

# Matter

Devendra Tewari

November 25, 2021

Matter

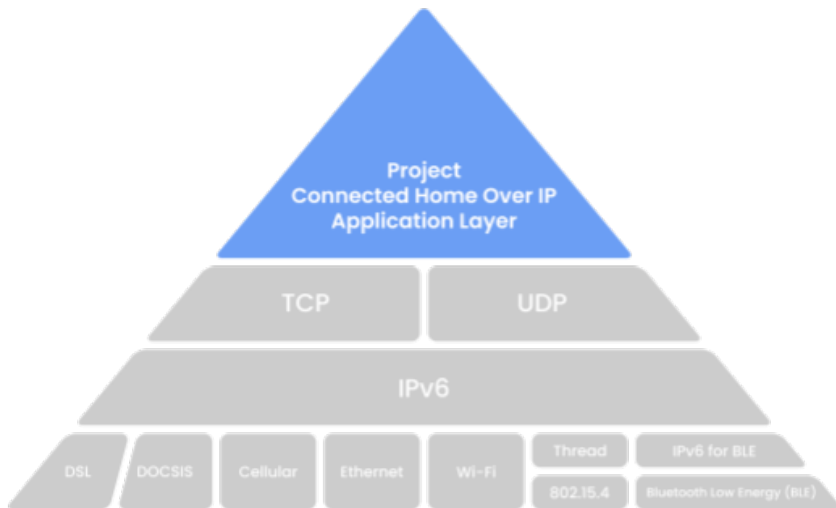
# Objective

- ▶ Why Matter?
- ▶ When will it be available?
- ▶ How to use it today?

# Why Matter?

- ▶ Why smart home?
- ▶ Limited smart home potential without interoperable devices
- ▶ Matter is driven by industry leading device manufacturers
- ▶ Matter implementation is open source and free of royalties
- ▶ Matter weaves together existing standards and fills in the blanks

# Internet Protocols



## When will it be available?

- ▶ Devices that support Matter pushed to sometime in 2022
- ▶ Preliminary support available on Android 12 and iOS 15
- ▶ Wide adoption is expected



Figure 2: Alliance Members

## How to use it today?

- ▶ Start by reading the docs at <https://github.com/project-chip/connectedhomeip>
- ▶ Try it out in Linux on a Raspberry Pi
- ▶ Try it out on an embedded device such as ESP32

## Core concepts

| Matter                     | HomeKit            | Zigbee         |
|----------------------------|--------------------|----------------|
| Attribute                  | Characteristic     | Attribute      |
| Binding                    | Event subscription | Binding        |
| Cluster                    | Services           | Cluster        |
| Commissioning / Rendezvous | Pairing            | Association    |
| Controller / Commissioner  | Admin              | Coordinator    |
| Device or Node             | Accessory          | Device or Node |
| Endpoint                   | Profile            | Endpoint       |
| Fabric                     | Network            | Network        |



# Architecture



## Linux Device Firmware Development

- ▶ Build and test on a Raspberry Pi 4

- ▶ Install toolchain

```
sudo apt-get install git gcc g++ python pkg-config \  
    libssl-dev libdbus-1-dev libglib2.0-dev \  
    ninja-build python3-venv python3-dev unzip
```

- ▶ Build and run all-clusters-app

```
git clone --recurse-submodules \  
    https://github.com/project-chip/connectedhomeip  
cd connectedhomeip  
unalias python  
source ./scripts/bootstrap.sh  
source ./scripts/activate.sh  
cd examples/all-clusters-app/linux  
gn gen out/debug  
ninja -C out/debug  
# Delete network  
./out/debug/chip-all-clusters-app --wifi
```

## ESP32 Device Firmware Development

- ▶ Build on macOS and test on M5STACK Core 2

- ▶ Install ESP-IDF

```
git clone https://github.com/espressif/esp-idf.git
```

```
cd esp-idf
```

```
git checkout v4.3
```

```
git submodule update --init
```

```
./install.sh
```

```
source ./export.sh
```

- ▶ Build and run all-clusters-app example on device

```
cd connectedhomeip
```

```
unalias python
```

```
source ./scripts/bootstrap.sh
```

```
source ./scripts/activate.sh
```

```
cd examples/all-clusters-app/esp32
```

```
idf.py build
```

```
idf.py -p /dev/cu.usbserial-022D45D6 erase_flash \
```

```
flash monitor
```

## chip-tool

Command line tool to commission and interact with devices

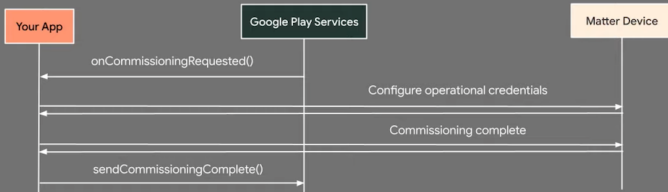
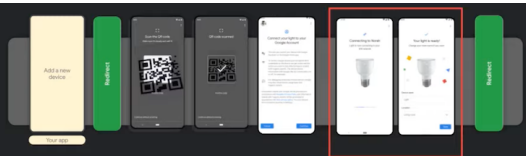
```
cd connectedhomeip
unalias python
source ./scripts/bootstrap.sh
source ./scripts/activate.sh
cd examples/chip-tool
gn gen out/debug
ninja -C out/debug
./out/debug/chip-tool onoff toggle 1 1
```

## Commissioning

- ▶ Configures device into a Matter fabric
- ▶ Pair device with multiple controllers
- ▶ Commissioning over BLE/Wi-Fi using chip-tool

```
chip-tool pairing ble-wifi \  
  ssid "password" \  
  0 20202021 3840
```

# Commission into your Fabric



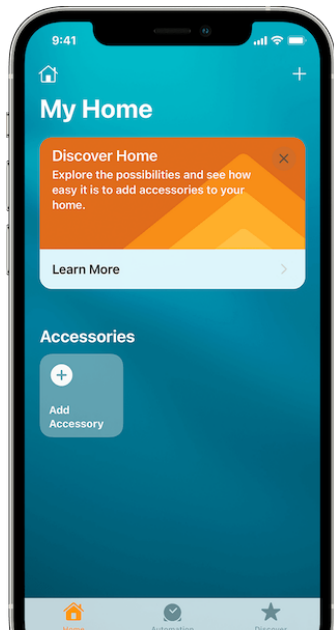
```
void onCommissioningRequested(CommissioningRequestMetadata metadata);
```

```
int sendCommissioningComplete(CommissioningCompleteMetadata metadata);
```

```
int sendCommissioningFailure(int errorCode);
```

## Commissioning on iOS

- ▶ Open the Home app and tap Add Accessory or Add
- ▶ Tap Add Accessory
- ▶ Use the camera on your iPhone, iPad, or iPod touch to scan the QR code on the accessory or accessory documentation
- ▶ When your accessory appears, tap it. If asked to Add Accessory to Network, tap Allow.
- ▶ Name your accessory and assign it to a room to help you identify it in the Home app and control it with Siri
- ▶ Tap Next, then tap Done.



## Read attributes using chip-tool

```
chip-tool onoff read on-off 1 1
chip-tool pressuremeasurement read measured-value 1 1
chip-tool relativehumiditymeasurement read measured-value 1 1
chip-tool temperaturemeasurement read measured-value 1 1
CHIP: [DMG]                                     }
CHIP: [DMG]
CHIP: [DMG]                                     Data = -32768,
CHIP: [DMG]                                     DataVersion = 0x0,
CHIP: [DMG]                                     },
```



## Write attributes using chip-tool

```
chip-tool onoff write on-time 5 1 1
```

```
chip-tool onoff read on-time 1 1
```

```
CHIP: [DMG] }
```

```
CHIP: [DMG]
```

```
CHIP: [DMG] Data = 5,
```

```
CHIP: [DMG] DataVersion = 0x0,
```

```
CHIP: [DMG] },
```

## Send commands using chip-tool

```
chip-tool onoff toggle 1 1
chip-tool onoff read on-off 1 1
CHIP: [DMG]                                }
CHIP: [DMG]
CHIP: [DMG]                                Data = true,
CHIP: [DMG]                                DataVersion = 0x0,
CHIP: [DMG]                                },
```

## View device configuration using ZAP Tool

- ▶ Endpoints are defined (along with the clusters and attributes they contain) in a .zap file which then generates code and static structures to define the endpoints
- ▶ Run Zigbee Cluster Configurator

```
brew install nvm
nvm use stable
cd connectedhomeip
cd third-party/zap/repo
npm i
npm run zap
```
- ▶ Open

```
examples/all-clusters-app/all-clusters-common/all-clusters-app.zap
```
- ▶ Data definition specified in Zigbee Cluster Library Specification

## Contributing to Matter

- ▶ Read CONTRIBUTING.md
- ▶ Submit bugs and features to <https://github.com/project-chip/connectedhomeip/issues>
- ▶ Change code
- ▶ Run automated test suite on host using act e.g.  

```
brew install act  
act -j test_suites_linux
```
- ▶ Run test on device using chip-tool  

```
chip-tool tests TestCluster 1
```
- ▶ Submit pull request via GitHub for maintainers to review and merge