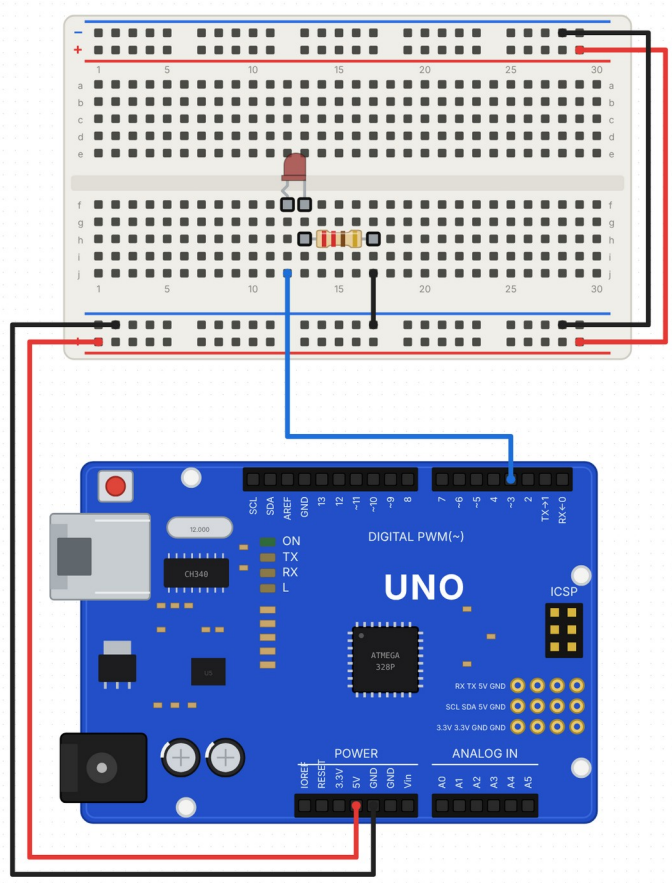
Codekäfer

Programmieren

mit dem Codekäfer

Einstieg: Aktoren, Sensoren und erste Projekte



Name	Klasse	Platz-Nr.

Inhalt: Erste Schritte · Der Arduino Uno · Digital, analog und PWM · LEDs · LCD-Display · Töne · Servo und Lüfter · Taster · Analoge Sensoren · Projekte · Anhang (weitere Aktoren und Sensoren)

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

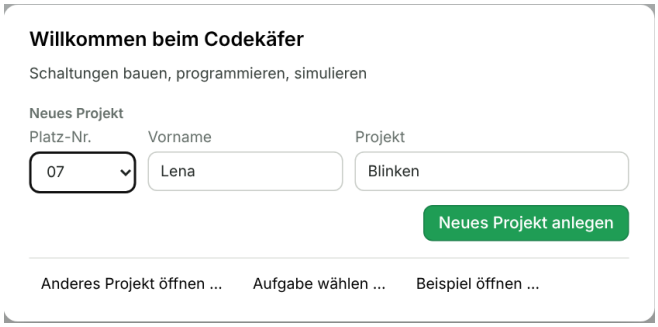
Erste Schritte mit dem Codekäfer

Im **Codekäfer** baust du die Schaltung auf einer Werkbank zusammen, programmierst sie mit Blöcken, probierst alles in der **Simulation** aus und überträgst das Programm dann mit einem Klick auf den echten Arduino.

Projekt anlegen

1. Starte **„Codekaefer starten“**.
2. Wähle deine **Platz-Nummer** und gib deinen **Vornamen** ein.
3. Gib dem Projekt einen Namen, z. B. **Blinken**, und klicke auf **„Neues Projekt anlegen“**.

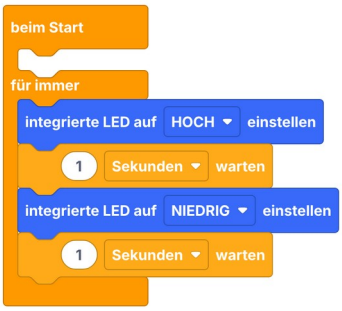
Der Codekäfer speichert alles **automatisch**. Beim ersten Start zeigt dir eine kurze Tour die wichtigsten Knöpfe (wieder aufrufen: **„?“** unten links).



Erster Test: die eingebaute LED „L“

Der Arduino hat eine eingebaute Leuchtdiode **„L“**, die du ohne zusätzliche Bauteile programmieren kannst.

1. Ziehe aus der Bauteilleiste rechts den **„Arduino Uno“** auf die Werkbank.
2. Klicke oben auf **„Code“**. Dort steht schon der Startblock mit **„beim Start“** und **„für immer“**.
3. Baue das Programm rechts nach. Die Blöcke findest du links unter **„Aktoren“** (blau) und **„Steuerung“** (orange).



Was in **„beim Start“** steht, läuft einmal, wenn der Arduino startet. Was in **„für immer“** steht, wiederholt sich ständig.

Aufgaben

a) Überlege, was das Programm macht.

b) Überprüfe deine Vermutung mit **„Simulation starten“**. Schau auf die LED „L“ neben dem USB-Anschluss.

c) Erkläre das Programm deinem Nachbarn.

d) Was verändert sich, wenn du den letzten „warten“-Block löschst – und warum? Vermute zuerst!

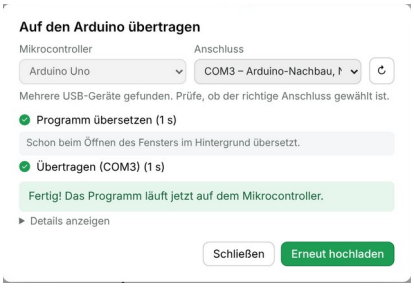
e) Die LED soll nur **einmal** ein- und wieder ausgeschaltet werden. Wie muss das Programm verändert werden?

f) **Für Experten:** Lass die LED genau **5-mal** blinken. Die Blöcke für das Ein- und Ausschalten darfst du jeweils nur einmal benutzen! *(Tipp: Kategorie „Steuerung“)*

Programm auf den Arduino übertragen

Läuft alles in der Simulation, verbindest du den Arduino mit dem **USB-Kabel** und klickst oben rechts auf **„Hochladen“**. Der Codekäfer findet den Anschluss selbst, übersetzt das Programm und überträgt es. Bei **„Fertig!“** läuft es auf dem Arduino.

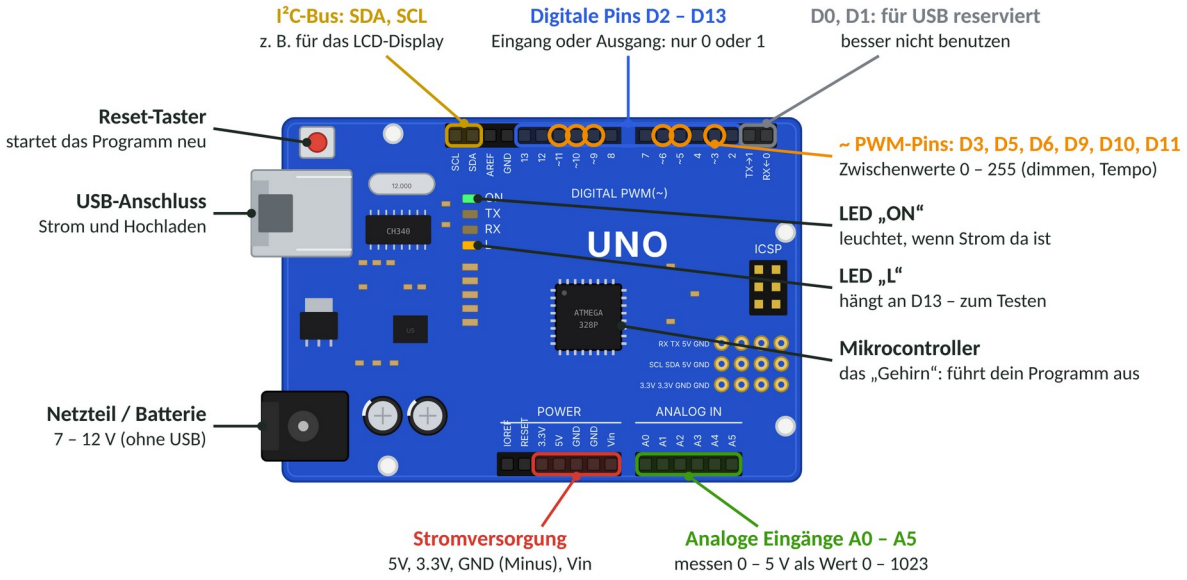
Das erste Hochladen dauert etwas länger. Klappt es nicht, probiere ein anderes USB-Kabel.



© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Der Arduino Uno

Bevor wir weiter programmieren, lernst du das Board kennen – vor allem seine **Anschlüsse** (die **Pins**). An sie steckst du Bauteile mit Kabeln an.



Ausgänge und Eingänge

	Was er macht	Beispiele	Blöcke im Codekäfer
Ausgang	schaltet oder steuert einen **Aktor**	LED, Summer, Servo, Lüfter, Display	Kategorien **„Aktoren“** und **„Aktor …“**
Eingang	liest den Wert eines **Sensors**	Taster, LDR, NTC, Poti	Kategorien **„Sensoren“** und **„Sensor …“**

Gut zu wissen

- Ein Ausgang liefert höchstens etwa **40 mA**. Für Motoren oder Lampen braucht man einen Transistor, einen Motortreiber oder ein Relais.
- Ist ein **Display** angeschlossen, sind **A4** und **A5** belegt: Sie sind dieselben Leitungen wie **SDA** und **SCL**.
- **GND** ist der Minuspol. Jede Schaltung braucht eine Verbindung zu GND.

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Digital, analog und PWM

Der Arduino kennt zwei Arten von Signalen. Wie bei einer Lampe: Ein **Lichtschalter** kennt nur „an“ und „aus“, mit einem **Dimmer** gibt es alle Stufen dazwischen.

Digital — wie ein Lichtschalter

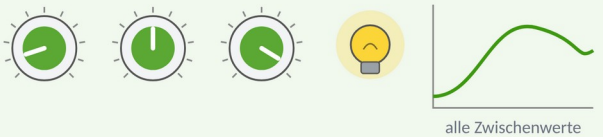


0 = aus = NIEDRIG (0 V) **1** = an = HOCH (5 V)

Pins: **D2 – D13** (als Eingang oder als Ausgang)

Sensoren: Taster, Bewegungsmelder, Sensortaste
Aktoren: LED, Lüfter, Summer

Analog — wie ein Dimmer



0 (0 V) **511** (2,5 V) **1023** (5 V)

Pins: **A0 – A5** (nur als Eingang)

Sensoren: LDR, NTC, Poti, Feuchtefühler
Aktoren „analog“? → mit PWM, siehe unten

Digitale Pins

- Als **Eingang**: Eine Spannung über **2,5 V** erkennt der Arduino als **1** (HOCH), eine niedrigere als **0** (NIEDRIG).
- Als **Ausgang**: **HOCH** liefert etwa **5 V**, **NIEDRIG 0 V**.

Analoge Pins

Sie messen Spannungen von **0 V** bis **5 V** und machen daraus Zahlen von **0** bis **1023**. So kann der Arduino zum Beispiel erkennen, wie hell oder wie warm es ist.

PWM: „Zwischenwerte“ an den Pins mit ~

Einen echten analogen **Ausgang** hat der Arduino nicht. Die Pins mit der Tilde **~** (**D3, D5, D6, D9, D10, D11**) können aber sehr schnell ein- und ausschalten. Man gibt Werte von **0** (immer aus) bis **255** (immer an) an. Bei **127** ist der Pin die Hälfte der Zeit an: Eine LED leuchtet dann schwächer, ein Motor läuft langsamer – unser Auge sieht das schnelle Blinken nicht.

PWM: Pulsweitenmodulation — schnelles Ein und Aus an den Pins mit ~



Achtung im Codekäfer

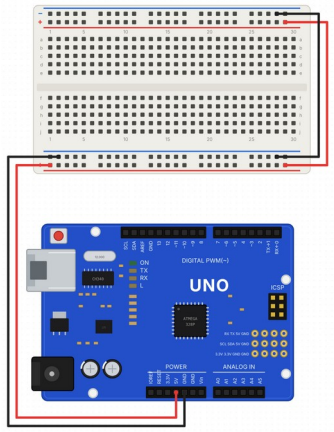
Der Block **„PWM-Ausgang … einstellen“** wirkt in der Simulation **nur an den Pins mit ~**. An anderen Pins passiert nichts, und unten erscheint ein Hinweis.

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Externe LEDs anschließen

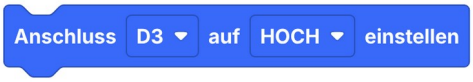
Bauteile steckst du auf eine **Steckplatine**. In der Bauteilleiste gibt es ganz oben die Vorlage **„Arduino + Steckplatine“**. Darin sind die rote Leiste (+) mit **5V** und die blaue Leiste (–) mit **GND** schon verbunden.

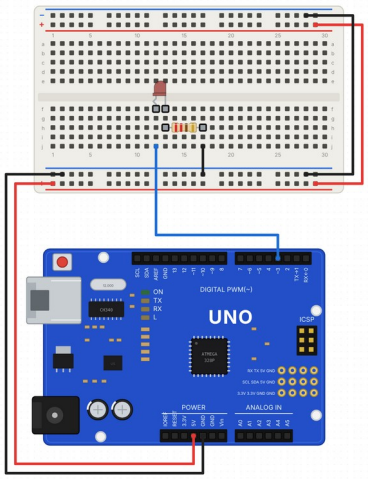
In jeder **Spalte** sind die fünf Löcher a–e bzw. f–j miteinander verbunden, in den Leisten am Rand jeweils die ganze Reihe.

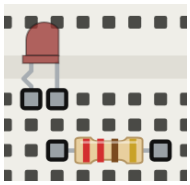


So schließt du eine LED an

1. **Anschluss festlegen** – im Beispiel **D3**.
2. LED und Widerstand auf die Steckplatine stecken und verbinden:
   - Vorwiderstand **220 Ω** nicht vergessen!
   - **Polung:** Das lange Bein (Anode, +) zeigt zu D3, das kurze Bein (Kathode, –) über den Widerstand zu **GND**.
   - **Kabel:** auf einen Anschluss klicken, dann auf das Ziel. **Drehen:** Taste **R**.
3. **Programmieren** mit dem Block **„Anschluss D3 auf HOCH einstellen“** aus **„Aktoren“**:







> **Der Codekäfer hilft mit**
>
> Steckt die LED verkehrt herum oder fehlt der Vorwiderstand, meldet der Codekäfer das unten unter **„Prüfung“**. Der Knopf **„Hilfe“** oben gibt dir Tipps.

Aufgaben

a) Schalte eine externe LED an **D3** ein.

b) Lass die LED blinken.

c) Schließe eine zweite LED an **D5** an (eigener Vorwiderstand!).

d) Lass die beiden LEDs abwechselnd blinken.

e) Lass die erste LED für 10 Sekunden nur **schwach** leuchten. Nutze dazu den Block **„PWM-Ausgang D3 auf ... einstellen“**.

f) **Für Experten:** Dimme die LED von AUS nach AN (von 0 bis 255). Das soll sich fortlaufend wiederholen. *(Hilfe auf der nächsten Seite)*

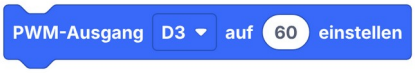
© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

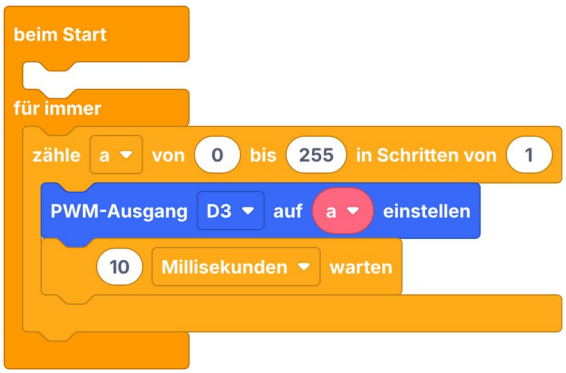
Hilfe zur Experten-Aufgabe: LED dimmen

Mit einer Zählschleife

Am einfachsten geht es mit dem Block **„zähle … von … bis …“** aus **„Steuerung“**. Er legt die **Variable a** selbst an und zählt sie hoch. Mit jedem Durchgang wird die LED ein bisschen heller.

Die Variable **a** ziehst du aus **„Variablen/Listen“** in den PWM-Block:

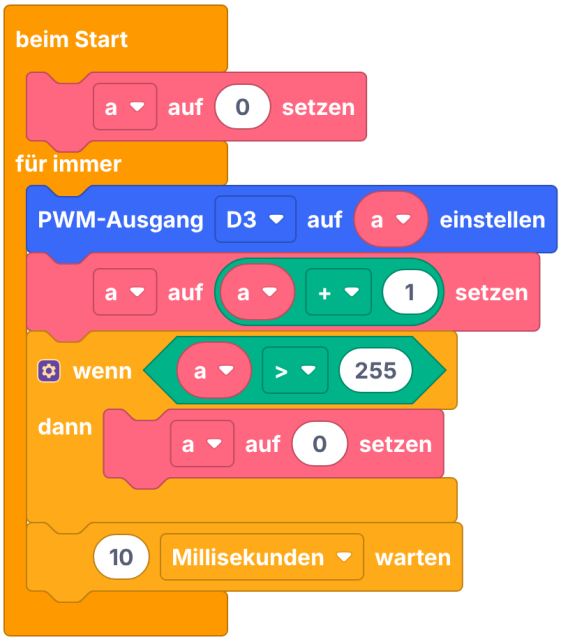




Selbst hochzählen

So geht es auch: Lege unter **„Variablen/Listen“** › **„Variable erstellen“** eine Variable **a** an. Bei jedem Durchlauf wird **a** um 1 größer. Ist **a** größer als 255, beginnt es wieder bei 0.

Vergleiche beide Programme: Was ist gleich, was ist anders?



© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

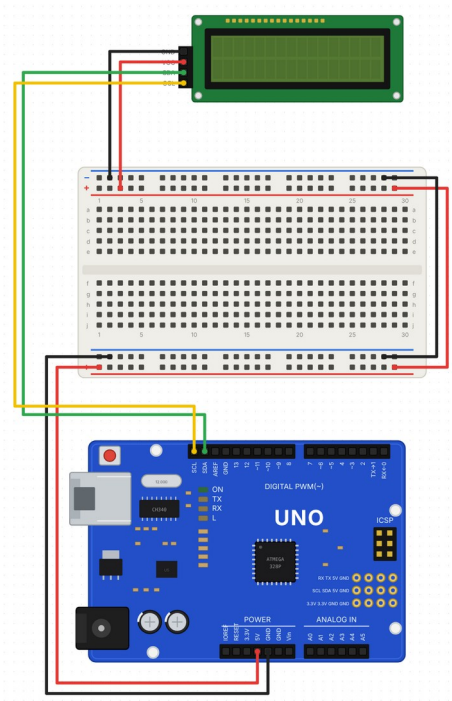
Das LCD-Display 16 × 2

Unser LCD-Display hat einen kleinen Controller für den **I²C-Datenbus**. Dadurch braucht es nur **vier Kabel**. Es hat **16 Spalten** (0 bis 15) und **2 Zeilen** (0 und 1).

1) Display anschließen

Nimm die Vorlage **„Arduino + Steckplatine + LCD“**. GND und VCC sind schon verbunden – **SDA und SCL verbindest du selbst**:

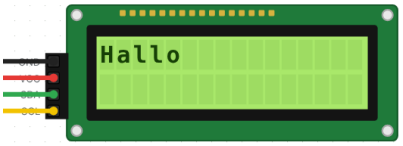
LCD-Display	Arduino
GND	GND
VCC	5V
SDA	SDA (oder A4)
SCL	SCL (oder A5)



2) Programmieren

Das Display muss in **„beim Start“** einmal eingerichtet werden. Die Blöcke findest du unter **„Aktor Anzeige“**:







Aufgaben

Starte jeweils die Simulation. Läuft alles wie geplant? Dann lade das Programm auf den Arduino!

a) Lass „Hallo“ in der oberen Zeile erscheinen. Schalte mit **„LCD-Hintergrundlicht einschalten“** das Licht an.

b) In der oberen Zeile soll abwechselnd **„Hallo“** und **„Du“** erscheinen. *Tipp: Bis auf das Einrichten und das Hintergrundlicht brauchst du jeden Block genau zweimal.*

c) Was ändert sich, wenn du den Block **„LCD-Bildschirm leeren“** weglässt?

d) Programmiere einen **Countdown**, der von 10 bis 0 zählt.

e) **Für Profis:** Mit einer Zählschleife wird der Countdown viel kürzer (siehe „LED dimmen“). Er soll in der **Mitte** der ersten Zeile erscheinen.

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Wir erzeugen Töne – der Piezo-Summer

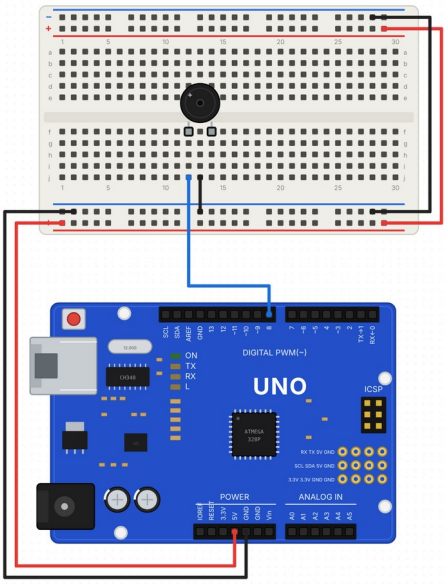
Der Piezo-Summer knackt bei jedem Stromimpuls. Folgen die Impulse ganz dicht aufeinander, hören wir einen **Ton**. Bei 440 Impulsen pro Sekunde (**440 Hertz**) hört man den Kammerton **A4**.

Im Codekäfer wählst du im Block einfach die **Note** (C, D, E …), ein **Vorzeichen** (# oder b) und die **Oktave** (4 = mittlere Lage). Die Frequenz rechnet der Codekäfer selbst aus.

Note	Frequenz in Hz
C4	262
D4	294
E4	330
F4	349
G4	392
A4	440
H4	494
C5	523

1) Anschluss

Ein Anschluss des Summers (+) kommt an einen digitalen Pin (im Beispiel **D8**), der andere an **GND**.



2) Programmieren

Die Blöcke findest du unter **„Aktoren“**:



Mit **„Ton mit … Hz“** spielst du jede beliebige Frequenz.

Aufgaben

a) Programmiere eine **Tonleiter** (Noten C4 bis C5 der Reihe nach).

b) Programmiere eine **Melodie** deiner Wahl!

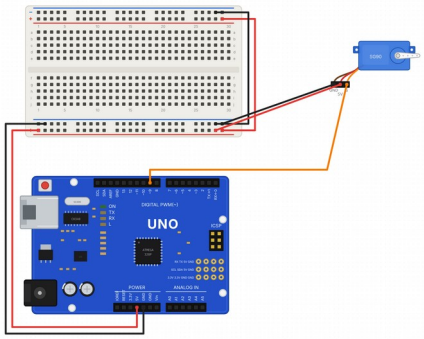
c) **Für Profis:** Programmiere eine **Sirene**: Erhöhe die Frequenz mit **„Ton mit … Hz“** in einer Schleife Schritt für Schritt (z. B. von 400 bis 800 Hz) und verringere sie danach wieder. Packe alles in eine Schleife, die 10-mal durchlaufen wird.

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

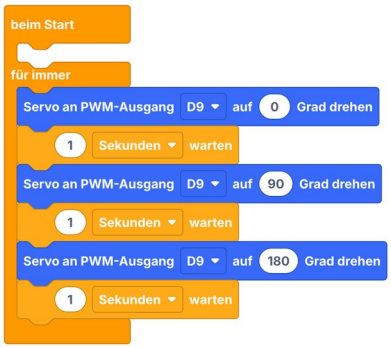
Der Servomotor

Ein Servo dreht seine Achse **gradgenau** auf einen Winkel – ideal für Links-/Rechts- oder Auf-/Ab-Bewegungen (z. B. eine Schranke). Unser **SG90** dreht bis zu **180°**.

Servo	Kabel	Arduino
GND	braun	GND
VCC	rot	5V
Puls	orange	z. B. D9



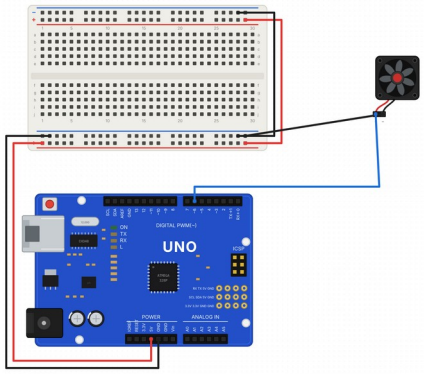
Programmieren: Block **„Servo an PWM-Ausgang … auf … Grad drehen“** aus **„Aktoren“**, Winkel von **0°** bis **180°**.



Der Lüfter

Der Lüfter läuft bei **5 V** und steht bei **0 V** – ein „digitaler“ Aktor. Im Codekäfer heißt er **„Lüfter“**.

Lüfter	Kabel	Arduino
+	rot	z. B. D6
–	schwarz	GND



- Block **„Anschluss … auf … einstellen“** (**„Aktoren“**), Anschluss **D6**.
- **HOCH** schaltet ihn ein, **NIEDRIG** wieder aus.
- Mit **„PWM-Ausgang D6 auf …“** läuft er langsamer (D6 hat eine ~).

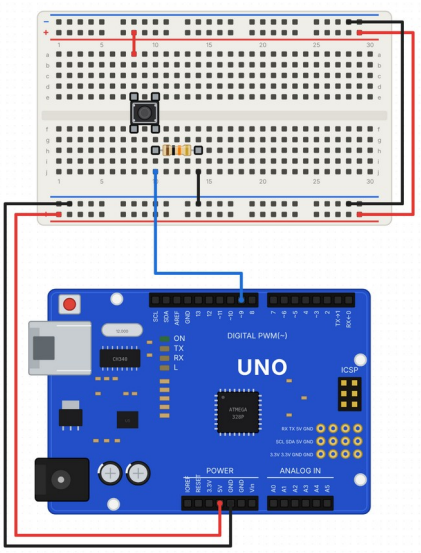


© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Der Taster

Ein Taster ist ein **digitaler Sensor**: gedrückt oder nicht gedrückt.

Er verbindet einen digitalen Eingang (hier **D9**) mit **5V**. Lässt man ihn los, „schwebt“ der Eingang, und der Arduino erkennt das nicht sicher. Deshalb verbindet man den Eingang zusätzlich über einen **Pull-down-Widerstand** (10 kΩ) mit **GND**.



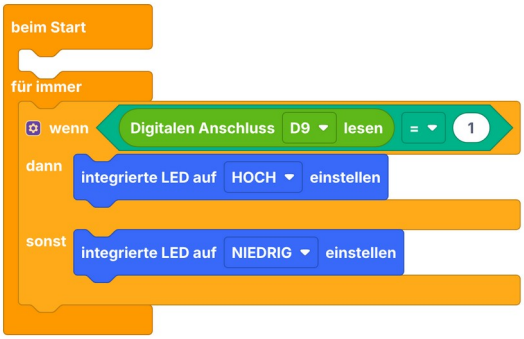
1) Anschluss mit Pull-down-Widerstand

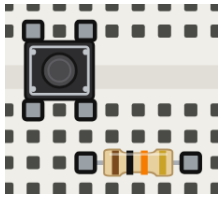
Von **5V** zu **GND** liegen Taster und Widerstand in Reihe, dazwischen der Anschluss zu **D9**. Der Taster hat **4 Beine** – stecke ihn über die Mittelrinne der Steckplatine.

2) Programmieren

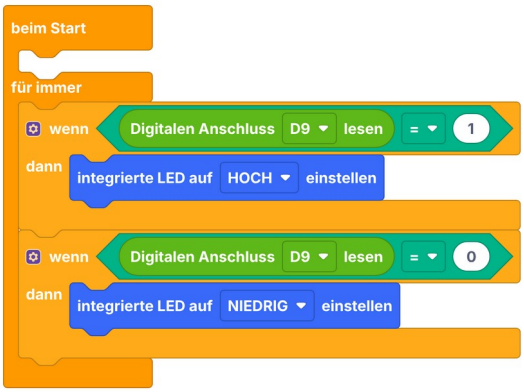
Die **eingebaute LED „L“** soll leuchten, solange der Taster gedrückt ist. In der Simulation drückst du den Taster mit der Maus. Teste die drei Varianten:

Variante 1

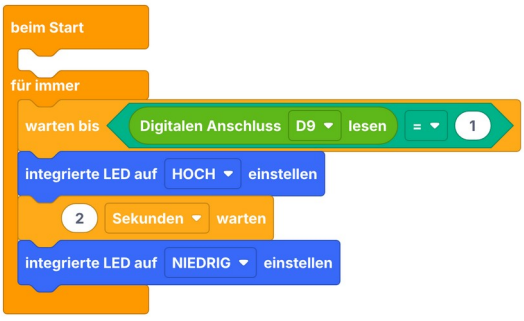




Variante 2



Variante 3



Aufgaben

a) Erkläre die Unterschiede zwischen den drei Varianten. Welche gefällt dir am besten?

b) Statt der eingebauten LED soll eine **externe LED an D3** leuchten.

c) Kehre die Funktion um: Die LED soll **ausgehen**, wenn der Taster gedrückt wird.

d) Der **Piezo-Summer an D8** soll bei Tastendruck einen Ton von sich geben.

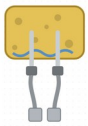
e) Auf dem **Display** soll „Taster gedrückt“ bzw. „Taster nicht gedrückt!“ stehen.

f) **Für Profis:** Der Arduino soll nur bei jedem **zweiten** Drücken „2x gedrückt“ anzeigen.

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

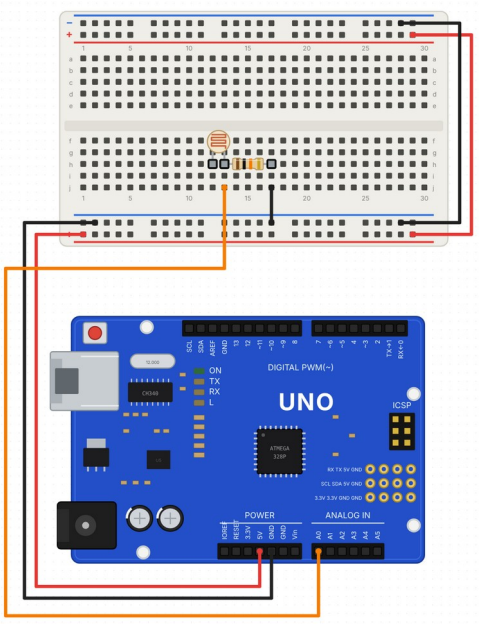
Analoge Sensoren

Analoge Sensoren liefern nicht nur 0 und 1, sondern auch **Zwischenwerte**. Je nach Temperatur, Licht oder Feuchtigkeit verändert sich ihr **Widerstand**:

Sensor		So reagiert er
	**LDR** (Fotowiderstand)	Je **heller** es wird, desto **kleiner** wird sein Widerstand.
	**NTC** (Heißleiter)	Je **wärmer** es wird, desto **kleiner** wird sein Widerstand.
	**Feuchtefühler** (zwei Drähte)	Je **feuchter** es zwischen den Drähten ist, desto **kleiner** wird der Widerstand.
	**Potentiometer** (Poti)	Beim Drehen ändert sich der Widerstand – je nach benutztem Anschluss im oder gegen den Uhrzeigersinn.

Der Spannungsteiler

Der Arduino kann keine Widerstände messen, sondern nur **Spannungen**. Deshalb schaltet man – wie beim Taster – einen **festen Widerstand** in Reihe. Dieser **Spannungsteiler** teilt die 5 V je nach Widerstand des Sensors auf. Die Spannung in der Mitte misst der analoge Eingang **A0**.



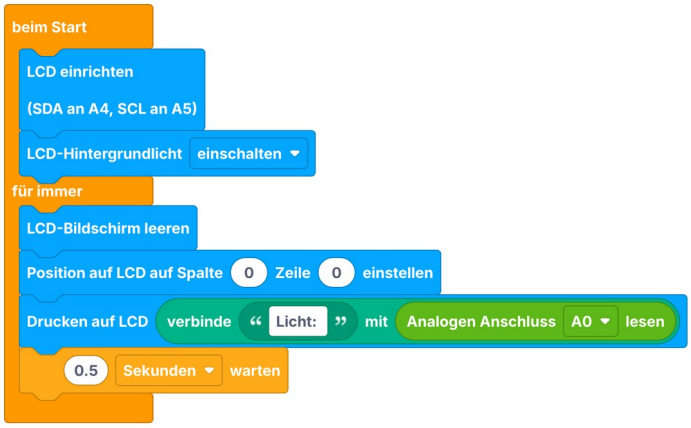
Merke

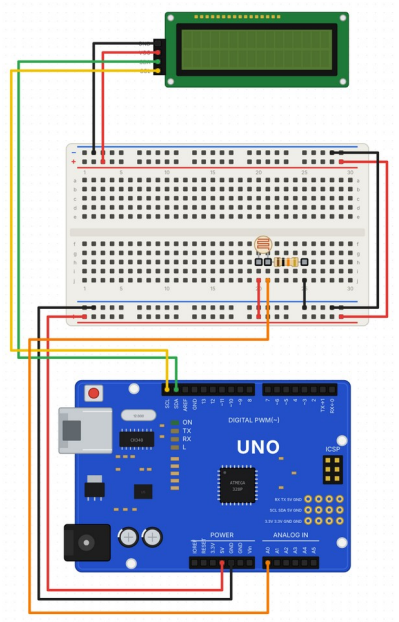
Reihenfolge **5V → Sensor → A0 → Widerstand → GND**: Je **höher** der Widerstand des Sensors, desto **kleiner** der Wert an **A0**. Bei umgekehrter Reihenfolge ist es genau andersherum.

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Am Beispiel LDR

1. Von **5V** zu **GND** liegen LDR und Widerstand in Reihe, dazwischen der Anschluss zu **A0**. Fester Widerstand: **10 kΩ** (Widerstand anklicken → Wert einstellen).
2. Die Werte des LDR sollen auf dem **LCD-Display** erscheinen. Schließe es wie auf der LCD-Seite an.
3. **Programmieren** – du brauchst diese Blöcke:

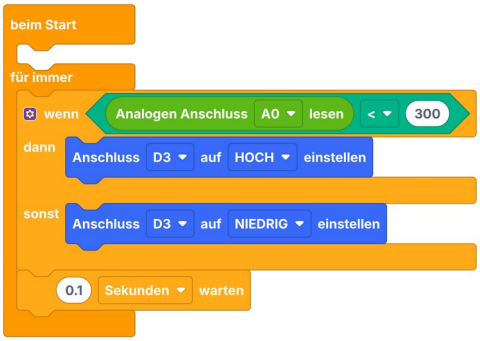




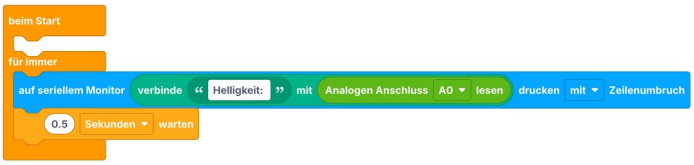
Simulation: Klicke während der Simulation auf den **LDR** und stelle mit dem Schieberegler die **Helligkeit** ein.

- Lies den Wert einmal im **hellen** und einmal im **abgedunkelten** Zustand ab (am echten LDR: mit der Hand abdecken).
- Lege dazwischen einen **Sollwert** fest, bei dem der Arduino reagieren soll. Den Sensorwert (**Istwert**) vergleichst du mit dem Sollwert – mit dem Vergleichsblock aus **„Operatoren“**.

Beispiel: Nachtlicht



Ohne Display: Werte im **seriellen Monitor** anzeigen (unten, Reiter „Serieller Monitor“). Mit **„Messwerte speichern“** öffnest du sie in Excel.



Aufgaben

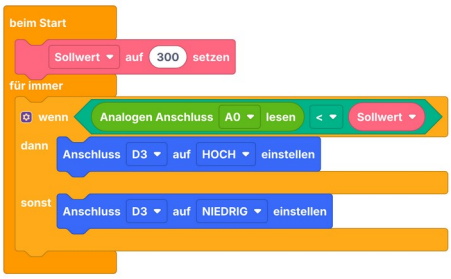
a) Bei **Dunkelheit** soll das Licht (externe LED) angehen.

b) Bei **Helligkeit** soll sich der Servo um 90° drehen (bei Dunkelheit wieder zurück).

c) Verwende einen **NTC**: Ermittle mit dem Display zuerst einen Schwellenwert (Raumtemperatur und Handwärme).

d) Bei **Wärme** soll der Lüfter angehen.

e) **Für Profis:** Lege gleich zu Beginn eine Variable **„Sollwert“** an. So änderst du den Schaltpunkt an einer einzigen Stelle (Beispiel rechts).



© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Projekte

Jetzt kennst du die wichtigsten Bauteile. Wende dein Wissen in **eigenen Projekten** an. Hier einige Ideen:

a) **Treppenhauslicht:** *Wird der Lichtschalter (Sensor = Taster) gedrückt, geht das Licht für 10 Sekunden an (Aktor = LED).*

b) **Heizungssteuerung:** *Eine Heizung (Aktor = rote LED) geht an, wenn es zu kalt wird (Sensor = NTC).*

c) **Abblendautomatik:** *Das Licht eines entgegenkommenden Autos (Taschenlampe → Sensor = LDR) senkt den eigenen Scheinwerfer um 10° (Aktor = Servo).*

d) **Bewässerung:** *Ist die Erde zu trocken (Sensor = 2 Drähte im Schwamm) und es dunkel (Sensor = LDR), springt die Pumpe an (Aktor = Lüfter).*

e) **Bahnübergang:** *Kommt ein Zug (Sensor = LDR oder Ultraschallsensor), senkt sich die Schranke (Aktor = Servo) und das Andreaskreuz blinkt (Aktor = LED).*

f) **Für Experten – Fußgängerampel:** *Wird der Taster gedrückt, wechselt die Autoampel auf Rot, die Fußgängerampel wird verzögert Grün (2 rote, 2 grüne und eine gelbe LED).*

g) … der eigenen Fantasie sind (fast) keine Grenzen gesetzt. Viel Spaß!

Hilfen im Codekäfer

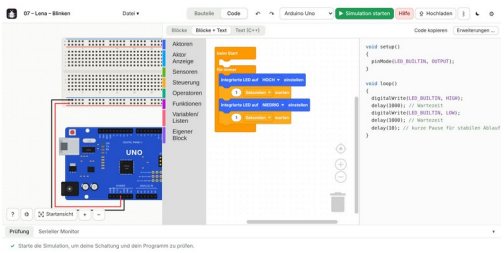
- **„Hilfe“** (oben): prüft Schaltung und Programm und gibt Tipps, wie es weitergeht.
- **„Datei › Aufgaben …“**: viele weitere Aufgaben – mit **„Aufgabe prüfen“** testet der Codekäfer deine Lösung.
- **„Datei › Beispiel öffnen …“**: fertige Beispiele (Ampel, Dimmer, Thermometer, Servo …) als Ausgangspunkt.
- **„Datei › Ältere Fassung wiederherstellen …“**: Etwas kaputt gemacht? Der Codekäfer hat ältere Stände gesichert.
- **„Datei › Schaltung als Bild speichern“** und **„Drucken …“**: für die Dokumentation deines Projekts.

Hilfreiche Links

1. Eine ausführlichere Anleitung (blockbasiert): https://www.nibis.de/uploads/nlq-jubitz-rahn/zuhause/steuern*regeln/Technik*Steuern-und-Regeln*Jg9-10*Arduino*Lernheft*Bl%C3%B6cke.pdf
2. Aktoren, Sensoren und Schaltungen gut erklärt: https://funduino.de/ *(dort mit Quellcode statt Blöcken – die Schaltungen sind aber gut erklärt).*

Für Weiterdenker und Weitermacher

1. An der Schule gibt es noch weitere Bauteile (RGB-LED, Ultraschallsensor, Bewegungsmelder, Relais, Motoren …). Viele davon findest du im Codekäfer – einige über **„Erweiterungen …“**.
2. Unter **„Code“ › „Blöcke + Text“** zeigt der Codekäfer dein Programm auch als **C++-Code** – ein guter Einstieg ins textbasierte Programmieren, z. B. später in der **Arduino IDE**.

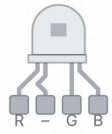


© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

ANHANG

Weitere Aktoren

RGB-LED



Eine RGB-LED enthält **drei LEDs** in den Farben **Rot, Grün und Blau**. Jede Farbe hat ihren eigenen Pluspol, der vierte und **längste** Anschluss ist der gemeinsame **Minuspol**.

Im Codekäfer ist die Reihenfolge **R – (–) – G – B**. Jede Farbe bekommt einen eigenen **Vorwiderstand** (220 Ω). Alle drei Farben kommen an **PWM-Pins** (~), z. B. **D9, D10, D11** – so lässt sich jede Farbe stufenlos mischen.



Aufgaben

a) Schließe die RGB-LED an und teste sie mit dem Beispielprogramm.

b) Probiere Farbmischungen aus (z. B. Rot + Grün = Gelb).

c) **Für Experten:** Programmiere einen sanften **Farbwechsel**.

Relais-Modul



Funktion: ein elektrisch gesteuerter **Schalter**. Bei **HOCH** zieht das Relais an (es klickt) und schaltet einen **zweiten, getrennten Stromkreis** – z. B. eine Lampe oder Pumpe mit eigener Batterie.

Hinweis: Dieser Stromkreis kommt an **COM** und **NO** (Schließer: verbunden, wenn das Relais angezogen ist) oder **NC** (Öffner). In der Schule nur **Kleinspannung** schalten – nie 230 V!

Relais-Modul	Arduino
IN	z. B. D7
DC+	5V
DC–	GND

Das Relais findest du über **„Erweiterungen …“ › „Motoren und Relais“**. **Nötiger Programmbefehl:** wie bei einer LED („Anschluss … auf HOCH/NIEDRIG einstellen“).



Aufgaben

a) Lass das Relais alle 2 Sekunden umschalten und beobachte die LED auf dem Modul.

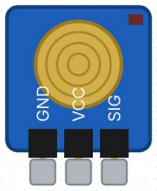
b) Schalte mit dem Relais eine Lampe mit eigener Batterie.

c) **Für Experten:** Eine Pumpe läuft, sobald der Feuchtigkeitsfühler „trocken“ meldet.

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Weitere digitale Sensoren

Sensortaste



Funktion: ähnlich wie ein Taster – bei Berührung liefert **SIG** den Wert **1** (HOCH), sonst **0** (NIEDRIG).

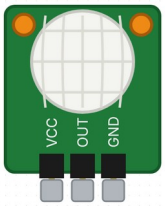
Hinweis: Der Pull-down-Widerstand ist schon eingebaut. In der Simulation „berührst“ du die Taste mit einem Mausklick.

Nötiger Programmbefehl:



Sensortaste	Arduino
GND	GND
VCC	5V
SIG	z. B. D2

Bewegungsmelder (PIR-Sensormodul)



Funktion: ähnlich wie ein Taster – bei **Bewegung** in der Umgebung liefert **OUT** den Wert **1** (HOCH), sonst **0** (NIEDRIG).

Hinweis: Unter der weißen Kuppel steht die Anschlussbelegung. Mit den beiden Drehreglern stellt man Empfindlichkeit und Haltezeit ein.

Nötiger Programmbefehl: wie bei der Sensortaste („Digitalen Anschluss … lesen“).

Bewegungsmelder	Arduino
GND	GND
VCC	5V
OUT	z. B. D2

Ultraschall-Abstandssensor



Funktion: misst mit Ultraschall den **Abstand** zu einem Gegenstand (2 bis 400 cm).

Hinweis: Er braucht **2 digitale Pins**. Der Sensor sendet einen kurzen Ton (Trig) und misst, wann das Echo zurückkommt (Echo). Daraus berechnet der Block den Abstand in cm.

Nötiger Programmbefehl:



Ultraschallsensor	Arduino
VCC	5V
Trig	z. B. D3
Echo	z. B. D2
GND	GND

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.

Weitere analoge Sensoren

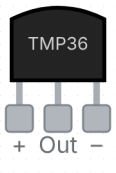
Temperatursensor TMP36

Nicht mit einem Transistor verwechseln – er sieht genauso aus!

Funktion: misst die Temperatur von **−40 bis +125 °C** (Genauigkeit ca. 2 °C).

Hinweis: Der Analogwert muss in °C umgerechnet werden. Das erledigt ein eigener Block aus **„Sensoren“**:





TMP36	Arduino
+	5V
Out	z. B. A0
–	GND

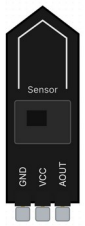
Kapazitiver Feuchtigkeitsfühler

Funktion: misst die Feuchtigkeit eines Materials (z. B. von Blumenerde).

Hinweis: Beim kapazitiven Sensor gilt: **trocken = großer Wert**, **feucht = kleiner Wert**. Probiere es in der Simulation mit dem Schieberegler aus!

Nötiger Programmbefehl:





Feuchtesensor	Arduino
GND	GND
VCC	5V
AOUT	z. B. A0

© Ulrich Aichler · codekaefer.de – kostenlos für Unterricht und Schule; Weitergabe erlaubt, Veränderung und Verkauf nicht.