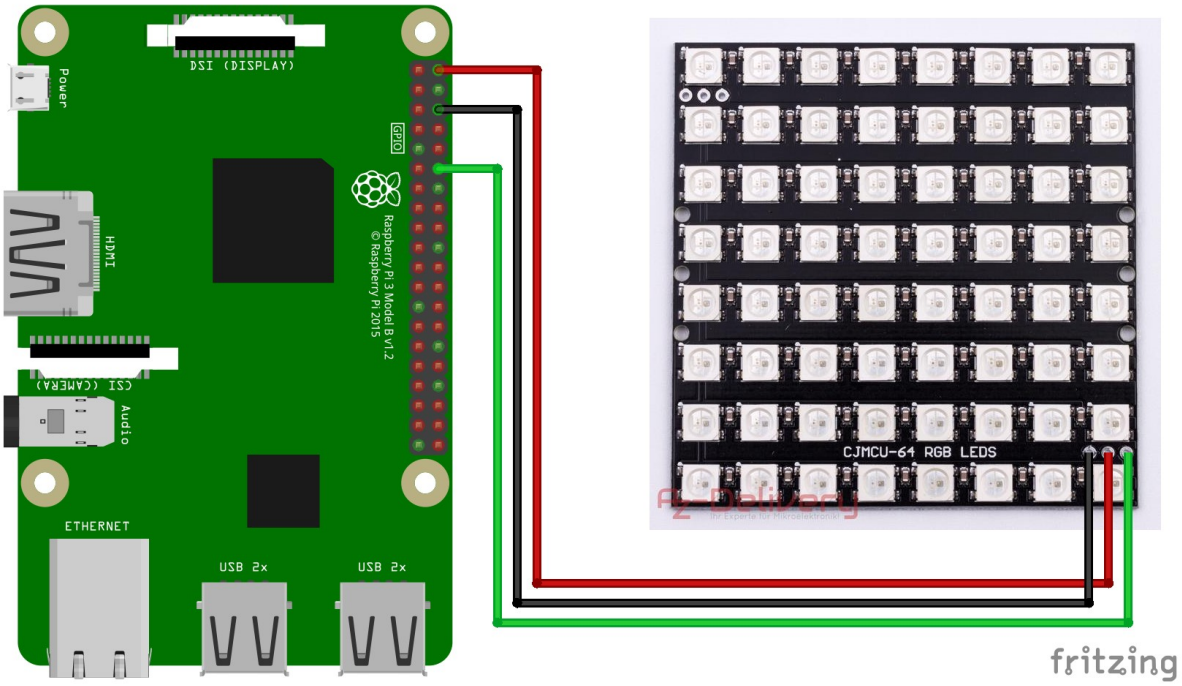
Verdrahten des Panels mit dem Raspberry Pi:



Das-Panel hat nur 3 Anschlüsse, +5V, GND und DIN.

+5V wird mit **PIN 2 (5V)** am Raspberry verbunden	Rote Leitung
GND wird mit **PIN 6 (GND)** verbunden	Schwarze Leitung
DIN wird mit **PIN 12 (GPIO 18)** verbunden	Grüne Leitung

Nachdem alles verdrahtet ist kann der Raspberry Pi gestartet werden.

Wenn du mehr als nur ein Panel verwenden möchtest, kannst du DOUT mit DIN vom nächsten Panel verbinden. Du solltest dann aber auch ein externes 5V Netzteil verwenden, ansonsten kann der Raspberry Pi überlastet werden.

Zur Information: Diese Anleitung basiert auf dem Raspberry Pi Image vom 29.11.2017 (Stretch - Lite) – Updates können leichte Modifikationen der Anleitung notwendig machen.

Alternativ zu den von uns beschriebenen Pins am Raspberry kann jeder Masse-Pin verwendet werden.

„Programmieren“ des Raspberry Pi:

Bevor man auf dem Raspberry Pi Software installiert, sollte der Raspberry Pi noch auf den aktuellsten Stand gebracht werden:

```
sudo apt-get update
sudo apt-get upgrade
```

```
Do you want to continue? [Y/n]
```
-> **y** (Y eingeben und mit Enter bestätigen)

Nachdem der Raspberry Pi nun aktuell ist können wir Software installieren.

```
sudo apt-get install gcc make build-essential python-dev git scons swig
```

```
gcc, make, build-essential, scons und swig:    Kompilierungswerkzeuge
python-dev:             Entwicklungsumgebung für Python
git:        Downloader für github
```

Do you want to continue? [Y/n] -> y

wenn alles fertig installiert ist, laden (`git clone`) wir uns von git die Software für die Ansteuerung der WS2812 LED. Anschließend muss die Software noch kompiliert und installiert werden.

```
git clone https://github.com/jgarff/rpi_ws281x
cd rpi_ws281x/
sudo scons
cd python
sudo python setup.py build
sudo python setup.py install
```

im Ordner „examples“ sind nun einige Beispiele der Ansteuerung mitgeliefert.

Die Beispieldateien müssen nur noch etwas angepasst werden.

```
sudo nano examples/strandtest.py
```

```
  GNU nano 2.7.4                File: examples/strandtest.py                Modified

import time

from neopixel import *

import argparse
import signal
import sys
def signal_handler(signal, frame):
        colorWipe(strip, Color(0,0,0))
        sys.exit(0)

def opt_parse():
        parser = argparse.ArgumentParser()
        parser.add_argument('-c', action='store_true', help='clear the display on exit')
        args = parser.parse_args()
        if args.c:
                signal.signal(signal.SIGINT, signal_handler)

# LED strip configuration:
LED_COUNT      = 64      # Number of LED pixels.
LED_PIN        = 18      # GPIO pin connected to the pixels (18 uses PWM!).
#LED_PIN        = 10      # GPIO pin connected to the pixels (10 uses SPI /dev/spidev0.0).
LED_FREQ_HZ    = 800000  # LED signal frequency in hertz (usually 800khz)
LED_DMA        = 10      # DMA channel to use for generating signal (try 10)
LED_BRIGHTNESS = 255     # Set to 0 for darkest and 255 for brightest
LED_INVERT     = False   # True to invert the signal (when using NPN transistor level shift)
LED_CHANNEL    = 0       # set to '1' for GPIOs 13, 19, 41, 45 or 53
LED_STRIP      = ws.WS2811_STRIP_GRB   # Strip type and colour ordering
```

Hier suchen wir nach der Zeile die mit LED_COUNT beginnt und ändern nach dem „=“ die Anzahl unserer LEDs auf 64. Solltestest du mehr LEDs (mehrere Panels) haben, dann gebe hier die entsprechende Anzahl ein.

LED_COUNT   = 64

Mit der Tastenkombination STRG + O speichern wir die Datei und STRG + X beenden wir die Eingabe.

Nun können wir unser erstes Beispielprogramm ausführen:

```
sudo python examples/strandtest.py
```

Es werden nun diverse Animationen abgespielt und die LED leuchten und blinken in allen Farben. Dies soll dir demonstrieren, wie schnell die LEDs angesteuert werden können und viele Möglichkeiten für deinen Einsatzzweck sich ergeben können. Mit STRG – C kannst du die Beispieldatei wieder beenden. Nun kannst du die anderen Beispieldateien ebenfalls ansehen und dann auch deine Projekte verwirklichen. Viel Spass!

Verdrahten des Panels mit einem □8□□□□□□y□b□N□□

+5V wird mit **5V** am Atmega328p verbunden — Rote Leitung
GND wird mit **GND** verbunden — Schwarze Leitung
DIN wird mit **D3 (PWM)** verbunden — Grüne Leitung
Nachdem alles verdrahtet ist kann der Atmega328p mit Spannung versorgt werden.

Wenn du mehr als nur ein Panel verwenden möchtest, kannst du DOUT mit DIN vom nächsten Panel verbinden. Du solltest dann aber auch ein externes 5V Netzteil verwenden, ansonsten kann der Atmega328p oder dein PC (USB Anschluss) überlastet werden.

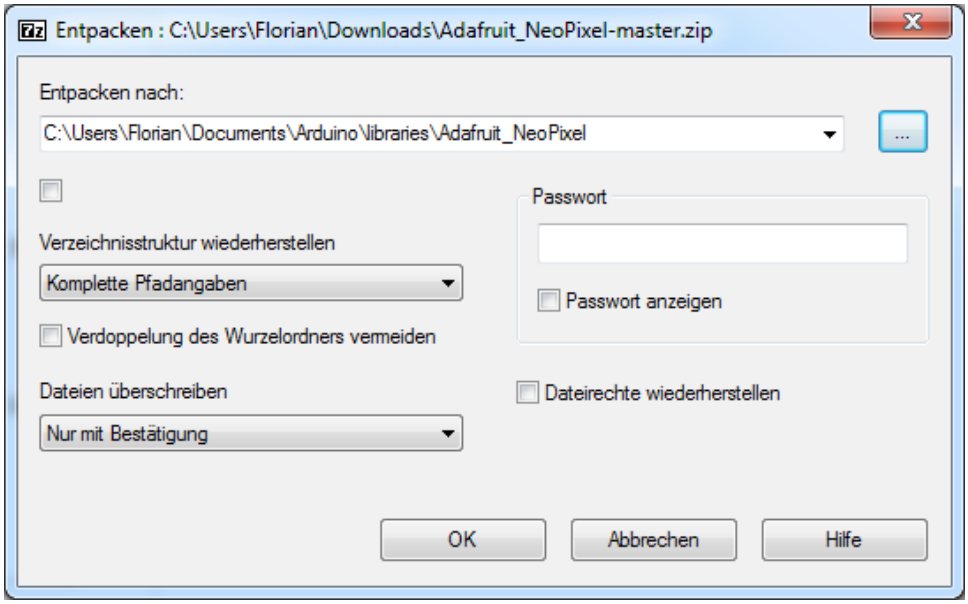
„Programmieren“ des □8□□□□□□y□b□N□□

Bevor wir mit dem Programmieren beginnen können, müssen wir zuerst von git (Adafruit) die entsprechenden Bibliotheken herunterladen und installieren.

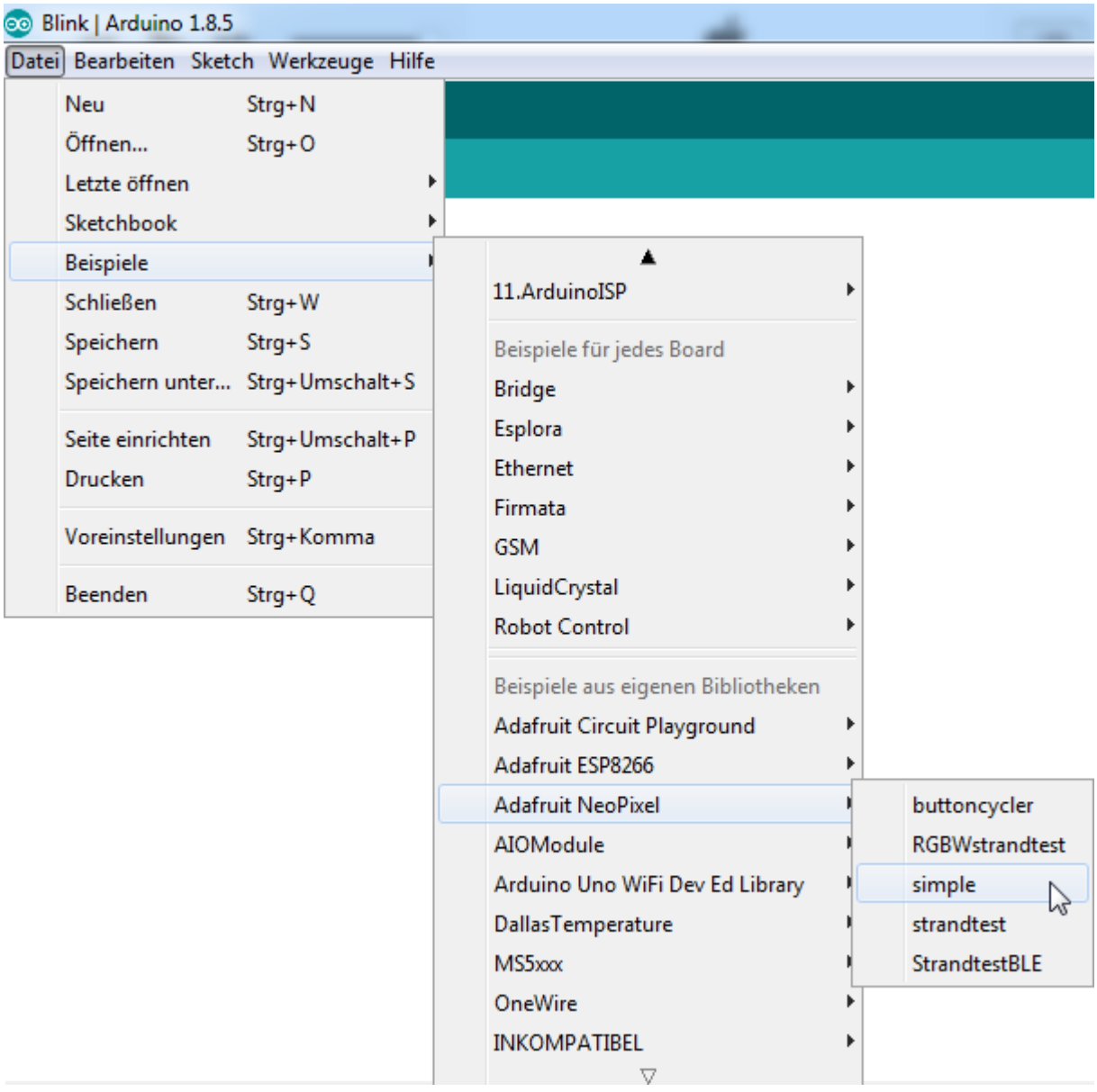
Laden wir die Ressourcen herunter:
https://github.com/adafruit/Adafruit_NeoPixel/archive/master.zip

Diese Zip Datei entpacken (mit 7zip) wir in den Ordner: [Eigenes Userverzeichnis (C:\ Benutzer\Florian\] \ Eigene Dokumente \ Arduino \ libraries \ Adafruit_NeoPixel

Hinweis: Sollten diese Ordner nicht existieren, dann lege diese einfach neu an.



Nach der Installation starten wir die Arduino-IDE Software und öffnen ein Beispiel:



Die nun geöffnete „simple“ Datei muss an 2 Stellen angepasst werden:

Und zwar

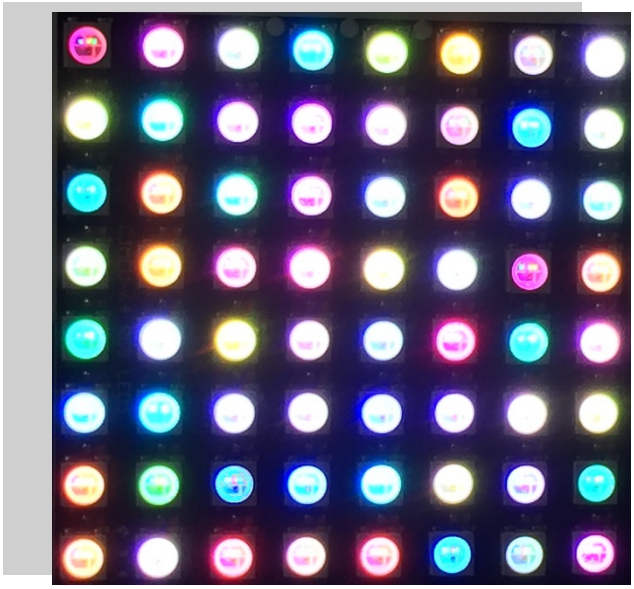
#define PIN 3

#define NUMPIXELS 64

Dein Atmega328p Board unter der Boardverwaltung sollte auch schon richtig konfiguriert sein, dann kannst du mit dem Übertragen beginnen.

Nach dem Übertragen werden alle 64 LEDs der Reihe nach angesteuert und leuchten Grün.

Nun kannst du deine eigenen Projekte verwirklichen und eigene Animationen anzeigen lassen.



Du hast es geschafft dein LED-Panel leuchtet!

Ab jetzt heißt es lernen und eigene Projekte verwirklichen.

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

https://az-delivery.de

Viel Spaß!
Impressum

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