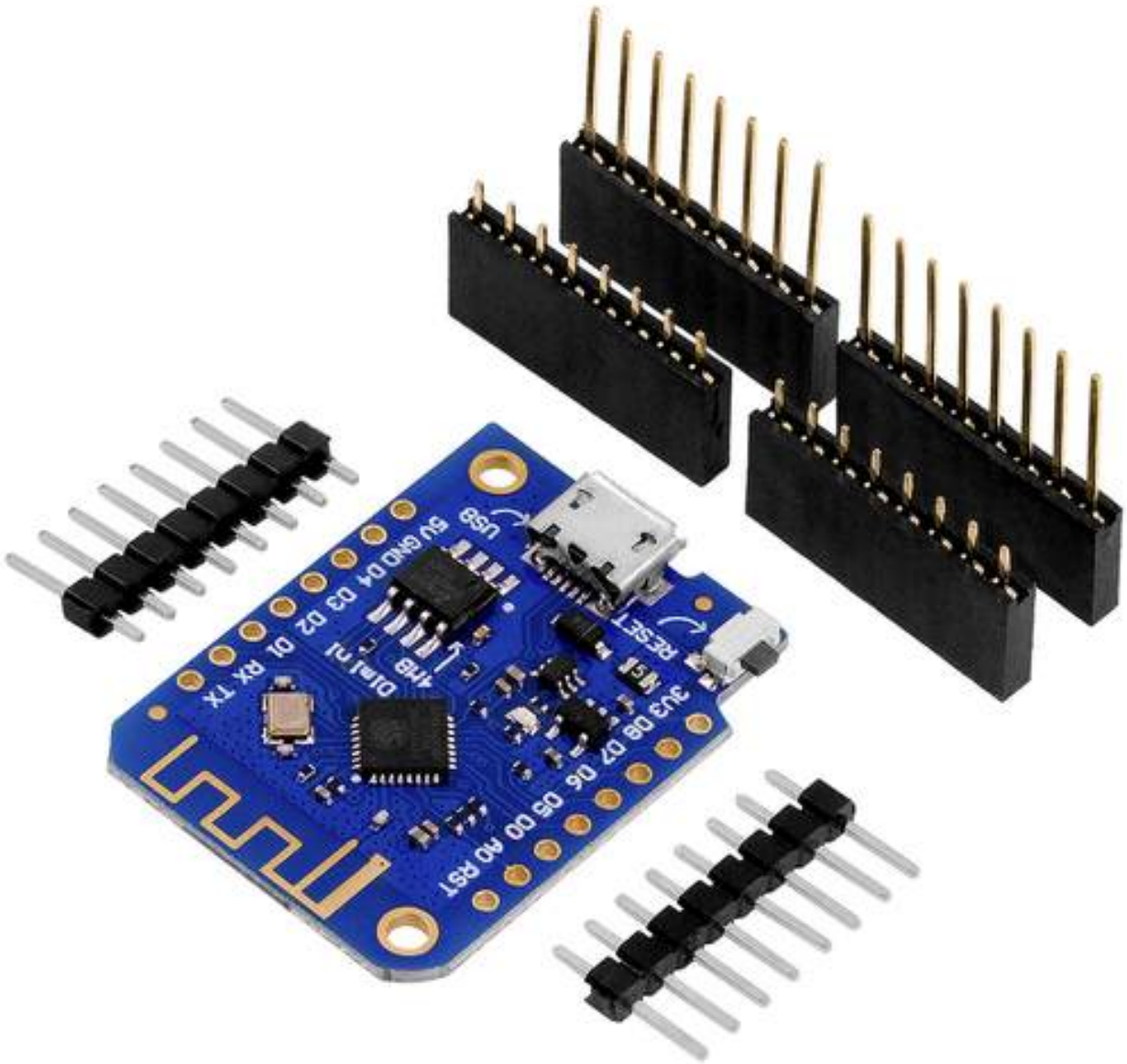
Willkommen!

Vielen Dank, dass sie sich für unser *D1 Mini ESP8266-12F v3 Modul* von *AZ-Delivery* entschieden haben. In den nachfolgenden Seiten werden wir Ihnen erklären wie Sie das Gerät einrichten und nutzen können.

Viel Spaß!



Inhaltsverzeichnis

Einführung....3
Technische Daten des ESP8266....4
Das D1 Mini Modul....5
Technische Daten....5
Digitale E/A Pins....6
PWM....7
Analoger Eingang....7
Seriell....7
I2C....8
SPI....8
Pinbelegung....9
Das D1 Mini Modul - Software....10
Digitale E/A Pins....10
Analoger Eingangspin....11
Serielle Kommunikation....12
Die I2C- und SPI-Schnittstelle....12
Gemeinsame Nutzung der CPU-Zeit mit dem RF-Teil....13
Wie man die Arduino IDE einrichtet....14
Der D1 Mini mit der Arduino IDE....18
Blink + PWM + Serielle Sketch-Beispiele....22
Blink Sketch-Beispiel....22
Software PWM Sketch-Beispiel....23
Serielle Kommunikation Sketch-Beispiel....25

Einführung

Das ESP8266-Modul ist ein "System on a Chip" (*SoC*). Es besteht aus einem Tensilica L106 32-Bit-Mikrocontroller und einem Wifi-Transceiver. Es hat 11 GPIO-Pins (General Purpose Input/Output) und einen analogen Eingang. Das bedeutet, dass Sie ihn wie jeden Arduino oder anderen Mikrocontroller programmieren können. Das Beste an dem ESP8266 ist, dass Sie mit ihm über Wifi kommunizieren können, so dass Sie mit ihm eine Verbindung zu Ihrem Wifi-Netzwerk herstellen, sich mit dem Internet verbinden, einen Webserver mit echten Webseiten hosten und Ihr Smartphone mit ihm verbinden können, usw. Er unterstützt Netzwerkprotokolle, wie Wi-Fi, TCP, UDP, HTTP, DNS, etc.

Das AZ-Delivery D1 Mini-Modul ist ein Entwicklungs-Board, das auf dem ESP8266-Chip basiert. Es hat 11 digitale Ein-/Ausgangspins und einen analogen E-Pin. Alle digitalen E/A-Pins sind durch Software Interrupt-, PWM-, I2C- und 1-Draht-fähig. Die analoge Eingangsspannung liegt zwischen *0V* und *3,3V* DC. Das Modul verwendet einen microUSB-Port und den CH340G-Chip mit einer Programmierschaltung zur Programmierung. Außerdem fungiert der microUSB-Port als Stromversorgung für das Modul.

Es gibt verschiedene Möglichkeiten, das D1-Mini-Modul zu programmieren. Neben der Arduino IDE gibt es weitere Möglichkeiten, das D1-Minimodul zu programmieren (offizielles ESP SDK für C-Programmierung, Lua-Interpreter, MicroPython-Firmware, sind nur einige von vielen).

Technische Daten des ESP8266

- 802.11 b/g/n
- Integrierte stromsparende 32-Bit-MCU
- Integrierte 10-bit ADC
- Integrierter TCP/IP Protokoll-Stapel
- Integrierter TR-Schalter, Balun, LNA, Leistungsverstärker und Anpassungsnetzwerk
- Integrierte PLL, Regler und Leistungsverwaltungseinheiten
- Unterstützte Antennendiversität
- Wi-Fi 2.4 GHz, unterstützt WPA/WPA2
- Unterstützt STA/AP/STA+AP Betriebsmodi
- Unterstützt Smart Link-Funktionen für Android- und iOS-Geräte
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO
- STBC, 1x1 MIMO, 2x1 MIMO
- A-MPDU & A-MSDU-Aggregation und 0,4s Schutzintervall
- Tiefschlafleistung <10uA, Abschaltleckstrom < 5uA
- Aufwecken und Übertragen von Paketen in < 2ms
- Standby-Leistungsaufnahme von < 1,0 mW (DTIM3)
- +20dBm Ausgangsleistung im 802.11b-Modus
- Betriebstemperaturbereich: -40 °C ~ 125 °C

Das D1 Mini Modul

Das D1 Mini Modul wird ungelötet mit einem Paar achtstiftiger Stiftleisten, einem Paar achtstiftiger Buchsenleisten und einem Paar achtstiftiger Buchsenleisten mit extra langen Beinen (s. Titelbild).

Technische Daten

» Betriebsspannung:	3.3V
» Hauptchip:	ESP8266EX
» Taktfrequenz:	80MHz/160MHz
» Flash-Speicher:	4MB
» Digitale E/A-Pins:	11
» Analoge Eingangspins:	1
» Analoger Eingangsspannungsbereich:	0V bis 3.3V DC
» USB-Anschluss:	Micro USB
» USB-Chip:	CH340C
» On-board LED:	verbunden mit dem GPIO2 Pin
» Dimensionen:	25 x 35 x 6mm [0.98 x 1.4 x 0.24in]
» Max. Spannung pro digitalen E/A Pin:	12mA

Digitale E/A Pins

Genau wie der Arduino verfügt das D1 Mini v3 über digitale Ein-/Ausgangspins oder GPIO - General Purpose Input/Output Pins. Wie der Name schon sagt, können sie als digitale Eingänge verwendet werden, um eine digitale Spannung zu lesen, oder als digitale Ausgänge, um entweder 0V (Senkenstrom) oder 3,3V (Quellenstrom) auszugeben.

Das D1 Mini v3 hat einen Mikrocontroller der mit einem Spannungsbereicch von 0V-3,3V arbeitet.

Die maximale Spannung, die von einem einem GPIO Pin bezogen werden kann beträgt 12mA!

Hinweis: Die Pins sind nicht 5V-tolerant, das Anlegen von mehr als 3,6V PRO PIN führt zur Zerstörung des Chips!

GPIO1 und *GPIO3* werden als *TX* und *RX* der seriellen Schnittstelle (UART) verwendet, so dass sie in den meisten Fällen nicht als normale E/A beim Senden/Empfangen von seriellen Daten verwendet werden kann.

Das D1 Mini v3 Modul hat eine integrierte LED, die mit dem *GPIO2* Pin verbunden ist.

PWM

Im Gegensatz zu den meisten Atmel-Chips (Arduino) unterstützt das D1 Mini Modul keine Hardware-PWM, jedoch wird Software-PWM auf allen digitalen Pins unterstützt. Der Standard-PWM-Bereich beträgt 10 Bit bei 1kHz, aber dies kann geändert werden (bis zu >14 Bit bei 1kHz).

Analoger Eingang

Das D1 Mini v3 Modul hat einen analogen Eingang mit einem eingangsbereich von 0V-3,0V. Sollte mehr zugeführt werden, könnte dies den Chip beschädigen. Der ADC (Analog-Digital-Wandler) hat eine Auflösung von 10 Bit.

Seriell

Das D1 Mini Modul verfügt über zwei Hardware-*UARTS* (serielle Schnittstellen):
UART0 an den Pins 1 und 3 (*TX0* bzw. *RX0*) und *UART1* an den Pins 2 und 8 (*TX1* bzw. *RX1*), jedoch wird der GPIO8 zum Anschluss des Flash-Chips verwendet. Das bedeutet, dass *UART1* nur Daten übertragen kann. In den meisten Fällen ist ein *UART*-Port mehr als ausreichend.

UART0 hat auch eine Hardware-Flusskontrolle an den Pins 15 und 13 (*RTS0* bzw. *CTS0*). Diese beiden Pins können auch alternativ als *TX0*- und *RX0*-Pins verwendet werden.

I2C

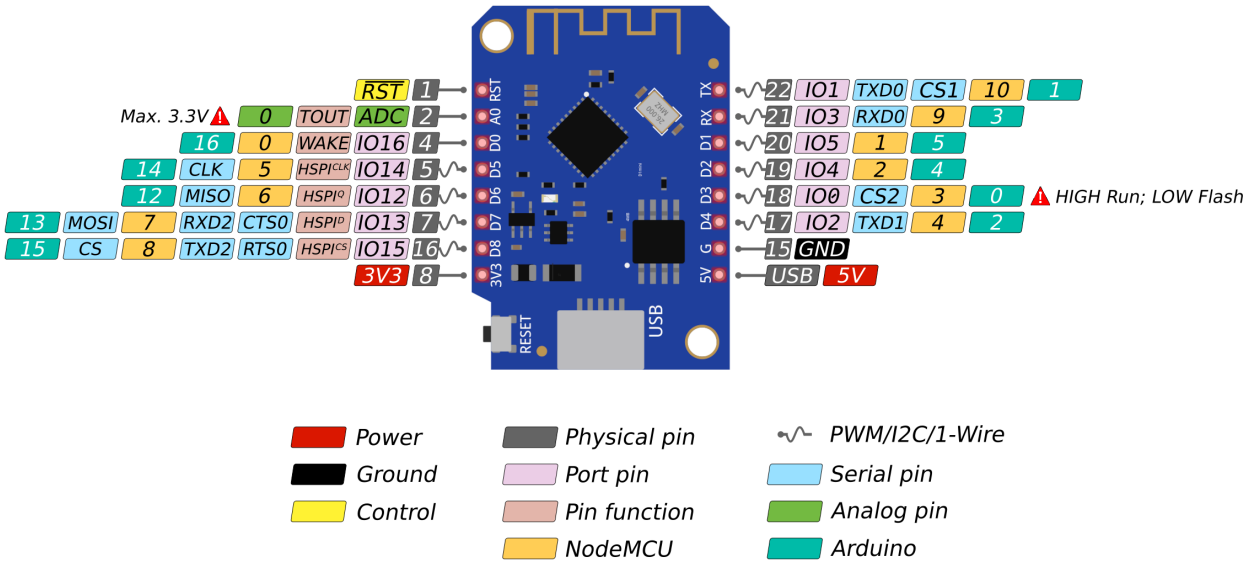
Das D1 Mini Modul v3 verfügt über keine Hardware TWI (Two Wire Interface), sondern ist in Software implementiert. Das bedeutet, dass Sie so ziemlich jede zwei digitalen Pins verwenden können. Standardmäßig verwendet die I²C-Bibliothek Pin 4 als *SDA* und Pin 5 als *SCL*. (Das Datenblatt spezifiziert *GPIO2* als *SDA* und *GPIO14* als *SCL*) Die maximale Geschwindigkeit beträgt etwa *450kHz*.

SPI

Das D1 Mini Modul v3 verfügt über einen *SPI*-Anschluss, der dem Benutzer zur Verfügung steht und als *HSPI* bezeichnet wird. Es kann sowohl im Slave- als auch im Master-Modus (in Software!) verwendet werden.
Es verwendet:
- *GPIO14* als clock – *CLK*,
- *GPIO12* als *MISO*,
- *GPIO13* als *MOSI* und
- *GPIO15* als Slave Select - *SS*.

Pinbelegung

Das D1 Mini v3 Modul hat zwei Reichen mit jeweils 8 Pins (16 insgesamt). Die Pinbelegung des Moduls sieht wie folgt aus:



Hinweis: Das D1 Mini v3 hat einen 5V-Spannungsausgangspin, um die angeschlossenen externen elektronischen Geräte zu versorgen, aber er ist nicht 5V-tolerant! Der 5V-Pin ist ein Leistungsausgangs-Pin, der die Leistung vom Mikro-USB-Port ausgibt.

Hinweis: Das D1 Mini v3 hat auch einen 3,3V-Spannungsausgangspin zur Versorgung externer elektronischer Geräte. Dieser wird vom 3,3V-Spannungsregler auf dem Board auf 3,3V begrenzt.

Der Großteil der Mikrocontroller-Funktionalität des ESP verwendet genau dieselbe Syntax wie ein normaler Arduino, was den Einstieg wirklich einfach macht.

Digitale E/A Pins

Genau wie bei einem normalen Arduino können Sie die Funktion eines Pins mit folgender Codezeile einstellen:

```
pinMode(pin, mode)
```

wobei "pin" die *GPIO*-Nummer ist und "mode" entweder *INPUT* (was die Voreinstellung ist), *OUTPUT* oder *INPUT_PULLUP* sein kann, um die eingebauten Pull-up-Widerstände für GPIO 0-15 zu aktivieren. Um den Pulldown-Widerstand für *GPIO16* zu aktivieren, müssen Sie *INPUT_PULLDOWN_16* verwenden.

Um einen Ausgangspin auf *HIGH* (3,3V) oder *LOW* (0V) zu setzen, nutzen Sie folgende Codezeile:

```
digitalWrite(pin, value)
```

wobei "pin" der digitale Pin ist und "value" entweder 1 oder 0 (*HIGH* und *LOW*).

Um einen Eingang zu lesen, nutzen Sie *digitalRead(pin)*

Um die PWM an einem bestimmten Pin zu aktivieren, verwenden Sie:
`analogWrite(pin, value)`
wobei "*pin*" der digitale Pin und "*value*" eine Zahl zwischen *0* und *1023* ist.

Um den Bereich des PWM-Ausgangs zu ändern, verwenden Sie: `analogWriteRange(new_range)`

Die Frequenz kann durch Verwendung folgender Codezeile verändert werden:
`analogWriteFreq(new_frequency)`
Wobei "*new_frequency*" sollte zwischen *100Hz* und *1000Hz* liegen.

Analoger Eingangspin

Genau wie bei einem Arduino können Sie *analogRead*(A0) verwenden, um die analoge Spannung am Analogeingang zu erhalten (0 = 0V, 1023 = 1,0V).

Das D1 Mini Modul v3 kann den ADC auch zur Messung der Versorgungsspannung (VCC) verwenden. Dazu fügen Sie *ADC_MODE(ADC_VCC)* oben in Ihre Skizze ein, und verwenden Sie *ESP.getVcc()*, um die Spannung tatsächlich zu erhalten.

NOTE: Wenn ein analoger Pin zum Auslesen der Versorgungsspannung verwendet wird, können Sie an den analogen Pin nichts anderes anschließen!

Serielle Kommunikation

Um *UART0* (*TX = GPIO1, RX = GPIO3*) zu verwenden, können Sie das Serial-Objekt wie bei einem Arduino verwenden: *`Serial.begin(baud_reate)`*.

Um die alternativen Pins (*TX = GPIO15, RX = GPIO13*) zu verwenden, verwenden Sie *`Serial.swap()`*
nach *`Serial.begin()`*.

Um *UART1* (*TX = GPIO2*) zu verwenden, verwenden Sie das Serial1-Objekt.

Hinweis: Alle Arduino Stream-Funktionen, wie *`read()`*, *`write()`*, *`print()`*, *`println()`* , ... werden ebenfalls unterstützt.

I2C- and SPI-Schnittstellen

Sie können einfach die Standardsyntax der Arduino-Bibliothek verwenden.

Gemeinsame Nutzung der CPU-Zeit mit dem RF-Teil

Eine Sache, die Sie beim Schreiben von Programmen für das D1 Mini Modul (ESP8266) beachten müssen, ist, dass Ihr Sketch Ressourcen (CPU-Zeit und Speicher) mit dem WLAN- und TCP-Stack (der Software, die im Hintergrund läuft und alle WLAN- und IP-Verbindungen handhabt) teilen muss. Wenn die Ausführung Ihres Codes zu lange dauert und Sie die TCP-Stacks nicht machen lassen, könnte es zu einem Absturz kommen oder Sie könnten Daten verlieren. Es ist am besten, die Ausführungszeit Ihres Codes unter ein paar hundert Millisekunden zu halten. Jedes Mal, wenn der Hauptloop wiederholt wird, gibt Ihr Sketch dem WLAN und TCP zugunsten nach, um alle WLAN- und TCP-Anfragen zu bearbeiten. Wenn Ihr loop länger dauert, müssen Sie den Wifi/TCP-Stacks explizit CPU-Zeit geben, indem Sie include `delay(0)` oder `yield()` verwenden. Wenn Sie das nicht tun, funktioniert die Netzwerkkommunikation nicht wie erwartet, und wenn sie länger als *3* Sekunden dauert, setzt der soft WDT (Watchdog-Timer) das ESP zurück. Wenn der soft WDT deaktiviert ist, setzt der Hardware-WDT nach etwas mehr als *8* Sekunden den Chip zurück. Aus der Sicht eines Mikrocontrollers sind *3* Sekunden jedoch eine sehr lange Zeit (*240* Millionen Taktzyklen), so dass Sie davon nicht betroffen sind, es sei denn, Sie führen ein extremes Zahlencrunching durch oder senden extrem lange Strings über die serielle Schnittstelle. Denken Sie einfach daran, dass Sie `yield()` innerhalb Ihrer for- oder while-loops hinzufügen, die länger als, sagen wir, *100ms* dauern könnten.

Wie man die Arduino-IDE einrichtet

Falls die Arduino-IDE nicht installiert ist, folgen Sie dem *link* und laden Sie die Installationsdatei für das Betriebssystem Ihrer Wahl herunter.

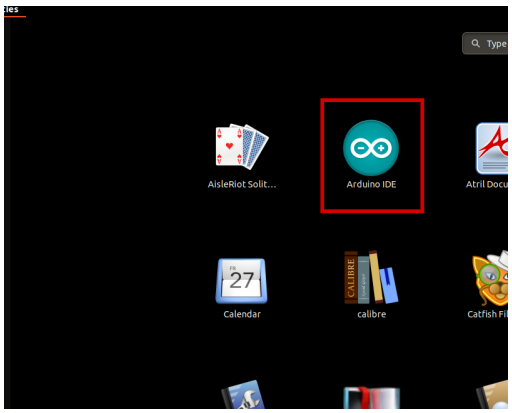
Für `Windows` Benutzer: Doppelklicken Sie auf die heruntergeladene `.exel`-Datei und folgen Sie den Anweisungen im Installationsfenster.

Für *Linux* Benutzer, laden Sie eine Datei mit der Erweiterung *.tar.xz* herunter, die extrahiert werden muss. Wenn sie extrahiert ist, gehen Sie in das extrahierte Verzeichnis und öffnen Sie das Terminal in diesem Verzeichnis. Zwei *.sh* Skripte müssen ausgeführt werden, das erste namens *arduino-linux-setup.sh* und das zweite heißt *install.sh.*

Um das erste Skript im Terminal auszuführen, öffnen Sie das Terminal im extrahierten Ordner und führen Sie den folgenden Befehl aus:
sh arduino-linux-setup.sh user_name
user_name - ist der Name eines Superusers im Linux-Betriebssystem. Ein Passwort für den Superuser muss beim Start des Befehls eingegeben werden. Warten Sie einige Minuten, bis das Skript vollständig abgeschlossen ist.

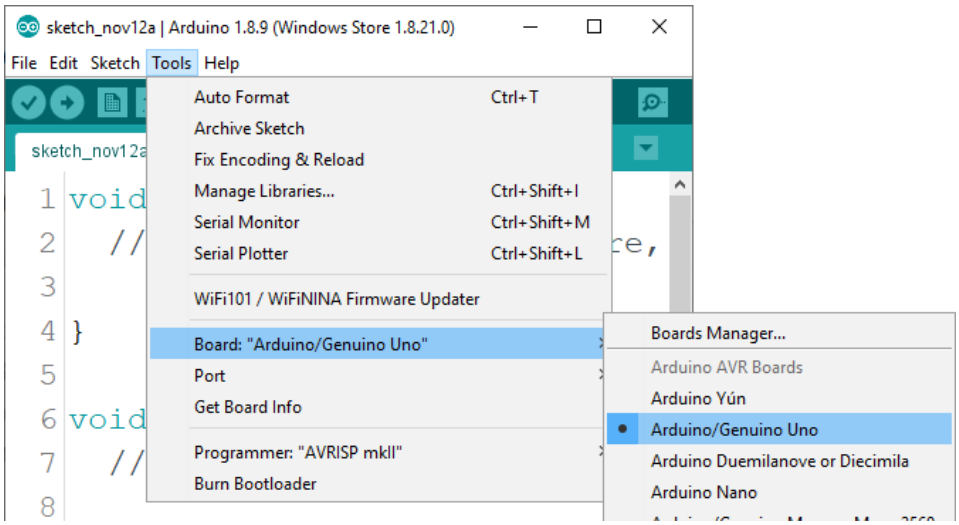
Das zweite Skript mit der Bezeichnung *install.sh*-Skript muss nach der Installation des ersten Skripts verwendet werden. Führen Sie den folgenden Befehl im Terminal (extrahiertes Verzeichnis) aus: **sh install.sh**

Nach der Installation dieser Skripte gehen Sie zu *All Apps*, wo die *Arduino-IDE* installiert ist.



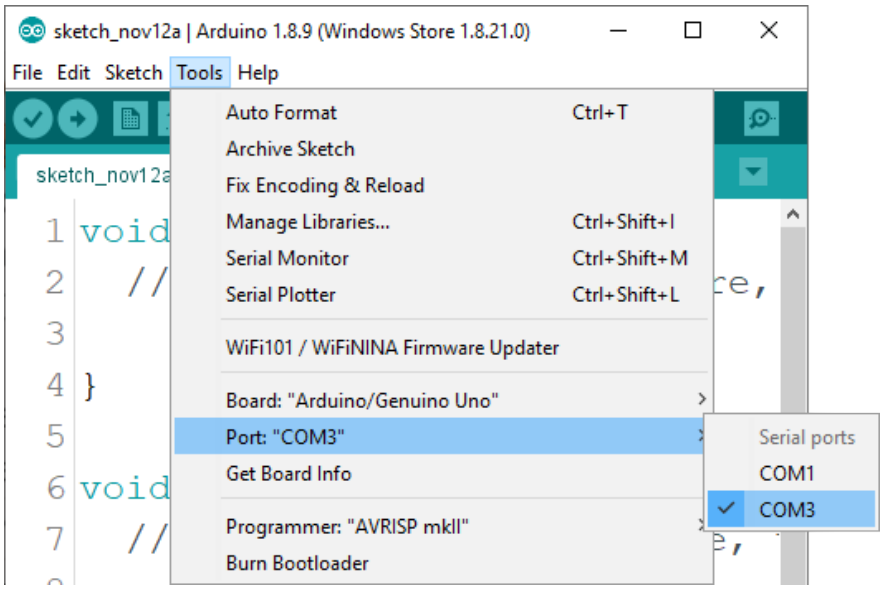
Fast alle Betriebssysteme werden mit einem vorinstallierten Texteditor ausgeliefert (z.B. *Windows* mit *Notepad*, *Linux* Ubuntu mit *Gedit*, *Linux Raspbian* mit *Leafpad* usw.). Alle diese Texteditoren sind für den Zweck des eBooks vollkommen in Ordnung.

Zunächst ist zu prüfen, ob Ihr PC ein Arduino-Board erkennen kann. Öffnen Sie die frisch installierte Arduino-IDE, und gehen Sie zu:
Tools > Board > {your board name here}
{your board name here} sollte der *Arduino/Genuino Uno* sein, wie es auf dem folgenden Bild zu sehen kann:



Der Port, an den das Arduino-Board angeschlossen ist, muss ausgewählt werden. Gehe zu: *Tools > Port > {port name goes here}* und wenn das Arduino-Board an den USB-Port angeschlossen ist, ist der Portname im Drop-down Menü auf dem vorherigen Bild zu sehen.

Wenn die Arduino-IDE unter Windows verwendet wird, lauten die Portnamen wie folgt:

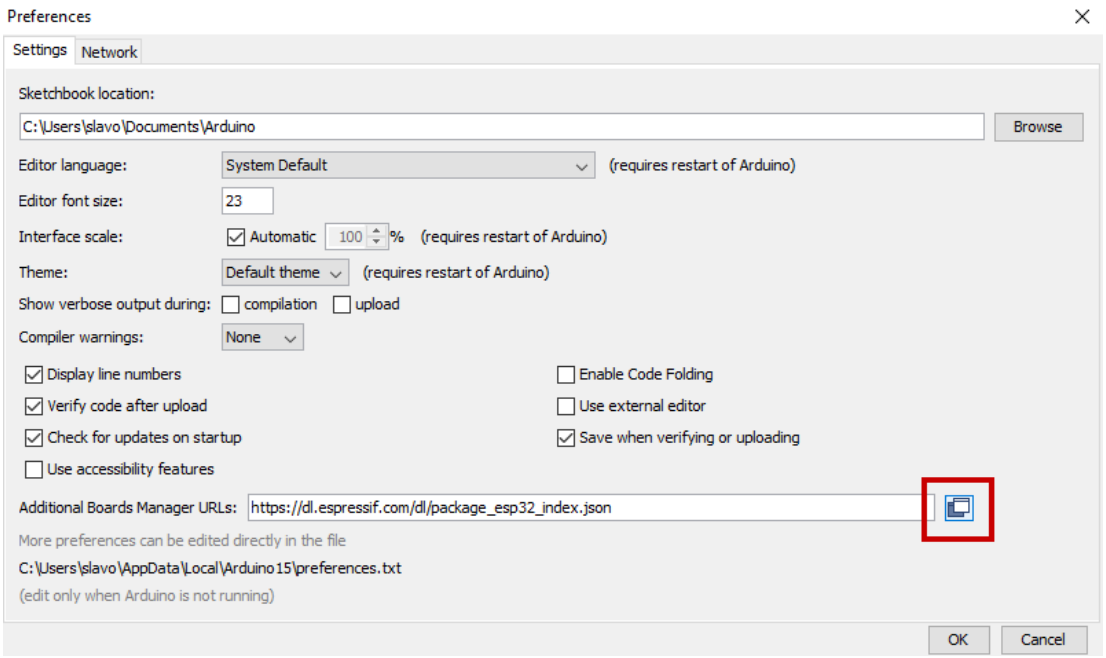


Für *Linux* Benutzer, ist zum Beispiel der Portname */dev/ttyUSBx*, wobei x für eine ganze Zahl zwischen *0* und *9* steht.

D1 Mini v3 mit Arduino IDE

Um die Arduino IDE so einzurichten, dass das D1 Mini darüber programmiert werden kann, folgen Sie ein paar einfachen Schritten.

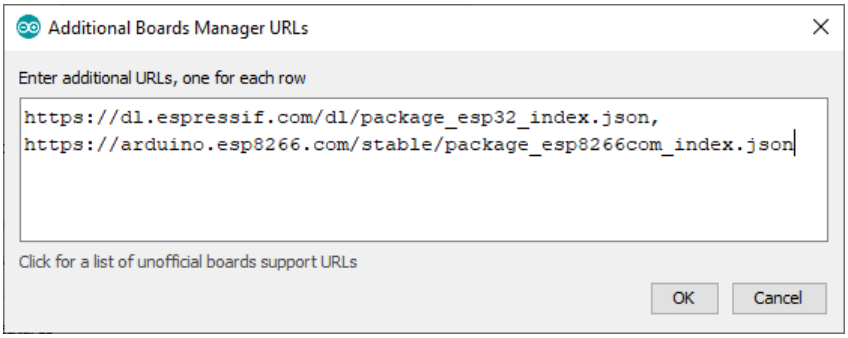
Als erstes müssen Sie den ESP8266-Core installieren. Um ihn zu installieren, öffnen Sie die Arduino IDE und gehen Sie zu:
`File > Preferences`
und finden Sie das Feld *`Additional URLs`*.



Dann kopieren Sie folgende URL:
https://arduino.esp8266.com/stable/package_esp8266com_index.json

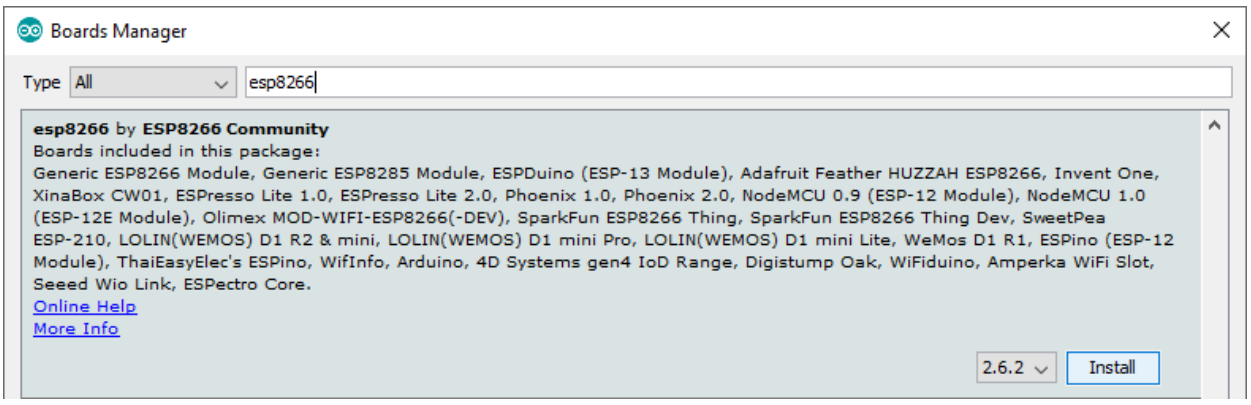
und fügen Sie ihn in das Feld *`Additional URLs`* ein.

Wenn Sie bereits einen oder mehrere Links in diesem Feld haben, fügen Sie einfach ein Komma nach dem letzten Link ein, fügen Sie den neuen Link nach dem Komma ein und klicken Sie auf die Schaltfläche *OK*. Schließen Sie dann die Arduino-IDE.



Öffnen Sie die Arduino IDE und gehen Sie zu:
Tools > Board > Boards Manager

Es öffnet sich ein neues Fenster, geben Sie "*esp8266*" in das Suchfeld ein und installieren Sie das Board mit dem Namen "*esp8266*" von "*ESP8266 Community*", wie unten abgebildet:

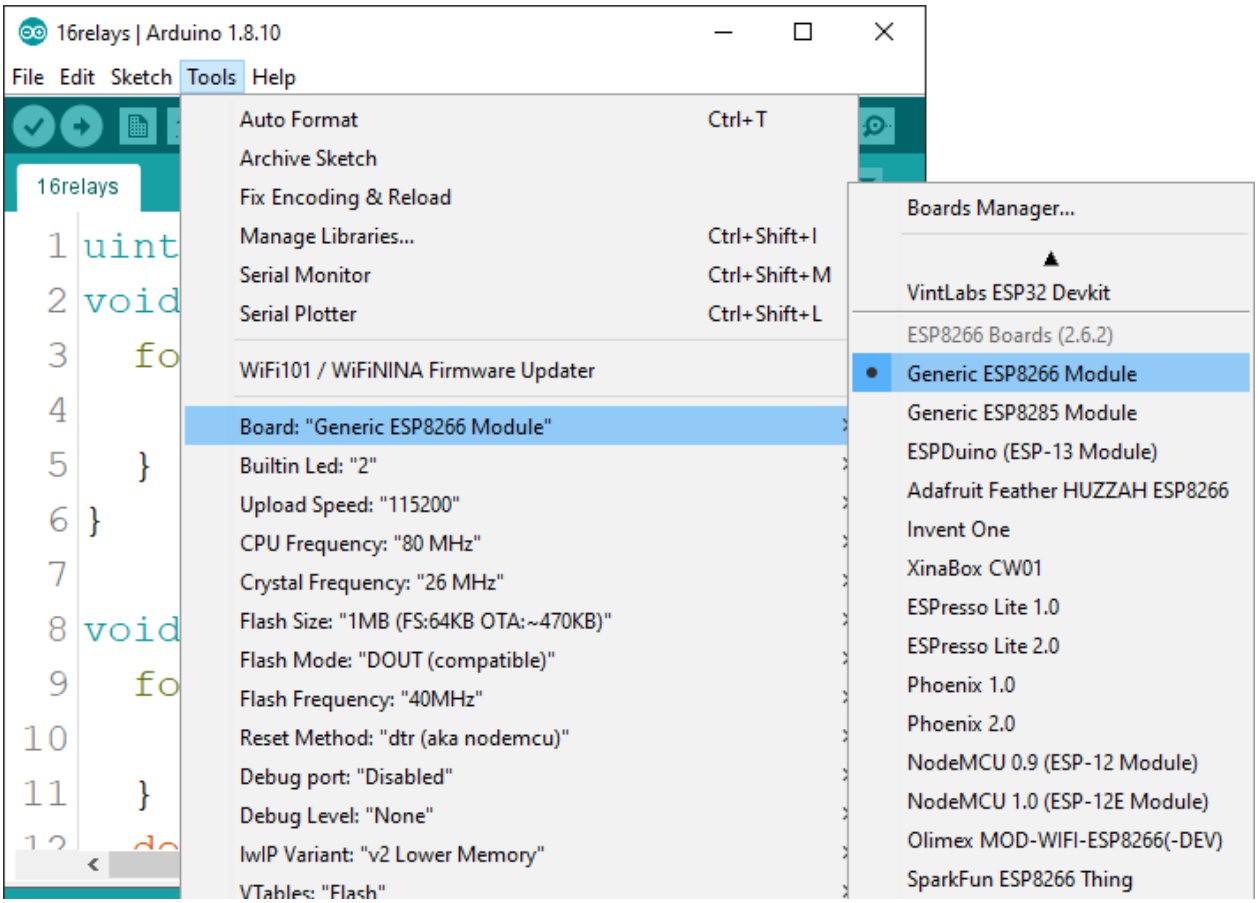


Jetzt haben Sie den ESP8266-Core installiert.

Als Nächstes wählen Sie das richtige Board in der Arduino IDE aus. Öffnen Sie die Arduino IDE und gehen Sie zu:

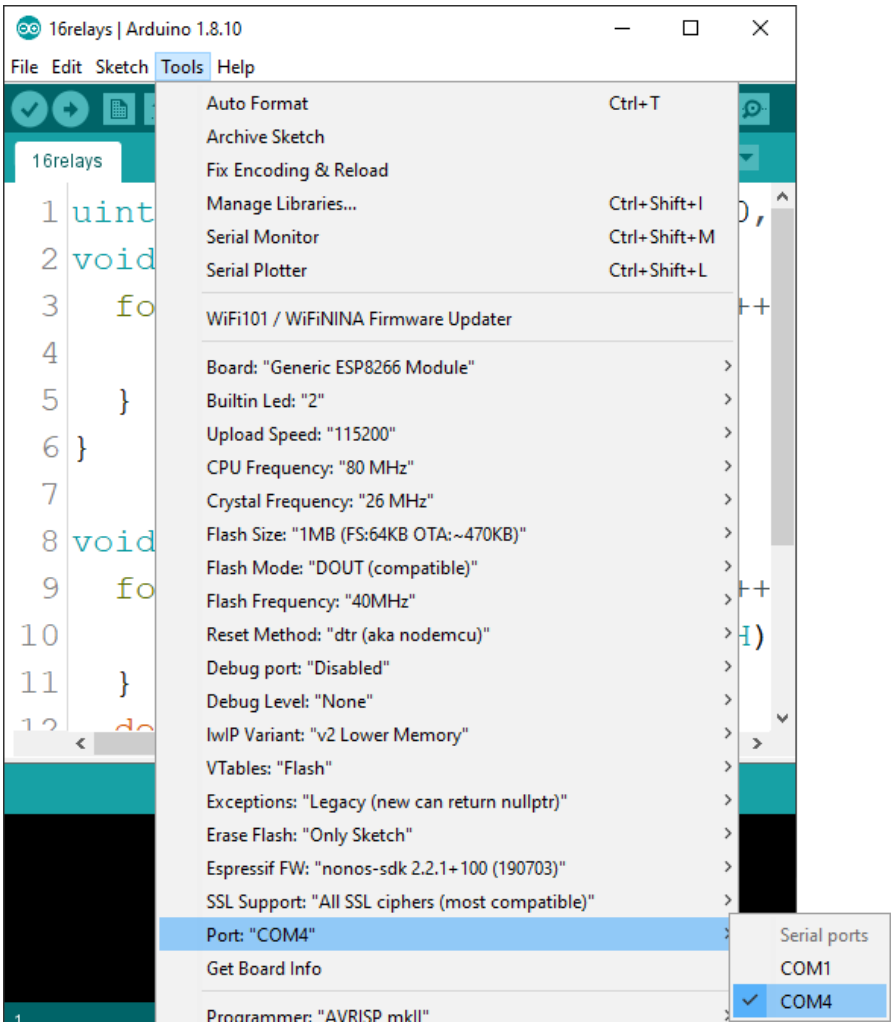
`Tools > Board > {board name}`

und wählen Sie das erste *`Generic ESP8266 Module`* aus, *wie unten abgebildet:*



Wählen Sie danach den Port aus, an dem das D1 Mini verbunden ist. Gehen Sie zu: *`Tools > Port > {port name goes here}`*

Wenn das D1 Mini Modul v3 über den USB-Port verbunden ist, sollten mehrere Portnamen verfügbar sein. In diesem eBook wird die *Arduino IDE* in *Windows* verwendet, die Portnamen lauten daher wie folgt:



For Linux users, port name is */dev/ttyUSBx* for example, where "*x*" represents specific integer number between *0* and *9*.

Blink + PWM + Serial-Sketch-Beispiele

Mit der ESP8266 Library kommt ein Beispiel eines Blink-Sketches.
Um es zu öffnen gehen Sie zu:
Files > Examples > ESP8266 > Blink:

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);     // Initialize the LED_BUILTIN
}
void loop() {
  digitalWrite(LED_BUILTIN, LOW);   // Turn the LED on
  //  Note that LOW is the voltage level
  //  but actually the LED is on; this is because
  //  it is active LOW on the ESP
  delay(1000);                      // Wait for a second
  digitalWrite(LED_BUILTIN, HIGH);  // Turn the LED off
                                    // by making the voltage HIGH
  delay(2000);                      // Wait for two seconds
}
```

Für das D1 Mini v3 Modul ist *LED_BUILTIN* gleich "2", was bedeutet, dass die Onboard-LED mit dem *GPIO2*-Pin verbunden ist. Um die LED **einzuschalten**, müssen wir den *GPIO2*-Pin auf **LOW** schalten, und um die LED **auszuschalten**, müssen wir den *GPIO2*-Pin auf **HIGH** schalten.

Software PWM Sketch-Beispiel:

```
int brightness = 1; //  do not set it to the zero
                    //  zero disables the PWM on a specific pin
uint8_t fadeAmount = 5;
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  analogWrite(LED_BUILTIN, 200);  //  high brightness
  delay(1000);
  analogWrite(LED_BUILTIN, 500);
  delay(1000);
  analogWrite(LED_BUILTIN, 800);
  delay(1000);
  analogWrite(LED_BUILTIN, 1000); //  low brightness
  delay(1000);

  //  fading led
  while (1) {
    analogWrite(LED_BUILTIN, brightness);
    brightness = brightness + fadeAmount;
    if (brightness < 0 || brightness >= 1023) {
      fadeAmount = -fadeAmount;
    }
    delay(10);
  }
}
```

Um PWM auf dem D1 Mini Modul (ESP8266) zu verwenden, verwenden wir einfach dieselbe Funktion wie auf Arduino-Boards

`analogWrite(pin, value)`

wobei *`Pin`* jeder freie *GPIO*-Pin von ESP8266 ist und "*value*" das Tastverhältnis ist, ein Wert zwischen 1 und 1023.

Verwenden Sie nicht die Null als Wert, weil dadurch die PWM-Funktion an diesem Pin ausgeschaltet wird!

Je näher der Wert an Null ist, desto höher ist die Helligkeit der LED.
Je näher der Wert an 1023 liegt, desto dunkler wird die LED.

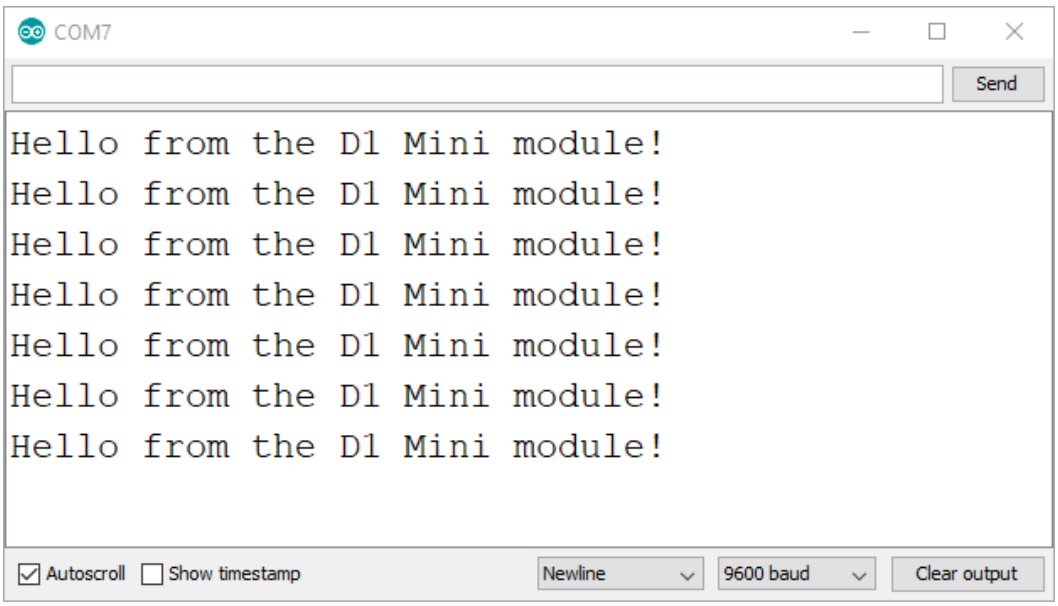
Um die LED ausblenden zu lassen, muss die *`while(1)`* loop innerhalb *`loop()`* Funktion verwendet werden, sonst wird sie nicht funktionieren.

Das Serielle-Kommunikations-Sketch-Beispiel

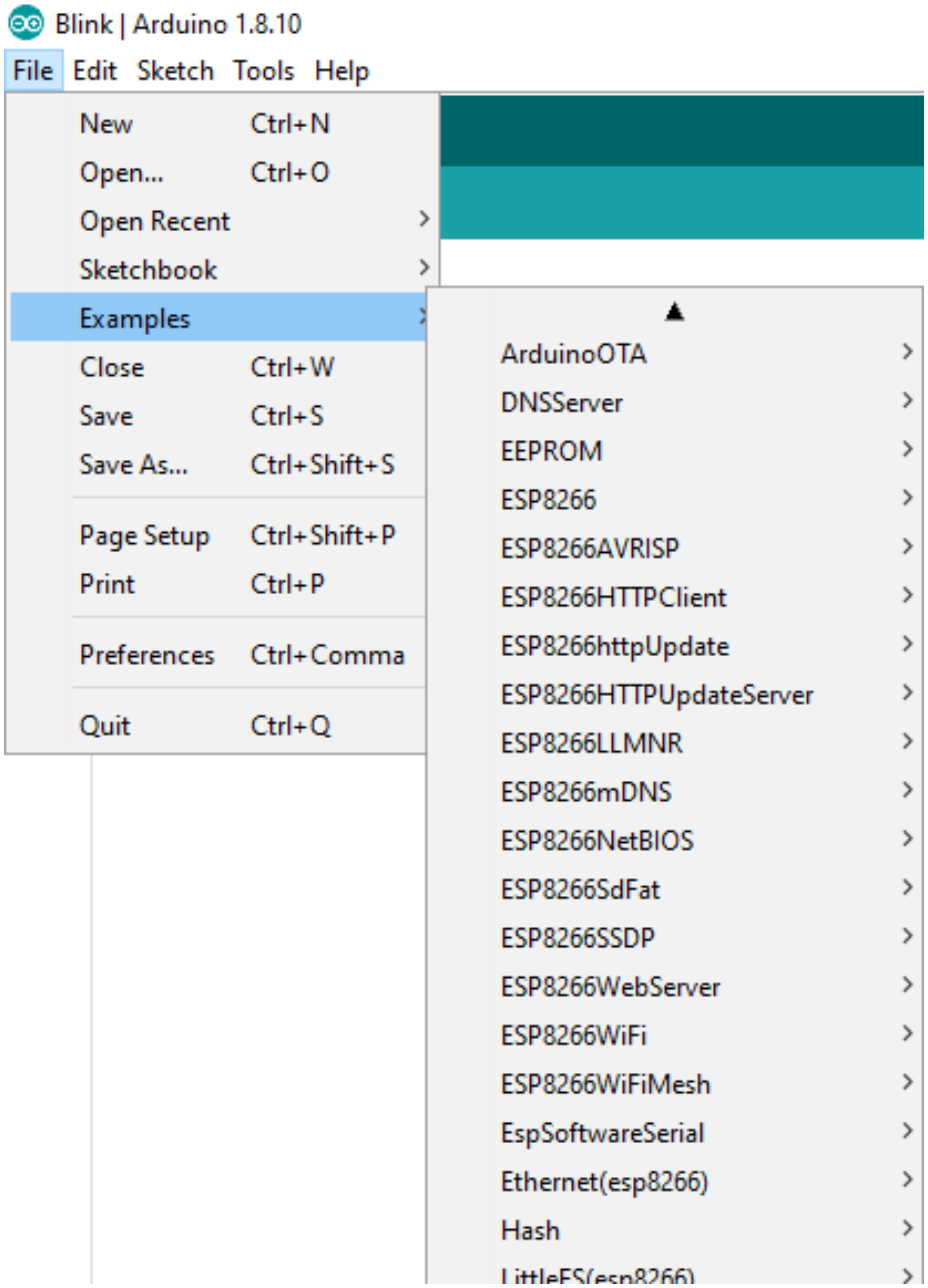
```
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println("Hello from the D1 Mini v3 module!");
  delay(1000);
}
```

Laden Sie den Sketch auf das D1 Mini v3 Modul und öffnen Sie den Serial Monitor (*Tools > Serial Monitor*). Die Ausgabe sollte wie folgt aussehen:



Mit der *ESP8266*-Library kommen viele weitere Sketch-Beispiele. Der Wifi-Teil vom D1 Mini v3 Modul könnte dort getestet werden, was in diesem eBook nicht behandelt wird.



Jetzt sind Sie dran! Entwickeln Sie Ihre eigenen Projekte und Smart-Home Installationen. Wie Sie das bewerkstelligen können, zeigen wir Ihnen unkompliziert und verständlich auf unserem Blog. Dort bieten wir Ihnen Beispielskripte und Tutorials mit interessanten kleinen Projekten an, um schnell in die Welt der Mikroelektronik einzusteigen. Zusätzlich bietet Ihnen auch das Internet unzählige Möglichkeiten, um sich in Sachen Mikroelektronik weiterzubilden.

Falls Sie nach weiteren hochwertigen Produkten für Arduino und Raspberry Pi suchen, sind Sie bei AZ-Delivery Vertriebs GmbH goldrichtig. Wir bieten Ihnen zahlreiche Anwendungsbeispiele, ausführliche Installationsanleitungen, E-Books, Bibliotheken und natürlich die Unterstützung unserer technischen Experten.

https://az-delivery.de

Viel Spaß!

Impressum

https://az-delivery.de/pages/about-us