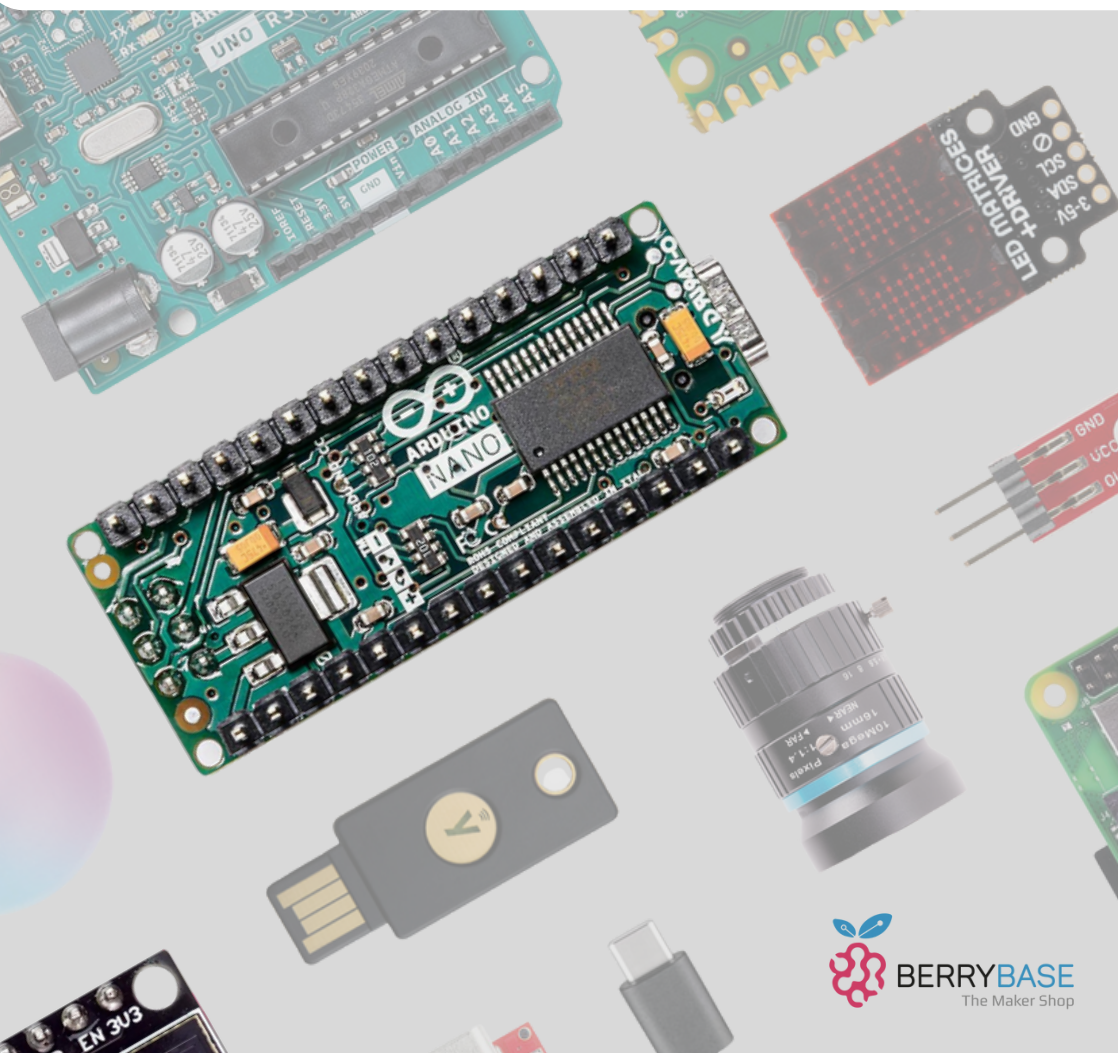
GET YOUR STUFF RUNNING

THE JUMPSTART GUIDE FOR **ARDUINO NANO**



BERRYBASE
The Maker Shop

Seite

Willkommen

Hallo und herzlich willkommen zu diesem Guide über den Arduino Nano.

Wir bei der BerryBase freuen uns, dich bei deinem Einstieg in die Welt der Mikrocontroller begleiten zu dürfen. Die BerryBase ist ein Maker Shop für jeden, mit einer großen Auswahl an Mikrocontrollern und deren Zubehör oder Ressourcen wie diesem Guide, um dir den idealen Start in dein Technik-Projekt zu ermöglichen.

In diesem Guide sollen Dir die Grundlagen mit der Arbeit am Arduino Nano näher gebracht werden. Also keine Scheu, es ist kein weiteres Wissen vorausgesetzt, sondern der Start beginnt bei null. Von Grundsätzlichen Informationen über konkretere Schaltpläne, bis hin zu Tipps und Tricks wie die Reise aussehen soll, erfährst du alles hier. Auch fängst du hier direkt mit deinem ersten Projekt an, aber keine Sorge es ist anfängerfreundlich. Außerdem heißt es nicht umsonst, “Probieren über Studieren”.

Daher ist es großartig, dass du dich für dieses Projekt entschieden hast!
Ich bin Timm, und bastel leidenschaftlich an Arduino Projekten seit einiger Zeit. Lass uns loslegen!

Ziel des Guides:

Nach Abschluss dieses Guides wirst du über das grundlegende Wissen und die Fähigkeiten verfügen, um den Arduino Nano erfolgreich zu programmieren und in deinen Projekten anzuwenden. Du wirst der Lage sein, eigenständige Schaltungen zu erstellen, verschiedene Sensoren und Aktoren anzusteuern und eigenen kreative Ideen mit dem Arduino Nano in die Realität umzusetzen."

Einführung in Mikrocontroller und die Vorzüge des Arduino Nano

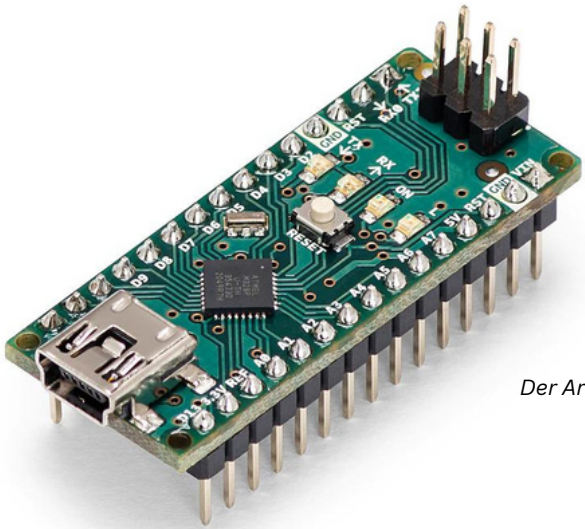
Mikrocontroller sind wie kleine Computer, die alle in einem einzigen integrierten Schaltkreis verpackt sind. Du hast sie wahrscheinlich in vielen elektronischen Geräten um dich herum gesehen, ohne es überhaupt zu merken. Sie sind dafür ausgelegt, spezifische Aufgaben auszuführen, und können Sensoren lesen und Aktoren steuern, um mit der physischen Welt zu interagieren. Ihr großer Vorteil ist, dass sie in Echtzeit arbeiten, wenig Strom verbrauchen und ziemlich kostengünstig sind.

Jetzt fragst du dich vielleicht, welcher Mikrocontroller für dich am besten geeignet ist. Hier kommt der Arduino Nano ins Spiel. Dieser kleine, aber mächtige Mikrocontroller ist besonders bei Bastlern und Experten beliebt. Warum? Er ist einfach zu verwenden, wird von einer großen Community unterstützt und es gibt unzählige Erweiterungen und Bibliotheken für ihn. Als Anfänger wirst du die Einfachheit des Arduino Nano schätzen, aber er bietet auch genug Tiefe für anspruchsvollere Projekte. Also, wenn du nach einem zuverlässigen und flexiblen Mikrocontroller suchst, solltest du den Arduino Nano in Betracht ziehen.

Der Arduino Nano

Der Arduino Nano ist ein kleiner Mikrocontroller, der auf dem Microchip ATmega328P basiert. Der Nano ist mit einer Reihe an Sensoren und Modulen kompatibel, die über verschiedene Pins des Arduino Nano angeschlossen werden können. Somit fungiert dieser als “Gehirn”, an welches z.B. “Augen” (Kameras, Bewegungssensoren, usw.) oder ein “Mund” (Lautsprecher, LEDS, usw.) angeschlossen werden können.

Der Arduino ist ein weit verbreiteter Mikrocontroller, daher gibt es unzählige Komponenten, die kompatibel mit dem Arduino sind. Über die einzelnen Pins können neben speziell für den Arduino Nano entwickelte Komponenten, auch viele weitere Komponenten von Dritten angeschlossen werden (Breadboards, LEDS, usw.) angeschlossen werden. Auch bei der Programmierung steht die Arduino IDE auf sämtlichen Betriebssystemen (Windows/Linux/MacOS) zur Verfügung.



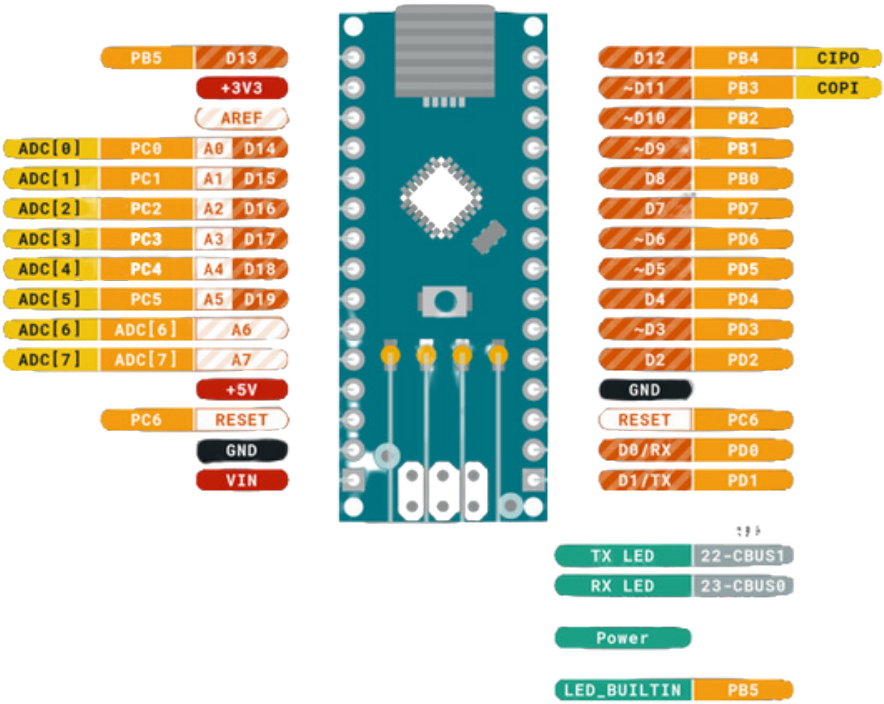
Der Arduino Nano

Zudem gibt es den Arduino in verschiedenen Modellen. Jedes dieser bringt erweiterte oder unterschiedliche Funktionen zum Basismodell. So umfasst das Basismodell mit
dem “Arduino Nano” und “Arduino Nano Every” die Standardfunktionen, wie Analoge Pins, Digitale Pins, USB-Anschluss + ISP. Der “Arduino Nano RP2040” oder der “Arduino Nano BLE” haben zusätzliche Funktionen, wie WLAN und Bluetooth, verbaut. Diese Modelle eignen sich, dann besonders für den Einsatz im Smart-Home-Bereich aufgrund von eingebautem Bluetooth und WLAN.
Jedes Modell hat seine eigenen Vor- und Nachteile. Aus diesem Grund kann die Wahl des richtigen Modells für den geplanten Einsatz entscheidend sein.

Der Arduino Nano ist besonders für kleinere Projekte geeignet, wo keine große Leistung gebraucht wird. Für Anfänger also genau das Richtige, weil er auch einfach zu benutzen ist. Die zwei größten Argumente für den Nano sind sein Preis und die Kompaktheit. Für einen günstigen Preis verfügt er über genug Funktionen, um in allerhand Projekten eingesetzt werden zu können. Er nimmt auch kaum Platz weg und ist ideal, besonders wenn nicht viel Platz zur Verfügung steht. Auch die Arduino Nano Pinbelegung bietet alle nötigen Arduino Nano Anschlüsse für dein Projekt

Arduino Nano Pinout

Die Arduino Nano Maße beträgt 18 x 45mm mit 4 Löchern zum Befestigen in jeder Ecke. Insgesamt verfügt er über 32 Pins.



- **Power:** Strom wird durch den Mini-B USB Stecker oder über Pin 30 (rot VIN) oder Pin 27 (5V; rot) bereitgestellt.
- **Digitale Pins:** Die 14 digitalen Pins können jeweils für Input oder Output benutzt werden, über die verschiedenste Komponente (LED, Schalter usw.) angeschlossen werden können. (D1-14)
- **Analoge Pins:** Die 7 Analogen Pins (A1-7) sind ähnlich wie die digitalen Anschlüsse für das Verbinden mit Komponenten, jedoch für analoge Kommunikation der beiden.
- **Masse Pins:** Die schwarzen Pins (GND) sind Masse Pins für das Anschließen verschiedener Elektronik.
- **Reset:** In der Mitte des Arduino Nano befindet sich ein Reset Knopf, dieser setzt die Schaltung zurück.
- **PWM-Pins:** Die Arduino Nano PMW-Pins sind mit (~) Symbolen gekennzeichnet. Diese ermöglichen zum Beispiel eine analoge Schaltung über digitale Pins. Weitere Informationen zu PWM findest du **hier**.
- **Interrupt Pins:** Die Arduino Nano Interrupt Pins 2 und 3 können über den Quellcode für Interrupt Ereignisse verwendet werden. So wird bei dem Ereignis eine definierte Funktion ausgeführt bevor das Hauptprogramm (main Funktion) weiterläuft.
- **USB-Anschluss:** Die primäre Arduino Nano Stromversorgung ist der USB Anschluss, während dieser gleichzeitig den Nano mit dem Rechner zum Programmieren verbindet. Über diesen kann der Nano mit dem PC verbunden werden und über die Arduino Software programmiert werden.
- Die beiden Pins A4 und A5 ermöglichen außerdem die Verbindung eines Arduino Nano über I2C.
- Für tiefgründige Schaltungsinformationen siehe **hier** oder **hier**.

Erste Vorbereitungen

Bevor du nun anfangen kannst, mit dem Arduino zu arbeiten, müssen wir erstmal ein paar Vorbereitungen treffen. Zuerst sind folgende Materialien notwendig:

1. PC/Laptop (Windows/Linux/MacOS)
2. USB-Kabel für den Arduino Nano

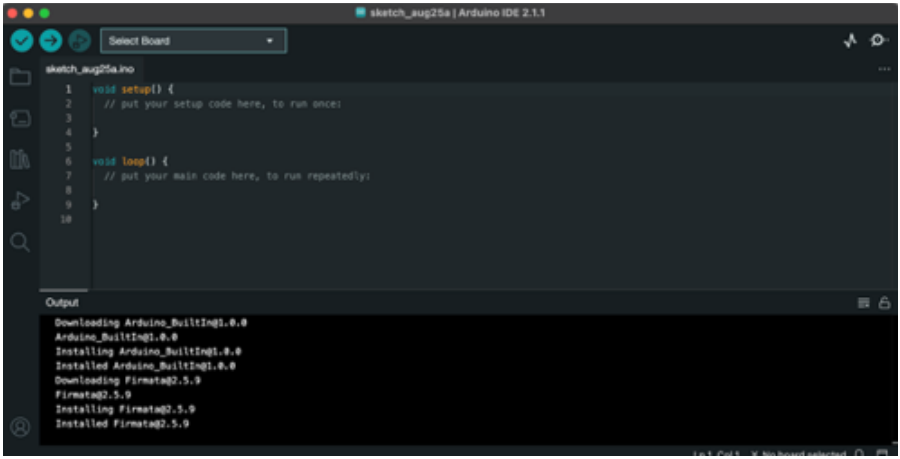
Als nächstes folgt ein weiterer wichtiger Schritt. Damit der Nano weiß, was er machen soll, müssen wir ihm dies mitteilen. Das passiert über die Arduino IDE, welche speziell für das Programmieren vom Arduino entwickelt wurde und somit alle notwendigen Funktionen bereitstellt. **Hier** findest du die Software. Nach dem Aufrufen vom Link, sollte auf der Seite eine Kachel wie diese erscheinen. Die Software heißt “Arduino IDE” und hier in unserem Beispiel ist 2.1.1 die aktuelle Version. Sollte deine Version eine andere, also eine aktuellere sein, kannst du trotzdem ohne Probleme dem Guide folgen. Wähle nun dein Betriebssystem aus, um den Download zu starten.

Folge nun den Anweisungen vom Installer, bis du die Anwendung vor dir siehst.

Mit dieser wirst du nun in Zukunft deinen Arduino Nano Projekte programmieren können. Die Anwendung umfasst auch weitere Funktionen, wie einen Debugger oder ein Terminal, die im Laufe deiner Projekte praktisch werden können.

Erstes SetUp

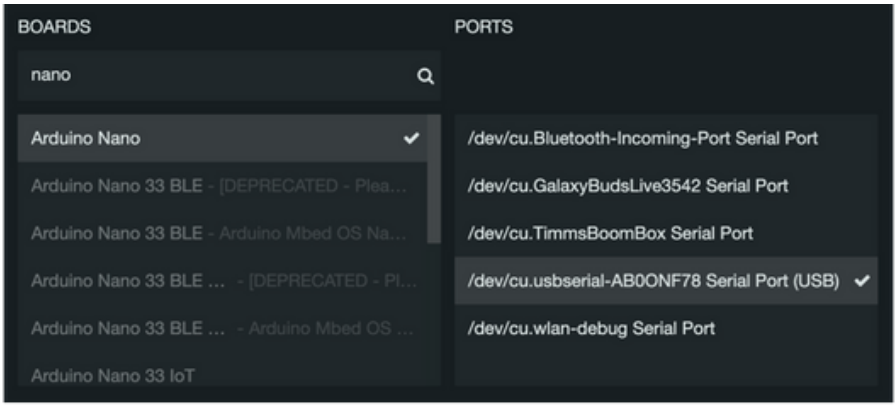
Vor dir solltest du jetzt die Arduino IDE sehen. Über diese Oberfläche wirst du in Zukunft alle möglichen deiner Projekte programmieren und testen können.



Als erstes verbinde nun den Arduino Nano mit deinem Rechner über das USB-Kabel. Jetzt fehlt nur noch eine kleine Konfiguration in der Software, damit wir loslegen können mit der ersten Inbetriebnahme.

Wähle daher oben “Select Boards” aus, dort wird das Modell festgelegt, mit welchem wir arbeiten. Und auf welchem Port die IDE den Arduino Nano erreichen kann.

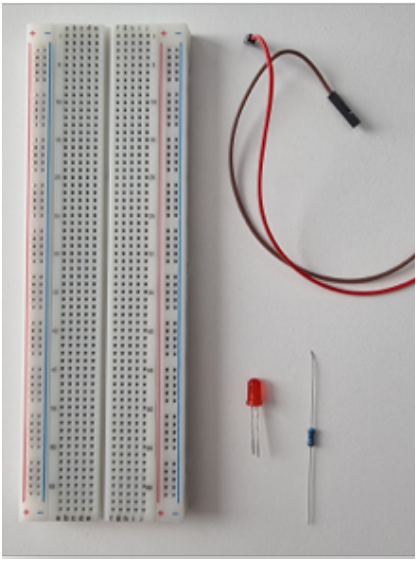
Nun sollte sich ein ähnliches Fenster wie das folgende öffnen. Wähle hier auf der linken Seite das Modell “Arduino Nano” aus und auf der rechten deinen Arduino Nano den du mit dem PC verbunden hast. Solltest du dort nichts dem Arduino sofort zuordnen können, dann suche einen Port, der namentlich etwas mit “usb” beinhaltet.



Bestätige deine Auswahl nun mit “ok” und das Fenster sollte sich schließen. Oben sollte jetzt statt “Select Board” “Arduino Nano” stehen. Nun zurück zur IDE und dem Programmieren.

In der Mitte der Arduino IDE solltest du den Quellcode mit den zwei “Funktionen” sehen. Alles, was nun in den Klammern {} bei “SetUp” steht wird einmal ausgeführt, wenn das Programm gestartet wird. Die Funktion “loop” verhält sich schleifenartig. Das heißt der Quellcode in den Klammern {} dieser Funktion wird wie in einer Schleife ausgeführt, bis das Programm endet.

Solltest du deinen Quellcode dann fertig geschrieben haben, kannst du durch das Klicken des “blauen Hackens” überprüfen, ob du Fehler gemacht hast. Sollte aber noch ein Fehler im Quellcode sein, taucht nun unten im Bildschirm eine Fehlermeldung in rot auf. Wenn dies nicht der Fall ist, dann kannst du nun deinen Quellcode auf den Arduino laden. Drücke hierzu einfach auf den Knopf mit dem Pfeil nach rechts, der Quellcode wird nun auf den Arduino hochgeladen und ausgeführt.



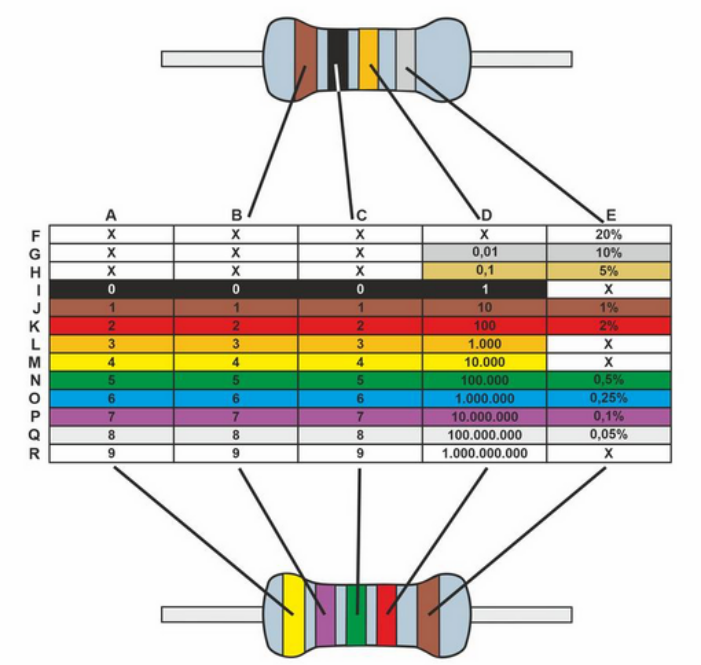
Erste Inbetriebnahme

Nun geht es darum zu testen, ob wir alles richtig eingestellt haben. Hierzu wollen wir eine LED zum Blinken bringen. So können wir gleich überprüfen, ob alles richtig läuft, während du dich gleich mit dem Ablauf vertraut machst. Viele der folgenden Schritte wirst du immer brauchen, wenn du Mikrocontroller aller Art programmierst.

Hierzu benötigst du: 2x Jumper-Kabel, 1x LED und 1x Schichtwiderstand.

Widerstände

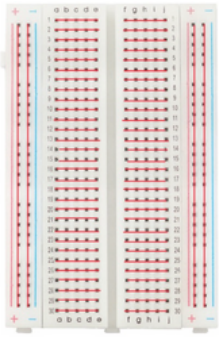
Der Widerstand von Widerständen wird in Ohm (Ω) angegeben. Dieser muss dann entsprechend der LED gewählt werden. Bei den Starter Kits sind meistens kleine Anleitungen beigelegt, welche dir die richtigen zeigen. Meistens sollte dieser zwischen 200Ω-500Ω liegen. Wie viel Ohm (Ω) ein Widerstand hat, kannst du mithilfe der Grafik und anhand der farbigen Striche erkennen. Benutze den oberen Teil bei vier Strichen und den unteren bei 5 Strichen. Die ersten 3 bzw. 4 sind dafür gedacht, um den Widerstand zu bestimmen.



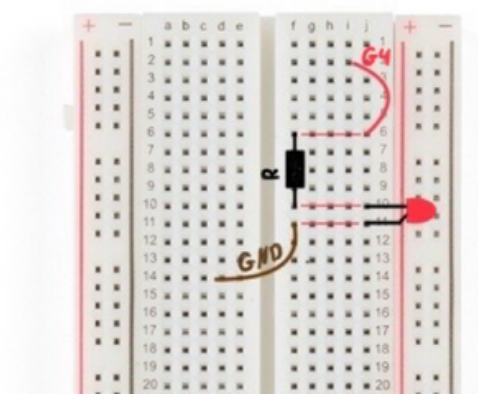
	A	B	C	D	E
F	X	X	X	X	20%
G	X	X	X	0,01	10%
H	X	X	X	0,1	5%
I	0	0	0	1	X
J	1	1	1	10	1%
K	2	2	2	100	2%
L	3	3	3	1.000	X
M	4	4	4	10.000	X
N	5	5	5	100.000	0,5%
O	6	6	6	1.000.000	0,25%
P	7	7	7	10.000.000	0,1%
Q	8	8	8	100.000.000	0,05%
R	9	9	9	1.000.000.000	X

Nehme die Werte von (A bei 5 strichen), B, C und multipliziere den Wert mit D.

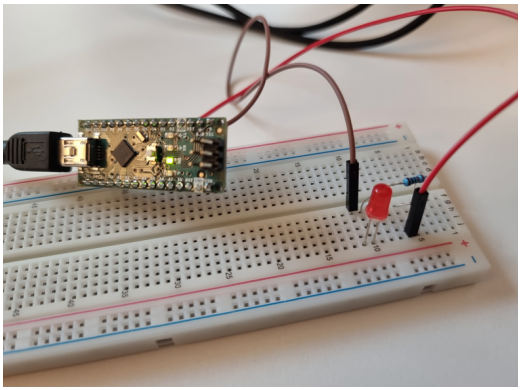
Die einzelnen Löcher des Breadboards sind nach dem folgenden Muster miteinander verbunden (rote Markierungen in Grafik). Die Einkerbung in der Mitte trennt a-e und f-j voneinander.
Die Streifen sind jeweils horizontal miteinander verbunden. Während an den beiden äußeren Seiten Plus (+) und Minus (-) der Versorgungsspannung vertikal miteinander verbunden sind.



Die LED hat eine Kathode und eine Anode, was entscheidend für die Beschaltung ist. Beim falschen Anschließen kann die LED beschädigt werden. Die kürzere Seite der beiden LED-Kontakte ist die Kathode und muss mit dem Ground (GND) verbunden werden. Der Widerstand ist hierbei vor die LED geschaltet, damit diese nicht, aufgrund zu hoher Spannung, durchbrennt.



Die LED hat eine Kathode und Anode, was entscheidend für die Schaltung ist. Beim falschen Anschließen kann diese beschädigt werden. Die kürzere Seite der beiden LED Kontakte ist die Kathode und muss mit dem Ground (GND) verbunden werden. Der Widerstand ist hierbei vor die LED geschaltet, damit diese nicht durchbrennt aufgrund zu hoher Spannung.



Verbinde nun das braune Kabel mit einem Ground (GND) Pin des Arduino Nano und das rote Kabel mit dem D4 Pin. (Siehe hierfür die Abbildung des Pinouts)

Als Ergebnis sollte die Schaltung ähnlich wie im Bild (Aufbau) aussehen. Wo genau du die Schaltung auf dem Breadboard platzierst ist nicht wichtig. Beachte aber, dass die Kabel und LED richtig verbunden sind. (siehe Breadboard Verkabelung)

Damit die LED nun leuchten kann, müssen wir jetzt die Logik dafür im Quelltext schreiben.
Zuerst definieren wir die LED Nr. 4 und setzten den Modus des Pins auf Ausgabe. Die Loop Funktion wird nun in einer Schleife ausgeführt. In unserem Kontext bedeutet HIGH Strom fließt und LOW bedeutet kein Strom fließt. Jetzt setzten wir den Pin G4, der als LED definiert ist, auf HIGH. Danach integrieren wir eine Verzögerung von 2 Sekunden (2.000 Millisekunden), um das Blinken zu erzeugen. Nun setzten wir die LED wieder auf LOW. Dieser Vorgang wiederholt sich nun und unsere LED blinkt durchgängig. (Zusätzliche Kommentare in grau im Quelltext).

Quelltext

```
#include <Arduino.h>

// definiere welcher Pin genutzt wurde
// (D4 in unserem Fall)
#define LED 4

// wird einmal ausgeführt
void setup() {
  Serial. begin(115200);

  // setzte den Modus auf Ausgabe,
  // damit Strom ausgegeben werden kann um die LED einzuschalten
  pinMode (LED, OUTPUT);
}

// wird in einer Schleife ausgeführt
void loop() {
  // HIGH = Strom
  // schalte den Strom auf Pin D4 ein
  digitalWrite(LED, HIGH);

  // warte 2 Sekunden
  delay (2000);

  // LOW = kein Strom
  // schalte den Strom auf Pin D4 aus
  digitalWrite(LED, LOW);

  // warte 2 Sekunden
  delay (2000);

}
```

Fehlerbehebung

Funktioniert der Upload nicht?
Stelle sicher, dass du den Arduino richtig verbunden hast und bei der Auswahl vom Board/Port alles stimmt. Versuche erste mit dem “Pfeil” den Quellcode auf Fehler zu überprüfen. Sollten dabei welche auftreten, lese den Hinweis aufmerksam durch. Überprüfe den Quellcode zusätzlich dann auf Tippfehler, eine einziger Tipp Fehler reicht schon aus, damit der Quellcode nicht mehr funktioniert. Die LEDs auf dem Arduino sollte bereits beim Anschließen an den PC leuchten, wenn nicht versuche es mit einem anderen Kabel.

Fehler im Quelltext?
Stell sicher, dass du an alle Klammer “{}” und Semikolons “;” gedacht hast. Kontrolliere auch auf Rechtschreibfehler im Quelltext. Vergiss auch nicht LOW und HIGH richtig zu benutzen.

LED blinkt nicht?
Überprüfe, ob der Quellcode richtig und ohne Fehler auf das Module hochgeladen wurde. (Keine Fehlermeldung und eine LED blinkt meistens am Module während des Hochladens) Überprüfe außerdem ob die Kabel mit den richtigen Pins verbunden sind. Das müssen GND und D4 sein! Kontrolliere auch ob die Verkabelung auf dem Breadboard stimmt, d. h. Anode/Kathode der LED richtig verkabelt, einen passenden Widerstand gewählt und die Komponenten sind auf dem Breadboard verbunden (siehe Breadboard Grafik).

Sollte es trotzdem nicht funktionieren kannst du dich an Foren für Hilfe wenden. In der nächsten Sektion “Wie geht es nun weiter?” kannst du unter dem Punkt “Funktioniert etwas nicht?” Links zu Hilfsforen finden.

Wie geht es nun weiter?

Du kennst dich jetzt mit den grundlegenden Funktionen des Arduino Nano aus, wie dem Verbinden des Arduino Nano mit deinem Rechner. Auch kennst du den grundlegenden Aufbau der Arduino IDE zum Programmieren vom Arduino. Darüber hinaus hast du dich mit dem Aufbau des Arduino Nano auseinandergesetzt und den unterschiedlichen Modellen.

Mit den Grundlagen im Hinterkopf, bist du jetzt bereit auf eigene Faust zu experimentieren. Probiere einfach mal all dein Zubehör aus, was du eventuell im Starter-Kit mitbekommen hast. Auf dieser Basis lasse sich dann zusätzliche, komplexere Projekte aufbauen, bei denen nur deine Kreativität Grenzen setzt. Dies könnte zum Beispiel ein Musiksensitiver LED-Strip, einen Lügendetektor oder als Thermometer mit Anzeige. Wenn du dir auch mehr zu traust wären auch Projekte wie ein Bewässerungssystem oder eine Wetterstation für dein Smart Home möglich. Auf ein fröhliches Experimentieren!

Nützlich hierfür können auch weitere Ressourcen sein:
Die Arduino Nano Modelle findest du **hier**.

Tutorials Einsteiger Projekte findest du **hier**.
(Einsteiger/Deutsch)

Tutorials für Projekte findest du **hier**.
(verschiedene Schwierigkeiten/English)

Einstieg programmieren mit C für den Arduino
Sprache Referenz Arduino IDE findest du **hier**.
Einstieg in C als pdf Datei findest du **hier**.

Funktioniert etwas nicht?
Das Arduino Help Center findest du **hier**. (Englisch)
Und das Arduino Forum **hier**.

Viel Spaß bei all deinen kommenden Arduino Projekten!