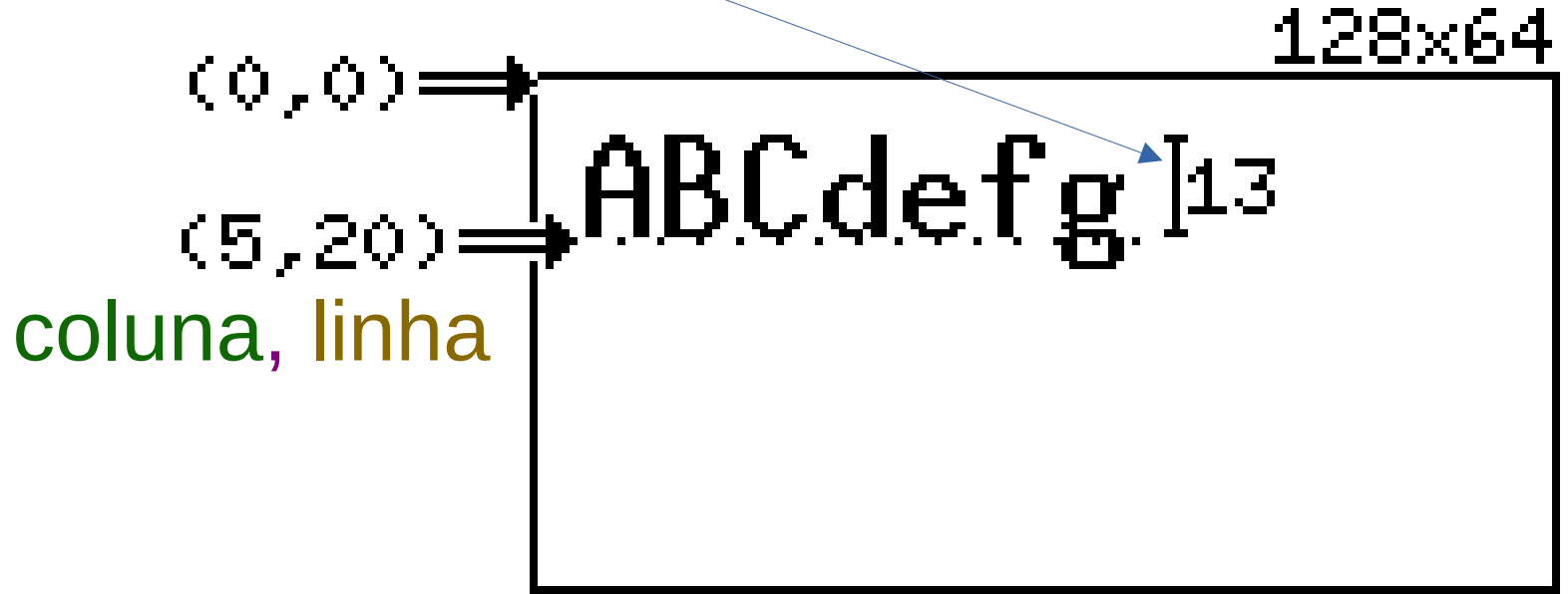
# Display



(128 colunas, 64 linhas – lembre-se que, em 'C', vetores começam em zero; portanto, os endereços vão de 0-127 para colunas e 0-63 para linhas)

# setFont e drawStr

```
u8g_setFont(u8g, u8g_font_10x20);  
u8g_drawStr(u8g, 5, 20, "ABCdefg");  
Int a = u8g_GetFontAscent(u8g);
```



# setFont

```
u8g_SetFont(u8g, u8g_font_10x20);
```

**u8g\_SetFont** função para informar qual conjunto de fontes de caracteres será utilizado

# setFont

```
u8g_SetFont(u8g, u8g_font_10x20);
```

**u8g** nome do objeto utilizado em seu programa.

(ele foi declarado no início do programa (linha 3):

```
3 U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE);
```

```
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE);)
```

# setFont

```
u8g_SetFont(u8g, u8g_font_10x20);
```

**u8g\_font\_10x20** nome da fonte de caracteres a ser utilizada (no mesmo programa, você poderá utilizar várias)



u8g\_font\_courR24: 123 ABC abcdefg [Adobe X11 Font](#)

u8g\_font\_helvB18: 123 ABC abcdefg [Adobe X11 Font](#)

u8g\_font\_helvR18: 123 ABC abcdefg [Adobe X11 Font](#)

u8g\_font\_profont29: 123 ABC abcdefg [ProFont](#)

## 20 Pixel Height

u8g\_font\_fur20: 123 ABC abcdefg [Free Universal](#)

u8g\_font\_fub20: 123 ABC abcdefg [Free Universal](#)

u8g\_font\_gdr20: 123 ABC abcdefg [Gentium](#)

u8g\_font\_gdb20: 123 ABC abcdefg [Gentium](#)

u8g\_font\_courB24: 123 ABC abcdefg [Adobe X11 Font](#)

## 21 Pixel Height

u8g\_font\_osb21: 123 ABC abcdefg [Old Standard](#)

u8g\_font\_osr21: 123 ABC abcdefg [Old Standard](#)

Exemplos  
de fontes de  
caracteres

# https://github.com/olikraus/u8glib/wiki/fontsize

## 12 Pixel Height

u8g\_font\_gdr12: 123 ABC abcdefg [Gentium](#)

u8g\_font\_gdb12: 123 ABC abcdefg [Gentium](#)

u8g\_font\_helvB12: 123 ABC abcdefg [Adobe X11 Font](#)

u8g\_font\_helvR12: 123 ABC abcdefg [Adobe X11 Font](#) →

u8g\_font\_ncenB12: 123 ABC abcdefg [Adobe X11 Font](#)

u8g\_font\_ncenR12: 123 ABC abcdefg [Adobe X11 Font](#)

u8g\_font\_courB10, courB10

BBX Width 13, Height 21, Capital A 9

Font data size: 3355

32/0x20	! " # \$ % & ' ( ) * + , - . /
48/0x30	0 1 2 3 4 5 6 7 8 9 : ; < = > ?
64/0x40	@ A B C D E F G H I J K L M N O
80/0x50	P Q R S T U V W X Y Z [ \ ] ^ _
96/0x60	` a b c d e f g h i j k l m n o
112/0x70	p q r s t u v w x y z {   } ~
128/0x80	
144/0x90	
160/0xa0	ı ċ € ₣ ¥ ¦ § ¨ © ª « ¬ ® ¯
176/0xb0	° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿
192/0xc0	À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï
208/0xd0	Ð Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß
224/0xe0	à á â ã ä å æ ç è é ê ë ì í î ï
240/0xf0	ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ

https://github.com/olikraus/u8glib/wiki/fontgroupadobex11

# <https://github.com/olikraus/u8glib/wiki/fontgroupadobex11>

u8g\_font\_courB10

u8g\_SetFont(u8g, u8g\_font\_courB10);

u8g_font_courB10, courB10	
BBX Width 13, Height 21, Capital A 9	
Font data size: 3355	
32/0x20	! " # \$ % & ' ( ) * + , - . /
48/0x30	0 1 2 3 4 5 6 7 8 9 : ; < = > ?
64/0x40	@ A B C D E F G H I J K L M N O
80/0x50	P Q R S T U V W X Y Z [ \ ] ^ _
96/0x60	` a b c d e f g h i j k l m n o
112/0x70	p q r s t u v w x y z {   } ~
128/0x80	
144/0x90	
160/0xa0	ı ċ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯
176/0xb0	° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿
192/0xc0	À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï
208/0xd0	Ð Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß
224/0xe0	à á â ã ä å æ ç è é ê ë ì í î ï
240/0xf0	ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ

O primeiro caractere da linha é o 224, que é o 'à'. Então, o seguinte é o 225 (á) e o caractere 'ã' é o 227, o 'é' tem o código 233 e assim por diante.

# Código

```

21 void mostraTextol() {
22     u8g.setFont(u8g_font_courB14);
23     u8g.drawStr( 0, 10, "Texto 1:"); //coluna, linha
24     u8g.setPrintPos(15, 28);
25     u8g.print("40");
26     u8g.print(char(176));
27     u8g.print("C");
28 }

```

u8g\_font\_courB14, courB14  
 BBX Width 17, Height 26,  
 Font data size: 4784

32/0x20	! " # \$
48/0x30	0 1 2 3 4
64/0x40	@ A B C D
80/0x50	P Q R S T
96/0x60	` a b c d
112/0x70	p q r s t
128/0x80	
144/0x90	
160/0xa0	i ¢ £ ¤
176/0xb0	° ± ² ³ ´

Para imprimir o símbolo de graus (°) eu procurei na tabela de fontes o símbolo e achei o valor 176.

Veja os símbolos para caracteres acentuados ....

# drawStr

```
u8g_drawStr(u8g, 5, 20, "ABCdefg");
```

**u8g\_drawStr** desenha uma *string* (sequência de caracteres) no objeto desejado (neste caso **u8g**), na posição coluna / linha passada (neste caso coluna **5** e linha **20**); a *string* vem indicada entre aspas duplas.

# drawStr

```
u8g_DrawStr(u8g, 5, 20, "ABCdefg");
```

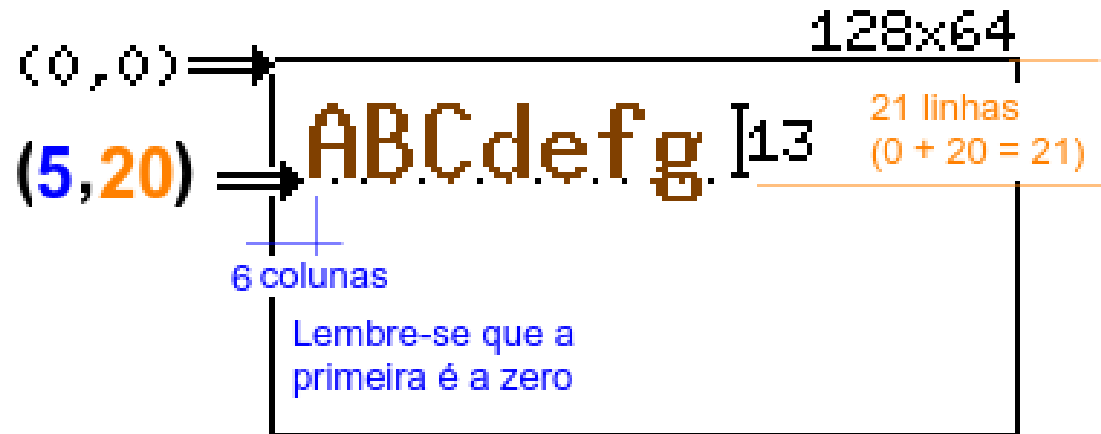
Objeto

coluna

linha

*string*

```
(U8GLIB_SSD1306_128X64  
u8g(U8G_I2C_OPT_NONE);)
```



# *Display* gráfico

- Como o *display* é do tipo gráfico, tudo pode ser tratado como 'desenho' (incluindo os textos)
- O desenho (ou imagem) que você vai exibir tem que caber na tela

Desenhar formas e *pixels*



# Criar pequenas imagens

```
5 const uint8_t desenho[] U8G_PROGMEM = {  
6     0x00,          // 00000000  
7     0x18,          // 00011000  
8     0x18,          // 00011000  
9     0x7a,          // 01111110  
10    0x7a,          // 01111110  
11    0x18,          // 00011000  
12    0x18,          // 00011000  
13    0x00           // 00000000  
14};
```



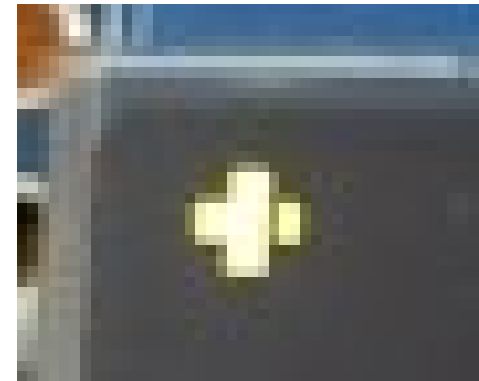
```
-----  
// 00000000  
// 00011000  
// 00011000  
// 00011000  
// 01111110  
// 01111110  
// 01111110  
// 00011000  
// 00011000  
// 00000000
```

# Criar pequenas imagens

Binário	Hexadecimal		Binário	Hexadecimal
0000	0		1000	8
0001	1		1001	9
0010	2		1010	a
0011	3		1011	b
0100	4		1100	c
0101	5		1101	d
0110	6		1110	e
0111	7		1111	f

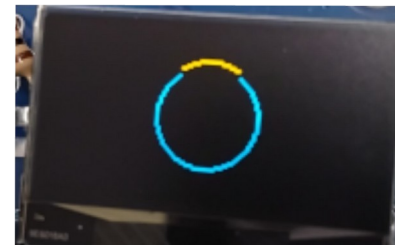
# Exibir pequenas imagens

```
5 const uint8_t desenho[] U8G_PROGMEM = {  
6     0x00,          // 00000000  
7     0x18,          // 00011000  
8     0x18,          // 00011000  
9     0x7a,          // 01111110  
10    0x7a,          // 01111110  
11    0x18,          // 00011000  
12    0x18,          // 00011000  
13    0x00           // 00000000  
14 };  
15  
16 void desenha(void) {  
17     u8g.drawBitmapP( 0, 0, 1, 8, desenho);  
18 }
```

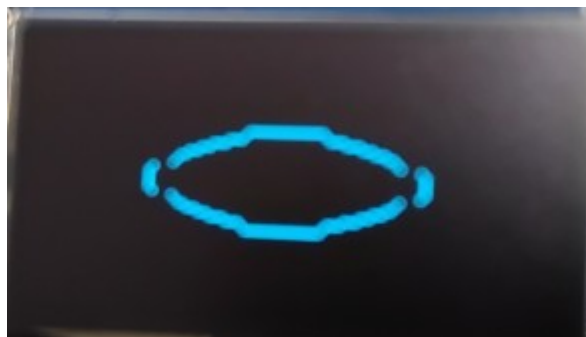
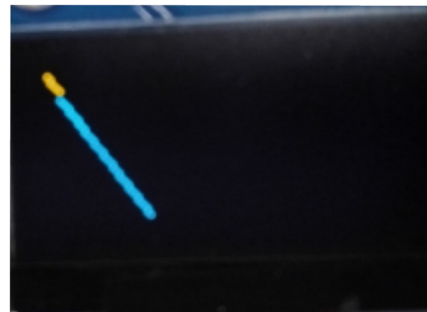


# Exibir pequenas imagens

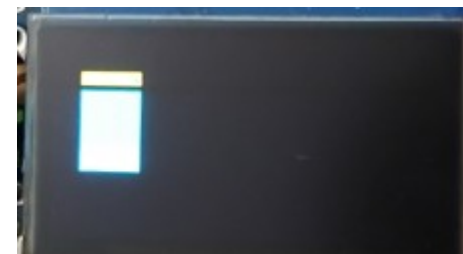
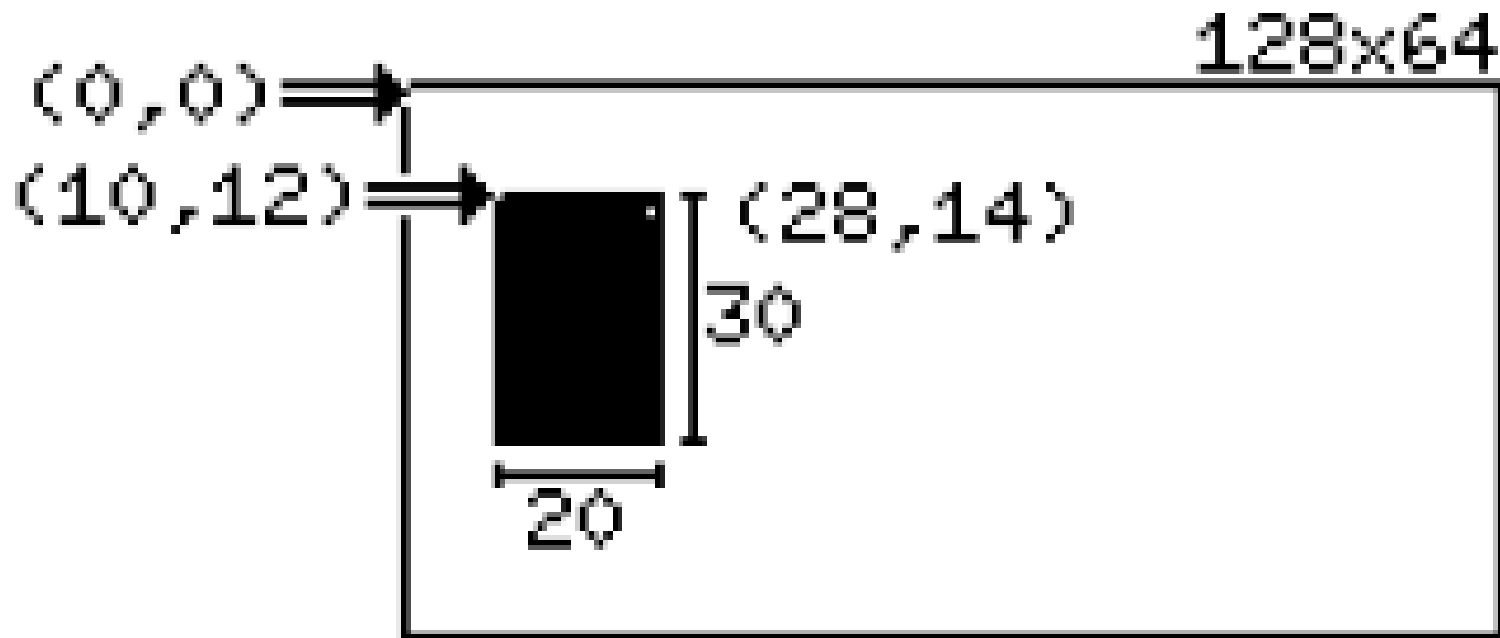
```
20 void setup()  
21 {  
22     //nada em especial a fazer  
23 }  
24  
25 void loop()  
26 {  
27     u8g.firstPage();  
28     do {  
29         desenha();  
30     }  
31     while( u8g.nextPage() );  
32     delay(1000);  
33 }
```



# Usar funções da biblioteca



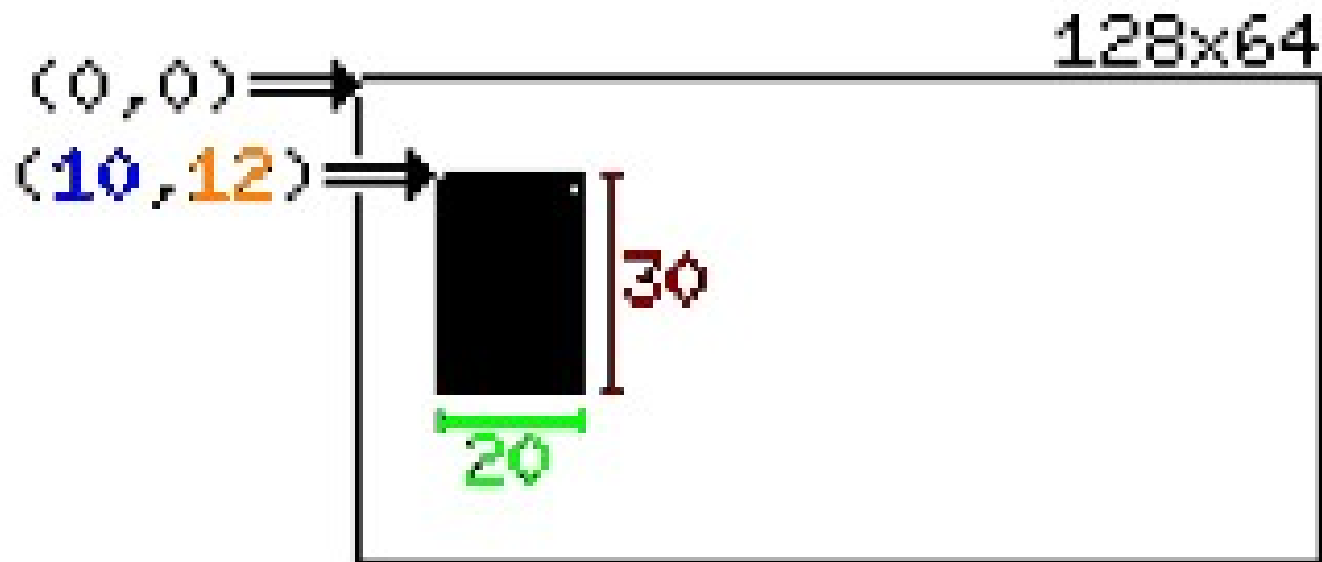
```
u8g.setColorIndex(1); //aceso  
u8g.drawBox(10, 12, 20, 30);  
u8g.setColorIndex(0); //apagado  
u8g.drawPixel(28, 14);
```



u8g.setColorIndex(1);  
Estabelece a cor 1 = aceso

u8g.drawBox(10, 12, 20, 30);  
Desenha uma 'caixa' começando na  
coluna 10 e linha 12, com 20 colunas  
de largura e 30 linhas de altura

```
u8g.setColorIndex(1);  
u8g.drawBox(10, 12, 20, 30);
```





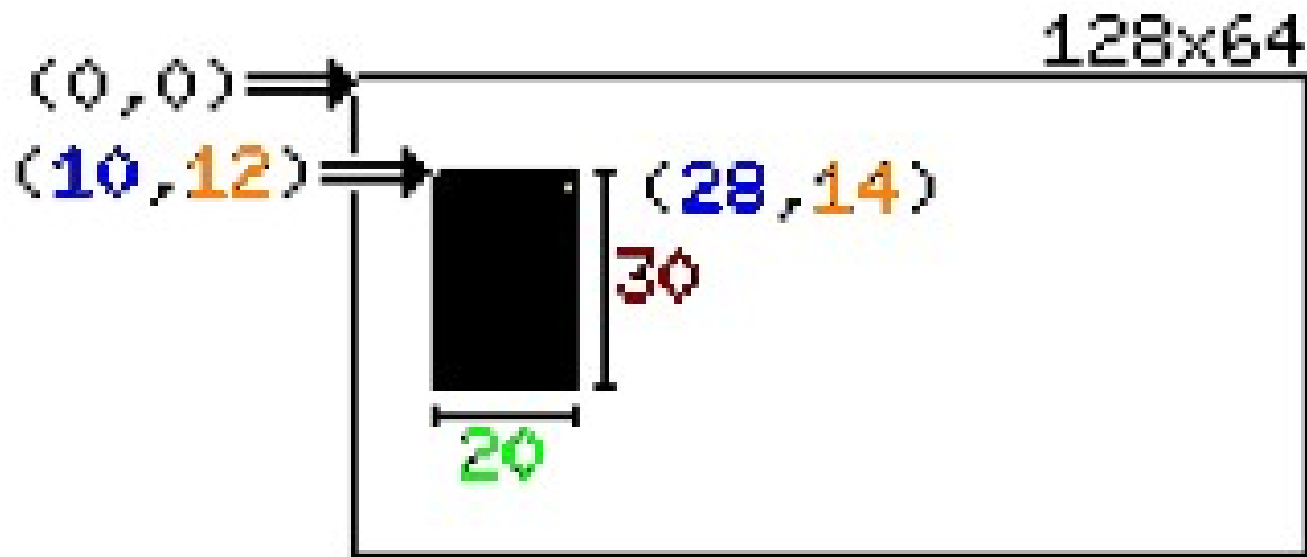
```
u8g.setColorIndex(0);
```

Estabelece a cor 0 = apagado

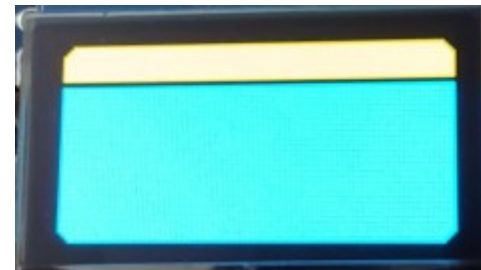
```
u8g.drawPixel (28, 14);
```

Desenha um único *pixel* na tela, na  
coluna 28 e linha 14

```
u8g.setColorIndex(1);  
u8g.drawBox(10, 12, 20, 30);  
u8g.setColorIndex(0);  
u8g.drawPixel(28, 14);
```



# Testes - 1

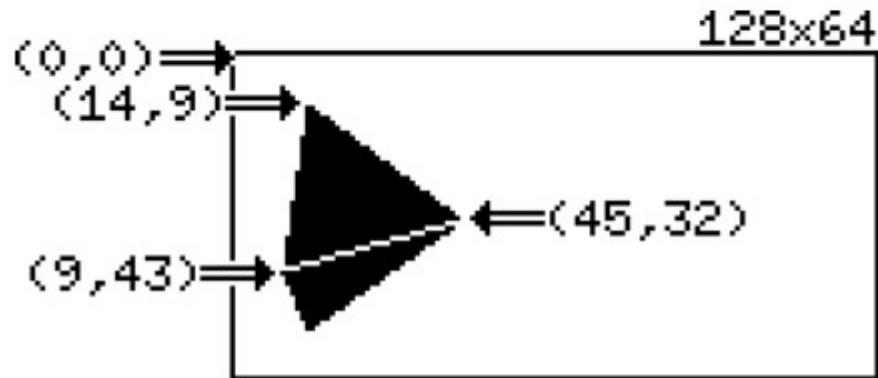


```
13 do {
14     //retire os comentários das linhas para ver as funcionalidades
15     u8g.setColorIndex(1);
16     u8g.setContrast(255); //teste outros valores, por exemplo 127
17     u8g.drawRBox(5, 5, 122, 59, 5);
18     //u8g.drawRFrame(5, 5, 122, 59, 5);
19     //u8g.drawLine(7, 10, 40, 55);
20     //u8g.drawTriangle(14,9, 45,32, 9,42);
21     //u8g.drawTriangle(14,55, 45,33, 9,43);
22     //u8g.drawCircle(64, 32, 20); //x, y, raio
23     //u8g.drawCircle(64, 32, 20, U8G_DRAW_LOWER_RIGHT); //x, y, raio, quadrante = U8G_DRAW_UPPER_RIGHT ou U8G_DRAW_UPPER_LEFT ou
24     // ou U8G_DRAW_LOWER_LEFT ou U8G_DRAW_LOWER_RIGHT ou U8G_DRAW_ALL
25     //u8g.drawDisc(64, 32, 20); //x, y, raio
26     //u8g.drawDisc(64, 32, 20, U8G_DRAW_LOWER_LEFT); //x, y, raio, quadrante = U8G_DRAW_UPPER_RIGHT ou U8G_DRAW_UPPER_LEFT ou
27     // ou U8G_DRAW_LOWER_LEFT ou U8G_DRAW_LOWER_RIGHT ou U8G_DRAW_ALL
28     //u8g.drawEllipse(64, 32, 34, 12);//, U8G_DRAW_UPPER_LEFT);
29     //u8g.drawFilledEllipse(64, 32, 34, 12, U8G_DRAW_UPPER_LEFT);
30     //u8g.drawFrame(5, 5, 122, 59);
31     //u8g.drawHLine(5, 32, 50);
32     //u8g.drawVLine(64, 32, 15);
33 }
```

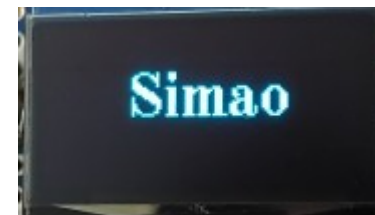
Retire os comentários ( o `('//')` ) no início das linhas de código (uma linha por vez); compile e teste. Depois recoloque o comentário e vá para a próxima linha (pode combinar o que deixar ou não comentado para criar efeitos).

# O triângulo tem vértices

```
U8GLIB u8g(...)  
...  
u8g.drawTriangle(14,9, 45,32, 9,42);  
u8g.drawTriangle(14,55, 45,33, 9,43);
```




# Testes - 2



```
10 void loop()
11 {
12     u8g.firstPage();
13     do {
14         //retire os comentários das linhas para ver as funcionalidades
15         u8g.setColorIndex(1);
16         u8g.setContrast(255); //teste outros valores, por exemplo 127
17         u8g.setFont(u8g_font_osb18);
18         u8g.drawStr(35, 35, "Simao");
19         //u8g.drawStr90(50, 0, "Simao");
20         //u8g.drawStr180(120, 20, "Simao");
21         //u8g.drawStr270(110, 63, "Simao");
22     }
```

Retire os comentários ( o `//` ) no início das linhas de código (uma linha por vez); compile e teste. Depois recoloque o comentário e vá para a próxima linha (pode combinar o que deixar ou não comentado para criar efeitos). Se quiser, troque a fonte.

```
13 do {
14     u8g.setCursorColor(1, 0); //1 foreground, 0 background
15     u8g.setCursorFont(u8g_font_cursor);
16     u8g.setCursorPos(64, 32);
17     u8g.setCursorStyle(86); //troque o número - entre 32 e 185
18     u8g.enableCursor();
19 }
```



32/0x20	
48/0x30	
64/0x40	
80/0x50	
96/0x60	
112/0x70	
128/0x80	
144/0x90	
160/0xa0	
176/0xb0	



# Imagens



A imagem deve estar no formato XBM.

O conteúdo do arquivo será copiado para dentro do código.

Em síntese, a biblioteca vai desenhar pontinho por pontinho na tela em relação àquilo que for lido na sequência copiada.



# Ajustar o tamanho da imagem

- O *display* possui 128 *pixels* de largura e 64 *pixels* de altura
  - A imagem tem que ser igual ou menor do que estas dimensões, senão ‘não cabe’
  - Sugestão para ajustar o tamanho:

<https://www.img2go.com/pt/redimensionar-imagem>

# Compactação x resolução



Imagem maior, com mais detalhes, mas não cabe inteira na tela (para o exemplo em que há um texto junto)...

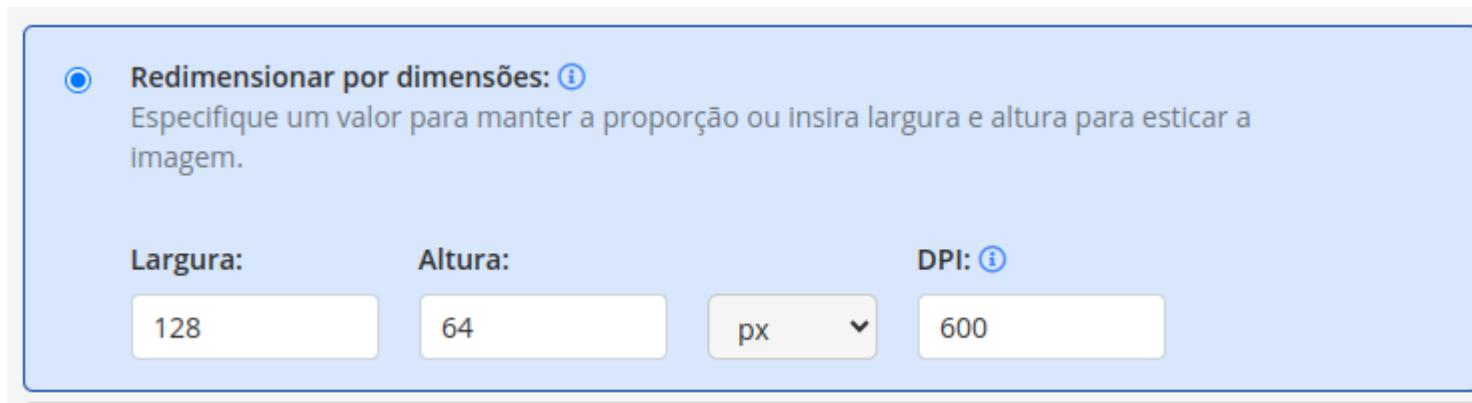
... versus

... imagem menor, mais compactada, cabe na tela mas perde detalhes.



Para redimensionar a imagem, eu usei:

<https://www.img2go.com/pt/redimensionar-imagem>



The screenshot shows the 'Redimensionar por dimensões' (Resize by dimensions) section of the img2go.com website. It includes a radio button selection, a descriptive text, and input fields for width, height, unit, and DPI.

☒ **Redimensionar por dimensões:** ⓘ  
Especifique um valor para manter a proporção ou insira largura e altura para esticar a Imagem.

Largura:  Altura:   DPI: ⓘ

Para transformar a imagem em 'XBM', eu usei:

<https://convertio.co/pt/>



curitiba128\_64.png

para

XBM



Largura máxima 128  
X  
Altura máxima 64

### Redimensionar por dimensões: ⓘ

Set either the width or height to keep the aspect ratio. If you set both, choose how the resizing should work.

Largura:

Altura:

px



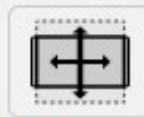
DPI: ⓘ

10 - 1200

### Redimensionamento: ⓘ



Esticar



Fill (Zoom)



Ajustar  
barras brancas

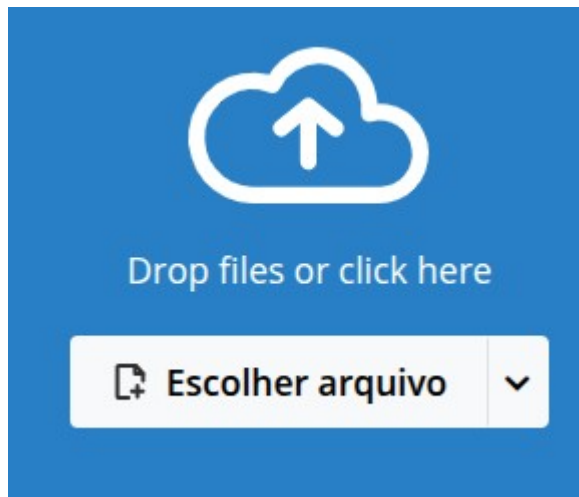


Ajustar  
barras pretas



Ajustar  
barras desfocadas

# Enviar o arquivo para edição



# Seleção e envie o arquivo

simao@ufpr.br - 2025

Cancelar

Recentes

Pasta pe...

Docume...

Downloads

Imagens

Música

Vídeos

202...

Área de ...

+ Outros l...

Abrir arquivos

simao

Documentos

2025

extensao2025

Senhorinha

aulasOLED

Nome

Tamanho

Tipo

Modificado

ajustarImagens.odp	65,9 kB	Presentation	08:30
arduino_32_32.png	217 bytes	Imagem	08:21
conv_img1_128_18.png	259 bytes	Imagem	08:26
conv_sorriso_32_32.png	478 bytes	Imagem	08:27
conv_sorrisoCoracao_32_32.png	614 bytes	Imagem	08:27
conv_sorrisoEstrela_32_32.png	547 bytes	Imagem	08:28
img1_128_18.png	260 bytes	Imagem	08:14
sorriso_32_32.png	221 bytes	Imagem	08:17
sorrisoCoracao_32_32.png	223 bytes	Imagem	08:18
sorrisoEstrela_32_32.png	223 bytes	Imagem	08:19



Arquivo adicionado! Inicie a tarefa ou adicione mais arquivos

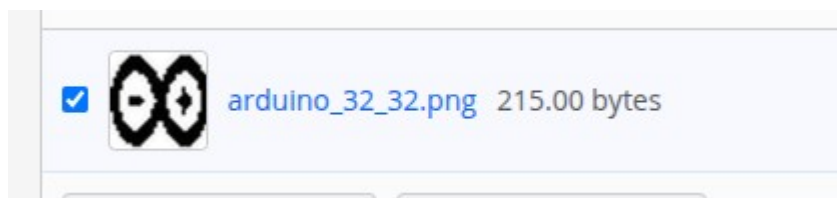


Adicionar mais arquivos



Configurações

INICIAR →







conv\_arduino\_32\_32.png



conv\_img1\_128\_128.png



conv\_sorriso\_32\_32.png



conv\_sorrisoCoracao\_32\_32.png



conv\_sorrisoEstrela\_32\_32.png

<https://convertio.co/pt/>

Selecionar arquivos



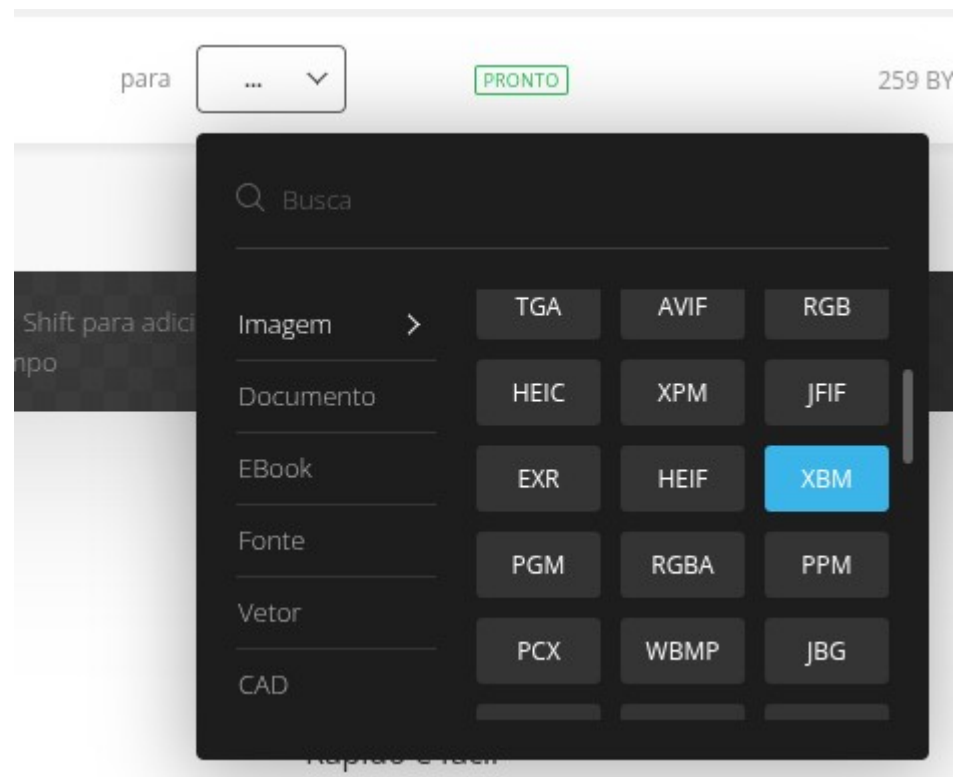


conv\_img1\_128\_18.png

para

...







conv\_img1\_128\_18.png

para

XBM ▼

PRONTO

259 BYTES



Converte todos os em ▼

+ Adicionar mais ficheiros

Use Ctrl ou Shift para adicionar vários arquivos ao mesmo tempo



Converter





conv\_img1\_128\_18.xbm

CONCLUIDO

XBm / 1.96 KB

Descarregar





0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x3F, 0xE0, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0xB8, 0x7F, 0xF7, 0xEF, 0xC0, 0x01, 0x0F, 0x00,  
0x80, 0x00, 0x00, 0x18, 0x80, 0x01, 0x00, 0x00, 0xB8, 0x60, 0x17, 0xEC,  
0x60, 0x81, 0x1F, 0x00, 0x80, 0x00, 0x00, 0x10, 0x80, 0x01, 0x00, 0x00,  
0xB8, 0x3F, 0xF7, 0xE7, 0x30, 0x81, 0x31, 0x8F, 0x8F, 0x0F, 0xCF, 0x9F,  
0x8F, 0x0D, 0x0F, 0x00, 0x38, 0x00, 0x07, 0xE0, 0x58, 0x81, 0x87, 0x9F,  
0x9D, 0x9F, 0xDF, 0x9B, 0x9B, 0x9B, 0x0D, 0x00, 0x38, 0x00, 0x07, 0xE0,  
0x6E, 0x01, 0x9F, 0x99, 0x99, 0xD9, 0xD8, 0x98, 0x99, 0x11, 0x0C, 0x00,  
0x38, 0x00, 0x07, 0xE0, 0x73, 0x01, 0xB0, 0x9F, 0x98, 0xD8, 0xD8, 0x98,  
0x91, 0x91, 0x0F, 0x00, 0x38, 0x00, 0x07, 0xE0, 0x5E, 0x81, 0xB1, 0x81,  
0x98, 0xD8, 0xD8, 0x98, 0x91, 0x91, 0x09, 0x00, 0x28, 0x00, 0x05, 0xE0,  
0x4C, 0x81, 0xBF, 0x9B, 0x98, 0x98, 0xDF, 0x98, 0x91, 0x91, 0x0D, 0x00,  
0xE8, 0x7F, 0xFD, 0xEF, 0x40, 0x01, 0x1F, 0x8F, 0x98, 0x18, 0xCF, 0x98,  
0x91, 0x91, 0x1F, 0x00, 0x08, 0x40, 0x01, 0xE8, 0x40, 0x01, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xF8, 0x7F, 0xFF, 0x6F,  
0xC0, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,



```

5 #define capivarinha_width 90
6 #define capivarinha_height 50
7 static unsigned char capivarinha_bits[] USG_PROGMEM = {
8   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
9   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
10  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
11  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
12  0x00, 0x00, 0xC0, 0xC3, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
13  0x00, 0x00, 0xC0, 0xF3, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
14  0x00, 0x00, 0xF0, 0xDF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
15  0x00, 0xF0, 0x1F, 0xC8, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
16  0x00, 0x1C, 0x00, 0xEC, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
17  0x00, 0x02, 0xF0, 0x00, 0x0E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
18  0x00, 0x11, 0xE0, 0x00, 0x3C, 0xC0, 0xFF, 0xFF, 0x7F, 0x00, 0x00, 0x00,
19  0x00, 0x00, 0x00, 0x00, 0xF0, 0xFF, 0x00, 0x00, 0xE0, 0x03, 0x00, 0x00,
20  0x00, 0x08, 0x00, 0x00, 0x80, 0x07, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00,
21  0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x80, 0x20, 0x30, 0x1C, 0x00, 0x00,
22  0x80, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x70, 0x00, 0x00,
23  0x80, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xEC, 0x00, 0x00,
24  0x80, 0x09, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x82, 0x01, 0x00,
25  0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x00,
26  0x00, 0x06, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0x00,
27  0x00, 0xFE, 0x1F, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x11, 0x06, 0x00,
28  0x00, 0xC0, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x0C, 0x00,
29  0x00, 0x00, 0xC0, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x00,
30  0x00, 0x00, 0x80, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x00,
31  0x00, 0x00, 0x00, 0x07, 0x03, 0x00, 0x00, 0x00, 0x00, 0x10, 0x08, 0x00,
32  0x00, 0x00, 0x00, 0x06, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x08, 0x00,
33  0x00, 0x00, 0x00, 0x0C, 0x20, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x00,
34  0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x0C, 0x00,
35  0x00, 0x00, 0x00, 0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x00,
36  0x00, 0x00, 0x00, 0x60, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x04, 0x00,
37  0x00, 0x00, 0x00, 0xC0, 0x00, 0x02, 0x00, 0x00, 0x01, 0x00, 0x06, 0x00,
38  0x00, 0x00, 0x00, 0x80, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x00,
39  0x00, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00, 0x20, 0x00, 0x80, 0x01, 0x00,
40  0x00, 0x00, 0x00, 0x00, 0x02, 0x02, 0x00, 0x10, 0x00, 0xC0, 0x00, 0x00,
41  0x00, 0x00, 0x00, 0x00, 0x04, 0x07, 0x00, 0x10, 0x02, 0x20, 0x00, 0x00,
42  0x00, 0x00, 0x00, 0x00, 0x1C, 0xFF, 0xF3, 0x1F, 0x00, 0x10, 0x00, 0x00,
43  0x00, 0x00, 0x00, 0x00, 0x0C, 0x07, 0x00, 0x18, 0x40, 0x1E, 0x00, 0x00,
44  0x00, 0x00, 0x00, 0x00, 0x08, 0x03, 0x00, 0x20, 0x00, 0x1F, 0x00, 0x00,
45  0x00, 0x00, 0x00, 0x00, 0x08, 0x03, 0x00, 0x20, 0xE0, 0x30, 0x00, 0x00,
46  0x00, 0x00, 0x00, 0x00, 0x88, 0x03, 0x00, 0x40, 0xE0, 0x30, 0x00, 0x00,
47  0x00, 0x00, 0x00, 0x00, 0x88, 0x03, 0x00, 0x80, 0xE0, 0x31, 0x00, 0x00,
48  0x00, 0x00, 0x00, 0x00, 0x8C, 0x03, 0x00, 0x00, 0xC0, 0x11, 0x00, 0x00,
49  0x00, 0x00, 0x00, 0x00, 0x86, 0x03, 0x00, 0x00, 0xE4, 0x19, 0x00, 0x00,
50  0x00, 0x00, 0x00, 0xC0, 0xE3, 0x03, 0x00, 0x80, 0xE0, 0x0C, 0x00, 0x00,
51  0x00, 0x00, 0x00, 0xF0, 0xF1, 0x00, 0x00, 0x20, 0x49, 0x06, 0x00, 0x00,
52  0x00, 0x00, 0x00, 0xF0, 0x0F, 0x00, 0x00, 0x00, 0x0F, 0x07, 0x00, 0x00,
53  0x00, 0x00, 0x00, 0x80, 0x01, 0x00, 0xC0, 0x81, 0x8F, 0x03, 0x00, 0x00,
54  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xDA, 0xF9, 0x00, 0x00, 0x00,
55  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
56  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
57  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
58  };

```



```

5 #define tiranossauro_width 110
6 #define tiranossauro_height 50
7 static unsigned char tiranossauro_bits[] USG_PROGMEM = {
8     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
9     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
10    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
11    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
12    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
13    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
14    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
15    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
16    0x00, 0x00, 0x00, 0xFE, 0xC0, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
17    0x00, 0x00, 0x00, 0x00, 0xC0, 0x01, 0xF7, 0xFC, 0x01, 0x00, 0x00, 0x00,
18    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0xFC, 0x09, 0x1E, 0x00,
19    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0xE8, 0x01,
20    0x70, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x68, 0x00,
21    0xE0, 0x31, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
22    0x2C, 0x00, 0x00, 0x20, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
23    0x00, 0x00, 0x04, 0x00, 0x00, 0x30, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00,
24    0x00, 0x00, 0x00, 0x00, 0x04, 0x00, 0xC0, 0x1F, 0x00, 0x06, 0x00, 0x00,
25    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x04, 0x00, 0xB8, 0x0D, 0x00, 0x04,
26    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x00, 0xFF, 0x0D,
27    0x00, 0x0C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF,
28    0x3C, 0x0F, 0x00, 0x08, 0xF0, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
29    0xB8, 0x99, 0xA7, 0x09, 0xC0, 0xF0, 0xFF, 0x7F, 0x00, 0x00, 0x00, 0x00,
30    0x00, 0x00, 0xB8, 0xFB, 0xE6, 0x04, 0xC0, 0x80, 0x03, 0xFC, 0x01, 0x00,
31    0x00, 0x00, 0x00, 0x00, 0xF8, 0xCE, 0xA0, 0x03, 0xC0, 0x00, 0x00, 0xC0,
32    0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7C, 0x01, 0x40,
33    0xE0, 0x3F, 0x1E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xEC, 0x00,
34    0x20, 0x00, 0x04, 0x60, 0xF8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80,
35    0x40, 0x00, 0x18, 0x00, 0x04, 0x00, 0xE0, 0x03, 0x00, 0x00, 0x00, 0x00,
36    0x00, 0x80, 0x3F, 0x00, 0x0E, 0x00, 0x06, 0x00, 0x80, 0x1F, 0x00, 0x00,
37    0x00, 0x00, 0x00, 0x20, 0x19, 0xC0, 0x19, 0x00, 0x02, 0x00, 0x03, 0xFC,
38    0x01, 0x00, 0x00, 0x0F, 0x00, 0xF0, 0x0F, 0x70, 0x30, 0x00, 0x03, 0x00,
39    0x03, 0xE0, 0xFF, 0xC3, 0xFF, 0x0F, 0x00, 0x3C, 0x07, 0x1C, 0x60, 0x01,
40    0x03, 0x00, 0x03, 0x00, 0xFC, 0xFF, 0x3F, 0x03, 0x00, 0xF4, 0x01, 0x07,
41    0xC0, 0x11, 0x03, 0x00, 0x03, 0x00, 0x00, 0x00, 0x80, 0x01, 0x00, 0x6C,
42    0xC0, 0x01, 0xC0, 0x18, 0x03, 0x00, 0x01, 0x00, 0x00, 0x00, 0x60, 0x00,
43    0x00, 0x38, 0x70, 0x00, 0xE0, 0x06, 0x03, 0x80, 0x01, 0x00, 0x00, 0x00,
44    0x18, 0x00, 0x00, 0x08, 0x1C, 0x00, 0x70, 0x07, 0x02, 0xC0, 0x00, 0x00,
45    0x00, 0x00, 0x07, 0x00, 0x00, 0x98, 0x03, 0x00, 0x9C, 0x19, 0x06, 0x60,
46    0x00, 0x00, 0x00, 0xC0, 0x01, 0x00, 0x00, 0xE0, 0x00, 0x00, 0x8E, 0x70,
47    0x04, 0x60, 0x00, 0x00, 0x00, 0x3C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
48    0xFF, 0xC0, 0x09, 0xC0, 0x00, 0x00, 0xE0, 0x03, 0x00, 0x00, 0x00, 0x00,
49    0x00, 0x00, 0x6F, 0x00, 0x3E, 0x80, 0xFF, 0xFF, 0x1F, 0x00, 0x00, 0x00,
50    0x00, 0x00, 0x00, 0x00, 0x1E, 0x00, 0x60, 0x80, 0x03, 0x00, 0x00, 0x00,
51    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x80, 0x01, 0x00,
52    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xC0, 0xC0,
53    0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
54    0x40, 0xC0, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
55    0x00, 0x00, 0x60, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
56    0x00, 0x00, 0x00, 0x00, 0x30, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
57    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x70, 0x00, 0x00, 0x00, 0x00,
58    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x0F, 0x38, 0x00, 0x00,
59    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFC, 0x07, 0x38,
60    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xCE,
61    0x07, 0x38, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
62    0x00, 0x66, 0x0C, 0x1C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
63    0x00, 0x00, 0x00, 0x3C, 0xFE, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
64    0x00, 0x00, 0x00, 0x00, 0x00, 0xE0, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00,
65    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
66    0x00, 0x00, 0x00, 0x00, };

```

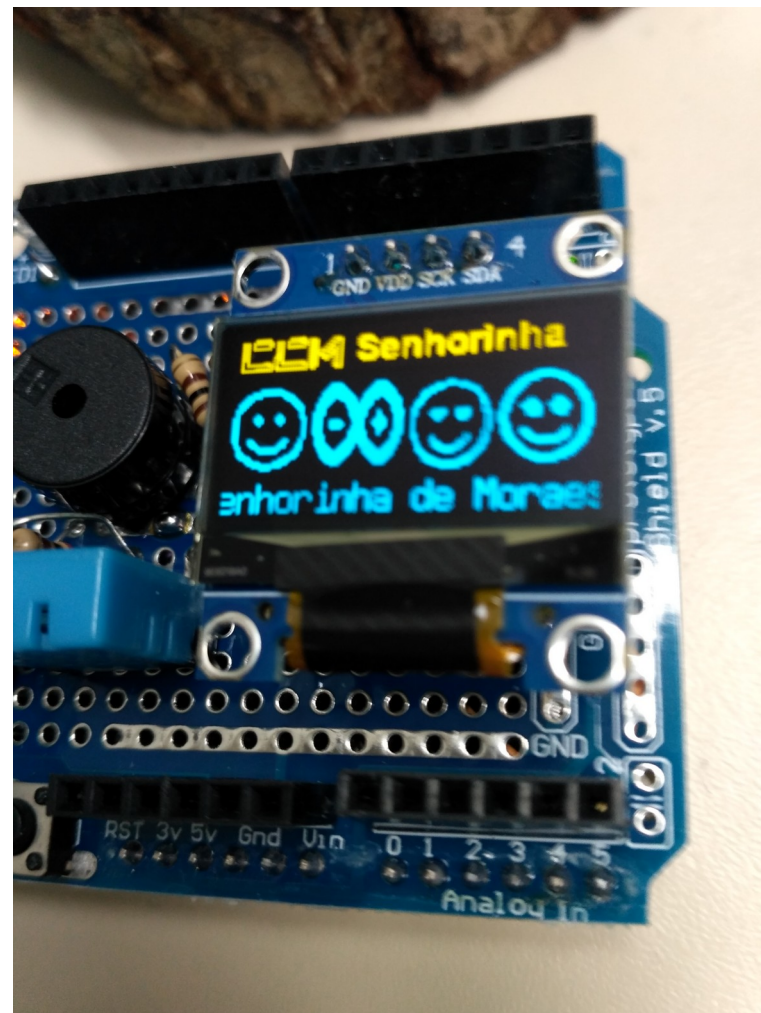




```

7 static unsigned char u8g_logo_bits1[] U8G_PROGMEM = {
8   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
9   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
10  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
11  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
12  0x00, 0x3F, 0xE0, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
13  0x00, 0x00, 0x00, 0x00, 0x00, 0xB8, 0x7F, 0xF7, 0xEF, 0xC0, 0x01, 0x0F, 0x00,
14  0x80, 0x00, 0x00, 0x18, 0x80, 0x01, 0x00, 0x00, 0xB8, 0x60, 0x17, 0xEC, 0x00,
15  0x60, 0x81, 0x1F, 0x00, 0x80, 0x00, 0x00, 0x10, 0x80, 0x01, 0x00, 0x00, 0x00,
16  0xB8, 0x3F, 0xF7, 0xE7, 0x30, 0x81, 0x31, 0x8F, 0x8F, 0x0F, 0xCF, 0x9F, 0x00,
17  0x8F, 0x00, 0x0F, 0x00, 0x38, 0x00, 0x07, 0xE0, 0x58, 0x81, 0x87, 0x9F, 0x00,
18  0x30, 0x9F, 0xDF, 0x98, 0x98, 0x98, 0x00, 0x00, 0x38, 0x00, 0x07, 0xE0, 0x00,
19  0x6E, 0x01, 0x9F, 0x99, 0x99, 0xD9, 0xD8, 0x98, 0x99, 0x11, 0x0C, 0x00, 0x00,
20  0x38, 0x00, 0x07, 0xE0, 0x73, 0x01, 0x80, 0x9F, 0x98, 0xD8, 0xD8, 0x98, 0x00,
21  0x31, 0x31, 0x0F, 0x00, 0x38, 0x00, 0x07, 0xE0, 0x5E, 0x81, 0x81, 0x81, 0x00,
22  0x98, 0xD8, 0xD8, 0x98, 0x91, 0x91, 0x09, 0x00, 0x28, 0x00, 0x05, 0xE0, 0x00,
23  0x4C, 0x81, 0x8F, 0x98, 0x98, 0x98, 0xDF, 0x98, 0x91, 0x91, 0x00, 0x00, 0x00, 0x00,
24  0xE8, 0x7F, 0xFD, 0xEF, 0x40, 0x01, 0x1F, 0x8F, 0x98, 0x18, 0xCF, 0x98, 0x00,
25  0x91, 0x91, 0x1F, 0x00, 0x08, 0x40, 0x01, 0xE8, 0x40, 0x01, 0x00, 0x00, 0x00, 0x00,
26  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xF8, 0x7F, 0xFF, 0x6F, 0x00,
27  0xC0, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
28  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
29  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
30  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
31  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 }; //CCM Senhorinha
32
33
34 static unsigned char u8g_logo_bits2[] U8G_PROGMEM = {
35   0x00, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x03, 0xC0, 0x03, 0xE0, 0x07, 0xE0, 0x07,
36   0xF0, 0x0F, 0xF0, 0x0F, 0xF8, 0x1F, 0xF8, 0x1F, 0xF8, 0x3F, 0xFC, 0x1F, 0x00,
37   0x7C, 0x7E, 0x7C, 0x3E, 0x3C, 0x78, 0x1E, 0x7C, 0x1E, 0xF0, 0x1E, 0x78, 0x00,
38   0x0E, 0xF0, 0x0F, 0x70, 0x0E, 0xE0, 0x07, 0x70, 0x0F, 0xE0, 0x07, 0xF0, 0x00,
39   0x07, 0xE0, 0x87, 0xE0, 0x07, 0xC0, 0x83, 0xE1, 0xC7, 0xC3, 0x87, 0xC3, 0xE3,
40   0xC7, 0x87, 0xE3, 0xE3, 0xC7, 0xC3, 0xC3, 0xE3, 0x07, 0xC0, 0x83, 0xE1, 0x00,
41   0x07, 0xE0, 0x87, 0xE0, 0x0E, 0xE0, 0x87, 0xF0, 0x0E, 0xE0, 0x07, 0x70, 0x00,
42   0x0E, 0xF0, 0x0F, 0x70, 0x1E, 0x78, 0x1E, 0x78, 0x3C, 0x7C, 0x3E, 0x3C, 0x00,
43   0x7C, 0x3E, 0x7C, 0x3E, 0xF8, 0x3F, 0xFC, 0x1F, 0xF8, 0x1F, 0xF8, 0x1F, 0x00,
44   0xF0, 0x0F, 0xF0, 0x0F, 0xE0, 0x07, 0xE0, 0x07, 0x80, 0x03, 0xC0, 0x03,
45   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00//arduino
46 };
47
48 static unsigned char u8g_logo_bits1[] U8G_PROGMEM = {
49   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0x0F, 0x00,
50   0x00, 0xFE, 0x3F, 0x00, 0x80, 0x0F, 0xF0, 0x00, 0xC0, 0x01, 0xE0, 0x01, 0x00,
51   0xE0, 0x00, 0x80, 0x03, 0x70, 0x00, 0x00, 0x07, 0x30, 0x00, 0x00, 0x0E, 0x00,
52   0x18, 0x00, 0x00, 0x0C, 0x08, 0x00, 0x00, 0x08, 0x08, 0x08, 0x08, 0x08, 0x08,
53   0x0C, 0x1C, 0x1C, 0x18, 0x0C, 0x1C, 0x1C, 0x10, 0x0C, 0x08, 0x08, 0x10, 0x00,
54   0x0C, 0x00, 0x00, 0x10, 0x0C, 0x00, 0x00, 0x10, 0x0C, 0x00, 0x00, 0x10, 0x00,
55   0x0C, 0x00, 0x00, 0x10, 0x0C, 0x00, 0x00, 0x10, 0x0C, 0x03, 0x60, 0x08, 0x00,
56   0x08, 0x07, 0x30, 0x08, 0x18, 0x0E, 0x38, 0x0C, 0x10, 0xF8, 0x0F, 0x0C, 0x00,
57   0x30, 0xF0, 0x07, 0x06, 0x70, 0x00, 0x07, 0xE0, 0x01, 0xC0, 0x03, 0x00,
58   0x80, 0x03, 0xF0, 0x00, 0x00, 0x1F, 0x7C, 0x00, 0x00, 0xFC, 0x1F, 0x00, 0x00,
59   0x00, 0xE0, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00//sorriso
60 };
61
62 static unsigned char u8g_logo_bits3[] U8G_PROGMEM = {
63   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x03, 0x00, 0x00, 0x00,
64   0x00, 0xF8, 0x3F, 0x00, 0x00, 0x3E, 0xF8, 0x00, 0x80, 0x07, 0xC0, 0x03, 0x00,
65   0xC0, 0x01, 0x00, 0x03, 0xC0, 0x00, 0x00, 0x06, 0x60, 0x00, 0x00, 0x0C, 0x00,
66   0x70, 0x00, 0x00, 0x0C, 0x30, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x10, 0x00,
67   0x18, 0x7C, 0x7C, 0x10, 0x18, 0x7E, 0x7C, 0x30, 0x08, 0x3C, 0x7C, 0x20, 0x00,
68   0x08, 0x18, 0x38, 0x20, 0x08, 0x00, 0x00, 0x20, 0x08, 0x00, 0x00, 0x20, 0x00,
69   0x08, 0x00, 0x00, 0x20, 0x08, 0x00, 0x00, 0x20, 0x18, 0x00, 0x00, 0x10, 0x00,
70   0x18, 0x04, 0x40, 0x10, 0x10, 0x04, 0x40, 0x10, 0x30, 0x18, 0x30, 0x18, 0x00,
71   0x20, 0x38, 0x18, 0x08, 0x60, 0xF0, 0x0F, 0x0C, 0xC0, 0xC0, 0x03, 0x06, 0x00,
72   0x80, 0x01, 0x00, 0x03, 0x00, 0x07, 0xC0, 0x01, 0x00, 0x1F, 0xF0, 0x00, 0x00,
73   0x00, 0xFC, 0x7F, 0x00, 0x00, 0xE0, 0x0F, 0x00//sorriso coraço
74 };
75
76 static unsigned char u8g_logo_bits4[] U8G_PROGMEM = {
77   0x00, 0x00, 0x00, 0x00, 0x00, 0xE0, 0x07, 0x00, 0x00, 0xFC, 0x3F, 0x00, 0x00,
78   0x00, 0x1F, 0x78, 0x00, 0x80, 0x03, 0xE0, 0x01, 0xC0, 0x01, 0x80, 0x03, 0x00,
79   0xE0, 0x00, 0x00, 0x07, 0x70, 0x00, 0x00, 0x06, 0x30, 0x00, 0x00, 0x0C, 0x00,
80   0x18, 0x00, 0x00, 0x1C, 0x18, 0x08, 0x18, 0x18, 0x0C, 0x1C, 0x38, 0x18, 0x00,

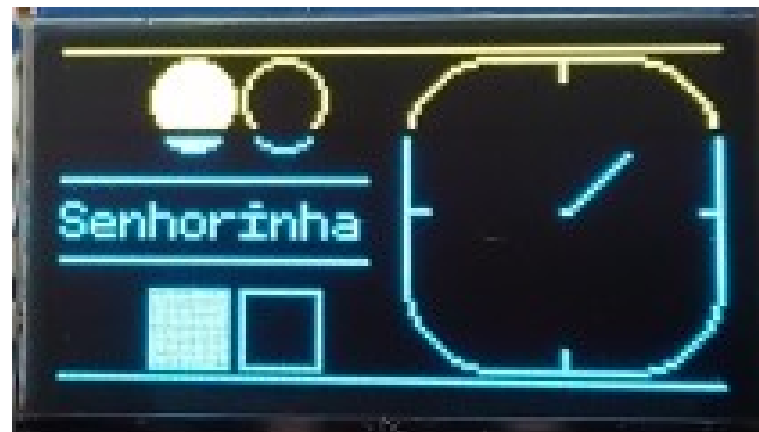
```



```

14 void draw(void)
15 {
16     //=====Senhorinha=====//
17     u8g.setFont(u8g_font_fixed_v0r);
18     u8g.drawStr(0, 36, "Senhorinha");
19     u8g.drawLine(0,24,60,24);
20     u8g.drawLine(0,40,60,40);
21     u8g.drawLine(0,0,128,0);
22     u8g.drawLine(0,63,128,63);
23     //===== Gráfico=====//
24     u8g.drawDisc(8+i,10,8);//(x,y,diametro)
25     u8g.drawCircle(26+i,10,8);//(x,y,diametro)
26     u8g.drawFrame(18+i,46,16,16);//(x,y,comprimento,largura)
27     u8g.drawBox(i,46,16,16);//(x,y,comprimento,largura)
28
29     //=====Ponteiros=====//
30     u8g.drawRFrame(68,2,60,60,18);//(x,y,comprimento,largura, curvatura do vértice)
31     u8g.drawLine(98, 2, 98, 6);//12horas
32     u8g.drawLine(124, 30, 128,30);//3horas
33     u8g.drawLine(98, 57, 98,61);//6horas
34     u8g.drawLine(68, 30, 72,30);//9horas
35     u8g.drawLine(98, 30, 98,30);//início x, y, fim x, y
36     //=====Segundos=====//
37     x=99+16*sin(3.14*6*time/180);
38     y=30-16*cos(3.14*6*time/180);
39     u8g.drawLine(99,30, x, y);
40 }

```



# Agora, crie seu(s) desenho(s)

```
13 do {  
14     u8g.setColorIndex(1);  
15     u8g.setContrast(127);  
16     u8g.drawLine(12, 10, 25, 2);  
17     u8g.drawLine(25, 2, 38, 10);  
18     u8g.drawLine(25, 2, 100, 2);  
19     u8g.drawLine(100, 2, 113, 10);  
20     u8g.drawFrame(12, 10, 100, 35);  
21     u8g.drawVLine(38, 10, 35);  
22     u8g.drawBox(17, 22, 18, 23);  
23     u8g.drawFrame(47, 18, 49, 19);  
24     u8g.drawVLine(62, 18, 19);  
25     u8g.drawVLine(80, 18, 19);  
26     u8g.drawLine(48, 26, 48);  
27     u8g.setFont(u8g_font_courR10);  
28     u8g.drawStr( 5, 62, "Minha casinha");  
29 }
```

