

# ARDUINO KENNENLERNEN

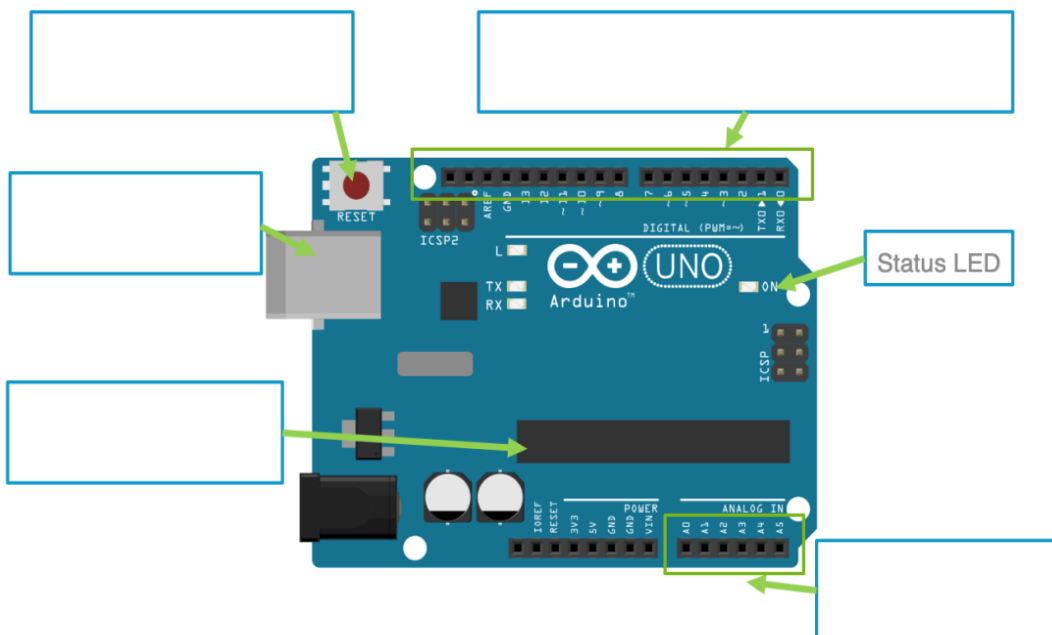
## WAS IST EIN ARDUINO?

Der Arduino ist eine Platine mit einem **Microcontroller** und sogenannten **Pins** (Anschlüssen), die mit elektronischen Bauteilen einfach verbunden werden können. Das können z.B. **Sensoren** wie ein Bewegungsmelder sein oder auch **Aktuatoren** wie eine LED.

Außerdem besitzt er einen USB-Anschluss. Über diesen kann er mit einem Computer verbunden werden. Mit einem speziellen Computerprogramm lässt sich der Arduino programmieren. Man kann ihm somit sagen, was er mit den angeschlossenen Bauteilen machen soll.

## AUFGABE 1

Schaue dir einen Arduino UNO an und beschrifte die folgende Abbildung.

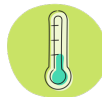


## WOFÜR SIND PINS AM ARDUINO?

In die Pins lassen sich einfach die Kabel z.B. eines Bewegungssensors und einer LED stecken. Ein Programm wird geschrieben, das den Pin mit dem Bewegungssensor überwacht und den Pin mit der LED an- und ausschalten kann. Dabei soll der Pin mit der LED dann angeschaltet werden, wenn der Pin des Bewegungssensors eine Bewegung meldet. Das Programm wird per USB-Kabel auf den Arduino übertragen. Der Arduino führt das Programm aus.

### Arduino

Ein Arduino ist ein einfacher und bekannter Microcontroller, an den Sensoren und Aktuatoren angeschlossen werden können.



**Tipp:** Folgende Begriffe sind möglich: USB-Anschluss, Reset-Taste, Microcontroller, digitale Pins, analoge Pins

### Pins

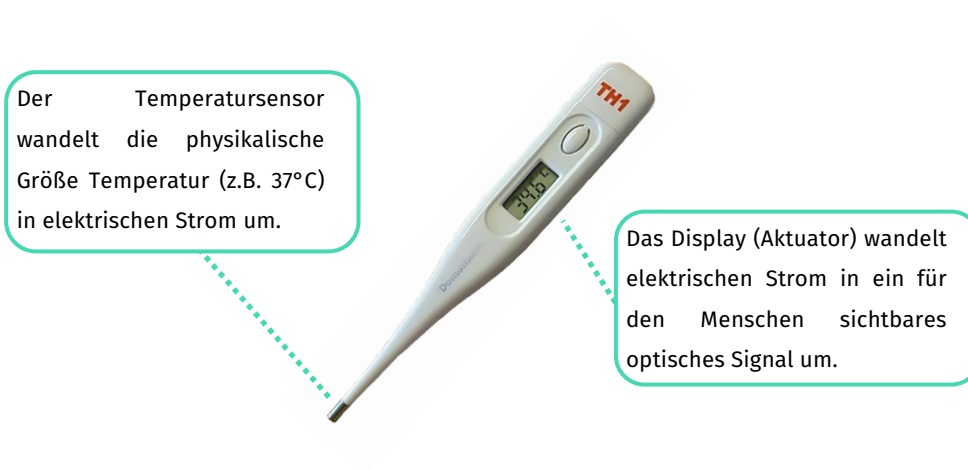
Über seine eingebauten Pins kommuniziert der Arduino z.B. mit LEDs oder Sensoren.

## WAS SIND SENSOREN UND AKTUATOREN?

Der Arduino kann nur mit **elektrischen Größen** arbeiten. Alles, was er steuern oder auslesen soll, muss daher elektrisch funktionieren. Solche Bauteile, mit denen er etwas auslesen oder steuern kann, nennt man **Sensoren** und **Aktuatoren**.

Ein Sensor (von lateinisch sentire, dt. fühlen/empfinden) wandelt eine physikalische Größe in eine elektrische Größe um. Ein Aktuator (aus dem Englischen actuator, dt. Auslöser/Handelnder) wandelt eine elektrische Größe in eine physikalische Größe um.

## BEISPIEL ELEKTRONISCHES THERMOMETER



## AUFGABE 2

Welche Sensoren oder Aktuatoren vermutest du in den folgenden Geräten?

Gerät	Sensor	Aktuator
elektronisches Fieberthermometer	Temperatursensor	Display
automatisches Fahrradlicht		
Einparkhilfe am Auto		

**Sensor**  
Ein Sensor wandelt eine physikalische Größe in eine elektrische Größe um.

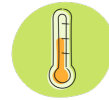
**Aktuator**  
Ein Aktuator wandelt eine elektrische Größe in eine physikalische Größe um.



### AUFGABE 3

Beschreibe für die folgenden Sensoren oder Aktuatoren, welche Größen umgewandelt werden.

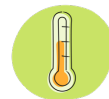
Sensor oder Aktuator	Größe → Größe
Temperatursensor	Wärme → elektrische Größe
Elektromotor	
Lichtsensord	
LED	



### AUFGABE 4

Welche weiteren Sensoren und Aktuatoren kennst du? Welche Größen werden umgewandelt und in welchen Geräten kommen Sie vor?

Sensor oder Aktuator	Größe → Größe	Gerät



**Tipp:** Überlege dir, welche Geräte „automatisch“ funktionieren. Du kannst auch „in Gedanken“ einmal durch dein Zimmer und dein Wohnzimmer laufen, und dir dort Geräte überlegen, ob dir dort Geräte mit Sensoren oder Aktuatoren auffallen.

# MIT DEM ARDUINO KOMMUNIZIEREN

## WAS IST EIN MICROCONTROLLER?

Ein **Microcontroller** besteht aus einem **Prozessor** sowie aus **Ein- und Ausgängen**. An seinen Eingängen kann er Informationen aus **Sensoren** auslesen. An seinen Ausgängen kann er **Aktuatoren** ansteuern. Die Ein- und Ausgänge des Arduinos nennt man auch **Pins**.

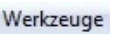



Ein Prozessor ist eine programmierbare elektronische Schaltung. Sie ist mit den Pins verbunden und entscheidet darüber, was mit ihnen passiert. Programmieren lässt sich der Arduino per USB über ein Computerprogramm, das auch mit der englischen Abkürzung IDE (*integrated development environment*) bezeichnet wird.

### Microcontroller

Ein Microcontroller besteht aus einem Prozessor sowie Ein- und Ausgängen mit denen Sensoren und Aktuatoren verbunden werden.

## AUFGABE 5

Öffne die **Arduino-IDE** und finde die folgenden Komponenten. Kreuze an, ob du sie oben im **Menü** oder unten im Programmcode (**Sketch**) gefunden hast.

Bild	Name	Beschreibung	Im Menü?	Im Sketch?
	Werkzeuge	Unter Werkzeuge wird eingestellt, mit welchem Arduino kommunizieren werden soll.		
	Setup	Im Setup wird u.a. festgelegt, welche Pins man benutzen möchte.		
	Loop	In die Loop kommen die Befehle, die der Arduino immer wieder ausführen soll.		
	Hochladen	Mit dem Hochlade-Button lädt man die geschriebenen Befehle, den Sketch, auf den Arduino hoch.		

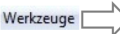







### Stromversorgung

Wenn der Arduino mit Strom versorgt wird (z.B. per USB-Kabel), leuchtet die Power-LED auf dem Board.

## AUFGABE 6

Verbinde per USB-Kabel einen Arduino UNO mit deinem Computer. Stelle ein, mit welchem Arduino du kommunizieren möchtest. Gehe dabei wie folgt vor:

- Die Bauart des Arduinos auswählen.   
- Den „Port“ deines Computers auswählen, über den du die Daten mit dem Arduino austauschen kannst:   



**Hinweis:** Der Port COM29 ist nur ein Beispiel. Die Zahl hinter COM kann bei dir anders sein. Wenn nur ein Arduino angeschlossen ist, sollte aber auch nur ein COM-Port angezeigt werden.

# ARDUINO PROGRAMMIEREN

## WAS IST EIN SKETCH?

Das Wort Programm ist nicht ganz eindeutig. Wir möchten zwischen den verschiedenen „Programmen“ **IDE** und **Sketch** unterscheiden. Die **IDE** ist das Programm, das auf deinem Computer läuft und über das du deinen **Sketch** schreiben kannst. Der **Sketch** ist das selbstgeschriebene Programm, das du in die **IDE** eintippst.

### Sketch

Mit Sketch wird das geschriebene Programm mit den Befehlen bezeichnet, die der Arduino ausführen soll.

## AUFGABE 7

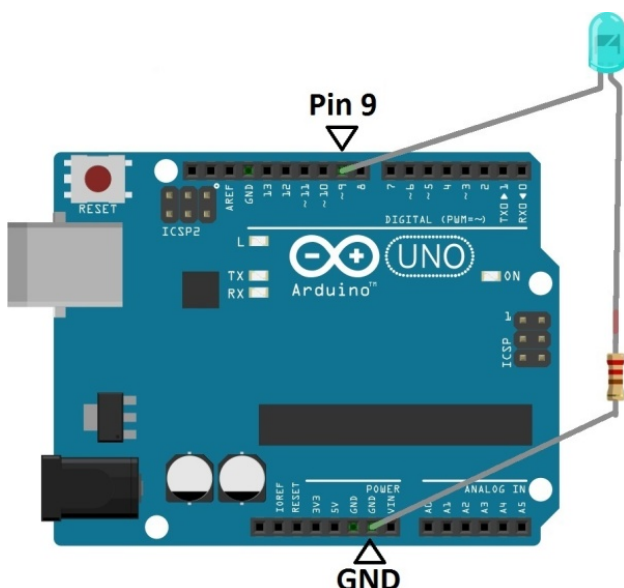
Tippe den folgenden Sketch in die IDE ein. Wichtig: Achte auf die spezielle Groß- und Kleinschreibung sowie auf die Zeichensetzung.

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

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

## AUFGABE 8


Stecke die LED mit Widerstand wie folgt in den Arduino: Der Draht an der LED kommt in Pin 9, der Draht am Widerstand in GND.



### LED anschließen

Die LED muss wie in einem normalen Stromkreis an Plus und Minus angeschlossen werden. Plus kann an die digitalen Pins per Programmbefehl geschaltet werden. Minus ist immer an den Pins mit GND (engl.: Ground).

## AUFGABE 9

Lade den Sketch auf den Arduino hoch. Klicke dafür auf den Hochlade-Button . Du wirst vorher noch aufgefordert, den Sketch zu speichern. Tu dies in einem Ordner, den du schnell wiederfindest. Beobachte danach, was mit der eingesteckten LED auf dem Arduino passiert.

## AUFGABE 10

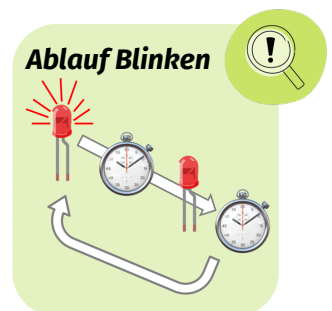
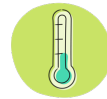
Überlege dir für deinen Sketch einen passenden Namen, der beschreibt, was der Arduino beim Ausführen tut.

# SKETCH VERSTEHEN & KOMMENTIEREN

## AUFGABE 11

Schaue dir die Abbildung rechts genau an. Überlege dir, was die einzelnen Befehle in der Loop bedeuten. Schreibe deine Vermutung daneben.

Befehl	Bedeutung
<code>void loop() {...}</code>	
<code>digitalWrite(9, HIGH);</code>	
<code>delay(1000);</code>	
<code>digitalWrite(9, LOW);</code>	
<code>delay(1000);</code>	



## AUFGABE 12

Tausche deine Gedanken mit deinem Sitznachbarn aus und überlegt euch zusammen die Antworten auf die folgenden Fragen.

Frage	Antwort
Warum taucht die Zahl 9 mehrfach auf?	
Was bedeuten HIGH und LOW?	
Wofür stehen geschweifte Klammern { } ?	
Was bedeutet loop?	
Was bedeutet delay?	
Warum blinkt die LED?	

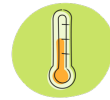
## KOMMENTIEREN

Ein Sketch kann sehr lang werden. Damit man den Überblick behält, ist es sinnvoll, ihn zu kommentieren. Ein Kommentar wird mit zwei Schrägstrichen // eingeleitet. Alles, was danach in der Zeile geschrieben wird, ignoriert der Arduino.

## AUFGABE 13

Kommentiere deinen Sketch in der Arduino-IDE. Nimm dafür deine Formulierungen aus den Aufgaben 11 und 12 oder formuliere diese gegebenenfalls neu. Du kannst hiermit anfangen:

`pinMode (6, OUTPUT) ; // Pin 9 als Ausgang definieren`



### Kommentare



Um zwei Schrägstriche zum Einleiten eines Kommentars zu erzeugen, drücke die Tasten **SHIFT** und **7** gleichzeitig. Alles, was hinter den beiden Schrägstrichen // steht, wird vom Arduino ignoriert.



# LISTE VON BEFEHLEN

Hier kannst du **Befehle** eintragen und aufschreiben, was sie bedeuten:

Bild	Name
pinMode (x, OUTPUT);	Pin x ist ein OUTPUT = Ausgang! An Ausgängen können Aktuatoren gesteuert werden.
pinMode (x, INPUT);	Pin x ist ein INPUT = Eingang! An Eingängen können Sensoren ausgelesen werden.
digitalWrite (x, HIGH);	Schalte Pin x high = an!
digitalWrite (x, LOW);	Schalte Pin x low = aus!
delay (1000);	Warte 1000 ms! 1000 ms entsprechen 1 Sekunde. 500 ms entsprechen einer halben Sekunde.
//	Kommentar! Alles, was hinter zwei Schrägstrichen in derselben Zeile steht, ignoriert der Arduino.

## Befehle

Beim Programmieren lernt man im Grunde eine neue Sprache mit Vokabeln und Syntax (d.h. Aufbau). Am besten lernt man neue Sprachen durch „sprechen“, was beim Programmieren bedeutet, Sketches zu schreiben.



# PROGRAMMIEREN LERNEN

Nun wollen wir Programmieren lernen, indem wir deinen **Sketch** aus der vorherigen Aufgabe anpassen und erweitern.

Lade den Sketch wieder in die Arduino IDE und schließe den Arduino per USB-Kabel an. Kontrolliere, ob Board und Port noch eingestellt sind, sonst kannst du den Sketch nicht auf den Arduino hochladen.

## AUFGABE 14

Ändere deinen Sketch so, dass...

3. die LED schneller blinkt.
4. die LED langsamer blinkt.
5. die LED beim Blinken doppelt so lange an ist als die LED aus ist.

## AUFGABE 15

Schreibe und erkläre, welchen Programmbefehl du für die Bearbeitung von Aufgabe 14 jeweils ändern musstest.

a) LED schneller blinken lassen

b) LED langsamer blinken lassen

c) LED doppelt so lange an lassen als aus

### Kann etwas

### kaputt gehen?

Durch Programmierung kann ein Arduino nicht kaputt gehen! Du kannst alles ausprobieren. Sollte nichts mehr klappen, lade den Grundsketch neu.



## ZEITLICHES AUFLÖSUNGSVERMÖGEN

Du hast bestimmt schon mal gehört, dass ein Film aus vielen einzelnen Bildern besteht, die sehr schnell hintereinander abgespielt werden. Würden sie langsamer abgespielt, könntest du die einzelnen Bilder erkennen und es würde „ruckeln“.

Dass ein Film flüssig wirkt, liegt am zeitlichen Auflösungsvermögen des menschlichen Auges. Wie viele Lichtreize pro Sekunde dein Auge benötigt, damit etwas flüssig wirkt, kannst du mit dem Arduino ermitteln.

### AUFGABE 16

Lasse die LED immer schneller blinken. Schreibe dazu schrittweise kleinere Zahlen in beide Delays und lade den Sketch jeweils neu auf den Arduino hoch.

Ab einer bestimmten Zeit für das Delay siehst du kein Blinken oder Flackern der LED mehr. Bestimme diese Zeit schrittweise und schreibe die ermittelte Zeit hier auf. Vergleiche auch mit deinem Nachbarn.

Das Blinken der LED ist nicht mehr zu erkennen ab einem Wert für das Delay unter...

### AUFGABE 17

Der Wert für das Delay, ab dem das Blinken nicht mehr zu erkennen ist, haben wir nun bestimmt. Berechne daraus nun die Anzahl der Mindest-Lichtreize pro Sekunde. Teile dazu 1000 ms durch deine doppelte Zeit des Delays (in ms)

$$\text{Mindest-Lichtreize pro Sekunde} = \frac{1000\text{ms}}{2 \cdot \text{delay}} = \frac{1000\text{ms}}{\quad} = \frac{1}{s}$$

#### Bildwiederholungsrate

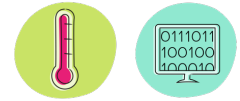
Fernsehbilder bestehen aus Einzelbildern. Es werden üblicherweise mindestens 30 oder gar 60 Bilder pro Sekunde angezeigt.



## AUFGABE 18

Stecke eine zweite LED in den Arduino und bringe Sie ebenfalls zum Blinken.  
Beantworte anschließend die folgenden Fragen dazu.

Frage	Antwort
In welche Pins muss die LED eingesteckt werden?	
Was muss dem Arduino im Setup noch über die LED mitgeteilt werden?	
Was muss in der Loop ergänzt werden?	



**Tipp:** Analysiere zuerst deinen Sketch und mache dir noch mal klar, wie eine LED zum Blinken gebracht wird. Überlege dir, woher der Arduino weiß, dass eine LED an einem Pin angeschlossen ist und wie man dem Arduino sagt, dass er eine LED an einem bestimmten Pin an- und ausschalten soll.

## AUFGABE 19

Ändere das Programm so, dass...

- die LEDs gleichzeitig blinken.
- die LEDs wechselseitig blinken.
- eine LED 2x blinkt und die andere danach einmal.

