So verwendest du Bluetooth am ESP32

Bluetooth ist allgegenwärtig und auch dein ESP32 kann damit umgehen. Lerne in diesem Tutorial, wie du Bluetooth Classic verwendest und Daten zwischen einem Smartphone und deinem ESP32 austauschst.

Für dieses Tutorial benötigst du nur einen ESP32 und ein Android-Smartphone, auf dem du die kostenfreie App **Serial Bluetooth Terminal** installieren kannst.

So machst du deinen ESP32 in der Arduino IDE verfügbar

Falls du deinen ESP32 bisher noch nicht mit der Arduino IDE programmiert hast, führe bitte erst die folgenden Schritte durch:

Öffne die Einstellungen der IDE und trage in des Feld **Zusätzliche Boardverwalter-URLs** folgenden Link ein.

https://dl.espressif.com/dl/package_esp32_index.json

Öffne anschließend den **Boardverwalter** unter **Werkzeuge/Board**. Suche dort nach **ESP32** und installiere dir die neueste Version des gleichnamigen Pakets von **espressif Systems**. Und das war es

auch schon – du solltest deinen ESP32 nun wie jedes andere Board in der Arduino IDE auswählen und verbinden können.

Der Sketch für deinen ESP32

Für ein grundlegendes Experiment mit Bluetooth benötigst du nur wenig Code:

```
#include "BluetoothSerial.h"

BluetoothSerial SerialBT;

void setup() {
  Serial.begin(115200);
  SerialBT.begin("ESP32test"); //Name des ESP32
  Serial.println("Der ESP32 ist bereit. Verbinde dich nun über Bluetooth.");
}

void loop() {
  if (Serial.available()) {
    SerialBT.write(Serial.read());
  }
  if (SerialBT.available()) {
    Serial.write(SerialBT.read());
  }
  delay(25);
}
```

Schauen wir uns die wichtigsten Teile genauer an: Zunächst bindest du die Bibliothek **BluetoothSerial.h** ein, die bereits in deiner Arduino IDE verfügbar ist und nicht extra installiert werden muss. Anschließend erstellst du eine Instanz namens **SerialBT**.

Im Setup startest du die serielle Kommunikation mit einer Baudrate von 115200 – **achte darauf, dass dein Serieller Monitor auch auf diese Rate eingestellt ist.** Anschließend initialisierst du Bluetooth und gibst deinem ESP32 den Namen

ESP32test.

Im Loop befinden sich zwei Abfragen: Die erste prüft, ob du etwas in den Seriellen Monitor eingetragen und abgeschickt hast.

Wenn das der Fall ist, schickst du die Daten an den ESP32:

```
SerialBT.write(Serial.read());
```

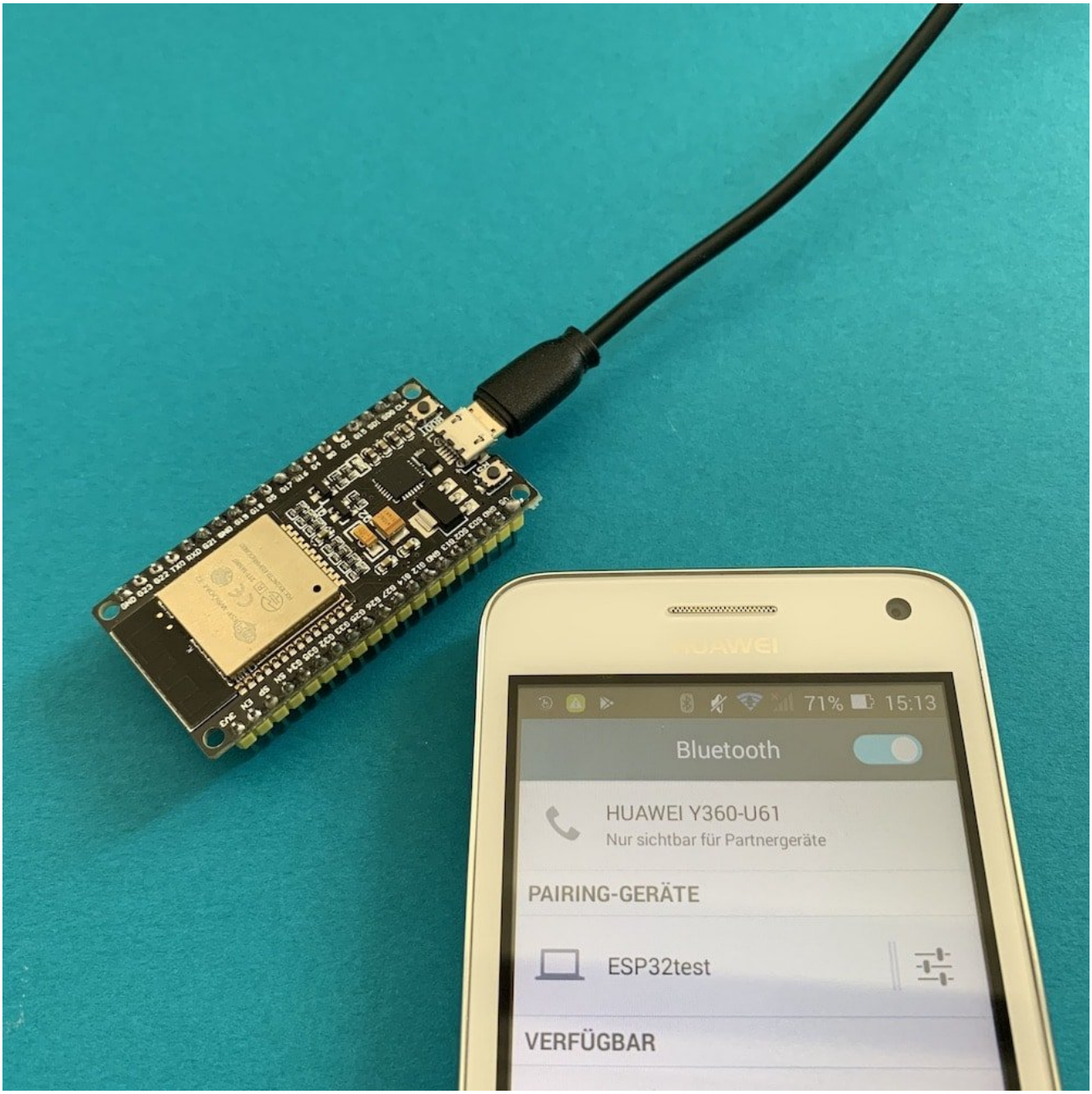
In der zweiten Abfrage wird geprüft, ob der ESP32 etwas über Bluetooth empfangen hat. Diese Daten werden dann im Seriellen Monitor ausgegeben.

Ein erster Test

Gehen wir also zur Praxis über. Für diesen Test verbindest du ein Android-Smartphone mit deinem ESP32 und schickst Nachrichten von einem Gerät aufs andere.

Lade zunächst den obigen Sketch wie gewohnt auf deinen ESP32 und starte den Seriellen Monitor mit der Baudrate 115200. Sollte sich hier nichts tun, drücke den Reset- oder Enable-Button am ESP32. Anschließend sollte nach einigen Hardware-Informationen folgender Satz aus deinem Sketch angezeigt werden: **Der ESP32 ist bereit. Verbinde dich nun über Bluetooth.**

In den Bluetooth-Einstellungen deines Smartphone solltest du nun deinen ESP32 mit dem Namen **ESP32test** sehen und eine Verbindung herstellen können.



Falls noch nicht geschehen, lade dir die App Serial Bluetooth Terminal aus dem Play Store und öffne sie. Unter dem Menüpunkt **Devices/Bluetooth Classic** erscheint nun ebenfalls dein ESP32, mit dem du dich mit einem Tap verbinden kannst.

Zeit für etwas Konversation! Schreibe deinem ESP32 in der App eine Nachricht und sende sie ab. Dein Text wird auf der Gegenseite empfangen und im Seriellen Monitor ausgegeben.

In die andere Richtung funktioniert das genauso. Schreibe etwas in das Eingabefeld oben in deinem Seriellen Monitor und klicke auf Senden. Diese Nachricht wird dann von deinem ESP32 aus an dein Smartphone gesendet und dort angezeigt:

Und das war auch schon dein erster Test. Wenn du Nachrichten hin und herschicken kannst, geht das natürlich auch mit allen anderen Daten.

Als nächstes kannst du einen Sensor an den ESP32 anschließen und dessen Daten in regelmäßigen Abständen an dein Smartphone senden. Oder andersherum mit deinem Smartphone z.B. eine LED an deinem Microcontroller an- und ausschalten. Wie du vom Smartphone aus Geräte steuert, lernst du auch in unserem Kurs Dein eigener ESP8266 Webserver.