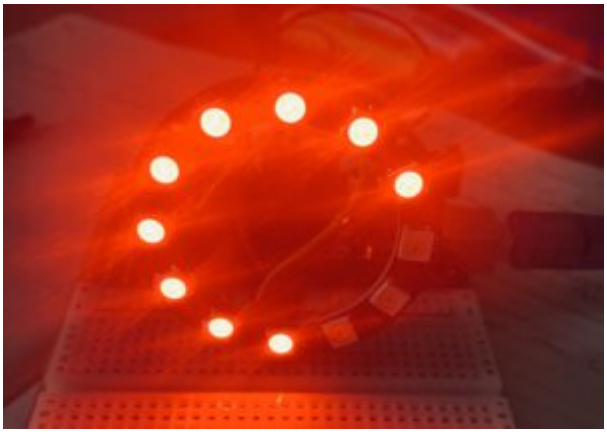
NeoPixel RGB LED Ring anschließen & verwenden



In diesem Tutorial erfährst du, wie du den Adafruit NeoPixel RGB LED Ring (oder einen kompatiblen Ring)* an deinem Arduino anschließt und verwendest. Den Ring gibt es mit unterschiedlich vielen LEDs, wir verwenden hier einen Ring mit 12.

Das Besondere am NeoPixel ist, dass du jede LED einzeln ansteuern und ihr auch einen individuellen RGB-Farbton zuweisen kannst. So kannst du ganz einfach Animationen mit tollen Farbverläufen zaubern.

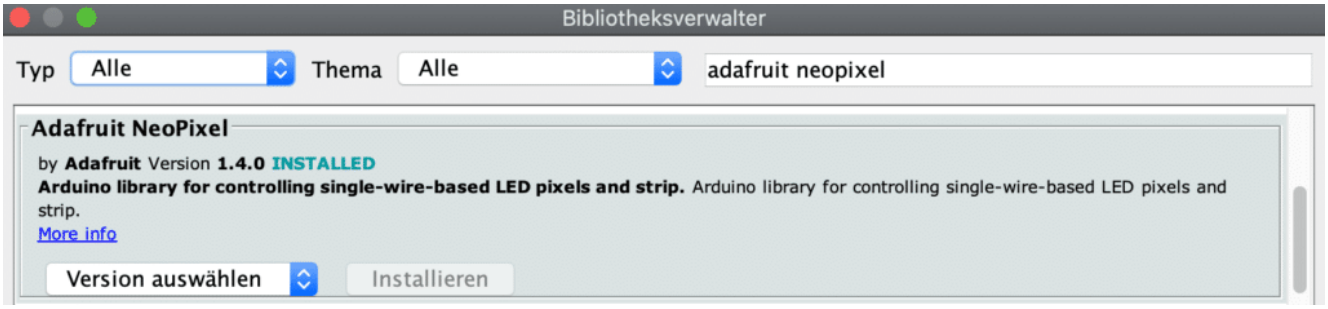
Vorbereitungen & Anschluss des RGB LED Rings

In den meisten Fällen erhältst du den NeoPixel LED Ring ohne Pins. **Um ihn verwenden zu können, musst du selbige also zunächst an die Anschlüsse GND, 5V und IN (oder auch DI) löten.** Pass hierbei auf, dass du die Anschlüsse auf dem Ring nicht allzu lange erhitzt, ansonsten können sie sich lösen und das Bauteil wird unbrauchbar.

Wenn die Anschlüsse sitzen, verbinde GND und 5V mit den gleichnamigen Pins an deinem Arduino. Der Pin IN (oder DI) kommt an einen Digitalpin deiner Wahl – in unserem Beispiel-Sketch ist das der Pin 2.

Die passende Bibliothek

Wie für viele Bauteile gibt es auch für den NeoPixel RGB LED Ring eine Bibliothek, die dir das Leben erleichtert. Öffne deinen Bibliotheksmanager und suche nach **Adafruit Neopixel**. Installiere anschließend die neueste Version:



Den RGB LED Ring ansteuern

Jetzt kann es auch schon losgehen. Kopiere den folgenden Sketch und lade ihn auf deinen Arduino. Wenn alles passt, leuchten alles LEDs auf deinem Ring nacheinander rot auf und gehen zusammen wieder aus.

```
//Bibliothek einbinden
#include <Adafruit_NeoPixel.h>

int leds = 12; //Anzahl der LEDs
int ledPin = 2; //Pin, an dem der NeoPixel angeschlossen ist

int red = 255; //Rot
int green = 0; //Grün
int blue = 0;  //Blau

//NeoPixel als "pixels" instanziieren
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(leds, ledPin,
NEO_GRB + NEO_KHZ800);

void setup() {
  pinMode (ledPin, OUTPUT);
  pixels.begin();
  pixels.setBrightness(100); //Helligkeit: 0 (aus) - 255
}

void loop() {
  for (int i = 0; i < 12; i++) {
    pixels.setPixelColor(i, pixels.Color(red, green, blue));
    pixels.show();
    delay(500);
  }
```

```
  pixels.clear();
  delay(500);
}
```

Gehen wir den Sketch Schritt für Schritt durch: Als erstes bindest du die Bibliothek ein und definierst einige Variablen für die Anzahl der LEDs, den Anschluss des Rings und die Farbwerte.

```
#include <Adafruit_NeoPixel.h>

int leds = 12; //Anzahl der LEDs
int ledPin = 2; //Pin, an dem der LED Ring angeschlossen ist

int red = 255; //Rot
int green = 0; //Grün
int blue = 0;  //Blau
```

Mit einer RGB LED kannst du über 16 Millionen verschiedene Farben ausgeben, die sich aus den drei Einzelfarben Rot, Grün und Blau zusammensetzen. Für jede dieser Farben kannst du einen Wert zwischen 0 und 255 bestimmen – die Mischung macht’s!

Mit diesem Tool kannst du die passenden Einzelwerte für deine Wunschfarbe bestimmen.

Als nächstes erstellst du die Instanz **pixels**:

```
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(leds, ledPin, NEO_GRB + NEO_KHZ800);
```

Hier definierst du die Anzahl der LEDs auf deinem Ring mit Hilfe der Variablen **leds** und den Pin **ledPin**, an den dein NeoPixel RGB LED Ring angeschlossen ist. Die anderen Parameter sind erst einmal nicht wichtig.

Die Setup-Funktion

Hier definierst du den **ledPin** als **OUTPUT** und aktivierst den LED Ring. Zuletzt legst du die Helligkeit der LEDs fest:

```
void setup() {
  pinMode (ledPin, OUTPUT);
  pixels.begin();
  pixels.setBrightness(100); //Helligkeit: 0 (aus) - 255
}
```

Der Loop

Jede LED hat eine eigene Nummer von 0 bis 11, mit der du sie individuell ansteuerst. Gezählt wird normalerweise von der LED unten an den Anschlüssen im Uhrzeigersinn.

In diesem Tutorial schaltest du die LEDs nacheinander mit einer kurzen Verzögerung an. Hierfür bietet sich ein For-Loop an:

```
void loop() {
  for (int i = 0; i < 12; i++) {
    pixels.setPixelColor(i, pixels.Color(red, green, blue));
    pixels.show();
    delay(500);
  }
  pixels.clear();
  delay(500);
}
```

Innerhalb des For-Loops legst du zunächst die Farbe fest, die die LED **i** ausgeben soll. Hierfür stehen in der Funktion **pixels.Color()** die Einzelwerte, die du zu Beginn des Sketchs definiert hast:

```
pixels.setPixelColor(i, pixels.Color(red, green, blue));
```

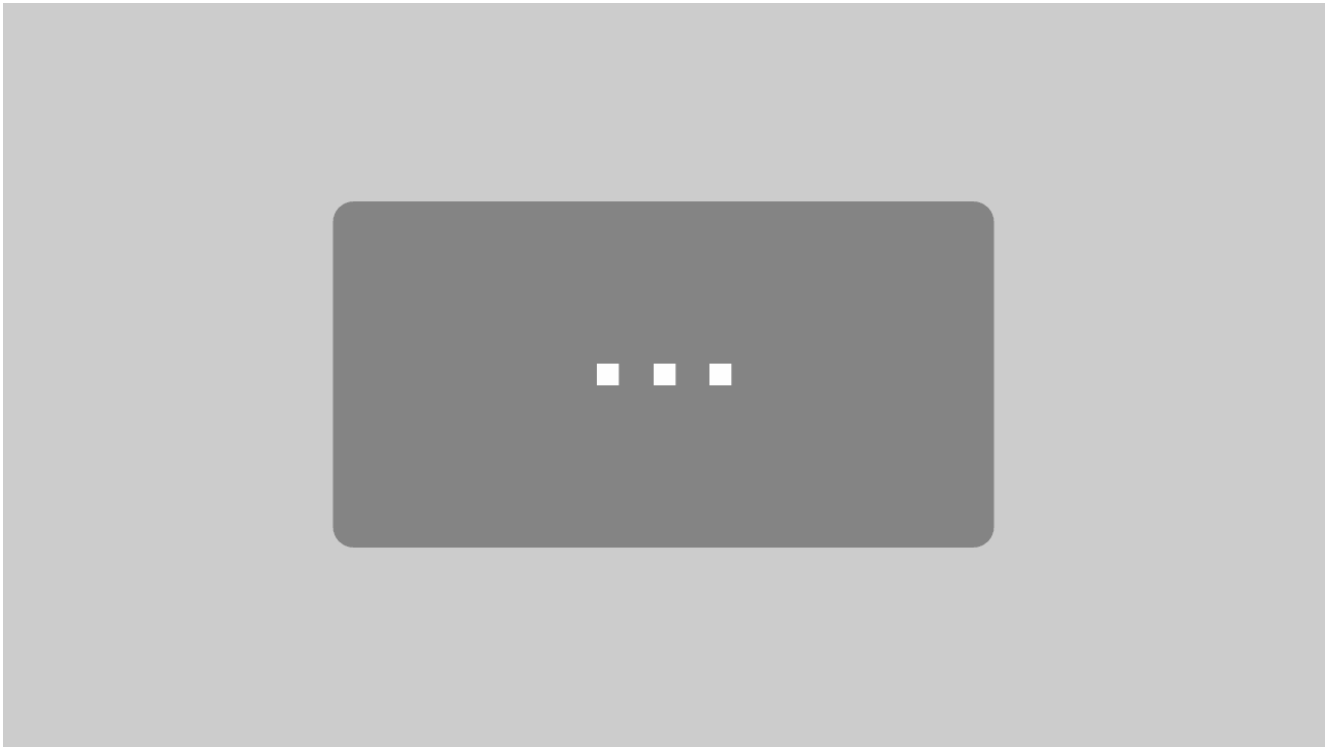
Anschließend bringst du die LED mit **pixels.show()** zum Leuchten und wartest 500 Millisekunden. Anschließend kümmert sich der

For-Loop um die nächste LED **i**.

Sind alle 12 LEDs erleuchtet springt der Sketch aus dem For-Loop und schaltet alle LEDs mit **pixels.clear()** wieder aus. Nach weiteren 500 Millisekunden beginnt dann alles wieder von vorne.

Wie geht es weiter?

Jetzt kennst du den Code für eine einfache Animation in einer Farbe. Deiner Fantasie sind jedoch keine Grenzen gesetzt, wie wäre es z.B. mit einer LEGO ISS, die für die Dauer des Überflug der echten ISS die Farbe wechselt? Du findest den Code für den ISS Tracker hier auf pollux labs.



Weitere Ideen findest du in unseren Arduino-Projekten.