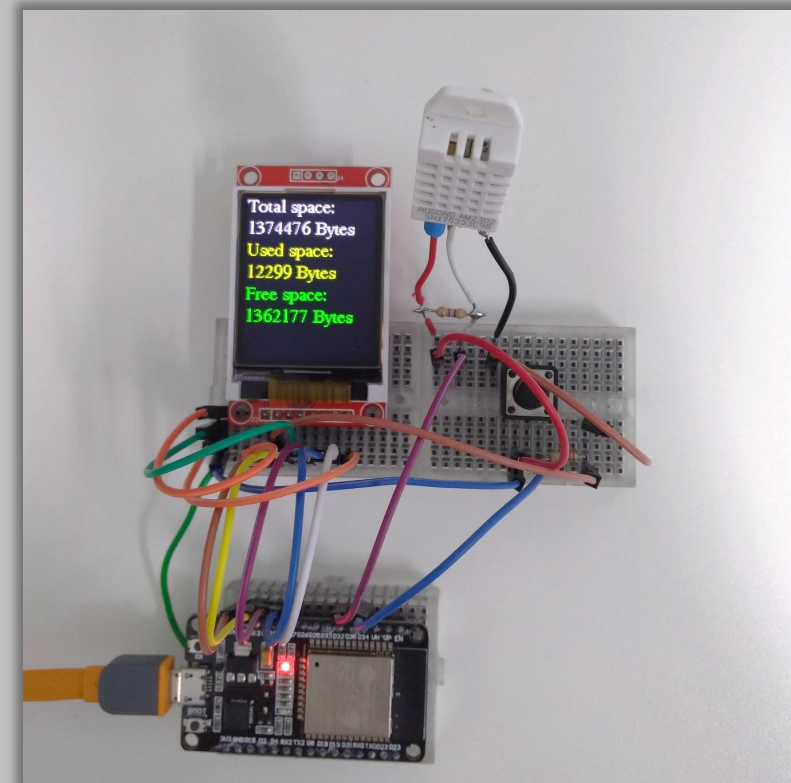
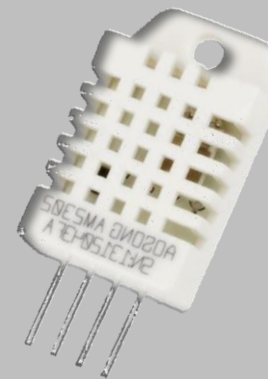


# ESP32 - Histórico de registros com SPIFFS

## Sistema de arquivos



SPIFFS



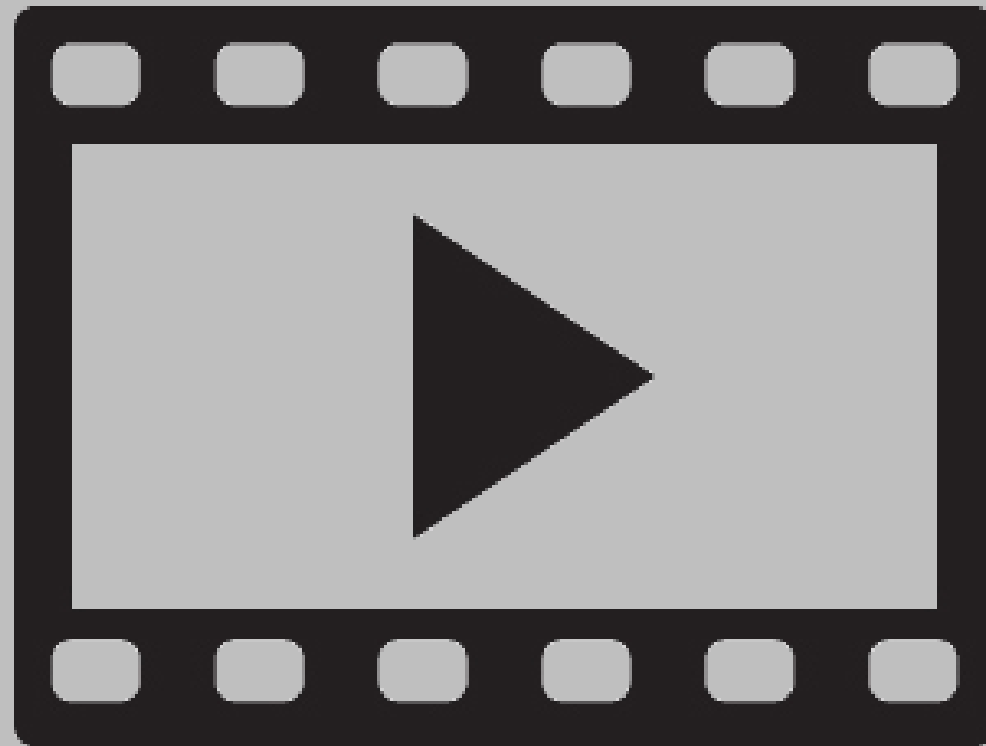
Por Fernando Koyanagi

# **Intenção dessa aula**

- 1. Apresentar uma aplicação com registros fixos em arquivo utilizando o SPIFFS do ESP32.**
- 2. Apresentar montagem e código fonte utilizado.**



# Demonstração







Em [www.fernandok.com](http://www.fernandok.com)

Seu e-mail



PRINCIPAL SOBRE FERNANDO K ARDUINO ESP8266 ESP32 LORAWAN MOTOR DISPLAY MATERIAIS DOWNLOAD

Receba o meu conteúdo  
GRATUITAMENTE

Insira aqui seu melhor email...

QUERO RECEBER GRÁTIS



## Motor de Passo Nema 23 com Driver TB6600 e Arduino Due

by **Fernando K Tecnologia** - 2:44 PM

Hoje vamos voltar a falar de Motor de Passo. Vamos utilizar um Nema 23 que será controlado por um Driver TB6600 e um Arduino Due. É p...

Leia mais



## ESP32 Longa Distância - LoRaWan

by **Fernando K Tecnologia** - 9:46 AM

Neste artigo vamos tratar da LoRaWAN, uma rede que vai longe gastando pouca energia. Mas, o quanto "longe"? Com o chip que uso no vídeo...

Leia mais



## Motor de HD com Arduino

by **Fernando K Tecnologia** - 2:00 PM

### QUAL ASSUNTO VOCÊ TEM

- ☐ Arduino
- ☐ ESP8266
- ☐ ESP32
- ☐ Motor
- ☐ Display
- ☐ Sensor

You may select multiple answers.

[Exibir resultados](#)


Votos até o momento: 32

Dias restantes para votar: 49

### FACEBOOK



# forum.fernandok.com



**Fórum Fernando K Tecnologia**  
Fórum sobre dúvidas com relação ao conteúdo disponibilizado pelo Fernando Koyanagi

[www.fernandok.com](#) [/fernandokoyanagi](#) [/fernandokoyanagi](#) [/fernandok\\_oficial](#) [/fernandok\\_oficial](#)



Links rápidos







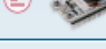

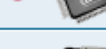



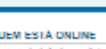

fernandokoyanagi

Bem-vindo: 05/Out/2018, 11:16

A sua última visita foi em 10/Set/2018, 15:47

Assinalar todos os fóruns como lidos

SUPPORT: FÓRUM FERNANDOK	TÓPICOS	MENSAGENS	ÚLTIMA MENSAGEM
 <div><b>Feedback</b> Dúvidas, críticas ou sugestões sobre o Fórum FernandoK. Para demais questões utilize o fórum correto.</div>	6	11	<b>Re: O russo voltou</b> por Iperito  01/Out/2018, 08:25

FERNANDO K	TÓPICOS	MENSAGENS	ÚLTIMA MENSAGEM
 <div><b>Arduino</b> Projetos de arduino</div>	31	79	<b>skard y txgji</b> por Soresorcem  05/Out/2018, 10:55
 <div><b>ESP32</b> Projetos de ESP32</div>	29	62	<b>Duvidas sobre como instalar a...</b> por Marcelo Jorge  04/Out/2018, 15:52
 <div><b>ESP8266</b> O ESP8266 é um microcontrolador do fabricante chinês Espressif que inclui capacidade de comunicação por Wi-Fi.</div>	24	51	<b>Re: NodeMCU não conecta em qu...</b> por rearsilva  04/Out/2018, 14:39
 <div><b>LoRa</b> Projetos com LoRa</div>	11	31	<b>Projeto de irrigação de jardim</b> por marlonc  04/Out/2018, 21:30
 <div><b>STM32</b> Projetos com STM32</div>	3	8	<b>Re: Imprecisão de tempo de de...</b> por biaroto  12/Set/2018, 09:15
 <div><b>Motor</b> Projetos com motor</div>	5	11	<b>Re: impressora 3d com motor dc</b> por Magetron  24/Set/2018, 19:06
 <div><b>Display</b> Projetos com Display</div>	4	11	<b>Re: Alguem conhece o VIRTUINO...</b> por Joel Luz  21/Set/2018, 11:39

**QUEM ESTÁ ONLINE**

No total, há **4** usuários online :: 2 usuários registrados, 0 anônimo e 2 visitantes (baseado em usuários ativos nos últimos 5 minutos)  
O recorde de usuários online foi de **19** em 11/Set/2018, 05:37

Usuários registrados: alberto, **fernandokoyanagi**  
Legenda: Administradores, Moderadores globais

**ANIVERSÁRIOS**

Não há aniversários hoje

**ESTATÍSTICAS**

Total de mensagens **703** • Total de tópicos **114** • Total de membros **469** • Novo usuário: **Soresorcem**

...

Powered by phpBB® Forum Software © phpBB Limited  
Tradução por: Suporte phpBB  
Painel de Controle de Administração



# Instagram

fernandok\_oficial



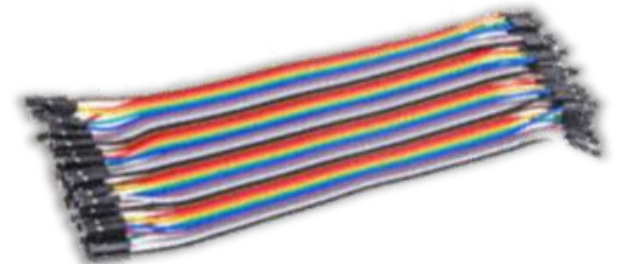
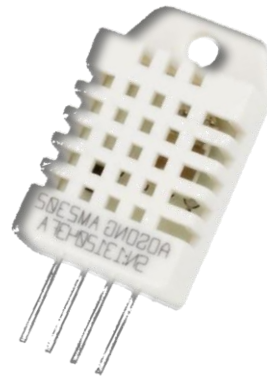
# Telegram

fernandok\_oficial

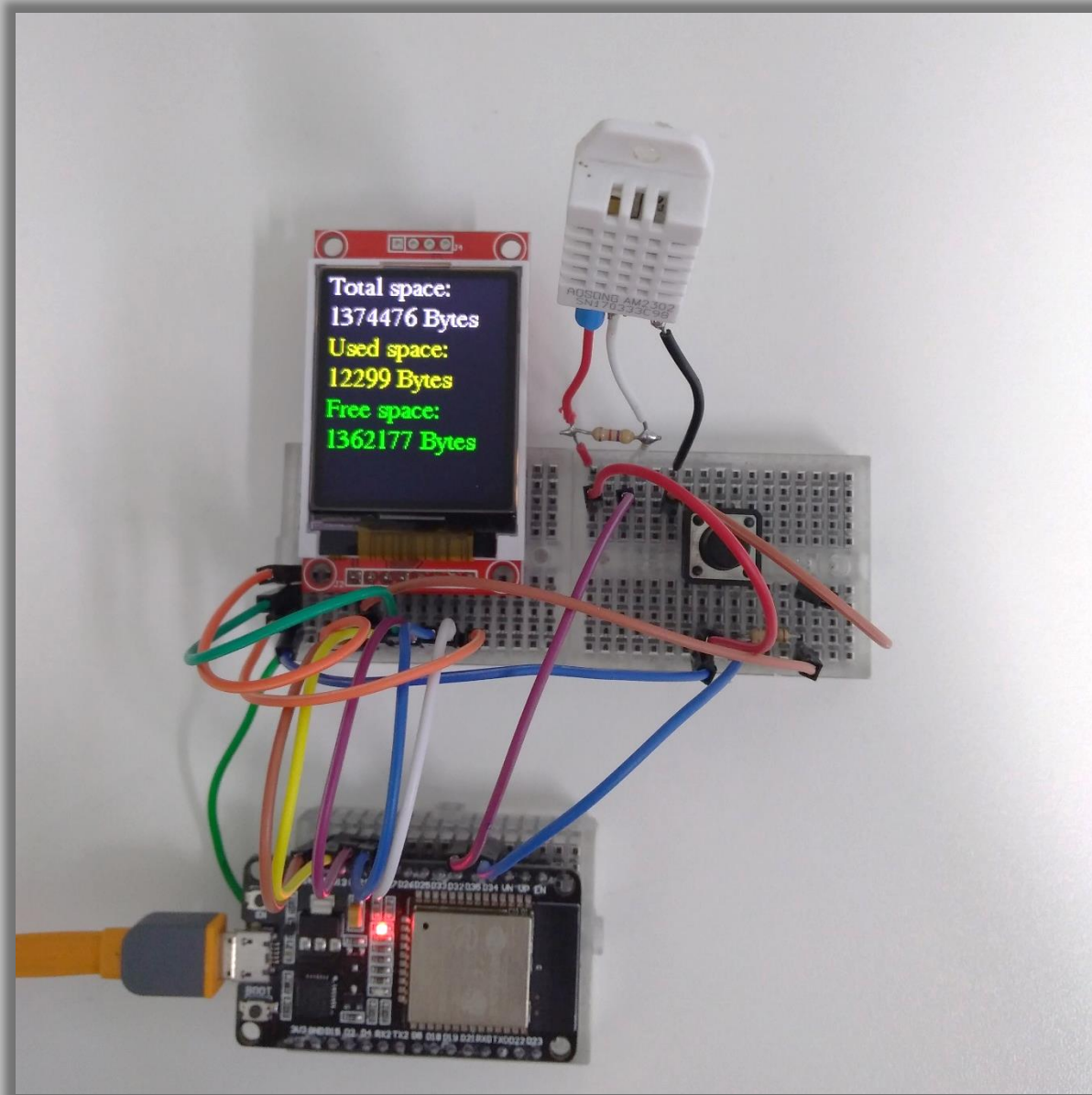


# Recursos usados

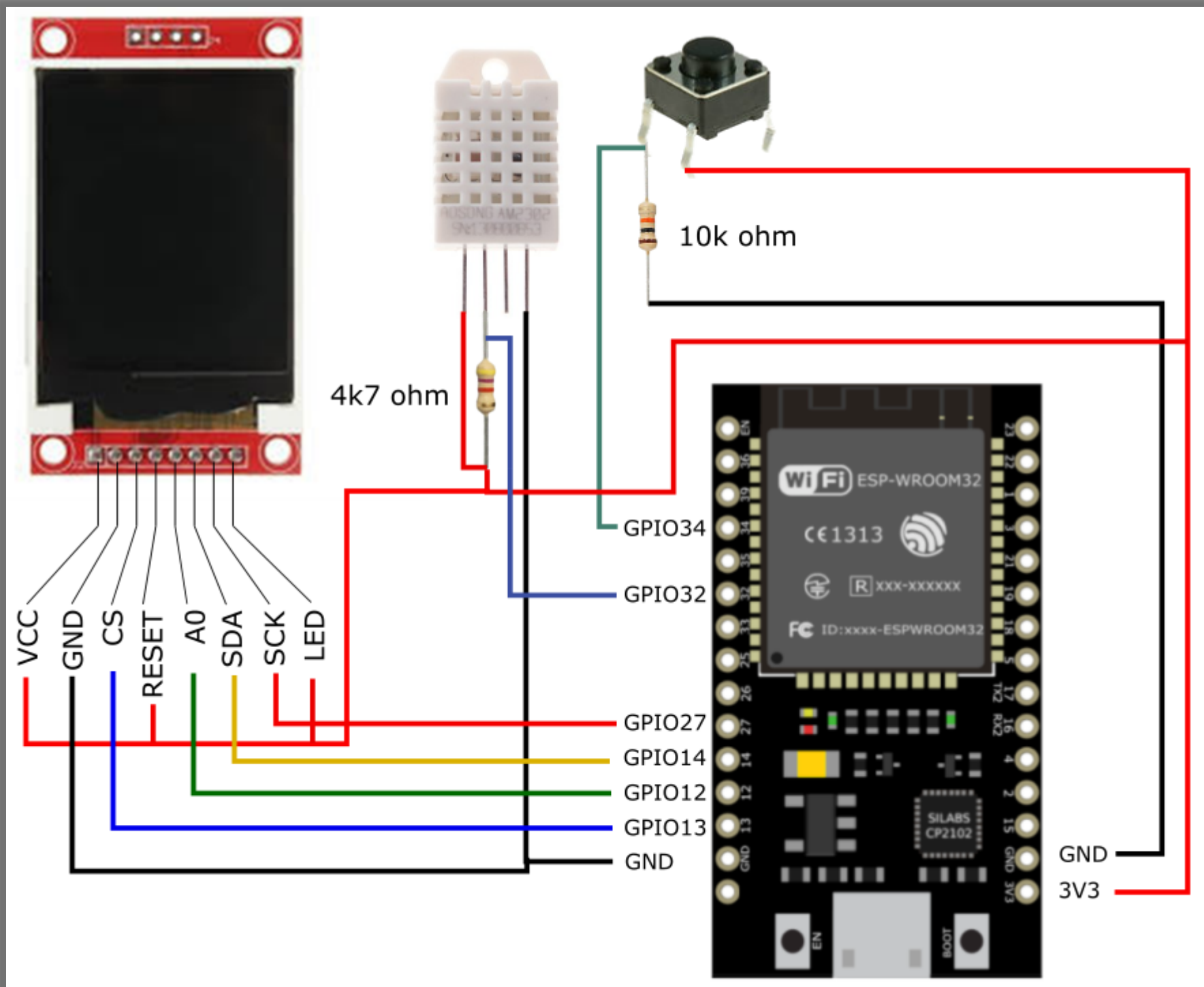
- ESP32 WROOM Dev Board
- Display SPI TFT 1.8"
- DHT22
- Resistor 4k7 ohm
- Botão
- Resistor 10k ohm
- Jumpers

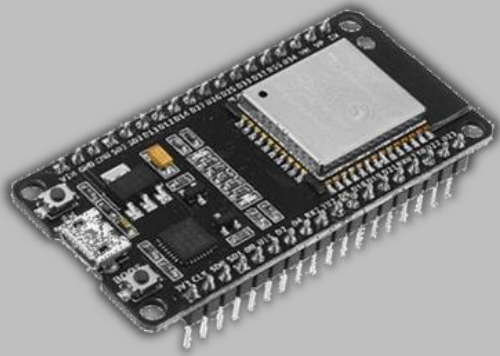


# Montagem









# Código

# Instalação de Bibliotecas

Instale na sua Arduino IDE as **bibliotecas** abaixo:

Adafruit GFX

<https://github.com/adafruit/Adafruit-GFX-Library>

Adafruit ST7735

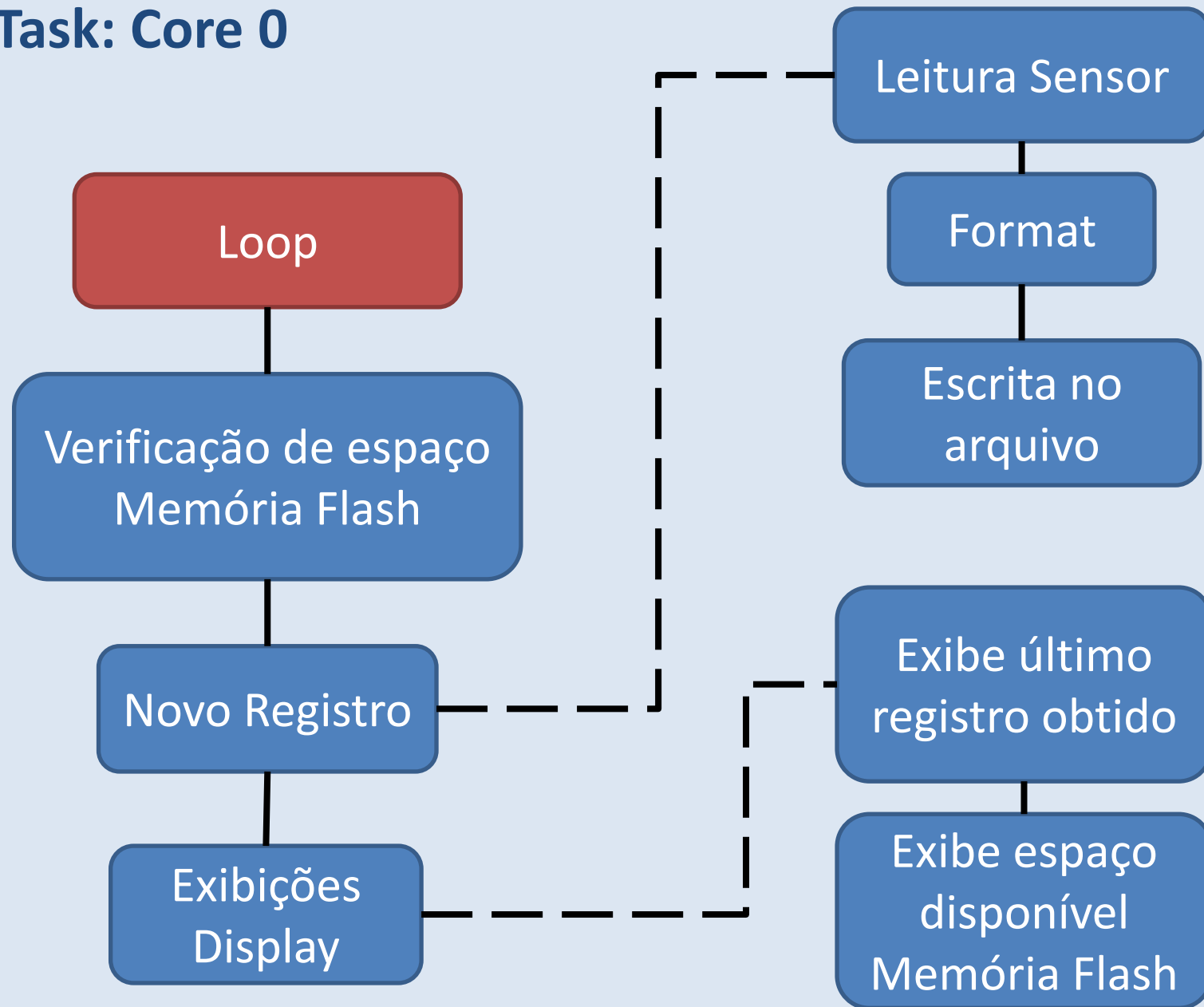
<https://github.com/adafruit/Adafruit-ST7735-Library>

SimpleDHT

<https://github.com/winlinvip/SimpleDHT>

# Código *Diagrama*

## Task: Core 0





# Código *Diagrama*

## Task: Core 1

Evento botão

Exclusão do  
arquivo

Exibição  
Display

# Código ESP32 *Declarações e Variáveis*

```
#include <Adafruit_GFX.h> // Biblioteca de gráficos
#include <Adafruit_ST7735.h> // Biblioteca do hardware ST7735
#include <SPI.h> // Biblioteca para comunicação SPI
#include <Fonts/FreeSerif9pt7b.h> // Fonte Serif que é usada no display
#include <SimpleDHT.h> // lib do DHT
#include "FS_File_Record.h" // Nossa lib personalizada do SPIFFS

// Pinos do display
#define TFT_DC 12 // A0
#define TFT_CS 13 // CS
#define TFT_MOSI 14 // SDA
#define TFT_CLK 27 // SCK
#define TFT_RST 0 // RESET
#define TFT_MISO 0 // MISO

// Pino do botão de exclusão de arquivo
const int buttonPin = 34;
```

# Código ESP32 *Declarações e Variáveis (continuação)*

```
// Tamanho dos registros de temperatura e umidade (13 caracteres), exemplo:
// 100.00;100.00
// temp;umid
const int sizeofRecord = 13;

//100.00
const int integralPartSize = 3, decimalPartSize = 2;

// Objeto da nossa lib que recebe o nome do arquivo e tamanho fixo de registro
FS_File_Record ObjFS("/dht22.bin", sizeofRecord);

// Flag que impede que o display pisque ao pressionar o botão por muito tempo
bool flag = false;

// Pino do DHT22
const int pinDHT22 = 32;
// Objeto do dht22
SimpleDHT22 dht22(pinDHT22);
// Objeto do display
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_MOSI, TFT_CLK, TFT_RST);

// Altura da fonte, usada na função resetDisplay
int fontHeight = 7;
```

# Código ESP32 *Declarações e Variáveis (continuação) e Setup*

```
// Variáveis usada para contagem/comparação de tempo (millis) na função timeout
long millisRefShowSpace;
bool flagShowSpace = false;

// String que recebe as mensagens de erro
String errorMsg;

// Variável que guarda o último registro obtido
String lastRecord = "";

void setup()
{
  // Inicializamos o display
  tft.initR(INITR_BLACKTAB);

  // Iniciamos a Serial com velocidade de 115200
  Serial.begin(115200);
  // Seta botão como entrada (INPUT)
  pinMode(buttonPin, INPUT);

  // Exibe na serial "Starting..." para debug
  Serial.print("Starting...");
```



# Código ESP32 Setup (continuação)

```
// Se não foi possível iniciar o File System, exibimos erro e reiniciamos o ESP
if(!ObjFS.init())
{
    Serial.println("File system error"); delay(1000); ESP.restart();
}
// Exibimos mensagem
Serial.println("File system ok");
// Se o arquivo não existe, criamos o arquivo
if(!ObjFS.fileExists())
{
    Serial.println("New file");
    ObjFS.newFile(); // Cria o arquivo
}

// Iniciamos uma task que irá ler o botão de exclusão
xTaskCreatePinnedToCore(
    buttonEvent, //Função que será executada
    "buttonEvent", //Nome da tarefa
    10000, //Tamanho da pilha
    &tft, //Parâmetro da tarefa (no caso não usamos)
    2, //Prioridade da tarefa
    NULL, //Caso queira manter uma referência para a tarefa que vai ser criada (no caso não precisamos)
    0); //Número do core que será executada a tarefa (usamos o core 0 para o loop ficar livre com o core 1)
```


# Código ESP32 Setup (continuação)

Para **buscar** um registro, a função usada é a **findRecord**. Ela não é necessária neste exemplo mas é uma função **importante** quando se trata de armazenamento em arquivo.

```
resetDisplay();

// Exemplo de busca (FIND)
// Obs: O primeiro registro se posiciona na pos 0 (zero)
// String reg = ObjFS.findRecord(10);
// tft.println(reg);

// Exibimos o arquivo
showFile();
} // fim do setup
```



# Código ESP32 *resetDisplay*

```
// Limpa o display e posiciona o  
cursor no início  
void resetDisplay()  
{  
    tft.setFont(&FreeSerif9pt7b);  
    tft.fillScreen(ST77XX_BLACK);  
    tft.setTextColor(ST7735_WHITE);  
    tft.setCursor(0,fontHeight+5);  
    tft.setTextSize(1);  
}
```

# Código ESP32 *showFile*

```
// Exibimos no display o arquivo, pausando a exibição a cada 6 registros
void showFile()
{
    int count = 0;
    String linha = "";
    // Exibe na serial e no display o início do arquivo
    Serial.println("# Begin of file #");
    tft.println("# Begin of file #");

    errorMsg = "";

    // Posiciona o ponteiro do arquivo no início
    ObjFS.rewind();
```



# Código ESP32 *showFile (continuação)*

```
// Exibe todos os registros até o fim
while(ObjFS.readFileNextRecord(&linha, &errorMsg) && linha != "")
{
    Serial.println(linha);
    count++;
    // A cada 6 registros, pausamos e resetamos o display
    if(count % 6 == 0)
    {
        //Exibe "...\" sinalizando que ainda não é o fim do arquivo
        tft.println("...");
        // Aguarda 1.5s para poder visualizar os valores
        delay(1500);
        // Limpa display
        resetDisplay();
    }
    tft.println(linha);
}
// Se existir mensagem de erro exibe na serial e no display
if(errorMsg != "")
{
    Serial.println(errorMsg);
    tft.println(errorMsg);
}
```

# Código ESP32 *showFile (continuação) e Loop*

```
// Exibe na serial e no display o fim do arquivo
Serial.println("# End of file #");
tft.println("# End of file #");
delay(1500);
}
void loop()
{
    // Se não houver memória disponível, exibe e reinicia o ESP
    if(!ObjFS.availableSpace())
    {
        Serial.println("Memory is full!");
        tft.println("Memory is full!");
        delay(10000);
        return;
    }

    // Lê temperatura e umidade do DHT
    String values = readDHTValues();
    // Escrevemos no arquivo e exibimos erro ou sucesso na serial para debug
    if(values != "" && !ObjFS.writeFile(values, &errorMsg))
        Serial.println(errorMsg);
    else
        Serial.println("Write ok");
}
```

Loop

Verificação de espaço  
Memória Flash

Leitura Sensor

Escrita no  
arquivo

# Código ESP32 *Loop (continuação) e readDHTValues*

```
// Atribui para a variável global a última amostra
lastRecord = values;
// Exibe a última amostra no display
showLastRecord();
// Exibimos o espaço total, usado e disponível no display, de tempo em tempo
showAvailableSpace(); delay(2500); // Aguarda 2500ms
}

// Função que lê e formata os dados de temperatura e umidade
String readDHTValues()
{
    float temperature = 0, humidity = 0; int err = SimpleDHTErrSuccess;
    // Lê sensor
    if ((err = dht22.read2(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess)
    {
        Serial.print("Read DHT22 failed, err=");
        Serial.println(err);
        delay(1000);
        return "-----";
    }
    // Retorna valores formatados, separados por ponto-virgula
    return formatValue(temperature)+";"+formatValue(humidity);
}
```

Exibições  
Display

Leitura Sensor

# Código ESP32 *showLastRecord e showAvailableSpace*

Exibições  
Display

```
// Exibe última amostra de temperatura e umidade
obtida
void showLastRecord()
{
    resetDisplay();
    tft.println("Last record:");
    tft.println(lastRecord);
}

// Exibe o espaço total, usado e disponível no display
void showAvailableSpace()
{
    if(timeout(10000, &millisRefShowSpace, &flagShowSpace))
    {
        Serial.println("Space: "+String(ObjFS.getTotalSpace())+" Bytes");
        Serial.println("Used: "+String(ObjFS.getUsedSpace())+" Bytes");
        resetDisplay();
        tft.println("Total space:");
        tft.println(String(ObjFS.getTotalSpace())+" Bytes");
        tft.setTextColor(ST7735_YELLOW);
    }
}
```



# Código ESP32 *showAvailableSpace* (continuação)

Exibições  
Display

```
tft.println("Used space:");
tft.println(String(ObjFS.getUsedSpace())+" Bytes");
tft.setTextColor(ST77XX_GREEN);
tft.println("Free space:");
tft.println(String(ObjFS.getTotalSpace()-ObjFS.getUsedSpace())+" Bytes");

// Registramos 4 vezes enquanto a mensagem aparece no display
for(int i=0; i<4; i++)
{
    // Lê temperatura e umidade do DHT
    String values = readDHTValues();

    // Escrevemos no arquivo e exibimos erro ou sucesso na serial para debug
    if(values != "" &&!ObjFS.writeFile(values, &errorMsg))
        Serial.println(errorMsg);
    else
        Serial.println("Write ok");

    delay(2500);
}
} // Fim do if(timeout...)
}
```

# Código ESP32 *buttonEvent*

```
// Função executada pela task de evento do botão
void buttonEvent(void* display)
{
    TickType_t taskDelay;

    // Ponteiro do display que é enviado para a função 'showFileDeleted'
    Adafruit_ST7735 *p_tft = (Adafruit_ST7735*)display;

    // IMPORTANTE: A tarefa não pode terminar, deve ficar presa em um loop infinito
    while(true)
    {
        // Se o botão foi pressionado e a flag estiver "false"
        if(digitalRead(buttonPin) == HIGH && !flag)
        {
            // Tenta excluir o arquivo
            if(ObjFS.destroyFile())
            {
                Serial.println("File deleted");
                showFileDeleted(true);
                lastRecord = "";
            }
        }
    }
}
```

Evento botão

Exclusão do  
arquivo

Exibição  
Display

# Código ESP32 *buttonEvent* (continuação)

```
else
{
    Serial.println("Failed to delete file");
    showFileDeleted(false);
    // Se aconteceu algum erro reiniciamos o ESP
    ESP.restart();
}
// Sinalizamos que o botão foi pressionado
flag = true;
}
else
if(digitalRead(buttonPin) == LOW && flag) // Se o botão foi solto e a flag estiver "true"
{
    // Sinalizamos que o botão foi solto
    showLastRecord();
    flag = false;
}

taskDelay = 10 / portTICK_PERIOD_MS;
// Executamos um delay de 10ms, os delays executado nas xTasks são diferentes
vTaskDelay(taskDelay);
//IMPORTANTE: SEMPRE DEIXAR UM DELAY PARA ALIMENTAR WATCHDOG
}
```

Evento botão

Exibição  
Display

# Código ESP32 *showFileDeleted*

```
// Posiciona cursor no centro e exibe a mensagem "FILE DELETED" em amarelo
void showFileDeleted(bool sucess)
{
    // Posição y aonde o texto será exibido
    int y = (tft.height()/2)-(fontHeight*2);

    resetDisplay();
    tft.setTextColor(ST7735_YELLOW); // Define cor do texto amarela
    tft.setCursor(0, y); // Posiciona no centro de eixo y

    // Se foi possível excluir
    if(sucess)
    {
        // Exibe mensagem "FILE DELETED"
        tft.println(" FILE"); tft.println(" DELETED!");
    }
    else // Se não foi possível excluir
    {
        // Exibe mensagem "CANNOT DELETE"
        tft.println(" CANNOT"); tft.println(" DELETE");
    }
    // Define cor do texto branca
    tft.setTextColor(ST7735_WHITE);
}
```

Exibição  
Display

# Código ESP32 Header *FS\_File\_Record*: Lista de funções

```
class FS_File_Record
{
    public:
    FS_File_Record(String, int);
    FS_File_Record(String);
    bool init();
    bool readFileLastRecord(String *, String *);
    bool destroyFile();
    String findRecord(int);
    bool rewind();
    bool writeFile(String, String *);
    bool seekFile(int);
    bool readFileNextRecord(String *, String *);
    String getFileName();
    void setFileName(String);
    int getSizeRecord();
    void setSizeRecord(int);
    void newFile();
    bool fileExists();
    bool availableSpace();
    int getTotalSpace();
    int getUsedSpace();
};
```

Baixe os arquivos .cpp e .h da biblioteca **FS\_File\_Record** anexados junto ao projeto!



FS\_File\_Record.h

Em [www.fernandok.com](http://www.fernandok.com)

Download arquivos PDF, **INO** e outros

