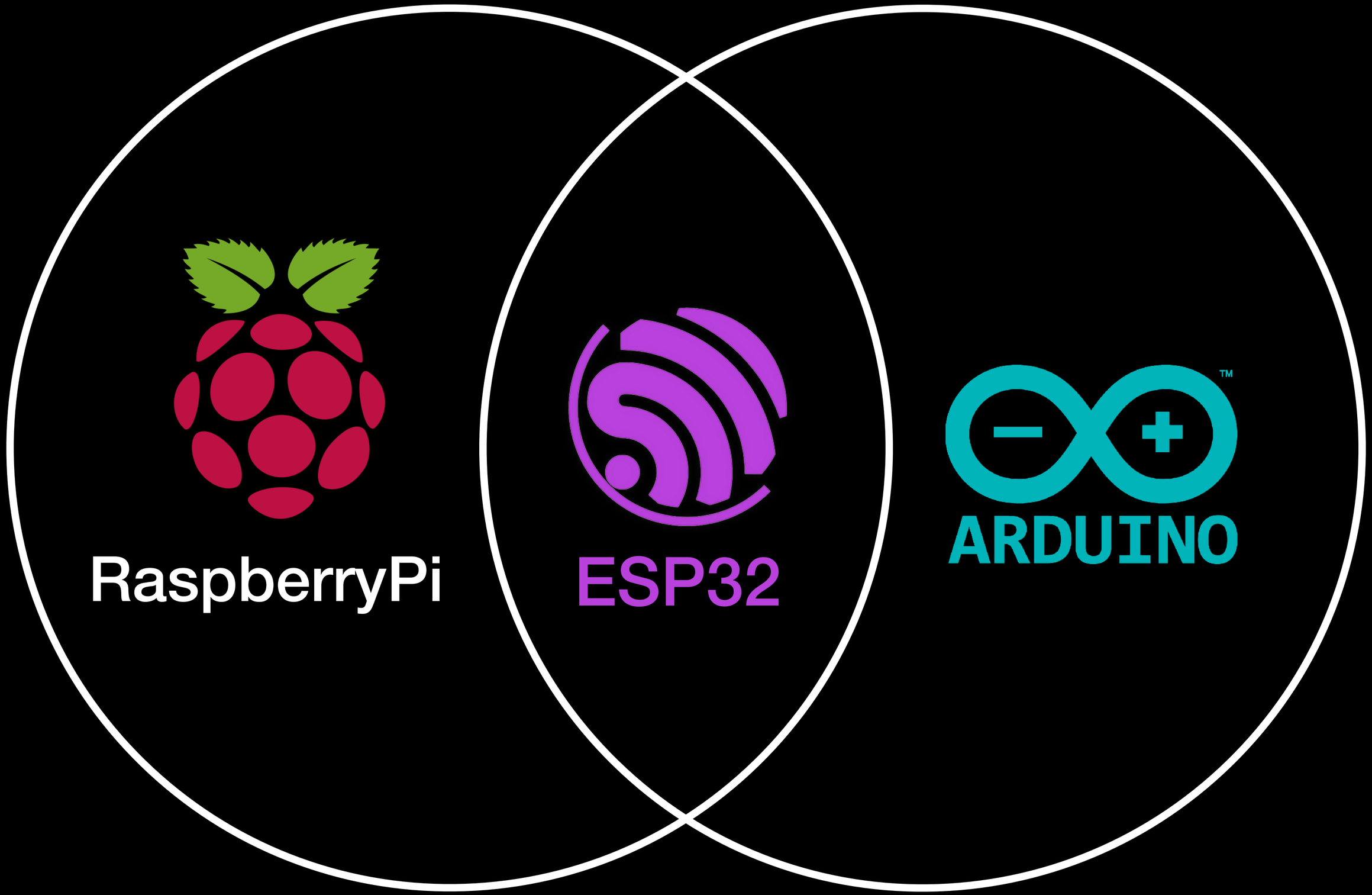
Projeto 01

Coisas Configuráveis – Teoria

Jan K. S. – janks@puc-rio.br

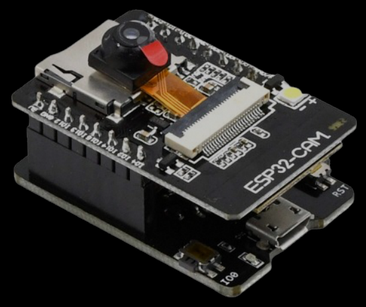
ENG4051 – Projeto Internet das Coisas

Coisas



Microcontrolador Usado no Curso

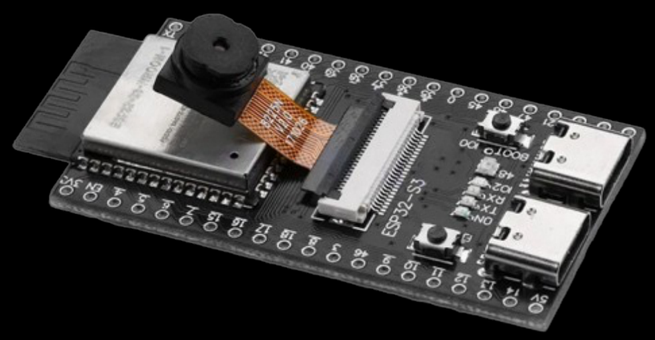
ESP32



ESP32-CAM

ESP32-S3 N32R8

ESP32-S3-BOX-3B



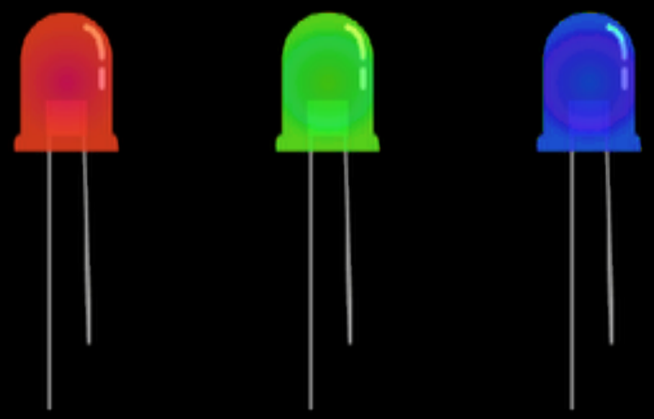
ESP32-S3-CAM

LILYGO ESP32 T-Display

Exemplos de Modelos do ESP32

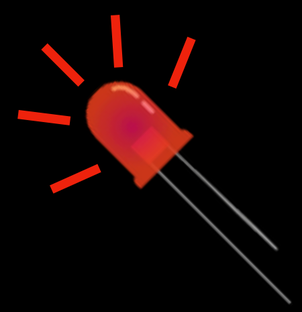
	Raspberry Pi 5	ESP32-S3-Cam	Arduino Mega (clone)
Processador	4 núcleos, 2.4GHz	2 núcleos, 240MHz	1 núcleo, 16MHz
RAM	4GB ou 8GB	512KB SRAM + 8MB PSRAM	8KB SRAM
Armazenamento	SD (até 2TB)	16MB Flash + SD (até 32GB)	256KB Flash
Pinos Digitais	28 pinos	34 pinos	54 pinos
Pinos Analógicos	não tem	20 pinos (12 bits)	16 pinos (10 bits)
Pinos PWM	1 pino	8 pinos	15 pinos
Wi-Fi	2.4GHz e 5GHz	2.4GHz	não tem
Bluetooth	5.0	5.0	não tem
Ethernet	Gigabit	não tem	não tem
Preço (2024)	R$ 650 - R$ 1000	R$ 130	R$ 90

Comparativo entre Raspberry Pi, ESP32 e Arduino Mega

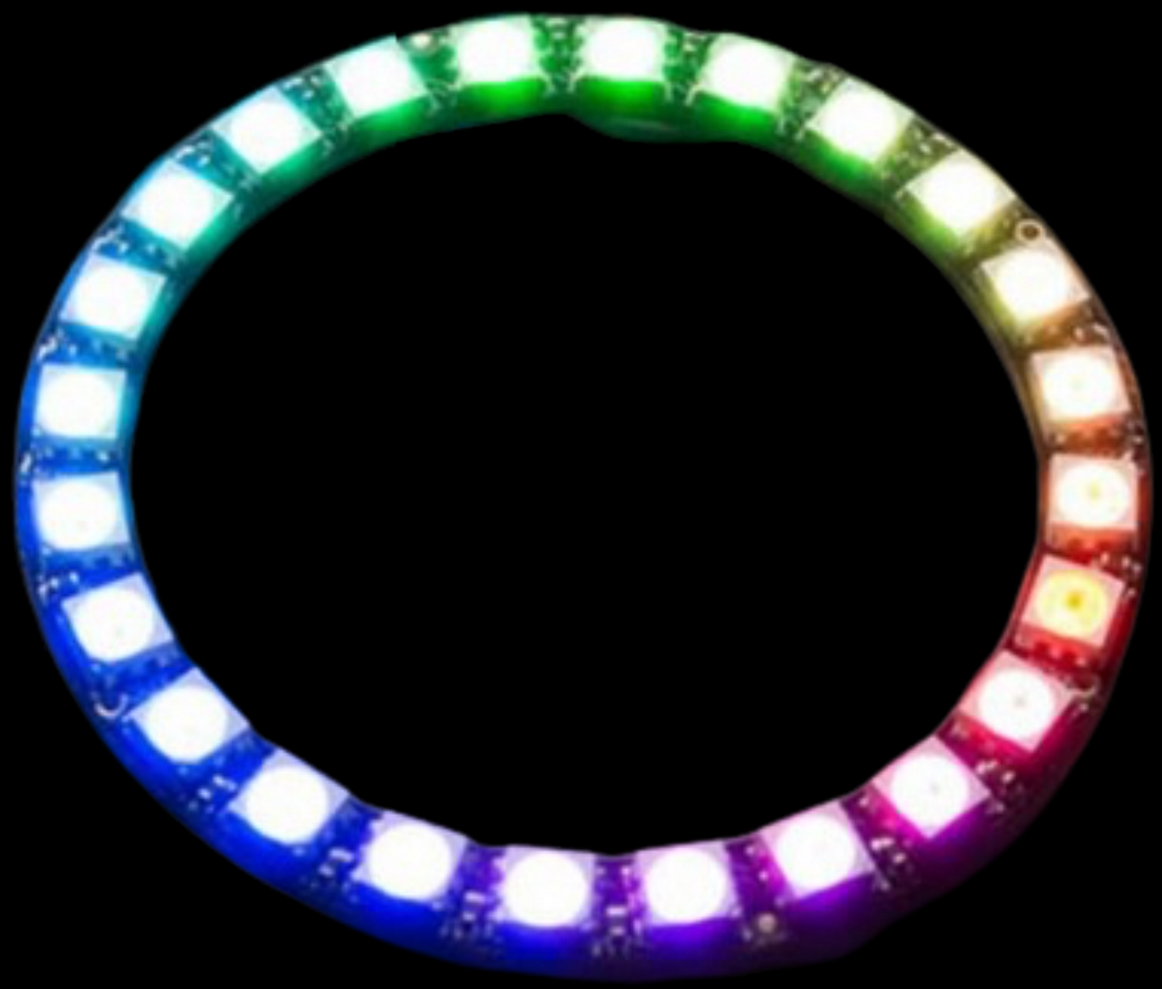


LED

```c
int pinoDoLED = 40;

void setup () {
  pinMode(pinoDoLED, OUTPUT);
}

                  // em algum lugar do código...

digitalWrite(pinoDoLED, LOW);

digitalWrite(pinoDoLED, HIGH);
```

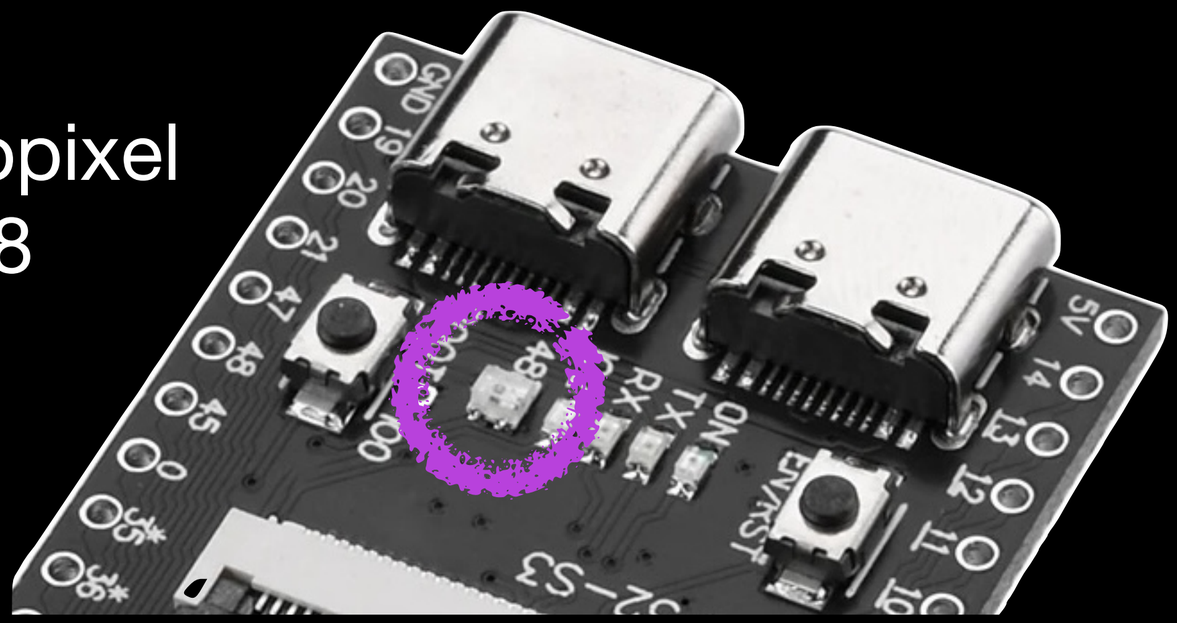


Código para Acender e Apagar LEDs

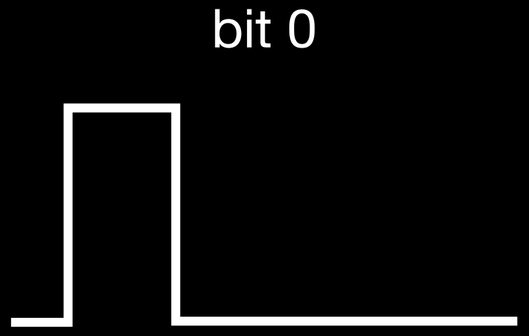


LED RGB Neopixel

LED RBG Neopixel
no pino 48



bit 0



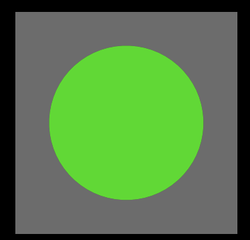
G R B

00000000 11111111 00000000

bit 1

G R B

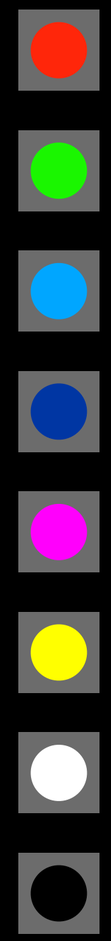
11111111 00000000 00000000



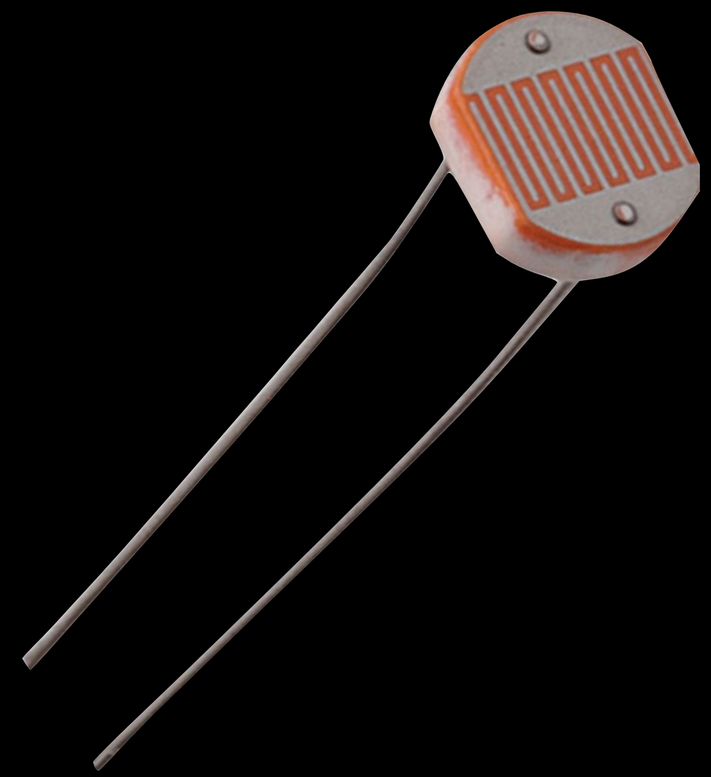
Codificação para a Cor do LED

```
    neopixelWrite(RGB_BUILTIN, vermelho, verde, azul);
```

```
neopixelWrite(RGB_BUILTIN, 255, 0,   0);
neopixelWrite(RGB_BUILTIN, 0,   255, 0);
neopixelWrite(RGB_BUILTIN, 0,   0,   255);
neopixelWrite(RGB_BUILTIN, 0,   0,   100);
neopixelWrite(RGB_BUILTIN, 255, 0,   255);
neopixelWrite(RGB_BUILTIN, 255, 255, 0);
neopixelWrite(RGB_BUILTIN, 255, 255, 255);
neopixelWrite(RGB_BUILTIN, 0,   0,   0);
```

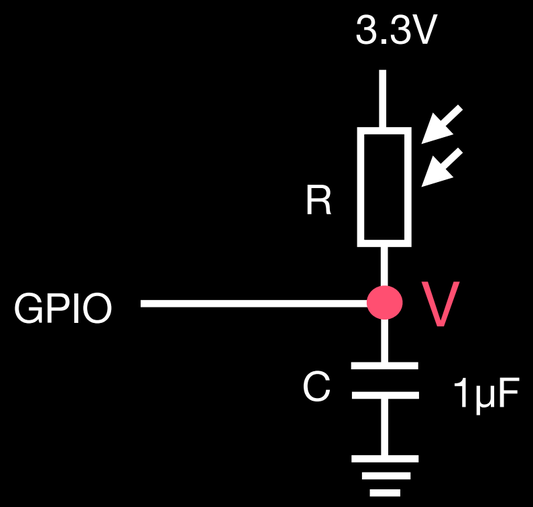


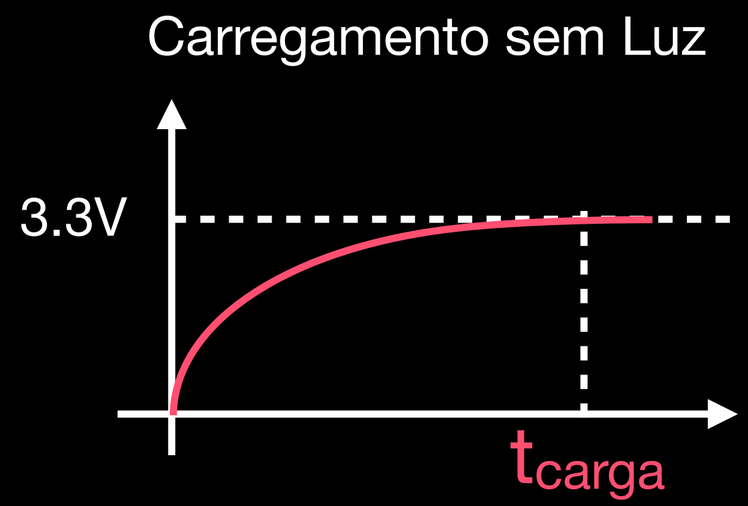
Controle da Cor do LED no Código

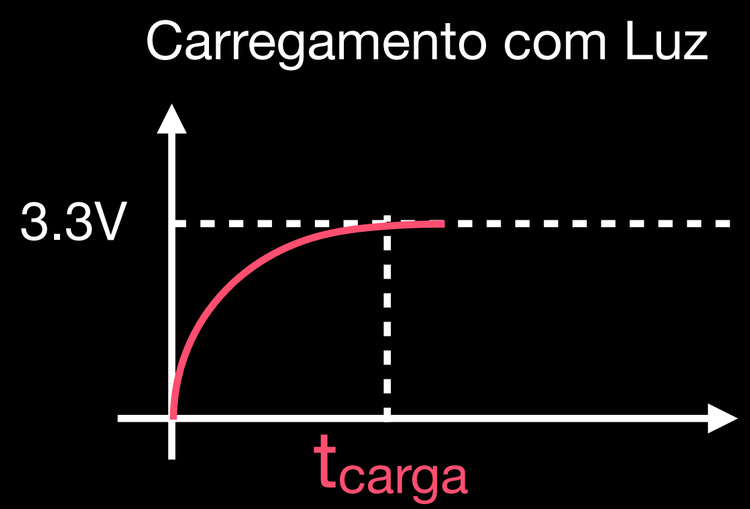


Sensor de Luz

RaspberryPi

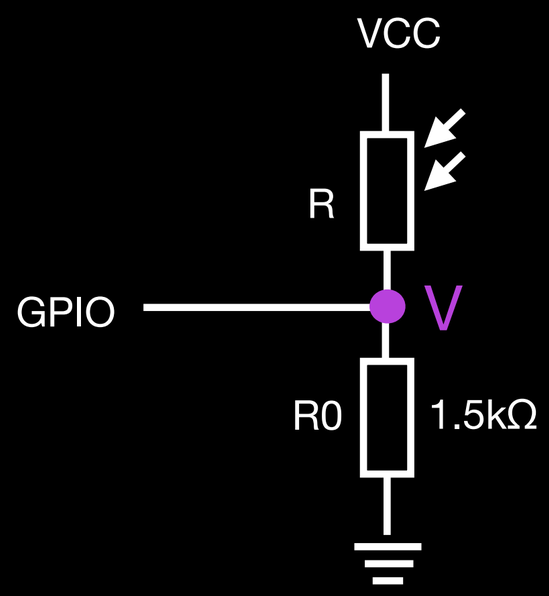






ARDUINO

ESP32



$$V = V_0 \frac{R_0}{R + R_0}$$

Leitura do Sensor de Luz no Raspberry Pi x Arduino / ESP32

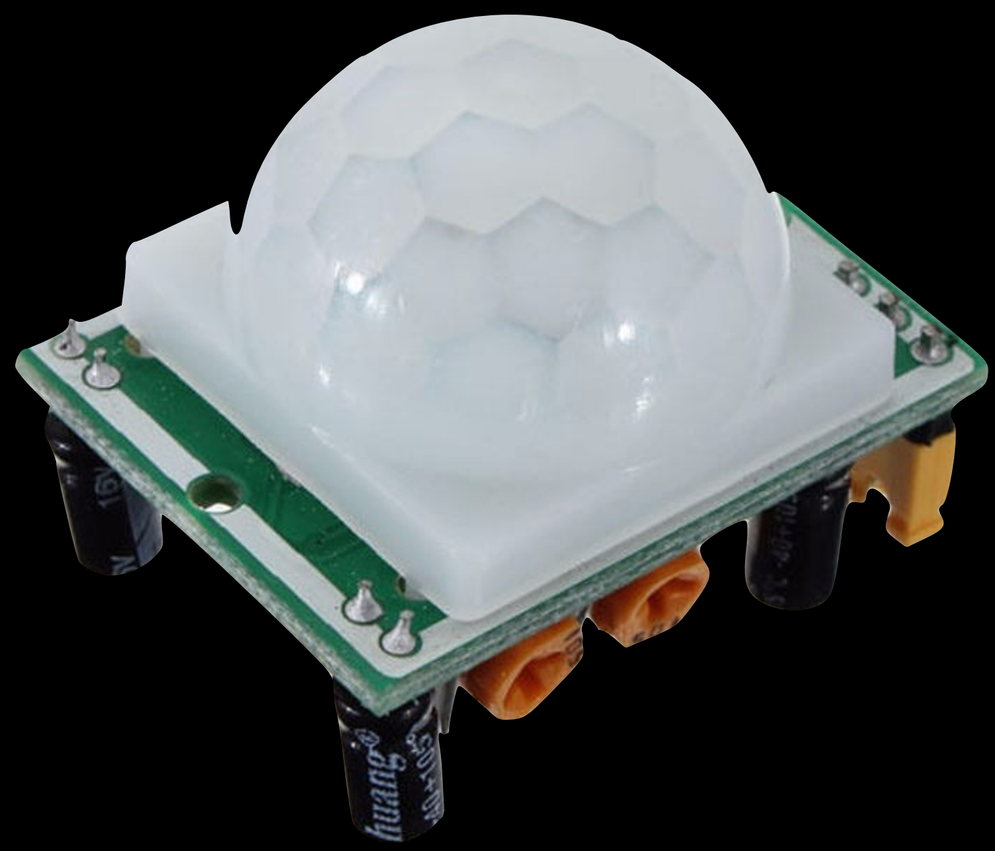
Conversor Analógico-Digital de 10 bits

```
int leituraAnalogica = analogRead(pino); // entre 0 e 1023
int porcentagemLuz = map(leituraAnalogica, 0, 1023, 0, 100);
```

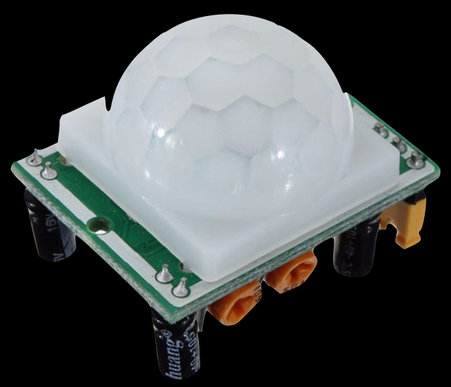
Conversor Analógico-Digital de 12 bits

```
int leituraAnalogica = analogRead(pino); // entre 0 e 4096
int porcentagemLuz = map(leituraAnalogica, 0, 4096, 0, 100);
```

Leitura Analógica no Código: Arduino x ESP-32

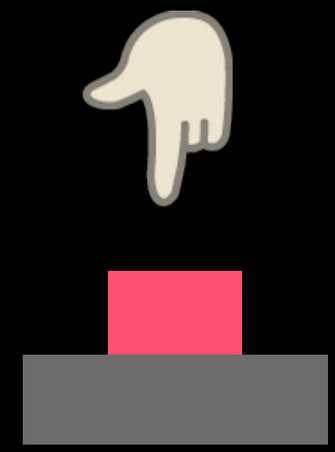


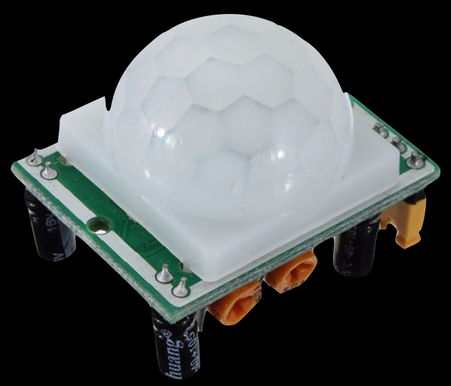
Sensor de Movimento



Movimento detectado!

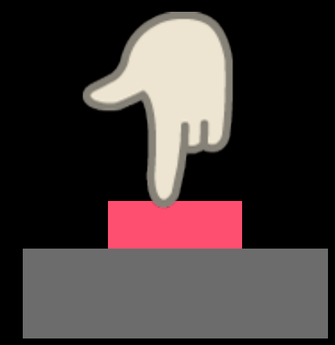
3.3V





Inércia detectada!

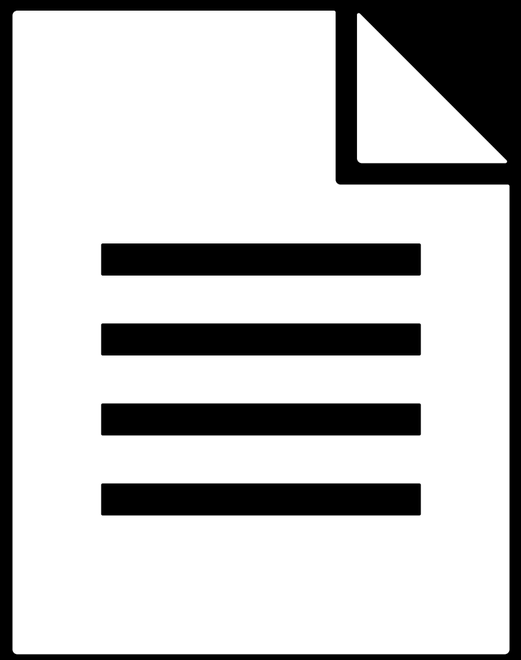
0V



Equivalência dos Eventos do Sensor de Movimento com os de um Botão

```
#include <GFButton.h>

GFButton sensorDeMovimento(48);

void movimentoDetectado (GFButton& sensor) {
    Serial.println("Detectei movimento!");
}

void inerciaDetectada (GFButton& sensor) {
    Serial.println("Detectei inércia!");
}

void setup () {
    Serial.begin(9600);
    sensorDeMovimento.setReleaseHandler(movimentoDetectado);
    sensorDeMovimento.setPressHandler(inerciaDetectada);
}

void loop () {
    botao.process();
}
```

Leitura Analógica no Código: Arduino x ESP-32



Arquivos

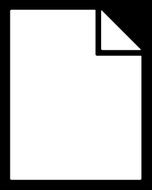
memória Flash

partitions.csv

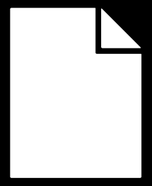
```
# Name,   Type, SubType, Offset,  Size, Flags
nvs,      data, nvs,     0x9000,  0x5000,
otadata,  data, ota,     0xe000,  0x2000,
app0,     app,  ota_0,   0x10000, 0x640000,
app1,     app,  ota_1,   0x650000,0x640000,
spiffs,   data, spiffs,  0xc90000,0x360000,
coredump, data, coredump,0xFF0000,0x10000,
```

Partição de Dados na Memória Flash do ESP32

data



arquivo.txt



dados.json

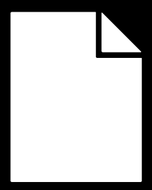
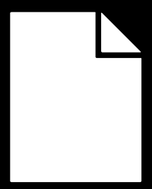
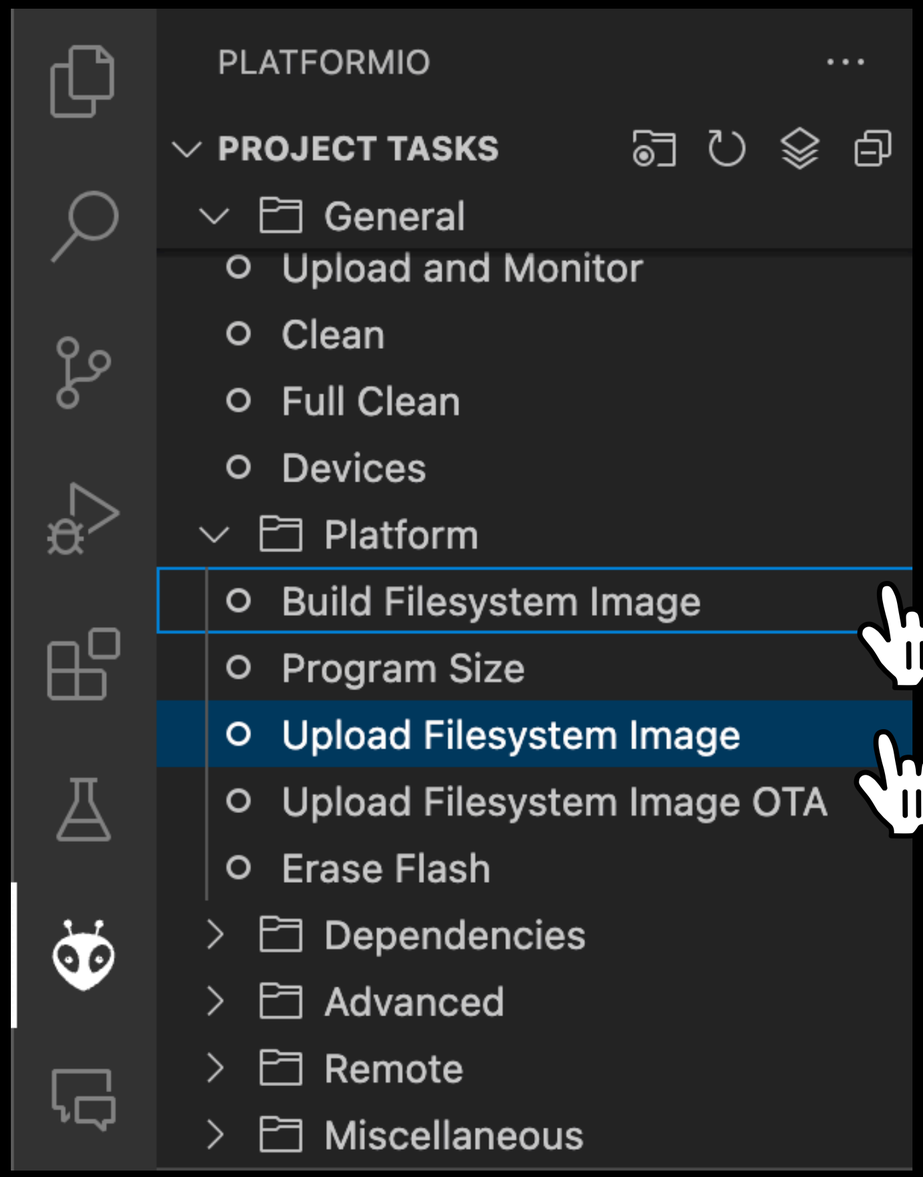


foto.jpg



index.html

...



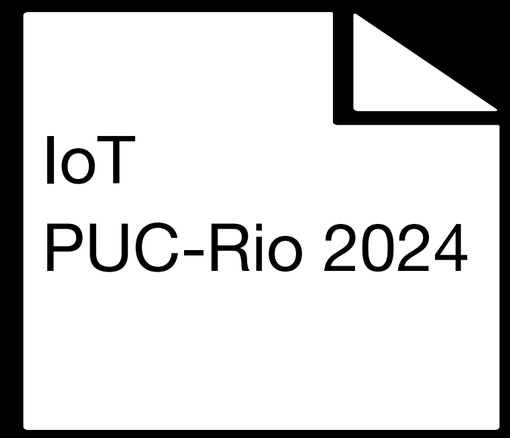
Transferência de Arquivos para o ESP-32 via PlatformIO no Visual Studio Code

```
#include <LittleFS.h>

void setup() {
  Serial.begin(115200);
  delay(500);

  if (!LittleFS.begin()) {
    Serial.println("Falha ao montar o sistema de arquivos!");
    while (true) {}; // trava programa aqui em caso de erro
  }
}
```

```
                    // em alguma parte do código...
File arquivo = LittleFS.open("/arquivo.txt", "r"); // "read"
if (!arquivo) {
  Serial.println("Erro ao abrir o arquivo!");
  while (true) {}; // trava programa aqui
}
String conteudo = arquivo.readString();
arquivo.close();

Serial.println(conteudo);
```

Leitura de Arquivos

```
#include <LittleFS.h>

void setup() {
  Serial.begin(115200);
  delay(500);

  if (!LittleFS.begin()) {
    Serial.println("Falha ao montar o sistema de arquivos!");
    while (true) {}; // trava programa aqui em caso de erro
  }
}

// em alguma parte do código...

File arquivo = LittleFS.open("/arquivo2.txt", "w"); // "write"
arquivo.println("IoT");
arquivo.print("PUC-Rio ");
arquivo.println(2020 + 4);
arquivo.close();
```



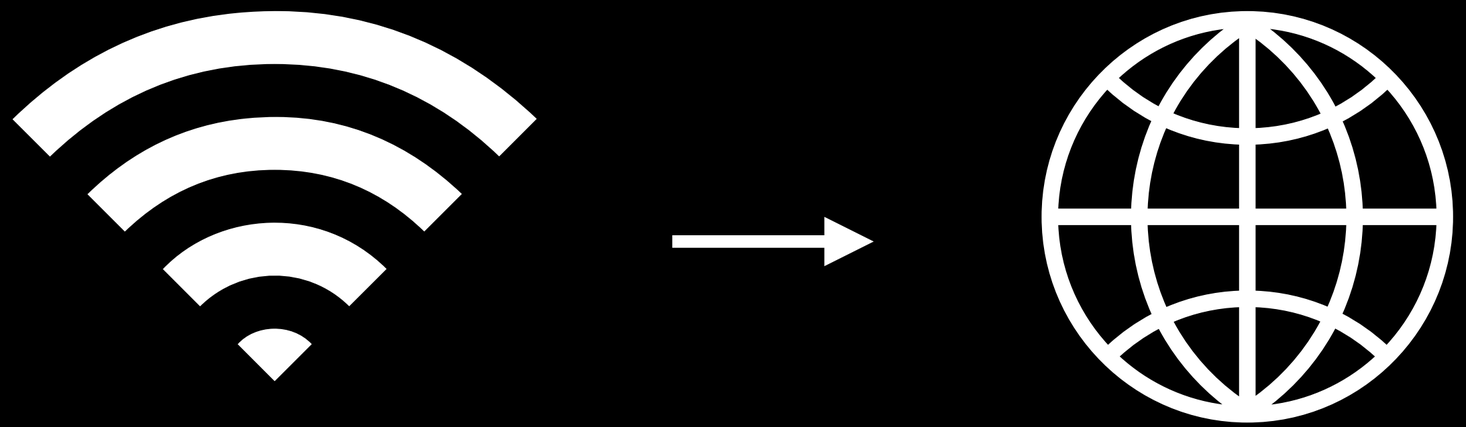
Escrita de Arquivos

```
// em alguma parte do código...

File arquivo = LittleFS.open("/dados.json", "r"); // "read"
if (!arquivo) {
  Serial.println("Erro ao abrir o arquivo JSON!");
  while (true) {}; // trava programa aqui em caso de erro
}
JsonDocument dados;
deserializeJson(dados, arquivo);
arquivo.close();

// em alguma parte do código...

File arquivo = LittleFS.open("/dados2.json", "w"); // "write"
JsonDocument dados;
dados["nome"] = "Jan K. S.";
dados["status"] = "cansado";
serializeJson(dados, arquivo);
arquivo.close();
```

Leitura e Escrita de Arquivos JSON

Internet

Servidor Web no ESP32



1. WiFi

2. Servidor

2 Etapas para Criar um Servidor

Vamos sempre chamar essa função na setup e no loop!

```
#include <WiFi.h>

void reconectarWifi() {
  if (WiFi.status() != WL_CONNECTED) {
    WiFi.begin("NOME DA REDE WIFI", "SENHA DA REDE WIFI");

    Serial.print("Conectando ao WiFi...");
    while (WiFi.status() != WL_CONNECTED) {
      Serial.print(".");
      delay(1000);
    }

    Serial.print("conectado!\nEndereço IP: ");
    Serial.println(WiFi.localIP());
  }
}
```

Conectado ao WiFi..... conectado!
Endereço IP: 192.168.0.XXX

```
#include <WiFi.h>
#include <WebServer.h>

WebServer servidor(80); // porta 80 (padrão do HTTP)

void paginaInicial () {
  servidor.send(200, "text/html", "Bem-vindo!");
}

void setup () {
  reconectarWifi(); // função do slide anterior
  servidor.on("/inicio", HTTP_GET, paginaInicial);
  servidor.begin();
}

void loop () {
  reconectarWifi();
  servidor.handleClient();
}
```

```
Conectado ao WiFi..... conectado!
Endereço IP: 192.168.0.XXX
```

192.168.0.XXX/**inicio**

Bem-vindo!

Exemplo de Rota Simples

```
#include <WiFi.h>
#include <WebServer.h>
#include <uri/UriBraces.h>

WebServer servidor(80); // porta 80 (padrão do HTTP)

void paginaComParametros () {
  int x = servidor.pathArg(0).toInt();
  int y = servidor.pathArg(1).toInt();
  servidor.send(200, "text/html", String(x+y));
}

void setup () {
  reconectarWifi(); // função do slide anterior

  servidor.on(UriBraces("/soma/{}/{}"), HTTP_GET, paginaComParametros);
  servidor.begin();
}

void loop () {
  reconectarWifi();
  servidor.handleClient();
}
```

Conectado ao WiFi..... conectado!
Endereço IP: 192.168.0.XXX

192.168.0.XXX/soma/3/5

8

Exemplo de Rota com Parâmetros

```
#include <LittleFS.h>

void paginaComArquivoHTML () {
  File arquivo = LittleFS.open("/pagina.html", "r");
  if (!arquivo) {
    servidor.send(500, "text/html", "Erro ao abrir HTML");
    return;
  }
  String html = arquivo.readString();
  arquivo.close();

  html.replace("{{nome}}", "Jan");
  html.replace("{{pontos}}", 10000);

  servidor.send(200, "text/html", html);
}
```

pagina.html

```
<h1>{{nome}}</h1>
<p>{{pontos}} XP</p>
```

192.168.0.XXX/pontuacao

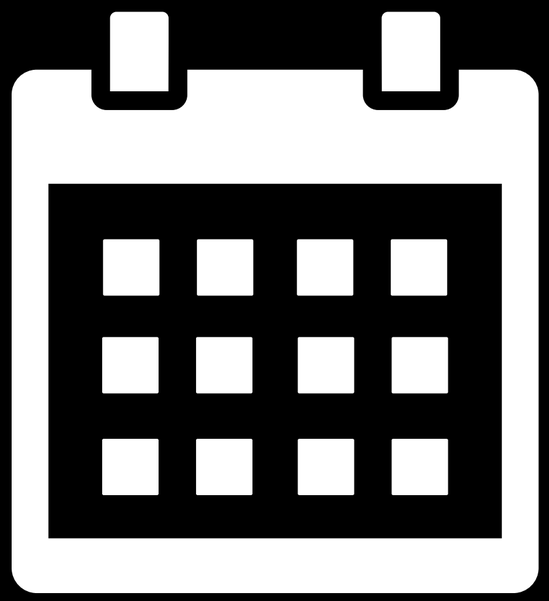
Jan K. S.

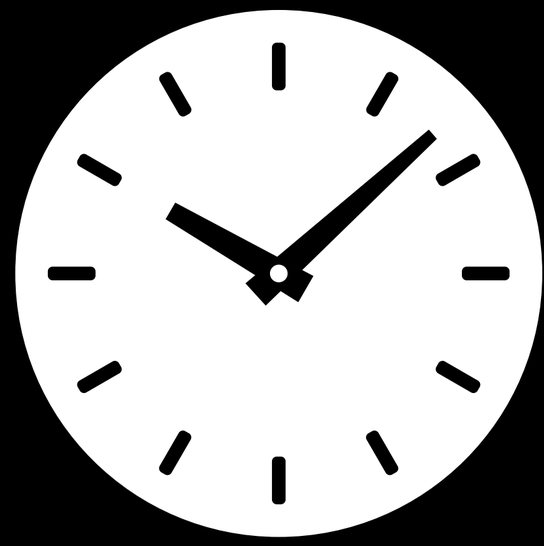
10000 XP

Exemplo de Rota com Arquivo HTML

```
#include <LittleFS.h>

void paginaComFormulario () {
  // mesma lógica de arquivo HTML anterior
}

void tratarDadosSubmetidos () {
  String usuario = servidor.arg("usuario");
  String senha = servidor.arg("senha");
  // faz alguma coisa com esses dados

  // redireciona para outra página
  servidor.sendHeader("Location", "/");
  servidor.send(303);
}

void setup () {
  // ...

  servidor.on("/formulario", HTTP_GET, paginaComFormulario);
  servidor.on("/formulario", HTTP_POST, tratarDadosSubmetidos);
  servidor.begin();
}
```

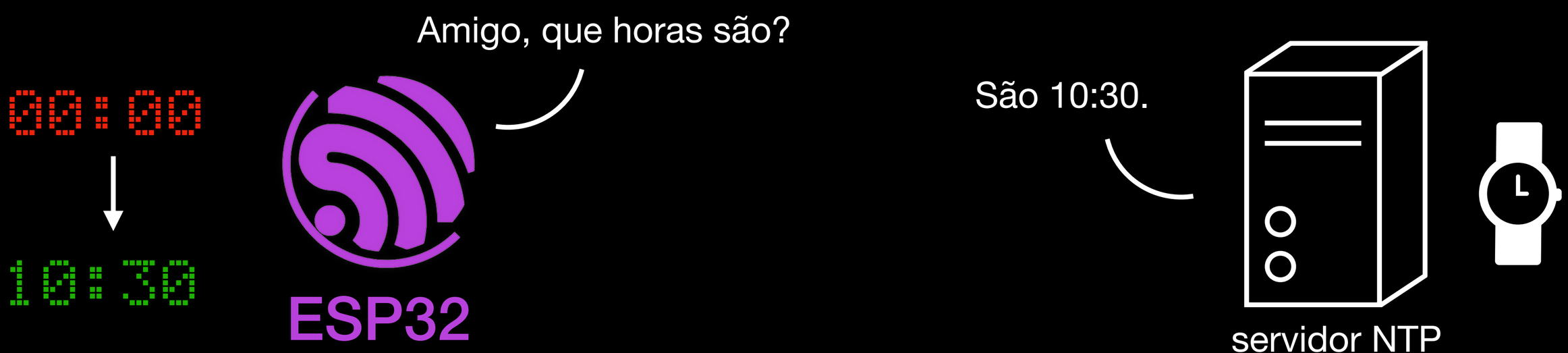
Submissão de Formulário



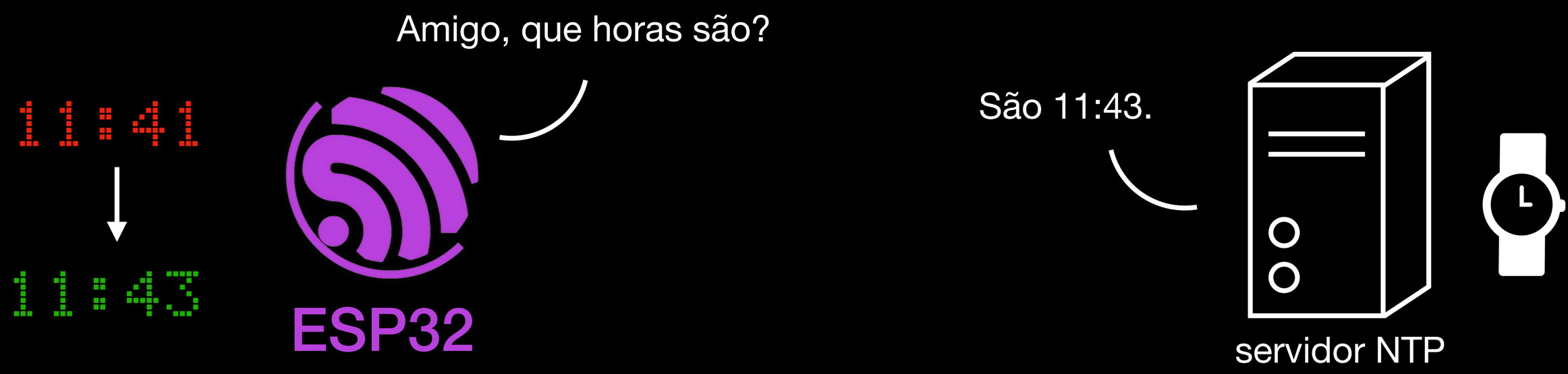


Data / Hora no ESP32

ao começar o programa...



um tempo depois...



Protocolo NTP para Sincronizar Data e Hora

```
#include <WiFi.h>
#include <time.h>

void setup() {
  Serial.begin(115200); delay(500);

  reconectarWifi();

  // fuso horário brasileiro, servidor, servidor alternativo
  configTzTime("BRT+3", "a.ntp.br", "pool.ntp.org");
}
```

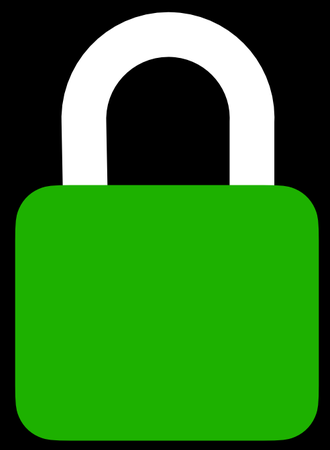
```
// em alguma parte do código...

struct tm tempo;
getLocalTime(&tempo);

Serial.println(&tempo, "%d/%m/%Y %H:%M:%S");

char buffer[100];
strftime(buffer, sizeof(buffer), "%d/%m/%Y %H:%M:%S", &tempo);
String tempoString = String(buffer);
```

```
23/08/2024 16:30:01
```

Obtenção da Data/Hora Formatada no ESP 32

```
int hora = tempo.tm_hour;
int minuto = tempo.tm_min;
int segundo = tempo.tm_sec;
int dia = tempo.tm_mday;
int mes = tempo.tm_mon + 1;
int ano = tempo.tm_year + 1900;
int diaDaSemana = tempo.tm_wday; // 0 = domingo, 1 = segunda...
```

Obtenção da Data/Hora Formatada no ESP 32



https://
(TLS)

Protocolo HTTPS (HTTP + TLS)

Pai, preciso falar contigo...

Acho que tem um Deus nórdico bisbilhotando a gente...

Diga lá, filha!

Confiança é para crianças e cães.

Comunicação entre Alice e Bob com Loki Bisbilhoteiro

chave simétrica
(tranca e destranca dados)

Me manda o seu cadeado!

Oi, pai!

Oi filha

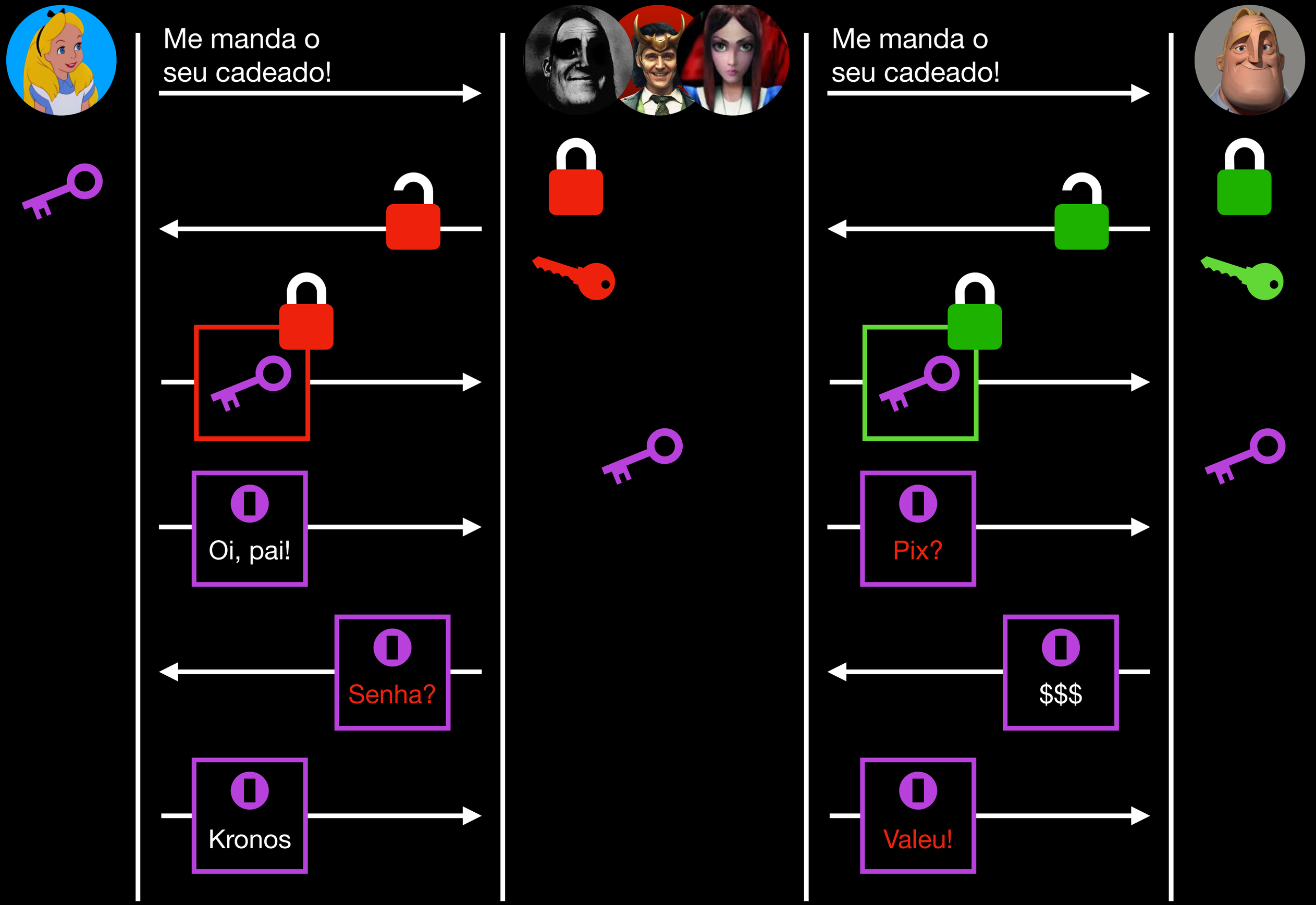
chave pública
(tranca dados)

chave privada
(destranca dados)

(cópia da chave da Alice)

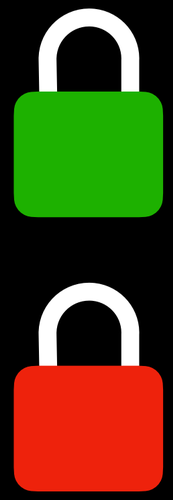
???

Criptografia com Chaves Assimétricas e Simétricas



Ataque Intermediário (Man in the Middle)

Como eu sei se o cadeado veio do Bob ou de outra pessoa?



Vou te mandar um certificado de autenticidade!

Problema de Confiança na Chave Pública

Não sei se essa é a assinatura mesmo do Bob ou não...

Não confio em político.

Em Deus eu confio e não temerei!

Autoridades de Certificação Raiz

Jesus
Alá
Zeus

Oi, Alice!

Só para certificar: sou eu mesmo mandando esse cadeado.

Rio de Janeiro, 23 de agosto de 2024
Bob

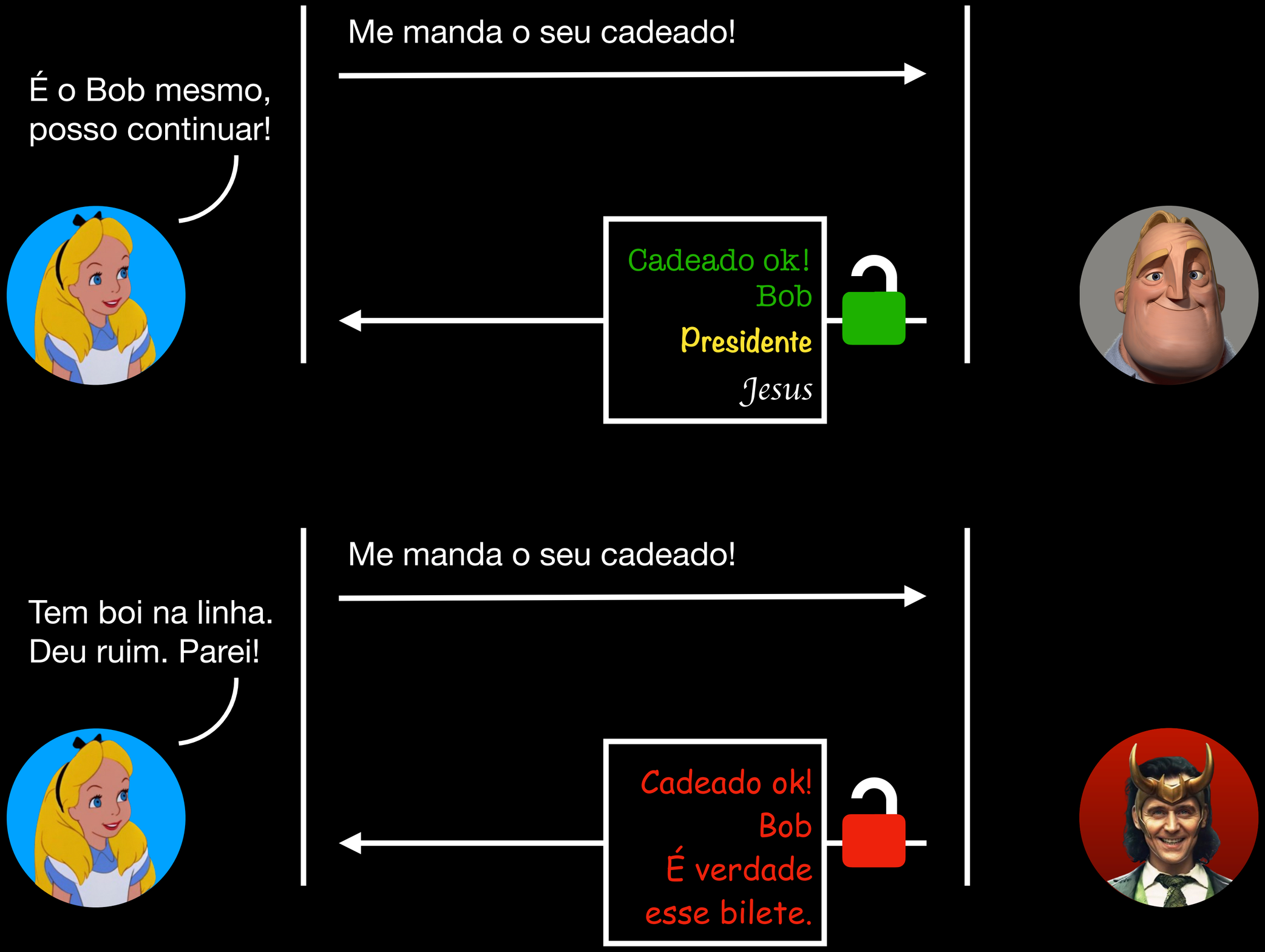
Certifico que a assinatura "Bob"
é válida e confiável.

Brasília, 1º de janeiro de 2023
Presidente

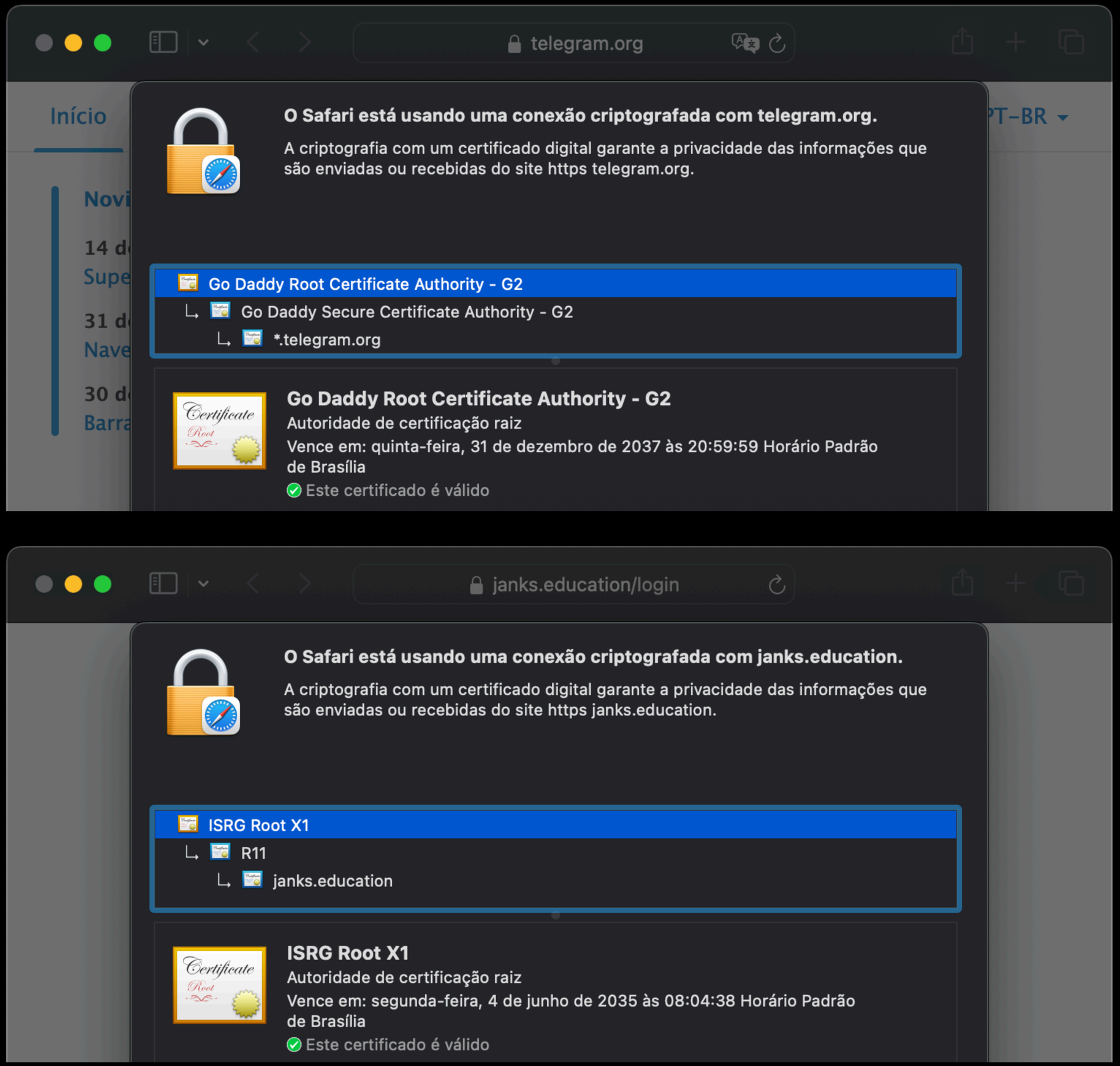
Certifico que a assinatura **"Presidente"**
é válida e confiável.

Céu, 1º de janeiro de 2002
Jesus

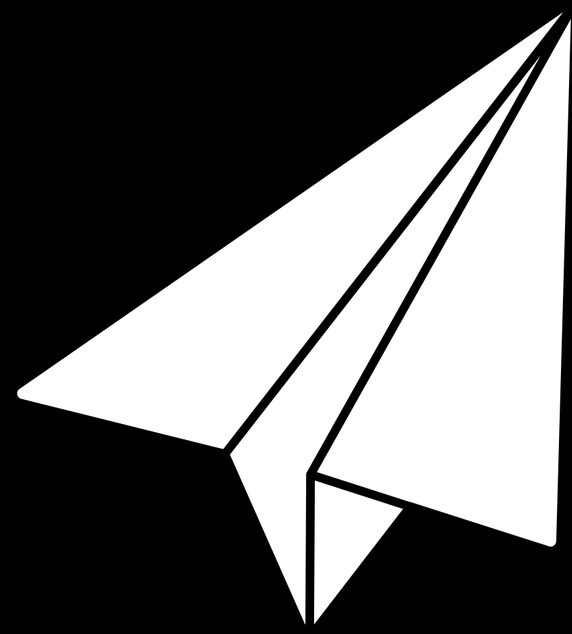
Cadeia de Certificados



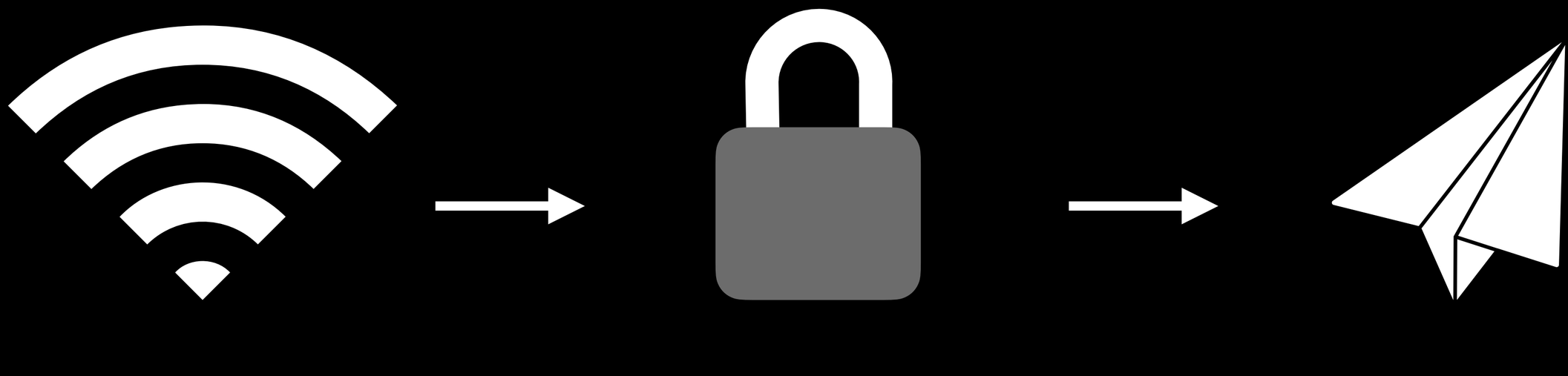
Envio da Chave Pública com Certificado



Exemplos de Certificados HTTPS nos Navegadores



Requisição Web no ESP32

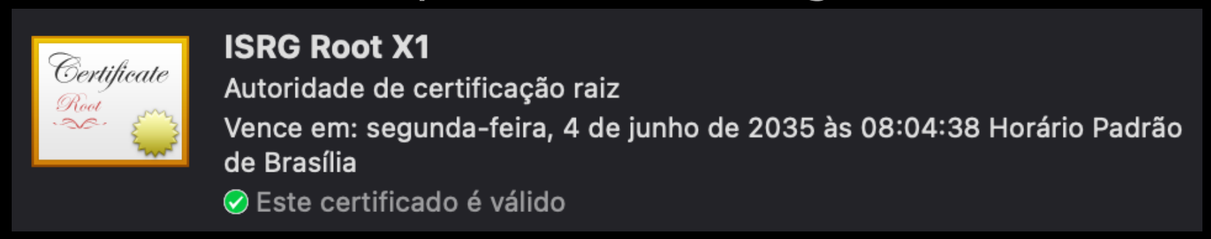


1. WiFi
2. Certificado
3. Requisição

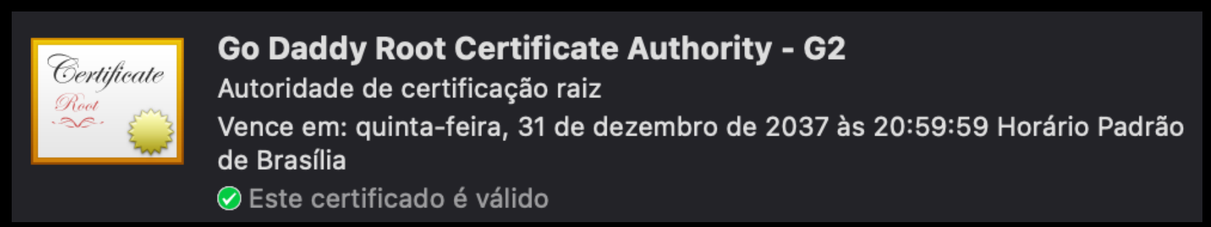
3 Etapas para uma Requisição Web

certificados.h

```
const char certificado1[] = R"=EOF=(
-----BEGIN CERTIFICATE-----
MIIFazCCA1OgAwIBAgIRAIIQz7DSQONZRGPgu2OCiwAwDQYJKoZIhvcNAQELBQAw
TzELMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2Vh
cmNoIEdyb3VwMRUwEwYDVQQDEwxJU1JHIFJvb3QgWDEwHhcNMTUwNjA0MTEwNDM4
WhcNMzUwNjA0MTEwNDM4WjBPMQswCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJu
...
-----END CERTIFICATE-----
)=EOF=";
```




```
const char certificado2[] = R"=EOF=(
-----BEGIN CERTIFICATE-----
MIIDxTCCAq2gAwIBAgIBADANBgkqhkiG9w0BAQsFADCBgzELMAkGA1UEBhMCVVMx
EDAOBgNVBAgTB0FyaXpvbmExEzARBgNVBAcTClNjb3R0c2RhbGUxGjAYBgNVBAoT
EUdvRGFkZHkuY29tLCBJbmMuMTEwLwYDVQQDEyhHbyBEYWRkeSBSb290IENlcnRp
...
4uJEvlz36hz1
-----END CERTIFICATE-----
)=EOF=";
```



Etapa 2: Arquivo com Certificados de Autoridade Raiz

```
#include <WiFiClientSecure.h>
#include "certificados.h"

WiFiClientSecure conexaoSegura;

void setup() {
  reconectarWifi();

  conexaoSegura.setCACert(certificado1);
  conexaoSegura.setCACert(certificado2);
}
```

Etapa 2: Registro dos Certificados de Autoridade Raiz

```
String chaveTelegram = "COLOQUE A SUA CHAVE DO BOTFATHER AQUI!";
String idDoChat = "COLOQUE O ID DA SUA CONVERSA AQUI!";
String enderecoBase = "https://api.telegram.org/bot" + chaveTelegram;

JsonDocument dados;
dados["chat_id"] = idDoChat;
dados["text"] = mensagem;
String dadosString;
serializeJson(dados, dadosString);

String enderecoMensagemTexto = enderecoBase + "/sendMessage";
HTTPClient requisicao;
requisicao.begin(conexaoSegura, enderecoMensagemTexto);
requisicao.addHeader("Content-Type", "application/json");
int codigoDoResultado = requisicao.POST(dadosString);
String resposta = requisicao.getString();
Serial.println(resposta);

if (codigoDoResultado != 200) {
  Serial.println("Erro ao enviar mensagem!");
}
```

Etapa 3: Exemplo de Requisição para o Telegram