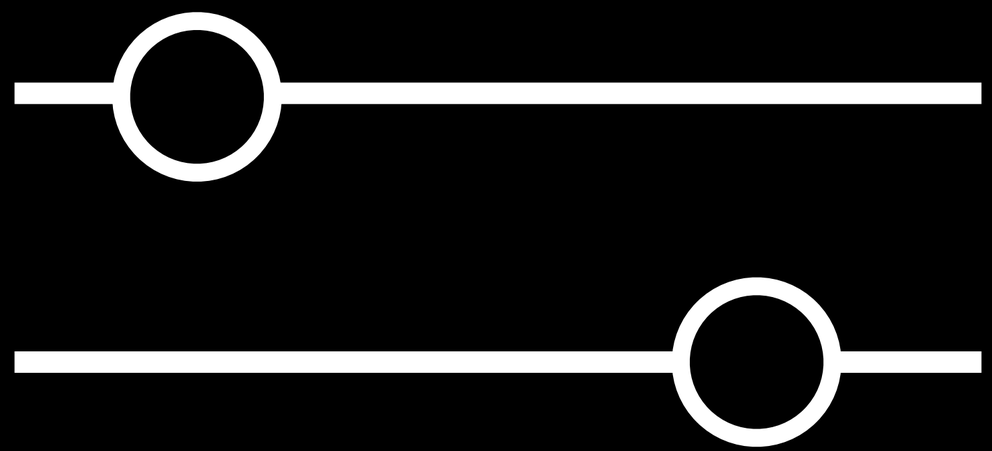
Projeto 08

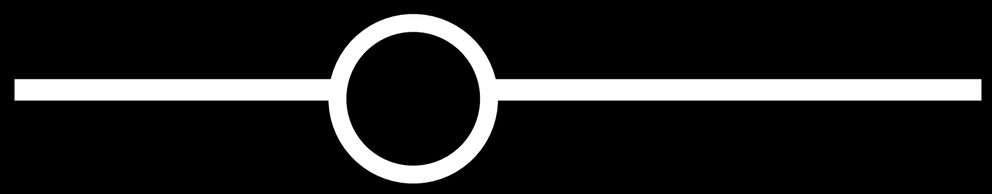
Coisas da Casa – Teoria

Jan K. S. – janks@puc-rio.br

ENG4051 – Projeto Internet das Coisas

Coisas





Preferences

memória Flash

nvs (dados não voláteis)

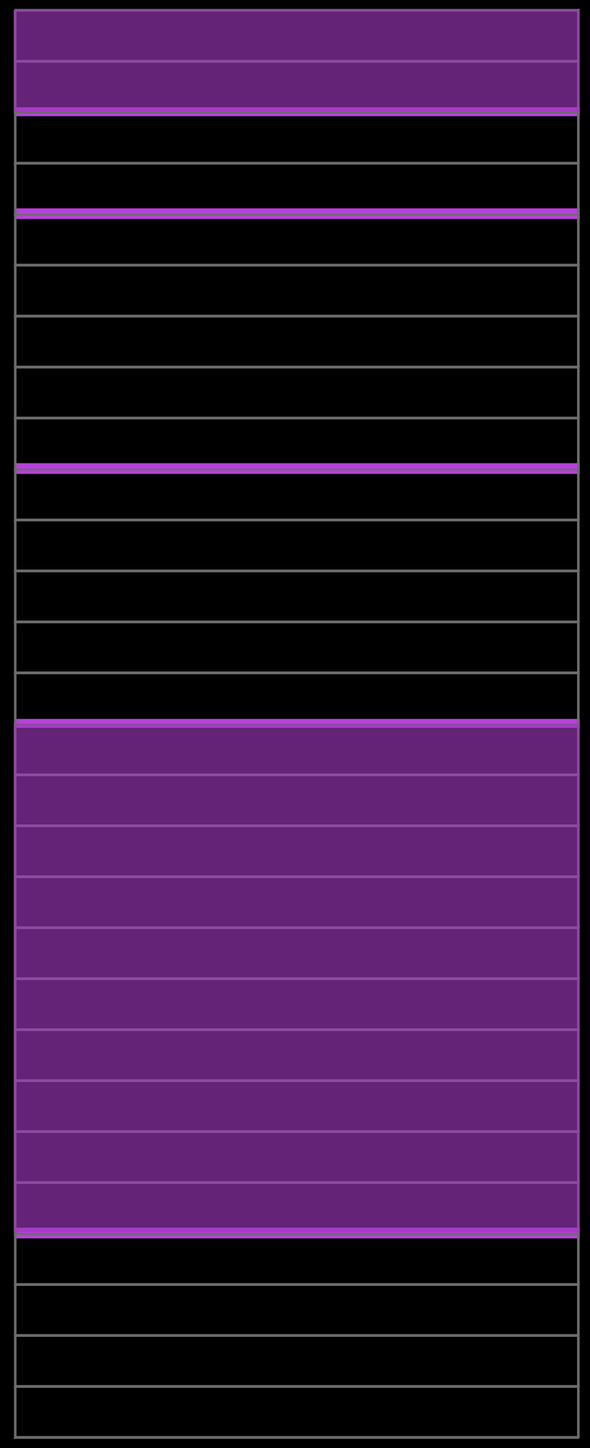
ota (atualização remota)

app0 (aplicação)

app1 (outra aplicação)

spiffs (arquivos)

coredump (debug)



Preferences

chave1: valor1

chave2: valor2

....

LittleFS

pagina1.html

pagina2.html

dados.json

imagem.jpeg

...

Partição NVS (Dados Não Voláteis) vs Partição Spiffs (Arquivos)

Preferences

"ajustesUsuario"

```
"nome": "Jan K. S."
"altura": 182
"professor": true
```

"ajustesRede"

```
"nome": "WiFi PUC"
"usuario": "Jan"
"senha": "1234abcd"
```

"minhaBiblioteca"

```
"escala": 462.4
"chaveAPI": "Av93Mw2..."
```

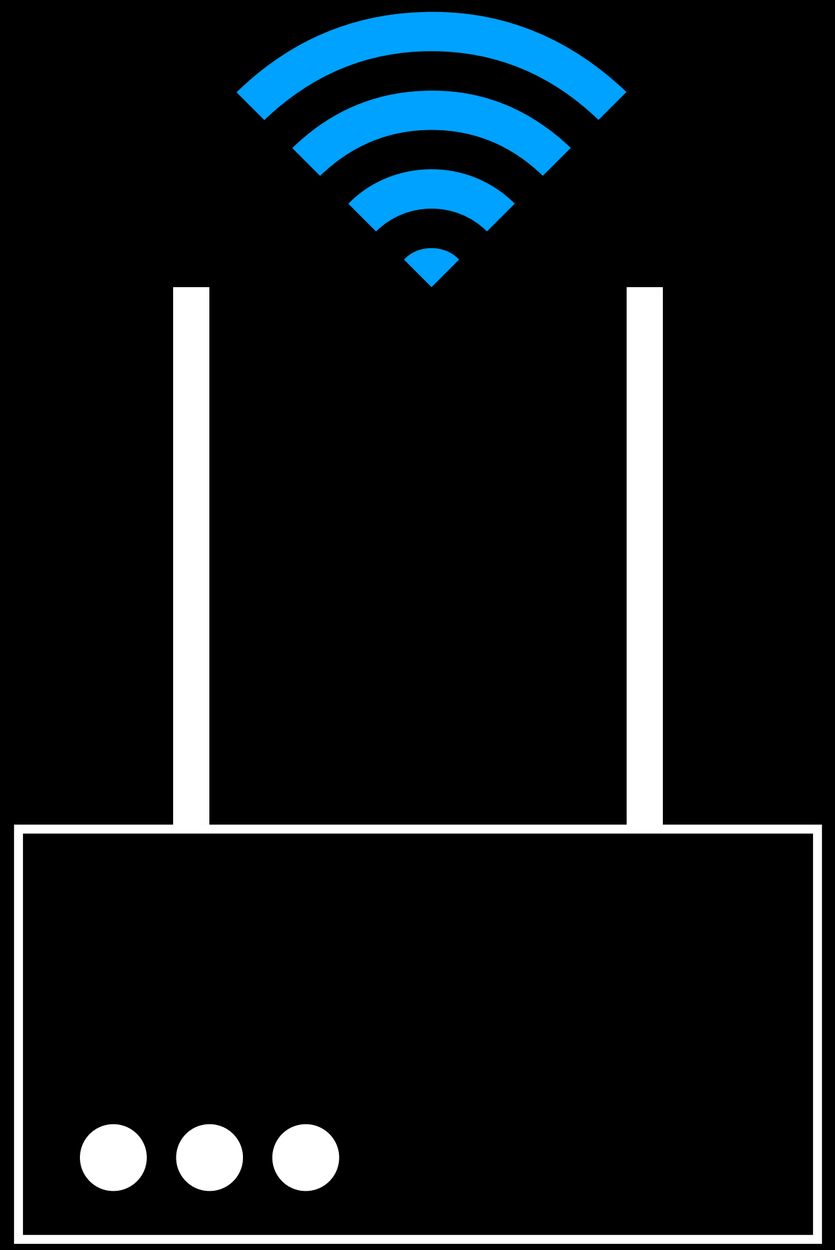
O namespace e as chaves têm um limite de 15 caracteres!

Namespaces com Chave+Valor

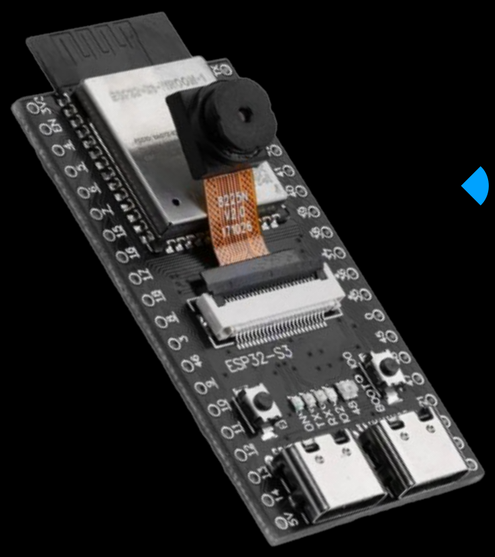
```
#include <Preferences.h>

Preferences preferencias;

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

  preferencias.begin("ajustesUsuario"); // nome único seu
}
        // em alguma função do código, para salvar dados...
preferencias.putString("nome", "Jan K. S.");
preferencias.putInt("contagem", 37);
preferencias.putFloat("preco", 10.99);
preferencias.putBool("somAtivado", true);

    // em alguma outra função do código, para carregar dados...
   // (o 2º parâmetro é o valor padrão se não tiver nada salvo)
String nome = preferencias.getString("nome", "");
int contagem = preferencias.getInt("contagem", 0);
float preco = preferencias.getFloat("preco", 0);
bool somAtivado = preferencias.getBool("somAtivado", false);
```

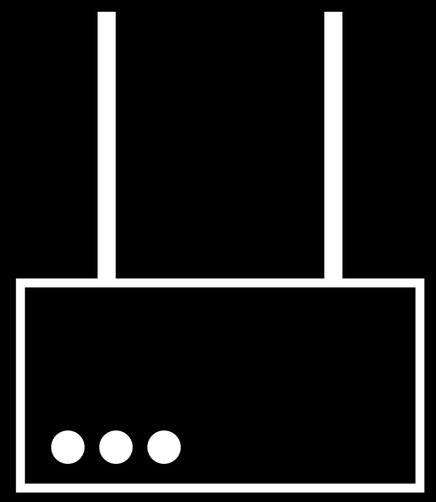
Uso da Biblioteca Preferences

Internet



WiFi Access Point (AP)



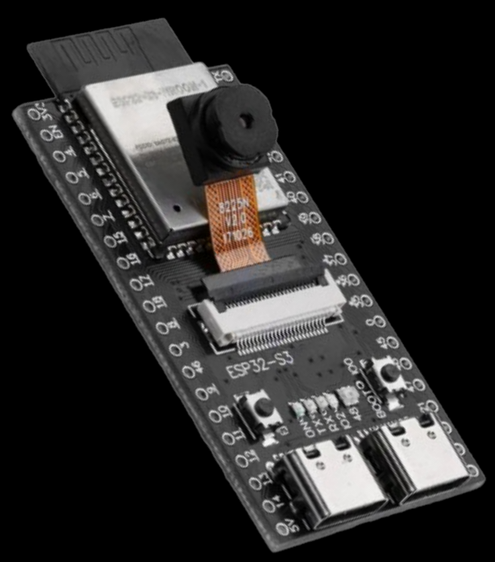


X

WiFi da Casa

192.168.0.XXX/pagina

Olá!



WiFi do ESP32

192.168.4.1/pagina

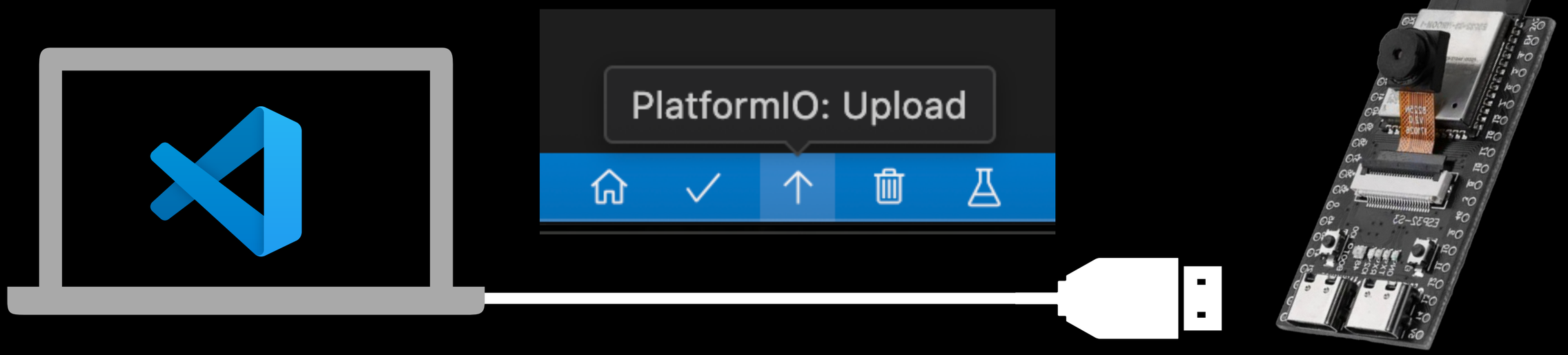
Olá!

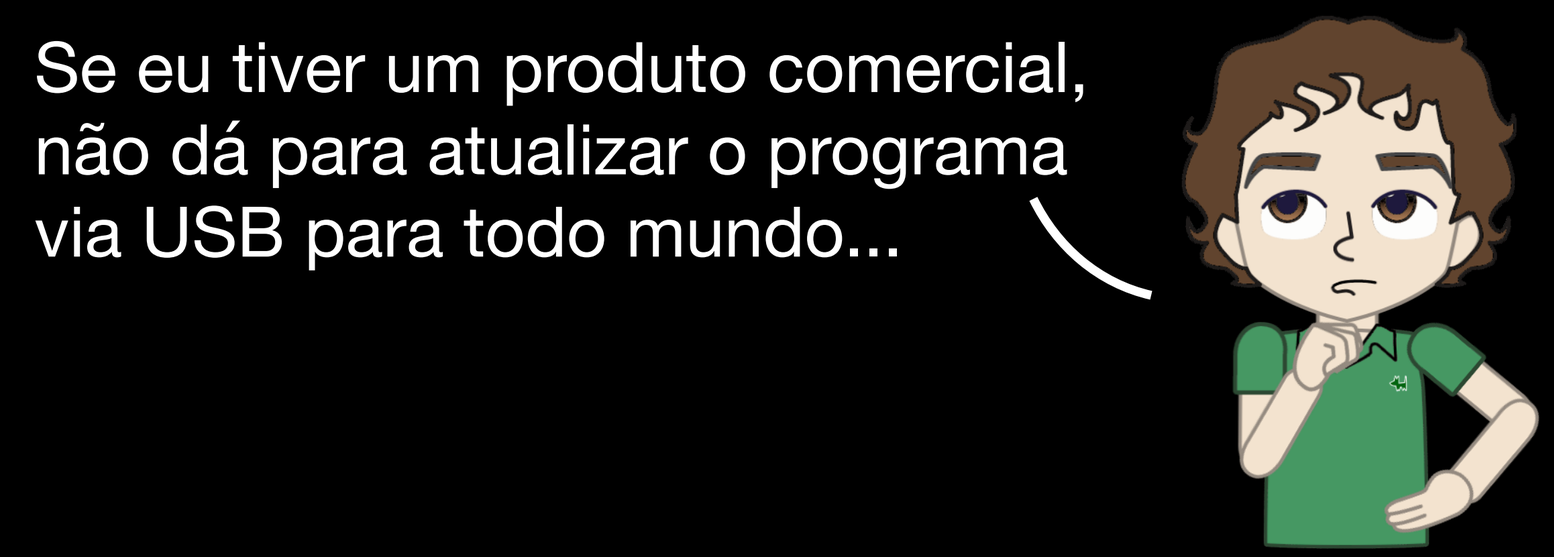
Servidor ESP com WiFi vs WiFi AP

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

WebServer servidor(80);

void pagina1() {
  servidor.send(200, "text/html", "Bem-vindo!");
}

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

   WiFi.softAP("SEU NOME DE REDE", "SUA SENHA DE AO MENOS 8 CARACTERES");
   Serial.println("WiFi Access Point iniciado!");
   Serial.print("IP: " + WiFi.softAPIP().toString());

   servidor.on("/pagina1", HTTP_GET, pagina3);
   servidor.begin();
}

void loop() {
   servidor.handleClient();
   int numeroDeConexoes = WiFi.softAPgetStationNum();
}
```

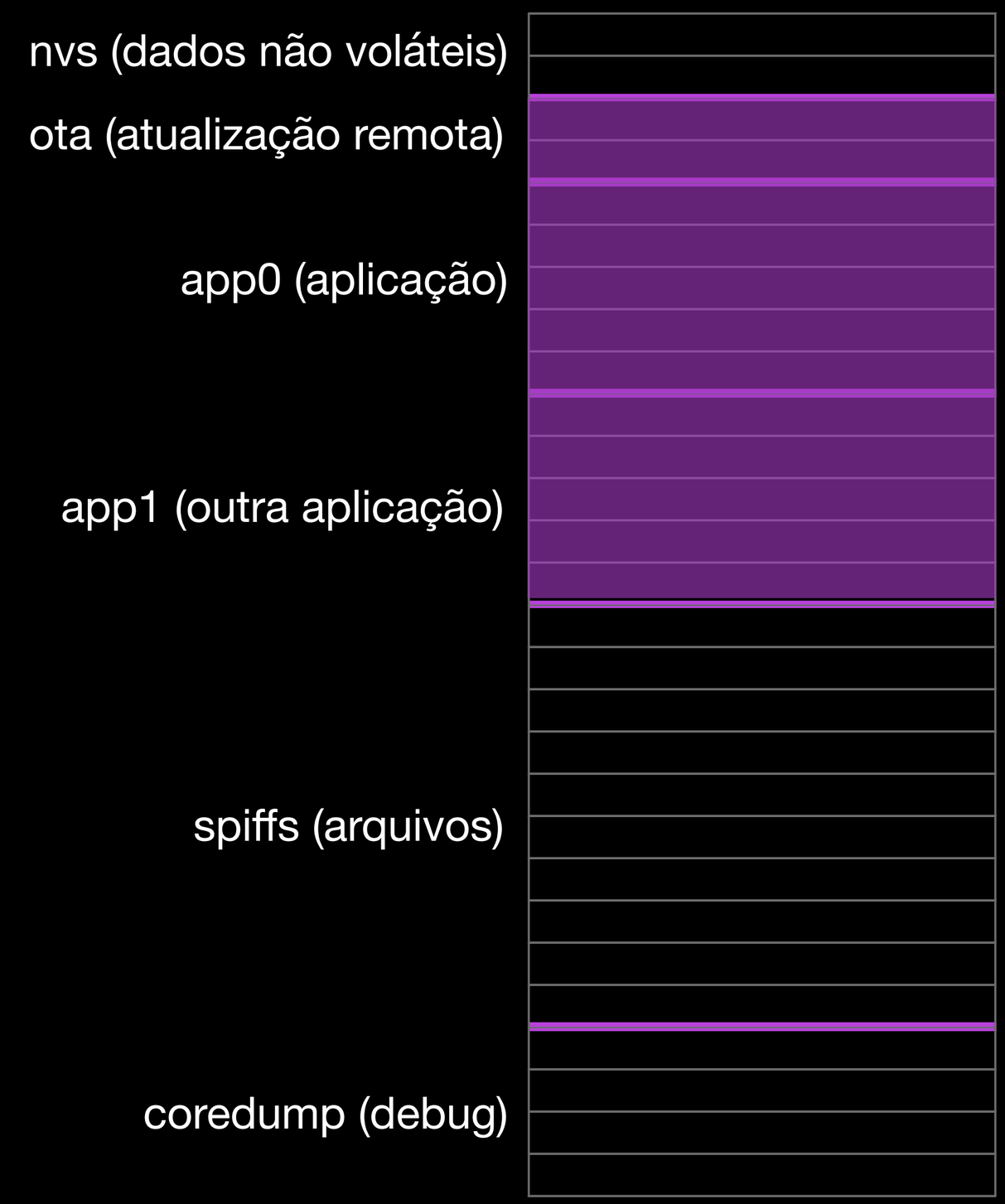
Exemplo de Servidor com WiFi AP

Atualizações Over The Air (OTA)





Atualização do Código do ESP32 via USB



ota escolhe partição app0
para rodar aplicação

↓

aplicação baixa novo binário
na partição app1

↓

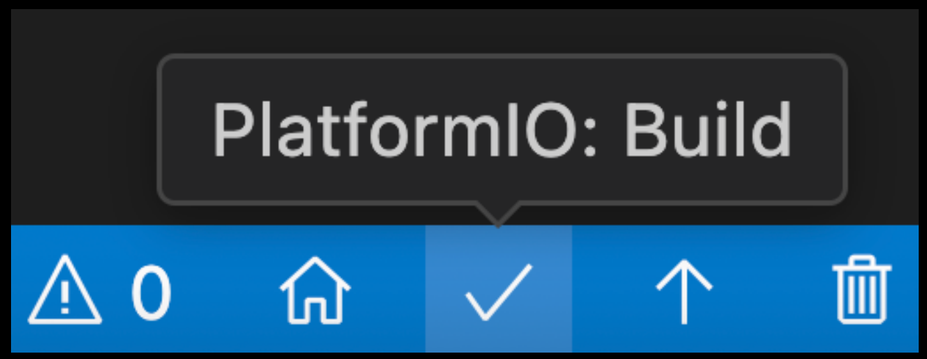
binário é verificado
ESP32 reinicia

↓

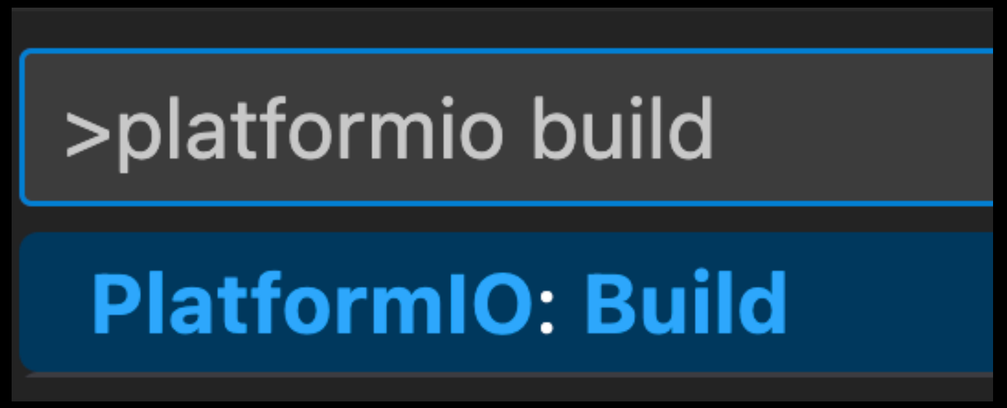
ota escolhe partição app1
para rodar aplicação

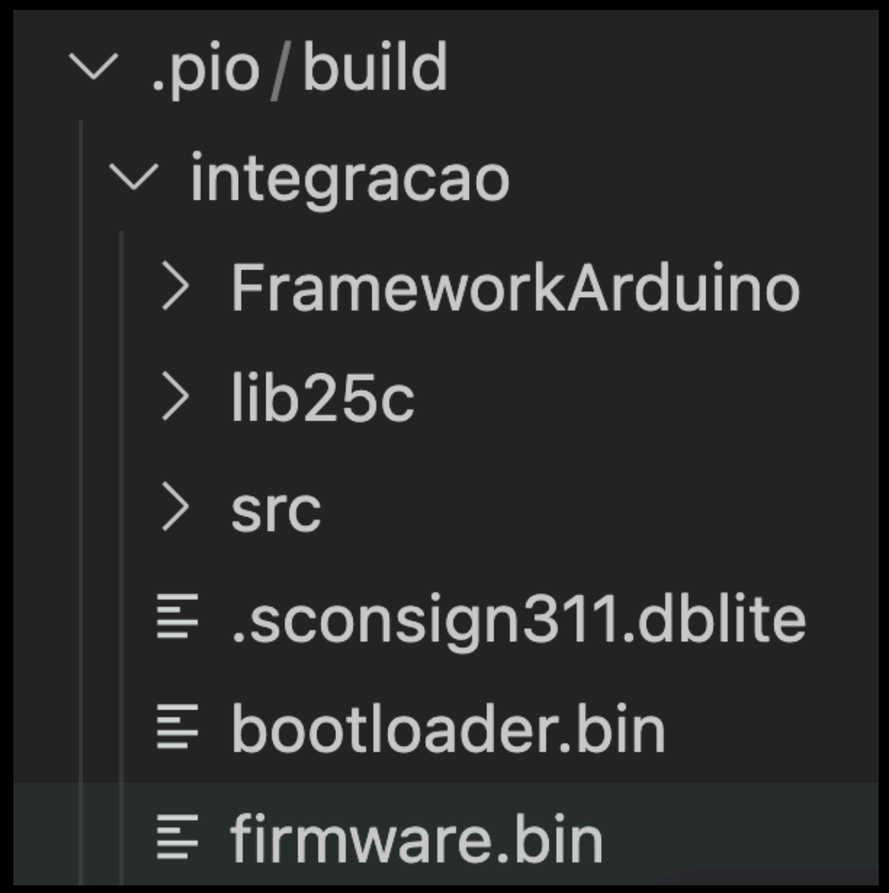
Partições para Atualização

Ctrl+Shift+P



ou





Compilação do Programa sem Upload

Verificação de Novas Atualizações

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <WiFiClientSecure.h>
#include "certificados.h"
#include <HTTPUpdate.h>

String enderecoBase = "https://MEU-SERVIDOR.COM";
String enderecoVersao = enderecoBase + "/versao";
String enderecoFirmware = enderecoBase + "/firmware.bin";

String versãoAtual = "1.0.1";

WiFiClientSecure conexaoSegura;
```

```
void setup() {
  Serial.begin(115200); delay(500);
  reconectarWiFi();
  conexaoSegura.setCACert(certificado1);
  conexaoSegura.setCACert(certificado2);
}

void loop() {
  // chama função para verificar atualizações periodicamente
}
                        // código continua a seguir...
```

Verificação da Versão Atual

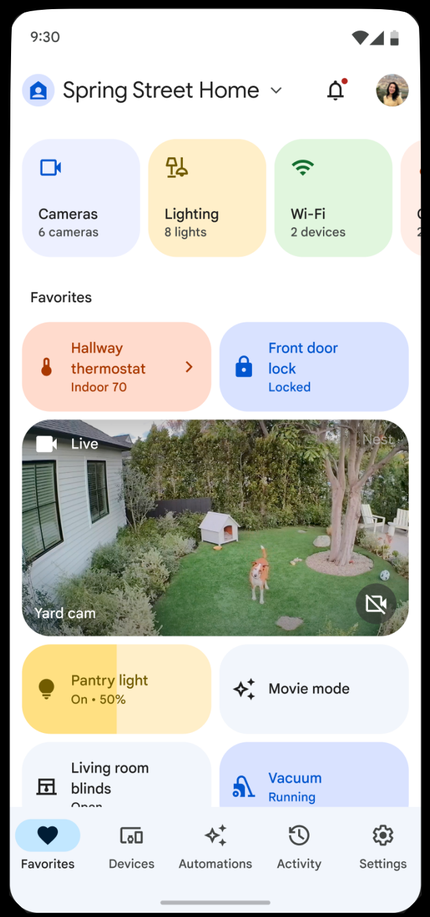
```
                    // continuação do código anterior...
void verificarAtualizacoes() {
  HTTPClient requisicao;
  requisicao.begin(conexaoSegura, enderecoVersao);
  int codigoHTTP = requisicao.GET();

  if (codigoHTTP == 200) {
    String versaoNova = requisicao.getString();
    requisicao.end();
    if (versaoAtual != versaoNova) {
      Serial.println("Atualizando firmware...");
      t_httpUpdate_return resultadoUpdate =
        httpUpdate.update(conexaoSegura, enderecoFirmware);
      if (resultadoUpdate != HTTP_UPDATE_OK) {
        Serial.println(httpUpdate.getLastErrorString());
      }
    }
  }
  else {
    Serial.println("Erro ao conectar com servidor de atualização!");
  }
}
```

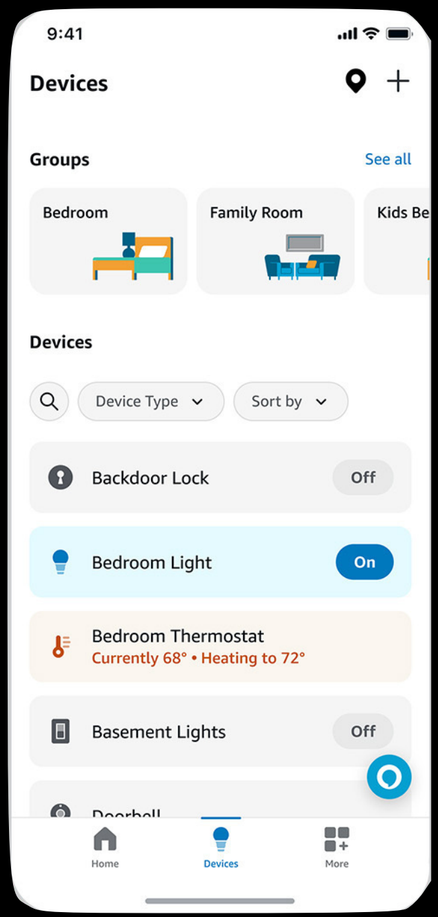
Verificação da Versão Atual

Matter

central
de controle
da casa

Dispositivos Inteligentes para a Casa





works with Google Home

Works with Apple Home

works with alexa

Ecossistemas de Dispositivos Inteligentes para a Casa

developers.home.google.com

Developer Center

Search /

English

Console

Develop — Android

Reference

Support

Filter

Get started

Pick a different platform

TRY THE SAMPLE APP

Build the Android sample app

Account authorization

Use the Android sample app

START BUILDING

1. Get the Android SDK
2. Set up OAuth
3. Initialize the home
4. Permissions API

INTEGRATE

Get started with the Google Home APIs

Welcome to the Home APIs documentation! Here you'll find everything you need to develop and deploy the Google Home APIs. This page provides an overview of the development cycle for your preferred platform, and quick links to technical documentation to guide you through the process. Let's get started!

1. Pick your platform

Android

iOS

developer.apple.com

Documentation

Language: Swift

All Technologies

HomeKit

Essentials

Enabling HomeKit in your app

HomeKit Entitlement

NSHomeKitUsageDescription

Home Manager

Configuring a home automation device

Testing your app with the HomeKit Ac...

HMHomeManager

Accessories

HMAccessorySetupManager

HMAccessorySetupResult

HMAccessorySetupRequest

Interacting with a home automation n...

HMAccessory

HMService

HMCharacteristic

Framework

HomeKit

Configure, control, and communicate with home automation accessories.

iOS 8.0+ | iPadOS 8.0+ | Mac Catalyst 14.0+ | tvOS 10.0+ | visionOS 1.0+ | watchOS 2.0+

Overview

HomeKit enables your app to coordinate and control home automation accessories from multiple vendors to present a coherent, user-focused interface.

Cada ecossistema tem uma API diferente...

developer.amazon.com

developer documentation

Sign In

Tools to Create and Manage Skills

Skill Developer Reference

Alexa Interface Reference

List of Alexa Interfaces

Message and Property Reference

Foundational APIs

Smart Home Skill APIs

AutomationManagement

BrightnessController

CameraStreamController

ChannelController

ColorController

ColorTemperatureController

Alexa > Documentation > Alexa Skills Kit > Alexa Interface Reference > Smart Home Skill APIs

Smart Home Skill APIs

Note: Sign in to the developer console to build or publish your skill.

You can use Smart Home APIs to build Alexa skills. These `Alexa` interfaces use the pre-built voice interaction model. For details about smart home skills, see Understand Smart Home Skills. For details about ACK-based devices, see What Is the Alexa Connect Kit?

For `Alexa` interfaces for video skills, see Video Skill APIs.

English (US)

© 2010 - 2025, Amazon.com, Inc. or its affiliates. All Rights Reserved.

Diferentes APIs para Integração com Ecossistemas

Google

SAMSUNG

zigbee

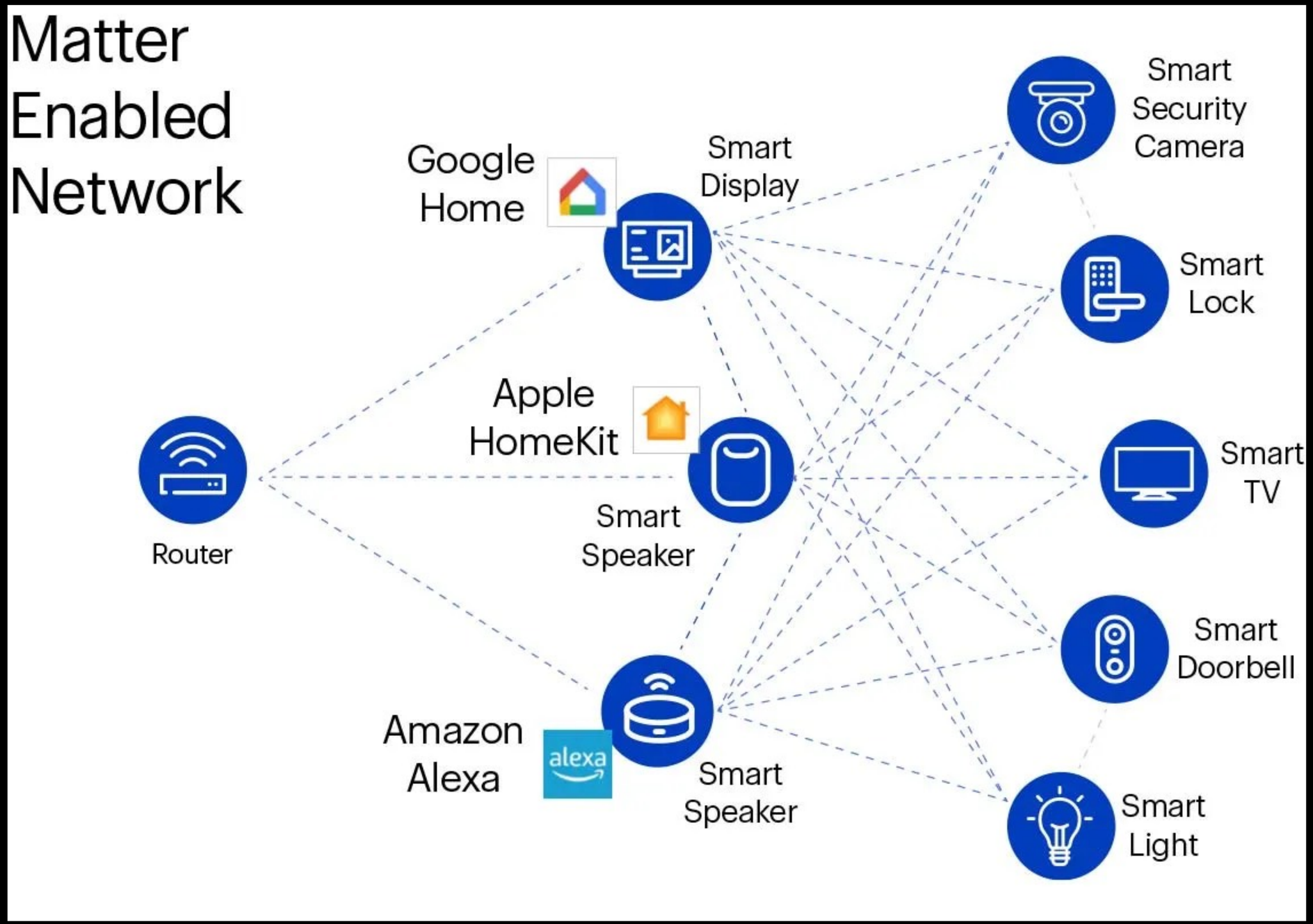
HUAWEI

amazon

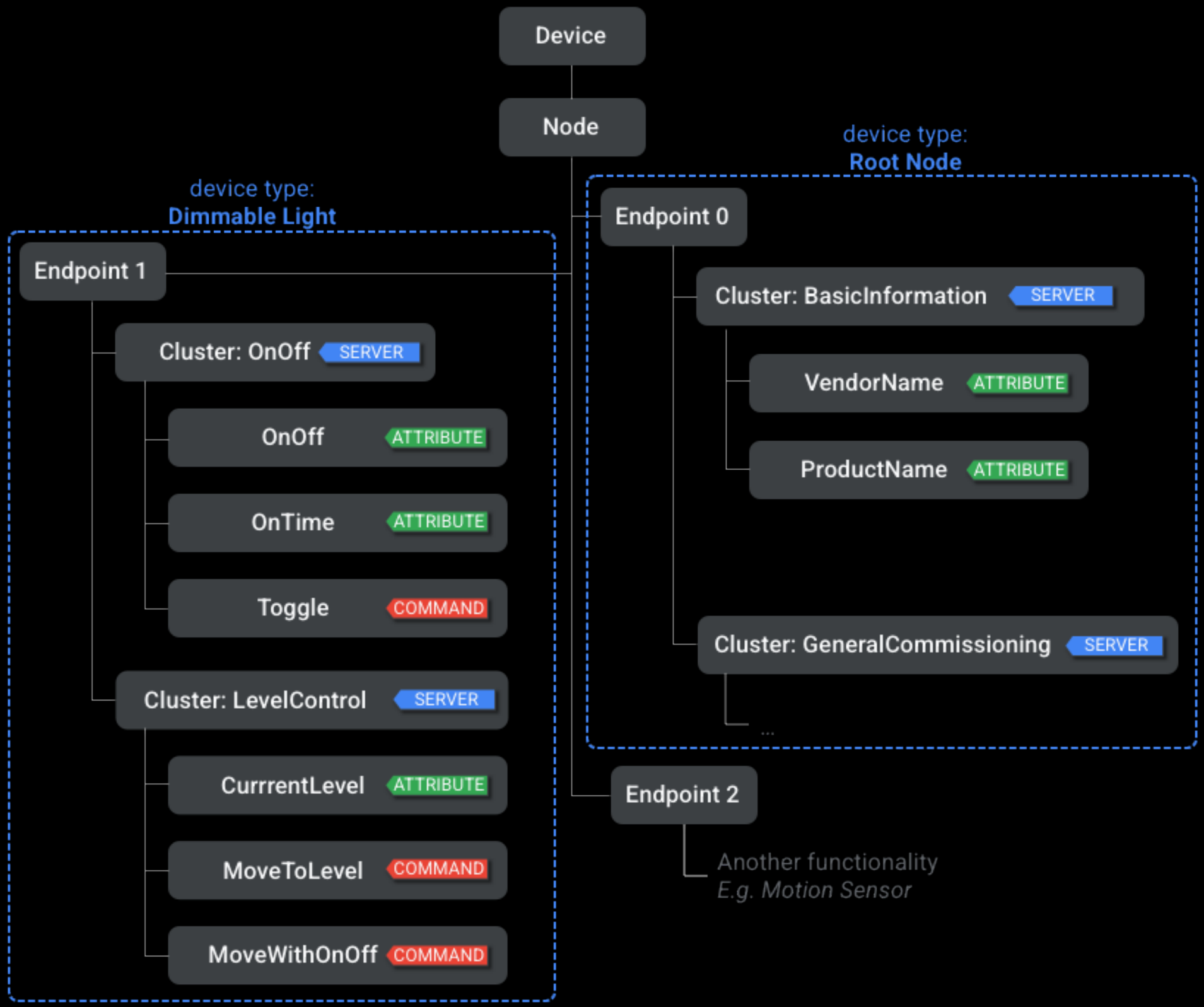
connectivity standards alliance

matter

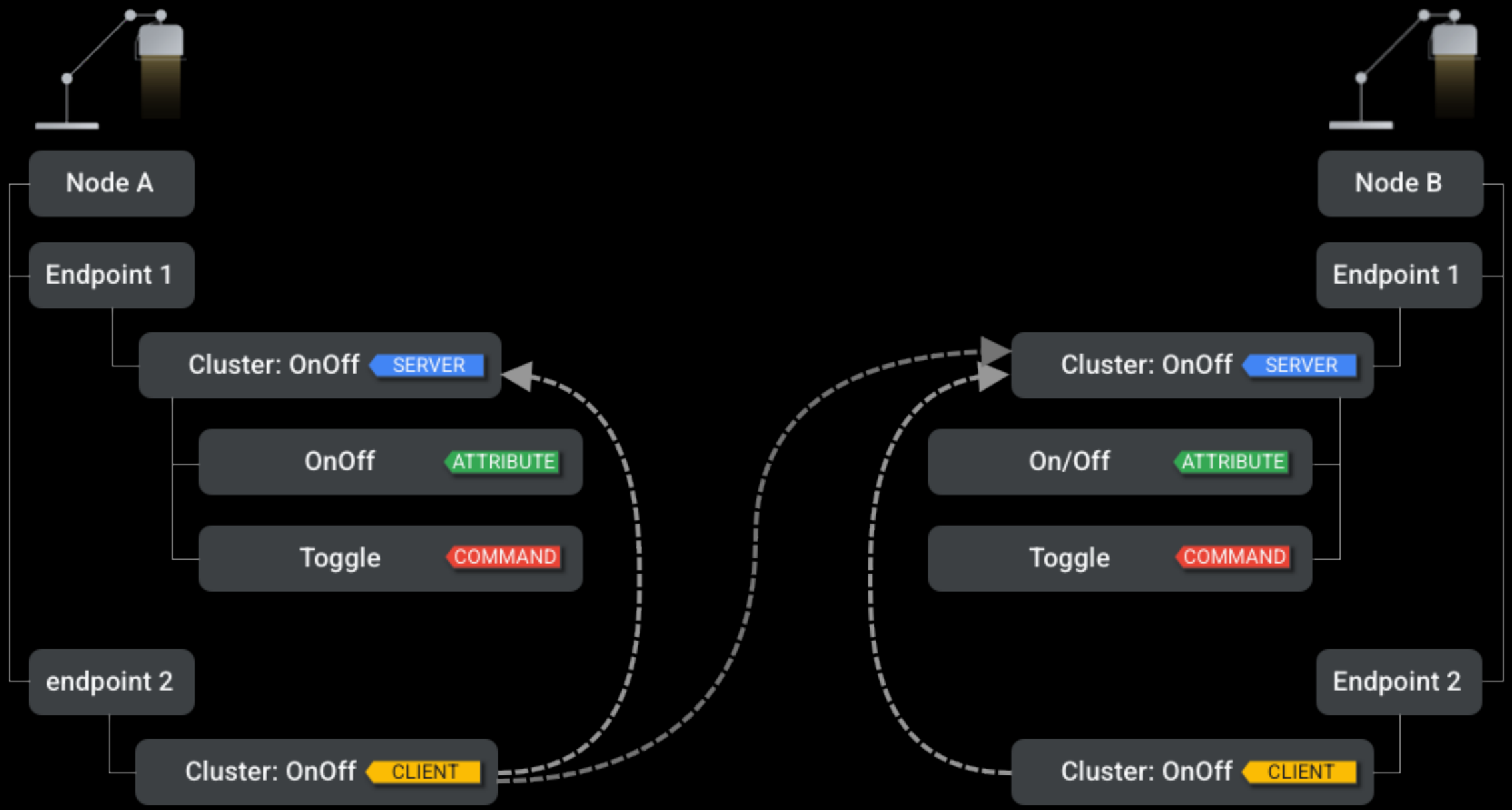
Consórcio para Protocolo Unificado


Matter
Enabled
Network
Google
Home
Smart
Display
Smart
Security
Camera
Smart
Lock
Apple
HomeKit
Smart
Speaker
Smart
TV
Router
Smart
Doorbell
Amazon
Alexa
alexa
Smart
Speaker
Smart
Light

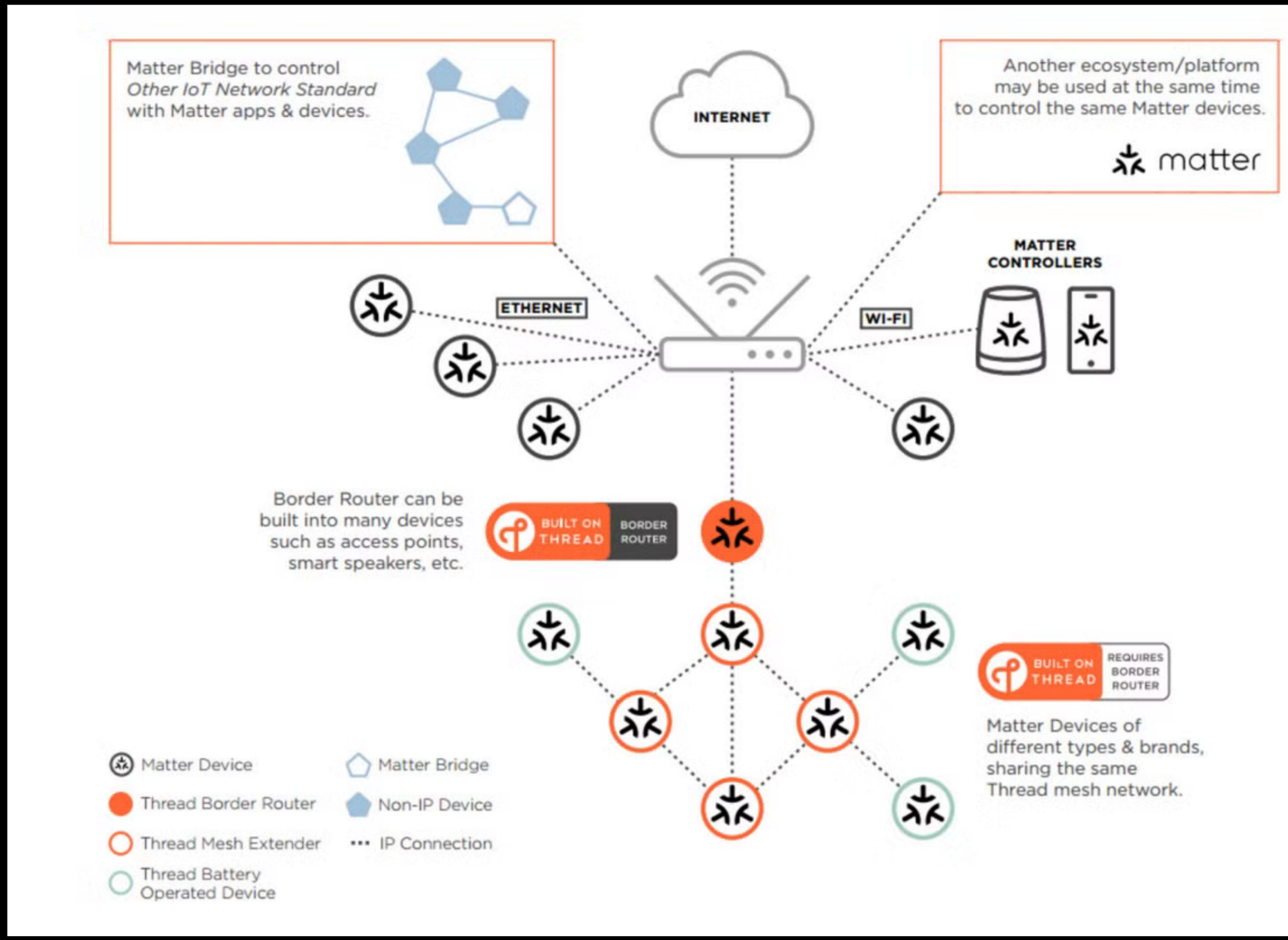
Rede Matter



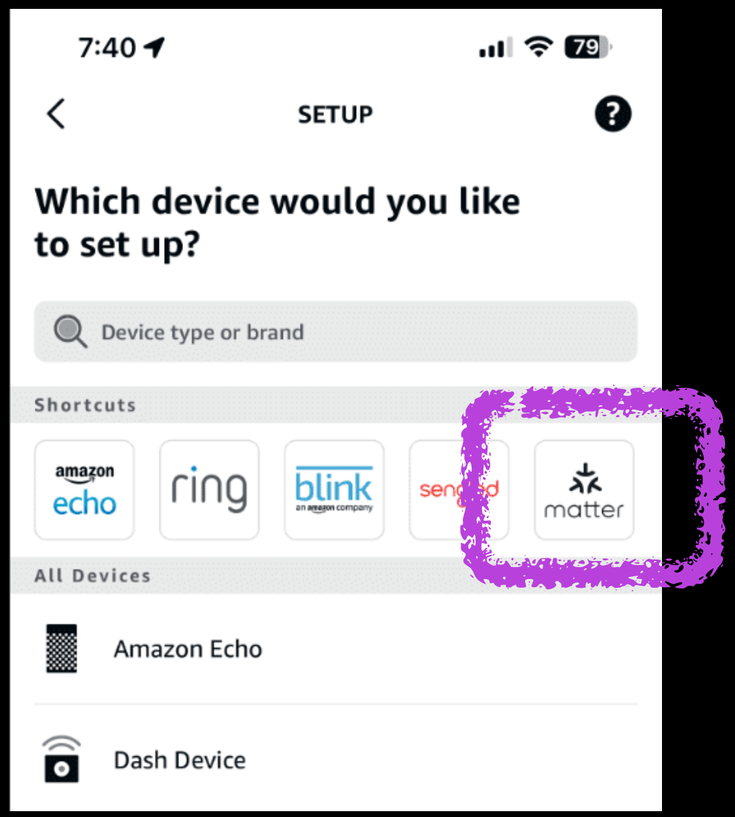
Funcionamento do Protocolo Matter

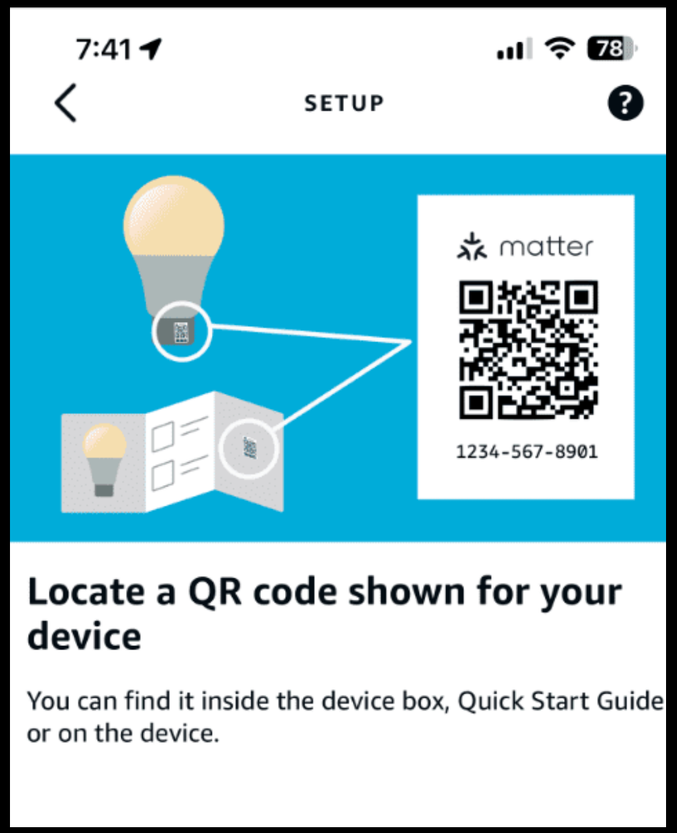


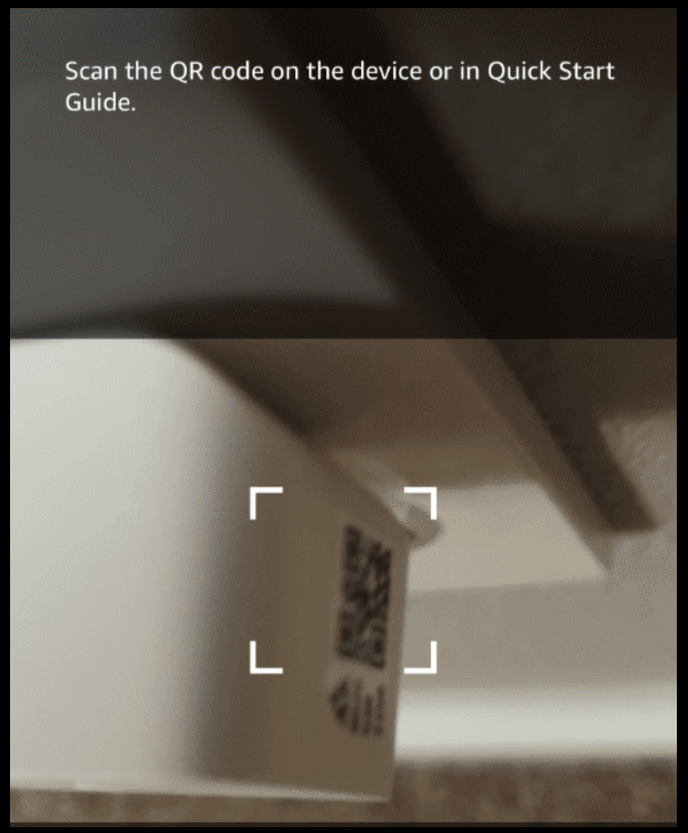
Funcionamento do Protocolo Matter

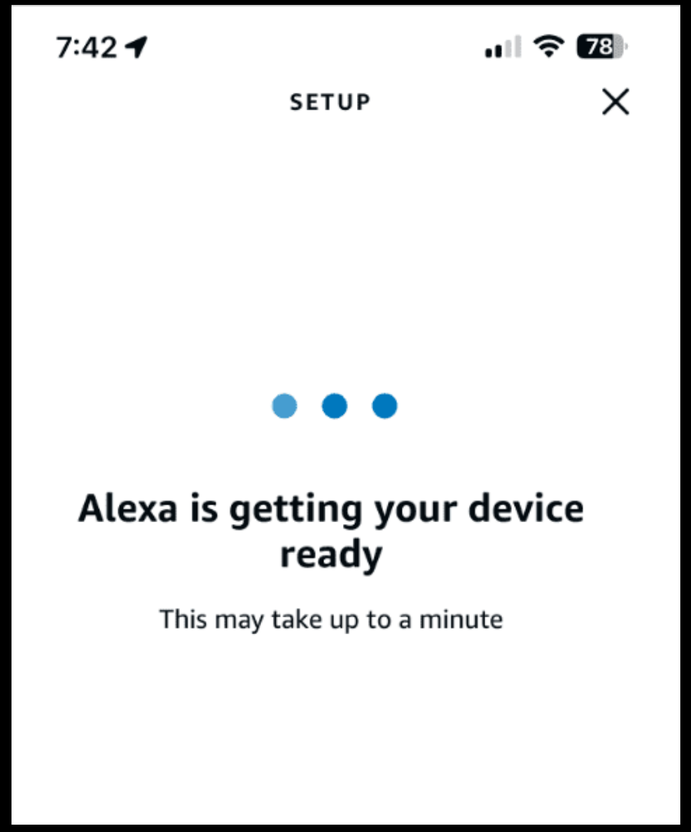


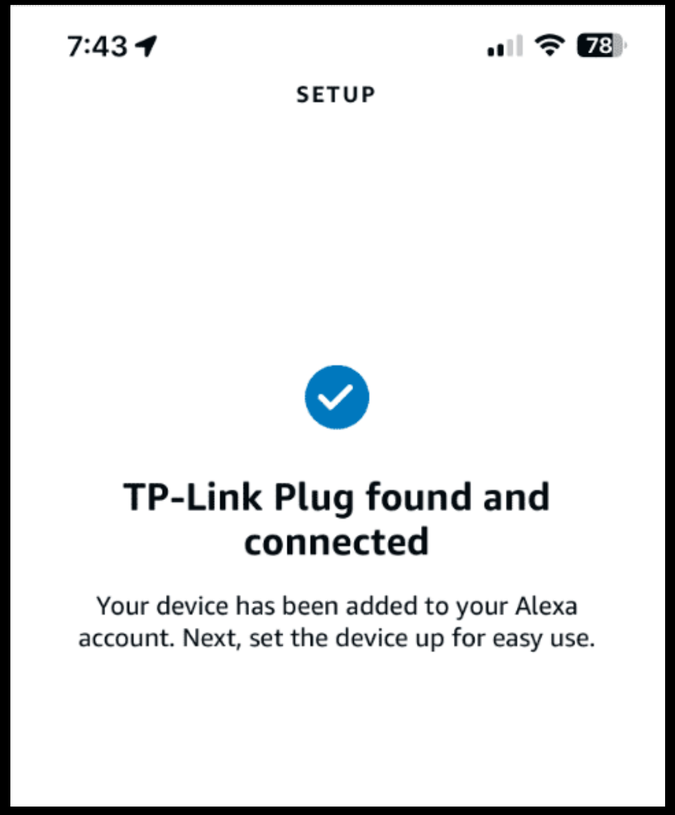
WiFi vs Thread

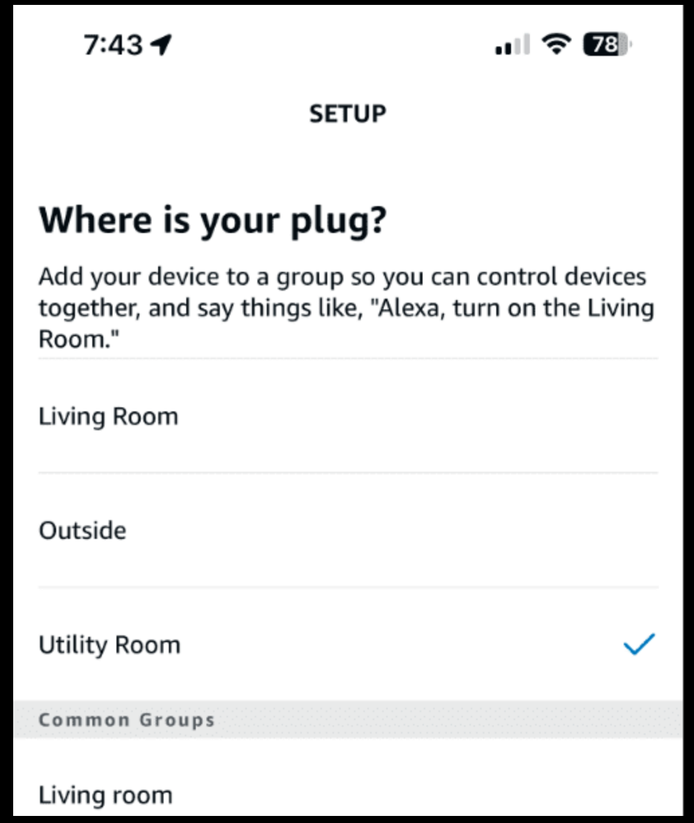












Comissionamento de Dispositivos na Rede da Casa

espressif.com

ESPRESSIF Hardware SDKs Cloud Solutions Support Ecosystem Company Contact English Subscribe

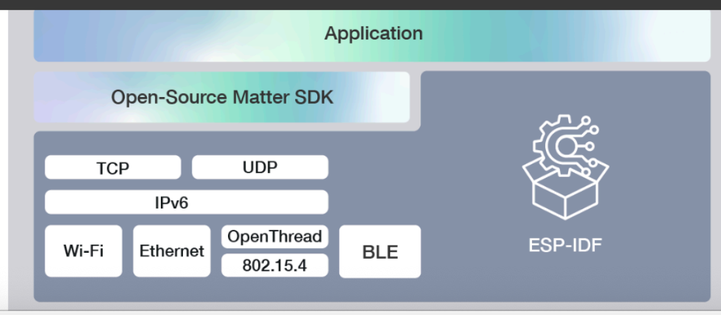
Solutions > ESP Matter Offerings

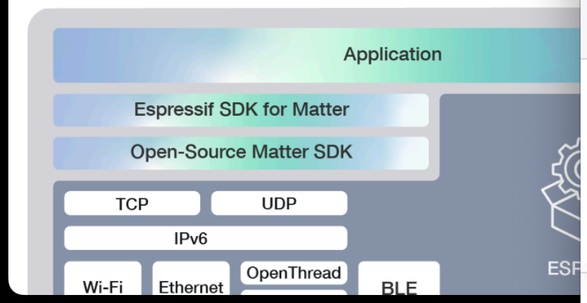
Overview Hardware Software ESP-ZeroCode Pre-Provisioning Video Get Started

Open-Source Matter SDK

This is the open-source implementation of the Matter protocol, which is jointly done by CSA member companies and the community. Not only does this provides a device implementation, but it also includes Android and iOS controller implementations. The ESP32 series of chips is an integral part of this open-source Matter SDK.

Open-Source Matter SDK on GitHub >





github.com/espressif/arduino-esp32/tree/master/libraries/Matter

espressif / arduino-esp32

Type / to search

Code Issues 114 Pull requests 24 Discussions Actions Projects 2 Wiki Security 1 Insights

Files

master

Go to file t

- Matter
  - examples
    - MatterColorLight
    - MatterCommissionTest
    - MatterComposedLights
    - MatterContactSensor
    - MatterDimmableLight
    - MatterEnhancedColorLight
    - MatterFan
    - MatterHumiditySensor
    - MatterMinimum
    - MatterOccupancySensor

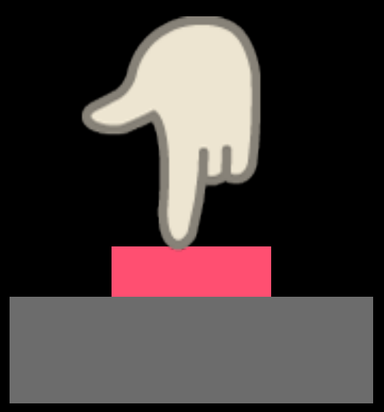
arduino-esp32 / libraries / Matter /

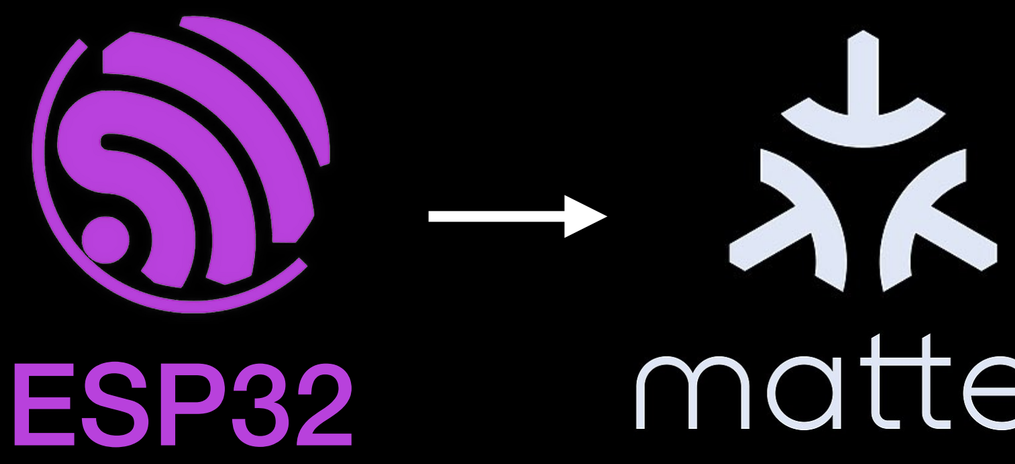
Add file

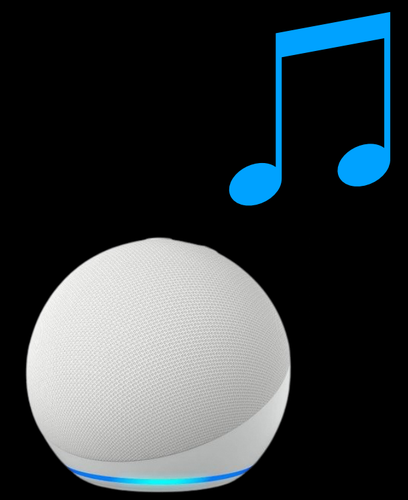
SuGlider and pre-commit-ci-lite[bot] fix(matter): examples must set pin to... d9dbc4a · 2 months ago History

Name	Last commit message	Last commit date
..		
examples	fix(matter): examples must set pin to Digital Mode after ...	2 months ago
src	fix(matter): removes a few matter 1.4 / IDF 5.4 compilati...	2 months ago
keywords.txt	feat(matter): new Matter Endpoint for Thermostat (#107...	4 months ago
library.properties	Update core version to 3.2.0	3 months ago

Biblioteca Matter para Arduino-ESP32







Eventos do Matter

```
#include <Matter.h>

void recomissionarMatter() {
  if (!Matter.isDeviceCommissioned()) {
    Serial.print("Código de pareamento: ");
    Serial.println(Matter.getManualPairingCode());
    Serial.print("Site com QR code de pareamento: ");
    Serial.println(Matter.getOnboardingQRCodeUrl());

    Serial.print("Procurando ambiente Matter...");
    while (!Matter.isDeviceCommissioned()) {
      Serial.print(".");
      delay(1000);
    }
    Serial.println("\nConectado no ambiente Matter!");
  }
}
```

Função de Comissionamento Matter

```
#include <Matter.h>

MatterColorLight luzRGBMatter;

bool mudouLuz(bool acender, espHsvColor_t corHSV) {
  espRgbColor_t corRGB = espHsvColorToRgbColor(corHSV);
  // acende ou apaga a sua luz de acordo com variável acender
  // atualizar cor da luz com corRGB.r, corRGB.g e corRGB.b
  return true;
}

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

  luzRGBMatter.begin();
  luzRGBMatter.onChange(mudouLuz);

  Matter.begin();
  recomissionarMatter();
}

void loop() {
  recomissionarMatter();
}
```

Código de Servidor Matter para Luz RGB

```
#include <Matter.h>

MatterGenericSwitch botaoMatter;

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

  botaoMatter.begin();

  Matter.begin();
  recomissionarMatter();
}

void loop() {
  recomissionarMatter();
}

    // quando o seu botão for pressionado pelo usuário...
botaoMatter.click(); // avisa rede Matter que botão foi clicado
```

Código de Cliente Matter para Botão Inteligente

```
MatterOccupancySensor movimentoMatter;
movimentoMatter.setOccupancy(true);  // avisa que teve movimento
movimentoMatter.setOccupancy(false); // avisa que teve inércia

MatterPressureSensor pressaoMatter;
pressaoMatter.setPressure(1100); // avisa nova pressão em hPa

MatterHumiditySensor umidadeMatter;
umidadeMatter.setHumidity(75.4); // avisa nova umidade em %

MatterTemperatureSensor temperaturaMatter;
temperaturaMatter.setTemperature(25.7); // temperatura em °C
```

Outros Tipos de Dispositivos Comandos de Eventos