

## Leuchtoobjekt Pflanze



### Leuchtoobjekt, Pflanze

#### Aufgabe

Erfinde und baue ein Leuchtoobjekt, eine Leuchtpflanze ganz nach deinen Vorstellungen.

#### Material

6-8 LEDs in verschiedenen Farben, Widerstände 180  $\Omega$  oder 200  $\Omega$ , Schalt draht, Arduino Uno Board, verkupferter Stahldraht (Autogenschweissdraht), Lötzinn, Holzbrett, farbige Mäppli, farbiges Papier, ev. Heissleim

#### Verfahren

Draht biegen und trennen, Schrauben, Weichlöten, Abisolieren, Holz sägen, Folie und Papier schneiden und kleben.

#### Vorgehen

Ideensammlung: Halte sechs Ideen in Farbe und in Skizzen fest.

Bespreche die Ideen mit deiner LP. Kläre dabei die Grösse und Umsetzbarkeit.

Studiere die Arbeitsschritte, organisiere dein Material und starte mit dem Bau des Objekts.

Starte nun deine ChatGPT/Arduino Umgebung. Nun gibst du den ersten Prompt in ChatGPT ein.

Wichtig ist, die Anschlüsse am Board zu erfragen. Danach beschreibst/erfragst du dein Lichtdesign

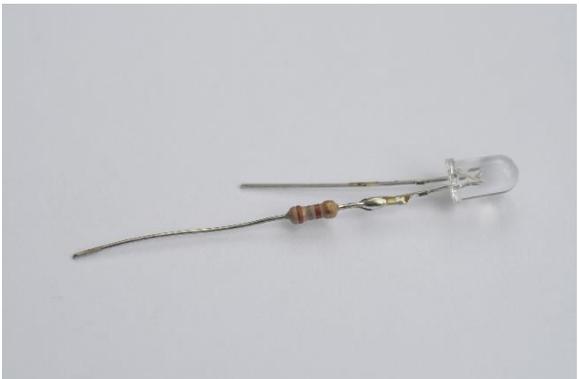
passend zu deinem Objekt. Wie sollen deine LEDs Leuchten, zufällig in verschiedenen Mustern usw. Du kannst im ChatGPT-Thread immer wieder nachfragen und das Lichtdesign anpassen bis es für dich stimmt.

## Tutorial Leuchtojektbau



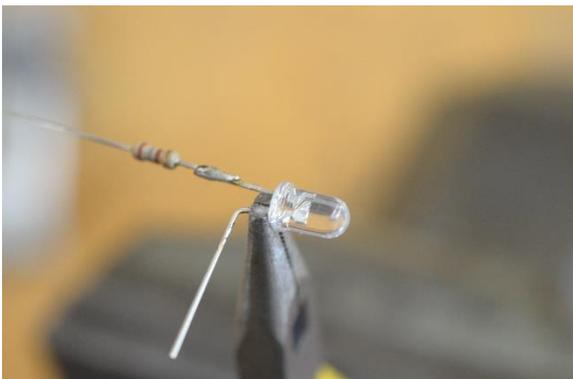
1

Mit verschiedenen Zangen und dem Schraubstock biegest du das Grundgerüst für dein Leuchtojekt. Das Zusammendrehen der 1,5mm Autogenschweisdrähte stabilisiert deine Konstruktion. Die Drahringe und Holzschrauben dienen der Befestigung auf einem Holzbrett.



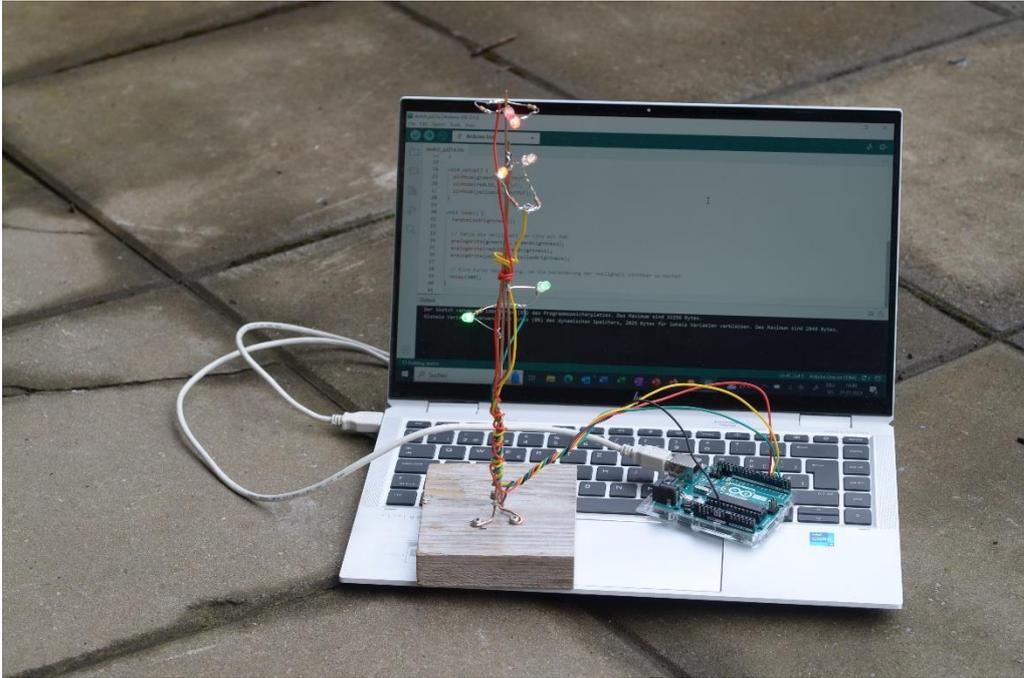
2

LEDs haben ein längeres (Pluspol) und ein kürzere (Minuspol) Beinchen. An das Längere, den Pluspol wird der Widerstand angelötet.



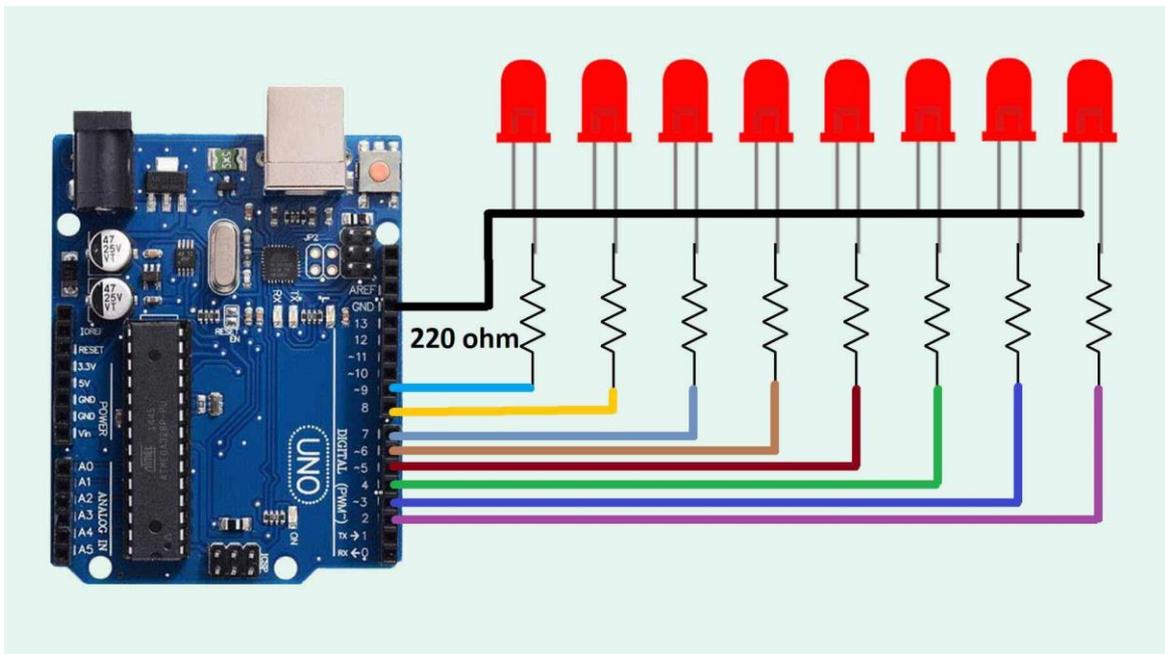
3

Mit einer Spitzzange werden, wenn nötig, die Beinchen nahe am Gehäuse gehalten und abgebogen. Das verhindert ein Abbrechen des Drahtes. Das Minusbeinchen wird direkt an die Konstruktion gelötet. Die Konstruktion dient gleichzeitig als elektrischer Leiter. Wenn LEDs nahe beieinander platziert sind können die Pluspole nach den Widerständen zusammengelötet (zusammengefasst) werden. Der weiterführende Draht muss natürlich isoliert sein, auch dürfen blanke Pluspoldrähte die Konstruktion NICHT berühren!



4

Die Verdrahtung kann nach Vorschlag von ChatGPT vorgenommen werden. Der Minuspol (GND) wird mit der Konstruktion verbunden, die verschiedenen LED-Gruppen werden mit Ausgangspins verbunden. Mit einem USB Kabel wird das Arduino Board mit dem PC verbunden. Damit werden die von ChatGPT generierten Codes via Arduino IDE auf das Board geladen, ebenfalls wird das Board damit mit Strom versorgt.



5

Die Abbildung zeigt den Anschluss von 8 LEDs. Dabei hat jede LED ihren Widerstand. In dieser Schaltung werden alle LEDs an einem eigenen Ausgang angeschlossen. Das ist bereits eine Erweiterung und ein entsprechender Prompt könnte bei ChatGPT eingegeben werden. Alle Minusbeinchen sind zusammengefasst auf GND geführt.

Weitere Ideen mit Petflaschen oder Kunst die Meere rettet



[Veronika Richterová](#)

## Folgend findest du die ganze Konversation mit ChatGPT und den immer wieder angepassten Arduino Codes

TO

Ich möchte drei LEDs in den Farben grün, rot und gelb an ein Arduino Uno Board anschliessen. Was muss ich beachten und an welche Anschlüsse werden die LED angeschlossen.

ChatGPT

Das Anschließen von LEDs an ein Arduino Uno Board ist ein einfaches Projekt, aber es gibt ein paar wichtige Dinge, die du beachten solltest. Hier sind die Schritte, um grüne, rote und gelbe LEDs an deinem Arduino Uno anzuschließen:

Materialien, die du benötigst:

1. Arduino Uno Board
2. Grüne LED
3. Rote LED
4. Gelbe LED
5. Widerstände (Vorwiderstände) - etwa 220 Ohm für jede LED
6. Steckbrett (Breadboard) und Jumper-Kabel

Anschluss der LEDs:

1. Schließe das Arduino Uno Board an deinen Computer an.
2. Platziere die grüne LED auf dem Steckbrett. Die längere Anschlussbein (positives Bein) der LED sollte in Richtung der oberen Seite des Steckbretts zeigen. Verbinde das positive Bein mit dem Vorwiderstand (220 Ohm) und verbinde das andere Ende des Widerstands mit Pin D2 des Arduino.
3. Platziere die rote LED auf dem Steckbrett. Die längere Anschlussbein (positives Bein) sollte ebenfalls in Richtung der oberen Seite des Steckbretts zeigen. Verbinde das positive Bein mit einem weiteren Vorwiderstand (220 Ohm) und verbinde das andere Ende des Widerstands mit Pin D3 des Arduino.
4. Platziere die gelbe LED auf dem Steckbrett. Die längere Anschlussbein (positives Bein) sollte in Richtung der oberen Seite des Steckbretts zeigen. Verbinde das positive Bein mit einem weiteren Vorwiderstand (220 Ohm) und verbinde das andere Ende des Widerstands mit Pin D4 des Arduino.

Hier ist eine schematische Darstellung, wie die LEDs und Widerstände auf dem Steckbrett verbunden werden sollten:

```
css
+----> [LED] ----> [220 Ohm] ----> [Pin D2]
|
+----> [LED] ----> [220 Ohm] ----> [Pin D3]
|
+----> [LED] ----> [220 Ohm] ----> [Pin D4]
```

Programmierung:

1. Öffne die Arduino-Entwicklungsumgebung auf deinem Computer.
2. Erstelle ein neues Projekt und schreibe den folgenden Code, um die Pins D2, D3 und D4 als Ausgang zu konfigurieren:

```
void setup() {  
  pinMode(2, OUTPUT); // Pin D2 als Ausgang  
  pinMode(3, OUTPUT); // Pin D3 als Ausgang  
  pinMode(4, OUTPUT); // Pin D4 als Ausgang  
}
```

3. Schreibe den folgenden Code, um die LEDs in verschiedenen Farben leuchten zu lassen:

```
void loop() {  
  digitalWrite(2, HIGH); // Grüne LED einschalten  
  delay(1000);           // 1 Sekunde warten  
  digitalWrite(2, LOW);  // Grüne LED ausschalten  
  delay(500);            // 0.5 Sekunden warten  
  
  digitalWrite(3, HIGH); // Rote LED einschalten  
  delay(1000);           // 1 Sekunde warten  
  digitalWrite(3, LOW);  // Rote LED ausschalten  
  delay(500);            // 0.5 Sekunden warten  
  
  digitalWrite(4, HIGH); // Gelbe LED einschalten  
  delay(1000);           // 1 Sekunde warten  
  digitalWrite(4, LOW);  // Gelbe LED ausschalten  
  delay(500);            // 0.5 Sekunden warten  
}
```

4. Lade den Code auf dein Arduino Uno Board hoch, indem du auf den Upload-Button in der Arduino-Entwicklungsumgebung klickst.

Jetzt sollten die LEDs abwechselnd in den Farben grün, rot und gelb leuchten. Beachte, dass das Beispielprogramm die LEDs einzeln einschaltet. Du kannst das Programm nach Belieben anpassen, um verschiedene Muster oder Farbkombinationen zu erzeugen.

TO

Danke die LEDs sollen ihre Helligkeit verändern aber nie ganz löschen. Das heller und dunkler werden der LEDs kann zufällig passieren, es sollen aber immer alle LEDs aktiv sein. Kennst du einen Code dazu?

ChatGPT

Ja, ich kann dir einen Code geben, der die Helligkeit der LEDs zufällig verändert, aber niemals ganz ausschaltet. Dafür verwenden wir die sogenannte Pulsweitenmodulation (PWM), die es uns ermöglicht, die Helligkeit der LEDs zu steuern.

Hier ist der Code, der zufällig die Helligkeit der grünen, roten und gelben LEDs ändert, während alle LEDs aktiv bleiben:

```
// Definition der LED-Pins  
const int greenLED = 2;  
const int redLED = 3;  
const int yellowLED = 4;
```

```

// Initialisierung der Helligkeit der LEDs
int greenBrightness = 128;
int redBrightness = 128;
int yellowBrightness = 128;

// Funktion zur Zufälligen Änderung der Helligkeit
void randomizeBrightness() {
  greenBrightness += random(-10, 11);
  redBrightness += random(-10, 11);
  yellowBrightness += random(-10, 11);

  // Stelle sicher, dass die Helligkeit im gültigen Bereich von 0 bis 255
  liegt
  greenBrightness = constrain(greenBrightness, 0, 255);
  redBrightness = constrain(redBrightness, 0, 255);
  yellowBrightness = constrain(yellowBrightness, 0, 255);
}

void setup() {
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);
  pinMode(yellowLED, OUTPUT);
}

void loop() {
  randomizeBrightness();

  // Setze die Helligkeit der LEDs mit PWM
  analogWrite(greenLED, greenBrightness);
  analogWrite(redLED, redBrightness);
  analogWrite(yellowLED, yellowBrightness);

  // Eine kurze Verzögerung, um die Veränderung der Helligkeit sichtbar zu
  machen
  delay(200);
}

```

In diesem Code verwenden wir die Funktion `analogWrite(pin, value)`, um die Helligkeit der LEDs zu steuern. Der Wert `value` liegt normalerweise zwischen 0 (aus) und 255 (volle Helligkeit). Durch die Verwendung von `random(-10, 11)` fügen wir einen zufälligen Wert zwischen -10 und 10 zur aktuellen Helligkeit hinzu, um die Veränderung zufällig zu machen. Die Funktion `constrain(value, min, max)` sorgt dafür, dass die Helligkeit immer im Bereich von 0 bis 255 bleibt.

Dieser Code lässt die LEDs abwechselnd heller und dunkler werden, ohne sie jemals ganz auszuschalten. Du kannst die Verzögerungszeit in `delay(200)` anpassen, um die Geschwindigkeit der Helligkeitsänderungen zu ändern.

TO

Danke, es funktioniert soweit allerdings verändert sich Ausgang drei rot nicht, rot leuchtet immer---

Es tut mir leid für das Missverständnis. Ich habe einen Fehler gemacht, indem ich die PWM-fähigen Pins für die rote und gelbe LED falsch ausgewählt habe. Der Arduino Uno hat nur PWM-fähige Pins auf den digitalen Pins 3, 5, 6, 9, 10 und 11. Da Pin 3 bereits von der grünen LED belegt ist, müssen wir einen anderen Pin für die rote LED verwenden.

Hier ist der korrigierte Code, der die rote LED an Pin 5 anschließt:

```
// Definition der LED-Pins
const int greenLED = 2;
const int redLED = 5; // Geändert zu Pin 5
const int yellowLED = 4;

// Initialisierung der Helligkeit der LEDs
int greenBrightness = 128;
int redBrightness = 128;
int yellowBrightness = 128;

// Funktion zur Zufälligen Änderung der Helligkeit
void randomizeBrightness() {
  greenBrightness += random(-10, 11);
  redBrightness += random(-10, 11);
  yellowBrightness += random(-10, 11);

  // Stelle sicher, dass die Helligkeit im gültigen Bereich von 0 bis 255
  liegt
  greenBrightness = constrain(greenBrightness, 0, 255);
  redBrightness = constrain(redBrightness, 0, 255);
  yellowBrightness = constrain(yellowBrightness, 0, 255);
}

void setup() {
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);
  pinMode(yellowLED, OUTPUT);
}

void loop() {
  randomizeBrightness();

  // Setze die Helligkeit der LEDs mit PWM
  analogWrite(greenLED, greenBrightness);
  analogWrite(redLED, redBrightness);
  analogWrite(yellowLED, yellowBrightness);

  // Eine kurze Verzögerung, um die Veränderung der Helligkeit sichtbar zu
  machen
  delay(200);
}
```

Jetzt sollte die rote LED wie die anderen LEDs ihre Helligkeit ändern und niemals ganz ausgeschaltet sein. Wenn du den Code auf deinen Arduino Uno hochlädst, sollte alles wie erwartet funktionieren. Viel Spaß beim Experimentieren mit den LEDs!

TO

Danke das funktioniert. Rot ist aber immer noch dominant, kannst du das Random so verändern, dass grün und gelb häufiger leuchten?

ChatGPT

Natürlich kann ich das anpassen. Wir können die Werte, die wir zu den Helligkeiten der LEDs hinzufügen, so ändern, dass grün und gelb häufiger leuchten als rot. Zum Beispiel könnten wir größere Zahlen für die Helligkeitsänderung von grün und gelb verwenden, um sie heller zu machen, während wir kleinere Zahlen für rot verwenden, um es weniger hell zu machen.

Hier ist der modifizierte Code, der grün und gelb häufiger leuchten lässt:

```
// Definition der LED-Pins
const int greenLED = 2;
const int redLED = 5; // Geändert zu Pin 5
const int yellowLED = 4;

// Initialisierung der Helligkeit der LEDs
int greenBrightness = 128;
int redBrightness = 128;
int yellowBrightness = 128;

// Funktion zur Zufälligen Änderung der Helligkeit
void randomizeBrightness() {
  // Grün und Gelb häufiger leuchten lassen
  greenBrightness += random(-15, 16);
  redBrightness += random(-5, 6);
  yellowBrightness += random(-15, 16);

  // Stelle sicher, dass die Helligkeit im gültigen Bereich von 0 bis 255
  // liegt
  greenBrightness = constrain(greenBrightness, 0, 255);
  redBrightness = constrain(redBrightness, 0, 255);
  yellowBrightness = constrain(yellowBrightness, 0, 255);
}

void setup() {
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);
  pinMode(yellowLED, OUTPUT);
}

void loop() {
  randomizeBrightness();

  // Setze die Helligkeit der LEDs mit PWM
  analogWrite(greenLED, greenBrightness);
  analogWrite(redLED, redBrightness);
  analogWrite(yellowLED, yellowBrightness);

  // Eine kurze Verzögerung, um die Veränderung der Helligkeit sichtbar zu
  // machen
  delay(200);
}
```

In diesem Code haben wir die Zahlen, die wir zu den Helligkeiten der LEDs hinzufügen, geändert. Grün und gelb können nun Werte von -15 bis +15 erhalten, während rot nur Werte von -5 bis +5 erhält. Dadurch leuchten grün und gelb häufiger und dominieren die Helligkeitsveränderungen, während rot weniger häufig verändert wird.

Jetzt sollten grün und gelb häufiger heller und dunkler werden als rot. Du kannst diese Werte natürlich nach Belieben anpassen, um das gewünschte Verhalten zu erzielen. Viel Spaß beim Experimentieren!