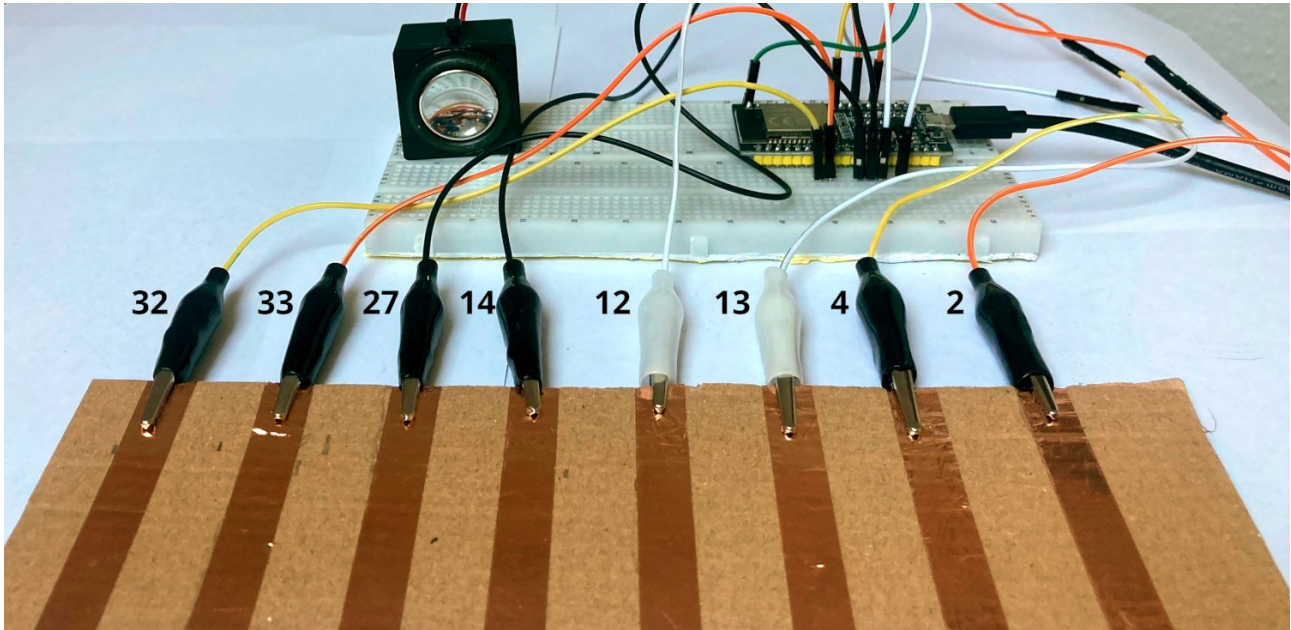




Ziel des Projekts

Das Programm reagiert auf die Touch-Pins des ESP32-Wroom und spielt zu jeder Taste einen Ton der C-Dur Tonleiter.



Sollte beim Spielen der Kontakt nicht sofort zustande kommen, feuchte deine Finger ein wenig an.

Die Hardware

Board installieren:

Einstellungen

Einstellungen

Netzwerk

Dateipfad des Sketchbooks:
/home/hartmut/Arduino

DURCHSUCHEN

☐ Dateien im Sketch zeigen

Editor Schriftgröße:
14

Größe der Benutzeroberfläche:
☒ Automatisch 100 %

Farbdesign:
Hell

Editorsprache:
Deutsch (Reload required)

Compiler-Meldungen anzeigen beim
☐ Kompilieren ☐ Hochladen

Compiler-Meldungen
Kein/e/r

☐ Code nach Hochladen überprüfen
☐ Automatisch speichern
☒ Schnelle Editor Vorschläge

Zusätzliche Boardverwalter-URLs

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

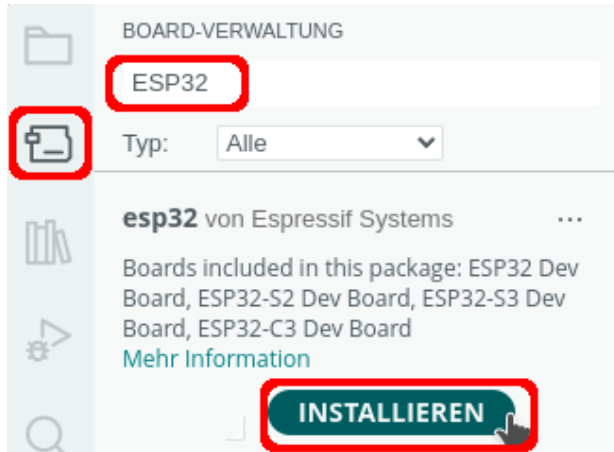
ABBRECHEN

OK



Trage unter Datei -> Einstellungen eine zusätzliche Boardverwalter-URL ein:

https://espressif.github.io/arduino-esp32/package_esp32_index.json



→ Icon für den Boardverwalter anklicken
oder:

→ Werkzeuge-> Board -> Boardverwalter

→ nach ESP32 suchen

→ Board installieren



Anschließend wird das Board ausgewählt. Je nach Betriebssystem wird der Port eine andere Bezeichnung haben.



Touch-Pins

32

33

27

13

4

2

15

Der ESP32-Wroom verfügt über Touch-Pins. Sie reagieren auf die Änderungen von elektrischer Kapazität. Je größer die Kapazität, desto kleiner der Messwert. Da der menschliche Körper auch eine Kapazität darstellt, reagieren die Touch-Pins auf Berührung.



Beispiel Kabel am Touch-Pin 32:

```
void setup()
{
  Serial.begin(9600);
  delay(1000);
}

void loop()
{
  Serial.print("Wert: ");
  Serial.print(touchRead(32));
  if (touchRead(32) < 20) Serial.println(" -> Pin berührt");
  else Serial.println(" -> Pin nicht berührt");

  delay(1000);
}
```

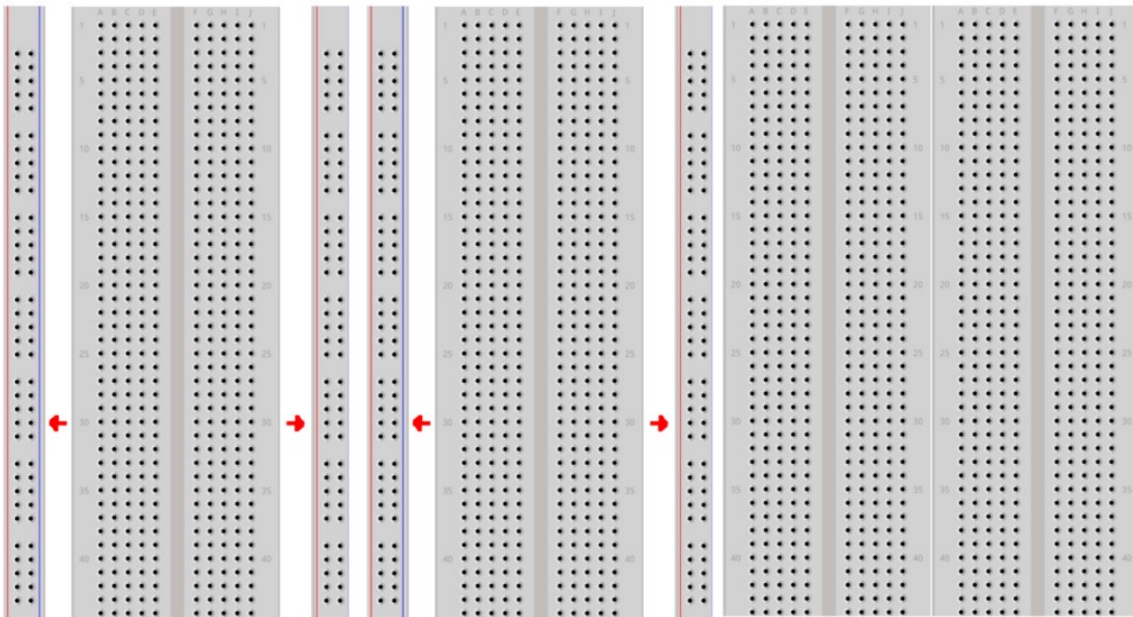


Für die Realisierung wird neben dem ESP32-Wroom und dem Lautsprecher noch ein wenig Material benötigt:

- ➔ 8 Kabel mit Krokodilsteckern an einem Ende und Steckern am anderen Ende
- ➔ stabile Pappe
- ➔ selbstklebende Kupferfolie oder Alufolie
- ➔ „doppeltes“ Steckbrett

Der ESP32-Wroom ist leider nicht sehr „steckbrettfreundlich“, daher habe ich das Steckbrett selbst erstellt:

Ich habe jeweils die äußeren Leisten zweier Steckbretter entfernt. In die übrig gebliebenen Mittelteile lässt sich das ESP32-Wroom einstecken.



Der Schaltplan



GND -Lautsprecher -

17 Lautsprecher +

Touch-Pins

Touch-Pins

32

33

27

13

4

2

15



Das Programm

Das Programm ist sehr kurz:

Mit `touchRead()` wird der Wert des jeweiligen Touch-Pins abgefragt, der Schwellwert des Touch-Pins wird auf 20 festgelegt. Du musst ihn eventuell anpassen.

Die Zuordnung der Frequenzen:

Note	Frequenz	Touch-Pin
c'	262	32
d'	294	33
e'	330	27
f'	349	14
g'	392	12
a'	440	13
h'	494	4
c''	523	2

```
// Pin des Lautsprechers
#define Lautsprecher 17

// Länge des gespielten Tons
#define TonLaenge 500

// Schwellwert des Touch-Pins
#define Schwellwert 20

void setup()
{
    // kein setup notwendig}
}

void loop()
{
    if (touchRead(32) < Schwellwert) tone(Lautsprecher, 262, TonLaenge); // T9
    if (touchRead(33) < Schwellwert) tone(Lautsprecher, 294, TonLaenge); // T8
    if (touchRead(27) < Schwellwert) tone(Lautsprecher, 330, TonLaenge); // T7
    if (touchRead(14) < Schwellwert) tone(Lautsprecher, 349, TonLaenge); // T6
    if (touchRead(12) < Schwellwert) tone(Lautsprecher, 392, TonLaenge); // T5
    if (touchRead(13) < Schwellwert) tone(Lautsprecher, 440, TonLaenge); // T4
    if (touchRead(4) < Schwellwert) tone(Lautsprecher, 494, TonLaenge); // T0
    if (touchRead(2) < Schwellwert) tone(Lautsprecher, 523, TonLaenge); // T2

    // notwendiges delay, damit sich der Wert wieder normalisieren kann
    // evtl anpassen um doppelte Töne zu vermeiden
    delay(200);
}
```