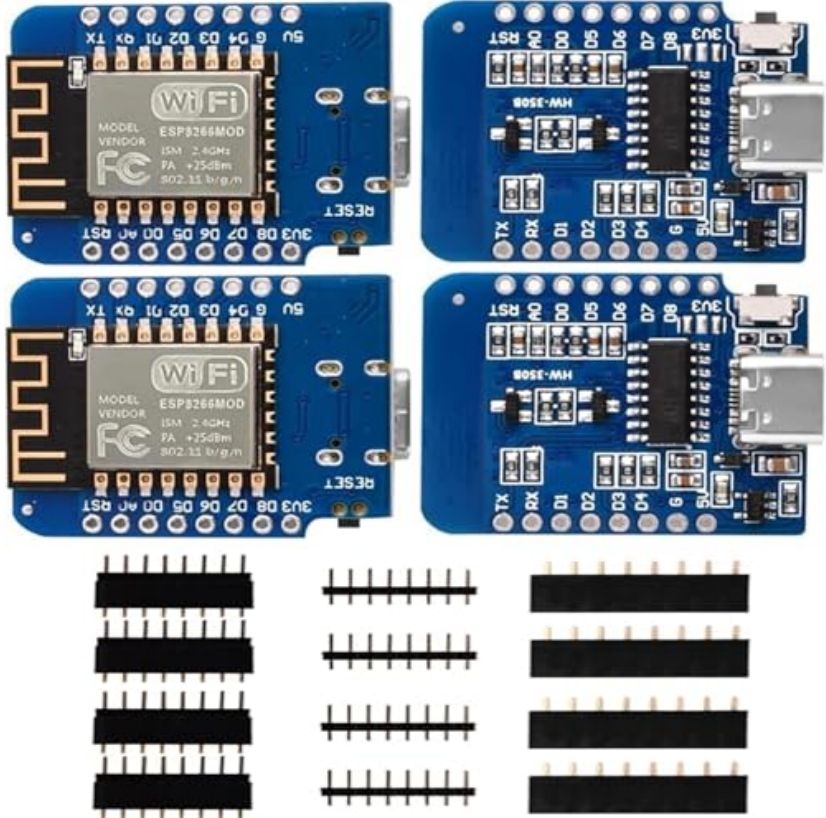



Die aktuelle Uhrzeit mit einem ESP8266 abfragen



Wenn du mit einem [ESP8266](#)* arbeitest und ohnehin damit im Internet bist, gibt es eine einfache Möglichkeit, die aktuelle Uhrzeit herauszufinden: **Mit der Bibliothek NTPClient.h**

Hinweis: Das funktioniert natürlich auch mit einem ESP32 oder einem Arduino mit WiFi Shield – hier konzentrieren wir uns allerdings auf den ESP8266 und den passenden Sketch.

#	Vorschau	Produkt	Preis	
1		4 x 5v D1 ESP8266 Mini Board NodeMCU WiFi ESP8266-12F FT232 WLAN Type C Module 4MBit Flash Memory...	10,99 €	Bei Amazon kaufen
2		AZDelivery 5X D1 Mini NodeMcu mit ESP8266-12F WLAN Module CH340G Lua kompatibel mit Arduino...	18,99 €	Bei Amazon kaufen

#	Vorschau	Produkt	Preis	
3		AZDelivery 3 x D1 Mini NodeMcu mit ESP8266-12F WLAN Module CH340G Lua kompatibel mit Arduino...	13,99 €	Bei Amazon kaufen

Zunächst brauchst du eine Verbindung ins Internet. [Lerne hier, wie du deinen ESP8266 mit dem Internet verbindest.](#)

Wenn du im Internet bist, kann es losgehen. Du benötigst die folgenden drei Bibliotheken. Die erste benötigst du, um deinen ESP8266 mit dem Internet zu verbinden. Die Bibliothek `<NTPClient.h>` verbindet dich mit dem Network Time Protocol und mit `<WiFiUdp.h>` sorgst du für die richtige Datenübertragung mit dem [User Datagram Protocol](#).

```
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
```

Wenn du mehr über die Technik hinter dieser Methode erfahren möchtest, erfährst du alles Wichtige über das Network Time Protocol [bei Wikipedia](#).

Übrigens: Wenn du eine Real Time Clock verwenden möchtest, erfährst du in [diesem Tutorial, wie du eine RTC anschließt](#).

Abfrage von Wochentag und Uhrzeit

Die Bibliothek `<NTPClient.h>` bietet dir die Möglichkeit, ganz einfach Daten wie den Wochentag oder auch die Stunde, Minute und Sekunden abzufragen. Die Zeitzone für die das geschieht, ist UTC. Diese liegt – in der Winterzeit – eine Stunde hinter der Mitteleuropäischen Zeit (MEZ), weshalb du diese Zeitverschiebung in deinem Sketch (in Sekunden) definieren musst. **Vorsicht: Wenn wir Winterzeit haben, sind das 3600 Sekunden – in der Sommerzeit 7200**, da die UTC dann zwei Stunden hinter der MEZ liegt.

```
const long utcOffsetInSeconds = 3600;
```

Für den Fall, dass du auch den aktuellen Wochentag abrufen möchtest, erstellst du dir vorsorglichen ein entsprechendes Array:

```
char daysOfTheWeek[7][12] = {"Sonntag", "Montag", "Dienstag",  
"Mittwoch", "Donnerstag", "Freitag", "Samstag"};
```

Nun erstellst du zwei Instanzen der UDP- und NTP-Bibliothek inklusive der notwendigen Parameter wie z.B. der oben definierten Zeitverschiebung:

```
WiFiUDP ntpUDP;  
NTPClient timeClient(ntpUDP, "pool.ntp.org",  
utcOffsetInSeconds);
```

Nachdem sich dein ESP8266 in der Setup-Funktion mit dem Internet verbunden hast, kannst du deinen NTP Client mit **timeClient.begin()** initialisieren und im Loop deines Sketches mit **timeClient.update()** eine Abfrage durchführen lassen. Diese Funktion besorgt sich die aktuelle Uhrzeit und verarbeitet sie, sodass sie in einem lesbaren Format ausgegeben werden kann.

Die Daten kannst du anschließend mit ein paar weiteren

Funktionen in deinem Seriellen Monitor ausgeben. Die aktuelle Uhrzeit z.B. so:

```
Serial.print(timeClient.getFormattedTime());
```

Den aktuellen Wochentag kannst du mit Hilfe dieser Funktion und deinem oben erstellen Array der Wochentage abrufen:

```
Serial.print(daysOfTheWeek[timeClient.getDay()]);
```

Hier nun der vollständige Sketch zum Rauskopieren und Ausprobieren:

```
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
```

```
const char *ssid      = "Name deines WiFi-Netzwerks";
const char *password = "Dein WiFi-Passwort";
```

```
//Zeitverschiebung UTC <-> MEZ (Winterzeit) = 3600 Sekunden (1
Stunde)
//Zeitverschiebung UTC <-> MEZ (Sommerzeit) = 7200 Sekunden (2
Stunden)
const long utcOffsetInSeconds = 3600;
```

```
char daysOfTheWeek[7][12] = {"Sonntag", "Montag", "Dienstag",
"Mittwoch", "Donnerstag", "Freitag", "Samstag"};
```

```
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP,      "pool.ntp.org",
utcOffsetInSeconds);
```

```
void setup(){
  Serial.begin(115200);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
```

```

    delay(1000);
    Serial.println("Ich verbinde mich mit dem Internet...");
}
Serial.println("Ich bin mit dem Internet verbunden!");

timeClient.begin();
}

void loop() {
    timeClient.update();

    Serial.print(daysOfTheWeek[timeClient.getDay()]);
    Serial.print(", ");
    Serial.println(timeClient.getFormattedTime());

    delay(1000);
}

```

Die Unix Time abfragen

Die Abfrage der Uhrzeit in obigem Format ist schön und gut, wenn du sie z.B. auf einem Display verständlich anzeigen möchtest. Wenn du jedoch mit der Zeit Berechnungen durchführen möchtest, z.B. für einen Countdown, eignet sie sich nicht besonders.

Hier kommt die Unix Time (auch Epoch Times, deutsch: [Unixzeit](#)) ins Spiel. Das sind ganz einfach die vergangenen Sekunden seit dem 1. Januar 1970, 00:00 Uhr UTC.

Ein Beispiel? Der 24. August 2020, 08:00 UTC entspricht der Zahl 1598256000. Ein weiterer Vorteil der Unix Time ist, dass sie überall auf der Welt identisch ist und du dich nicht mit Zeitzonen herumschlagen musst.

Um die Unixzeit abzufragen, benötigst du die Funktion **getEpochTime()**. Ergänze den obigen Sketch wie folgt, um sie im Seriellen Monitor auszugeben:

```
Serial.println(timeClient.getEpochTime());
```

Ein spannendes Projekt, bei dem du die Uhrzeit abfragst, ist ein [Würfel, der leuchtet, wenn die ISS über ihm fliegt](#). Auch hier verwendest du die Unix Time, um den Countdown bis zum Überflug zu berechnen – allerdings kommt die Zeit in diesem Projekt nicht von einem NTP-Server, sondern von einer anderen API.

Apropos NTP: [Hier findest du eine Übersicht weiterer Funktionen der Bibliothek <NTPClient.h>](#).