Willkommen!

Und herzlichen Dank für den Kauf unseres AZ-Delivery ESP8266-01. Auf den folgenden Seiten gehen wir mit dir gemeinsam die ersten Schritte der Programmierung durch.

Viel Spaß!

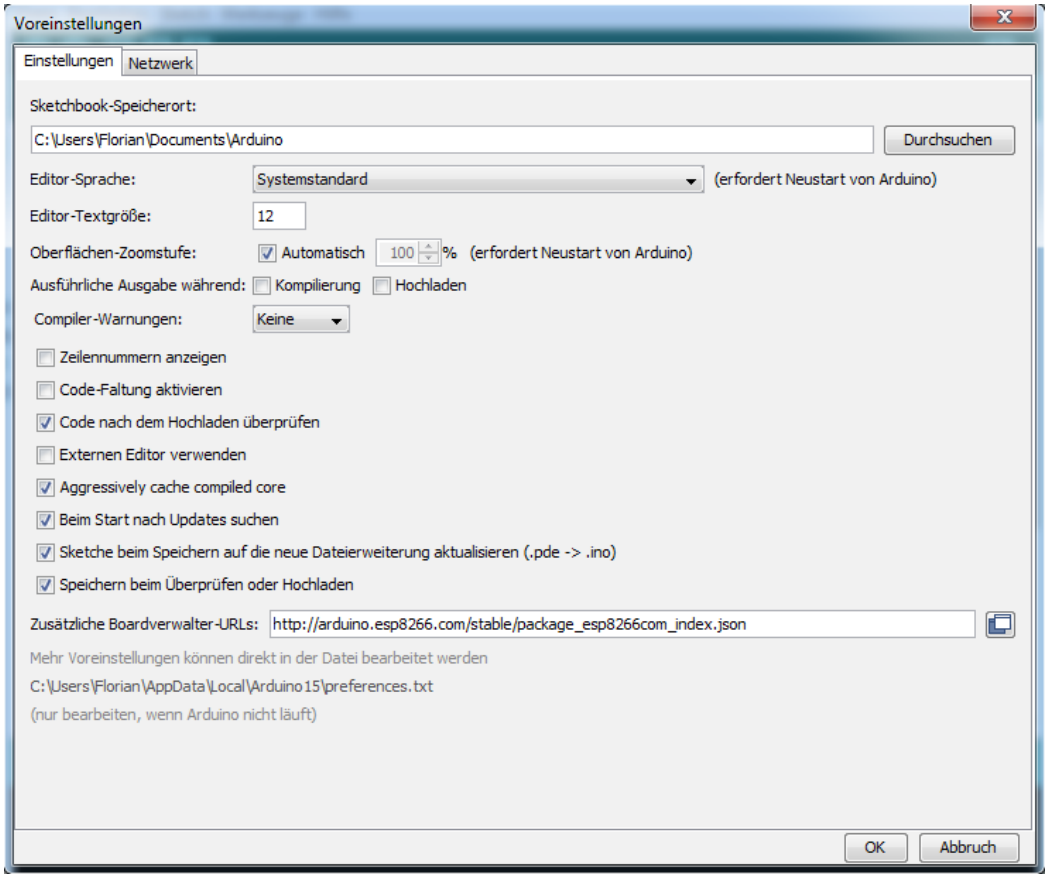
Der ESP8622-01 hat durch seinen WLAN 802.11 b/g/n Standard eine große Reichweite und ist Universell einsetzbar. Das Modul unterstützt 3 Betriebsmodi: WLAN Router (AP), WLAN Client (STA) sowie beides gleichzeitig (AP + STA)! Der leistungsstarke 80MHz Prozessor und 1MB Speicher lassen den ESP8266-01 für viele Anwendungen einsetzten. Begrenzt ist er im Gegensatz zum großen Bruder (ESP8266-12E) durch 2 GPIO-Pins.

Vorbereiten der Software:

Die Arduino Software sehen wir in diesem Schritt als Installiert an, sollte diese bei dir noch fehlen, so kannst du diese unter https://www.arduino.cc/en/Main/Software# herunterladen und auf deinen PC installieren. Außerdem die Treiber für den CH340 solltest du auch schon installiert haben, wenn nicht, bekommst du die Treiber hier bei uns: CH340.

Nachdem alle Grundvoraussetzungen getätigt wurden, starten wir nun mit der Einrichtung der Software. Die Arduino Software benötigt zunächst einmal alle Informationen zum ESP8266-01S, dies können wir tun, indem wir unter dem „Voreinstellungen“ > „Zusätzliche Boardverwalter-URLs“ folgende Adresse eingeben:

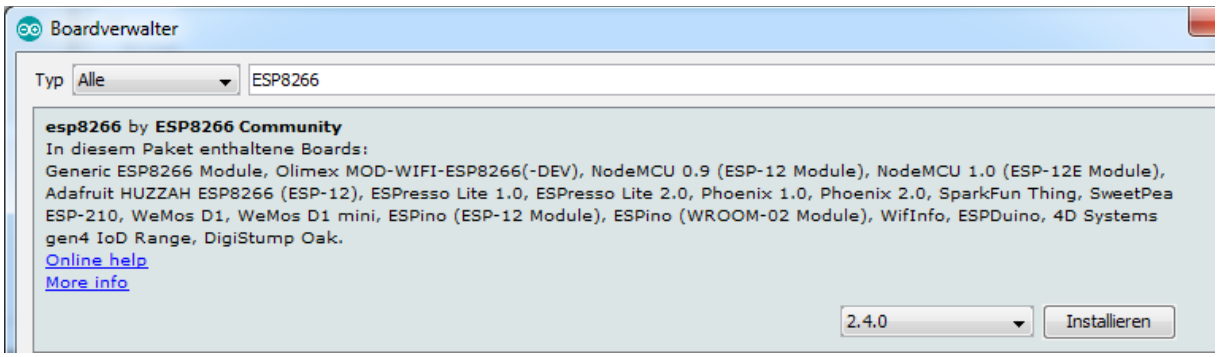
http://arduino.esp8266.com/stable/package_esp8266com_index.json



Evtl. wenn du schon einen Link eingetragen hast, auf den Button  klicken und in dem Fenster eine neue Zeile hinzufügen.

Bestätigen wir die Eingabe mit „OK“.

Ist das erledigt, gehen wir auf „Werkzeuge“ > „Board“ > „Boardverwalter“ und installieren die ESP8266 Bibliothek. In dem Boardverwalter geben wir in der Suchleiste oben rechts „ESP8266“ ein, es wird das Paket von ESP8266 Community angezeigt. Dieses wählen wir aus und klicken auf Installieren.



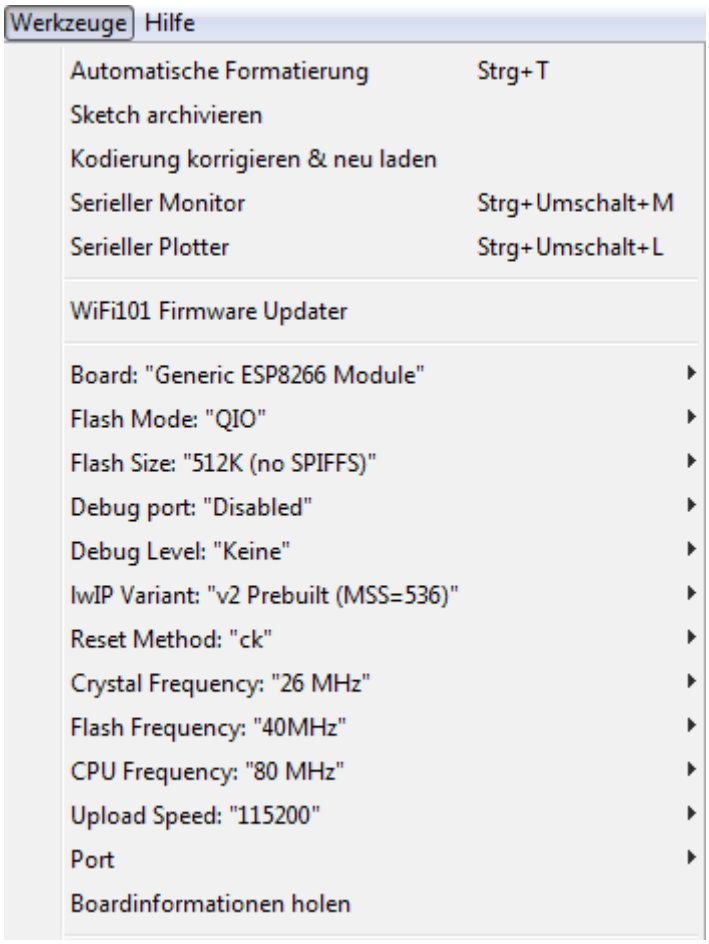
Nach erfolgter Installation steht neben dem Paket nun INSTALLED.

Als nächsten Schritt müssen wir das richtige Board auswählen:

Unter Werkzeuge >

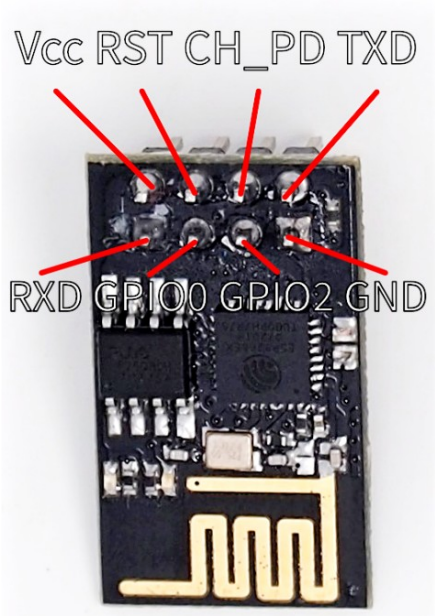
Board: „Generic ESP8266 Module“

Port: “COMxx” (hier dein Port des Serial Adapters)



Jetzt sind alle Grundeinstellungen getätigt, jetzt geht es an die Verdrahtung. Um den ESP8266-01 Programmieren zu können, muss dieser zuerst in den Programmiermodus versetzt werden.

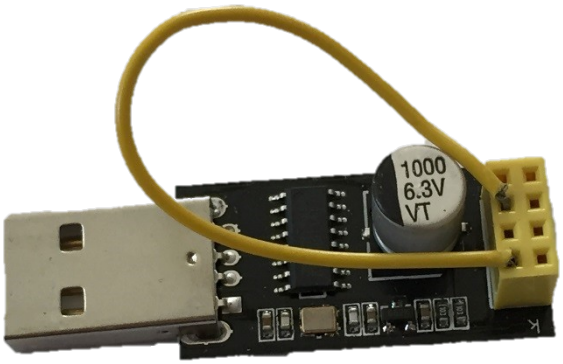
Verdrahten des Moduls mit dem Serial Adapter:

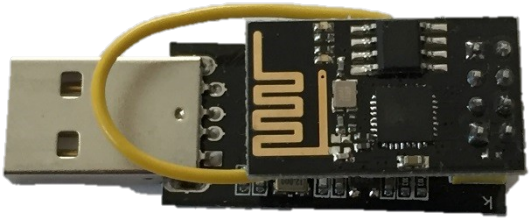


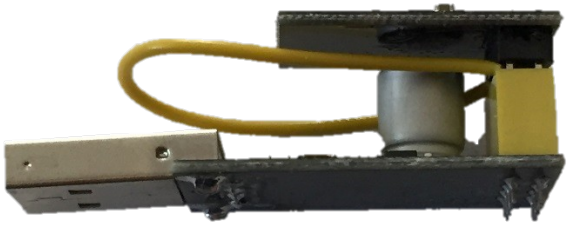
GPIO0 muss während des Bootens auf Masse liegen, damit der Chip in den Programmiermodus geht.

Dazu eine Brücke zwischen GPIO0 und GND legen.

Diese Brücke muss nach dem Programmieren wieder entfernt werden.



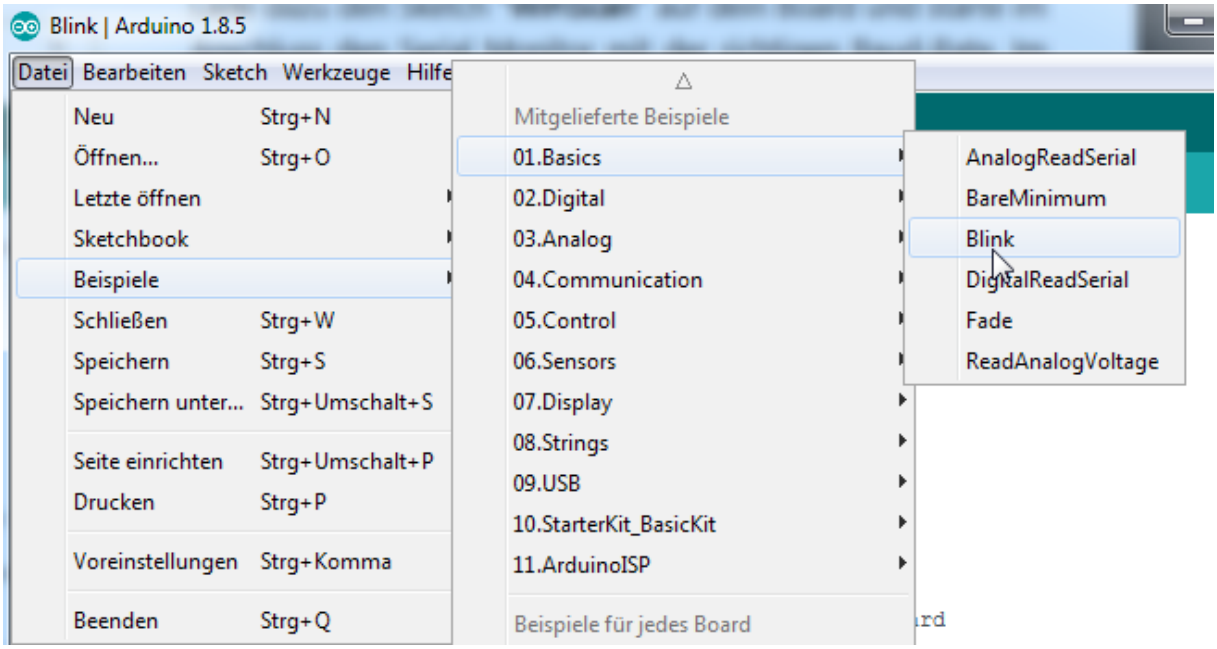




Der Arduino Code:

Nachdem nun die Verdrahtung erledigt wurde und der Adapter eingesteckt wurde, schreiben wir unseren ersten Code. Lassen wir die LED direkt auf dem ESP8266 blinken.

Wähle dazu unter Datei > Beispiele > 01.Basics > Blink aus.




```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

Sollte bei dir die LED mit „LED_BUILTIN“ nicht blinken, dann musst du LED_BUILTIN durch Port 1 ersetzten und erneut versuchen:

```
void setup() {
  pinMode(1, OUTPUT);
}
void loop() {
  digitalWrite(1, HIGH);
  delay(1000);
  digitalWrite(1, LOW);
  delay(1000);
}
```

.

Nachdem wir den Code geöffnet oder geändert haben klicken wir oben auf  und Verifizieren unser Programm:

Wenn alles stimmt und unser Programm keine Fehler enthält

```
Der Sketch verwendet 247055 Bytes (23%) des Programmspeicherplatzes. Das Maximum sind 1044464 Bytes.
Globale Variablen verwenden 32868 Bytes (40%) des dynamischen Speichers, 49052 Bytes für lokale Varia
```

können wir es auf den ESP8622-01 hochladen. Dazu klicken wir oben auf 

Ist unser ESP8266-01 nicht im Programmiermodus, dann bekommen wir folgende Meldung:

```
error: espcomm_upload_mem failed
error: Failed to open COM11
error: espcomm_open failed
error: espcomm_upload_mem failed
error: espcomm_upload_mem failed
```

Dann einfach kurz Spannungsversorgung trennen und wieder verbinden oder Reset betätigen und überprüfen ob der GPIO02 auch auf Masse liegt.

Dann nochmal das Hochladen  starten und es sollte nun klappen.

Nach dem Programmieren, wie schon beschrieben wurde, die hellblauen markierten Leitungen wieder trennen!

Nun blinkt die LED im Sekundentakt, aber dies ist nur ein kleiner Teil, was dein ESP8266-01 kann.

Verwendung des ESP8266-01 als WLAN Modul für den Arduino

Grundlagen der Kommunikation über AT Befehle

Das AZ-Delivery esp8266 WLAN Modul hat standardmäßig eine Firmware vorinstalliert, mit Hilfe derer die angeschlossenen Geräte, beispielsweise ein Arduino, mit ihm über AT Befehle kommunizieren können. Der AT Befehlssatz ist relativ alt (ca. 30 Jahre) und wurde ursprünglich für die Kommunikation mit Modems entwickelt. Durch seine Simplizität und seine Sparsamkeit bezüglich Ressourcen ist er jedoch ideal für die Kommunikation mit dem ESP8266-01 geeignet.

Alle AT Befehle beginnen mit „**AT+**“ und anschließend kommt erst der eigentliche Befehl. Außerdem wichtig ist, dass alle AT Befehle mit einem Zeilenumbruch (CR und NL) enden müssen. Aus diesem Grund sollte man, wenn man über die serielle Verbindung zwischen ESP8266 und Arduino einen Befehl sendet immer println verwenden, um den Zeilenumbruch mitzuschicken, und dem Modul mitzuteilen den Befehl auszuführen.

Die AT-Befehle können im Datenblatt von ESPRESSIF nachgelesen werden:

https://www.espressif.com/sites/default/files/documentation/4a-esp8266_at_instruction_set_en.pdf

Download einer Website

In diesem Beispiel wollen wir dem AZ-Delivery ESP8266-01 über AT Befehle mitteilen eine Website herunterzuladen. Dazu muss das WLAN Modul zunächst über den Befehl **AT+CWMODE=1** in den „WLAN Client“ Modus versetzt werden. Anschließend kann es über **AT+CWJAP=“SSID“,“Passwort“** mit einem WLAN Netzwerk verbunden werden.

Nun kann man über **AT+CIPSTART=“TCP“,“website.de“,80** eine Verbindung mit dem Zielserver aufbauen. Anschließend muss man dem WiFi Modul über **AT+CIPSEND=ZAHL** mitteilen, wie lange die Anfrage ist, welche man an den Server schicken möchte. Dabei muss man immer noch ein paar Zeichen dazurechnen, da auch die Zeilenumbrüche am Ende mitgerechnet werden müssen.

Die Anfrage an den Server erfolgt nach dem Prinzip GET /Pfad_zur_Website.html. Nun muss man evtl. noch Leerzeichen einfügen, bis man die vorher angegebene Zahl an Zeichen erreicht hat, woraufhin der ESP8266 die Anfrage automatisch an den Server schickt, und automatisch die Antwort (HTML Code der Website) zurück an den Arduino sendet. Solange die angegebene Zahl an Zeichen nicht erreicht ist, tut der ESP8266 nichts. Sollte die Abfrage länger sein, als angegeben gibt er „busy“ aus.

Aufgrund der Komplexität dieses Verfahrens sollte man es zunächst am PC über die serielle Konsole ausprobieren, bevor man das Verfahren in Arduino Code übersetzt, da sich ansonsten die Fehlersuche sehr schwer gestaltet.

Du hast es geschafft, du kannst nun deine Projekte mit ESP8266-01 programmieren und verwirklichen!

Ab jetzt heißt es Experimentieren.

Und für mehr Hardware sorgt natürlich dein Online-Shop auf:

https://az-delivery.de

Viel Spaß!
Impressum

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