HC-SR04 Wemos WiFi Setup



const int trigPin2 = D1;

const int echoPin2 = D2; a hc sr 04 bekötése a wemos esp 32 d1 minibe. csináld meg hogy a mini csatlakozzon a Bwifi hálózathoz jelszeva: "password" és dhcpn kapjon ipt mait kiir serial porton és a weblapon 250 ms en ként frissítse a hc04 által márt távolságot

Az alábbi példa megmutatja, hogyan csatlakoztasd a **HC-SR04 ultrahangos szenzort** a **Wemos D1 Mini ESP32**-re, majd csatlakoztasd a `**Bwifi**` nevű WiFi hálózathoz jelszóval `**password**`, és jelenítsd meg a távolságot:

- a soros porton
- egy weboldalon, 250 ms-onként frissítve az értéket.

🧠 Hardver bekötés (Wemos D1 Mini ESP32):

HC-SR04	ESP32 (D1 Mini)
VCC	5V
GND	GND
Trig	D1 (GPIO5)
Echo	D2 (GPIO4)

Arduino kód:

```
срр
#include <WiFi.h>
#include <WebServer.h>
const char* ssid = "Bwifi";
const char* password = "password";
const int trigPin = 5; // D1 = GPIO5
const int echoPin = 4; // D2 = GPIO4
WebServer server(80);
long duration;
float distance;
unsigned long lastUpdate = 0;
const unsigned long updateInterval = 250;
void setup() {
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  WiFi.begin(ssid, password);
  Serial.print("Csatlakozás a WiFi-hez");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
```

```
Serial.print(".");
  .
Serial.println("\nWiFi kapcsolat létrejött");
  Serial.print("IP-cím: ");
Serial.println(WiFi.localIP());
  server.on("/", handleRoot);
server.begin();
  Serial.println("Webszerver elindítva");
void loop() {
  server.handleClient();
  if (millis() - lastUpdate > updateInterval) {
    distance = measureDistance();
    lastUpdate = millis();
}
float measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  float dist = duration * 0.034 / 2;
  Serial.print("Távolság: ");
  Serial.print(dist);
  Serial.println(" cm");
  return dist;
void handleRoot() {
  String html = "<!DOCTYPE html><html><head><meta http-equiv='refresh' content='0.25'>";
  html += "<meta charset='UTF-8'><title>Távolságmérés</title></head><body>";
  html += "<h1>Távolság: ";
  html += String(distance, 2);
html += " cm</hl>/html>";
  server.send(200, "text/html", html);
```

Megjegyzések:

- A HTML oldal 250 ms-onként automatikusan frissül (`<meta http-equiv='refresh' content='0.25'>`).
- A távolság mérés is 250 ms időközönként frissül a `loop()` függvényben.

Ha szeretnél AJAX-os megoldást automata frissítésre anélkül, hogy újratöltené az egész oldalt, jelezd, és átalakítom!



a webserver . h melyik packageban van

A `WebServer.h` az ESP32 esetén a következő könyvtárban található:

Könyvtár:



срр

#include <WebServer.h>

Ez a `WebServer` osztály az ESP32-hez tartozó alapértelmezett Arduino könyvtár része, és az alábbi csomagban található:



ESP32 Arduino Core

Telepítéshez:

Arduino IDE-ben →

Eszközök > Kártya > Kártyakezelő > ESP32 → Telepítsd:

`esp32 by Espressif Systems`

Telepítési lépések Arduino IDE-ben:

- 1. Menj a "Eszközök > Kártya > Kártyakezelő..." menüpontra.
- 2. Keresd meg: `esp32`
- 3. Telepítsd az "esp32 by Espressif Systems" csomagot.
- 4. Ezután válaszd ki a megfelelő táblát: `WEMOS D1 MINI ESP32` vagy `ESP32 Dev Module`.

📏 Ellenőrzés:

Miután telepítetted, ez a fájl elérhető lesz:

bash

ini

<Arduino sketchbook folder>/packages/esp32/hardware/esp32/.../libraries/WebServer/src/WebServer.h

A `WebServer` automatikusan elérhető az `esp32` platformmal. A `platformio.ini` fájlban ez lehet például:

[env:esp32dev] platform = espressif32 board = esp32dev

framework = arduino

Ha szeretnéd, segítek automatikusan beállítani a PlatformIO vagy Arduino IDE környezetet ehhez a projekthez.



C:\Users\Bence\Documents\Arduino\sketch_feb24b\sketch_feb24b.ino:2:10: fatal error: WebServer.h: No such file or directory

2 | #include <WebServer.h>

٨~~~~~

compilation terminated. exit status 1

Compilation error: WebServer.h: No such file or directory

Ez a hiba azt jelenti, hogy az `WebServer.h` fájlt nem találja az Arduino fordító — valószínűleg azért, mert:

- Nem a megfelelő ESP32 kártya van kiválasztva, VAGY
- Nem az ESP32, hanem az ESP8266 kártya van kiválasztva.

🔽 Megoldás lépésenként

1. 📥 Telepítsd az ESP32 board támogatást

- 1. Arduino IDE-ben menj ide: Eszközök > Kártya > Kártyakezelő...
- 2. A keresőmezőbe írd be: `esp32`
- Telepítsd: `esp32 by Espressif Systems`
- 4. Várd meg a telepítést.

2. V Válaszd ki a megfelelő ESP32 kártyát

Menj ide:

Eszközök > Kártya és válaszd ki:

`WEMOS D1 MINI ESP32`

vagy

`ESP32 Dev Module` (ez is működik, ha az előbbi nincs)

3. 🔁 Indítsd újra az Arduino IDE-t

Ezután próbáld újra lefordítani a programot.

i Fontos megjegyzés

- Az `ESP8266` használ `ESP8266WebServer.h`-t
- Az `ESP32` használ `WebServer.h`-t

Tehát ha véletlenül az **ESP8266 kártya** van kiválasztva, de `WebServer.h`-t próbálsz használni, ez a hiba jön elő.

Ha nem vagy biztos benne, melyik kártya van kiválasztva, kérlek másold ide a `Tools > Board` menüpont beállításaidat, és megnézem.



esp 8266 om van

Köszönöm a pontosítást! Mivel **ESP8266**-od van (pl. Wemos D1 Mini ESP8266), a `WebServer.h` nem fog működni — helyette az ESP8266 saját könyvtárát kell használnod:

A helyes include:

```
cpp
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
```

🔁 Átdolgozott kód ESP8266-ra:

```
срр
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
const char* ssid = "Bwifi";
const char* password = "password";
const int trigPin = D1;
const int echoPin = D2;
ESP8266WebServer server(80);
long duration;
float distance;
unsigned long lastUpdate = 0;
const unsigned long updateInterval = 250;
void setup() {
   Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  WiFi.begin(ssid, password);
Serial.print("Csatlakozás a WiFi-hez");
while (WiFi.status() != WL_CONNECTED) {
     delay(500);
Serial.print(".");
  Serial.println("\nWiFi kapcsolat létrejött");
  Serial.print("IP-cím: ");
  Serial.println(WiFi.localIP());
  server.on("/", handleRoot);
server.begin();
  Serial.println("Webszerver elindítva");
void loop() {
  server.handleClient();
  if (millis() - lastUpdate > updateInterval) {
     distance = measureDistance();
     lastUpdate = millis();
  }
float measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
```



```
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);
float dist = duration * 0.034 / 2;

Serial.print("Távolság: ");
Serial.print(dist);
Serial.println(" cm");
return dist;
}

void handleRoot() {
   String html = "<!DOCTYPE html><html><head><meta http-equiv='refresh' content='0.25'>";
   html += "<meta charset='UTF-8'><title>Távolságmérés</title></head><body>";
   html += "string(distance, 2);
   html += "cm</hl>
// hl>
// html += "cm</hl>
// hl>
// server.send(200, "text/html", html);
}
```

🔏 Ne felejtsd:

- A kártyánál válaszd ki: Wemos D1 R1, Wemos D1 Mini, vagy NodeMCU 1.0 (ESP8266-alapú modul)
- A `pulseIn()` megbízhatósága függ a vezeték hosszától és zavaroktól. Lehet érdemes szoftveres timeoutot is hozzáadni, ha később fagyásokat tapasztalsz.

Szükséged van AJAX-alapú webes frissítésre is, vagy jó ez az automatikus újratöltős verzió?



C:\Users\Bence\Documents\Arduino\sketch_feb24b\sketch_feb24b.ino:69:3: error: 'server' was not declared in this scope; did you mean 'Server'?

```
69 | server.send(200, "text/html", html);
| ^~~~~
| Server
```

exit status 1

Compilation error: expected unqualified-id before numeric constant

Ez a hiba valószínűleg **nevet ütköztet**, méghozzá a `server` azonosítóval. Mivel az ESP8266 könyvtárban már létezhet `Server` néven valami más definíció, a `server` változó neve ütközhet ezzel.

Megoldás: nevezd át `server`-t pl. `webServer`-re

Íme az **átnevezett, javított kód**:

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

const char* ssid = "Bwifi";
const char* password = "password";

const int trigPin = D1;
```

```
const int echoPin = D2;
ESP8266WebServer webServer(80); // <-- itt átneveztük!
long duration;
float distance;
unsigned long lastUpdate = 0;
const unsigned long updateInterval = 250;
void setup() {
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  WiFi.begin(ssid, password);
  Serial.print("Csatlakozás a WiFi-hez");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  Serial.print("IP-cím: ");
  Serial.println(WiFi.localIP());
  webServer.on("/", handleRoot);
webServer.begin();
                                     // <-- frissítve
// <-- frissítve</pre>
  Serial.println("Webszerver elindítva");
void loop() {
  webServer.handleClient();
                                     // <-- frissítve
  if (millis() - lastUpdate > updateInterval) {
    distance = measureDistance();
    lastUpdate = millis();
}
float measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  float dist = duration * 0.034 / 2;
  Serial.print("Távolság: ");
Serial.print(dist);
  Serial.println(" cm");
  return dist;
void handleRoot() {
   String html = "<!DOCTYPE html><html><head><meta http-equiv='refresh' content='0.25'>";
  html += "<meta charset='UTF-8'><title>Távolságmérés</title></head><body>";
  html += "<h1>Távolság: ";
  html += String(distance, 2);
html += " cm</hl></body></html>";
  webServer.send(200, "text/html", html); // <-- frissitve</pre>
```

💡 Összefoglalva:

- A `server` változónév ütközhetett beépített névvel vagy más könyvtárral.
- Átneveztük `webServer`-re, hogy elkerüljük az ütközést.

Most már fordulnia és működnie kell az ESP8266-on.

Ha szeretnéd, megmutatom, hogyan tudod AJAX-szal szépen frissíteni az értéket reloading nélkül is.



csak pontokat ir

Ha a soros porton vagy a weboldalon **csak pontokat (`.`)** jelennek meg, akkor az azt jelenti, hogy az **ESP8266 nem tud csatlakozni a WiFi hálózathoz**.

Mi történik?

A kódban van ez a rész:

```
while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
}
```

Ez addig ír ki pontokat, amíg **nem jön létre WiFi kapcsolat**. Tehát, ha **csak pontokat** látsz, akkor a **WiFi kapcsolat sikertelen**.

Ellenőrizendők:

1. III SSID és jelszó helyes?

Győződj meg róla, hogy ezt a két értéket helyesen adtad meg:

```
const char* ssid = "Bwifi";  // A hálózat neve
const char* password = "password"; // A jelszó
```

🔁 Ha bármelyik el van írva (pl. kis- és nagybetű nem stimmel), nem fog csatlakozni!

2. 🛑 A router engedi az ESP8266-ot csatlakozni?

- Bizonyos routerek tiltják az ismeretlen eszközöket.
- Próbáld ki a telefonod hotspotját:
 - Hozz létre hotspotot pl. `Bwifi` néven, `password` jelszóval.
 - o Így ki tudod zárni, hogy a routerrel van a gond.

3. 📈 A WiFi jelerősség gyenge?

- Ha a Wemos D1 Mini túl messze van a routertől, a kapcsolat nem fog sikerülni.
- Próbáld meg közelebb hozni.

4. → SSID rejtett?

- Ha a WiFi hálózat rejtett (hidden), az ESP8266 nem találja meg. Ebben az esetben:
 - o Állítsd be a routert, hogy a WiFi ne legyen rejtett (legalább ideiglenesen).

Teszt: Kapcsolódik-e?

Próbáld meg ideiglenesen ezt a tesztkódot:

```
#include <ESP8266WiFi.h>

const char* ssid = "Bwifi";
const char* password = "password";

void setup() {
    Serial.begin(115200);
    WiFi.begin(ssid, password);
    Serial.print("Csatlakozás a WiFi-hez");

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    Serial.println("WiFi kapcsolat létrejött!");
    Serial.println("WiFi kapcsolat létrejött!");
    Serial.println(WiFi.localIP());
}

void loop() {
}
```

Ha ez a tesztkód **még mindig csak pontokat ír**, akkor a hiba biztosan a hálózat beállításaiban van (nem szoftveres).

Ha szeretnéd, segítek **újra konfigurálni a routert** vagy **teszt-hálózatot beállítani**, csak írd meg, mire van lehetőséged.



csatlakozik a wifihez kap ipt de nem érem elC:\Users\Bence>ping 192.168.0.69

Pinging 192.168.0.69 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 192.168.0.69:
Távolság: 192.168.0.69
7.68 cm
Távolság: 192.168.0.69
7.68 cm



Távolság: 192.168.0.69

7.70 cm

Távolság: 192.168.0.69

7.70 cm

C:\Users\Bence>

Köszönöm a részletes infót — így látni, hogy az **ESP8266 csatlakozik a WiFi-re, IP-t kap (192.168.0.69)**, **soros porton működik a távolságmérés**, **de a weboldalt nem éred el**, és a **ping sem működik**. Ez alapján:

Mi lehet a gond? – Részletes hibakeresés

1. 🔥 A ping nem működik – ez normális ESP8266-nál

- Az ESP8266 alapból nem válaszol ping-re (ICMP-re).
- Ez nem jelenti azt, hogy a webserver nem működik.
- Tehát ha `ping`-re nincs válasz, attól még elérhető lehet a weboldal.

Próbáld elérni a böngészőből:

Menj a böngészőbe, és írd be:

срр

http://192.168.0.69/

Ez **vagy a helyi weboldalt jeleníti meg**, vagy hibaüzenetet ad.

2. 📛 A tűzfal blokkolhatja (PC-n vagy routeren)

➤ Windows tűzfal

- Ha a számítógéped privát helyett nyilvános hálózatként kezeli az otthoni WiFi-t, blokkolhatja a kérést.
- Próbáld ideiglenesen kikapcsolni a Windows tűzfalat vagy engedélyezd a `ping` és a `HTTP` kapcsolatokat.

➤ Router izolációs beállítás

- Néhány routerben van olyan beállítás, hogy kliensek elszigetelése:
 - Pl. "AP isolation" vagy "Client Isolation".
 - Ha ez be van kapcsolva, a számítógéped nem tud kommunikálni a mikrokontrollerrel, csak a netre.
 - 👉 Megoldás: Router admin felületén kapcsold ki a kliens izolációt, ha van ilyen opció.



3. 📲 Használj telefonos hotspotot (teszteléshez)

Ha nem vagy biztos a router beállításaiban, próbáld ki ezt:

• Telefonodon hozz létre hotspotot:

o SSID: `Bwifi`

Jelszó: `password`

- Csatlakoztasd rá az ESP8266-ot.
- Csatlakoztasd rá a laptopot is.
- Próbáld elérni újra: `http://<az ESP8266 IP-je>/`

4. Ellenőrzés: valóban elindult a webserver?

Tedd ezt a soros porton a kódba:

```
cpp
Serial.println("Webszerver fut: http://" + WiFi.localIP().toString());
```

Ez segít látni, hogy valóban beindult a szerver.

5. **Extra ellenőrzés: használj `curl` vagy `telnet`-et**

Parancssorból kipróbálhatod:

```
bash
curl http://192.168.0.69/
```

vagy

```
bash
telnet 192.168.0.69 80
```

Ha csatlakozik, akkor a szerver hallgat, csak valami más akadályozza a kapcsolatot (tűzfal, router izoláció stb.).

🕂 Összefoglalva

Probléma	Megoldás
Nincs ping válasz	Normális, ESP8266 nem válaszol ICMP-re
Weboldal nem elérhető	Próbáld böngészőben, ne csak `ping` -gel
Tűzfal, router blokkol	Ellenőrizd a kliens izolációt és tűzfalat
Hotspot teszt	Gyorsan kizárja router hibát

Ha szeretnéd, elküldöm a pontos lépéseket tűzfal beállításra vagy router izoláció kikapcsolására. Melyik routered van?



A webhely nem érhető el A(z) 192.168.0.69 túl hosszú ideje nem válaszol. Próbálja ki a következőket:

A kapcsolat ellenőrzése A proxy és a tűzfal ellenőrzése A Windows Hálózati diagnosztika futtatása ERR_CONNECTION_TIMED_OUT

Köszi, most már egyértelmű a helyzet:

Az ESP8266 csatlakozik a WiFi-re, IP-címet kap, de a webszerver nem érhető el, és timeoutol.

Ez **nem hálózati probléma** elsődlegesen, hanem az **ESP8266 oldalon valamiért nem fut a webserver**, vagy nem figyel port 80-on.



Gyorsteszt: egyszerű webszerver 1 oldallal

Futtasd le ezt az **ultra-minimál kódot**, ami csak a webszervert teszteli:

```
срр
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
const char* ssid = "Bwifi";
const char* password = "password";
ESP8266WebServer webServer(80);
void handleRoot() {
  webServer.send(200, "text/plain", "ESP8266 működik!");
void setup() {
  Serial.begin(115200);
 WiFi.begin(ssid, password);
  Serial.print("Csatlakozás WiFi-hez");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  Serial.println();
  Serial.print("Csatlakozva! IP: ");
  Serial.println(WiFi.localIP());
 webServer.on("/", handleRoot);
webServer.begin();
  Serial.println("Webszerver elindítva");
void loop() {
 webServer.handleClient();
```

Ezután:

- 1. Soros monitoron látni fogod az IP-címet (pl. `192.168.0.69`)
- 2. Menj a böngészőbe és írd be:

```
cpp
http://192.168.0.69/
```

3. Ha minden jó, ezt fogod látni:

```
nginx
ESP8266 működik!
```

Ha ez SEM működik:

Akkor 99%, hogy a gond **a következő lehet**:

1. 🛕 Túl sok soros port kiírás, túl gyakran

Az eredeti kódod 250 ms-onként ír soros portra + újratölt weblapot = ez túlterhelheti az ESP8266-ot.

2. / Tápellátás elégtelen

Az **ultrahangos szenzor (HC-SR04)** hirtelen áramigénye miatt instabillá válhat a mikrokontroller.

- Ha USB-ről laptopra van kötve, és HC-SR04 is tápot onnan vesz, könnyen lehet alultáplálás.
- Megoldás:
 - Külső 5V táp a HC-SR04-nek (pl. USB powerbank + 5V->3.3V átalakító)
 - o vagy csak próbáld le a szenzort a kódból.
- 3. Nincs `webServer.handleClient()` a `loop()`-ban

Ez is gyakori hiba — a webszerver csak akkor válaszol, ha ez a sor rendszeresen fut.

📌 Következő lépés

✓ Ha a fenti tesztkód működik, akkor a gond a távolságmérő kód vagy a frissítés túl sűrűsége. Ebben az esetben szívesen újraírom neked stabilabb, AJAX-alapú változatra.

Ha a **tesztkód SEM működik**, akkor hálózati szintű problémáról van szó (router, táp, firmware).

Szeretnéd, hogy küldjek egy végleges verziót stabil webes frissítéssel is?



A `duration` pontosabbá tételéhez az alábbi dolgokat lehet tenni:



1. Használj `micros()` alapú mérést

A `pulseIn()` függvény egyszerű és jól működik, de nem a legpontosabb. Ha nagyon precíz mérést szeretnél, válts át `micros()` használatára manuálisan:

```
срр
float measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Precízebb időmérés
  unsigned long startTime = micros();
while (digitalRead(echoPin) == LOW)
    if (micros() - startTime > 30000) return -1; // timeout
  unsigned long echoStart = micros();
while (digitalRead(echoPin) == HIGH) {
    if (micros() - echoStart > 30000) return -1; // timeout
  unsigned long echoEnd = micros();
  unsigned long duration = echoEnd - echoStart;
  float distance = duration * 0.0343 / 2.0; // cm
  return distance;
```

Ez pontosabb, mert a `pulseIn()` néha torzít hosszabb vagy túl gyors impulzusoknál. A fenti módszer:

- manuálisan figyeli a pin állapotát,
- mikroszekundum pontossággal számol,
- leáll timeouttal, ha nincs visszaverődő jel.

V

2. Hardveres pontosság javítása

Ha a szenzor hibázik, az nem mindig a kód hibája. Ezeket is ellenőrizd:

- **Tiszta 5V táp** (ne csak az ESP8266 USB-jéről!)
- Közvetlen, rövid vezetékek a HC-SR04-hez
- Kerüld a zajos környezetet vagy visszaverődéseket (túl közeli tárgy, textil, tükröződés)

3. Átlagolás: többszöri mérés simítása

Ha stabil értéket szeretnél (pl. 3 mérést átlagolni):

```
срр
float getStableDistance() {
  float sum = 0;
const int samples = 5;
  for (int i = 0; i < samples; i++) {
    sum += measureDistance();
    delay(10); // kis szünet a mérések közt
  return sum / samples;
```

💡 Végül

A HC-SR04 szenzor elméleti felbontása: kb. **0.3 cm**, de a valóságban 1–2 cm pontosság a jellemző. A fenti trükkökkel kihozható belőle a legjobb.

Ha szeretnéd, írok teljes, AJAX-os webes verziót pontos méréssel és átlagolással együtt. Csak szólj!



írj egy html css normálisan kinéző weblapot ahol a raw értéket is kiirja sötét desinggel

Rendben! Itt van egy letisztult, **sötét témájú**, modern HTML+CSS weblap, amely az ESP8266-ról érkező távolság (cm) és nyers `duration` értéket jeleníti meg.

A tartalom AJAX-szal frissül 250 ms-onként.

🚅 HTML + CSS + JavaScript (AJAX) – sötét dizájn

```
срр
const char MAIN_page[] PROGMEM = R"=====(
<!DOCTYPE html>
<html lang="hu">
<head>
  <meta charset="UTF-8">
  <title>Távolságmérő</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <style>
    body {
      background-color: #121212;
      color: #e0e0e0;
      font-family: 'Segoe UI', sans-serif;
      text-align: center;
      padding: 2rem;
    h1 {
      color: #bb86fc;
    .box {
      background: #1e1e1e;
      border-radius: 12px;
      padding: 1.5rem;
      margin: 1rem auto;
      width: 90%;
      max-width: 400px;
      box-shadow: 0 0 15px rgba(0,0,0,0.3);
```



```
.value {
       font-size: 2.5rem;
       color: #03dac6;
     .label {
      font-size: 1rem;
      color: #aaaaaa;
  </style>
</head>
<body>
  <h1>Ultrahangos Távolságmérés</h1>
  <div class="box">
    <div class="value" id="distance">--</div>
<div class="label">Távolság (cm)</div>
  </div>
  <div class="box">
    <div class="value" id="duration">--</div>
<div class="label">Raw Duration (µs)</div>
  </div>
  <script>
    setInterval(function() {
       fetch("/data")
         .then(response => response.json())
         .then(data => {
           document.getElementById("distance").textContent = data.distance.toFixed(2);
           document.getElementById("duration").textContent = data.duration;
    }, 250);
  </script>
</body>
</html>
)=====":
```

🔏 Backend (ESP8266 oldali kiszolgálás)

A `/data` endpoint JSON-t ad vissza:

```
void handleData() {
  long duration = pulseIn(echoPin, HIGH);
  float distance = duration * 0.0343 / 2.0;

  String json = "{\"duration\": " + String(duration) + ", \"distance\": " + String(distance, 2) + ":
  server.send(200, "application/json", json);
}
```

🟂 Fontos: a szerver beállítása

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

const char* ssid = "Bwifi";
const char* password = "password";

const int trigPin = D1;
```

```
const int echoPin = D2;
ESP8266WebServer server(80);
void handleRoot() {
  server.send_P(200, "text/html", MAIN_page);
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  Serial.println();
  Serial.println("WiFi kapcsolódva: ");
  Serial.println(WiFi.localIP());
  server.on("/", handleRoot);
server.on("/data", handleData);
  server.begin();
void loop() {
  server.handleClient();
```

Kipróbálás után

Látni fogod:

- szép sötét dizájn,
- élő távolság frissítés,
- nyers duration mikrosekundumban is látszik.

Ha szeretnél:

- grafikont is,
- mérések mentését memóriába,
- vagy log fájlba küldést szólj bátran!



írd le a teljes kódot

Persze! Itt van a teljes ESP8266 kód, amely:

- Csatlakozik a Wi-Fi hálózathoz (`Bwifi` jelszó: `password`)
- 250 ms-enként méri a távolságot a HC-SR04 segítségével
- Fut egy beépített webszerver

- Szép, sötét dizájnú weboldalt szolgál ki
- Kiírja a távolságot **centiméterben** és a nyers **duration (mikroszekundum)** értéket

🔽 Teljes Arduino-kód ESP8266-hoz

Csatlakoztasd a szenzort:

- `TRIG` D1 (GPIO5)
- `ECHO` D2 (GPIO4)

```
срр
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
const char* ssid = "Bwifi";
const char* password = "password";
const int trigPin = D1;
const int echoPin = D2;
ESP8266WebServer server(80);
const char MAIN page[] PROGMEM = R"=====(
<!DOCTYPE html>
<html lang="hu">
<head>
  <meta charset="UTF-8">
  <title>Távolságmérő</title> <meta name="viewport" content="width=device-width, initial-scale=1">
  <style>
    body {
      background-color: #121212;
      color: #e0e0e0;
      font-family: 'Segoe UI', sans-serif;
      text-align: center;
      padding: 2rem;
    h1 {
      color: #bb86fc;
    .box {
      background: #1e1e1e;
      border-radius: 12px;
      padding: 1.5rem;
margin: 1rem auto;
      width: 90%;
      max-width: 400px;
      box-shadow: 0 0 15px rgba(0,0,0,0.3);
    .value {
      font-size: 2.5rem;
      color: #03dac6;
    .label {
      font-size: 1rem;
      color: #aaaaaa;
  </style>
</head>
<body>
  <h1>Ultrahangos Távolságmérés</h1>
  <div class="box">
    <div class="value" id="distance">--</div>
```

```
<div class="label">Távolság (cm)</div>
  </div>
  <div class="box">
    <div class="value" id="duration">--</div>
    <div class="label">Raw Duration (μs)</div>
  </div>
  <script>
    setInterval(function() {
      fetch("/data")
         .then(response => response.json())
         .then(data => {
           document.getElementById("distance").textContent = data.distance.toFixed(2);
           document.getElementById("duration").textContent = data.duration;
         });
    }, 250);
  </script>
</body>
</html>
)=====";
// Mérési függvény
float distanceCM = 0.0;
long duration = 0;
void measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH, 30000); // max 30ms = ~5m
  distanceCM = duration * 0.0343 / 2.0;
}
void handleRoot() {
  server.send_P(200, "text/html", MAIN_page);
void handleData()
  measureDistance();
String json = "{\"duration\": " + String(duration) + ", \"distance\": " + String(distanceCM, 2) +
  server.send(200, "application/json", json);
void setup() {
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
  WiFi.begin(ssid, password);
  Serial.print("Csatlakozás WiFi-hez");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  Serial.println();
  Serial.print("Csatlakozva! IP-cím: ");
  Serial.println(WiFi.localIP());
  server.on("/", handleRoot);
server.on("/data", handleData);
server.begin();
  Serial.println("Webszerver elindult.");
void loop() {
```



```
server.handleClient();
}
```

៉ុក Fontos tanácsok

- Az ESP8266 nem válaszol `ping`-re a weboldalt közvetlen IP-vel érd el:
 - \rightarrow `http://192.168.0.xx/` (ahol az IP-t a soros monitoron látod)
- A `pulseIn(..., timeout)` használat védi a mikrokontrollert lefagyástól
- Ha instabil, győződj meg róla, hogy a HC-SR04 megfelelő 5V tápot kap, és esetleg tegyél 1k/2k osztót az `ECH0`-ra

Ha szeretnél grafikont vagy extra funkciókat (pl. max / min távolság naplózása), szívesen segítek!



JEDLIKBOTS

Jedlik Ányos Technikum Robotika Csapata log_____

Innovatív robotika a Jedlikből

Üdvözlünk a JedlikBots oldalán! Mi vagyunk a Győri Jedlik Ányos Technikum lelkes szoftverfejlesztő diákcsapata, akik a robotika világában alkotnak. Csapatunk szenvedélyesen dolgozik innovatív robotikai megoldásokon, és büszkén mutatjuk be az elmúlt évek során elért eredményeinket.

Célunk, hogy ne csak megvalósítsuk elképzeléseinket, hanem folyamatosan fejlesszük készségeinket és tudásunkat, miközben gyakorlati tapasztalatot szerzünk a szoftverfejlesztés, a mechanikai tervezés és az elektronika területén.

Jelenleg egy professzionális versenyzésre és komplex technológiai demonstrációkra is alkalmas új robot fejlesztésén dolgozunk, és keressük azokat a támogatókat, akik hisznek a fiatal technológiai tehetségekben és segítenének álmaink megvalósításában!

Csapattagok Nagy Bence Nagy Bence Szoftverfejlesztő

Szakterület: szenzoradatok feldolgozása, robotvizualizáció

Bence feladata a robotok érzékelőinek adatgyűjtése és valós idejű feldolgozása. Kiváló 3D-modellező, robotjaink vizuális megjelenítésén és szimulációs környezetének fejlesztésén dolgozik.

Nagy Bence Szitás László Szoftverfejlesztő

Szakterület: felhasználói felületek, projektmenedzsment



László fejleszti a robotjaink vezérlő felületeit és webes alkalmazásait. Innovatív megoldásaival biztosítja, hogy bárki könnyen kommunikálhasson és vezérelhesse robotjainkat. Emellett ő tartja kézben a projektek ütemezését és koordinálását.

Nagy Bence Schmidt Nándor Szoftverfejlesztő

Szakterület: beágyazott rendszerek, elektronikai tervezés

Nándor a robotok "agya" mögött álló beágyazott rendszerek szakértője. Az alacsony szintű programozás és firmware fejlesztés mestere, aki biztosítja, hogy robotjaink gyorsan és hatékonyan reagáljanak.

Buda Bálint Buda Bálint Szoftverfejlesztő

Szakterület: robotvezérlés, Gépi látás

Bálint felelős a robotok vezetékes és vezeték nélküli kommunikációjáért, valamint a gépi tanulási algoritmusok implementálásáért, amelyek segítenek robotjainknak alkalmazkodni a változó környezeti feltételekhez.

Projektjeink

WRO 2025 regionális döntő Budapest

Üdvözlünk a JedlikBots oldalán! Mi vagyunk a Győri Jedlik Ányos Technikum lelkes szoftverfejlesztő diákcsapata, akik a robotika világában alkotnak. Csapatunk szenvedélyesen dolgozik innovatív robotikai megoldásokon, hogy versenyeken érhessünk el sikereket.

A Budapesti WRO (World Robot Olympiad) regionális döntőjéig jutottunk el EV3-as Mindstorms robotunkkal, ahol sikeresen 2. helyezést értünk el, ezzel kvalifikálva magunkat a nemzeti döntőre. Jelenleg erre a kihívásra készülünk, de már a jövő évi céljainkat is kitűztük.

Jövőre ugyanezen a versenyen szeretnénk elérni a világdöntőre való kijutást, hogy képviselhessük Magyarországot a nemzetközi mezőnyben. A lenti videón látható jelenlegi robotunk és a használt technológia sajnos nem elegendő ahhoz, hogy reális esélyünk legyen a világdöntőn való eredményes szereplésre, ezért szeretnénk egy fejlettebb robotot építeni a következő évi megmérettetésre.

Jelenleg támogatókat keresünk, akik hisznek a fiatal technológiai tehetségekben és segítenének álmaink megvalósításában. A jelenlegi robotunk bemutatóját az alábbi videón tekintheted meg!

Jelenlegi projektünk: ProBot X1

A ProBot X1 egy multifunkcionális, moduláris robot platformunk, amellyel nemzetközi versenyeken szeretnénk részt venni. A robot jelenlegi fejlesztési szakaszában vagyunk, és olyan támogatókat keresünk, akik a fejlesztéshez szükséges alkatrészekkel, eszközökkel tudnák segíteni a munkánkat.

Fejlesztési előrehaladás:



10%

Műszaki tervek:

Beépített mélységérzékelő kamerák a térbeli tájékozódáshoz

Fejlett gépi tanulási algoritmusok az objektumfelismeréshez

Moduláris kialakítás, könnyen cserélhető alkatrészekkel

Akkumulátor üzemidő: 2+ óra aktív használat mellett

Több irányba fordítható kamera, gépi látással feldolgozva adatait

Professzionális robot, bármilyen feladatra

Eszközigény – támogatással segíthető alkatrészek:

Raspberry Pi bármely változata (akár Compute Module is)

Léptetőmotorok, motorvezérlők, kisebb villanymotorok

Szenzorok: színérzékelő, gyorsulásmérő, giroszkóp, kameramodulok

Próbapanelek, jumperkábelek, USB-TTL konverterek

STM-alapú mikrokontroller panelek

3D nyomtató filament (1.75 mm)

Érintőkijelzők (akár hibás, ha javítható)

Logikai analizátor a hibakereséshez

RGB LED-ek állapotjelzéshez

Bármilyen támogatás – akár egyetlen eszközzel – nagy segítséget jelent számunkra a projekt sikeres befejezéséhez és a versenyeken való részvételhez.

Eddigi eredményeink

2022. március

Robotprogramozó Országos Csapatverseny - 2. helyezés

Az Országos Robotprogramozó csapatversenyen a csapatunk 2. helyezést ért el. Ez volt az első országos versenyünk, amely bebizonyította, hogy csapatunk képes versenyképes robotokat építeni.

2023. március

Robotprogramozó Országos Csapatverseny - 1. helyezés

Ismét részt vettünk az Országos Robotprogramozó csapatversenyen, ahol ebben az évben a csapatunk már 1. helyezést ért el. Bebizonyítottuk, hogy képesek vagyunk az országos csúcsra törni.

2025. május

WRO 2025 Regionális Döntő Budapest - 2. helyezés

Az első WRO versenyünk, ahol a budapesti regionális döntőben 2. helyezést értünk el. Így bejutottunk a WRO országos döntőjébe. Jövőre ugyanezen a versenyen az új professzionális robottal a világdöntő a cél.

Támogasd a jövő mérnökeit!

Segítsd csapatunkat, hogy megvalósíthassuk álmainkat és innovatív projektjeinket!

Anyagi támogatás

Projektjeink megvalósításához folyamatosan eszközökre, alkatrészekre és anyagokra van szükségünk. Az anyagi támogatással közelebb kerülünk céljainkhoz.

Mire költjük: Elektronikai komponensek, mikrokontrollerek, mechanikai alkatrészek, szerszámok

Eszköz adományozás

Ha rendelkezel olyan használaton kívüli eszközökkel, amelyek segíthetik a munkánkat – például mikrokontrollerek, szenzorok, motorok –, hálásan fogadjuk!



Szakmai támogatás

Mentorálással, tanácsadással, vagy akár csak ötletekkel is sokat segíthetsz nekünk. Szívesen együttműködünk tapasztalt szakemberekkel, fejlesztőkkel. irj descripciont githubra ezek alapján

Íme egy profi, figyelemfelkeltő GitHub **repository description** magyar és angol nyelven, a megadott szöveg alapján – tökéletesen illik egy nyílt forráskódú vagy bemutató projekthez, például a *ProBot X1*-hez vagy a JedlikBots csapat arculatához:

Magyar – README leírás / GitHub description

JedlikBots - A jövő robotikai mérnökei Győrből

Üdvözlünk a JedlikBots GitHub oldalán! 🦈 🤖

Mi vagyunk a Győri Jedlik Ányos Technikum szoftverfejlesztő és robotika iránt elkötelezett csapata. Célunk, hogy innovatív, valós problémákra épülő robotikai megoldásokat hozzunk létre, miközben folyamatosan bővítjük tudásunkat a szoftverfejlesztés, elektronika és gépészet területén.

📌 Fő projekt: ProBot X1

Egy professzionális, moduláris robotplatform, mellyel a nemzetközi WRO versenyekre készülünk. A robot fejlett gépi látással, tanuló algoritmusokkal és cserélhető moduláris alkatrészekkel rendelkezik.

o Küldetésünk

- Versenyrobotok tervezése és fejlesztése (WRO, országos megmérettetések)
- Szoftveres és beágyazott rendszerek integrálása
- Oktatási célú nyílt forrású projektek bemutatása
- Támogatók és mentorok bevonása a magyar műszaki utánpótlás erősítéséért

🖍 Eszközigény & támogatás

Támogass minket eszközzel vagy szakmai tanáccsal, hogy elérhessük a világdöntőt és inspiráljuk a jövő mérnökeit! Részletekért görgess lejjebb a README-ben vagy látogasd meg weboldalunkat.

English – GitHub Repository Description

JedlikBots - Young Engineers from Hungary Building the Future of Robotics

Welcome to the official GitHub page of **JedlikBots**, a high school robotics team from Győr, Hungary —. We are passionate software developers and hardware tinkerers working on competitive and educational robotics projects.

Main Project: ProBot X1

A modular, AI-powered robot platform designed for international robotics competitions such as the WRO. Features include 3D vision, smart object recognition, and swappable hardware components.

Our Mission



- Build competitive and educational robots from scratch
- Master embedded systems and real-time data processing
- Share open-source solutions and promote youth engineering
- Seek sponsors and mentors to help us grow professionally

🤝 How You Can Help

We're looking for supporters, mentors, and hardware donations (sensors, microcontrollers, motors, etc.) to bring our robots to life and represent Hungary on the world stage. More info in the full README.

Ha szeretnéd, csinálhatok egy látványos `**README.md**`-et is GitHubra képekkel, badge-ekkel, technológiákkal. Mondhatod, mi legyen benne még!



platform io ban mi a neve board nak?

A Wemos D1 Mini-től és az ESP8266-tól függően a PlatformIO-ban használt board neve a következő:

✓ PlatformIO board neve Wemos D1 Mini-hez (ESP8266):

```
ini
board = d1_mini
```

% Teljes `platformio.ini` példa:

```
ini

[env:dl_mini]
platform = espressif8266
board = dl_mini
framework = arduino
monitor_speed = 115200
upload_speed = 921600
```

Ha ESP32-es lenne, akkor például `board = esp32dev` lenne, de mivel jelezted, hogy ESP8266 van, ez a megfelelő.

Ha szeretnéd, le is generálok neked egy teljes PlatformIO-projektet induláshoz (pl. `main.cpp` + `platformio.ini` + könyvtárak listája). Szólj!