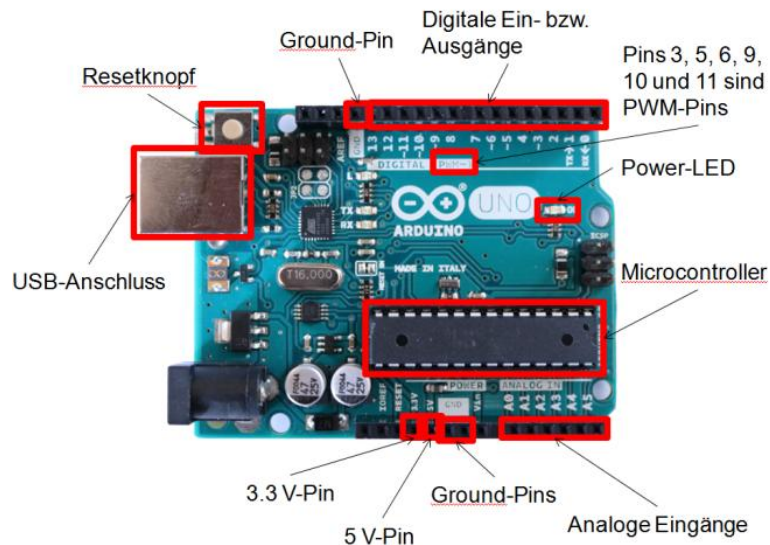


Einführung in die Welt des Arduinos

Hardware:



Software:

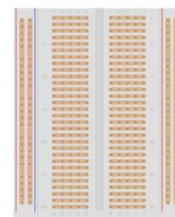
Download: <https://www.arduino.cc/en/main/software>

Referenz unter <https://www.arduino.cc/en/Reference/HomePage>

Weitere Hardware:

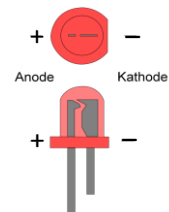
Breadboard:

Äußere Reihen sind vertikal, innere horizontal durch Metallschienen miteinander verbunden.

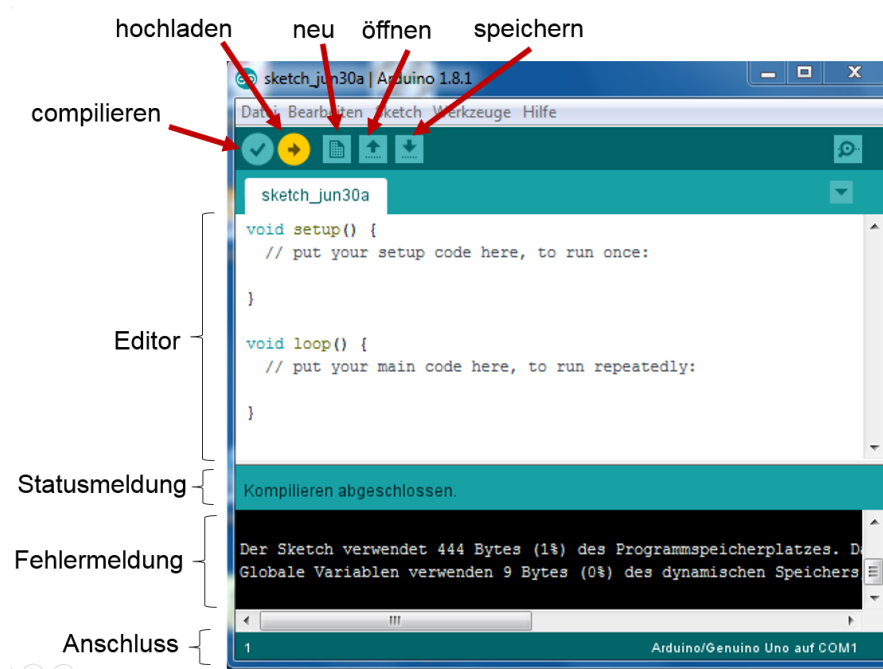


LED:

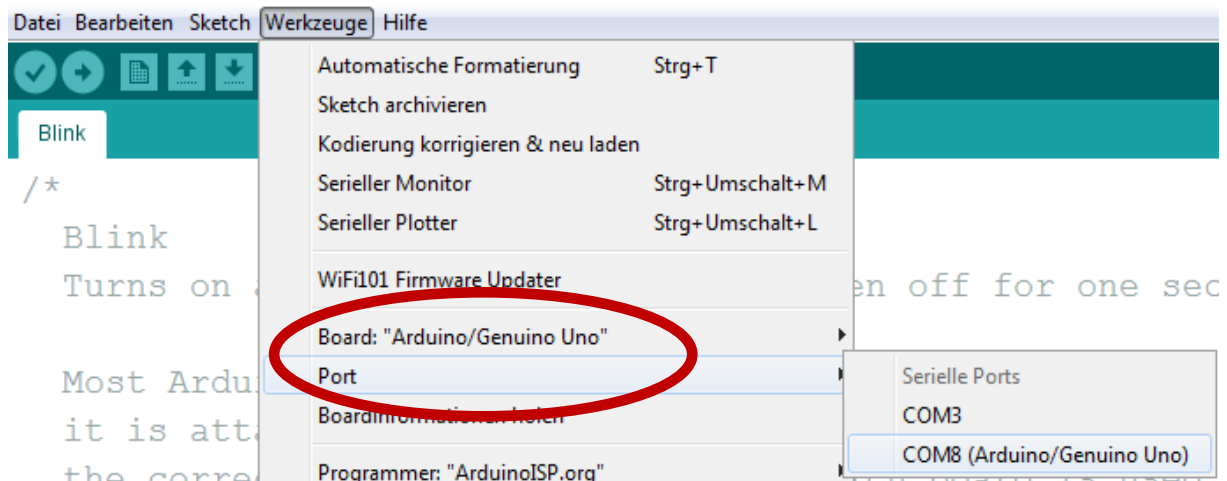
Am längeren Beinchen liegt der Pluspol (Anode), am kürzeren Beinchen bzw der Minuspol (Kathode). Bei der Kathode ist die LED zusätzlich etwas abgeflacht.



Arduino-IDE



Einstellungen:



Leuchten von LEDs:

Im Arduino Uno besitzen die Stromkreise i. A. eine Spannung von 5 Volt. Durch diese Spannung kann eine LED kaputt gehen, deshalb wird vor diese ein sogenannter Vorwiderstand (220 Ohm) angeschlossen.

Hinweis: In einem Stromkreis fließt der Strom von Plus nach Minus,

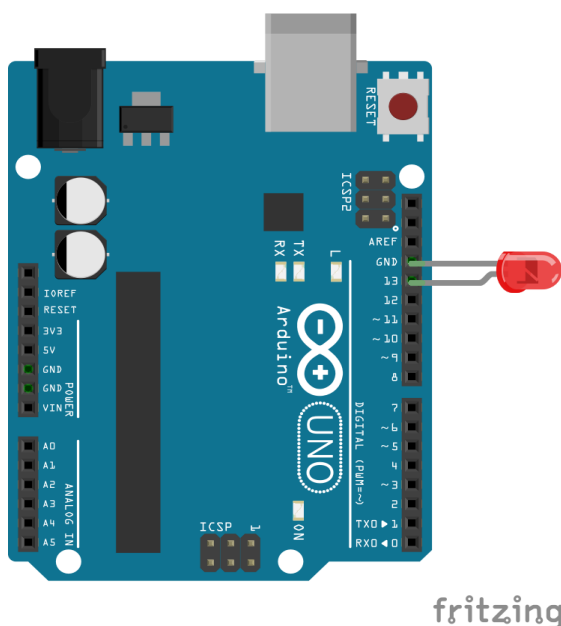
Built-In-LED blinken lassen:

An Pin 13 ist sowohl eine auf dem Arduino enthaltene LED angeschlossen, als auch schon ein Vorwiderstand eingebaut.

Das in der Arduino-IDE enthaltene Beispiel kann also ohne zusätzliche Hardware auf den Arduino geladen werden, die Built-In-LED blinkt.

LED an Pin 13 ohne Steckbrett anschließen:

Wie oben erwähnt, ist an Pin 13 kein zusätzlicher Vorwiderstand notwendig.

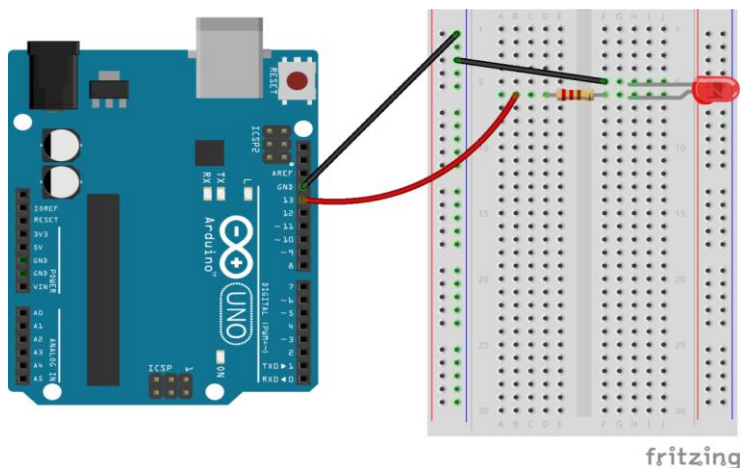


LED an Pin 13 mit Breadboard:

Eine LED wird mit Hilfe eines Breadboards und eines Vorwiderstandes an Pin 13 angeschlossen.

Hinweis:

Da der „Ground“ relativ häufig gebraucht wird, schließt man ihn i. A. einmal mit der blauen vertikalen Reihe des Steckbretts an und verbindet dann alle erforderlichen horizontalen Steckplätze mit dieser vertikalen Reihe (vgl. die schwarzen Kabel, ähnlich wird mit dem 5 Volt Pin verfahren (dann anstatt zu blau zu rot)).



Der zugehörige Code:

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

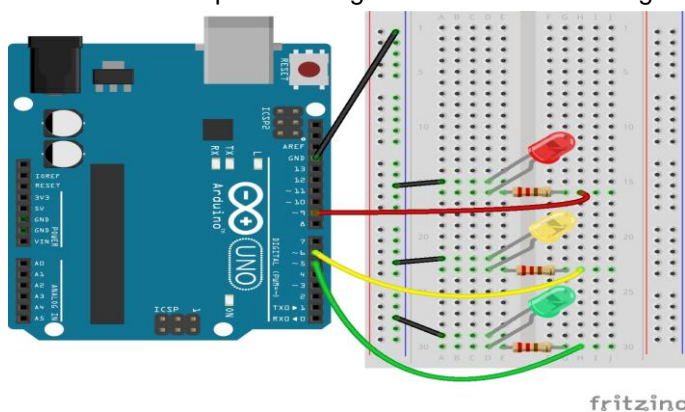
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);                     // wait for a second
  digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);                     // wait for a second
}
```

Hinweis:

Anstatt LED_BUILTIN hätte genauso gut die Zahl 13 geschrieben werden können.

Aufgabe: Ampelschaltung:

Erstelle eine Ampelschaltung und lasse die LEDs originalgetreu blinken.



LED dimmen:

Um eine LED zu dimmen, müsste die Spannung reduziert werden. Dies ist bei einem Arduino nicht möglich. Deshalb wird das menschliche Auge durch schnelles An- und Ausschalten getäuscht. Je länger die LED an ist, desto heller erscheint sie, je kürzer sie an ist, desto dunkler. Dieses Verfahren nennt man Pulsweitenmodulation, kurz PWM.

Alle durch eine Tilde (~) gekennzeichneten Pins können PWM.

Der zugehörige Befehl lautet:

```
analogWrite(Pinnummer, Helligkeit);
```

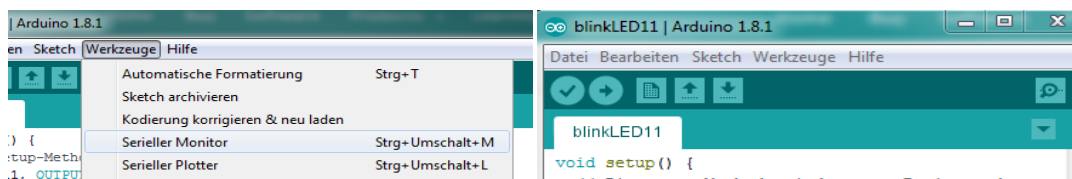
Helligkeit ist dabei ein Wert zwischen 0 und 255 (also 1 Byte).

Beispiel:

```
for (int wert=0; wert <=255; wert = wert +1) {
  analogWrite(ledPin, wert);
  delay(20);
}
```

Serieller Monitor:

Die Ausgabe von Werten ist über den „seriellen Monitor“ möglich. Start der Ausgabe erfolgt über das Menü Werkzeuge oder die Lupe am oberen rechten Bildschirmrand.



Der serielle Monitor wird in der Setup-Methode aktiviert:

```
Serial.begin(9600)
```

Die Zahl 9600 ist die Baudrate.

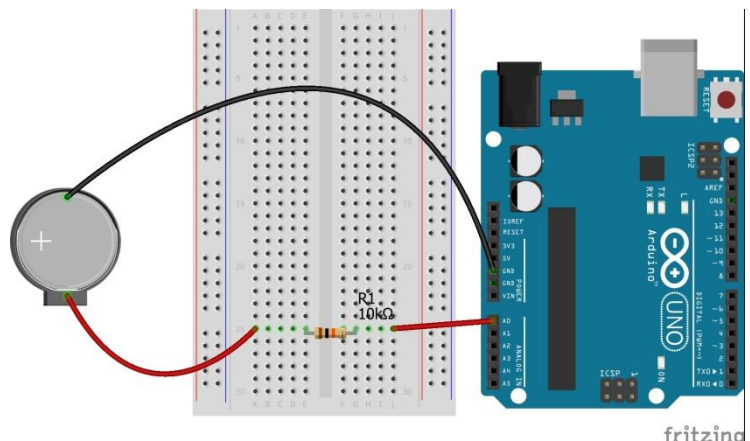
In der Loop-Methode erfolgt die eigentliche Ausgabe mit Hilfe folgender Befehle:

```
Serial.print(„Text“); //Ausgabe ohne Zeilenumschaltung
Serial.println(„Text“); //Ausgabe mit Zeilenumschaltung
```

Batterietester:

Die Spannung einer Batterie soll gemessen werden. Dazu wird der Pluspol einer Batterie über einen Widerstand mit einem analogen Eingangspin (A0 bis A5) verbunden, der Minuspol der Batterie mit GND.

Achtung: Der Arduino kann max. 5 Volt lesen, deshalb nie eine höhere Batterie testen!



batterietester

```
int batterie;
int batPin = 0;
float spannung;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  batterie = analogRead(batPin);
  spannung = 0.0049 * batterie;
  Serial.println(spannung);
  delay(1000);
}
```

$spannung = (batterie * 5) / 1023 = 0,0049 * batterie$

Der analoge Eingangspin liefert einen Wert zwischen 0 und 1023 (10 Bit). Die mögliche Spannungsbereich, die der Arduino verarbeiten kann ist von 0 Volt bis 5 Volt. Deshalb gilt:

0 \triangleq 0 Volt
1023 \triangleq 5 Volt

$batterie \triangleq x \text{ Volt}$

Der gemessene Wert (abgespeichert in der Variablen batterie) muss demnach in den möglichen Spannungsbereich umgewandelt werden. Es gilt:

Fotowiderstand:



Der Fotowiderstand ist ein Lichtsensor, dieser ändert seinen Wert in Abhängigkeit der Lichtstärke. Je heller es ist, desto größer ist der Widerstand.

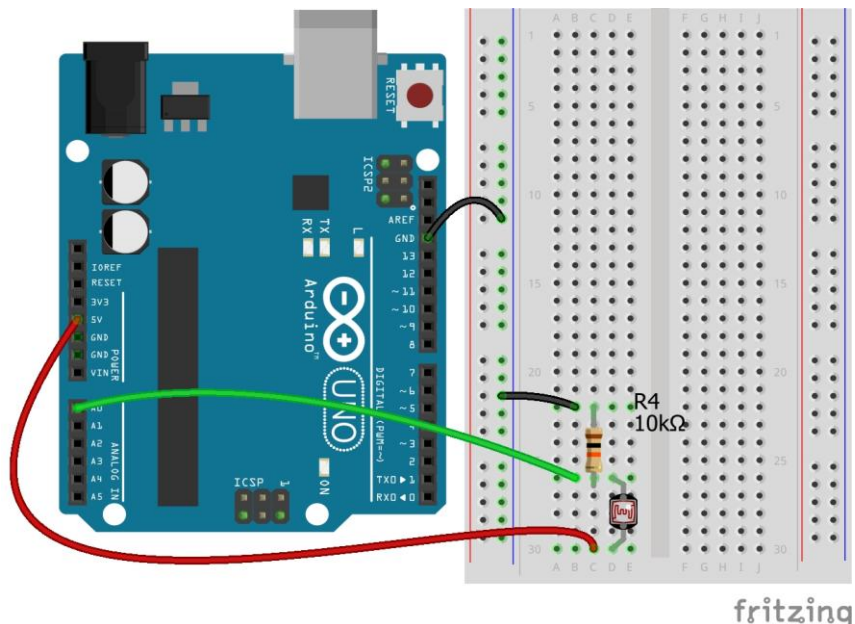
Um die Helligkeit bestimmen zu können, muss also sein Widerstand an einem analogen Eingangspin (A0 ... A5) gemessen werden. Der Pin liefert einen 10-Bit-Wert, also eine Zahl zwischen 0 und 1023.

Bei einem Fotowiderstand muss ein Referenzwiderstand geschaltet werden.

Begründung umgangssprachlich: Viele Elektronen schwirren um den Pin, bei voller Verdunkelung kommen keine neuen Elektronen mehr hinzu, die schon vorhandenen entweichen jedoch nur langsam über Kriechströme. Dies führt dazu, dass der Arduino sehr langsam reagiert. Er muss somit über einen Referenzwiderstand geerdet werden. D. h. der Fotowiderstand wird mit dem GND verbunden, dazwischen wird ein Referenzwiderstand geschaltet. Der

Referenzwiderstand sollte in etwa gleichen Widerstand wie Fotowiderstand besitzen (z. B. 10 kOhm).

Die Versorgung des Fotowiderstands erfolgt über 5 Volt, d. h. ein Ende muss mit 5 Volt verbunden werden. Das andere mit dem analogen Eingangspin. Der Referenzwiderstand wird zwischen dem Fotowiderstand und dem Eingangspin mit GND geerdet.

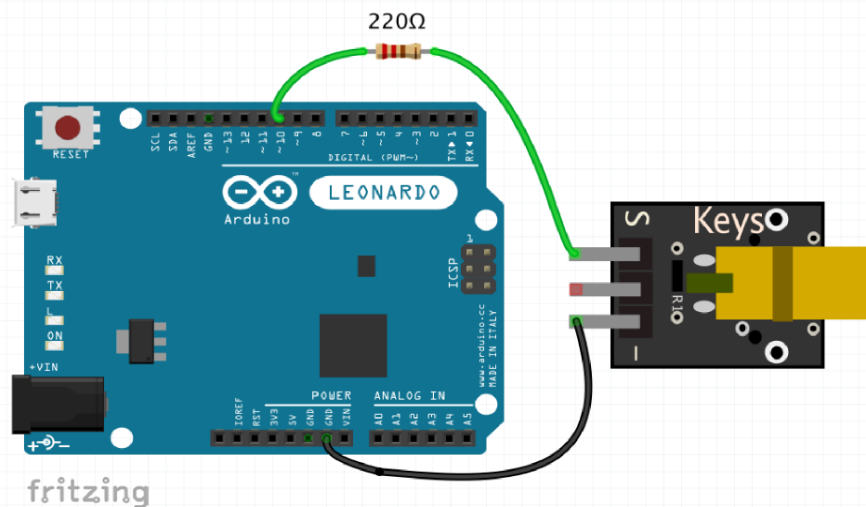
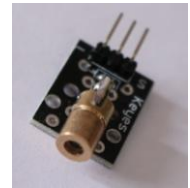


Aufgabe Fotowiderstand:

Lass dir die gemessenen Werte eines Fotowiderstand über den seriellen Monitor anzeigen.

Lasermodul

Ein Lasermodul wird wie eine LED behandelt. Ein Fotowiderstand ist dabei nicht dringend notwendig, der Laserstrahl ist dann jedoch nicht so intensiv.

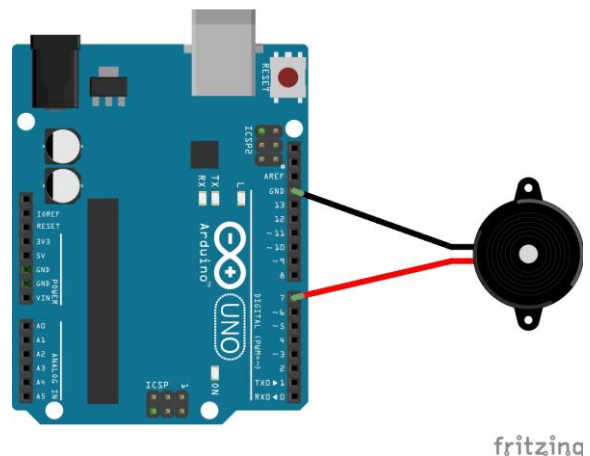


Aufgabe Lasermodul:

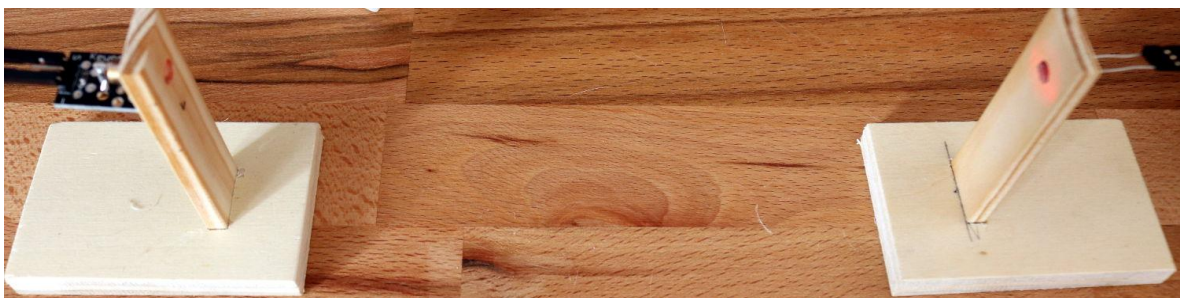
Erzeuge einen blinkenden Laserstrahl.

Active Buzzer

Ein Active Buzzer erzeugt einen einfachen Ton (mit einem passive Buzzer können unterschiedliche Töne erzeugt werden). Es ist kein Vorwiderstand notwendig.

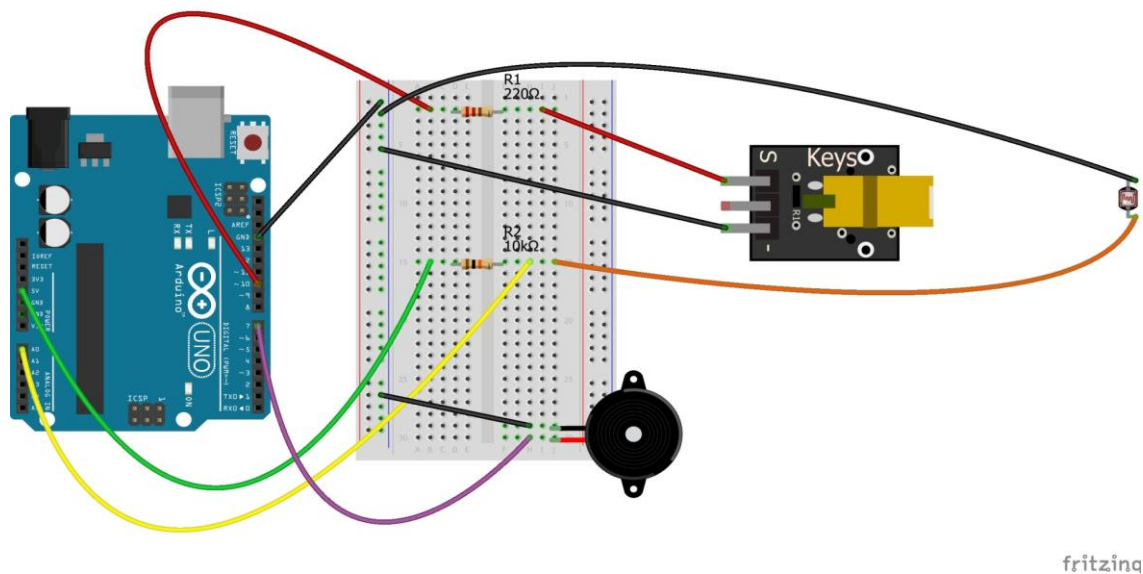


Aufgabe Lichtschranke:



Mit Hilfe eines Lasermoduls und eines Fotowiderstands soll eine Lichtschranke erstellt werden. Laser und Fotowiderstand befinden sich dabei auf gleicher Höhe. Der Fotowiderstand misst Helligkeit des Lasers, bei Abweichung soll ein Alarm (Buzzer) losgehen.

Möglicher Aufbau:



Weitere Aufgabenvorschläge:

- Zufallssteuerung von LEDs (der Befehl random(3) erzeugt zufällig eine der ganzen Zahlen 0 bis 2).
- Statt active buzzer einen passive buzzer bei der Lichtschranke verwenden und eine Melodie spielen lassen. (Anleitung für passive buzzer z. B. unter <https://funduino.de/nr-08-toene-erzeugen>)
- Einen Temperatursensor anschließen (Anleitung z. B. unter <https://funduino.de/nr-9-temperatur-messen>)
- Einen Ultraschallsensor anschließen und die Entfernung eines Gegenstands z. B über LEDs anzeigen lassen (z. B. rot leuchtet, falls kein Gegenstand in der Nähe, gelb, falls sich ein Gegenstand bis zu 10 cm nähert, grün, falls ganz nah). (Anleitung für Ultraschallsensor z. B. unter <https://funduino.de/nr-10-entfernung-messen>)
- Eine Lärmampel bauen (mit Hilfe von analogem Geräuschsensor und Servo, Anleitungen für die einzelnen Elemente z. B. unter funduino.de)

Link zu guten, schülergerechten Anleitungen:

<http://funduino.de/anleitungen>