

Implementare rilevamento presenze tramite ESP32 e Tag BTLe by SDeSalve

Prerequisiti: arduino IDE, Broker MQTT preconfigurato e relativi dati di accesso, Scheda ESP32

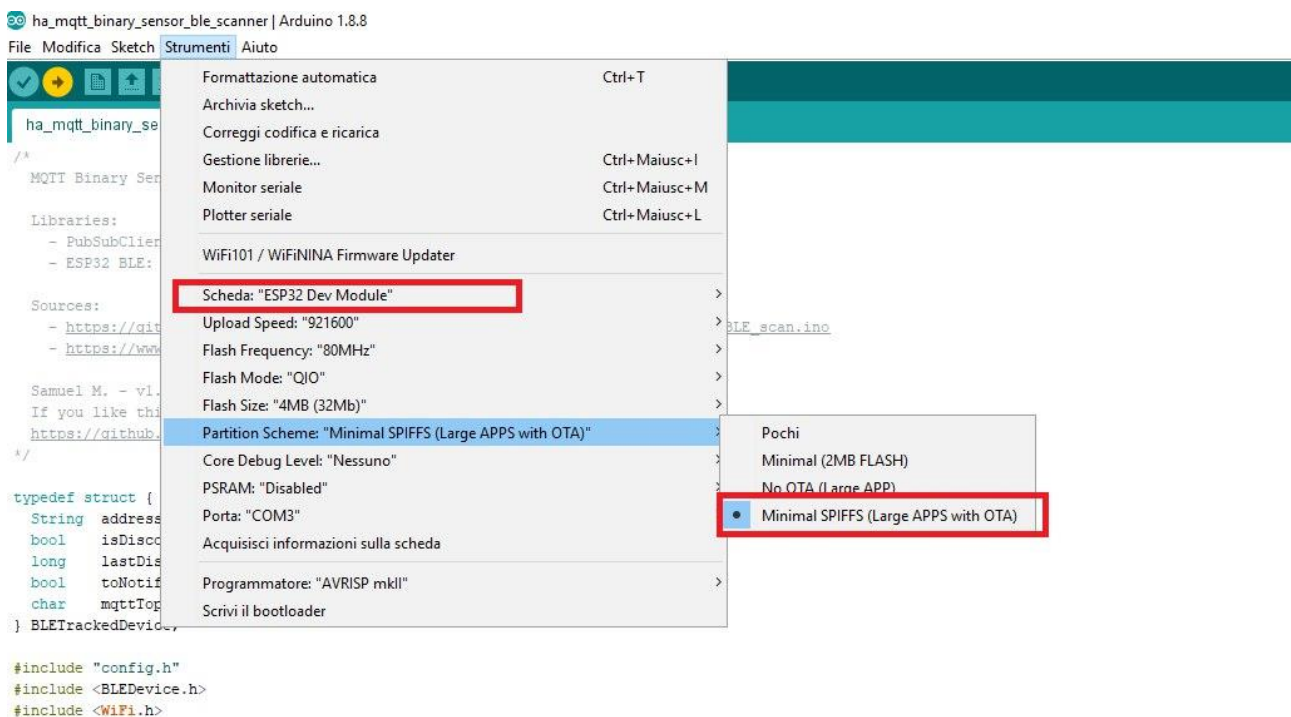
1) aggiungere scheda ESP32 ad Arduino IDE (seguire questa guida

<https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/>)

assicurarsi di avere i Driver corretti. Altrimenti scaricarli ed installarli da

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

2) Collegare la scheda ESP32 al computer. Aprire Arduino IDE ed impostare la scheda a ESP32 Dev Module e selezionare la Porta COM corretta. Seleziona le partizioni da utilizzare come nell'immagine seguente:



3) Testare il funzionamento della scheda ESP32 caricando questo sketch:

```
#include <BLEDevice.h>

#include <BLEUtils.h>

#include <BLEScan.h>

#include <BLEAdvertisedDevice.h>

int scanTime = 30; //In seconds

class MyAdvertisedDeviceCallbacks: public BLEAdvertisedDeviceCallbacks {

    void onResult(BLEAdvertisedDevice advertisedDevice) {

        Serial.printf("Advertised Device: %s \n", advertisedDevice.toString().c_str());

    }

};

void setup() {

    Serial.begin(115200);

}

void loop() {

    // put your main code here, to run repeatedly:

    delay(5000);

    Serial.println("Scanning...");

    BLEDevice::init("");

    BLEScan* pBLEScan = BLEDevice::getScan(); //create new scan

    pBLEScan->setAdvertisedDeviceCallbacks(new MyAdvertisedDeviceCallbacks());

    pBLEScan->setActiveScan(true); //active scan uses more power, but get results faster

    BLEScanResults foundDevices = pBLEScan->start(scanTime);

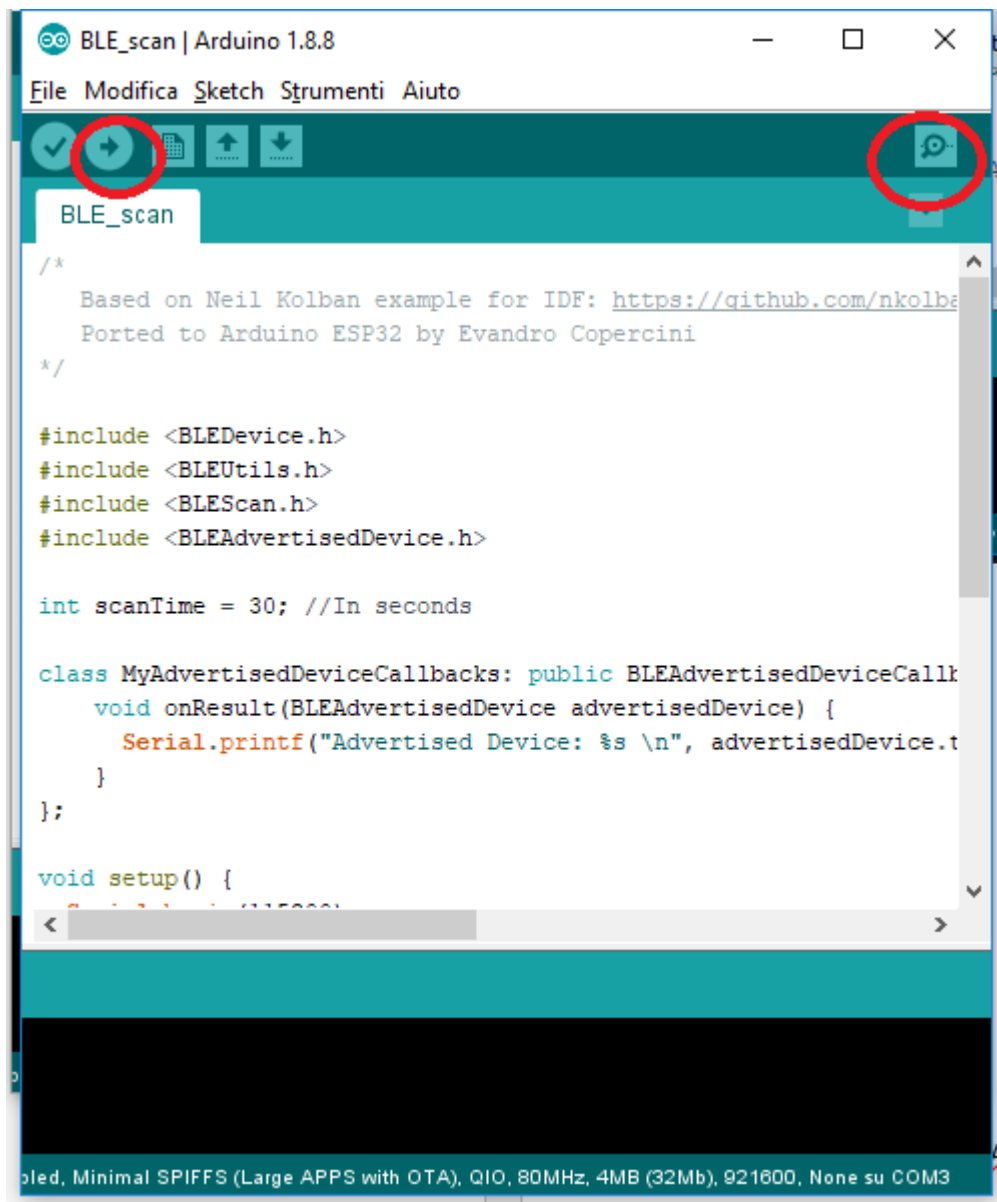
    Serial.print("Devices found: ");

    Serial.println(foundDevices.getCount());

    Serial.println("Scan done!");

}
```

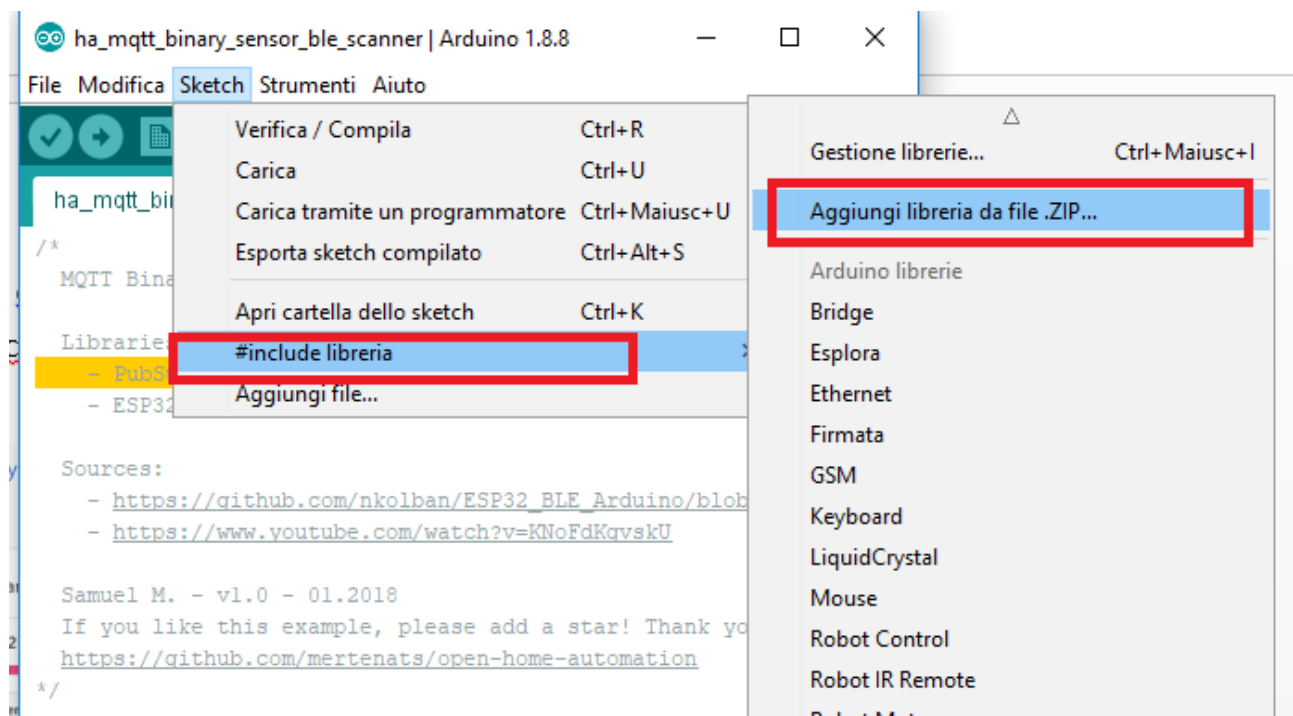
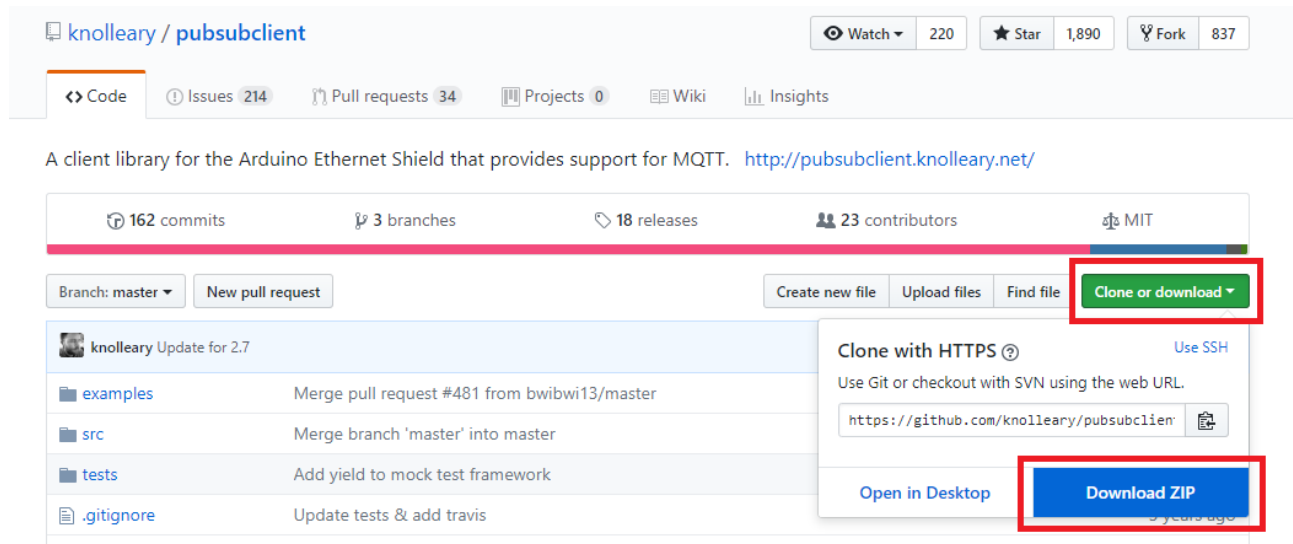
4) Inviare lo sketch alla ESP32 con la freccia a DX e premere la lente per vedere l'output della console:



Prendere nota dei MAC ADDRESS dei dispositivi BTL e da tracciare nella finestra della console

5) Scaricare ed installare in Ardiuno IDE la seguente libreria:

- PubSubClient: <https://github.com/knolleary/pubsubclient>



6) scaricare i seguenti file dal repository

https://github.com/sdesalve/dss_mqtt_binary_sensor_ble_scanner

- dss_mqtt_binary_sensor_ble_scanner_vX.X.ino
- example.config.h

7) rinominare example.config.h in config.h e personalizzarlo con i dati della propria rete Wifi e quelli del Broker MQTT.

8) Installare un client MQTT per collegarsi al Broker e vedere i topic.

Consiglio per Chrome questa APP:

<https://chrome.google.com/webstore/detail/mqttbox/kaajoficamnjjhkeomgfljpificfbkaf>

Configurarla per farla collegare al broker di Hassio inserendo i propri parametri nei campi evidenziati:

MQTT CLIENT SETTINGS

Menu Client Settings Help

MQTT Client Name: Hassio

MQTT Client Id: dfca50eb-60b7-49c5-a82f-2cc

Append timestamp to MQTT client id? ☒ Yes

Clean Session? ☒ Yes

Reschedule Pings? ☒ Yes

Broker is MQTT v3.1.1 compliant? ☒ Yes

Auto connect on app launch? ☒ Yes

Queue outgoing QoS zero messages? ☒ Yes

Protocol: mqtt / tcp

Host: 192.168.5.149

Username: mqtt

Password:

Reconnect Period (milliseconds): 1000

Connect Timeout (milliseconds): 30000

KeepAlive (seconds): 10

Will - Topic: Will - Topic

Will - QoS: 0 - Almost Once

Will - Retain: ☐ No

Will - Payload:

Save Delete

Sottoscrivere il topic # per vedere tutti i topic MQTT che verranno ricevuti dal Broker MQTT

9) compila, invia ed apri la schermata della console come indicato al punto 4 di questa guida.

Se tutto ha funzionato correttamente verrà visualizzato qualcosa del genere nella console di Arduino IDE:

```
#####INFO: MQTT availability topic: AFC40A24/availability
INFO: MQTT sensor topic: /sensor/casa/e0e1/ state
INFO: MQTT sensor topic: /sensor/casa/fc58/ state
INFO: Device discovered, Address: 6c:00:00:00:00:00, RSSI: -85
INFO: Device discovered, Address: 6c:00:00:00:00:00, RSSI: -85
INFO: Device discovered, Address: ca:00:00:00:00:00, RSSI: -65
INFO: Device discovered, Address: ca:00:00:00:00:00, RSSI: -65
INFO: Device discovered, Address: 41:00:00:00:00:00, RSSI: -74
INFO: Device discovered, Address: 41:00:00:00:00:00, RSSI: -74
INFO: Device discovered, Address: 72:00:00:00:00:00, RSSI: -66
INFO: Device discovered, Address: 72:00:00:00:00:00, RSSI: -66
INFO: Tracked device newly discovered, Address: e0:e1:00:00:00:00, RSSI: -66
INFO: Device discovered, Address: e0:e1:00:00:00:00, RSSI: -66
INFO: Device discovered, Address: 4c:00:00:00:00:00, RSSI: -78
INFO: Device discovered, Address: 4c:00:00:00:00:00, RSSI: -78
INFO: WiFi connecting to: TP-LINK_
..
192.168.5.114
INFO: The client is successfully connected to the MQTT broker
INFO: MQTT message published successfully, topic: AFC40A24/availability, payload: online
INFO: MQTT message published successfully, topic: /sensor/casa/e0e1/ state, payload: ON
INFO: WiFi connecting to: TP-LINK_
..
192.168.5.114
INFO: The client is successfully connected to the MQTT broker
INFO: MQTT message published successfully, topic: AFC40A24/availability, payload: online
INFO: MQTT message published successfully, topic: /sensor/casa/e0e1/ state, payload: OFF
```

ha_mqtt_binary_sensor_ble_scanner | Arduino 1.8.8

File Modifica Sketch Strumenti Aiuto

ha_mqtt_binary_sensor_ble_scanner config.h

MQTT Binary Sensor - Bluetooth LE Device Tracker - Home Assistant

Libraries:

- PubSubClient: <https://github.com/knolleary/pubsubclient>
- ESP32 BLE: https://github.com/nkolban/ESP32_BLE_Arduino

Sources:

- https://github.com/nkolban/ESP32_BLE_Arduino/blob/master/ex

Caricamento completato

```
Writing at 0x000d4000... (98 %)
Writing at 0x000d8000... (100 %)
Wrote 1422288 bytes (820009 compressed) at 0x00010000 in 11.7 seconds
Hash of data verified.
Compressed 3072 bytes to 144...

Writing at 0x00008000... (100 %)
Wrote 3072 bytes (144 compressed) at 0x00008000 in 0.0 seconds (effective 115200 bps)
Hash of data verified.

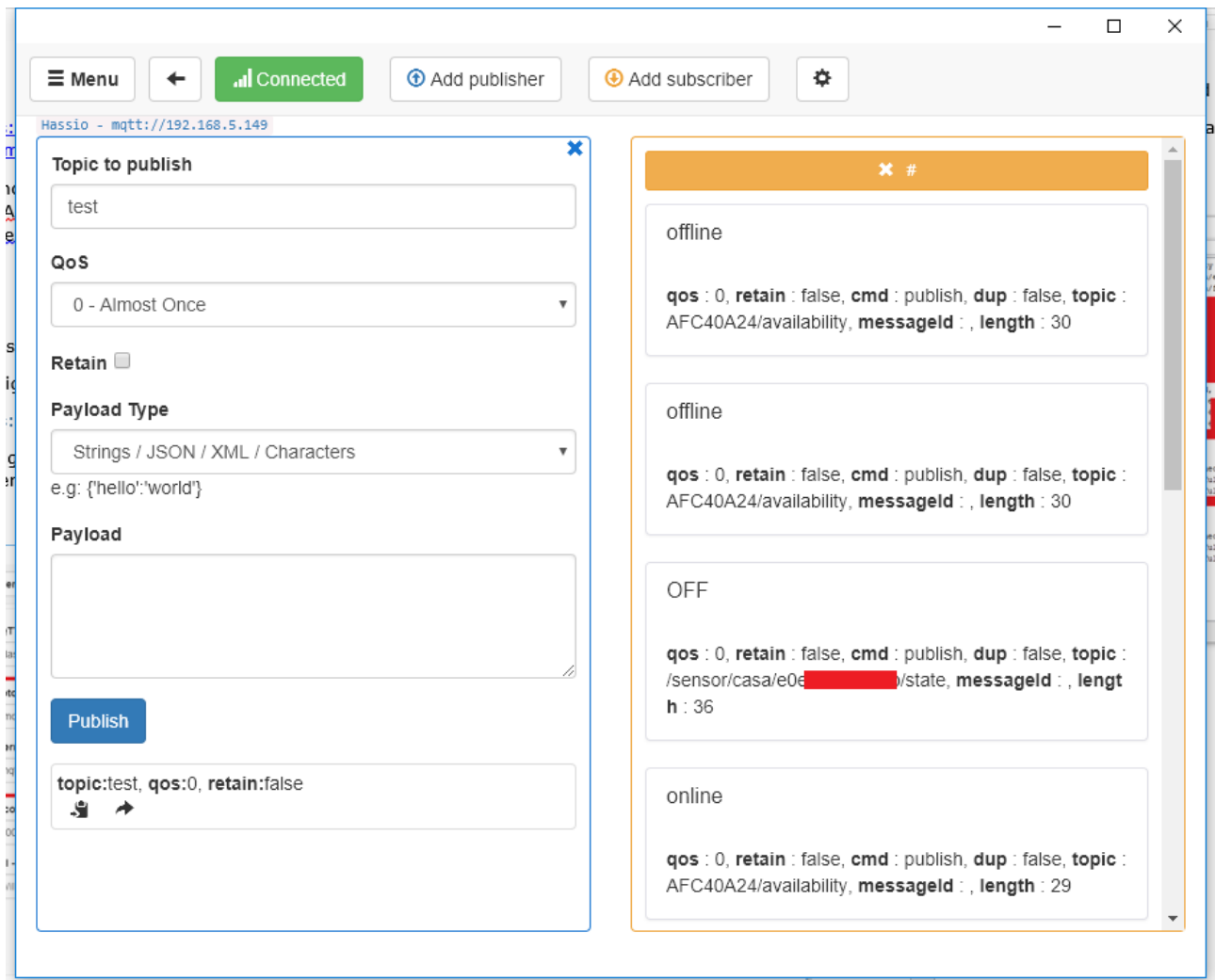
Leaving...
Hard resetting via RTS pin...
```

Ad, Minimal SPIFFS (Large APPS with OTA), QIO, 80MHz, 4MB (32Mb), 921600, None su COM3

☒ Scorrimento automatico ☐ Visualizza orario

A capo (NL) 115200 baud Ripulisci l'output

E qualcosa del genere nel client MQTT



10) Aggiungi ad Home Assistant uno o più sensor MQTT per tracciare la presenza dei tag BTL sensor:

```
- platform: mqtt
  name: "Tracker iTag Utente"
  state_topic: "dss_ble2mqtt/XXXXXXXX/casa/XXXXXXXX/state"
  availability_topic: "dss_ble2mqtt/XXXXXXXX/availability"
  value_template: >-
    {% if value == 'ON' %}
      home
    {% elif value == 'OFF' %}
      away
    {% else %}
      offline
    {% endif %}
```

Può essere aggiunto anche un sensore per tracciare gli RSSI del tracker:

```
- platform: mqtt  
  name: "RSSI Tracker iTag Utente"  
  state_topic: "dss_ble2mqtt/XXXXXXXXX/casa/XXXXXXXXX/rssi"  
  availability_topic: "dss_ble2mqtt/XXXXXXXXX/availability"
```

Scritta da @SDeSalve il 08/12/2018 22:30:11

Aggiornata il 23/07/2019 22:06

Possono essere tracciati tutti i dispositivi Bluetooth Low Energy che fanno advertising del loro MAC address.

Attenzione il MAC Address deve essere fisso e non variabile. E di solito il dispositivo non deve essere associato ad un cellulare per continuare a fare advertising...

Consiglio questi:

<https://amzn.to/2VReXq1>



Decdeal Key Finder BT Anti-Perso Tracker Smart Tag Allarme Localizzatore Bambini/Telefono/Chiave/Portafoglio/Oggetto Ricerca per iOS/Android

di Decdeal
★★★★☆ 9 recensioni clienti

Prezzo consigliato: EUR 13,99
Prezzo: EUR 6,59 Spedizione senza costi aggiuntivi con Prime Dettagli
Risparmi: EUR 7,40 (53%)
Tutti i prezzi includono l'IVA.

Nuovi: 2 venditori da EUR 6,59

Colore: **Blau**



Nota: Questo articolo può essere consegnato in un **punto di ritiro**. Dettagli

- **ANTI-LOST**- Quando tracker e telefono sono separati, entrambi emetteranno un segnale acustico per allarmarti, quindi non preoccuparti di lasciare oggetti indietro. Ottimo per i bambini, i bagagli, gli animali domestici, la chiave, il portafoglio, la borsa e altro ancora.
- **UN TASTO TROVA**- Quando non riesci a trovare i tasti o un portafoglio, puoi fare clic sul pulsante 'Nut' nell'app, il localizzatore emetterà un segnale acustico, seguendo il suono puoi trovare facilmente chiavi o un portafoglio.
- **REGISTRAZIONE LOCATION**- Se il tuo oggetto è stato perso e hai perso allarme. App registrerà automaticamente ultima posizione di disconnessione, per aiutarti a richiamare e trovare gli oggetti velocemente.
- **GESTIONE MULTI-OGGETTO**- I telefoni Android possono gestire 4-6 tracker e il telefono iOS può gestire 12 tracker allo stesso tempo. Inoltre, puoi nominarli per una facile gestione.
- **GESTIONE DELLE AZIONI**- Quando oggetto è utilizzato da molte persone, come il telecomando TV, è possibile aprire la modalità oggetto e dividerlo con altri utenti, il tracker sarà connesso solo quando lo si sta cercando.

☐ Segnala informazioni inesatte.

Amazon Business
Se sei un cliente Amazon per lavoro ti consigliamo di creare un account business Amazon Business