```
#include <HID-Project.h>                    // biblioteca para usar teclado
#include <Keypad.h>                         // biblioteca para usar teclado
#include <Wire.h>                           // biblioteca para usar tela
#include <Adafruit_GFX.h>                   // biblioteca de usar a tela
#include <Adafruit_SSD1306.h>               // biblioteca de usar a tela

#define SCREEN_WIDTH 128                    // tamanho do display, em pixels
#define SCREEN_HEIGHT 64                    // largura do display, em pixels
#define SCREEN_ADDRESS 0x3C                 // endereço do display
#define oledTime 2200                       // tempo provisório de espera do OLED
#define oledTimeSlow oledTime / 2           // tempo de tela

const unsigned char delphosMacro [] PROGMEM = {
  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, 0x7f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xff, 0xf8,
0x00, 0x00, 0x00,
  0x00, 0x00, 0x03, 0x80, 0xff, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x0f, 0x00, 0x07, 0xff, 0x80,
0x00, 0x00, 0x00, 0x1c, 0x00, 0x00, 0x0f, 0xc0, 0x00, 0x00, 0x00, 0x78, 0x3f, 0x00, 0x00,
0xe0, 0x00,
  0x00, 0x00, 0xe0, 0xff, 0xfc, 0x00, 0x60, 0x00, 0x00, 0x03, 0xc1, 0xff, 0xff, 0xf8, 0x30,
0x00, 0x00, 0x0f, 0x07, 0xff, 0xff, 0xf8, 0x30, 0x00, 0x00, 0x1c, 0x0f, 0xff, 0xff, 0xf8, 0x30,
0x00,
  0x00, 0x38, 0x3f, 0xff, 0xff, 0xf0, 0x18, 0x00, 0x00, 0x70, 0x7f, 0xff, 0xff, 0xe0, 0x18, 0x00,
0x00, 0x61, 0xff, 0xff, 0xff, 0xc0, 0x0c, 0x00, 0x00, 0xc3, 0xff, 0xff, 0xff, 0x80, 0x0c, 0x00,
  0x00, 0xc0, 0xff, 0xff, 0xff, 0x80, 0x0c, 0x00, 0x01, 0xc0, 0x3f, 0xff, 0xff, 0x00, 0x06, 0x00,
0x01, 0x88, 0x0f, 0xff, 0xfe, 0x00, 0x06, 0x00, 0x01, 0x8e, 0x00, 0xff, 0xfc, 0x00, 0x03,
0x00,
  0x03, 0x1f, 0x80, 0x0f, 0xf8, 0x00, 0x03, 0x00, 0x03, 0x1f, 0xf0, 0x00, 0x00, 0x00, 0x03,
0x00, 0x07, 0x1f, 0xff, 0x00, 0x00, 0x00, 0x01, 0x80, 0x06, 0x3f, 0xe0, 0x7c, 0x00, 0x00,
0x01, 0x80,
  0x06, 0x3f, 0x00, 0x1f, 0xf8, 0x00, 0x00, 0xc0, 0x0c, 0x7c, 0x00, 0x07, 0xf8, 0x00, 0x00,
0xc0, 0x0c, 0x78, 0x07, 0x03, 0xf8, 0x00, 0x00, 0xc0, 0x18, 0x70, 0x1f, 0xc1, 0xf8, 0x00,
0x00, 0x60,
  0x18, 0xf0, 0x3f, 0xc0, 0xfc, 0x00, 0x00, 0x60, 0x18, 0xf0, 0x70, 0xe0, 0x7c, 0x00, 0x00,
0xc0, 0x31, 0xf0, 0x60, 0x60, 0x7c, 0x00, 0x00, 0xc0, 0x31, 0xf8, 0x60, 0x70, 0x3c, 0x00,
0x01, 0xc0,
  0x31, 0xf8, 0x60, 0x60, 0x3c, 0x00, 0x01, 0x80, 0x63, 0xfc, 0x70, 0x60, 0x3c, 0x00, 0x03,
0x00, 0x31, 0xfc, 0x30, 0xe0, 0x7c, 0x00, 0x07, 0x00, 0x30, 0xfe, 0x1f, 0xc0, 0xfc, 0x00,
0x0e, 0x00,
  0x30, 0x1f, 0x0f, 0x01, 0xfc, 0x00, 0x0c, 0x00, 0x18, 0x03, 0x80, 0x07, 0xfc, 0x00, 0x18,
0x00, 0x0f, 0x00, 0x60, 0x1f, 0xfc, 0x00, 0x30, 0x00, 0x07, 0xe0, 0x07, 0xff, 0xfe, 0x00,
0x70, 0x00,
  0x01, 0xfc, 0x00, 0xff, 0xfe, 0x00, 0xe0, 0x00, 0x00, 0x1f, 0x80, 0x1f, 0xfe, 0x00, 0xc0,
0x00, 0x00, 0x03, 0xf0, 0x03, 0xfe, 0x01, 0x80, 0x00, 0x00, 0x00, 0x7f, 0x00, 0x7e, 0x03,
0x00, 0x00,
```

```
  0x00, 0x00, 0x0f, 0xe0, 0x06, 0x07, 0x00, 0x00, 0x00, 0x00, 0x01, 0xfc, 0x00, 0x0e, 0x00,
0x00, 0x00, 0x00, 0x00, 0x1f, 0x80, 0x0c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xf0, 0x38,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x7f, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0f, 0xe0, 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
};

const unsigned char delphosText [] PROGMEM = {
  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, 0x1f, 0xc0, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3b, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0x1f, 0xf8, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xc0, 0x00, 0x7f, 0xc0, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0x07, 0x00, 0x01, 0xe0, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1c, 0x1f, 0xf8, 0x00, 0x60, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x38, 0x3f, 0xff, 0xf0, 0x20, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0xff, 0xfc, 0x30, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xc1, 0xff, 0xff, 0xf8, 0x30, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x07, 0xff, 0xff, 0xf8, 0x18, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x0f, 0xff, 0xff, 0xf0, 0x18, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x3f, 0xff, 0xff, 0xe0, 0x08, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x3f, 0xff, 0xff, 0xc0, 0x0c, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x08, 0x1f, 0xff, 0xff, 0x80, 0x0c, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x18, 0x03, 0xff, 0xff, 0x80, 0x06, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x19, 0xc0, 0x7f, 0xff, 0x00, 0x06, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x31, 0xf0, 0x07, 0xfe, 0x00, 0x02, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x31, 0xfc, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x23, 0xff, 0xc0, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x63, 0xfc, 0x1e, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x67, 0xe0, 0x07, 0xfc, 0x00, 0x01, 0x80, 0x00, 0x00,
0x00, 0x00,
```

0x00, 0x00, 0x00, 0x00, 0x00, 0xc7, 0x80, 0x01, 0xfc, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc7, 0x01, 0xe0, 0xfe, 0x00, 0x00, 0xc0, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x8e, 0x03, 0xf0, 0x7e, 0x00, 0x00, 0xc0, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x8e, 0x07, 0x38, 0x3e, 0x00, 0x00, 0xc0, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x01, 0x9f, 0x0c, 0x18, 0x3e, 0x00, 0x00, 0xc0, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x1f, 0x0c, 0x0c, 0x1e, 0x00, 0x01, 0x80, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x03, 0x1f, 0x0c, 0x0c, 0x0e, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x3f, 0x8c, 0x18, 0x0e, 0x00, 0x03, 0x00, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x03, 0x3f, 0x86, 0x18, 0x1e, 0x00, 0x06, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x1f, 0xc7, 0xf0, 0x3e, 0x00, 0x0c, 0x00, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x01, 0x81, 0xe1, 0xe0, 0x7f, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xc0, 0x30, 0x01, 0xff, 0x00, 0x18, 0x00, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0x04, 0x07, 0xff, 0x00, 0x30, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3e, 0x00, 0xff, 0xff, 0x00, 0x60, 0x00, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xc0, 0x1f, 0xff, 0x00, 0xe0, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0x03, 0xff, 0x00, 0xc0, 0x00, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x80, 0x3f, 0x01, 0x80, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xf0, 0x07, 0x03, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7e, 0x00, 0x06, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xe0, 0x0e, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfc, 0x1c, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xf8, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xe0, 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, 0x03, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xbf, 0xdc, 0x0f, 0xf7, 0x3b, 0xfd, 0xfe, 0x00,
0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x7f, 0xbf, 0xdc, 0x0f, 0xf7, 0x3b, 0xfd, 0xfe, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xbf, 0xdc, 0x0f, 0xf7, 0x3b, 0xfd, 0xfe, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x73, 0xb8, 0x1c, 0x0e, 0x77, 0x3b, 0x9d, 0xc0, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x73, 0xbb, 0xdc, 0x0f, 0xf7, 0xfb, 0x9d, 0xfe, 0x00, 0x00,
0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x73, 0xbb, 0xdc, 0x0f, 0xf7, 0xfb, 0x9d, 0xfe, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x73, 0xbb, 0xdc, 0x0f, 0xf7, 0xfb, 0x9d, 0xfe, 0x00, 0x00,
0x00, 0x00,

```
  0x00, 0x00, 0x00, 0x00, 0x73, 0xb8, 0x1c, 0xee, 0x07, 0x3b, 0x9c, 0x0e, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xbf, 0xdf, 0xee, 0x07, 0x3b, 0xfd, 0xfe, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x7f, 0xbf, 0xdf, 0xee, 0x07, 0x3b, 0xfd, 0xfe, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xbf, 0xdf, 0xee, 0x07, 0x3b, 0xfd, 0xfe, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x08, 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
};

const unsigned char miniEye [] PROGMEM = {
  0x00, 0x00, 0x00, 0x00, 0x0f, 0x00, 0x19, 0x80, 0x30, 0xc0, 0x66, 0x60, 0x66, 0x60, 0x20,
0xc0,
  0x19, 0x80, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00
};

//ARDUINO PRO MICRO

//TELA
//Os pinos do display devem ser conectados da seguinte maneira:
//SDA pino 2
//SCL pino 3

//ROTARY ENCODERS
const int N_ENCODERS = 1;                       // numero de rotary encoders
byte clkPin[N_ENCODERS] = { A1};                  // numero dos pinos CLK
byte dtPin[N_ENCODERS] = { 16};                  // numero dos pinos DT
int clkState[N_ENCODERS] = { 0 };                // etado do rotary
int rotaryPState[N_ENCODERS] = { 0 };             // estado prévio do Rotary
int rotaryCounter[N_ENCODERS] = { 0 };            // contsador do Rotary Encoder

//BOTOES
const int N_BUTTON = 1;                         // Numero de botoes
byte buttonPin[N_BUTTON] = {A2};                  // lista com porta de cada botao
byte buttonState[N_BUTTON] = { 0 };               // Lista com estado de cada botao
byte buttonPState[N_BUTTON] = { 0 };              // Lista com estado prévio de cada botao
unsigned long lastBounce[N_BUTTON] = { 0 };       // reseta o relogio
unsigned long buttonTimer[N_BUTTON] = { 0 };      // timer do botao
int buttonTimeOut = 15;

//TECLADO
const byte qtdLinhas = 4;                       // Quantidade de linhas do teclado
const byte qtdColunas = 4;                      // Quantidade de colunas do teclado
char matrizTeclas[qtdLinhas][qtdColunas] = {
```

```
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
};

byte pinosQtdLinhas[qtdLinhas] = {A0, 4, 5};        // Pinos utilizados pelas linhas
byte pinosQtdColunas[qtdColunas] = {7, 8, 9, 10};     // Pinos utilizados pelas colunas

Keypad meuTeclado = Keypad(makeKeymap(matrizTeclas), pinosQtdLinhas,
pinosQtdColunas, qtdLinhas, qtdColunas); // Inicialização do teclado pela biblioteca
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1); //Inicia display

char teclaPressionada;                          // variável para conferir qual a tecla que foi
pressionada
int perfil = 1;                                 // variável para selecionar o pefil do teclado
bool oledSwitch = true;
String acaoTeclado;

//--------------------------------
void setup(){
  Serial.begin(9600);                           // Inicia a serial
  BootKeyboard.begin();                           // Inicia o teclado
  Consumer.begin();                             // inicia o controle de Volume

  for(int i = 0; i < N_ENCODERS; i++){
    pinMode(clkPin[i], INPUT);                    // Define o pino clk como entrada
    pinMode (dtPin[i], INPUT);                    // Define o pino Dt como entrada
    rotaryPState[i] = digitalRead(clkPin[i]);         // Variável que faz a leitura Inicial do pino Clk
  }

  // Inicializa o display
  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
Serial.println(F("Erro na inicialização!")); while(1); }

  //delphosLogo();
  delphosTextLogo();
  credits();
  clear();
}

//--------------------------------
void loop(){
  rotaryEncoders();                             // Função que faz a leitura dos Encoders
  buttons();                                  // Função que faz a leitura dos botões
  info();                                   // Atualiza informação

  teclaPressionada = meuTeclado.getKey();           // Leitura de qual tecla foi pressionada
```

```
  if (teclaPressionada){                        // Se alguma tecla for pressionada,
    switch(perfil) {
    case 1:
      perfil1();
      break;
    case 2:
      perfil2();
      break;
    }
  }
}

//---------------------------------
void buttons() {
  for (int i = 0; i < N_BUTTON; i++) {
    buttonState[i] = digitalRead(buttonPin[i]);             // Leitura do pino do botão
    buttonTimer[i] = millis() - lastBounce[i];              // corre o timer do botão

    if (buttonTimer[i] > buttonTimeOut) {                   // Se o tempo do botão for maior do que
a latência,
      if (buttonState[i] != buttonPState[i]) {              // Se o estado do botão for diferente do
prévio,

        lastBounce[i] = millis();                          // última vez pressionada igual ao tempo atual

        if (buttonState[i] == 0) {                         // Se o botão for pressionado
          if (i == 0) { Consumer.write(MEDIA_VOLUME_MUTE); }
          if (i == 1) {}

          Serial.print("Botao ");
          Serial.print(i);
          Serial.println(" on");                           // prints do sistema
          //oledCommands(12);

        } buttonPState[i] = buttonState[i];                 // Estado prévio == ao estado atual
      }
    }
  }
}

//---------------------------------
void rotaryEncoders() {
  for (int i = 0; i < N_ENCODERS; i++) {
    clkState[i] = digitalRead(clkPin[i]);                   // Estado do Rotary igual leitura do pino
Clk

    if (clkState[i] != rotaryPState[i]) {                   // Se a leitura do Clk for diferente do
estado prévio,
```

```
    if(digitalRead(dtPin[i]) != clkState[i]) {//              //    Se a leitura do pino Dt for diferente
do pino Clk,
      rotaryCounter[i]++;                                  //        contador do rotary ++

      if (i == 0) { Consumer.write(MEDIA_VOLUME_UP); }          //        função do primeiro
Rotary
      //if (i == 1) { Consumer.write(); }                       //        função do segundo Rotary
    } else {                                             // Se não,
      rotaryCounter[i]--;                                    //    contador do rotary --
      if (i == 0) { Consumer.write(MEDIA_VOLUME_DOWN); }        //    função do primeiro
Rotary
      //if (i == 1) { Consumer.write(MEDIA_VOLUME_UP); }        //    função do segundo
Rotary
    }
    Serial.print("Rotary ");
    Serial.print(i); Serial.print(" : ");
    Serial.println(rotaryCounter[i]);                        // prints de informações
    rotaryPState[i] = clkState[i];                           // estado prévio igual ao estado
   }
  }
}

//---------------------------------
void barra() {
  Keyboard.press(KEY_LEFT_ALT);          // aperta a tecla alt
  Keyboard.press(KEYPAD_4);              // aperta a tecla 4 do numpad
  Keyboard.press(KEYPAD_7);              // aperta a tecla 7 do numpad
  Keyboard.releaseAll();                 // solta todos os botõess
}

//---------------------------------
void iniciaExecutar(){
  Keyboard.press(KEY_LEFT_GUI);          // aperta a tecla windows,
  Keyboard.press('r');                   // abre o executar do windows
  Keyboard.releaseAll();                 // Solta as teclas
  delay(500);
  Keyboard.print("C?"); barra();
  Keyboard.print("delphos"); barra();
}

//---------------------------------
void finalizaExecutar(){
  Keyboard.press(KEY_LEFT_SHIFT);        // Pressiona a tecla shift
  Keyboard.println();                    // Pressiona a tecla Enter
  Keyboard.releaseAll();                 // Solta todas as teclas
  delay(100);
}
```

```
//--------------------------------
void perfil2() { //MODDED
  switch(teclaPressionada) {
    case '1':
      iniciaExecutar();
      Keyboard.print("num1.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case '2':
      iniciaExecutar();
      Keyboard.print("num2.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case '3':
      iniciaExecutar();
      Keyboard.print("num3.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case 'A':
      iniciaExecutar();
      Keyboard.print("num4.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case '4':
      iniciaExecutar();
      Keyboard.print("num5.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case '5':
      iniciaExecutar();
      Keyboard.print("num6.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case '6':
      iniciaExecutar();
      Keyboard.print("num7.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case 'B':
      iniciaExecutar();
      Keyboard.print("num8.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case '7':
      iniciaExecutar();
      Keyboard.print("num9.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
```

```
    case '8':
      iniciaExecutar();
      Keyboard.print("num10.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case '9':
      iniciaExecutar();
      Keyboard.print("num11.bat"); // Nome do arquivo .bat correspondente
      finalizaExecutar();
      break;
    case 'C':
      selectPerfil();             //Seleciona perfil
      break;
    default:
      Serial.print("Deu caca");
  }
}

//--------------------------------
void perfil1() { //DEFAULT
  switch(teclaPressionada) {
    case '1':                     //Seleciona tudo
      Consumer.write(MEDIA_PREVIOUS);
      Keyboard.releaseAll();
      oledCommands(1);
      break;
    case '2':                     //Cortar
      Consumer.write(MEDIA_PLAY_PAUSE);
      Keyboard.releaseAll();
      oledCommands(2);
      break;
    case '3':                     //Copiar
      Consumer.write(MEDIA_NEXT);
      Keyboard.releaseAll();
      oledCommands(3);
      break;
    case 'A':                      //Colar
      Keyboard.press(KEY_LEFT_ALT);
      Keyboard.press(KEY_F4);
      Keyboard.releaseAll();
      oledCommands(4);
      break;
    case '4':                     //Salvar
      Keyboard.press(KEY_LEFT_CTRL);
      Keyboard.print("s");
      Keyboard.releaseAll();
      oledCommands(5);
      break;
```

```
case '5':                       //Reabrir guia do navegador
  Keyboard.press(KEY_LEFT_CTRL);
  Keyboard.press(KEY_LEFT_SHIFT);
  Keyboard.print("t");
  Keyboard.releaseAll();
  oledCommands(6);
  break;
case '6':                       //Buscar
  Keyboard.press(KEY_LEFT_CTRL);
  Keyboard.print("f");
  Keyboard.releaseAll();
  oledCommands(7);
  break;
case 'B':                       //Captura de tela
  Keyboard.press(KEY_LEFT_SHIFT);
  Keyboard.press(KEY_LEFT_GUI);
  Keyboard.print("s");
  Keyboard.releaseAll();
  oledCommands(8);
  break;
case '7':                       //Bloqueia pc
  Keyboard.press(KEY_LEFT_GUI);
  Keyboard.print("l");
  Keyboard.releaseAll();
  oledCommands(9);
  break;
case '8':                       //Expolorador de arquivio
  Keyboard.press(KEY_LEFT_GUI);
  Keyboard.print("e");
  Keyboard.releaseAll();
  oledCommands(10);
  break;
case '9':                       //Museu Digital
  Keyboard.press(KEY_LEFT_GUI);
  Keyboard.press('r');
  Keyboard.releaseAll();
  delay(100);
  Keyboard.print("https?");
  barra();
  barra();
  Keyboard.print("kerlonr.vercel.app");
  finalizaExecutar();
  oledCommands(11);
  break;
case 'C':                       //Seleciona perfil
  selectPerfil();
  break;
default:
```

```
      Serial.print("Deu caca");
  }
}

//--------------------------------
void selectPerfil() {
  switch (perfil) {                                   // Analisa a variável
  case 1:                                            // Se for 1,
    perfil = 2;                                      // perfil == 2,
    Serial.print("Perfil selecionado: ");
    Serial.println(perfil);                             // printa informações,
    break;
  case 2:                                            // Se for 2,
    perfil = 1;                                      // perfil == 1,
    Serial.print("Perfil selecionado: ");
    Serial.println(perfil);                             // printa informações
    break;
  }
  oledSwitch = true;
}

//--------------------------------
void info() {
  if (oledSwitch == true) {
    display.clearDisplay();
    display.drawRect(0, 0, 123, 63, 1);
    display.setTextSize(2);
    display.setCursor(21, 7);
    display.print("DELPHOS");
    display.setTextSize(1);
    display.setCursor(18, 48);

    if(perfil == 1){
      display.print("Perfil: Default");
    }
    if(perfil == 2){
      display.print("Perfil: Modded");
    }

    display.display();
    oledSwitch = false;
  }
}

//--------------------------------
void credits() {
  display.clearDisplay();
  display.drawRect(0, 0, 123, 63, 1);
```

```
  display.setTextSize(1);
  display.setTextColor(1);
  display.setCursor(14, 33);
  display.setTextSize(2);
  display.setCursor(14, 12);
  display.display();
}

//--------------------------------
void delphosLogo() {
  display.clearDisplay();
  display.drawBitmap(128-90, 0, delphosMacro, 60, 60, 1);
  display.display();
  delay(oledTime);
}

//--------------------------------
void delphosTextLogo() {
  display.clearDisplay();
  display.drawBitmap(0, 0, delphosText, 128, 64, 1);
  display.display();
  delay(oledTimeSlow);
}

//--------------------------------
void clear() {
  display.clearDisplay();
  display.display();
}

//---------------------------------
void oledCommands(int mensagem) {
  oledSwitch = true; info();

  display.setTextSize(1);

  switch (mensagem) {
    case 1:
      display.setCursor(39, 30);
      display.print("Anterior");
      Serial.print("Anterior");
      display.display();
      break;
    case 2:
      display.setCursor(45, 30);
      display.print("Pausar");
      Serial.print("Pausar");
      display.display();
```

```
  break;
 case 3:
  display.setCursor(43, 30);
  display.print("Proxima");
  Serial.print("Proxima");
  display.display();
  break;
 case 4:
  display.setCursor(43, 30);
  display.print("Finalizar");
  Serial.print("Finalizar");
  display.display();
  break;
 case 5:
  display.setCursor(43, 30);
  display.print("Salvar");
  Serial.print("Salvar");
  display.display();
  break;
 case 6:
  display.setCursor(14, 30);
  display.print("Restaurar Chrome");
  Serial.print("Restaurar Chrome");
  display.display();
  break;
 case 7:
  display.setCursor(43, 30);
  display.print("Buscar");
  Serial.print("Buscar");
  display.display();
  break;
 case 8:
  display.setCursor(43, 30);
  display.print("Print");
  Serial.print("Print");
  display.display();
  break;
 case 9:
  display.setCursor(40, 30);
  display.print("Bloquear");
  Serial.print("Bloquear");
  display.display();
  break;
 case 10:
  display.setCursor(10, 30);
  display.print("Explorar Arquivos");
  Serial.print("Explorador Arquivos");
  display.display();
```

```
      break;
    case 11:
      display.setCursor(36, 30);
      display.print("Portfolio");
      Serial.print("Portfolio");
      display.display();
      break;
    case 12:
      display.setCursor(41, 30);
      display.print("Volume");
      Serial.print("Volume");
      display.display();
      break;
    default:
      Serial.print("Deu caca");
  }
}
```