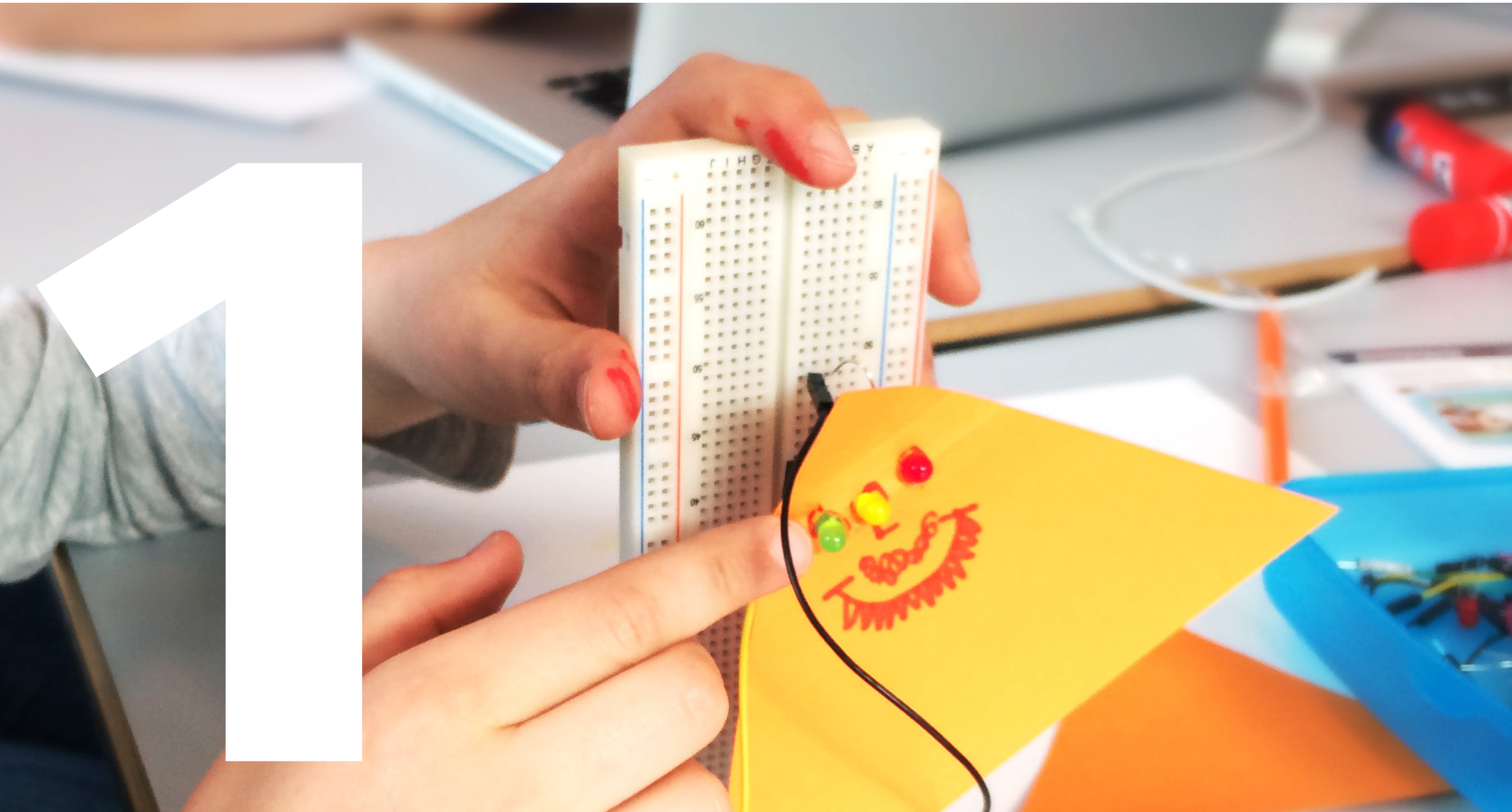
StartHardware.org

Von der Schönheit und Eleganz programmierbarer Gegenstände



1

Was könnten programmierbare Gegenstände sein?

Was bedeutet überhaupt programmieren?

Was brauchen Dinge, um programmierbar zu sein?

Foto von Jon Fingas

Foto von Kārlis Dambrāns

Foto von Phil Long

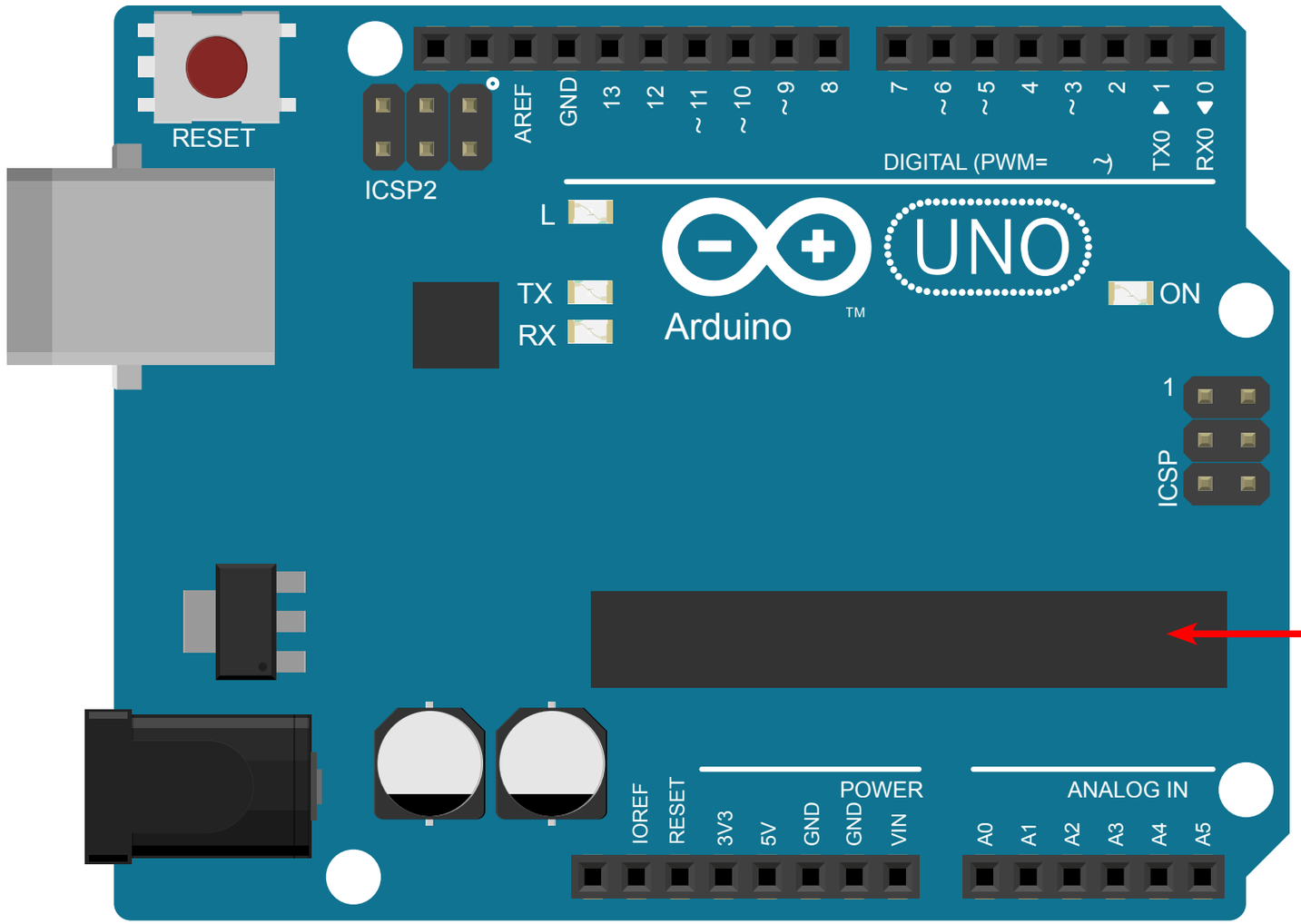
Foto von Cary Bass-Deschenes


CAUTION!
ZOMBIES!
AHEAD!!!

Wir arbeiten mit dem Arduino!

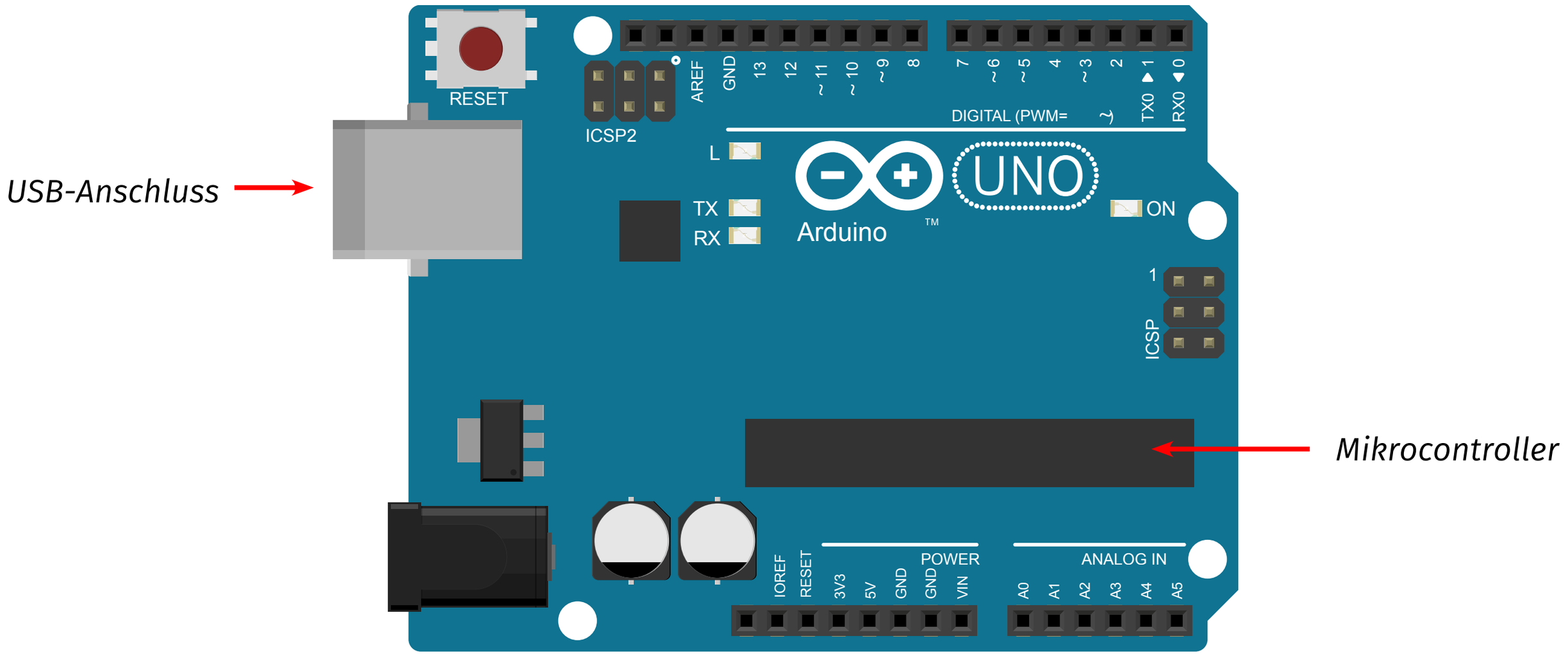
Alle Teilnehmer sollten jetzt ein Arduino-Board in die Hand nehmen und beschreiben.

USB-Anschluss →



→ *Mikrocontroller*

An das Arduino können wir alle möglichen elektronischen Dinge anschließen.

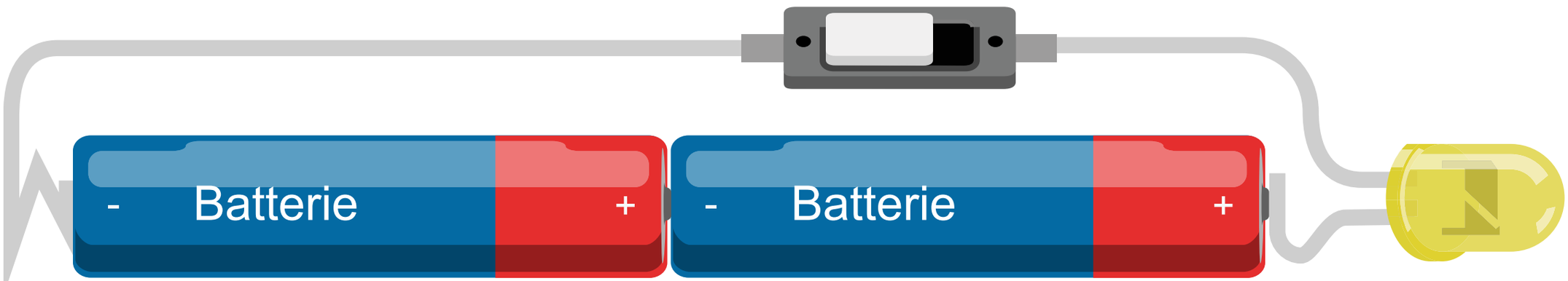


Was ist Elektronik?

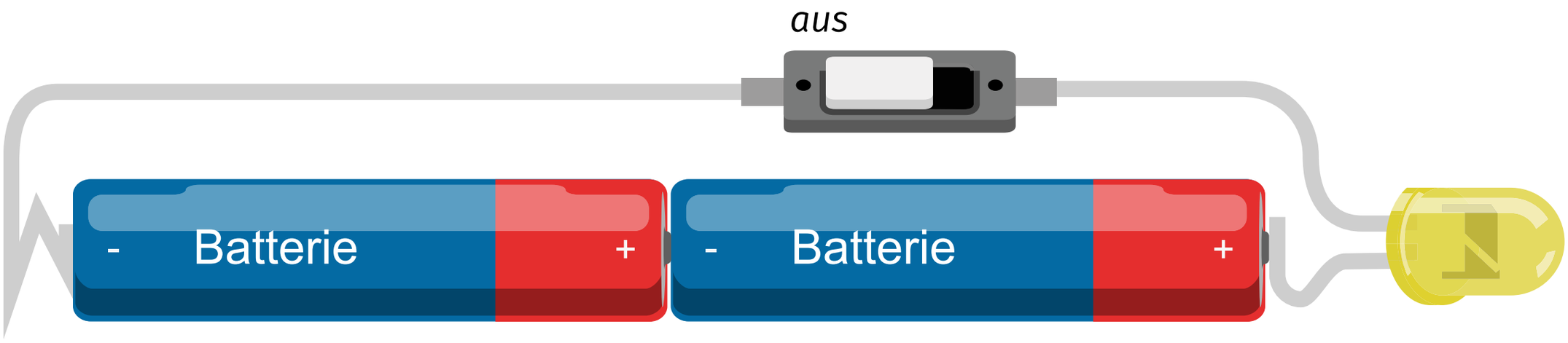
Was ist Strom?

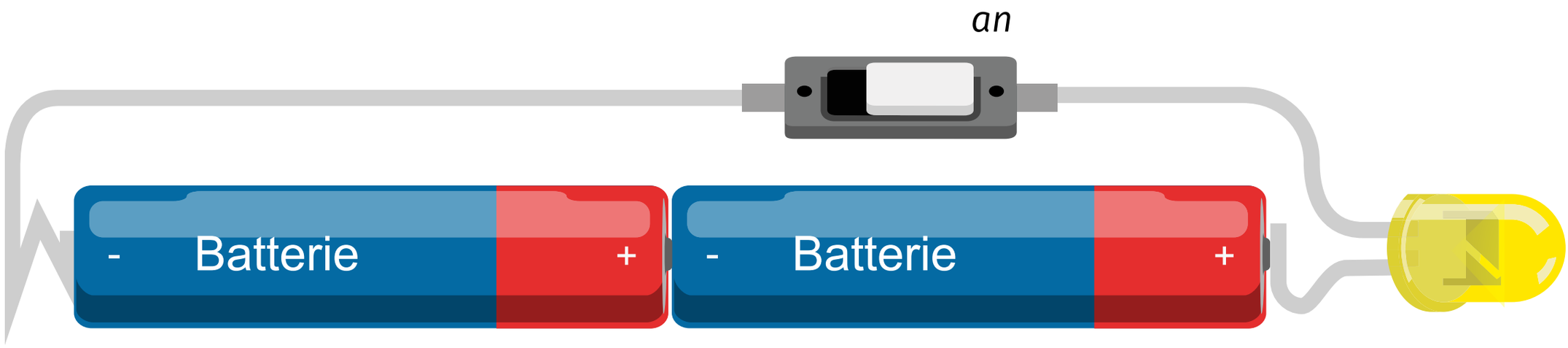
Woraus besteht eine Taschenlampe?

Woraus besteht eine Taschenlampe?

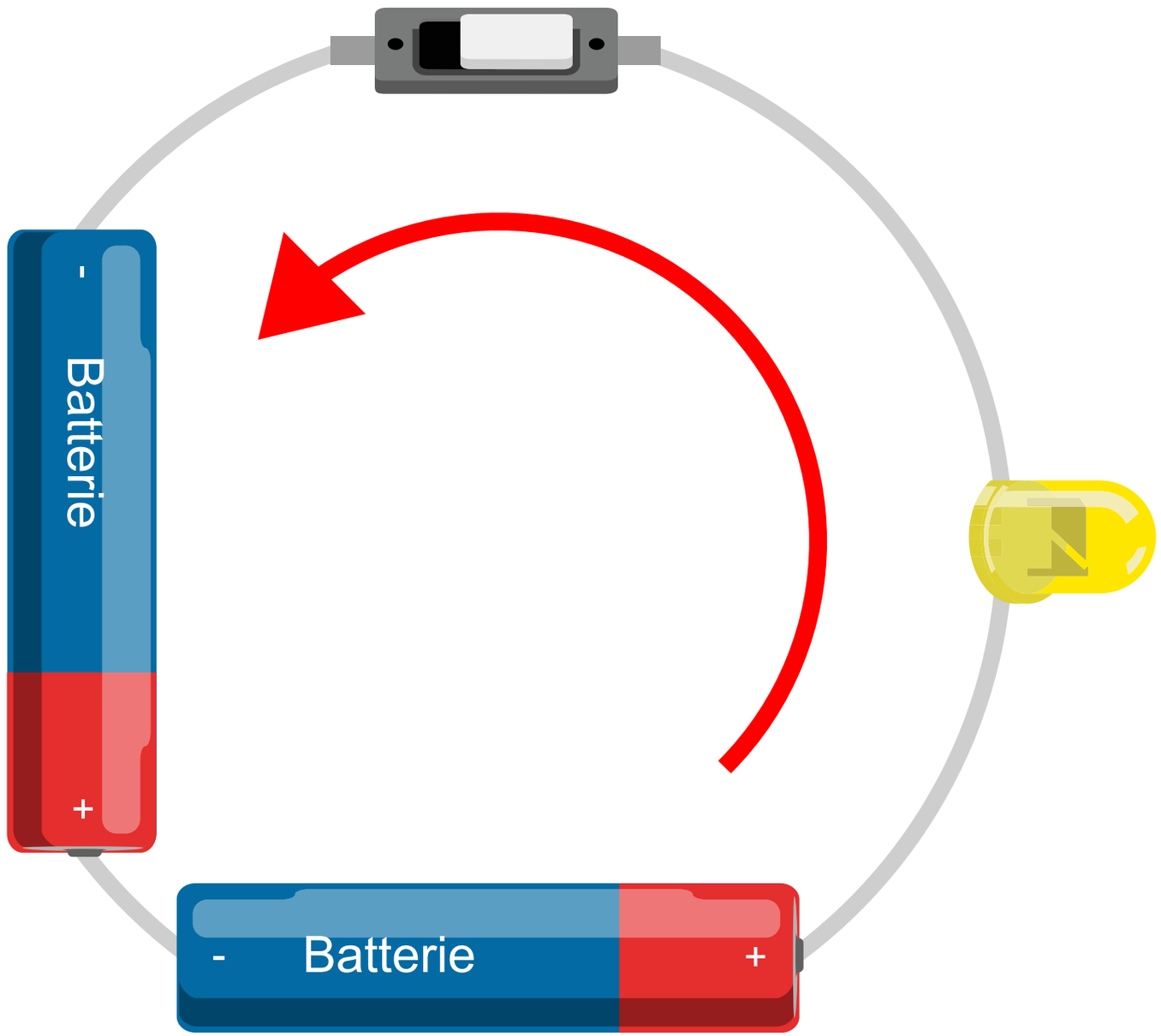


Wieso leuchtet die Lampe, wenn man sie einschaltet?





Strom fließt immer von Plus zu Minus

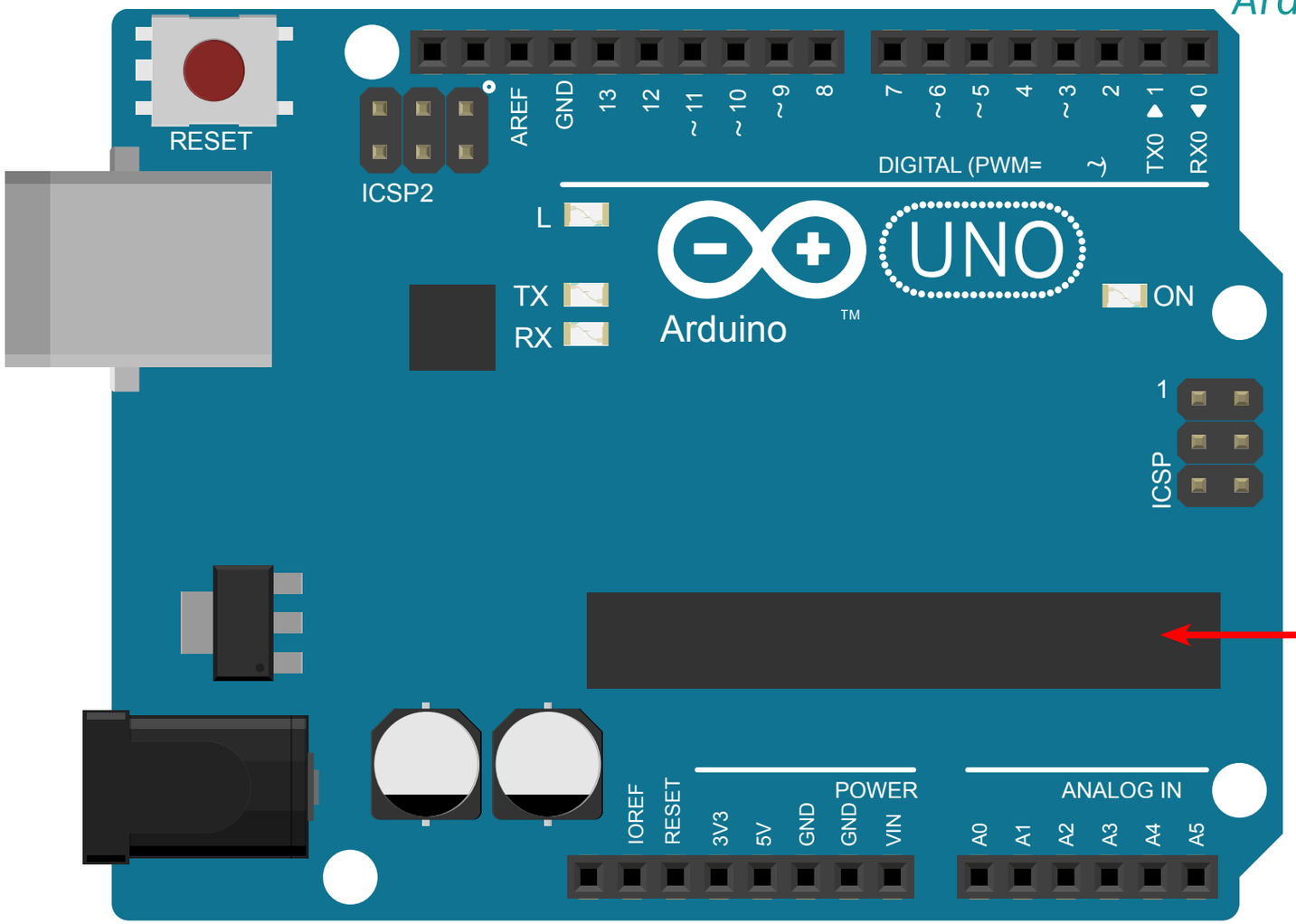


Was passiert, wenn man eine Batterie falsch herum einsetzt?

Wo ist beim Arduino Plus und Minus?

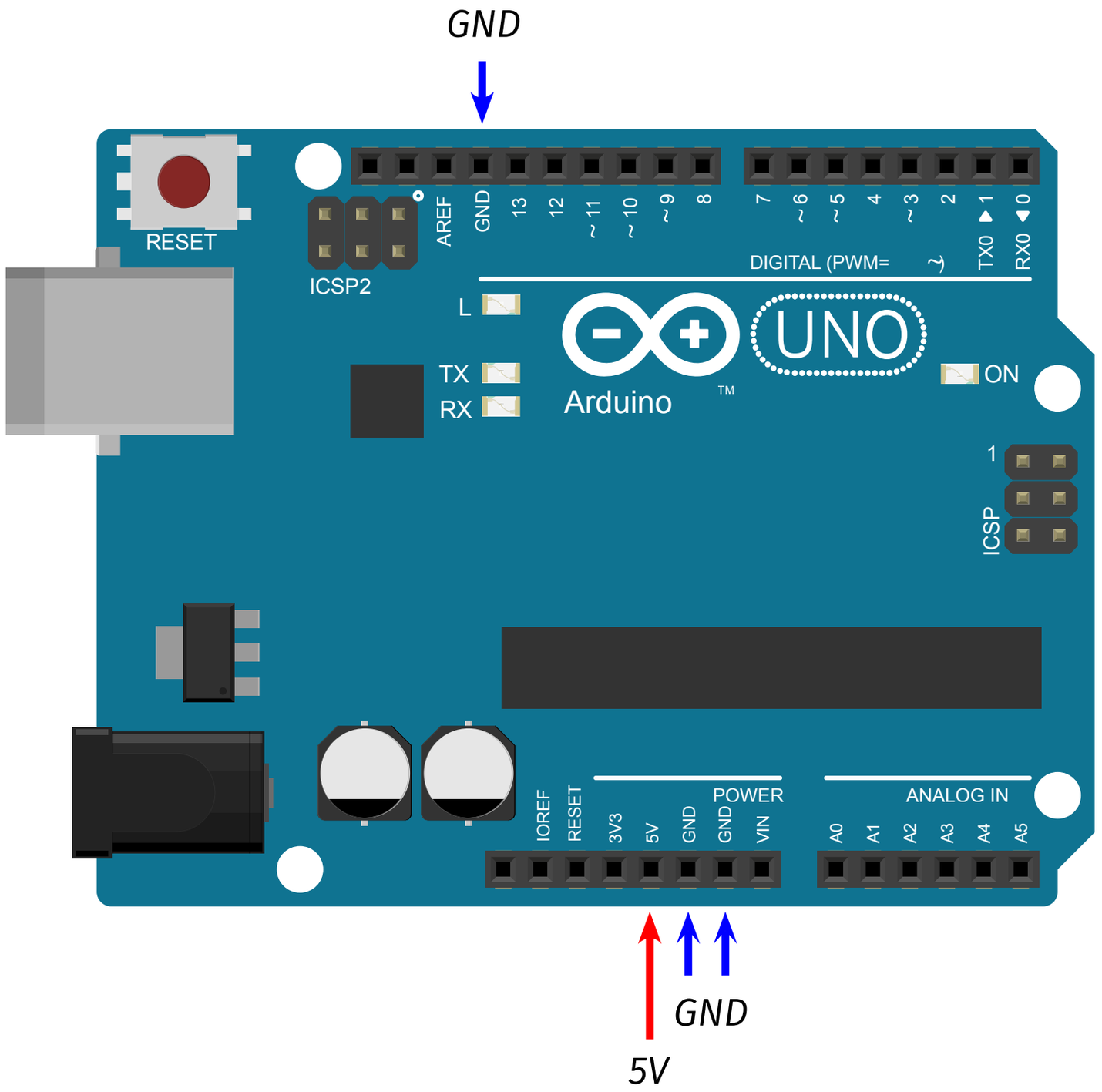
Hier kommen viele Vorschläge auf. Gern werden USB[1]- und DC[2]-Buchse, die Kondensatoren[3] oder das Arduino-Logo[4] gezeigt.

USB-Anschluss →



Mikrocontroller ←

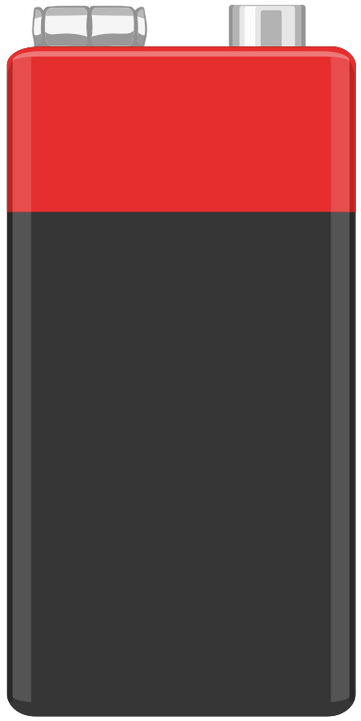
Was könnte 5V bedeuten?

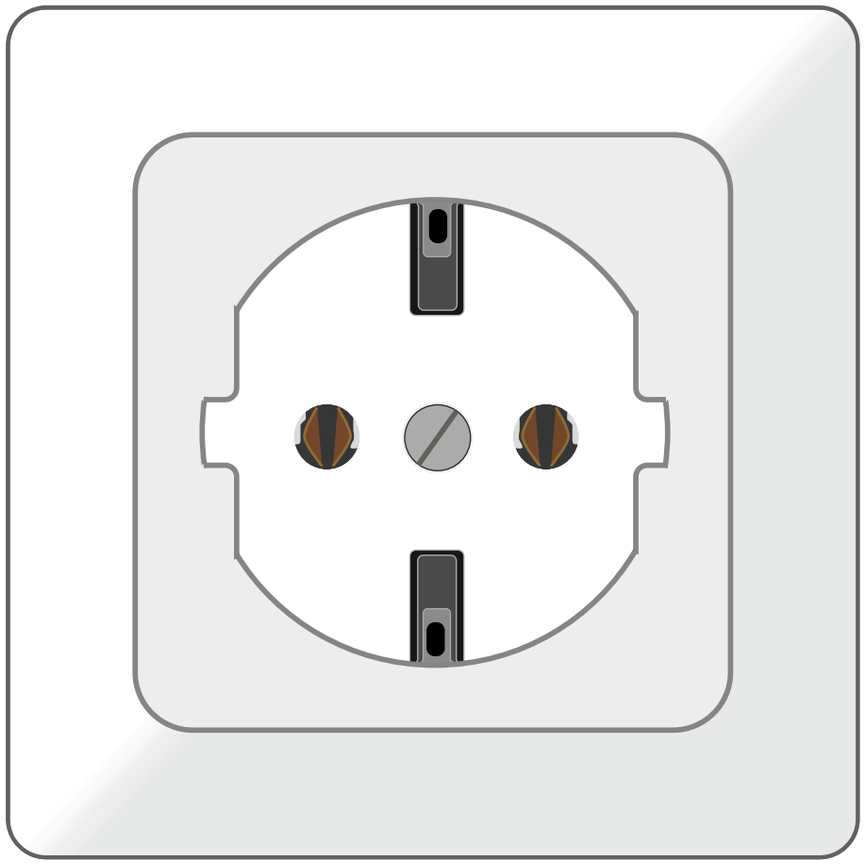


- 5V = Plus
- GND = Minus

Hier können die Teilnehmer etwas raten.

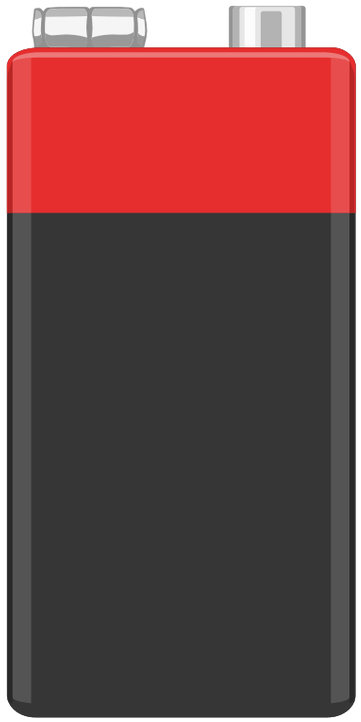
Was hat wie viel Spannung?



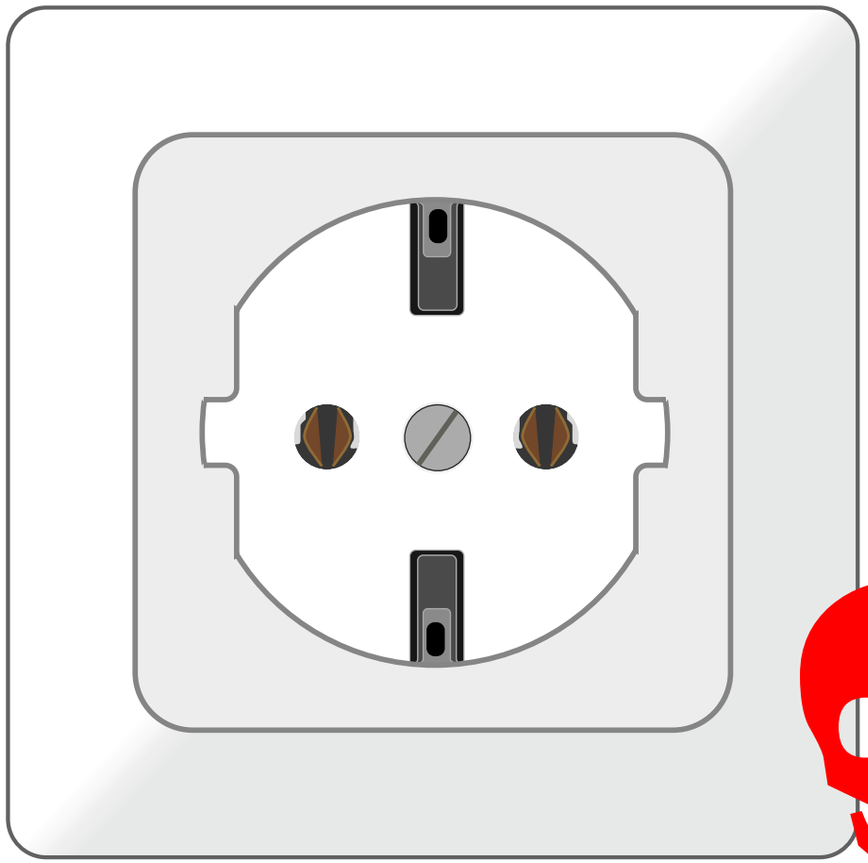


Was hat wie viel Spannung?

1,5 Volt



9 Volt



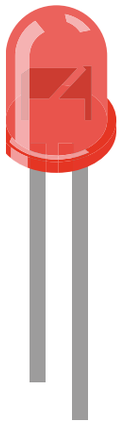
230 Volt

Wir bauen eine Taschenlampe

Bauteile

Widerstand, 220 Ohm
(rot-rot-braun-gold)

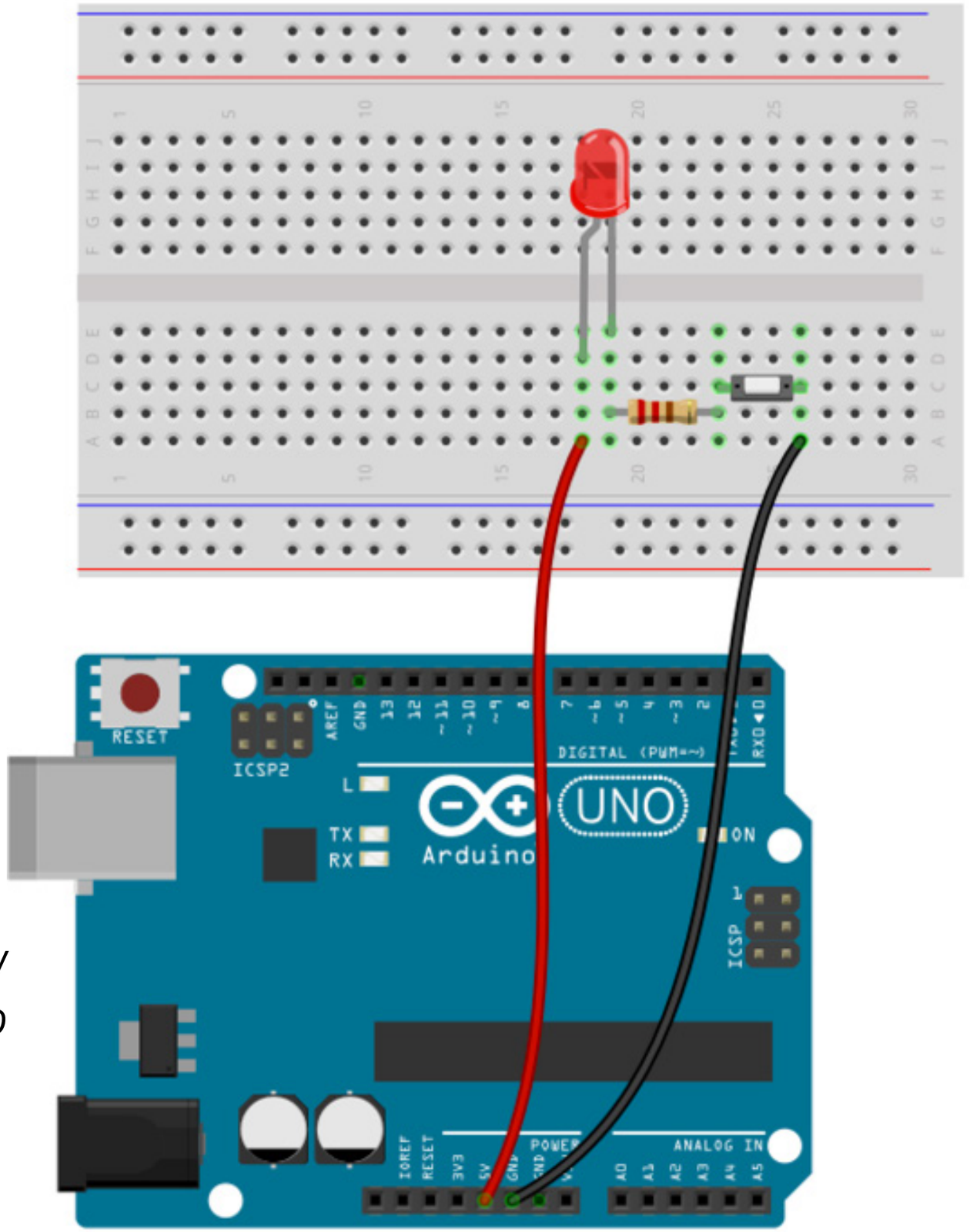
Taster



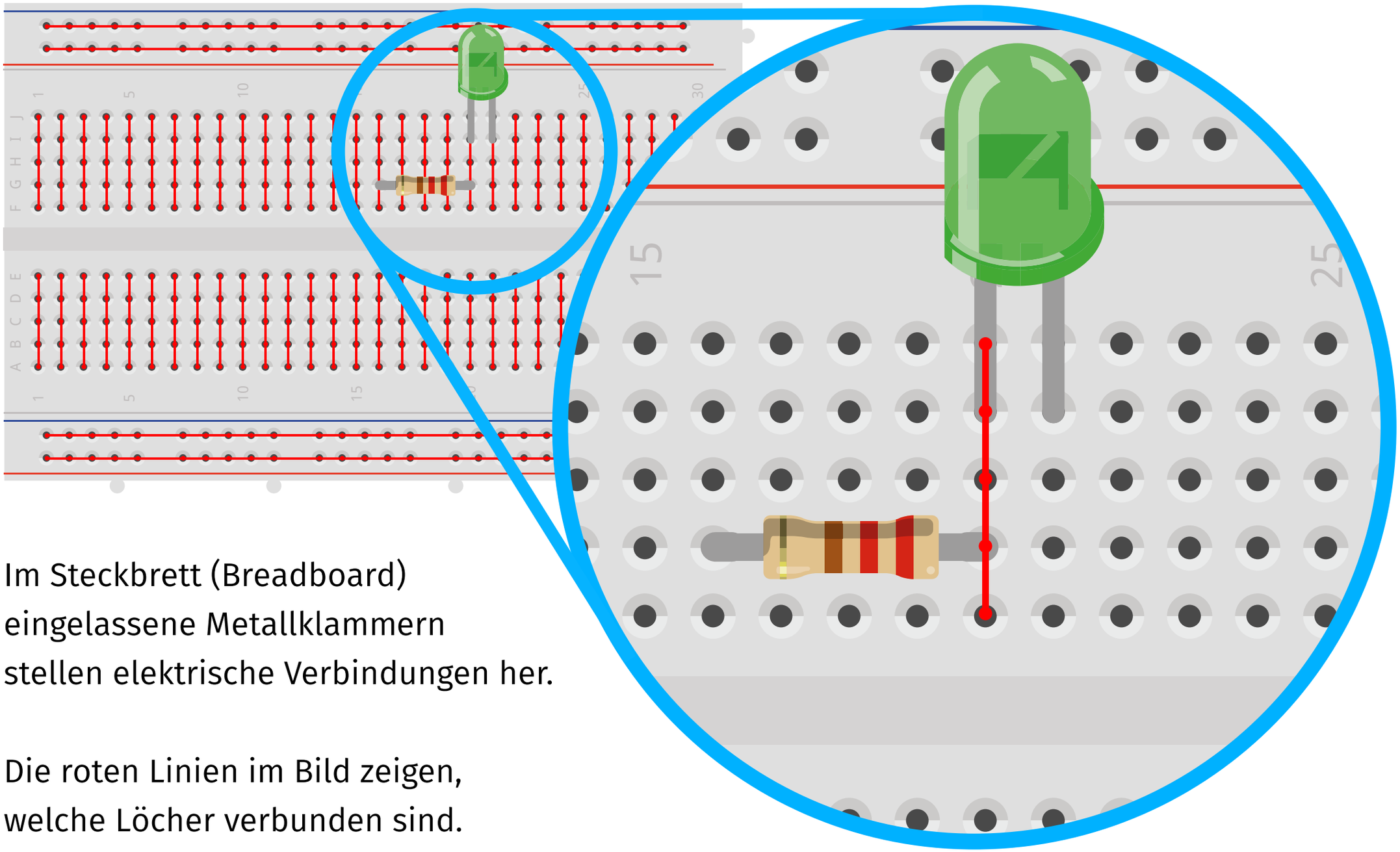
LED (Leuchtdiode)
(kurzes Beinchein
zum GND)

rotes Kabel an 5V
schwarzes Kabel an GND

Schaltung einer Taschenlampe



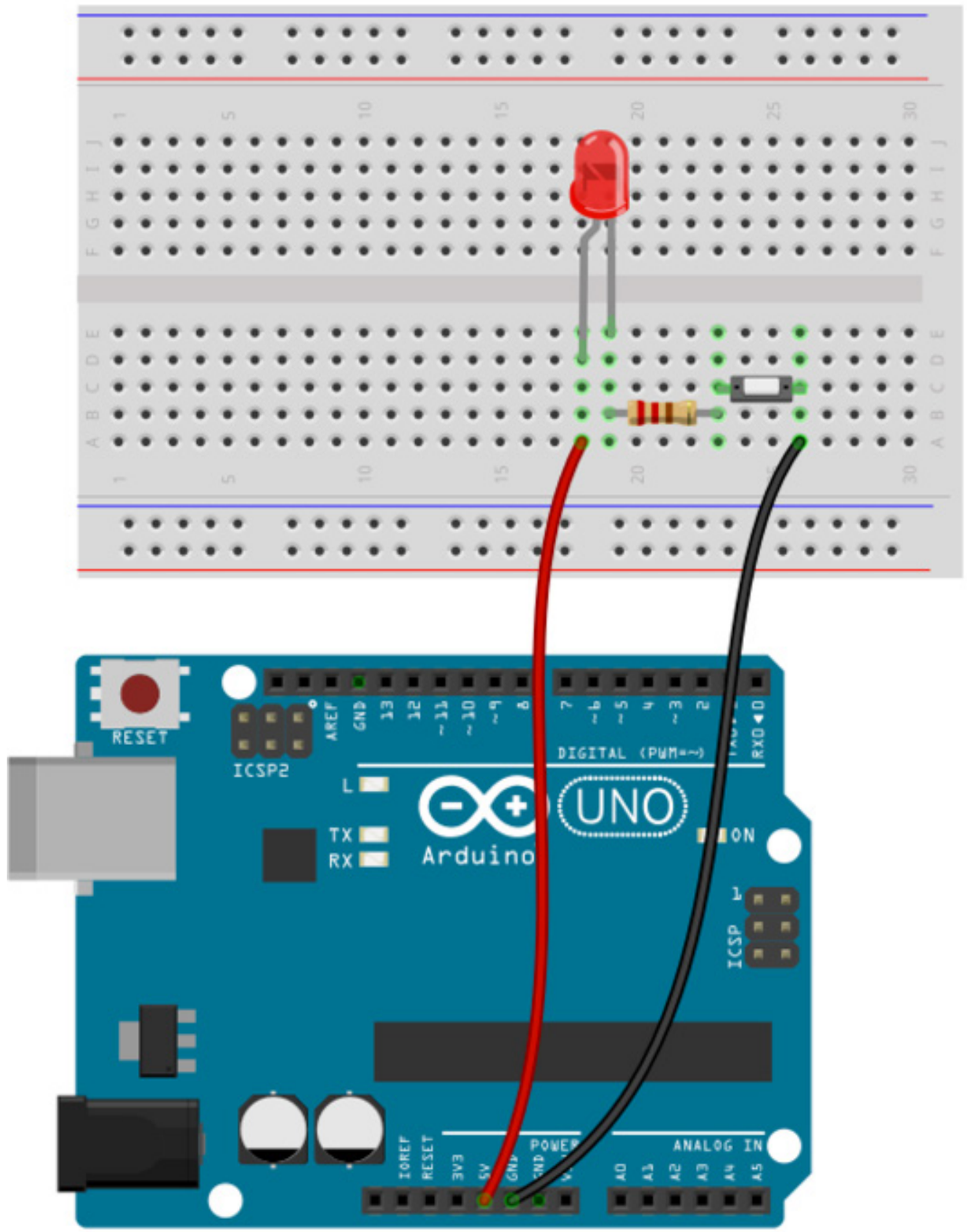
So sind die Löcher des Steckbretts verbunden



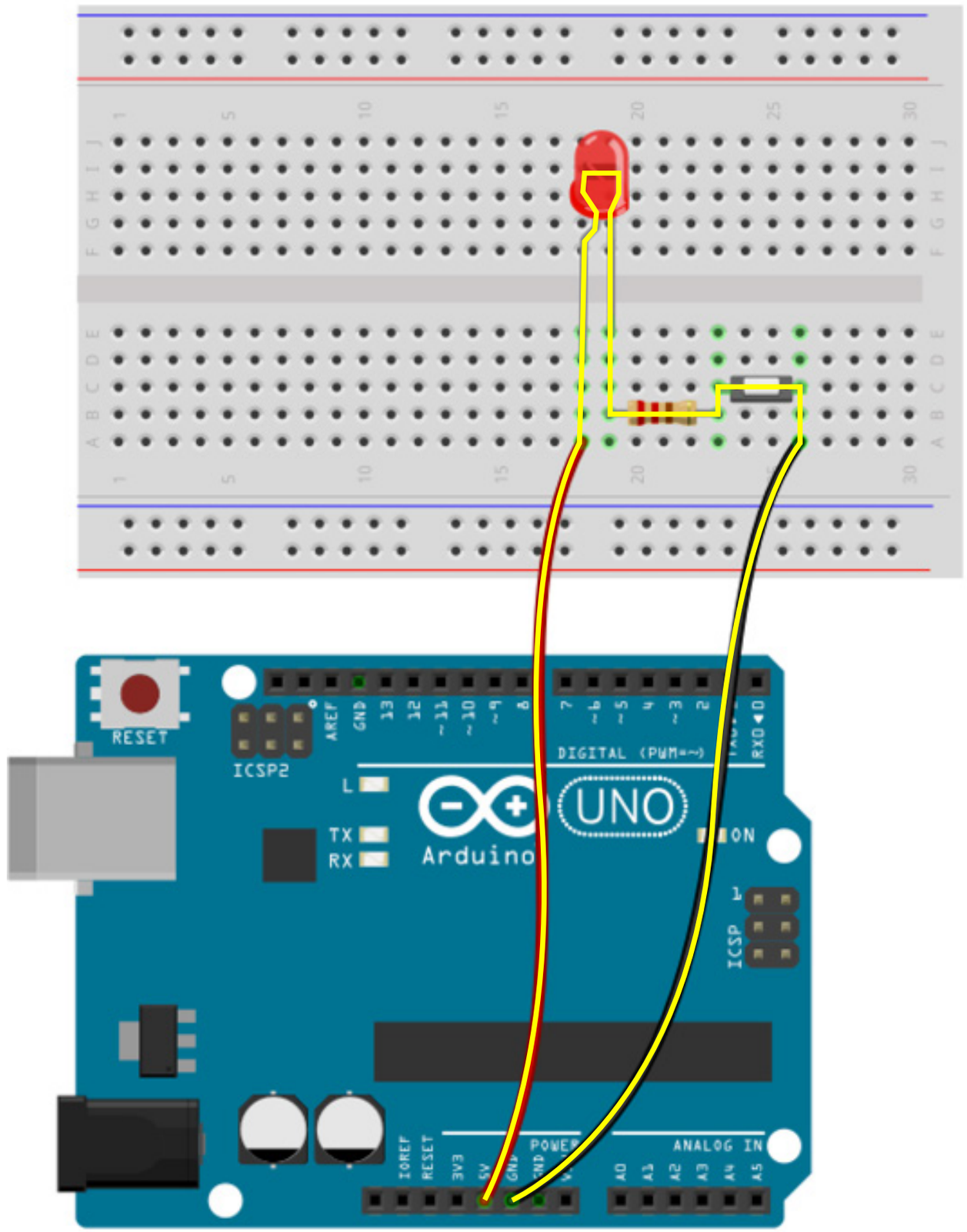
Im Steckbrett (Breadboard)
eingelassene Metallklammern
stellen elektrische Verbindungen her.

Die roten Linien im Bild zeigen,
welche Löcher verbunden sind.

Wo fließt der Strom lang?



Wo fließt der Strom lang?



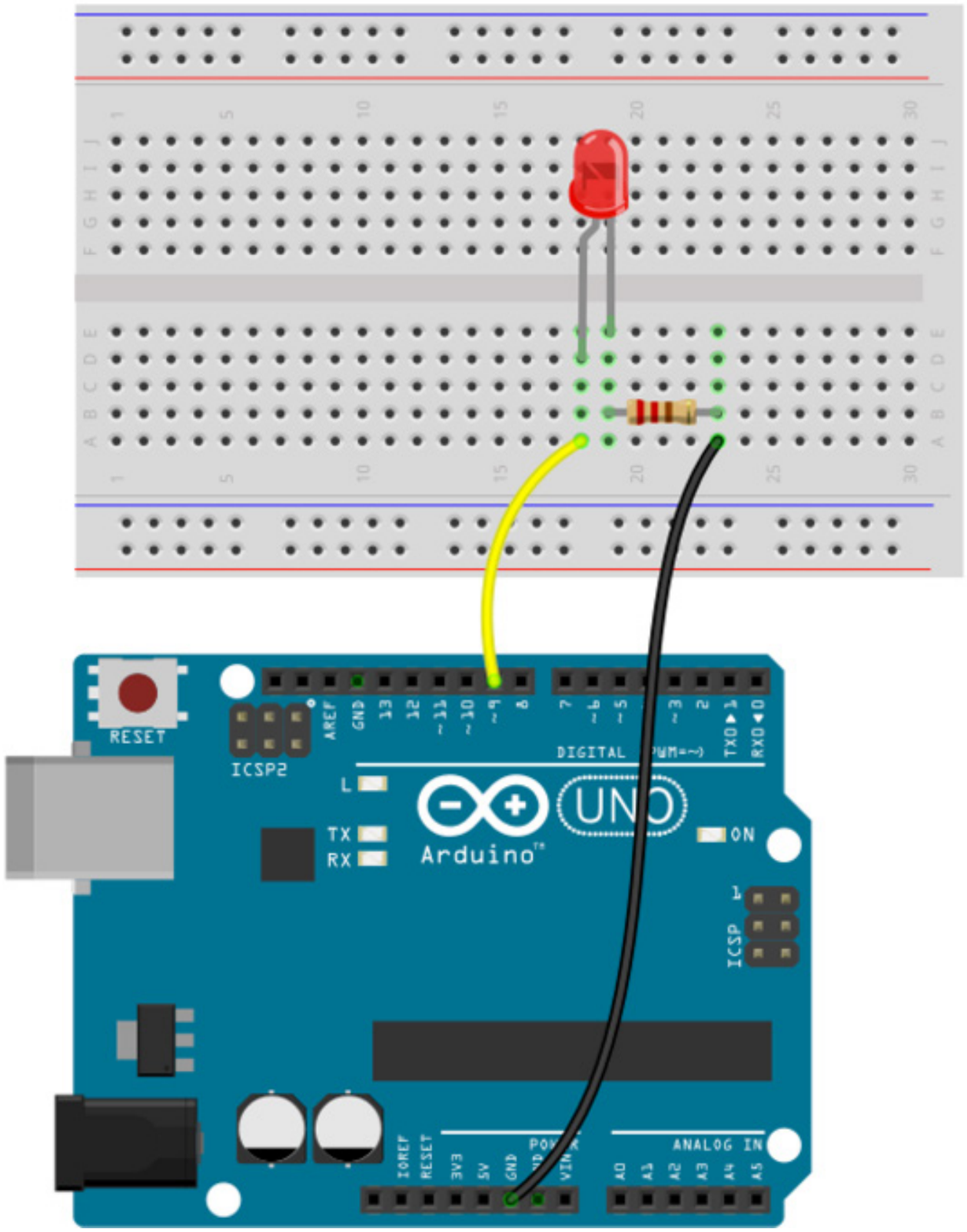
Jetzt wollen wir eine LED blinken lassen.

Baue die Schaltung nach

In welche Phasen lässt sich das Blinken unterteilen?

1. *LED an*
2. *warten*
3. *LED aus*
4. *warten*
5. *alles Wiederholen*

Schaltung



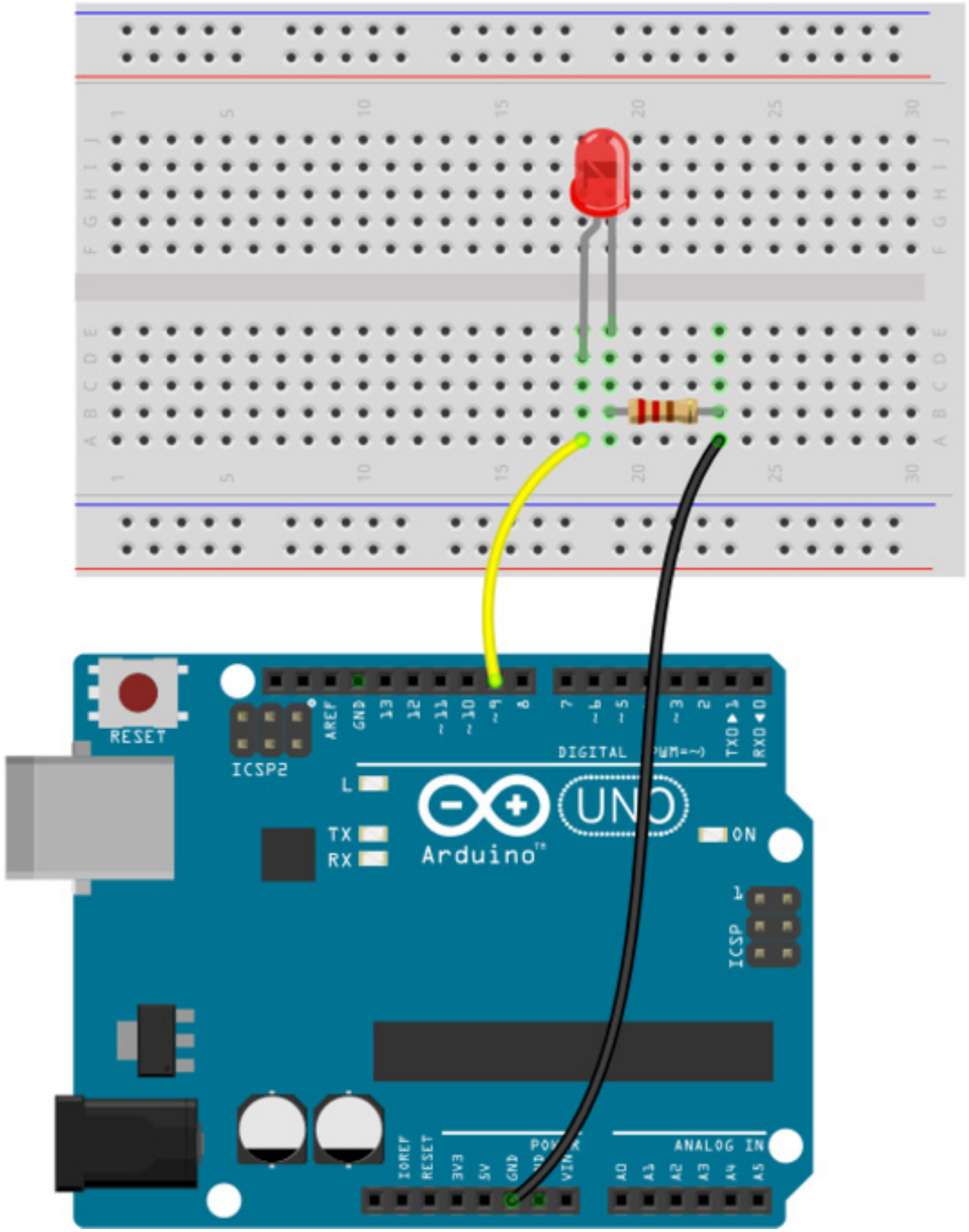
Code

Sketch

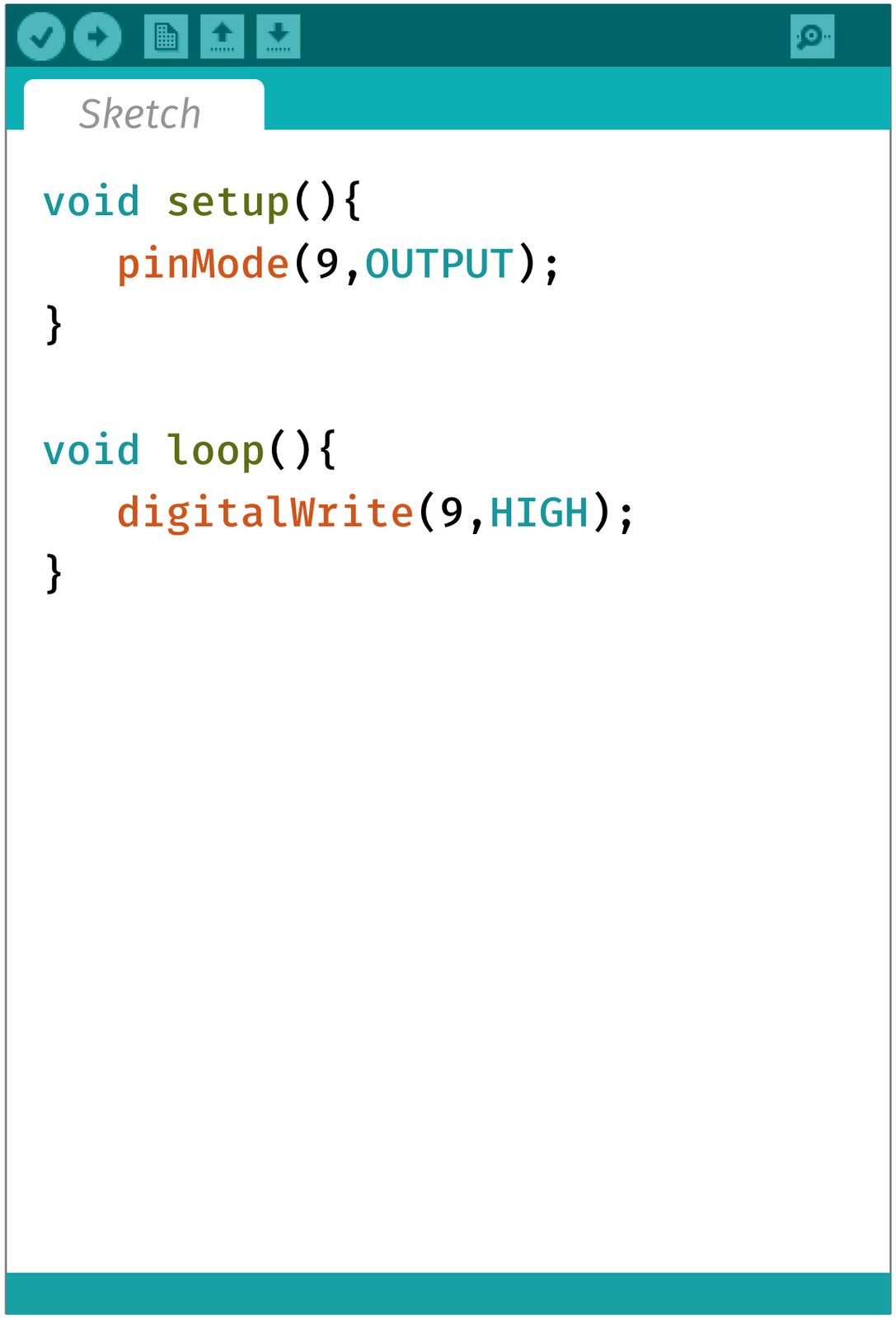
```
void setup(){
   pinMode(9,OUTPUT);
}

void loop(){
   digitalWrite(9,HIGH);
}
```

(Die Arduino-Software muss von www.arduino.cc auf den Computern installiert sein.)

Schaltung



Code



Achtung! *Das Programm muss auf das Arduino-Board übertragen werden. Dazu muss der richtige Port im Werkzeuge-Menü ausgewählt sein:*

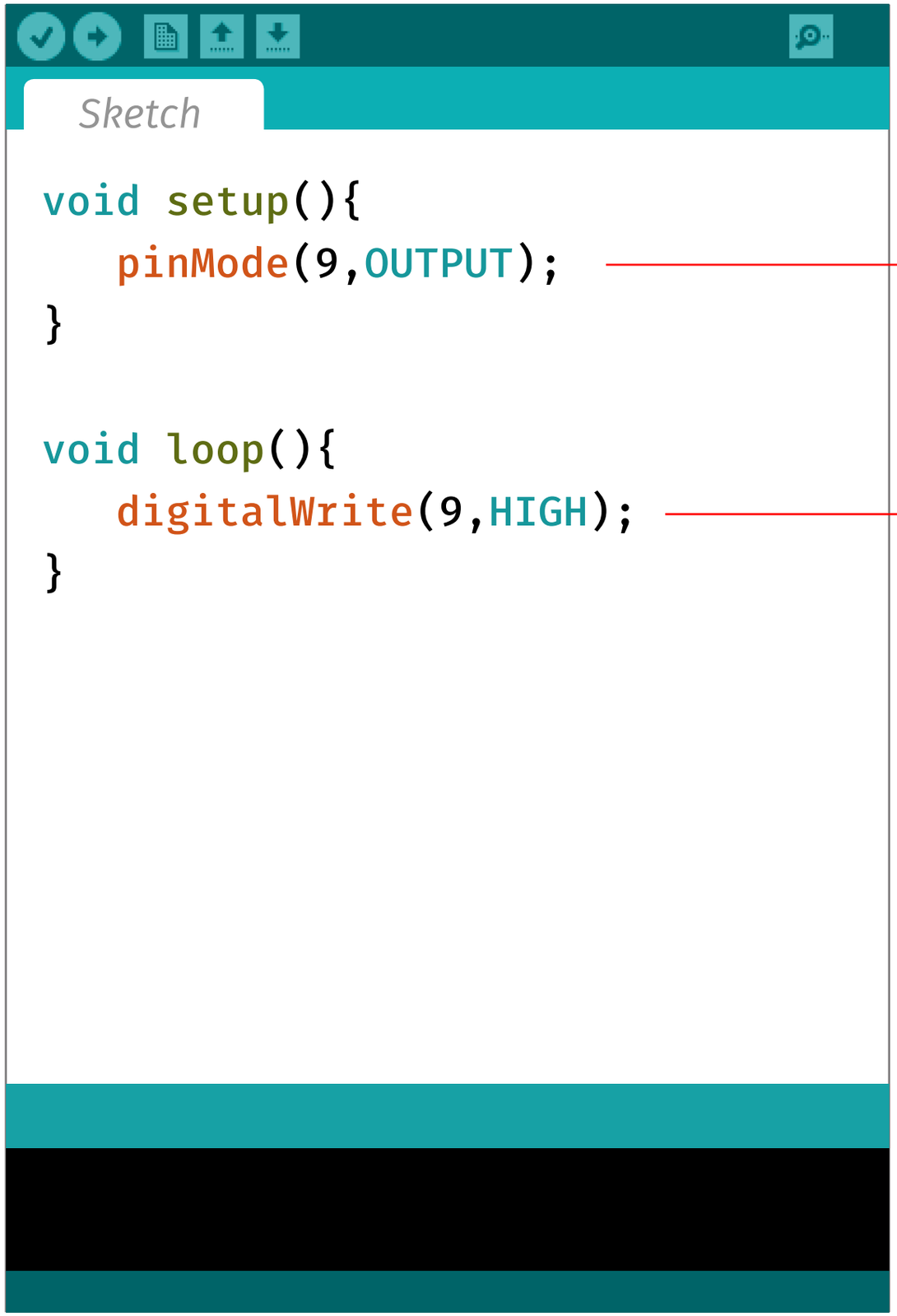
Werkzeuge > Port > xxxxx (Arduino UNO)

Übertrage das Programm dann mit

Datei > Upload

oder klicke in der Icon-Leiste das Symbol mit dem Pfeil nach rechts

Code



Dieser Befehl sagt dem Arduino, dass wir einen Pin (Pin 9) als Output verwenden wollen. Damit können wir ihn nun ein- und ausschalten.

Dieser Befehl schaltet den Pin 9 ein (HIGH).

Wie könnte man ihn wieder ausschalten?

Code

Sketch

```
void setup(){
   pinMode(9,OUTPUT);
}

void loop(){
   digitalWrite(9,LOW);
}
```

Dieser Befehl schaltet den Pin 9 aus (LOW).

Jetzt lass die LED blinken!

Code

Sketch

```
void setup(){
   pinMode(9,OUTPUT);
}

void loop(){
   digitalWrite(9,HIGH);
   digitalWrite(9,LOW);
}
```

LED einschalten

LED ausschalten

Blinkt die LED jetzt?

Code

Sketch

```
void setup(){
   pinMode(9,OUTPUT);
}

void loop(){
   digitalWrite(9,HIGH);
   delay(1000);
   digitalWrite(9,LOW);
   delay(1000);
}
```

Das Arduino-Board ist schnell. So schnell, dass wir das Ein- und Ausschalten der LED nicht sehen können. Wir brauchen also eine Bremse im Programmcode.

LED einschalten

stoppt das Programm für 1000 Millisekunden

LED ausschalten

stoppt das Programm für 1000 Millisekunden

Verstanden? Gut, dann lass die LED schneller blinken!

Code

Sketch

```
void setup(){
   pinMode(9,OUTPUT);
}

void loop(){
   digitalWrite(9,HIGH);
   delay(200);
   digitalWrite(9,LOW);
   delay(200);
}
```

Je kleiner der Wert im delay ist, desto schneller blinkt die LED.

Code

Sketch

```
...
   // kurzes Blinken
   digitalWrite(9,HIGH);
   delay(100);
   digitalWrite(9,LOW);
   delay(100);

   // langes Blinken
   digitalWrite(9,HIGH);
   delay(400);
   digitalWrite(9,LOW);
   delay(400);
...
```

Nun eine kleine Übung. Lass die LED deinen Namen im Morse-Code blinken.

Ein Punkt bedeutet kurz blinken, ein Strich bedeutet lang blinken.

A •-	H ••••	O ---	V •••-
B -•••	I ••	P •--•	W •--
C -•-•	J •---	Q --•-	X -••-
D -••	K -•-	R •-•	Y -•--
E •	L •-••	S •••	Z --••
F ••-•	M --	T -	
G --•	N -•	U ••-	

Mache nach jedem Buchstaben eine längere und nach dem Namen eine sehr lange Pause.

StartHardware.org

Von der Schönheit und Eleganz programmierbarer Gegenstände

2

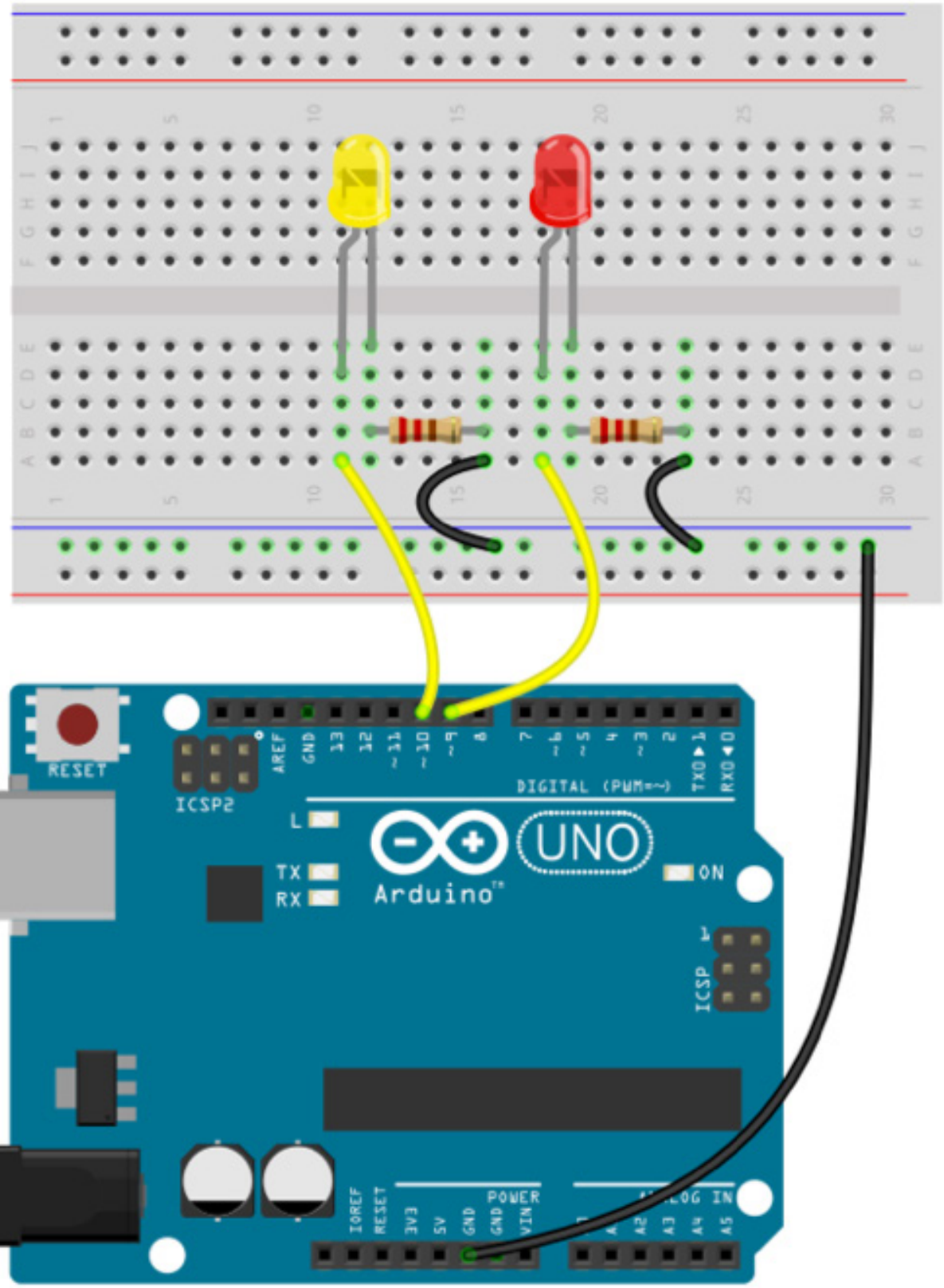
Code

Beginne mit dem Blink-Code und ändere ihn so ab, dass beide LEDs abwechselnd blinken.

Sketch

```
void setup(){
   pinMode(9,OUTPUT);
}

void loop(){
   digitalWrite(9,HIGH);
   delay(200);
   digitalWrite(9,LOW);
   delay(200);
}
```

Schaltung mit zwei LEDs

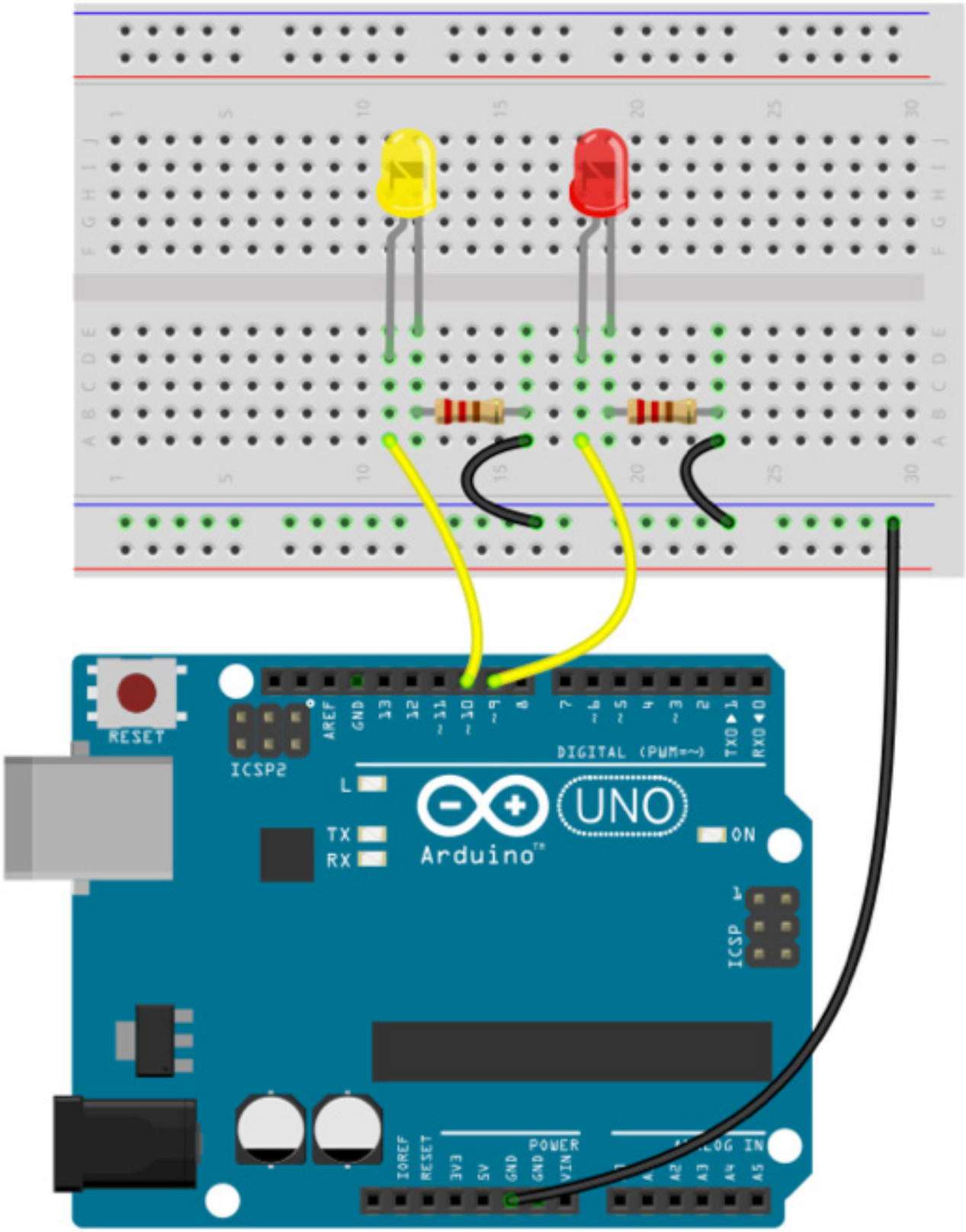


Code

Sketch

```
void setup(){
   pinMode(9,OUTPUT);
   pinMode(10,OUTPUT);
}

void loop(){
   digitalWrite(9,HIGH);
   digitalWrite(10,LOW);
   delay(200);
   digitalWrite(9,LOW);
   digitalWrite(10,HIGH);
   delay(200);
}
```

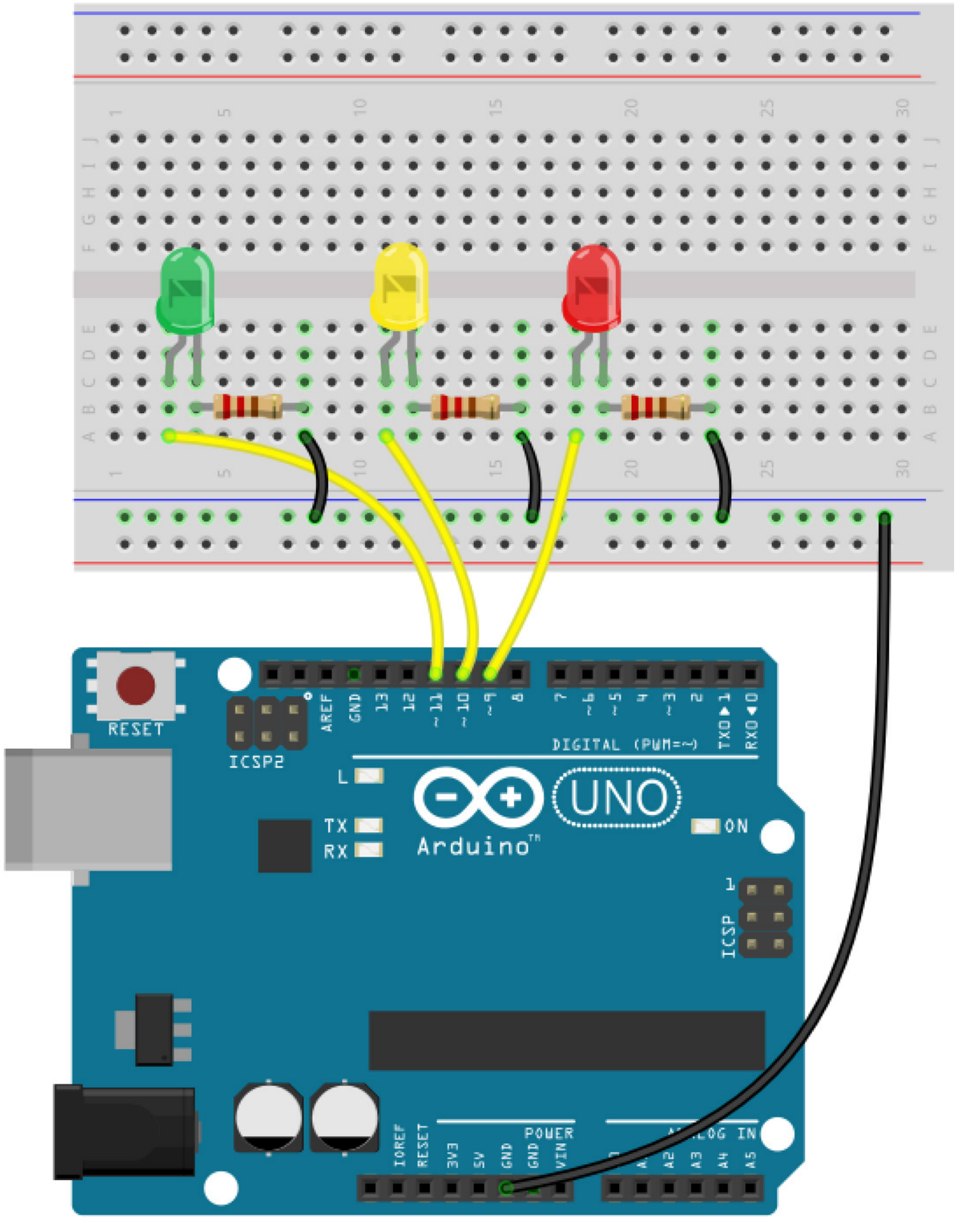
Schaltung mit zwei LEDs



Programmiere nun eine Ampel.

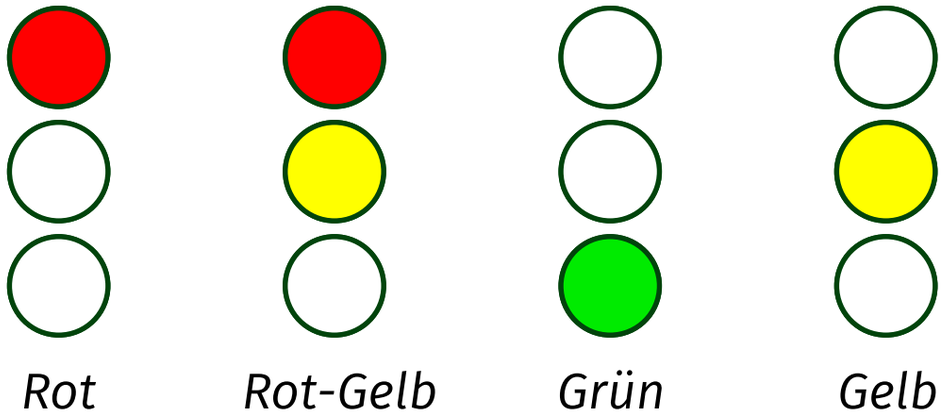
Welche Ampelphasen gibt es?

Schaltung mit drei LEDs



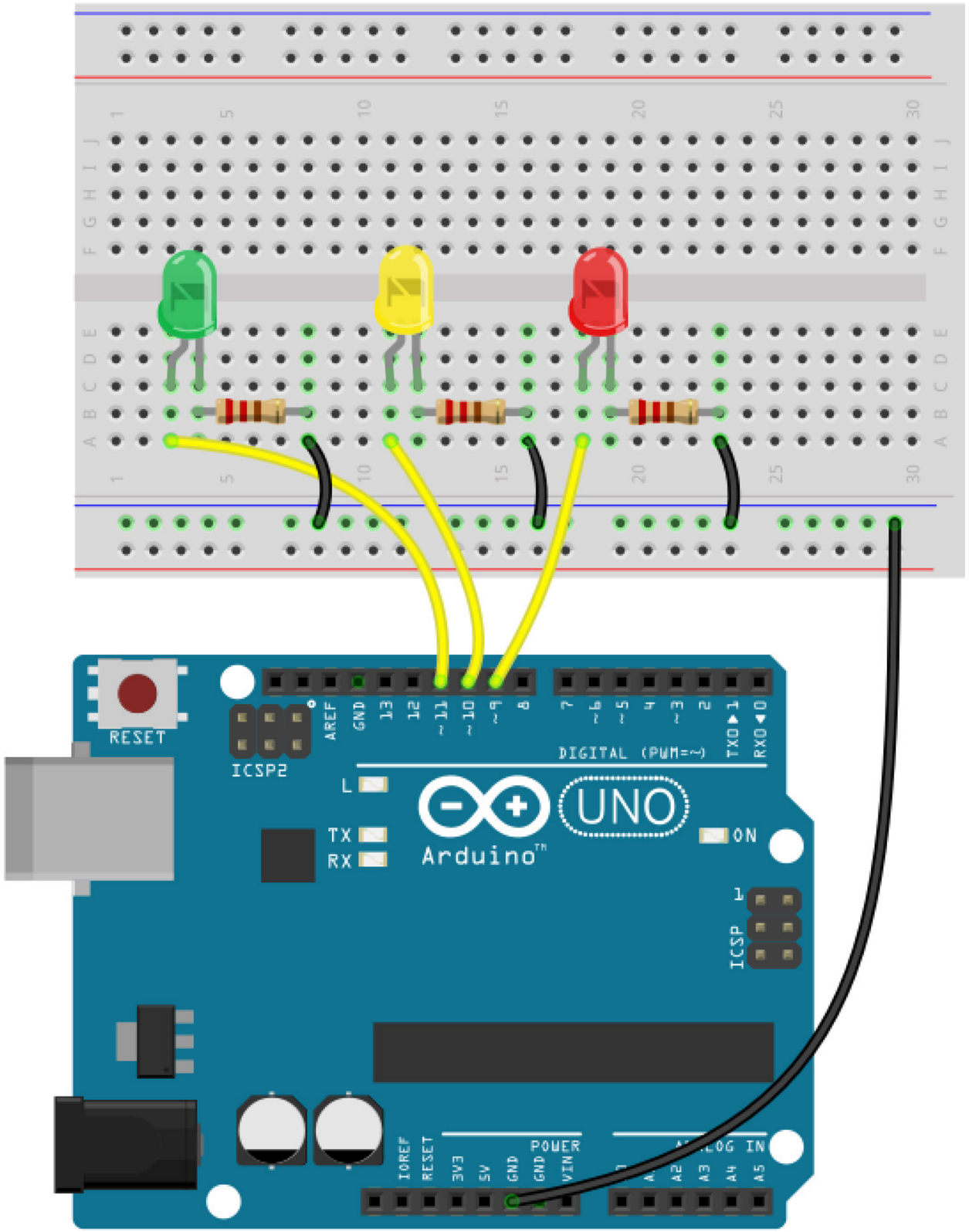
Programmiere nun eine Ampel.

Welche Ampelphasen gibt es?



Rot *Rot-Gelb* *Grün* *Gelb*

Schaltung mit drei LEDs

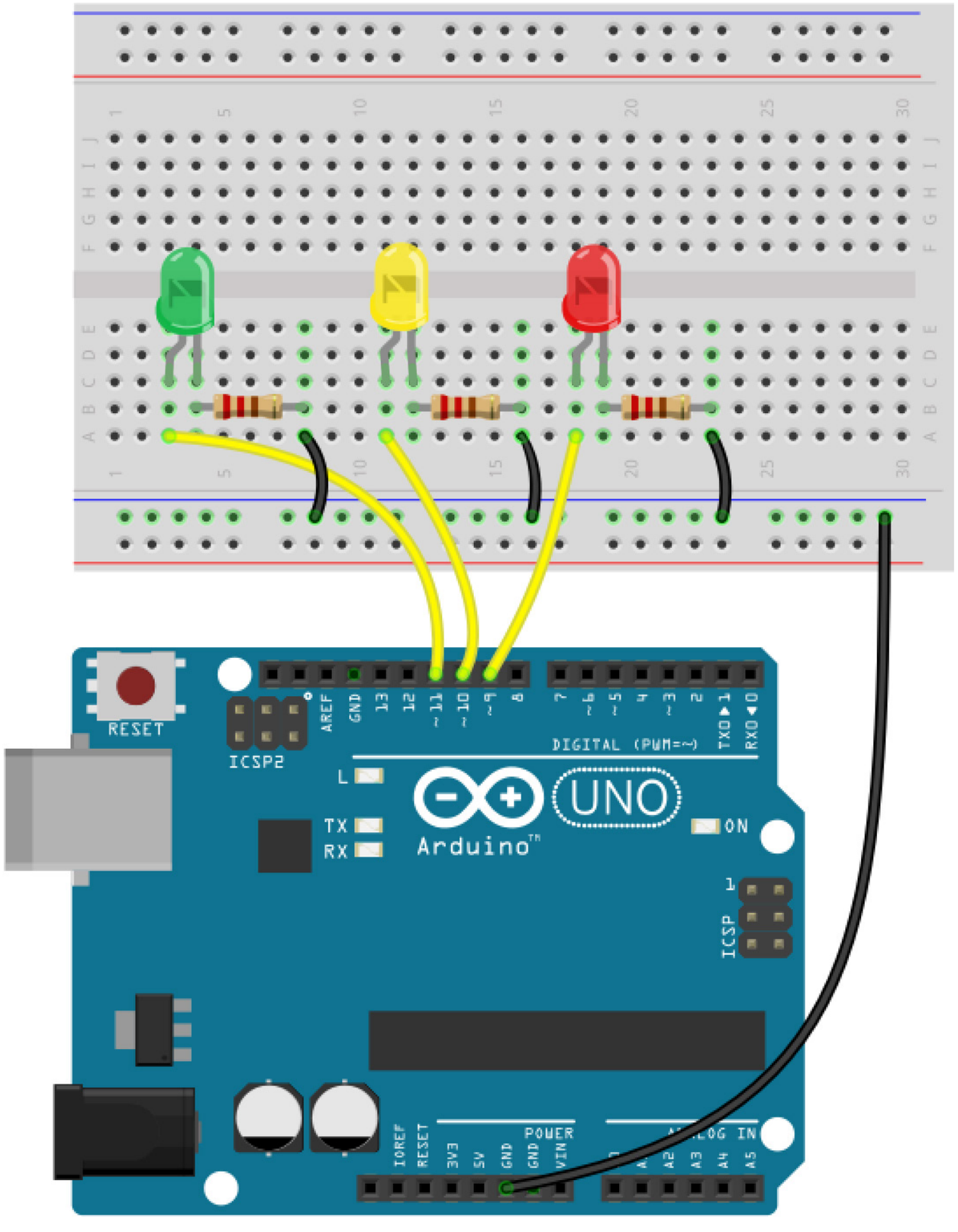


Code einer Ampel

Sketch

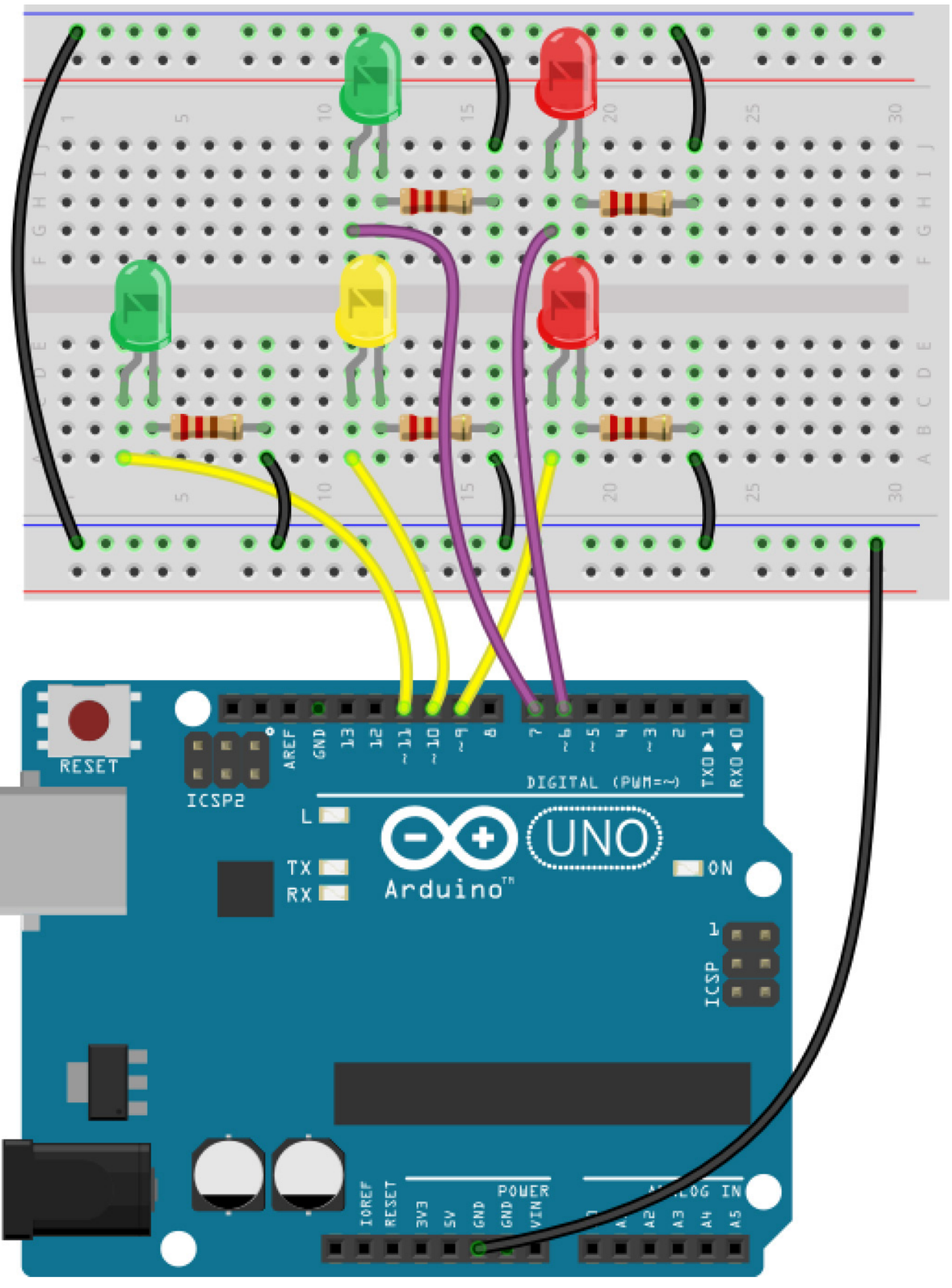
```
void setup(){
   pinMode(9,OUTPUT);
   pinMode(10,OUTPUT);
   pinMode(11,OUTPUT);
}

void loop(){
   digitalWrite(9,HIGH);  // rot an
   digitalWrite(10,LOW);  // gelb aus
   delay(4000);
   digitalWrite(10,HIGH); // gelb aus
   delay(500);
   digitalWrite(9,LOW);   // rot aus
   digitalWrite(10,LOW);  // gelb aus
   digitalWrite(11,HIGH); // grün an
   delay(4000);
   digitalWrite(9,HIGH);  // rot an
   digitalWrite(11,LOW); // grün aus
   delay(500);
}
```

Schaltung mit drei LEDs

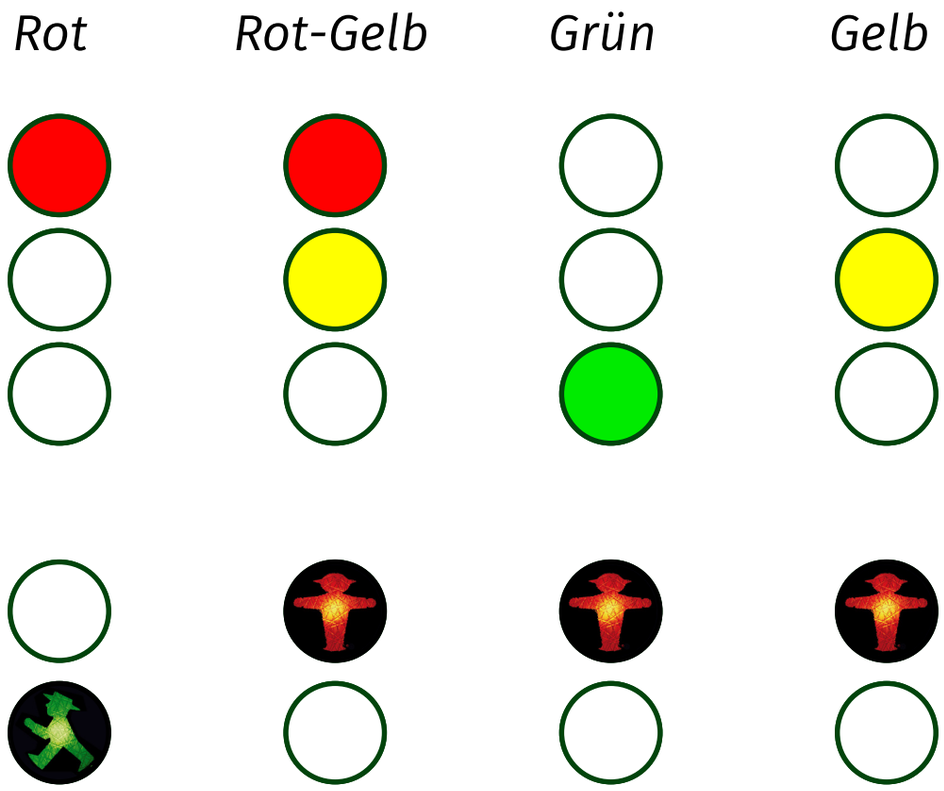


Nun haben wir eine Autofahrer- und eine Fußgängerampel. Welche Phasen gibt es jetzt?

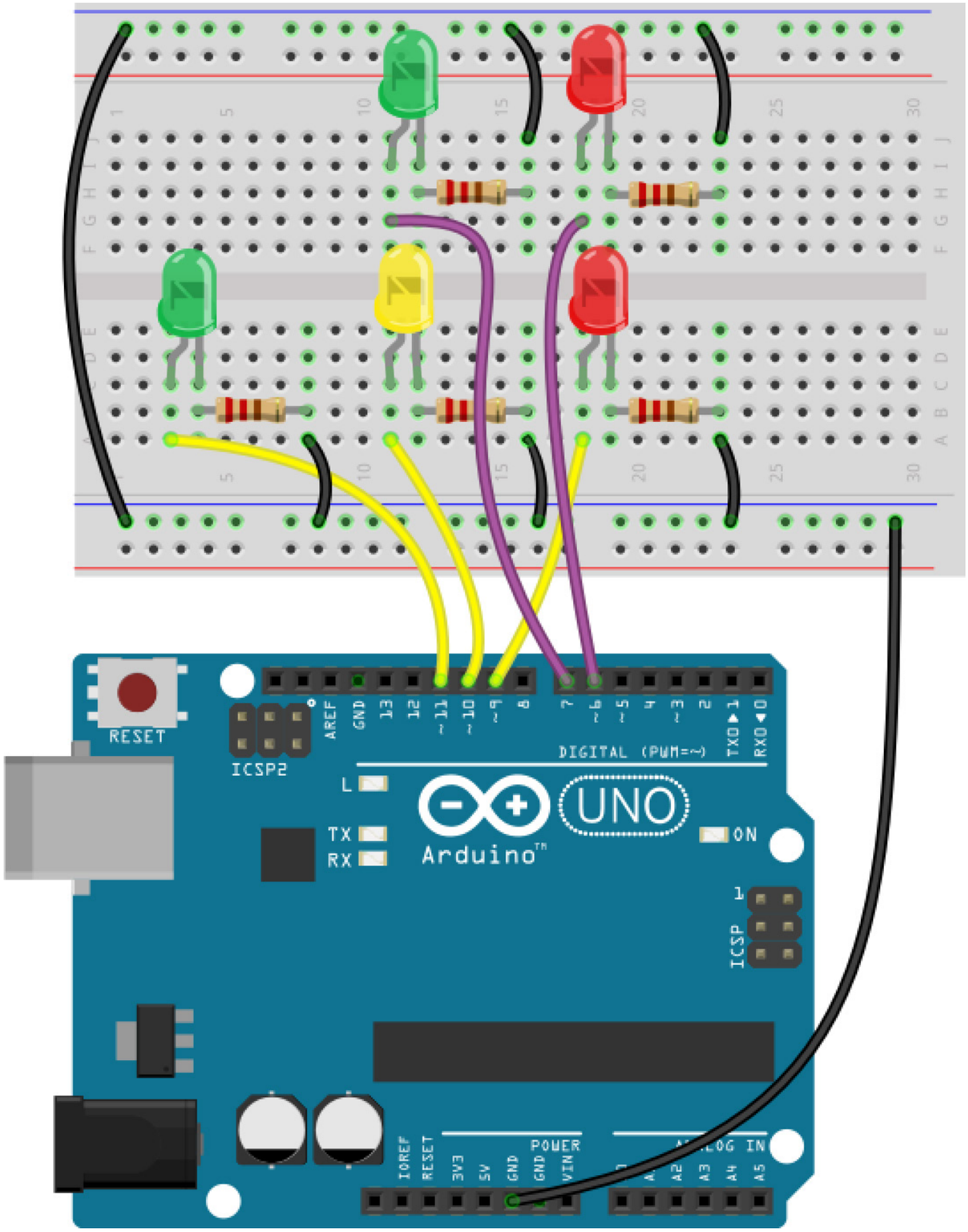
Schaltung mit fünf LEDs



Nun haben wir eine Autofahrer- und eine Fußgängerampel. Welche Phasen gibt es jetzt?



Schaltung mit fünf LEDs

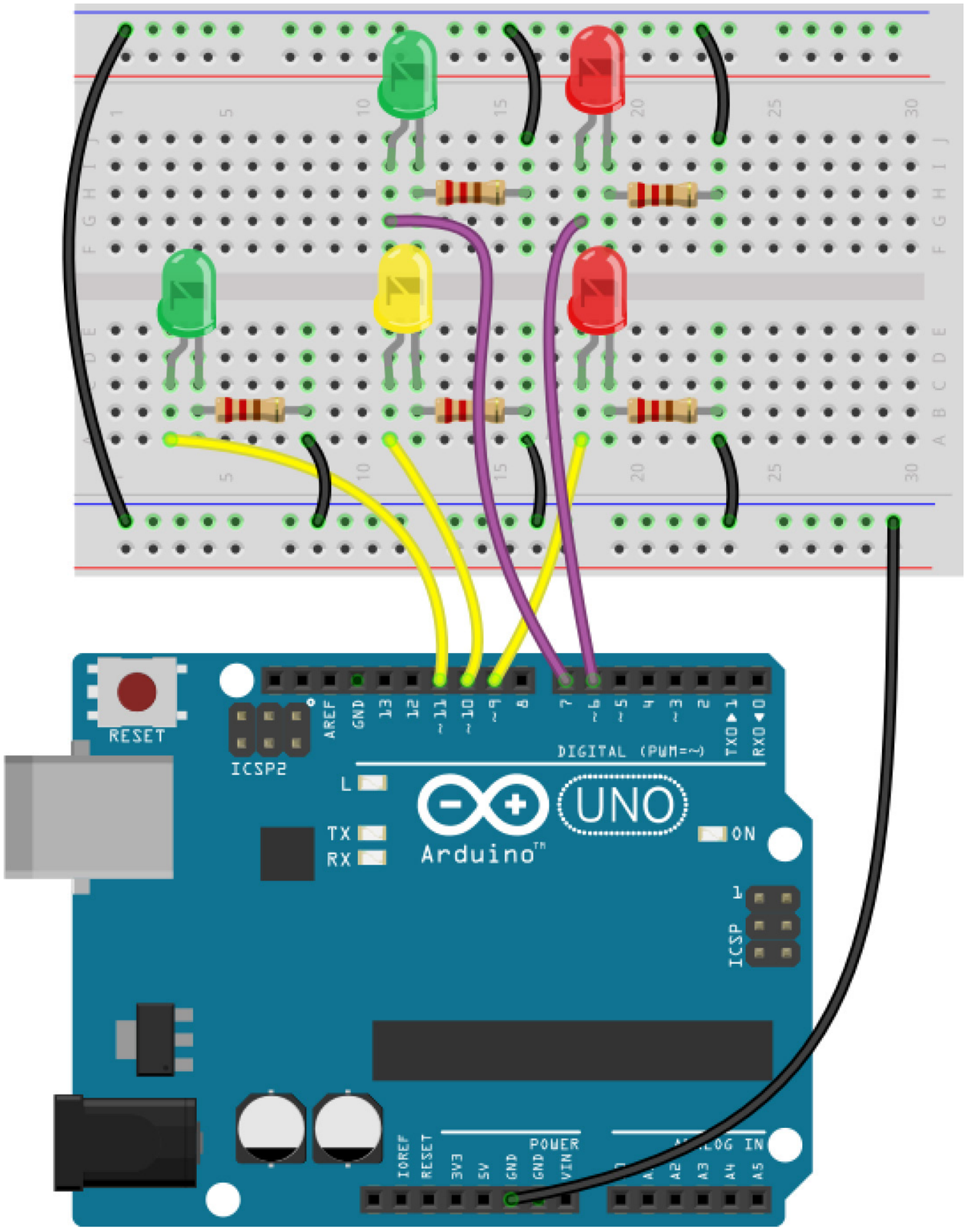


Code einer Ampel

Sketch

```
void setup(){
   pinMode(6,OUTPUT);  // rot FG
   pinMode(7,OUTPUT);  // grün FG
   pinMode(9,OUTPUT);  // rot Auto
   pinMode(10,OUTPUT); // gelb Auto
   pinMode(11,OUTPUT); // grün Auto
}

void loop(){
   digitalWrite(6,LOW);   // rot FG aus
   digitalWrite(7,HIGH);  // grün FG an
   digitalWrite(9,HIGH);  // rot an
   digitalWrite(10,LOW);  // gelb aus
   delay(4000);
   digitalWrite(6,HIGH);  // rot FG an
   digitalWrite(7,LOW);   // grün FG aus
   digitalWrite(10,HIGH); // gelb aus
   delay(500);
   digitalWrite(9,LOW);   // rot aus
   digitalWrite(10,LOW);  // gelb aus
   digitalWrite(11,HIGH); // grün an
   delay(4000);
   digitalWrite(9,HIGH);  // rot an
   digitalWrite(11,LOW); // grün aus
   delay(500);
}
```

Schaltung mit fünf LEDs

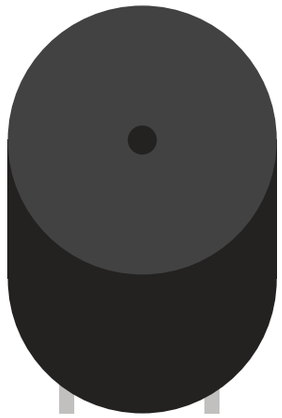


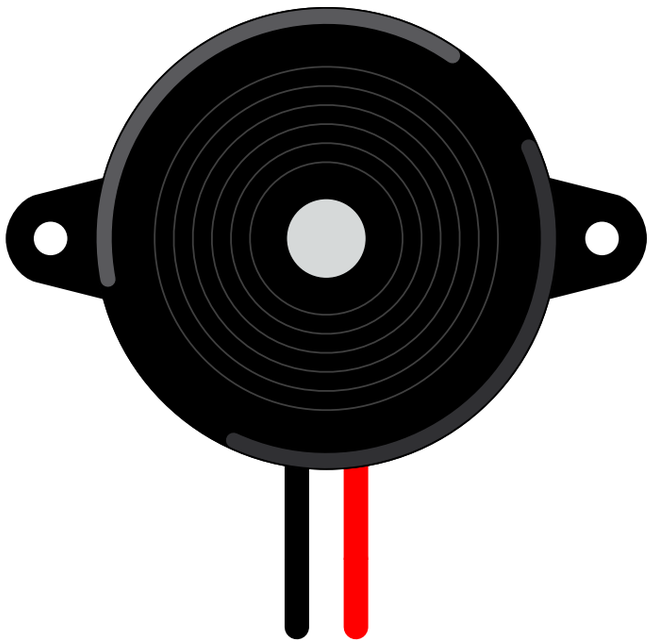
StartHardware.org

Von der Schönheit und Eleganz programmierbarer Gegenstände

3

Musik!





Ein Piezo-Lautsprecher wandelt Strom in Klang

Es gibt sie in unterschiedlichen Bauarten.

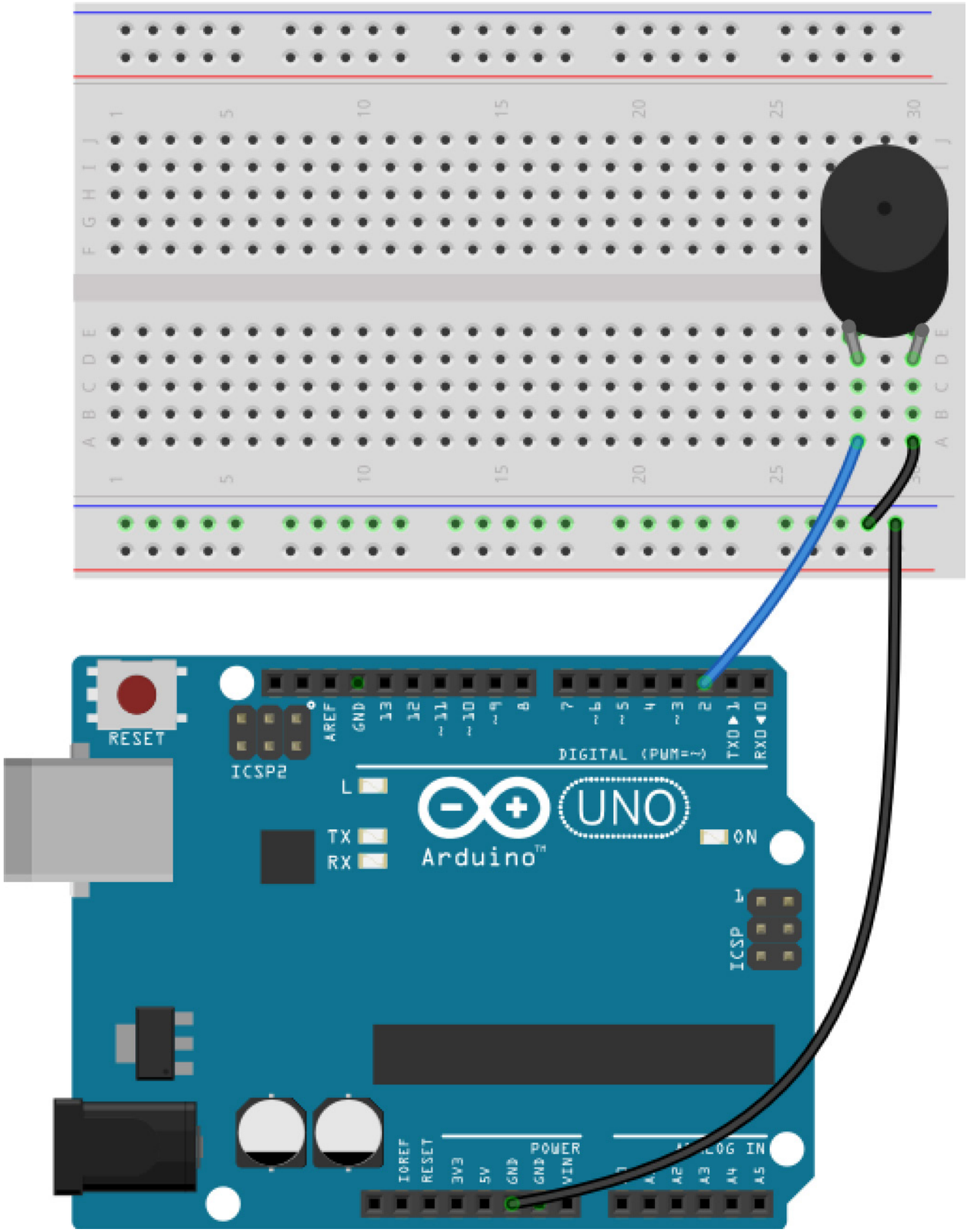
Piezo-Lautsprecher werden z.B. in elektronischem Spielzeug und in Grußkarten eingesetzt.

Code

Sketch

```
void setup(){

}

void loop(){
   tone(2, 400);       // Ton wird gespielt
   delay(200);         // Pause
   noTone(2);          // Ton wird ausgeschaltet
   delay(2000);        // Pause
}
```

Schaltung mit Piezo-Lautsprecher

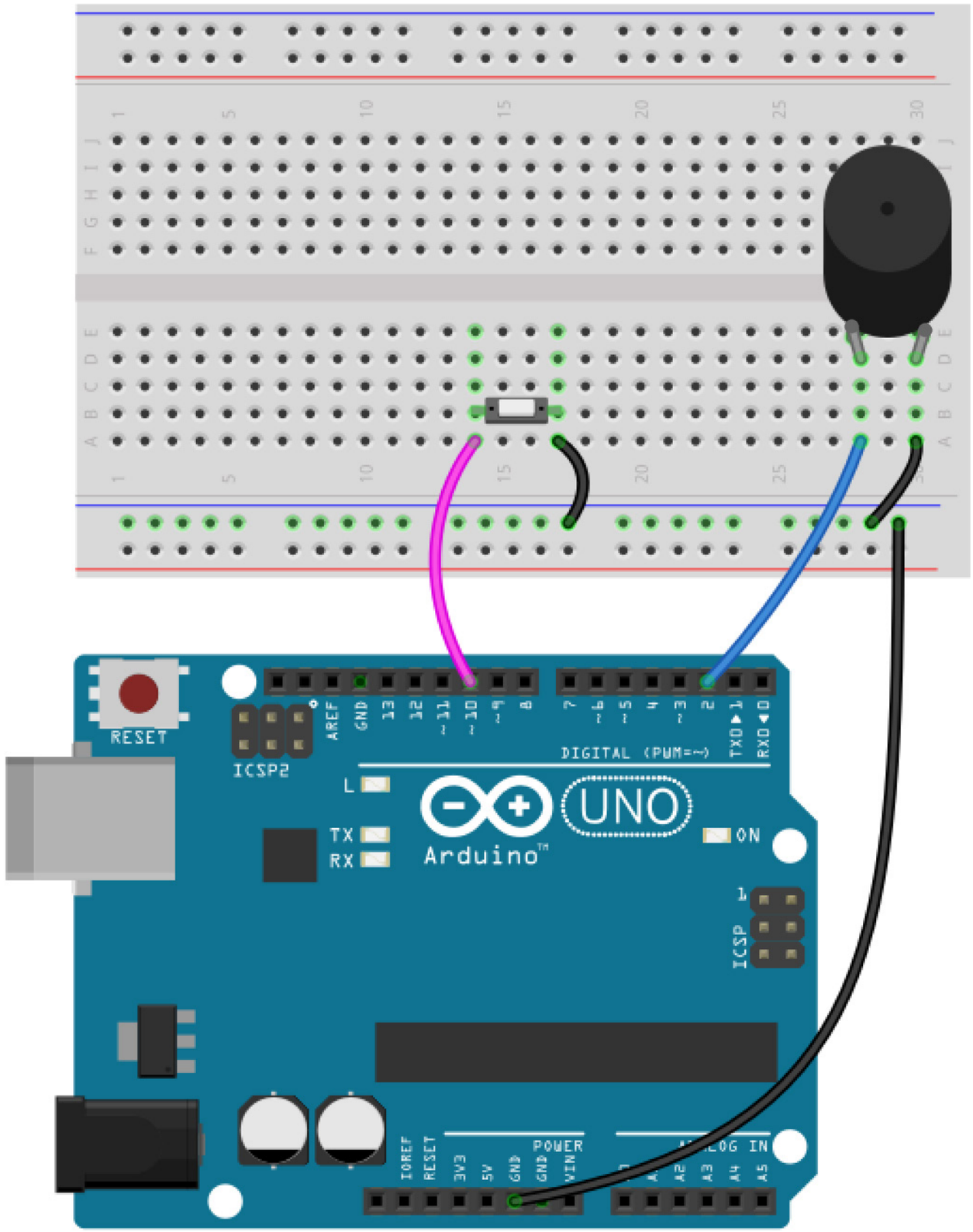


Baue die Schaltung nach

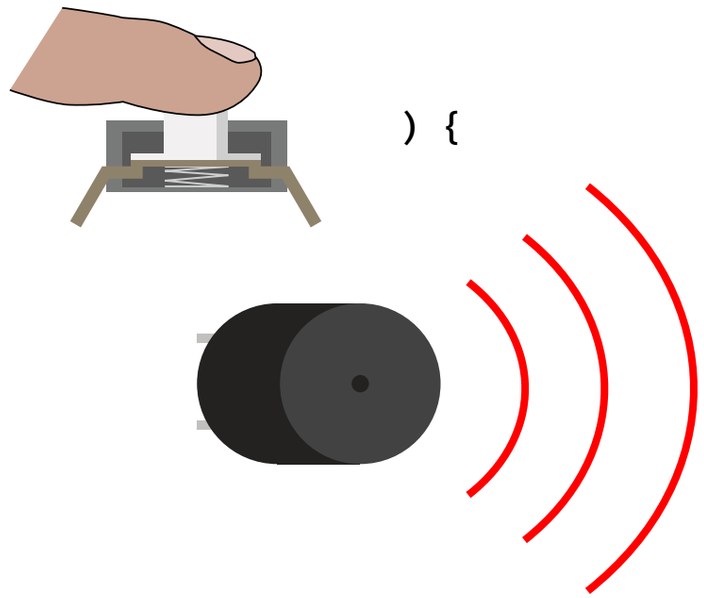
Wenn der Taster gedrückt ist, soll der Ton erklingen.

Wie programmiert man das?

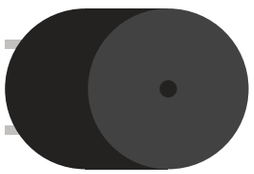
Schaltung mit Piezo-Lautsprecher



```
wenn (                    ) {
```

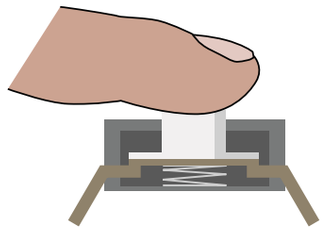


```
} sonst {
```

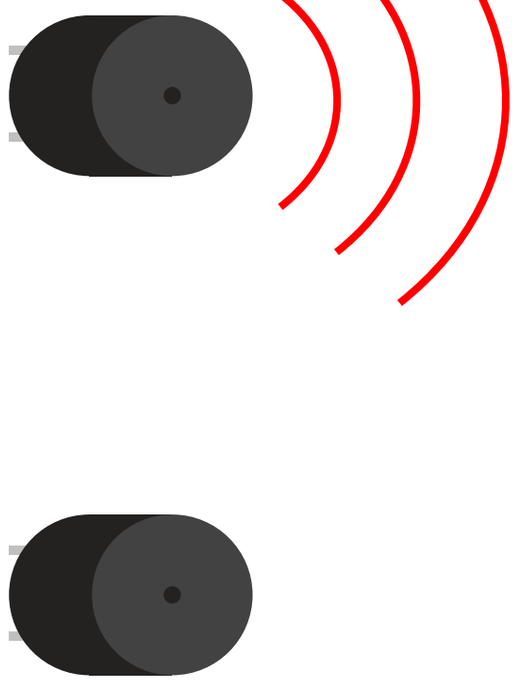


```
}
```

```
if( digitalRead(10)==LOW ){
```

wenn () {

```
    tone(2, 400);
```



```
} else {
```

} sonst {

```
    noTone(2);
```



```
}
```

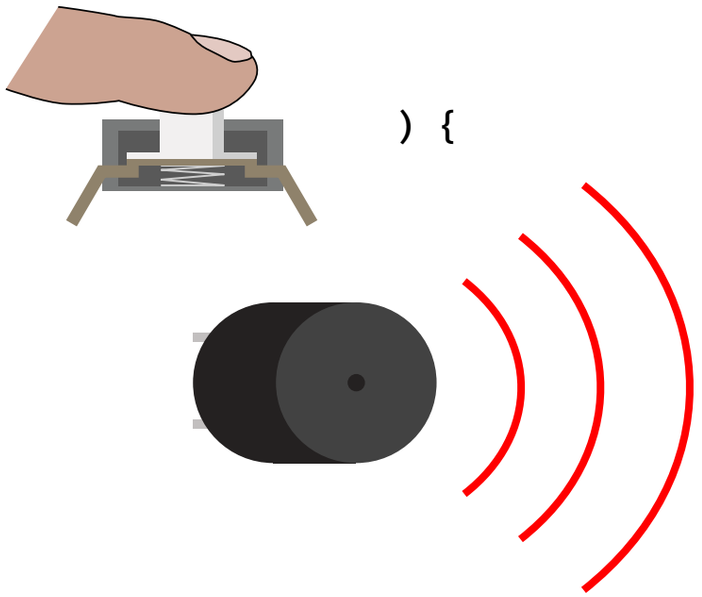
}

Code

Sketch

```
void setup(){
   pinMode(10,INPUT_PULLUP);       // Taster
}

void loop(){
   if(digitalRead(10)==LOW){
     tone(2, 400);
   } else {
     noTone(2);
   }
}
```

```
if (                          ) {

                                                    } else {

                                                    }
```



Melodie

Code

Sketch

```
void setup(){

}

void loop(){
   tone(2,262,400); // Pin, Tonhöhe, Tondauer
   delay(400);      // Pause zwischen den Noten
   tone(2,196,800); // Pin, Tonhöhe, Tondauer
   delay(800);      // Pause zwischen den Noten
   tone(2,196,800); // Pin, Tonhöhe, Tondauer
   delay(800);      // Pause zwischen den Noten
   tone(2,220,400); // Pin, Tonhöhe, Tondauer
   delay(400);      // Pause zwischen den Noten
   tone(2,196,400); // Pin, Tonhöhe, Tondauer
   delay(400);      // Pause zwischen den Noten
   tone(2,0,400);   // Pin, Tonhöhe, Tondauer
   delay(400);      // Pause zwischen den Noten
   tone(2,247,400); // Pin, Tonhöhe, Tondauer
   delay(400);      // Pause zwischen den Noten
   tone(2,262,400); // Pin, Tonhöhe, Tondauer
   delay(400);      // Pause zwischen den Noten
   noTone(2);       // Ton aus
   delay(5000);     // Pause am Ende
}
```

Die Melodie besteht aus Tonhöhen und Tonlängen:

```
262, 196, 196, 220, 196, 0, 247, 262
  4,   8,   8,   4,   4, 4,   4,   4
```

Schon bei einer kleinen Melodie wird der Code sehr schnell unübersichtlich. Besser wäre es, wenn die Melodie anders gespeichert werden könnte.

Ideen?

Code

Sketch

```
int melodie[] = {
  262, 196, 196, 220, 196, 0, 247, 262
};

void setup(){

}

void loop(){
   tone(2,melodie[0],4);

   delay(5000);      // Pause am Ende
}
```

Diese Konstruktion ist ein »Array«. Durch das Wort int kann es ganzzahlige Werte speichern.

Wenn man einen Wert auslesen will, schreibt man einfach die Stelle, die einen interessiert, in eckige Klammern. 0 ist die erste Stelle, also 262.

Jetzt brauchen wir noch etwas, das diesen Befehl wiederholt und dabei die Stelle verschiebt.

Code

Sketch

```
int melodie[] = {
  262, 196, 196, 220, 196, 0, 247, 262
};

void setup(){

}

void loop(){


   for (int i=0; i<8; i=i+1){
     tone(2,melodie[i],400);
     delay(400);
   }


   delay(5000);      // Pause am Ende
}
```

Die for-Schleife wiederholt Befehle eine festgelegte Dauer.

```
for (Startwert, Bedingung, Fortsetzung){
    Befehle
}
```

Startwert der Zählervariable sollte 0 sein

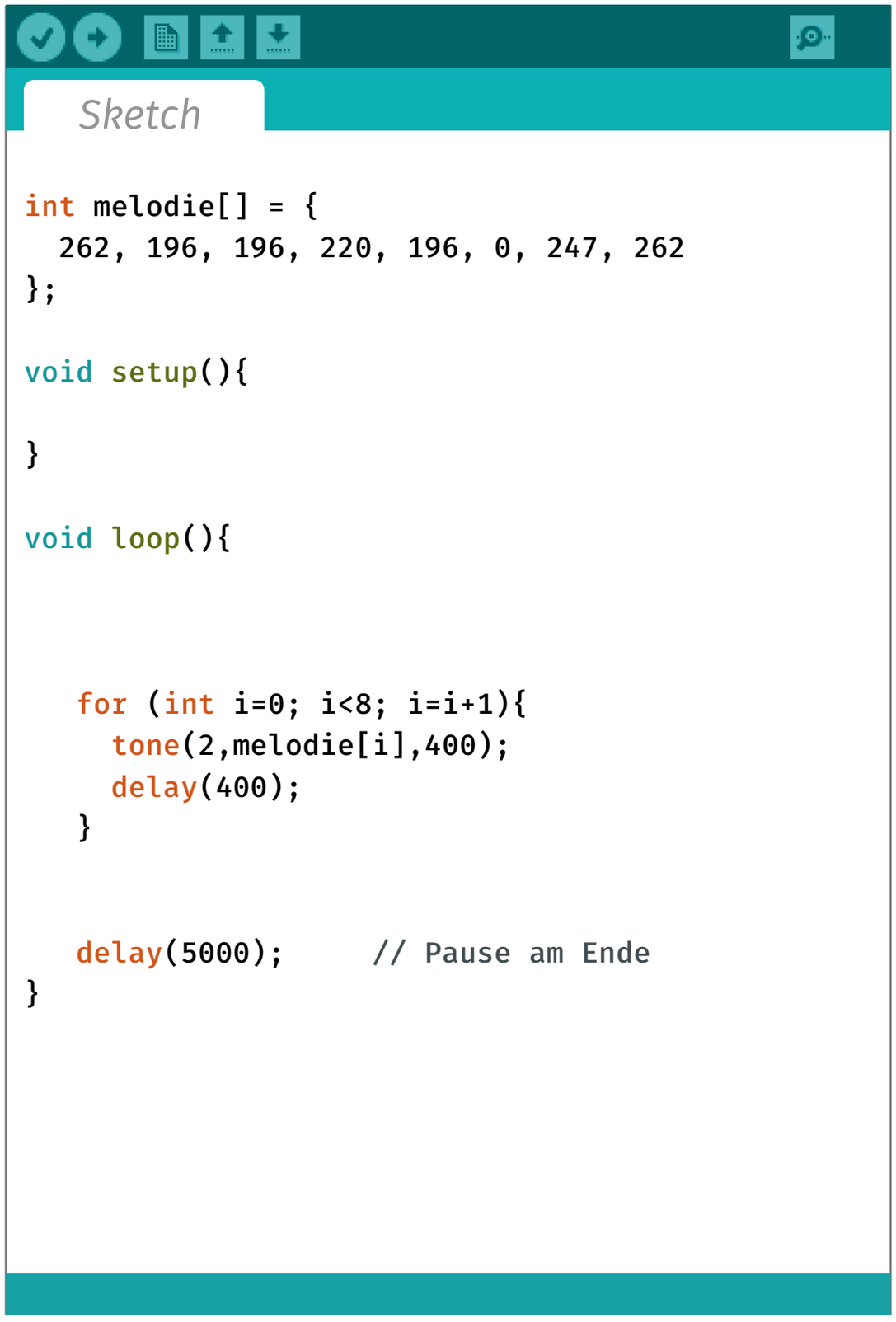
Bedingung: Solange Zähler kleiner als 8

Fortsetzung: Zählervariable pro Durchlauf +1

Die Variable i erhöht sich nun pro for-Schleifen-durchlauf um 1, fängt bei 0 an und endet bei 7.

Nun müssen wir sie nur noch ans Array schicken.

Code




```
int melodie[] = {
  262, 196, 196, 220, 196, 0, 247, 262
};

void setup(){

}

void loop(){



   for (int i=0; i<8; i=i+1){
     tone(2,melodie[i],400);
     delay(400);
   }


   delay(5000);      // Pause am Ende
}
```

Jetzt mach das Gleiche für die Tonlänge.

```
400, 800, 800, 400, 400, 400, 400, 400
```

Ein zweites Array soll die Tonlänge speichern.

Code

Sketch

```
int melodie[] = {
  262, 196, 196, 220, 196, 0, 247, 262
};

int tonLaenge[] = {
  400, 800, 800, 400, 400, 400, 400, 400
};

void setup(){

}

void loop(){

   for (int i=0; i<8; i=i+1){
     tone(2,melodie[i], tonLaenge[i]);
     delay(tonLaenge[i]);
   }

   delay(5000);      // Pause am Ende
}
```

Ein zweites Array speichert die Tonlängen.

Wir müssen die Tonlängen einmal für den Ton und einmal für die Pause übergeben.

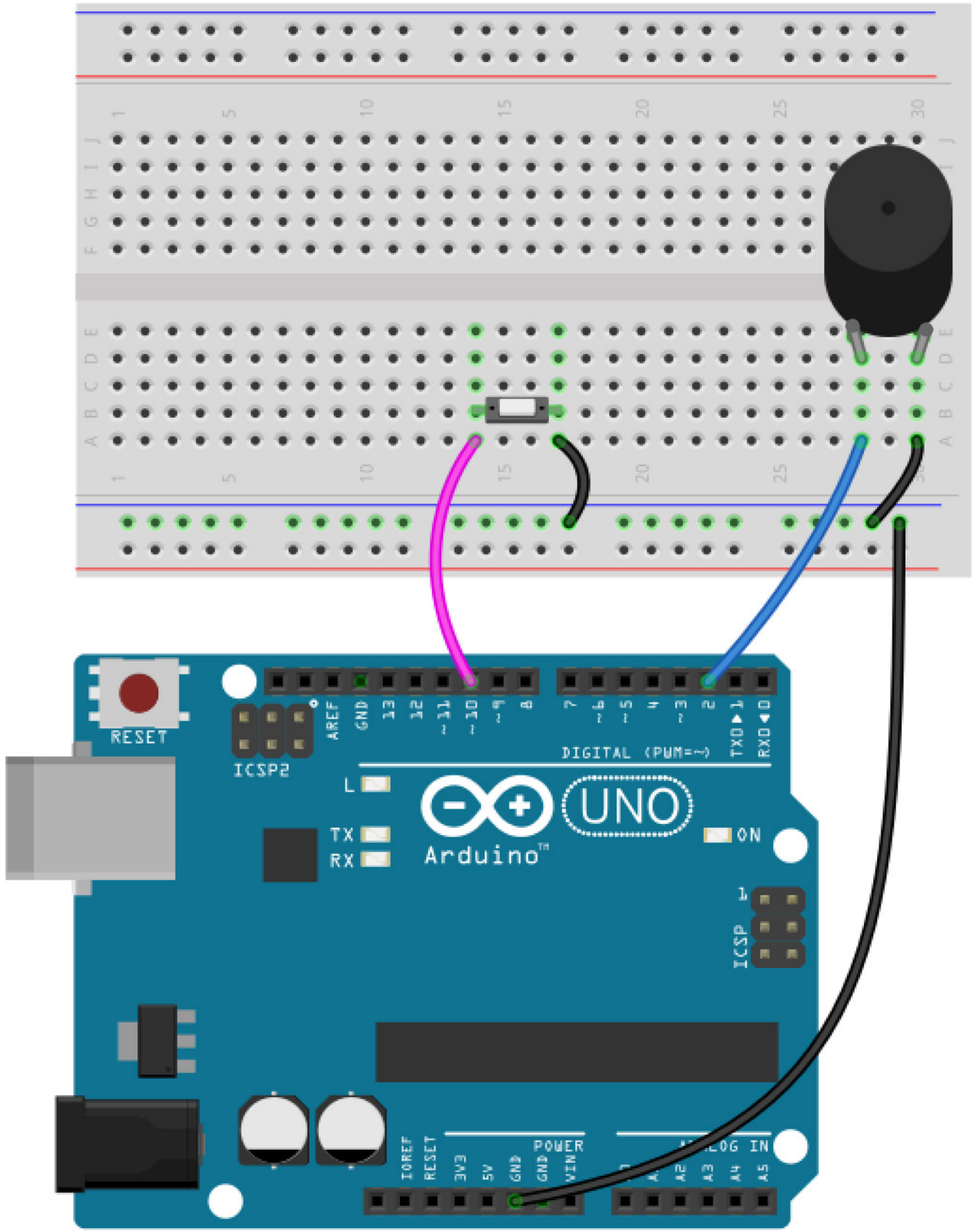
Ein-Tasten-Piano

Baue die Schaltung nach

Wir bauen ein Piano mit nur einer Taste!

Die Schaltung bleibt also gleich.

Schaltung mit Piezo-Lautsprecher



Code

Sketch

```
int melodie[] = {262,196,196,220,196,0,247,262};

int stelle = 0;

void setup(){
  pinMode(10,INPUT_PULLUP);
}

void loop(){
   if(digitalRead(10)==LOW){
    tone(2,melodie[stelle]);


    stelle = stelle+1;


    if(stelle>6) stelle=0;

   }
}
```

Immer, wenn der Taster gedrückt ist, soll der nächste Ton der Melodie gespielt werden. (Der Ton 0 kann übrigens gelöscht werden).

Die Variable `stelle` *soll die Stelle im Array* `melodie` *speichern, die als letztes abgespielt wurde.*

Wird der Taster gedrückt, wird der Ton aus dem `stelle`*-Array abgespielt.*

Die Variable »stelle« wird um eins erhöht.

Wird `stelle` *größer, als die Melodie, wird sie wieder auf 0 gesetzt.*

Nach dem Upload können wir festhalten: Das klingt noch nicht sehr gut. Woran könnte das liegen?

Code

Sketch

```
int melodie[] = {262,196,196,220,196,247,262};
int stelle = 0;

int tasterZustand = 0;
int letzterTasterZustand = 0;

void setup(){
  pinMode(10,INPUT_PULLUP);       // Taster
}

void loop(){
  tasterZustand=digitalRead(10);

  if(tasterZustand != letzterTasterZustand){
    if(tasterZustand == LOW){
     tone(2,melodie[stelle]);

     stelle = stelle+1;
     if(stelle>6) stelle=0;

    } else {
      noTone(2);
    }
  }
  letzterTasterZustand = tasterZustand;
}
```

Die Variable `stelle` *ändert sich, solange der Taster gedrückt wird, also ständig. Das Ergebnis ist noch nicht als unsere Melodie erkennbar.*

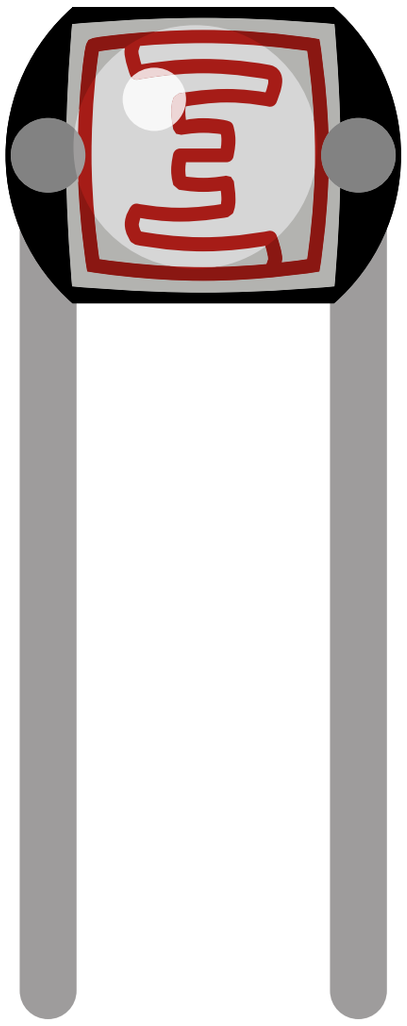
Wir wollen also im Grunde, dass sich die Variable `stelle` *nur ändert, wenn der Taster ausgelöst wird.*

Der Zustand des Tasters wird in der Variable `stelle` *gespeichert.*

Hat sich der Zustand des Tasters geändert (z.B., weil er jetzt gedrückt ist), wird das Programm abgearbeitet.

Am Ende wird letzterTasterZustand der aktuelle Zustand des Tasters zugewiesen.

Das Licht spielt Musik



Fotowiderstand

Ein Fotowiderstand reagiert auf Licht

Je mehr Licht auf den Fotowiderstand fällt, desto kleiner wird dessen Widerstandswert (Innenwiderstand). Je dunkler es ist, desto größer ist sein Widerstand.

Arduino kann diese Änderungen feststellen. Dazu brauchen wir aber immer eine Schaltung mit Fotowiderstand und passendem Festwiderstand (z.B. 100 kΩ).

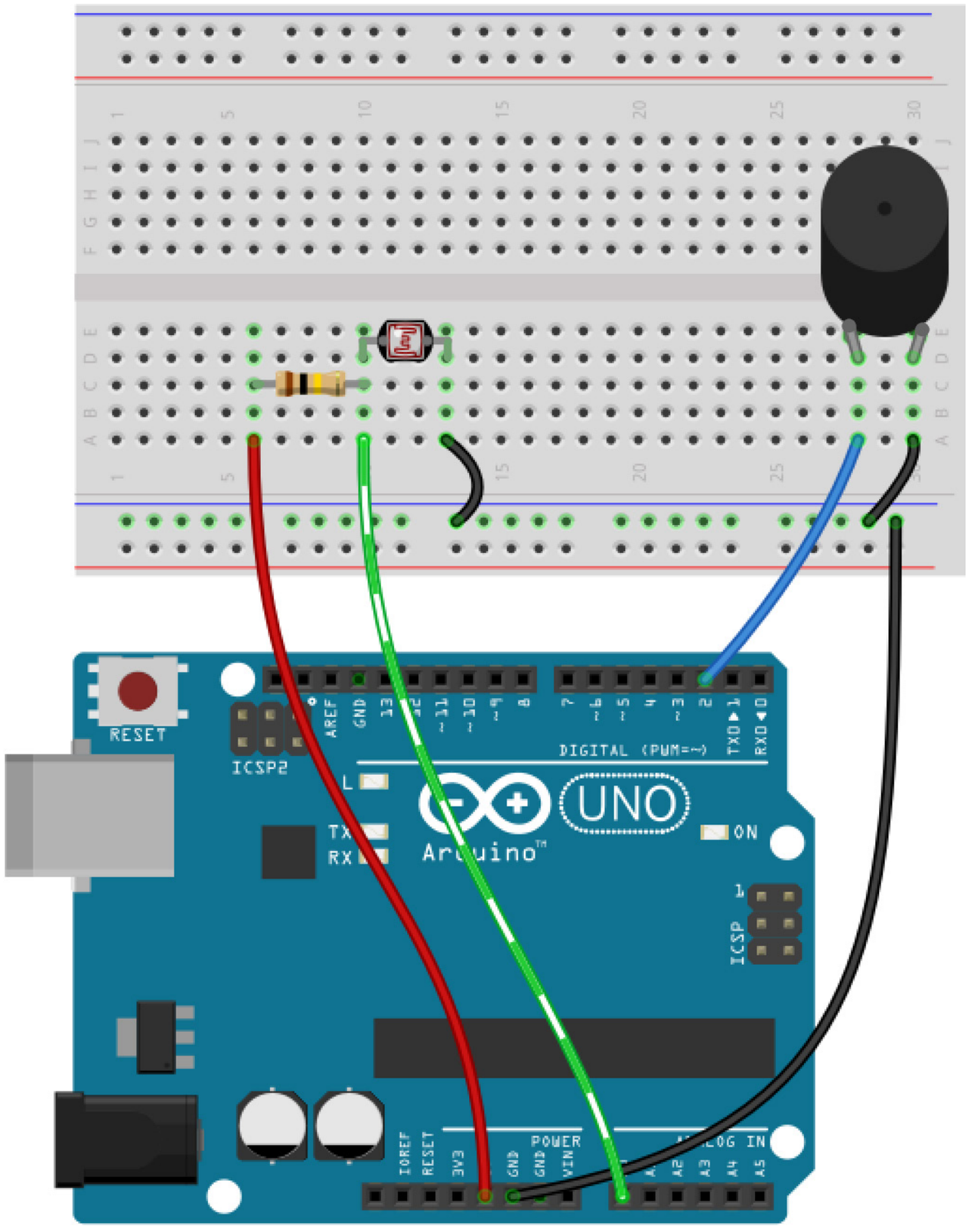
Festwiderstand 100 kΩ

Baue die Schaltung nach

Fotowiderstand und Festwiderstand sind in Reihe verbunden. Die eine Seite wird mit dem 5V (rotes Kabel), die andere mit dem GND (schwarz) verbunden.

Die Verbindung von beiden wird an einem Analog Input Pin angeschlossen (grün).

Schaltung mit Fotowiderstand



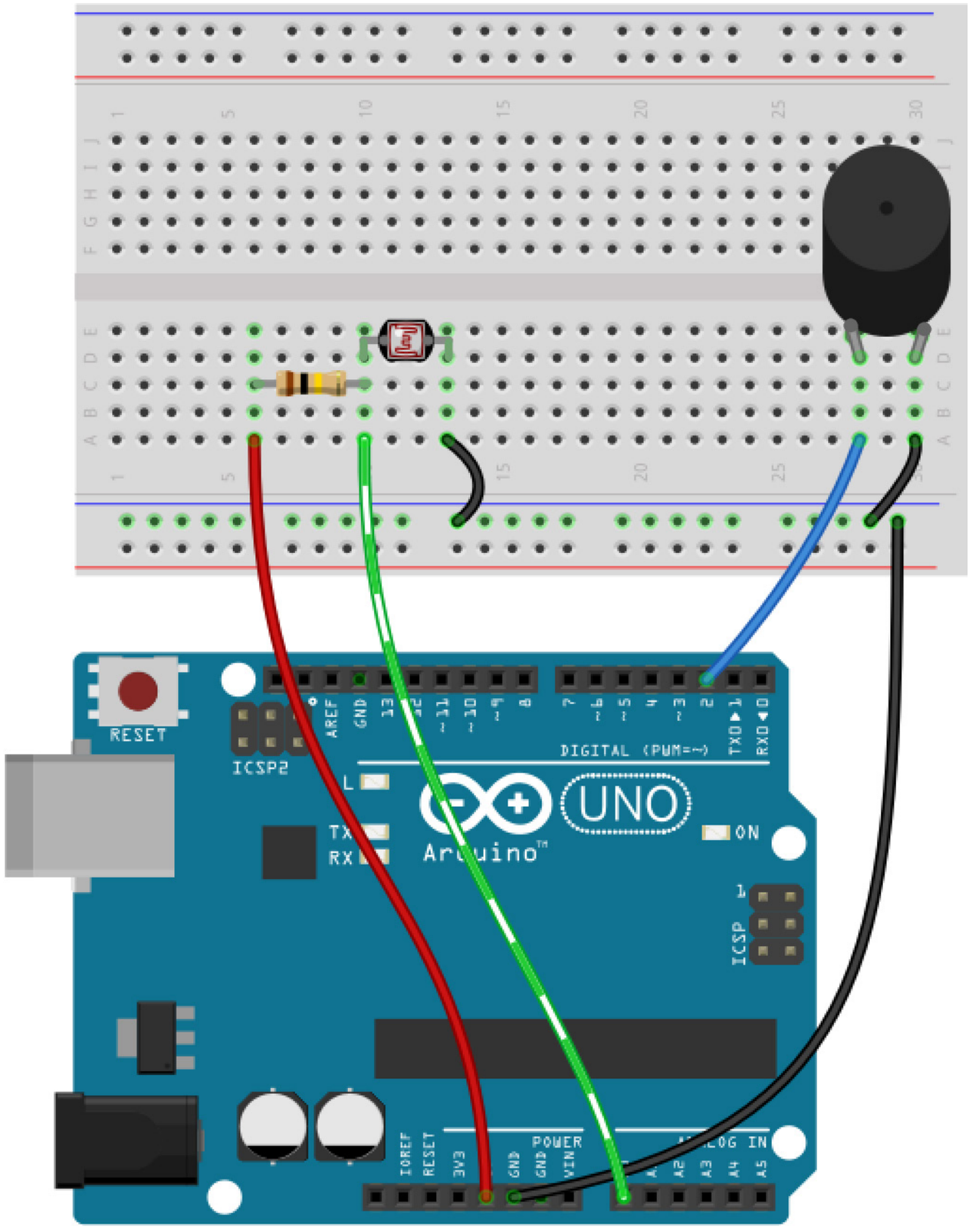
Code

Sketch

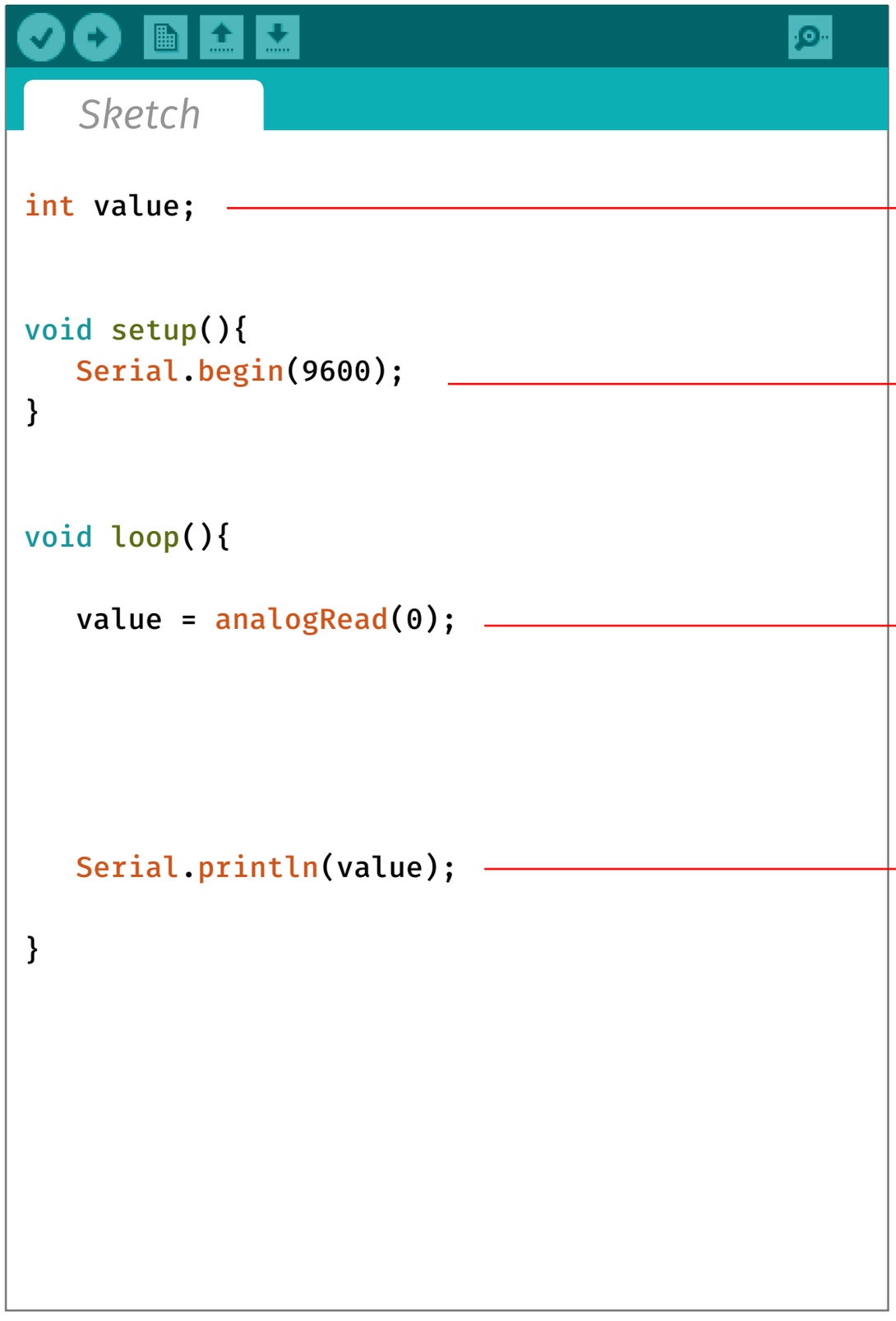
```
void setup(){

}

void loop(){
   value = analogRead(0);
}
```

Die analoge Eingabe (analogRead();) muss nicht im Setup deklariert werden.

Schaltung mit Fotowiderstand



Code



Diese Variable soll den Wert des Fotowiderstandes zwischenspeichern.

Startet die serielle Schnittstelle. Sie ermöglicht es, dass das Arduino Werte an den Computer sendet, die wir uns ansehen können.

Der Befehl analogRead(Pin) *liefert Werte zwischen 0 und 1023 – in unserem Beispiel in Abhängigkeit vom einfallenden Licht.*

Der Wert wird an die Serielle Schnittstelle gesendet.

Drücke auf das Lupen-Symbol (oben rechts) um den Seriellen Monitor zu starten. Welche Werte siehst Du?

Code

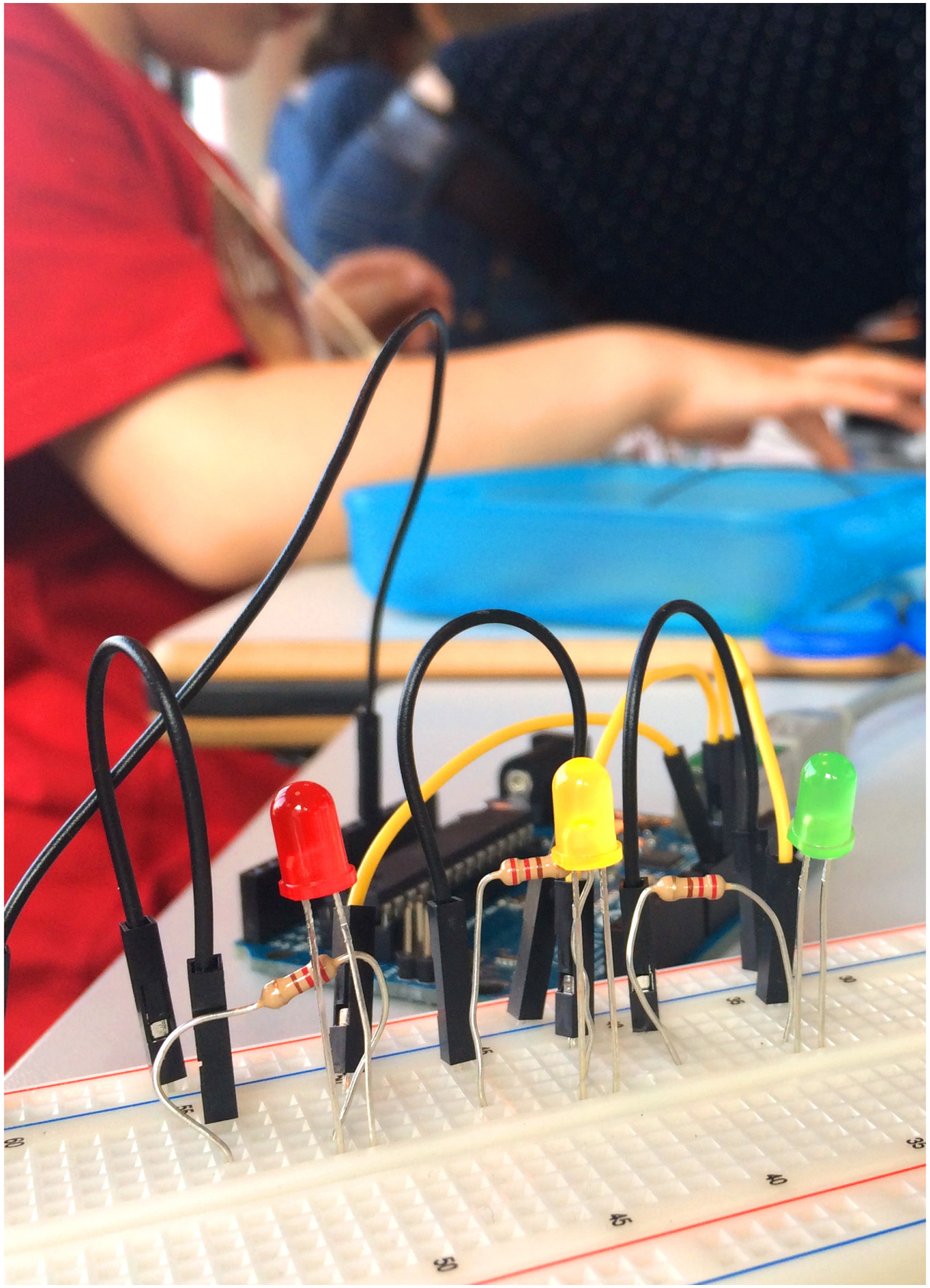
Sketch

```
int value;

void setup(){
   Serial.begin(9600);
}

void loop(){
   value = analogRead(0);
   tone(2,value+100);

   Serial.println(value);
}
```

Bei hohem Lichteinfall ist der Innenwiderstand des Fotowiderstandes klein, bei geringem Lichteinfall groß.

Jetzt erzeugen wir aus diesem Wert wieder ein Ton im Piezo-Lautsprecher. Dafür übergeben wir den Wert value an tone(); Wir addieren noch den Startwert 100 dazu.

Zum Schluss



Zusammenfassung

- *Stromkreise (Taschenlampe)*
- *das Arduino*
- *das Breadboard*
- *Befehle*
- *Morsen*
- *Ampel für Autos*
- *Ampel für Fussgänger und Autos*
- *Piezo-Lautsprecher*
- *if-Abfrage*
- *Ein-Tasten-Piano*
- *Variablen*
- *Analoge Eingabe*
- *Serielle Kommunikation*

Ende

Von der Schönheit und Eleganz programmierbarer Gegenstände

Präsentation, PDF

StartHardware.org

Stefan Hermann, 2015
Gülser Weg 23
12681 Berlin

hallo@starthardware.org