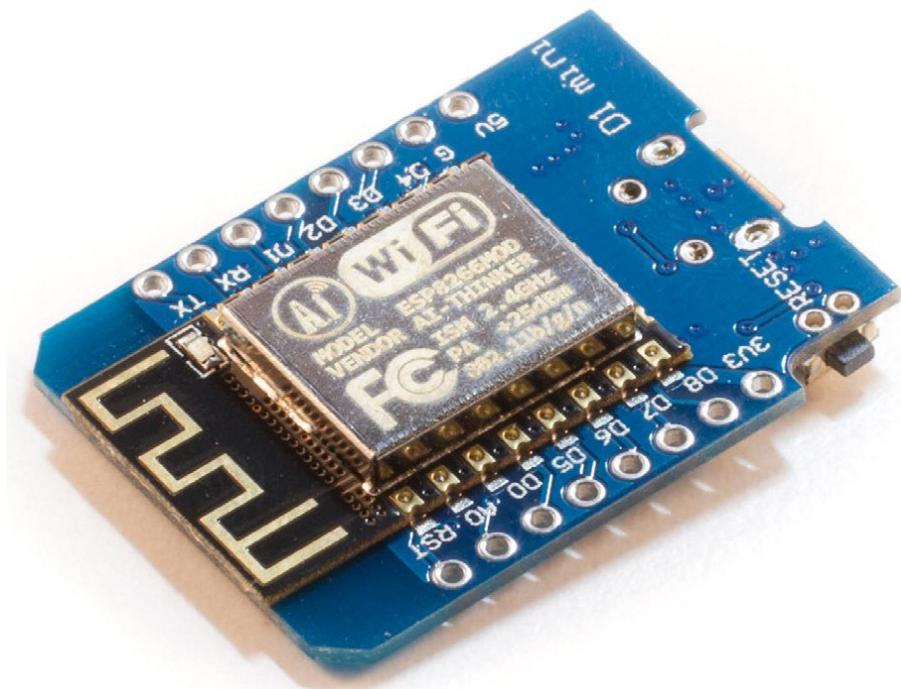




Willkommen!

Und herzlichen Dank für den Kauf unserer **AZ-Delivery D1 Mini NodeMCU!** Auf den folgenden Seiten gehen wir mit dir gemeinsam die ersten Schritte von der Einrichtung bis zu den ersten Scripten. Viel Spaß!



<http://flyt.it/D1Mini>

Das Pin-Layout der **AZ-Delivery D1 Mini NodeMCU** entspricht Version 2.3 des Originals der Firma WeMos Electronics. Sie besitzt die Breadboard-freundliche Breite der NodeMCU V2 Amica und die Dank des **CH340G** USB-Konverters gleiche Systemkompatibilität wie die NodeMCU Lolin V3. Dazu ist sie wesentlich kleiner als die anderen beiden, besitzt aber auch weniger ausgeführte Pins. Betrieben wird sie über einen Micro-USB-Anschluss.

Die wichtigsten Informationen in Kürze

- » Programmierung über Micro USB-B-Kabel
- » Stromversorgung über:
 - » Micro USB-B am USB-Anschluss des Rechners
 - » Micro USB-B am 5V USB-Netzteil
- » 9 digitale I / O-Pins (3,3V!)
- » 1 analoger I / O-Pin
- » ESP8266 Controller, Variante ESP-12F
- » CH340G USB-Schnittstelle
- » Programmierbar über Arduino Code und Lua

Auf den nächsten Seiten findest du Informationen zur

- » ***Treiber-Installation und Vorbereitung der Arduino IDE,***
eine Anleitung für
 - » ***das erste Script per Arduino Code,***
im Anschluss die
 - » ***Systemvorbereitung zur Arbeit mit Lua***
 - und eine Anleitung für
 - » ***das erste Lua-Script.***

Alle Links im Überblick

WeMos D1 Mini Schaltbild:

» https://wiki.wemos.cc/_media/products:d1:mini_new_v2_2_0.pdf

Treiber:

- » Windows: http://www.wch.cn/download/CH341SER_ZIP.html
- » Mac: http://www.wch.cn/download/CH341SER_MAC_ZIP.html

Lua-Services:

- » Firmware-Generator: <https://nodemcu-build.com/>
- » esptool.py: <https://github.com/espressif/esptool>
- » NodeMCU-Flasher:
<https://github.com/nodemcu/nodemcu-flasher/blob/master/Win32/Release/ESP8266Flasher.exe>
- » Esplorer: <http://esp8266.ru/esplorer/>
- » Luatool: <https://github.com/4refr0nt/luatool>
- » Lua-Tutorials - WLAN Access Points auflisten:
https://raw.githubusercontent.com/pradeesi/NodeMCU-WiFi/master/list_ap.lua

Sonstige Tools:

- » Python: <https://www.python.org/downloads/>

Interessantes von AZ-Delivery

- » AZ-Delivery G+Community:
<https://plus.google.com/communities/115110265322509467732>
- » AZ-Delivery auf Facebook:
<https://www.facebook.com/AZDeliveryShop/>

Treiberinstallation

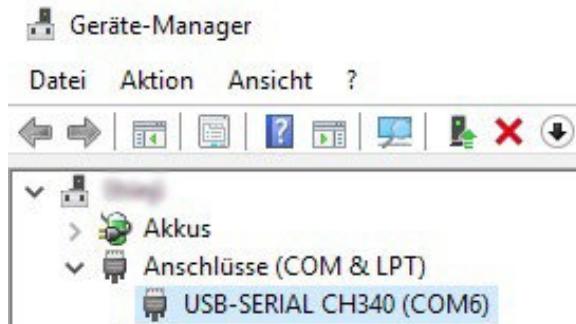
Die **AZ-Delivery D1 Mini NodeMCU** verbindest du über ein Micro-USB-Kabel mit deinem Rechner. Wie bei den meisten Boards von AZ-Delivery ist für die Kommunikation ein **CH340-Chip** im Einsatz, der von Windows automatisch erkannt wird.

Sollte das einmal nicht der Fall sein, lade dir hier den aktuellen Treiber herunter und entpacke ihn.

- » Windows: http://www.wch.cn/download/CH341SER_ZIP.html
- » Mac: http://www.wch.cn/download/CH341SER_MAC_ZIP.html

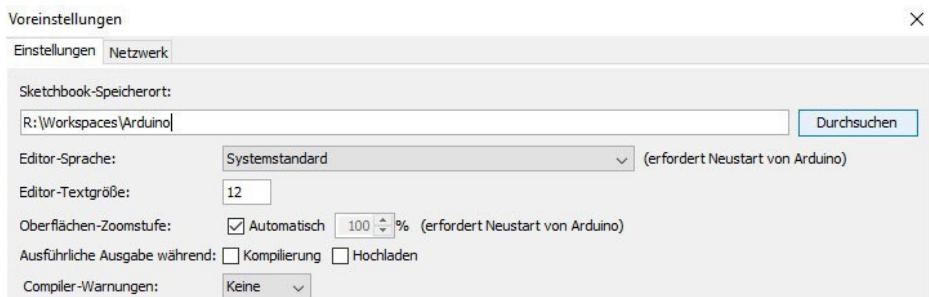
Unter Windows installierst du ihn einfach durch das Ausführen der "**SETUP.EXE**" im Ordner "**CH341SER**". Mac-Nutzer folgen am besten den Installationsanweisungen, die dem Treiberpaket beiliegen.

Nach dem erneuten Anschließen des UNOs sollte dieser als "**USB-SERIAL CH340**"-Gerät (Windows) erkannt werden.



Vorbereitung der Arduino IDE

Besuche die Seite <https://www.arduino.cc/en/Main/Software> und lade die aktuelle Version für dein Betriebssystem herunter. Alternativ kannst du dich für den Arduino Web-Editor registrieren und den leicht verständlichen Installationshinweisen folgen. Die folgenden ersten Schritte nutzen Desktop-Variante für Windows.

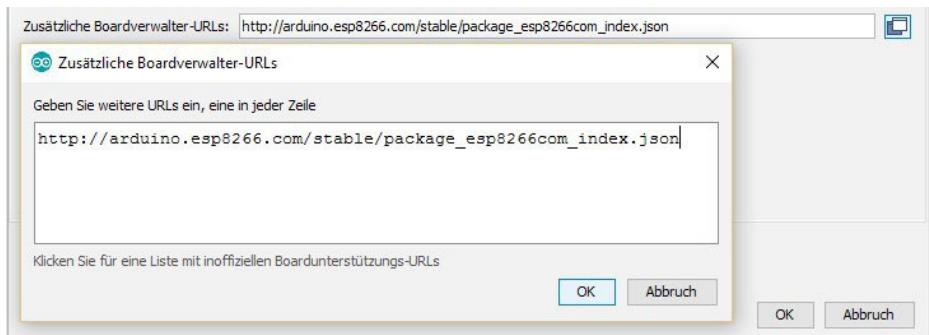


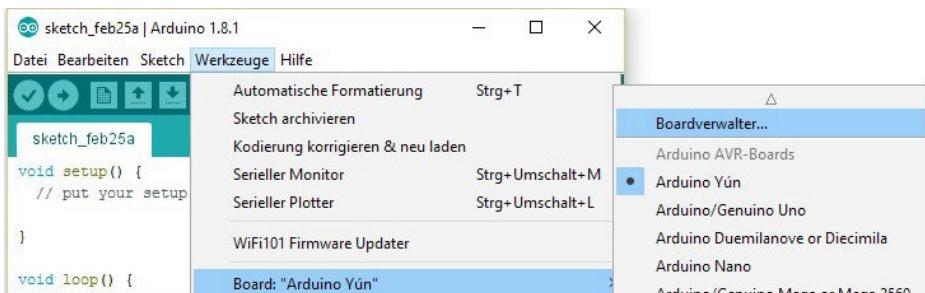
Ist das Programm gestartet, sollte unter **Datei > Voreinstellungen** der an erster Stelle stehende Sketchbook-Speicherort festgelegt werden, beispielsweise unter **Eigene Dokumente\Arduino**. Damit landen deine bei Arduino "Sketche" genannten Scripte auch dort, wo du sie haben möchtest.

Die NodeMCU gehört allerdings nicht zum Standardrepertoire der IDE, weshalb der Boardmanager erweitert werden muss.

Füge dazu im gleichen Fenster unter "**Zusätzliche Boardverwalter-URLs**" folgende Adresse ein:

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

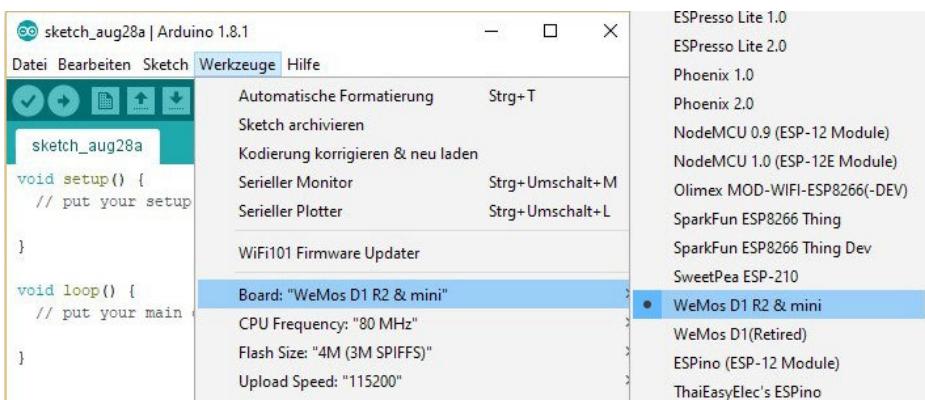




Ist das erledigt gehe zu **Werkzeuge > Board > Boardverwalter** und installiere die Board-Bibliothek "**esp8266 by ESP8266 Community**".



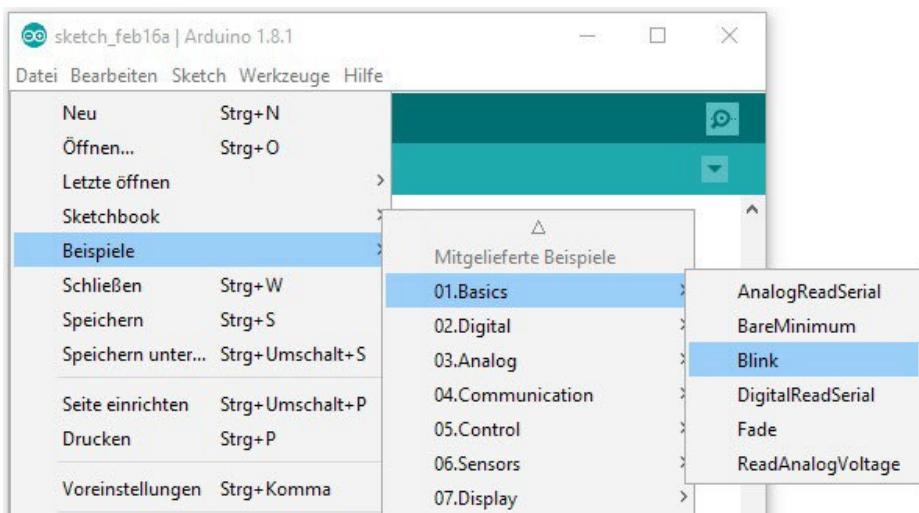
Nun kannst du das richtige Board namens "**WeMos D1 R2 & mini**" auswählen, dazu die CPU Frequenz von **80 MHz**, die Speichergröße "**4M (3M SPIFFS)**", eine Upload-Geschwindigkeit von z. B. **115200** und den passenden Port ("**COM**" bei Windows, "**ttyUSB**" bei MacOS).



Das erste Script per Arduino Code

Während in den meisten Programmiersprachen der erste Erfolg ein zu lesendes "Hello World!" darstellt, ist es bei Arduinos das Blinken der boardinternen LED. Das Script heißt entsprechend "**Blink**".

- » Starte die Arduino IDE und öffne unter "Start" das Blink-Script.



Jeder Sketch enthält immer die Methoden "**setup**" und "**loop**". Erstere wird zu Beginn ausgeführt und in der Regel zur Initialisierung von Pins und angeschlossener Hardware verwendet. Die loop-Methode wird im Anschluss permanent wiederholt und enthält damit fast alle anderen Funktionen.

Die Board-interne LED wird seit einiger Zeit über die IDE-eigene Variable "**LED_BUILTIN**" automatisch ausgewählt. Bei der **D1 Mini**

ist sie für den **Pin D4** definiert. Um die LED anzusprechen kannst du den Code also entweder einfach so belassen oder für eine besser nachvollziehbare Adressierung jedes "LED_BUILTIN" in "D4" ändern.

Mit dem zweiten Symbol unter der Befehlsleiste lädst du den Sketch auf die NodeMCU.

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** Displays "Blink | Arduino 1.8.1".
- Menu Bar:** Contains "Datei", "Bearbeiten", "Sketch", "Werkzeuge", and "Hilfe".
- Toolbar:** Includes icons for file operations (New, Open, Save, Print, Upload, Download) and a preferences icon.
- Sketch Name:** The current sketch is named "Blink".
- Code Editor:** Shows the "Blink" sketch code:

```
// the setup function runs once when you press reset or power the
void setup() {
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);
}

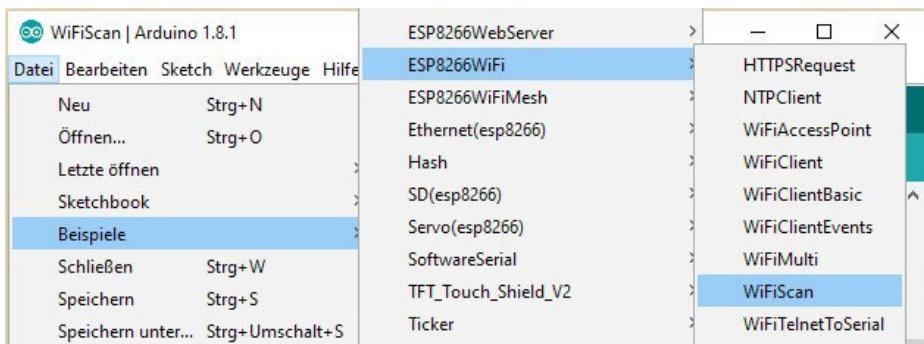
// the loop function runs over and over again forever
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);      // turn the LED on (HIGH is t
    delay(1000);                      // wait for a second
    digitalWrite(LED_BUILTIN, LOW);       // turn the LED off by making
    delay(1000);                      // wait for a second
}
```
- Status Bar:** Shows the message "Der Sketch verwendet 222201 Bytes (21%) des Programmspeicherplatzes. Globale Variablen verwenden 31576 Bytes (38%) des dynamischen Speichers." followed by "Uploading 226352 bytes from C:\Users\steph\AppData\Local\Temp\arduino..." and "WeMos D1 R2 & mini, 80 MHz, 115200, 4M (3M SPIFFS) auf COM3".

War der Upload erfolgreich, blinkt die LED deiner **D1 Mini** im Sekudentakt.

Du hast es geschafft! Herzlichen Glückwunsch!

Als nächstes solltest du die Besonderheit der NodeMCU, nämlich das WLAN-Modul ausprobieren.

Lade dazu den Sketch "WiFiScan" auf dein Board und starte im Anschluss den Serial Monitor mit der richtigen Baud-Rate. Im Anschluss solltest du wenige Sekunden später eine Liste aller in deiner Umgebung funkenden WLAN-Access-Points mitsamt der jeweiligen Signalstärke sehen.



Mithilfe von Arduino Code kannst du noch vieles mehr mit einer NodeMCU erreichen. Starte deine Suche nach weiteren Möglichkeiten am besten bei den anderen Beispielsketchen der Arduino-Bibliothek und im Web beispielsweise auf <http://michael-sarduino.blogspot.de/search?q=8266>. Für Hardwareunterstützung sorgt natürlich unser Online-Shop:

<https://az-delivery.de>

Wenn du aber gleich weitermachen und lernen möchtest, wie du die **D1 Mini NodeMCU** mit Lua-Scripten bedienst, dann gehe gleich zur nächsten Seite.

Systemvorbereitung zur Arbeit mit LUA

Die D1 Mini kommt in der Regel mit einer AT-Firmware des Herstellers AI-Thinker. Um den Chipsatz mit der Scriptsprache LUA zu bedienen, muss zunächst die Grundlage dafür geschaffen werden. Dafür musst du dir die für dein Vorhaben passende Firmware zusammenstellen:

» <https://nodemcu-build.com/>

Neben der Auswahl der stabilen oder der Entwickler-Version stehen jede Menge Optionen zur Verfügung, mit denen du die Funktionalität deines Boards erweitern kannst. Zu viele unnötige Erweiterungen verlangsamen allerdings nur die NodeMCU. Für unser Tutorial-Script genügen die Standardangaben.

Select modules to include

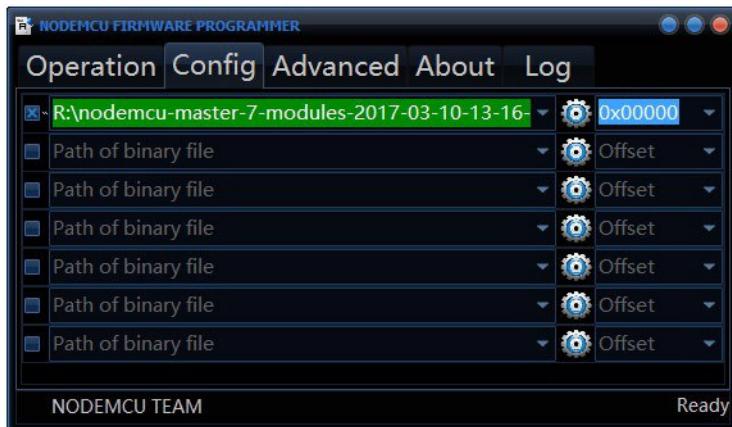
<input type="checkbox"/> ADC	<input checked="" type="checkbox"/> file	<input type="checkbox"/> PCM	<input type="checkbox"/> struct
<input type="checkbox"/> ADXL345	<input type="checkbox"/> gdbstub	<input type="checkbox"/> perf	<input type="checkbox"/> Switec
<input type="checkbox"/> AM2320	<input checked="" type="checkbox"/> GPIO	<input type="checkbox"/> PWM	<input type="checkbox"/> TM1829
<input type="checkbox"/> APA102	<input type="checkbox"/> HMC5883L	<input type="checkbox"/> RC (no docs)	<input checked="" type="checkbox"/> timer
<input type="checkbox"/> bit	<input type="checkbox"/> HTTP	<input type="checkbox"/> rfswitch	<input type="checkbox"/> TSL2561
<input type="checkbox"/> BME280	<input type="checkbox"/> HX711	<input type="checkbox"/> rotary	<input type="checkbox"/> U8G
<input type="checkbox"/> BMP085	<input type="checkbox"/> I ² C	<input type="checkbox"/> RTC fifo	<input checked="" type="checkbox"/> UART
<input type="checkbox"/> CJSON	<input type="checkbox"/> L3G4200D	<input type="checkbox"/> RTC mem	<input type="checkbox"/> UCG
<input type="checkbox"/> CoAP	<input type="checkbox"/> mDNS	<input type="checkbox"/> RTC time	<input type="checkbox"/> websocket
<input type="checkbox"/> Cron	<input type="checkbox"/> MQTT	<input type="checkbox"/> Sigma-delta	<input checked="" type="checkbox"/> WiFi
<input type="checkbox"/> crypto	<input checked="" type="checkbox"/> net	<input type="checkbox"/> SNTP	<input type="checkbox"/> WPS
<input type="checkbox"/> DHT	<input checked="" type="checkbox"/> node	<input type="checkbox"/> Somfy	<input type="checkbox"/> WS2801
<input type="checkbox"/> encoder	<input type="checkbox"/> 1-Wire	<input type="checkbox"/> SPI	<input type="checkbox"/> WS2812
<input type="checkbox"/> end user setup			

Gib also nur im ersten Block zwei Mal deine E-Mail-Adresse an und klicke ganz unten auf "Start your build". In den folgenden Minuten erhältst du eine Auftragsbestätigung und eine E-Mail mit Links zum Download der Firmware. Dabei stehen eine Integer und eine Float-Version zur Auswahl. Sie unterscheiden sich nur dadurch, dass letztere mit Fließkommazahlen umgehen kann. Welche Variante du auswählst, ist für unser Tutorial unerheblich.

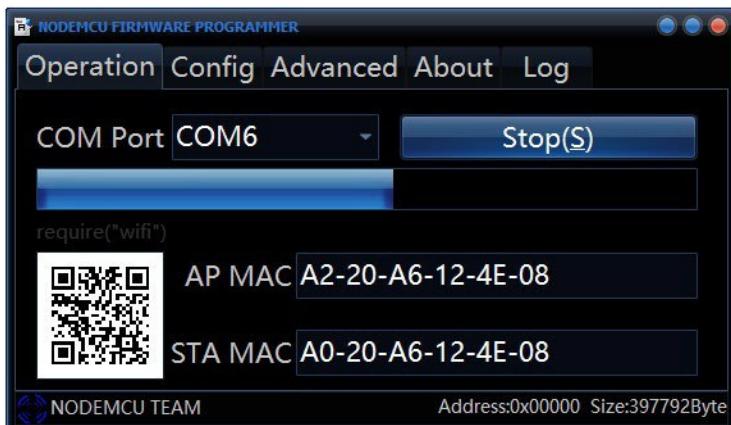
Zur Installation der Firmware ist ein Flash-Tool nötig wie das systemunabhängige Python-Script "**esptool.py**". Komfortabler lässt sich unter Windows das Programm "**NodeMCU-Flasher**" benutzen, welches du hier herunterlädst:

» <https://github.com/nodemcu/nodemcu-flasher/blob/master/Win32/Release/ESP8266Flasher.exe>

Starte das Programm und wähle unter "**Config**" deine heruntergeladene Firmware aus. Belasse die Adresse bei **0x00000**.



Unter "Advanced" findest du Feineinstellungen für das Board. Uns genügen die Standardeinstellungen mit der Baud-Rate von **115200**, **4 MB** Flashspeicher, **40 MHz** Speichergeschwindigkeit und dem "**DIO**"-SPI-Modus. Starte anschließend den Flash-Vor- gang für den COM-Port deinerangeschlossenen NodeMCU und warte auf das grüne Häkchen unten links.

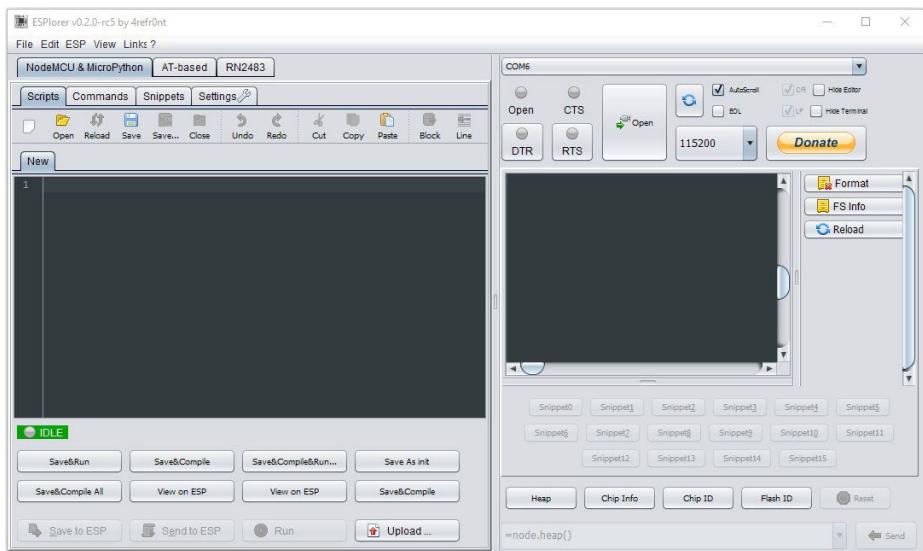


Zuletzt benötigst du noch ein Tool, mit dessen Hilfe du deine Lua- Scripte schreiben und vor allem auf die NodeMCU laden kannst. Reine Konsolennutzung bietet das Paket "**Luatool**" an.

Der "**Esplorer**" ist ebenfalls plattformunabhängig und gehört mit seiner grafischen Benutzeroberfläche zu den beliebtesten Varianten. Auch wir werden ihn für das Tutorial einsetzen.

» Esplorer: <http://esp8266.ru/esplorer/>

Lade die für dein System passende Version herunter und starte nach Entpacken des Archivs die "**Esplorer.bat**" (Windows).



Wie du siehst, kommt das Programm mit einigen vordefinierten Befehlen und kann neben der NodeMCU auch andere Systeme bedienen. Hätten wir im Vorfeld nicht eine neue Firmware geflasht, könnten wir nun die begrenzten Befehle unter dem Reiter "AT-based" nutzen.

Das erste LUA-Script

Als "Hello World"-Script soll für die NodeMCU eine ähnliche Funktion zum Einsatz kommen, wie beim "**WiFiScan**" für die Arduino IDE. Der fertige Code hierfür kann von der folgenden Seite kopiert werden:

» https://raw.githubusercontent.com/pradeesi/NodeMCU-WiFi/master/list_ap.lua

Zunächst überprüfe aber, ob die Installation der Firmware aus dem letzten Schritt tatsächlich funktioniert hat. Dazu wählst du auf der rechten Seite den passenden COM-Port und die richtige Baud-Rate (bei uns **115200**), dann klickst du auf der rechten Seite auf "**Open**".

Im Terminal erscheint "Communication with MCU.." Drücke nun auf den Reset-Taster an der NodeMCU und das Board läuft die Startroutine durch, wobei es die Informationen zur installierten Firmware angibt. Das sollte ähnlich diesem Screenshot aussehen:

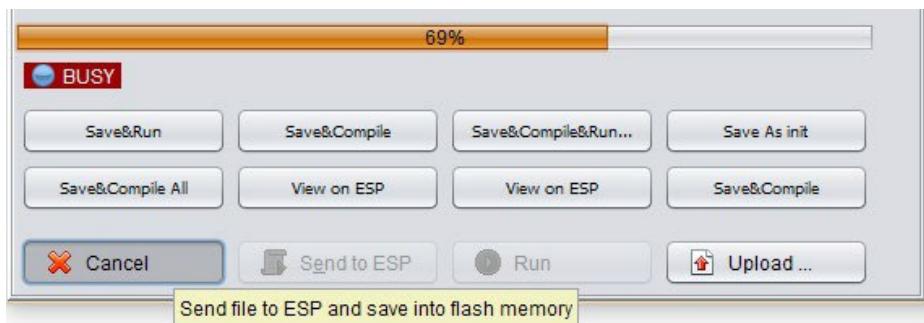
The screenshot shows a terminal window titled "COM6". The top bar includes buttons for "Open", "Close", "DTR", "RTS", and "Donate", along with checkboxes for "AutoScroll", "CR", "LF", "EDL", and "Hide Terminal". The main terminal area displays the following text:

```
NodeMCU custom build by frightanic.com
branch: master
commit: b96e31477ca1e207aa1c0cdc334539b1f7d3a7-
SSL: false
modules: file, gpio, net, node, tmr, uart, wifi
build built on: 2017-03-10 13:15
powered by Lua 5.1.4 on SDK 2.0.0(656edb)
lua: cannot open init.lua
> print(uart.setup(0, 115200, 8, 0, 1, 1 ))
115200
```

Below the terminal are several buttons: Snippet0 through Snippet15, Heap, Chip Info, Chip ID, Flash ID, and a red "Reset" button. At the bottom left is the text "AT" and at the bottom right is a yellow button labeled "Soft reset by command node.reset()".

In das dunkle Fenster auf der linken Programmseite unter dem Reiter "**Scripts**" kopierst du nun den Beispiel-Code.

Wenn du an dieser Stelle abkürzen willst oder einfach ungeduldig bist, dann klicke nun schon einmal unten auf "**Send to ESP**". Der Code wird daraufhin Zeile für Zeile auf der NodeMCU ausgeführt und das Ergebnis im Terminal ausgegeben, ohne dass das Script auf dem Board gespeichert wird.



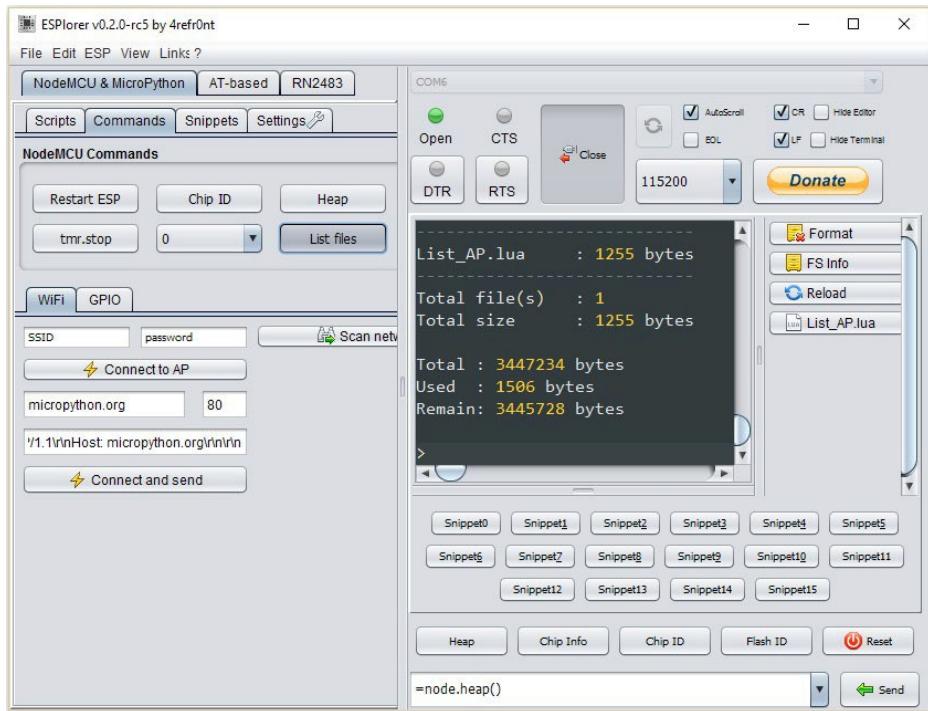
Zum Speichern klicke über dem Script auf "**Save**" und gib dem Script einen Namen, z.B. "**List_AP.lua**". Anschließend wird es automatisch auf die NodeMCU geladen und ausgeführt. Falls nicht, klicke unten links auf "**Save to ESP**".

Um zu überprüfen, ob sich die Datei nun auf dem Board befindet, gehst du wie im Bild rechts zu sehen in den Bereich "**Commands**" und klickst auf "**List files**". Im Terminal sollte nun die Datei "**List_AP.lua**" aufgeführt sein.

Das Script kannst du nun auch über die Kommandozeile unten rechts mit folgendem Befehl direkt von der **D1 Mini** starten.

```
» dofile("List_AP.lua");
```





Ab jetzt heißt es lernen. Das kannst du mithilfe vieler Beispielscripte und weiterer Tutorials im Netz. Unter http://nodemcu.com/index_en.html#fr_5475f7667976d8501100000f kannst du deine Suche nach ihnen beginnen.

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

<https://az-delivery.de>

Viel Spaß!

Impressum

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