

Willkommen!

Und vielen Dank, dass Sie sich für unseren AZ-Delivery ESP8266 Mikrocontroller mit integriertem 0,91" OLED-Display entschieden haben. Auf den folgenden Seiten werden wir Sie durch die ersten Programmierschritte führen.

Wir wünschen Ihnen viel Spaß!



Dieses Board mit einem ESP8266-Chip und einer integrierten Ladeschnittstelle für ein Akkupack kann unter NodeMCU und Mikrocontroller-Board programmiert werden.

In diesem eBook behandeln wir den Programmiervorgang auf einem Mikrocontroller-Board.

Das Board hat 32MB Flash-Speicher, WLAN und 12 I/O-Pins (i²c, SPI, PWM und Analog Input). Das Display ist 0,91" groß und hat eine Auflösung von 128 x 32 Pixeln.

Programmieren des ESP8266 mit OLED

Dieser ESP8622 hat im Vergleich zum ESP8266-01 mehr Flash-Speicher und mehr GPIO-Ports.

Bereiten Sie die Software vor:

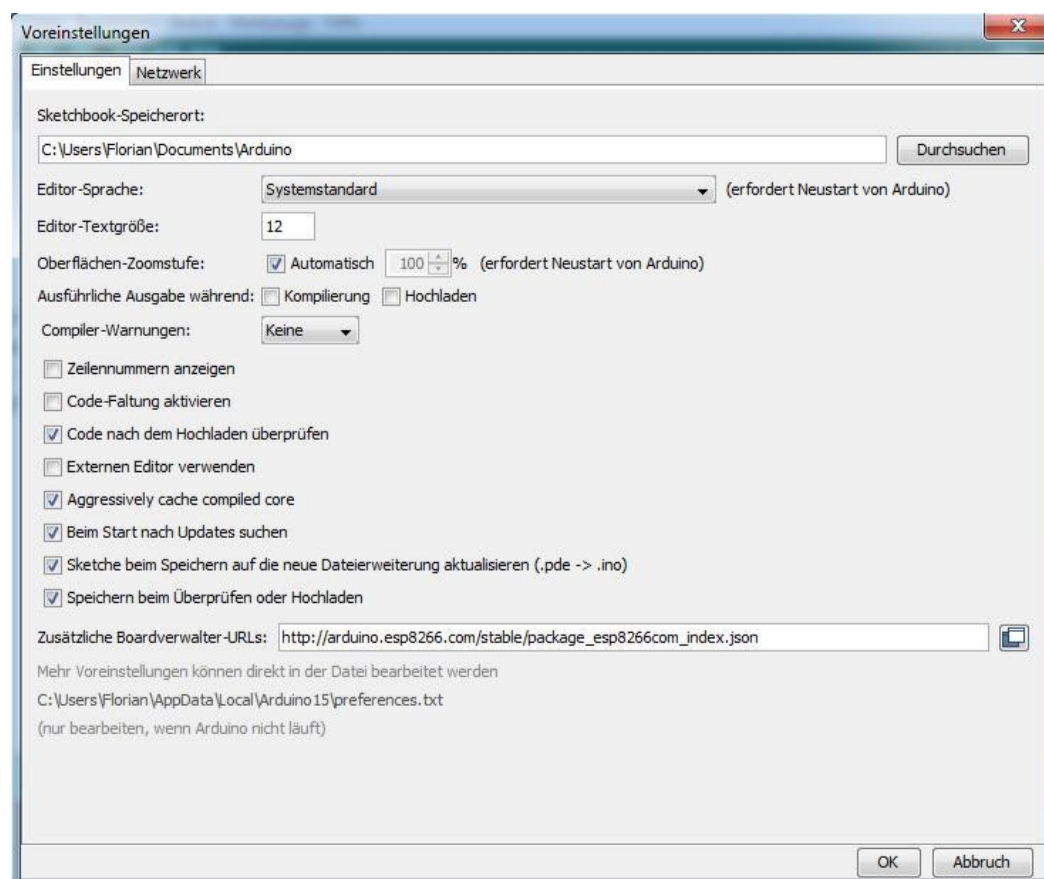
Die Arduino-IDE-Software, die Sie in diesem Schritt sehen und verwenden, wurde bereits installiert. Wenn Sie sie jedoch nicht haben, laden Sie sie bitte von dieser Webadresse herunter und installieren Sie sie auf Ihrem PC:

<https://www.arduino.cc/en/Main/Software#>.

Außerdem sollte auch der Treiber für den CP210x installiert sein. Wenn nicht, dann sollten Sie wissen, dass er mit der Arduino-IDE-Software geliefert wird.

Nachdem alle Grundvoraussetzungen erfüllt sind, können wir nun mit der Installation der Software beginnen. Die Arduino-IDE Software benötigt zunächst alle Informationen über den ESP8266, die wir über die folgende Webadresse unter "Einstellungen" > "Zusätzliche Board-Administratoren-URLs" eingeben können:

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

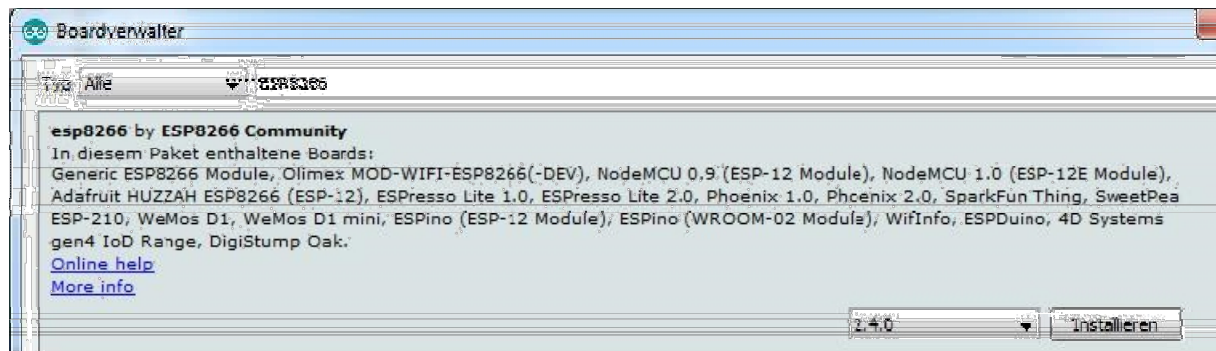


Wenn Sie den Link bereits eingegeben haben, klicken Sie auf die Schaltfläche  und fügen Sie eine neue Zeile in das Fenster ein.

Bestätigen Sie die Eingabe, indem Sie auf "OK" klicken.

Sobald dies abgeschlossen ist, gehen Sie zu "Tools" > "Board" > "Board Administrator" und installieren Sie die ESP8266-Bibliothek. Suchen Sie im Board-Administrator nach "ESP8266", in

die Suchmaschine, die sich oben rechts befindet. Als Ergebnis erhalten Sie das Paket von ESP8266 Gemeinschaft. Wählen Sie dieses ein und klicken Sie auf auf installieren.



Nach erfolgreicher Installation wird nun neben dem Paket *INSTALLED* angezeigt.



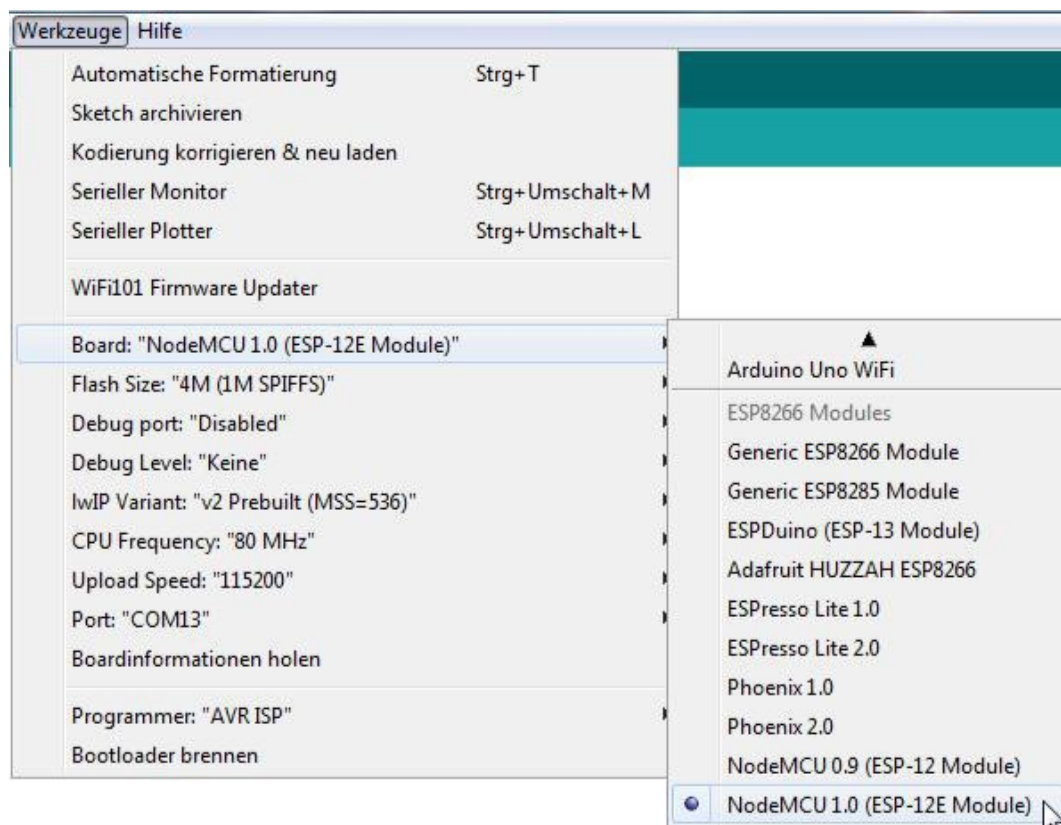
Der nächste Schritt besteht darin, die richtige Karte auszuwählen: Unter Extras

>

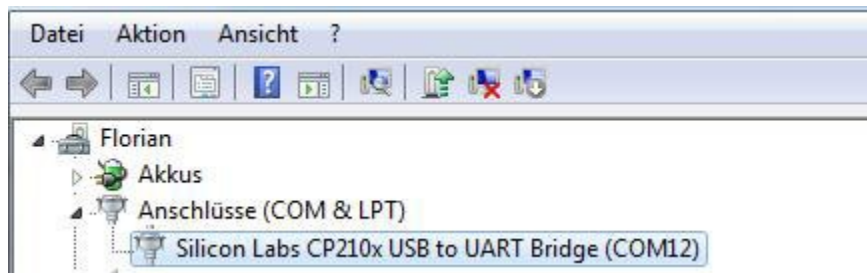
Karte: "Generisches ESP8266-Modul"

Anschluss: "COMxx"

(hier ist Ihr Port des seriellen Adapters)



Der COMPort ist über den Gerätemanager zugänglich und kann auch geändert werden:

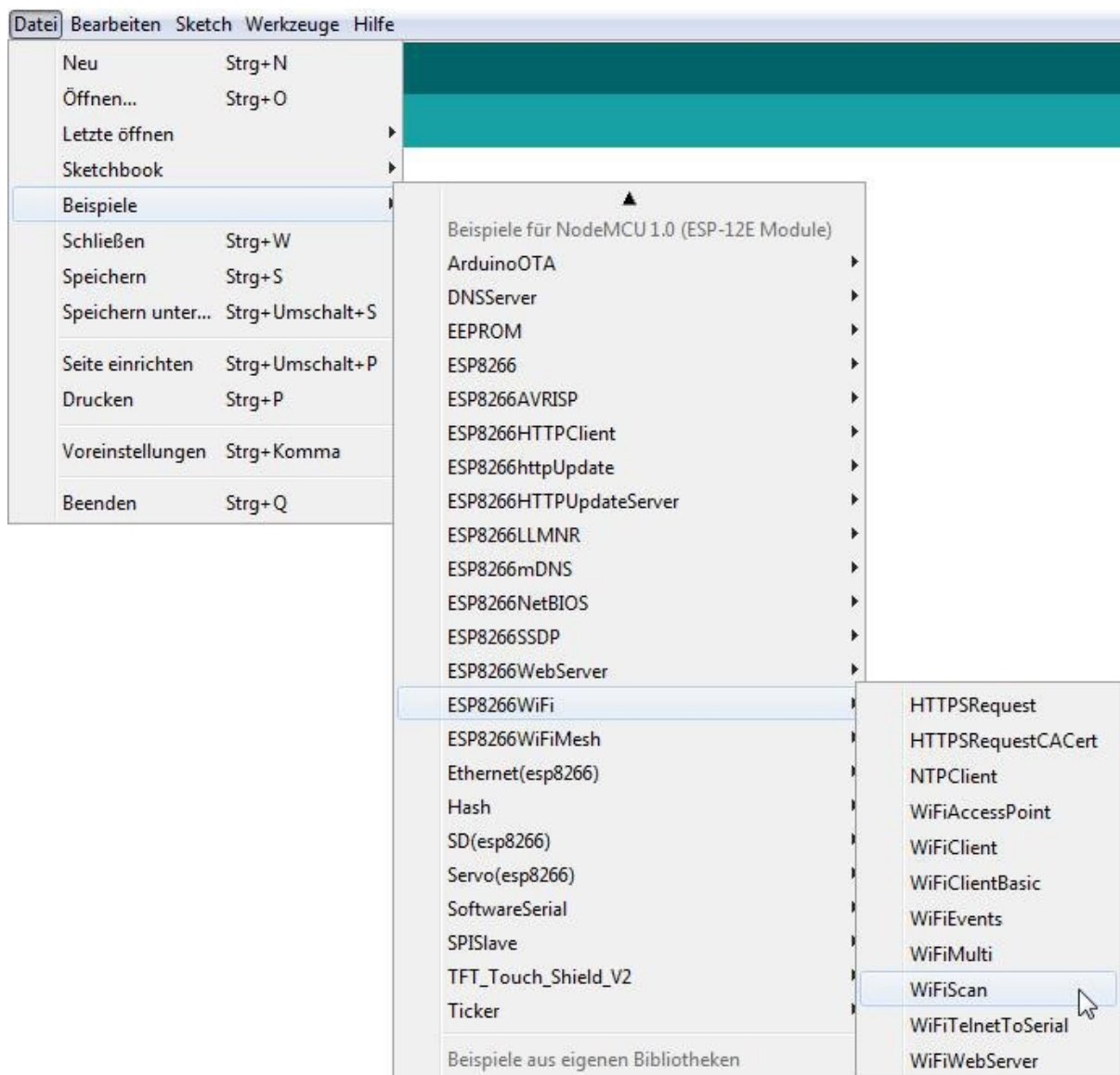


Der Code:

Nun, da die Verkabelung abgeschlossen ist, können wir mit dem Schreiben unseres ersten Codes beginnen.

Wir lassen uns alle Wi-Fi-Netzwerke in unserer Umgebung anzeigen.

Wählen Sie dazu unter *Datei > Beispiele > ESP8266Wifi > **WifiScan***.



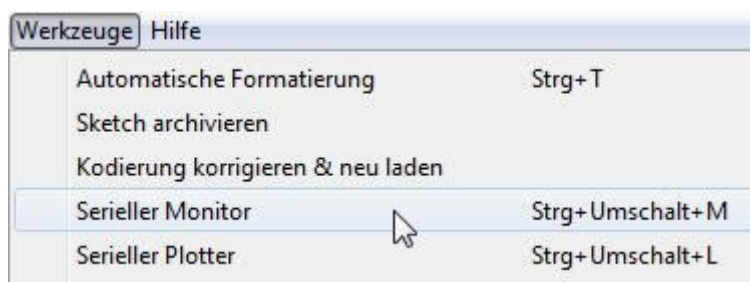
Nachdem der Code angezeigt wurde, klicken wir auf  und verifizieren unser Programm: Wenn alles korrekt ist und unser Programm keine

```
Kompilieren abgeschlossen.  
Der Sketch verwendet 252567 Bytes (24%) des Programmspeicherplatzes. Das Maximum sind 1044464 Bytes.  
Globale Variablen verwenden 33216 Bytes (40%) des dynamischen Speichers, 48704 Bytes für lokale Variablen verbleiben. Das Maximum si
```

Wir können es auf den ESP8622 hochladen. Dazu sollten wir auf  klicken. Nach einer kurzen Zeit wird das Programm auf den Chip geladen:

```
Hochladen...  
Globale Variablen verwenden 33216 Bytes (40%) des dynamischen Speichers, 48704 Bytes für lokale Variablen verbleiben. Das Maximum si  
Uploading 256720 bytes from C:\Users\Florian\AppData\Local\Temp\arduino_build_493968/WiFiScan.ino.bin to flash at 0x00000000
```

Sobald die 100% erreicht sind, ist das Programm vollständig übertragen worden. Öffnen Sie den Seriellen Monitor in der Arduino-IDE Software: Werkzeuge > Serieller Monitor



Alle 5 Sekunden wird automatisch ein neuer Scan gestartet und das Ergebnis wird über die serielle Schnittstelle verteilt.

Zu Beginn eines jeden Scans

Scan-Start

angezeigt, und es endet mit

Scan abgeschlossen.

Schließlich werden alle verfügbaren Wi-Fi-Netzwerke aufgelistet. Sie sieht ähnlich aus wie in der Abbildung rechts.

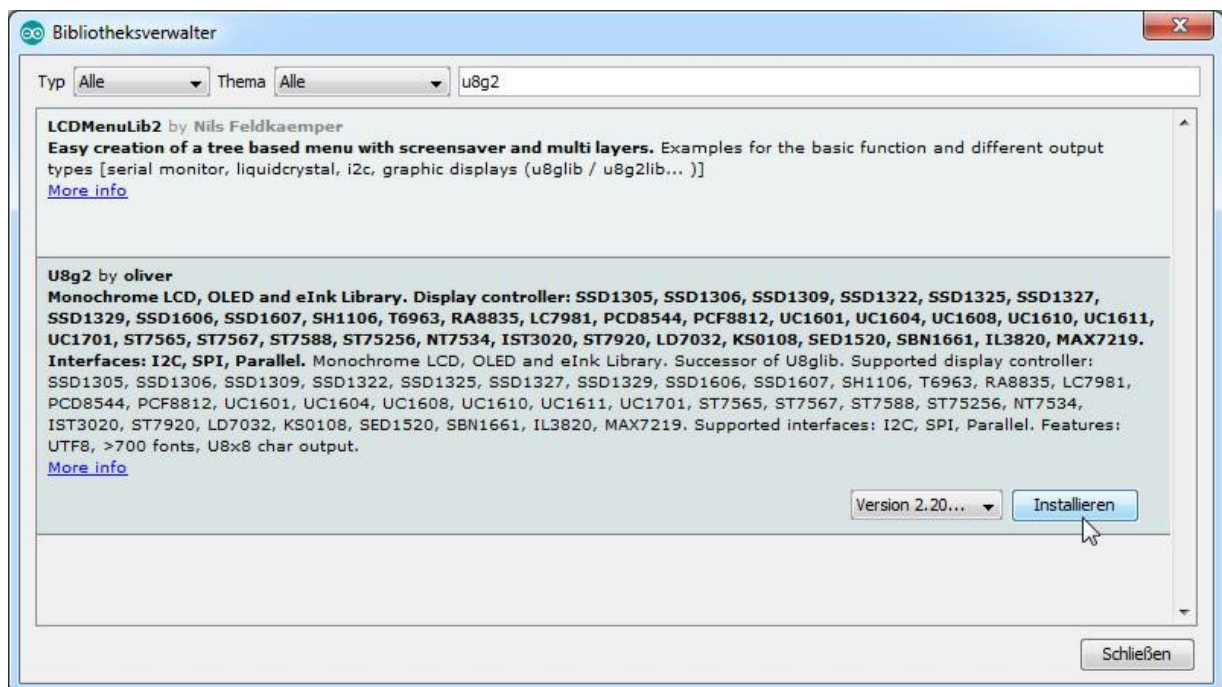
```
COM13  
  
scan start  
scan done  
14 networks found  
1: WLAN-419458 (-78)*  
2: FRITZ!Box Fon WLAN 7170 (-81)*  
3: Fritz-DVBC (-75)*  
4: FRITZ!Box Fon WLAN 7170 (-82)*  
5: Netgear WGR-814 (-60)*  
6: Telekom_FON (-78)  
7: devolo-f4068dbb4335 (-93)*  
8: Netgear WGR-814 (-83)*  
9: FRITZ!Box 7362 SL (-83)*  
10: KabelBox-36C4 (-79)*  
11: Vodafone Hotspot (-80)  
12: Vodafone Homespot (-77)  
13: o2-WLAN05 (-91)*  
14: Netgear WGR-814 (-47)*
```


Steuerung des OLED-Displays:

Um das Display steuern zu können, benötigen wir die entsprechenden Bibliotheken (Informationen) in der Arduino-IDE Software. Beginnen Sie unter Sketch > Bibliothek einbinden > Bibliotheken verwalten ...

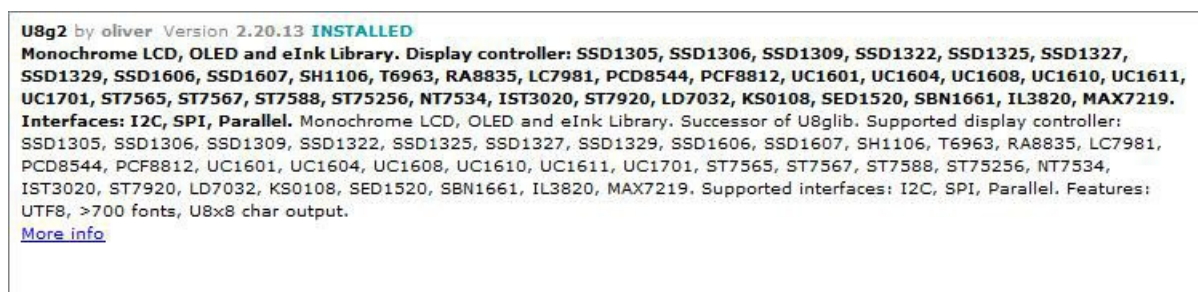


Der Bibliotheksverwalter und suchen Sie dort nach "u8g2".



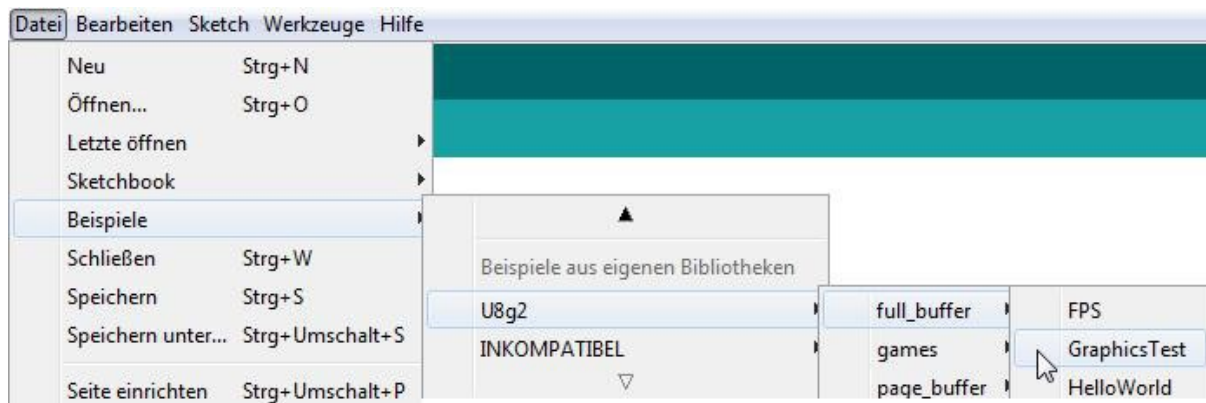
Nachdem Sie das Paket ausgewählt haben, klicken Sie unten rechts auf Installieren.

Nachdem Sie einige Sekunden gewartet haben, erscheint die Meldung "INSTALLIERT".



Jetzt können Sie das Fenster schließen und mit dem Programmervorgang beginnen.

Wählen Sie unter *Beispiele* > *U8g2* > *full_buffer* > **GraphicsTest**:



Nun wird ein längerer Code geöffnet. In den ersten Zeilen werden viele Anzeigetypen registriert. Diese haben jedoch die Zeichen "//" am Anfang der Zeile. Für unsere Anzeige müssen wir nun diese Zeile suchen und aktivieren, indem wir die "//"-Zeichen am Anfang der Zeile entfernen:

```
U8G2_SSD1306_128X32_UNIVISION_F_HW_I2C u8g2(U8G2_R0, /* reset=*/  
U8X8_PIN_NONE); // Adafruit ESP8266/32u4/ARM Boards + FeatherWing OLED
```

Nach der Übertragung  zeigt das Display nun Demo-Texte und -Bilder an.

Auf der Grundlage dieser Demonstration können wir auch einen Lauftext programmieren.

Für eine reibungslose Wiedergabe sollten wir die CPU-Frequenz auf 160MHz und den SPI-Speicher auf 3MB (3M SPIFFS) einstellen.

```
Board: "NodeMCU 1.0 (ESP-12E Module)"  
Flash Size: "4M (3M SPIFFS)"  
Debug port: "Disabled"  
Debug Level: "Keine"  
lwIP Variant: "v2 Prebuilt (MSS=536)"  
CPU Frequency: "160 MHz"  
Upload Speed: "115200"  
Port: "COM13"
```

Hinweis: Wenn Sie einen längeren Lauftext erstellen möchten, müssen Sie die 16-Bit-Unterstützung in der Datei u8g2.h aktivieren. Die Datei finden Sie im

Verzeichnis: Arduino

libraries\arduino_168079\src\clib\u8g2.h

In Zeile 72 befindet sich `//#define U8G2_16BIT`

Es sollte geändert werden in `#define`

`U8G2_16BIT`

Speichern Sie anschließend die Datei und übertragen Sie den Code erneut.

Der Code lautet wie folgt:

```
#include <U8g2lib.h>
U8G2_SSD1306_128X32_UNIVISION_F_HW_I2C u8g2(U8G2_R0, /* reset= */
U8X8_PIN_NONE);
u8g2_uint_t offset;
u8g2_uint_t width;
const char *text = "Florian ";
void setup(void) {
    u8g2.begin();
    u8g2.setFont(u8g2_font_logisoso32_tf);
    width = u8g2.getUTF8Width(text);
    u8g2.setFontMode(0);
}

void loop(void) {
    u8g2_uint_t x;
    u8g2.firstPage();
    do {
        x = offset;
        u8g2.setFont(u8g2_font_logisoso32_tf);
        do {
            u8g2.drawUTF8(x, 32, text);
            x += width;
        } while ( x < u8g2.getDisplayWidth() );
        u8g2.setFont(u8g2_font_logisoso32_tf);
        u8g2.setCursor(0, 64);
        u8g2.print(width);
    } while ( u8g2.nextPage() );
    offset -= 1;
    if ( (u8g2_uint_t)offset < (u8g2_uint_t) - width ){
        offset = 0;
        delay(1000);
        u8g2.clearBuffer();
    }
}
```

**Sie haben es geschafft! Sie können jetzt Ihre Projekte mit
ESP8266-OLED programmieren und verwirklichen!**

Jetzt ist es an der Zeit zu experimentieren.

Und für weitere Hardware steht Ihnen unser Online-Shop jederzeit zur
Verfügung:

<https://az-delivery.de>

Viel Spaß!

Impressum

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