


Tinkercad Circuits Level 2: Programmiere einen Arduino und lass eine LED Blinken

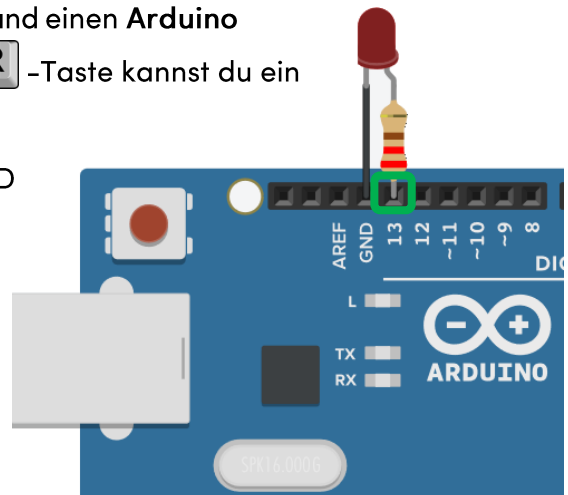
Mit „Tinkercad Circuits“ kannst du ganz einfach virtuelle elektrische Schaltungen bauen und dabei spielerisch die Welt der Elektronik entdecken! Du brauchst dafür nichts weiter als diese Anleitung und einen Computer oder ein Tablet mit Internetzugang. Für diese Anleitung solltest du bereits **Level 1** ausprobiert haben, in dem du lernst, wie du in Tinkercad Circuits Bauteile miteinander verbindest und eine Simulation startest.

So geht's:

1. Öffne deinen Internet-Browser und logge dich auf www.tinkercad.com ein.
2. Klicke links auf **Circuits** und dann auf **Neuen Schaltkreis erstellen**

3. Los geht's! Ziehe eine **LED**, einen **Widerstand** und einen **Arduino** auf die Arbeitsfläche (siehe rechts). Mit der  -Taste kannst du ein angeklicktes Bauteil drehen

4. Verbinde das lange geknickte Beinchen der LED (die „Anode“) mit dem Widerstand, und das andere Ende des Widerstands mit **Pin 13** am Arduino (siehe rechts). Das kurze gerade Beinchen der LED (die „Kathode“) wird mit einem Draht mit dem „GND“ Pin verbunden.

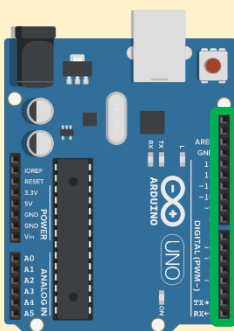


Tipp: Du kannst verbundene Bauteile gemeinsam bewegen, indem du sie

zusammen auswählst. Halte dafür die  - Taste und ziehe mit gedrückter linker Maustaste einen Rahmen über die Bauteile. Verschiebe danach alles


zusammen mit gedrückter linker Maustaste, oder den Pfeiltasten    .


Was ist ein Arduino?



Ein Arduino ist ein kleiner Computer, den du selbst programmieren kannst. Mit ihm kannst du LEDs ein- und ausschalten, Druck- und Drehknöpfe lesen, Sensormessungen an deinen Computer schicken und vieles mehr. Die Bauteile schließt du an den **Pin-Anschlüssen** an. Der Arduino kann alles was du brauchst, um elektronische Musikinstrumente, nützliche Maschinen, oder sogar einen Aufräum-Roboter zu bauen!

5. Klicke auf den Widerstand  und ändere im kleinen Fenster rechts oben seinen Wert auf 220 Ω. Achte auf die richtige Einheit Ω !


Widerstand	
Name	R1
Widerstand	220 

6. Klicke auf  **Simulation starten** . Die Led sollte nun blinken!

Blinkt deine LED nicht? → Überprüfe, ob alle Teile miteinander verbunden sind.

Klicke auf  **Simulation stoppen** , bevor du weitermachst.

Warum blinkt die LED eigentlich? Der Programm-Code macht's!

7. Klicke rechts oben auf  **Code** und wähle im Menü darunter „Text“ aus (bestätige die Warnung). Jetzt siehst du rechts den Programmcode, der die LED blinken lässt. Der Code ist für den Arduino eine Art „Schritt-für-Schritt Anleitung“ – er macht genau das, was du ihm sagst!

Text
Blöcke
Blöcke und Text
Text

Arduino Code-Struktur

Ein Arduino-Programm besteht aus den zwei Hauptfunktionen **setup()** und **loop()**. Alle Befehle die zwischen den geschwungenen Klammern { } stehen, werden vom Arduino nacheinander ausgeführt. Nach jedem Befehl **muss** ein Strichpunkt ; gesetzt werden, sonst kommt es zu Fehlern.

```
1 void setup()
2 {
3   pinMode(13, OUTPUT);
4 }
5
6 void loop()
7 {
8   digitalWrite(13, HIGH);
9   delay(1000); // Wait for 1000 millisecond(s)
10  digitalWrite(13, LOW);
11  delay(1000); // Wait for 1000 millisecond(s)
12 }
```

Alles was in **setup()** steht, wird nach dem Einschalten nur einmal ausgeführt.

Alles was in **loop()** steht, wird danach endlos wiederholt.

Doppelte Schrägstriche // Kennzeichnen einen Kommentar. Alles was **danach** in derselben Zeile steht (in Orange), wird vom Arduino ignoriert. Du kannst // verwenden, um dir im Code Notizen zu machen!

8. Sehen wir uns den Code etwas genauer an! In **Zeile 3** in der `setup()`-Funktion:

```
3 pinMode(13, OUTPUT);
```

wird nur festgelegt, dass später vom **Pin 13** Strom „ausgegeben“ wird. In der `loop()`-Funktion geht das Programm dann richtig los! Der Code in **Zeile 8**

```
8 digitalWrite(13, HIGH);
```

bewirkt, dass **Pin 13** (an den du die LED angeschlossen hast) „eingeschaltet“ und mit Strom versorgt wird. Die LED leuchtet! Der Befehl in der nächsten Zeile:

```
9 delay(1000); // Wait for 1000 millisecond(s)
```

lässt den Arduino für 1000 Millisekunden (= 1 Sekunde) warten. Der **Pin 13** wird in dieser Zeit weiter mit Strom versorgt und die LED leuchtet, sonst passiert nichts.

```
10 digitalWrite(13, LOW);
```

Dieser Befehl schaltet den Strom auf **Pin 13** wieder aus. Die LED leuchtet nicht mehr.


9. Jetzt ist es Zeit selbst zu programmieren! Ändere die Dauer der beiden Wartezeiten von `delay(1000)` auf `delay(200)`. Was Passiert? Probiere verschiedene Werte aus!

Bonus-Level!

Mache folgenden Änderungen an deinem Schaltkreis und im Programm-Code (*vergiss nicht die Simulation zu stoppen/starten*):



- Verbinde den Widerstand mit **Pin 11** anstelle von **Pin 13**. Blinkt die LED, wenn du die Simulation startest?
- Suche die Stellen im Programmcode, die du ändern musst, um die LED wieder zum Blinken zu bekommen. Tipp: *Du musst den Code an 3 Stellen ändern: 1 x in der `setup()`-Funktion und 2 x in der `loop()`-Funktion.*
- Versuche 2 LEDs anzuschließen und sie abwechselnd blinken zu lassen!

Tipp: Klicke oben rechts auf  Code, um wieder in den „Baumodus“ zu kommen.

Im **techLAB** haben wir „echte“ Arduinos und viele elektronische Bauteile, mit denen du experimentieren kannst! Im **techLAB** gibt es aber noch viel mehr spannende Technologien zu entdecken. Arbeite mit 3D-Drucker, Laser Cutter, Stickmaschine und mehr im **Technischen Museum Wien**! Die Öffnungszeiten des **techLAB** findest du [online](#). Der Eintritt ins Museum ist unter 19 Jahren gratis!

Wir freuen uns auf deinen Besuch!