

Arduino-Projekte für Young Makers

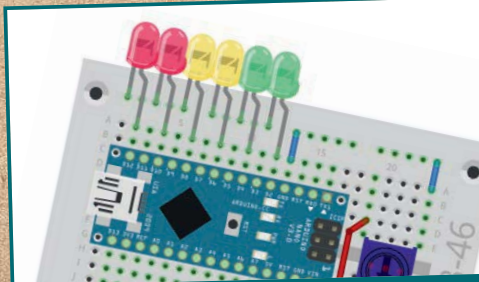
Selber programmieren hat noch nie so viel Spaß gemacht ...

... wie mit den Arduino-Projekten für Young Makers. Denn mit diesem Paket geht es gleich ans Eingemachte: Schon bald blinkt die erste LED, und kurz darauf hast du eine ganze Ampelanlage aufgebaut und programmiert. Oder hast du schon mal einen elektronischen Würfel mit einem Knete-Controller gesteuert? Nein? Dann leg gleich los!

! Das Handbuch in englischer Sprache ist als Download unter www.conrad.de erhältlich.
The English version of the handbook can be downloaded at www.conrad.de.



Speziell für Kinder konzipierte Programmiersprache



Projekte, die Spaß machen

Diese Teile sind enthalten:

Arduino-kompatible Nano-Platine, LEDs, Widerstände und weitere elektronische Bauteile, Steckbrett und Knete.

Diese Projekte setzt du um:

Wechselblinklicht, Fußgängerampel, LED-Würfel, LED-Dimmer, Pegelanzeige

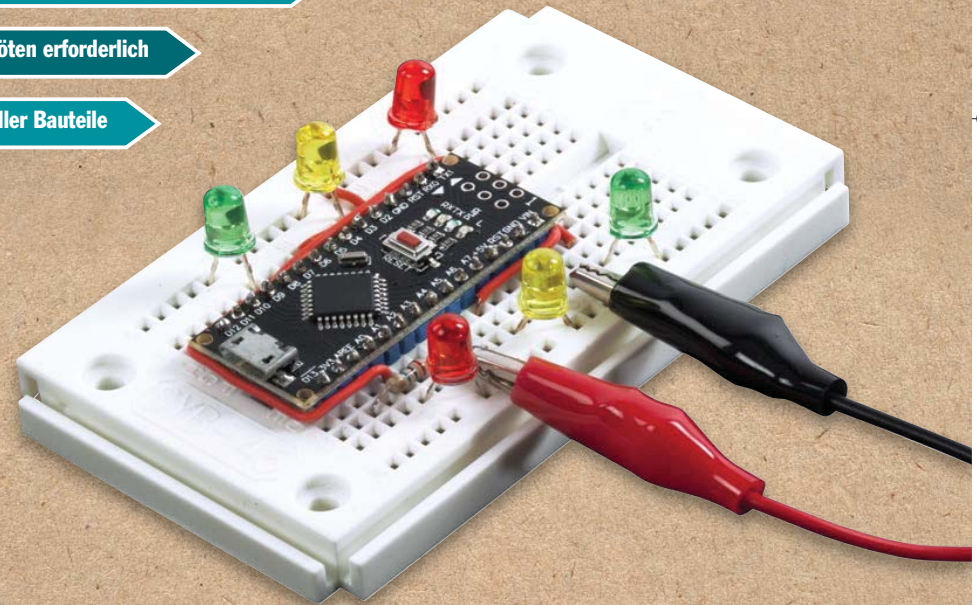
Arduino-Projekte für Young Makers

Mit Arduino-kompatibler Platine

einfache Programmierung mit Snap!

kein Löten erforderlich

inkl. aller Bauteile



Arduino-Projekte für Young Makers

Für Kinder unter 14 Jahren nicht geeignet.
This is not a toy! Not suitable for children under 14 years.

2017/01



© 2017 Franzis Verlag, Richard-Reitzner-Allee 2, D-85540 Haar, Innovationen, Irrtümer und Druckfehler vorbehalten.
Subject to innovation, errors and printing errors.

FRANZIS

FRANZIS

POWERED BY
CONRAD



Arduino-Projekte für Young Makers





**Liebe Kunden!**

Dieses Produkt wurde in Übereinstimmung mit den geltenden europäischen Richtlinien hergestellt und trägt das CE-Zeichen. Der bestimmungsgemäße Gebrauch ist in der beiliegenden Anleitung beschrieben.



Bei jeder anderen Nutzung oder Veränderung des Produktes sind allein Sie für die Einhaltung der geltenden Regeln verantwortlich. Bauen Sie die Schaltungen deshalb genau so auf, wie es in der Anleitung beschrieben wird. Das Produkt darf nur zusammen mit dieser Anleitung weitergegeben werden.



Das Symbol der durchkreuzten Mülltonne bedeutet, dass dieses Produkt getrennt vom Hausmüll als Elektroschrott dem Recycling zugeführt werden muss. Wo Sie die nächstgelegene kostenlose Annahmestelle finden, sagt Ihnen Ihre kommunale Verwaltung.

Achtung! Augenschutz und LEDs:

Blicken Sie nicht aus geringer Entfernung direkt in eine LED, denn ein direkter Blick kann Netzhautschäden verursachen! Dies gilt besonders für helle LEDs im klaren Gehäuse sowie in besonderem Maße für Power-LEDs. Bei weißen, blauen, violetten und ultravioletten LEDs gibt die scheinbare Helligkeit einen falschen Eindruck von der tatsächlichen Gefahr für Ihre Augen. Besondere Vorsicht ist bei der Verwendung von Sammellinsen geboten. Betreiben Sie die LEDs so wie in der Anleitung vorgesehen, nicht aber mit größeren Strömen.

Alle Rechte vorbehalten, auch die der fotomechanischen Wiedergabe und der Speicherung in elektronischen Medien. Das Erstellen und Verbreiten von Kopien auf Papier, auf Datenträgern oder im Internet, insbesondere als PDF, ist nur mit ausdrücklicher Genehmigung des Verlags gestattet und wird widrigenfalls strafrechtlich verfolgt.

Die meisten Produktbezeichnungen von Hard- und Software sowie Firmennamen und Firmenlogos, die in diesem Werk genannt werden, sind in der Regel gleichzeitig auch eingetragene Warenzeichen und sollten als solche betrachtet werden. Der Verlag folgt bei den Produktbezeichnungen im Wesentlichen den Schreibweisen der Hersteller.

Alle in diesem Buch vorgestellten Schaltungen und Programme wurden mit der größtmöglichen Sorgfalt entwickelt, geprüft und getestet. Trotzdem können Fehler im Buch und in der Software nicht vollständig ausgeschlossen werden. Verlag und Autor haften in Fällen des Vorsatzes oder der groben Fahrlässigkeit nach den gesetzlichen Bestimmungen. Im Übrigen haften Verlag und Autor nur nach dem Produkthaftungsgesetz wegen der Verletzung des Lebens, des Körpers oder der Gesundheit oder wegen der schuldhaften Verletzung wesentlicher Vertragspflichten. Der Schadensersatzanspruch für die Verletzung wesentlicher Vertragspflichten ist auf den vertragstypischen, vorhersehbaren Schaden begrenzt, soweit nicht ein Fall der zwingenden Haftung nach dem Produkthaftungsgesetz gegeben ist.

Autor: Christian Immler
Art & Design: www.ideehoch2.de
Satz: DTP-Satz A. Kugge, München

© 2017 Franzis Verlag GmbH, Richard-Reitzner-Allee 2, 85540 Haar

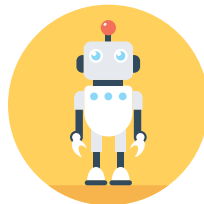




Inhaltsverzeichnis



1 Nano-Platine und PC vorbereiten.....	6
Arduino und kompatible Platinen	7
Was du sonst noch brauchst	7
Software installieren	8
2 Die Teile im Paket	12
Steckbretter.....	14
Knete, Draht und Krokodilklemmenkabel.....	14
Widerstände.....	20
Potenziometer	22
LEDs.....	22
3 Wechselblinklicht.....	26
Programmieren mit Snap4Arduino.....	27
So funktioniert das Programm	31
4 Fußgängerampel	36
Es geht los	41
So funktioniert das Programm	45
5 LED-Würfel	50
So funktioniert das Programm	54
LED-Würfel mit echtem Würfeffekt.....	60
6 LED dimmen.....	62
So funktioniert das Programm	66
7 Pegelanzeige	68
So funktioniert das Programm	70



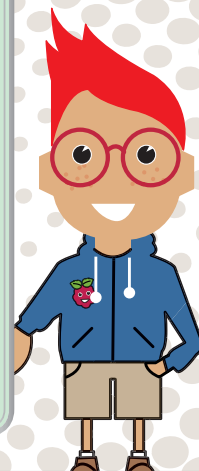
1 Nano-Platine und PC vorbereiten



Das Programmieren von Mikrocontrollern war früher nur etwas für Ingenieure und Informatiker. Arduino ermöglicht dank übersichtlicher Hardware und einfach zu verstehender Software auf einmal jedem den Einstieg in die Mikrocontrollertechnik. Die Arduino-Plattform, eine Serie kleiner, kostengünstiger Platinen auf der Basis von Atmel-Mikrocontrollern, wurde ursprünglich für Bastler und Künstler entwickelt, die damit interaktive elektronische Objekte schufen. Die Plattform ist inzwischen in

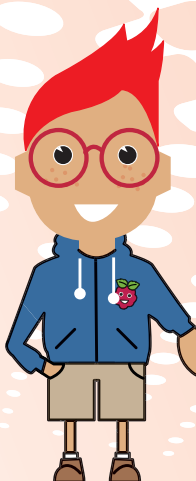
DER NAME „ARDUINO“

Der Arduino kommt aus Italien und wurde nach dem italienischen König Arduino benannt, der bis ins Jahr 1005 in Ivrea, dem Firmensitz des Arduino-Herstellers, herrschte. Nach diesem König ist dort auch die Lieblingsbar der Arduino-Entwickler Massimo Banzi und David Cuartielles benannt.



WAS IST EIN MIKROCONTROLLER?

Ein Mikrocontroller ist ein kleiner (mikro = winzig) programmierbarer Elektronikchip, der zur Steuerung (to control = steuern) von unterschiedlichster Hardware verwendet wird. Mikrocontroller sind heute in den meisten Geräten eingebaut, die eine elektronische Steuerung haben, wie unter anderem Küchengeräte, Elektronikspielzeug, Alarmanlagen, Heizungsanlagen oder Fahrradachos. Im Gegensatz zu einem „echten“ Computer haben sie kein Betriebssystem und auch keine Festplatte. Prozessor, Programmspeicher und Datenspeicher befinden sich alle auf dem gleichen Chip, der zusätzlich auch Anschlüsse zur Steuerung externer Hardware wie LEDs, LCD-Anzeigen, Tastern und Sensoren hat.





der Maker-Szene zum allgemeinen Standard für mikrocontrollergesteuerte Hardwareprojekte geworden.

Nano-Platine enthalten. Der Nano hat gegenüber dem klassischen UNO eine kleinere Bauform und kann direkt, d. h. ohne zusätzliche Kabel, auf ein Steckbrett gesteckt werden.

ARDUINO UND KOMPATIBLE PLATINEN

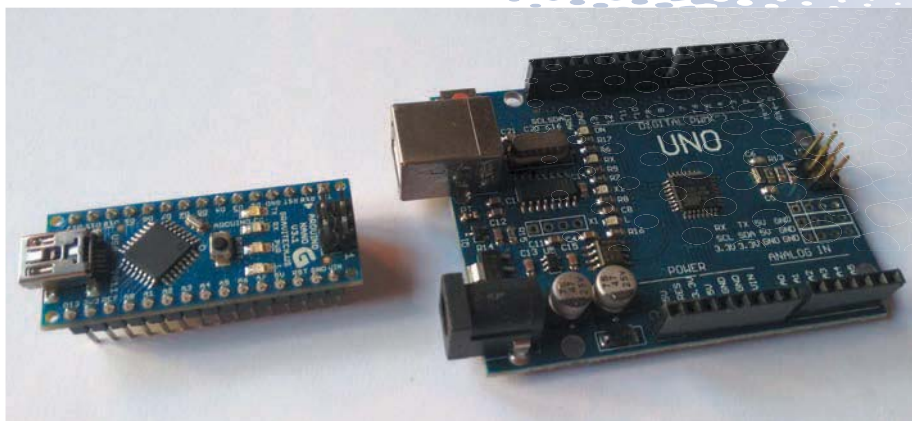
Die Arduino-Plattform bietet mittlerweile eine große Vielfalt von Platinen für unterschiedliche Anwendungszwecke.

WAS DU SONST NOCH BRAUCHST

In unserem Paket sind bereits alle Bauteile enthalten, die du zum Aufbau der Experimente benötigst.

Neben dem Original-Arduino gibt es jede Menge kompatibler Platinen, die nicht nur fast gleich aussehen, sondern auch wie der echte Arduino funktionieren. Für die Experimente ist in diesem Paket eine kompatible

Zur Programmierung des Arduino verwenden wir in den Experimenten einen Windows-PC. Die Arduino-Software wie auch die interaktive Programmiersprache Snap4Arduino werden außer für Windows auch für



Nano und UNO im Größenvergleich

1 Nano-Platine und PC vorbereiten



Linux und Mac OS angeboten. Eine App für Chromebooks befindet sich noch in der Experimentierphase, funktioniert aber schon ziemlich zuverlässig.

USB-KABEL

Die Verbindung zwischen PC und Nano erfolgt über ein USB-Kabel mit einem MicroUSB-Stecker auf einer Seite. Du brauchst dir ein solches Kabel nicht extra zu besorgen, fast alle modernen Smartphones verwenden diesen Steckertyp. Dieses USB-Kabel wird gleichzeitig zur Programmierung und zur Stromversorgung der Platine verwendet. Der Nano braucht also für unsere Experimente kein eigenes Netzteil.

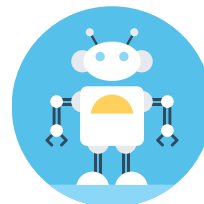
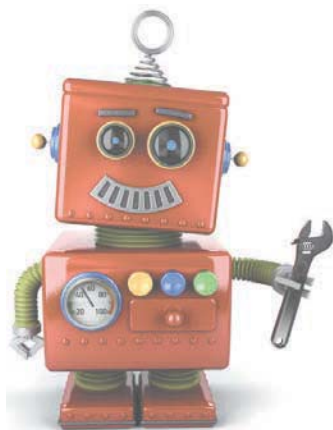
SOFTWARE INSTALLIEREN

Zur Programmierung des Nano sind ein paar Programme auf dem PC erforderlich, die alle kostenlos heruntergeladen werden können.

Alle notwendige Software und auch die Beispielprogramme aus diesem Paket findest Du auf www.buch.cd. Gib dort den Code **15000-4** ein und lade dir die Zip-Datei herunter. Entpacke diese anschließend in ein Verzeichnis auf der Festplatte.

TREIBER INSTALLIEREN

Der Nano braucht zur Verbindung mit dem PC über das USB-Kabel einen speziellen Treiber.



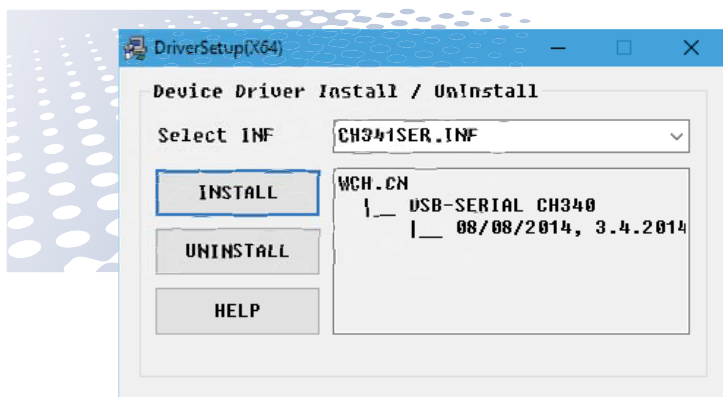


1 Schließe den Nano über das USB-Kabel am PC an. Verwende nach Möglichkeit einen USB-2.0-Anschluss, da es an USB-3.0-Anschlüssen eher zu Verbindungsproblemen kommen kann.

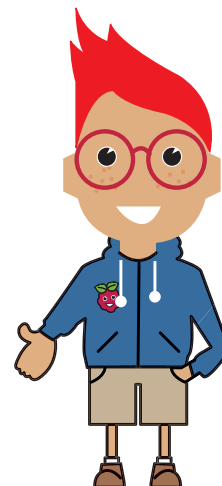
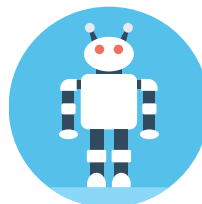
Anfrage der Windows Benutzerkontensteuerung bestätigt werden. Solltest du nicht mit Administratorrechten angemeldet sein, muss ein Administrator diese Anfrage bestätigen.

2 Installiere den Treiber mit einem Doppelklick auf die Datei **CH341SER.EXE** aus dem entpackten Download. Zur Installation muss eine

3 Klicke im Installationsdialog auf **Install** und warte, bis eine Bestätigung erscheint, dass der Treiber installiert wurde.



4 Danach kannst du das Fenster der Treiberinstallation schließen.



1 Nano-Platine und PC vorbereiten



DIE PROGRAMMIERSPRACHE SNAP4ARDUINO

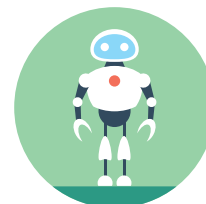
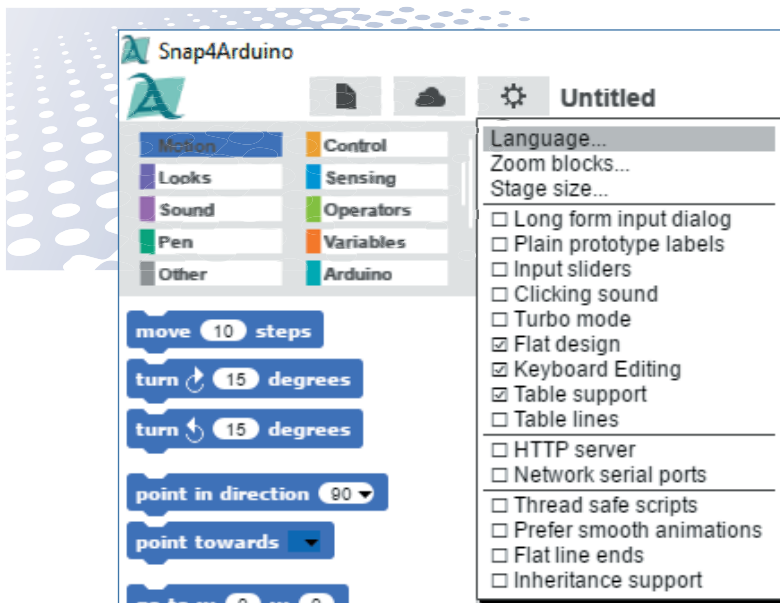
Für die Projekte im Paket verwenden wir die besonders einfach zu erlernende Programmiersprache Snap4Arduino. Lade dir die aktuelle Version bei snap4arduino.org herunter oder verwende einfach die Datei **Snap4Arduino.exe** aus den Downloads zum Paket.

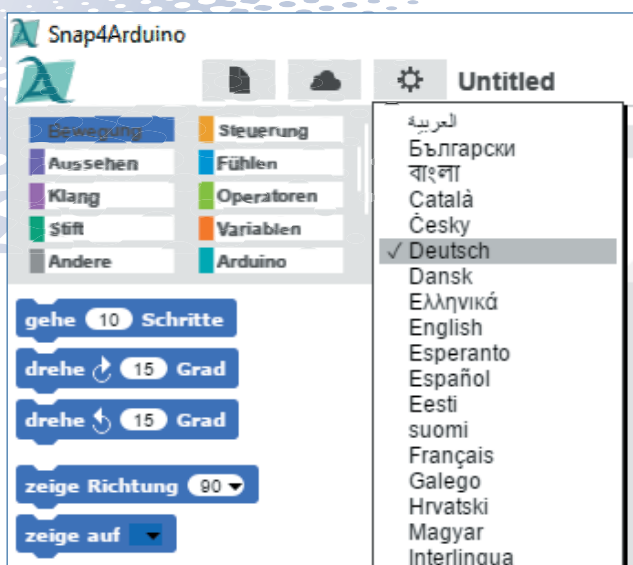
Snap4Arduino verwendet wie viele andere interaktive Programme die StandardFirmata als Grundlage

zur Kommunikation zwischen PC und Arduino. Auf der Nano-Platine in diesem Paket ist StandardFirmata bereits vorinstalliert. Darum brauchst du dich also nicht zu kümmern.

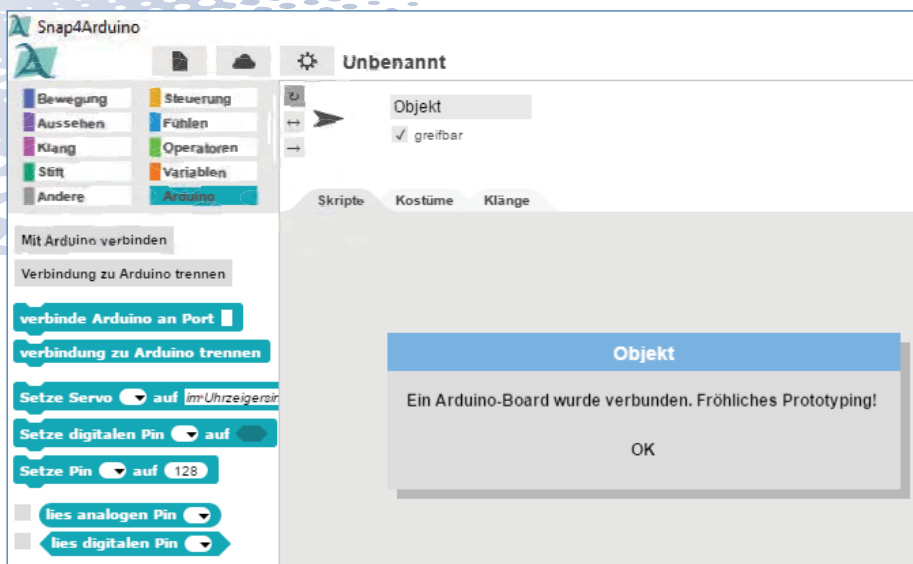
Klicke in Snap4Arduino oben links auf das Einstellungen-Symbol und wähle im Menü **Language**. Wähle dort in der Liste **Deutsch** aus.

Bevor du mit dem Programmieren beginnen kannst, muss eine Verbindung zum Nano hergestellt werden. Gehe dazu oben links auf die Block-





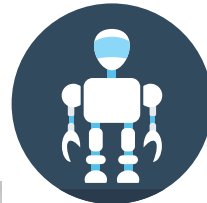
palette **Arduino** und klicke auf **Mit Arduino verbinden**. Wähle dann den vom Nano verwendeten Port aus. Snap4Arduino unterstützt mehrere Arduinos an einem PC. Wenn mehrere Ports angezeigt werden, ist meistens der mit der höheren Nummer der richtige. Probiere es einfach aus. Bestätige die erfolgreiche Verbindung mit einem Klick auf **OK**.



2 Die Teile im Paket



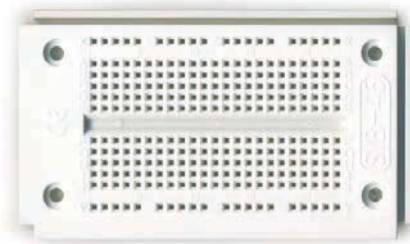
In diesem Kapitel erhältst du das nötige Expertenwissen, um mit den Bauteilen in diesem Paket perfekt umgehen zu können.



1 Arduino-kompatible Nano-Platine



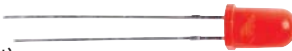
1 Steckbrett



2 LEDs grün
(mit Vorwiderstand)



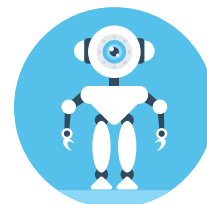
2 LEDs rot
(mit Vorwiderstand)



2 LEDs gelb
(mit Vorwiderstand)



1 20-MOhm-Widerstand



12 Der kleine Hacker



1 Potenziometer 15 kOhm



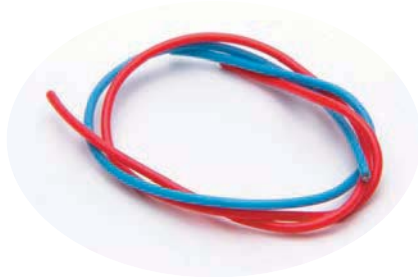
2 Krokodilklemmenkabel



2 Knete in verschiedenen
Farben (die Farben der
Knete können abweichen)



1 Schaltdraht
(isoliert)



1 blanker Schaltdraht



2 Die Teile im Paket



STECKBRETTER

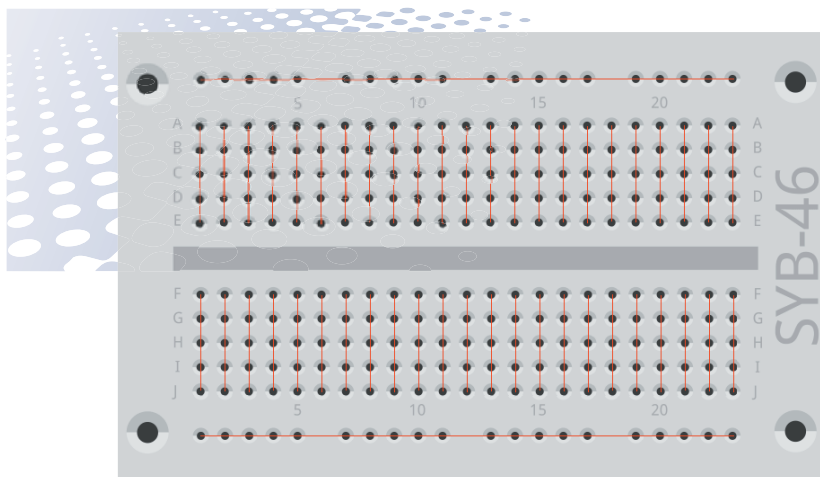
Für den schnellen Aufbau elektronischer Schaltungen, ohne löten zu müssen, ist im Paket ein Steckbrett enthalten. Elektronische Bauteile können direkt in ein Lochraster gesteckt werden.

Bei diesen Steckbrettern sind die äußeren Längsreihen alle mit Kontakten (X und Y) miteinander verbunden. Diese Kontaktreihen werden oft als Plus- und Minuspol zur Stromversorgung der Schaltungen genutzt. In den anderen Kontaktreihen sind jeweils fünf Kontakte (A bis E und F bis J) quer miteinander ver-

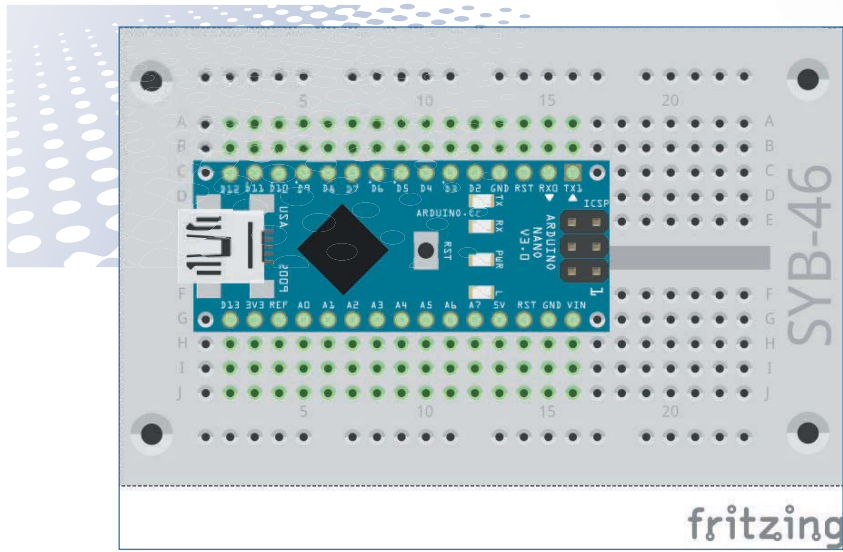
bunden, wobei in der Mitte der Platine eine Lücke ist. So können in die Mitte größere Bauelemente, wie z. B. die Nano-Platine, gesteckt und nach außen hin verdrahtet werden.

KNETE, DRAHT UND KROKODILKLEMMENKABEL

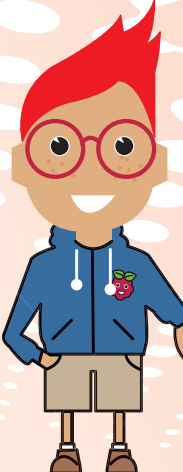
Im Paket befinden sich Knete, Draht und Krokodilklemmenkabel. Damit bauen wir Kontaktschalter, um die Elektronikexperimente zu steuern. Knete kann so die sonst üblichen Tasten ersetzen.



Das Bild zeigt, welche Löcher auf dem Steckbrett miteinander verbunden sind.



Der Nano auf einem Steckbrett - links außen der USB-Anschluss



WARUM EIGENTLICH KNETE?

Knete leitet den Strom etwa so gut wie deine Haut. Sie lässt sich leicht in jede beliebige Form bringen, und ein Knetekontakt fasst sich viel besser an als ein einfaches Stück Draht. Die Fläche, mit der deine Hand den Kontakt berührt, ist deutlich größer. So kommt es nicht so leicht zu einem „Wackelkontakt“. Natürlich kannst du auch andere leitfähige Dinge, wie zum Beispiel einen Löffel oder eine Münze, an ein Krokodilklemmenkabel klemmen und als Kontakt nutzen. Die neuartige, leichte „Kugelnknete“ mit ihren eingebauten Styroporkügelchen eignet sich nicht. Sie leitet den Strom nicht gut genug.

2 Die Teile im Paket



KNETEKONTAKTE BAUEN

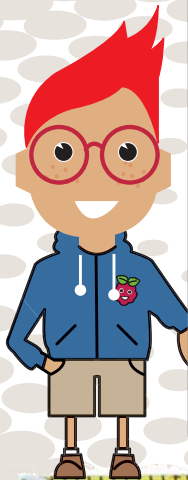
1 Für einen Knetekontakt brauchst du als Erstes ein Stück Knete. Forme aus einem Stück der Knete eine etwa 2 bis 3 cm große Kugel oder eine andere Form, die gut in der Hand liegt.

2 Schneide mit einer Zange ein etwa 4 bis 5 cm langes Stück vom blanken Draht ab, biege es zu einem „U“ und stecke die beiden Enden so weit in die Knetekugel, dass der Draht nicht mehr von allein herausrutscht.

3 Am Ende sollte der Draht noch etwa 0,5 cm aus der Knete herausstehen. Klemme hier eines der Krokodilklemmenkabel an.

KROKODILKLEMMEN

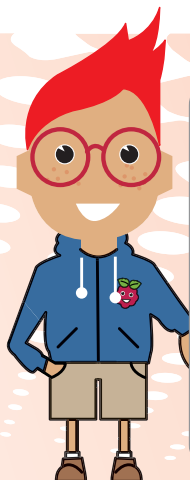
Warum die Krokodilklemmenkabel so heißen, kann sich jeder leicht vorstellen. Wir haben uns diesen Namen übrigens nicht ausgedacht. Es ist wirklich die unter Elektrikern gebräuchliche Bezeichnung für diese Art von Kabel.





9 Das andere Ende des Krokodilklemmenkabels soll mit der Elektronik auf dem Steckbrett verbunden werden. Schneide dazu noch ein kürzeres, nur etwa 2 cm langes Stück vom blanken Draht ab und biege daraus ebenfalls ein „U“. Stecke dieses dann in das Steckbrett. Auf den Zeichnungen sind diese Kabelanschlüsse in Grau dargestellt,

um die blanken Drähte von den farbigen isolierten Verbindungskabeln zu unterscheiden. Die farbigen Drahtbrücken in den Zeichnungen kannst du aus kurzen Stücken des isolierten Schalt drahts bauen. Du musst nur an beiden Enden mit einem scharfen Messer etwa 0,5 cm der Isolierung entfernen.



DRAHT SCHRÄG ABSCHNEIDEN

Der mitgelieferte Draht ist relativ dick, damit er sich nicht zu leicht verbiegt und mit den Krokodilklemmen gut greifen lässt. Dafür kann es passieren, dass er sich besonders am Anfang, wenn die Steckbretter noch neu sind, schwer in die Kontaktlöcher stecken lässt. Schneide deshalb den Draht für die Anschlüsse am Steckbrett mit der Zange schräg ab. Dann entsteht eine leichte Spitze, die sich einfacher in das Steckbrett stecken lässt.



2 Die Teile im Paket



5 Klemme zuletzt das Krokodilklemmenkabel des Knetekontakts an den gerade gebauten Anschluss auf dem Steckbrett.

Da du jetzt weißt, was ein Knetekontakt ist und wie man diese nützlichen Dinge bastelt, werden wir das bei den einzelnen Experimenten nicht mehr jedes Mal erklären.

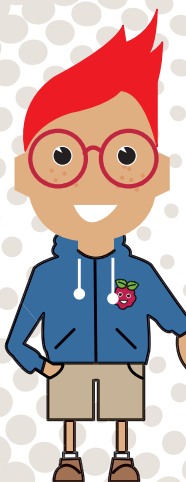
WENN DIE KNETE HART WIRD ...

Die Knete ist so weich, weil sie ein bisschen Wasser enthält. Bei trockener Luft, wenn die Knete zum Beispiel in geheizten Räumen gelagert wird, kann sie, nachdem sie ein paar Tage offen herumgelegen hat, austrocknen und lässt sich dann nicht mehr so gut kneten. Außerdem kann die Leitfähigkeit nachlassen, und die Knetekontakte funktionieren dann nicht mehr so gut wie am ersten Tag.

Wenn du die Knete ein paar Stunden nicht brauchst, packe sie immer in eine geschlossene Plastikdose.

MERKLISTE – FÜR JEDEN KNETEKONTAKT BRAUCHST DU:

- 1 Stück Knete
- 1 U-förmig gebogener blanker Draht, etwa 4 bis 5 cm, für den Anschluss der Krokodilklemme an die Knete
- 1 U-förmig gebogener blanker Draht, etwa 2 bis 3 cm, für den Anschluss der Krokodilklemme am Steckbrett
- 1 Krokodilklemmenkabel



Die Dosen, die man zum Lagern von Lebensmitteln im Kühlschrank verwendet, eignen sich gut. Du kannst auch einfach den Plastikbeutel verwenden, in dem die elektronischen Bauteile in diesem Paket verpackt sind.

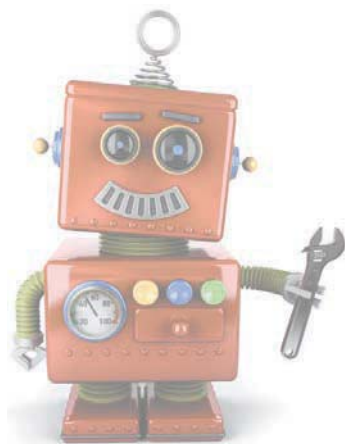
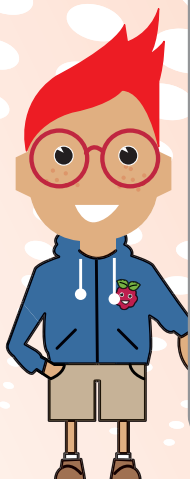
Wenn die Knete doch einmal zu hart geworden ist, hilft der folgende Trick.



KNETE WIEDER WEICH MACHEN

Nimm ein Stofftuch, das nicht fusselft (sonst hast du die Fusseln später an der Knete kleben) und mache es mit Wasser nass. Drücke es dann so weit aus, dass es sich nur noch ganz leicht feucht anfühlt. Wickle die Knete in das Tuch ein und lass sie über Nacht darin liegen. Am nächsten Morgen wird sich die Oberfläche leicht glitschig anfühlen. Knete die Masse dann gut durch. Dadurch verteilt sich die Feuchtigkeit gleichmäßig, und die Knete kann wieder ganz normal benutzt werden.

Damit die Feuchtigkeit besser in die Knete hineingelangt, forme keine Kugeln, sondern drücke die Knete platt, bevor du sie in das Tuch packst. Die Kugel ist die Form, die im Verhältnis zum Volumen die geringste Oberfläche hat, es kann also wenig Feuchtigkeit eindringen. Umgekehrt kommt über die geringe Oberfläche auch kaum Feuchtigkeit heraus. Deshalb sind die meisten Früchte annähernd kugelförmig – damit sie im Sommer nicht so schnell austrocknen.

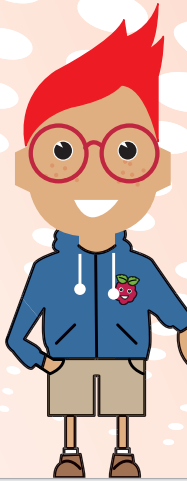


2 Die Teile im Paket



VORSICHTSMASSNAHMEN

Verbinde auf keinen Fall irgendwelche Pins des Nano miteinander und warte ab, was passiert. Einige Pins sind direkt mit Anschlüssen des Mikrocontrollers verbunden, ein Kurzschluss kann den Nano komplett zerstören – zumindest theoretisch. Die Platinen sind erstaunlich stabil gegen Schaltungsfehler. Nicht alle Pins lassen sich frei programmieren. Einige sind für die Stromversorgung und andere Zwecke fest eingerichtet. Für Logiksignale immer den 3,3-V-Pin (in den Zeichnungen: **3V3**) verwenden. Der 5-V-Pin (in den Zeichnungen: **5V**) dient zur Stromversorgung externer Hardware. Hier kann (fast) so viel Strom entnommen werden, wie das angeschlossene Netzteil liefert. Dieser Pin darf aber nicht mit einem digitalen Eingang verbunden werden.



WIDERSTÄNDE

Ein Widerstand begrenzt den Strom, der durch eine Leitung fließt. Man kann ihn sich vorstellen wie ein dünnes Stück Gartenschlauch, das in ein Wasserrohr eingebaut wird. Durch die Rohrleitung fließt dann im Ganzen weniger Wasser, nämlich nur noch so viel, wie durch den Schlauch gelangt.

Widerstände werden zur Strombegrenzung an empfindlichen elektronischen Bauteilen sowie als Vorwiderstände für LEDs verwendet. Die Maßeinheit für Widerstände ist Ohm. 1.000 Ohm entsprechen einem Kiloohm, abgekürzt kOhm. 1.000 kOhm entsprechen einem Megaohm, abgekürzt MOhm. Oft wird für die Einheit Ohm das Omega-Zeichen Ω verwendet.



Die farbigen Ringe auf den Widerständen geben den Widerstandswert an. Mit etwas Übung sind sie deutlich leichter zu erkennen als winzig kleine Zahlen, die man nur noch auf ganz alten Widerständen findet.

Die meisten Widerstände haben vier solcher Farbringe. Die ersten beiden Farbringe bezeichnen die Ziffern,

der dritte einen Multiplikator und der vierte die Toleranz. Dieser Toleranzring ist meistens gold oder silberne Farben, die auf den ersten Ringen nicht vorkommen. Dadurch ist die Leserichtung immer eindeutig. Der Toleranzwert selbst spielt in der Digitalelektronik kaum eine Rolle. Die Tabelle zeigt die Bedeutung der farbigen Ringe auf Widerständen.

Farbe	Widerstandswert in Ohm			
	1. Ring (Zehner)	2. Ring (Einer)	3. Ring (Multiplikator)	4. Ring (Toleranz)
Silber			$10^{-2} = 0,01$	$\pm 10\%$
Gold			$10^{-1} = 0,1$	$\pm 5\%$
Schwarz		0	$10^0 = 1$	
Braun	1	1	$10^1 = 10$	$\pm 1\%$
Rot	2	2	$10^2 = 100$	$\pm 2\%$
Orange	3	3	$10^3 = 1.000$	
Gelb	4	4	$10^4 = 10.000$	
Grün	5	5	$10^5 = 100.000$	$\pm 0,5\%$
Blau	6	6	$10^6 = 1.000.000$	$\pm 0,25\%$
Violett	7	7	$10^7 = 10.000.000$	$\pm 0,1\%$
Grau	8	8	$10^8 = 100.000.000$	$\pm 0,05\%$
Weiß	9	9	$10^9 = 1.000.000.000$	

Im Paket ist nur ein Widerstand enthalten:

Wert	1. Ring (Zehner)	2. Ring (Einer)	3. Ring (Multipl.)	4. Ring (Toleranz)	Verwendung
20 MOhm	Rot	Schwarz	Blau	Gold	Widerstand für Knetkontakte

2 Die Teile im Paket



In welcher Richtung ein Widerstand eingebaut wird, ist egal. Bei LEDs dagegen spielt die Einbaurichtung eine wichtige Rolle.

POTENZIOMETER

Ein Potenziometer ist ein einstellbarer Widerstand. Der Widerstand zwischen den beiden äußeren Anschlüssen ist immer gleich. Durch Drehen des Knopfs lässt sich der Widerstand zwischen dem mittleren Anschluss und den äußeren einstellen.

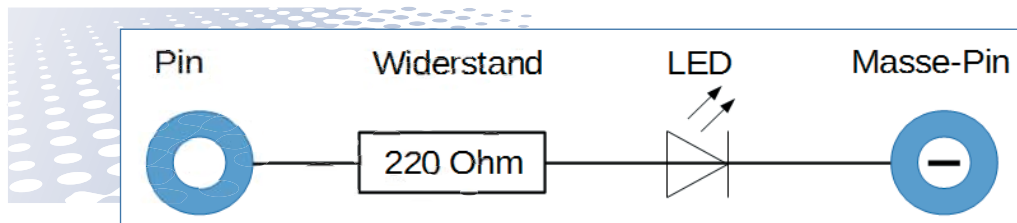
LEDS

LED ist die Abkürzung für das englische Wort „light emitting diode“, was wörtlich übersetzt „Licht abstrahlende Diode“ oder einfach Leuchtdiode bedeutet. LEDs können schön bunt leuchten, wenn Strom in

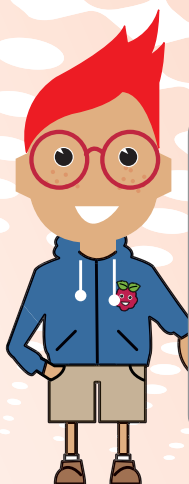
einer Richtung durch sie fließt. In der anderen Richtung lassen sie nichts durch und leuchten auch nicht.

LEDs werden in Schaltungen mit einem pfeilförmigen Dreiecksymbol dargestellt, das die Flussrichtung vom Pluspol zum Minuspol oder zur Masseleitung angibt.

Eine LED lässt in der Durchflussrichtung nahezu beliebig viel Strom durch, sie hat nur einen sehr geringen Widerstand. Um den Durchflussstrom zu begrenzen und so ein Durchbrennen der LED zu verhindern, muss üblicherweise zwischen dem verwendeten GPIO-Pin und der Anode der LED ein 220-Ohm-Vorwiderstand (Rot-Rot-Braun) eingebaut werden. Dieser Vorwiderstand schützt auch den Ausgang des Nano vor zu hohen Stromstärken. Die LEDs im Paket haben bereits einen Vorwiderstand eingebaut. Deshalb



Schaltplan einer LED mit Vorwiderstand



LED IN WELCHER RICHTUNG ANSCHLIESSEN?

Die beiden Anschlussdrähte einer LED sind unterschiedlich lang. Der längere ist der Pluspol (die Anode), der kürzere der Minuspol (die Kathode). Einfach zu merken: Das Pluszeichen hat einen Strich mehr als das Minuszeichen und macht damit den Draht optisch etwas länger. Außerdem sind die meisten LEDs auf der Minusseite abgeflacht, so wie ein Minuszeichen. Leicht zu merken: Kathode = kurz = Kante.



brauchst du keine zusätzlichen Vorwiderstände anzuschließen.

Probiere einfach einmal eine LED aus. Baue dazu die abgebildete Schaltung auf einem Steckbrett auf und schließe danach den Nano am USB-Port des PCs an. In diesem ersten Experiment wird der Nano nur als Stromversorgung für die LED genutzt. Die beiden verwendeten Pins am Nano sind immer mit +3,3 V und Masse geschal-



BENÖTIGTE BAUTEILE

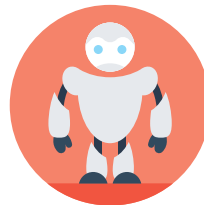
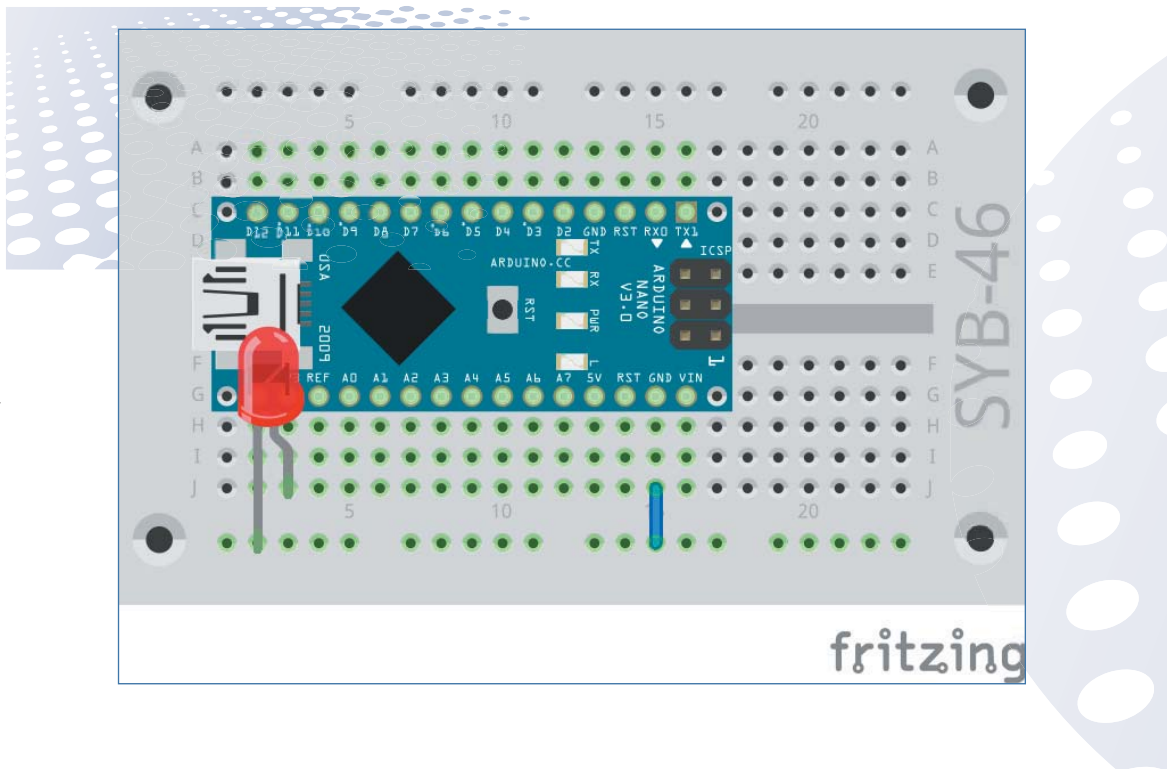
- 1x Nano
- 1x Steckbrett
- 1x LED rot (mit Vorwiderstand)
- 1x Drahtbrücke

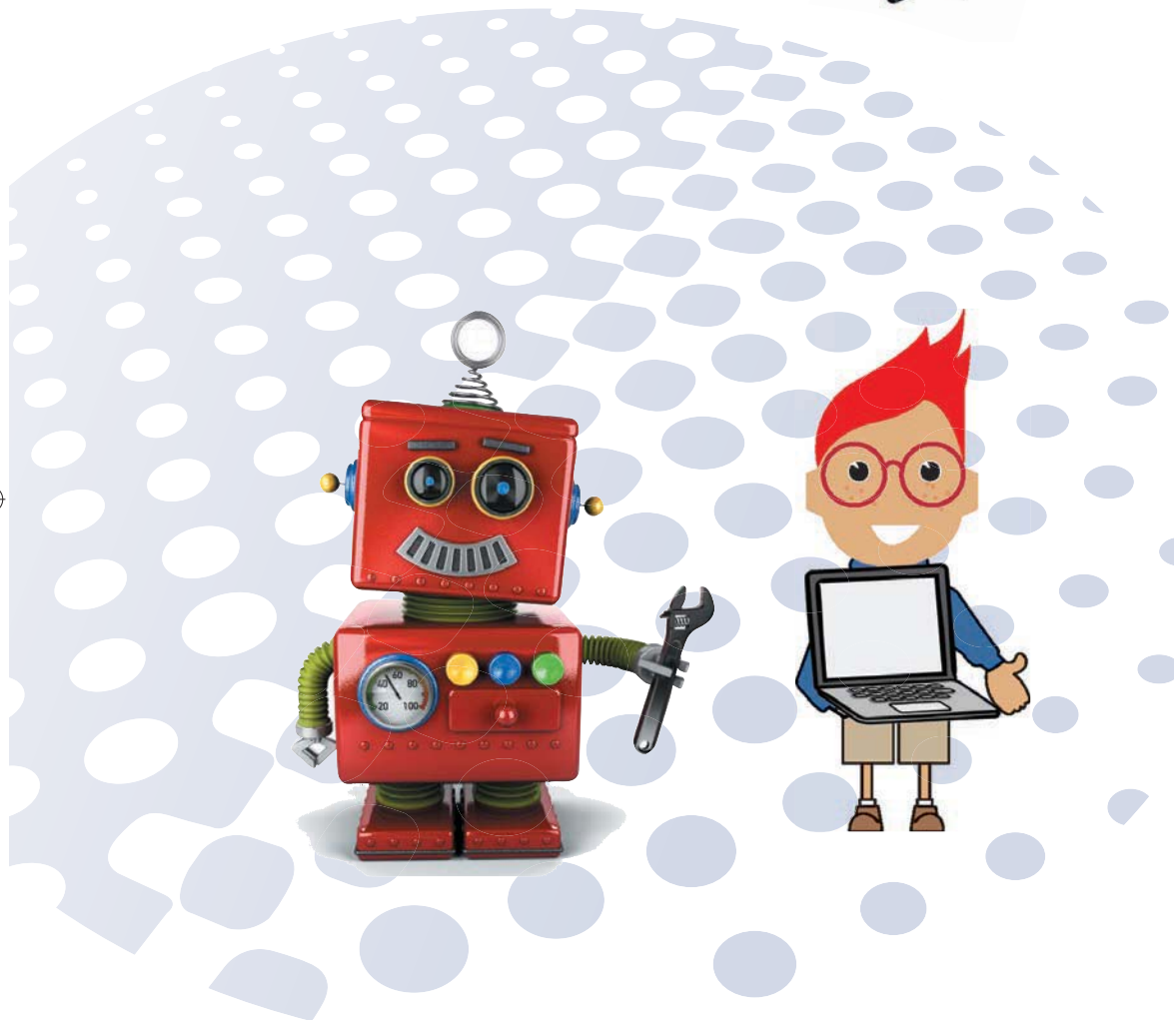
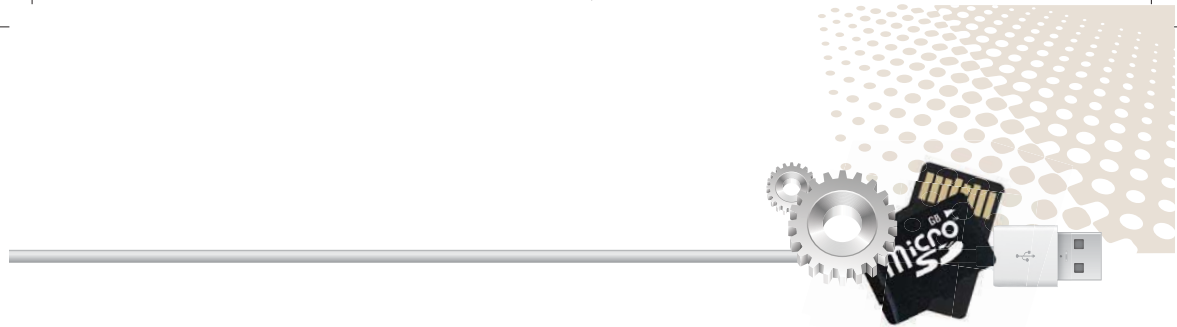


2 Die Teile im Paket

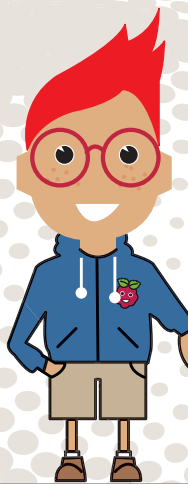


tet. Die LED leuchtet immer, man braucht keinerlei Programm dazu.





3 Wechselblinklicht

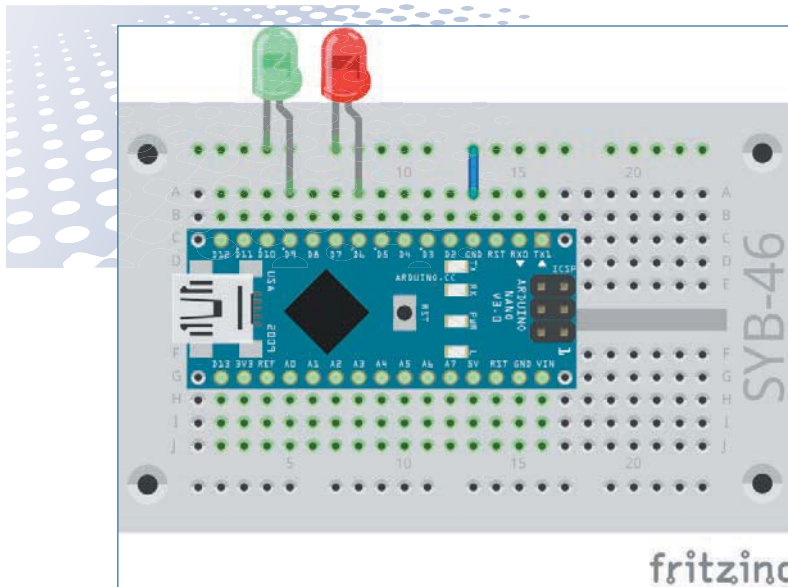


BAUTEILE

- 1x Nano
- 1x Steckbrett
- 1x LED rot (mit Vorwiderstand)
- 1x LED grün (mit Vorwiderstand)
- 1x isolierte Drahtbrücke

Jetzt geht es los. Unser erstes Snap4Arduino-Programm soll zwei LEDs abwechselnd blinken lassen. Das klingt nicht weiter aufregend, liefert aber wichtiges Fachwissen, um später weitere Elektronik anzuschließen.

Baue die LEDs wie auf dem Steckbrett in der Abbildung auf, schließe sie an die Pins D6 und D9 an und verbinde die Masseleitung mit dem GND-Pin. Das gleiche Prinzip werden wir beim Anschließen von LEDs immer anwenden.





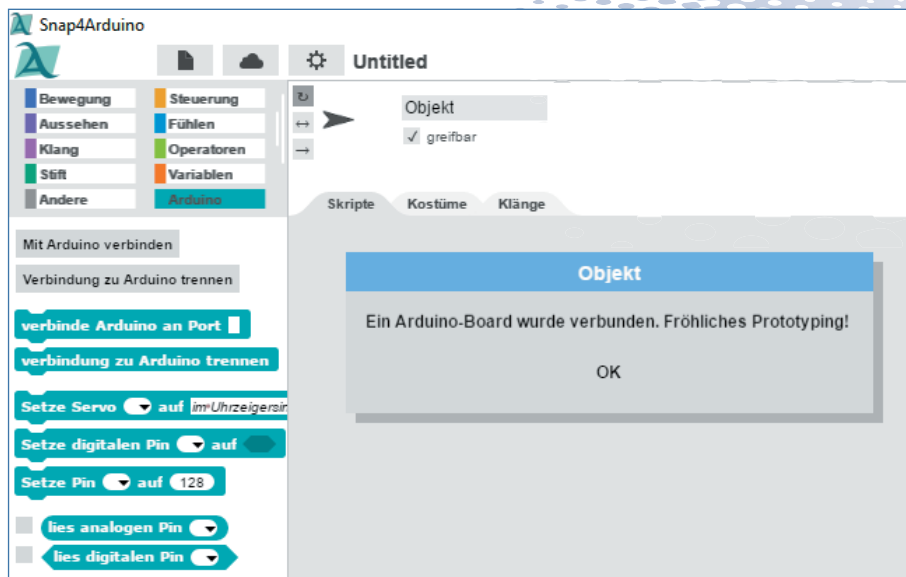
PROGRAMMIEREN MIT SNAP4ARDUINO

Starte auf dem PC Snap4Arduino. Bevor du mit dem Programmieren beginnen kannst, muss eine Verbindung zum Nano hergestellt werden. Schalte dazu oben links auf die Blockpalette **Arduino** und klicke auf **Mit Arduino verbinden**. Wähle den vom Nano verwendeten Port aus. Snap4Arduino unterstützt mehrere Arduinos an einem PC. Solange nur ein Arduino angeschlossen ist, wird

der Port automatisch ausgewählt. Bestätige die erfolgreiche Verbindung mit einem Klick auf **OK**.

Beim Öffnen eines neuen Programms in Snap4Arduino geht oft die Verbindung zum Arduino verloren. Sollte ein Fehler auftreten, wenn du ein neues Programm startest, verbinde den PC einfach mit einem Klick wieder mit dem Nano.

In Snap4Arduino braucht man beim Programmieren keinen Programm-

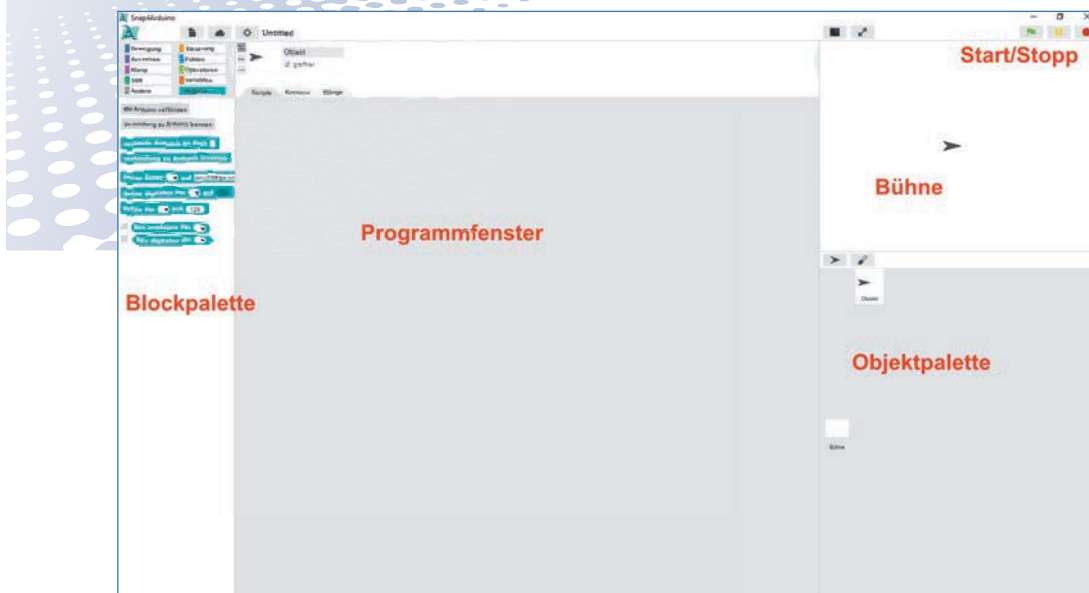
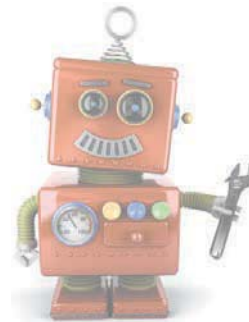


3 Wechselblinklicht



code einzugeben. Die Blöcke werden einfach per Drag and Drop aneinandergehängt. Die Programmieroberfläche enthält vier Teilfenster, die so bezeichnet werden, wie auf der Abbildung zu sehen ist. Die Blockpalette im linken Teil des Fensters enthält nach Themen geordnet die verfügbaren Blöcke. Diese Themenbereiche lassen sich über die farbigen Schaltflächen oben links in der

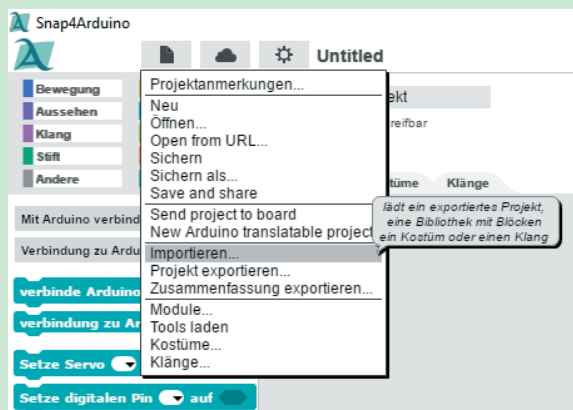
Blockpalette auswählen. Die Blöcke haben zur besseren Unterscheidung auch die entsprechenden Farben.



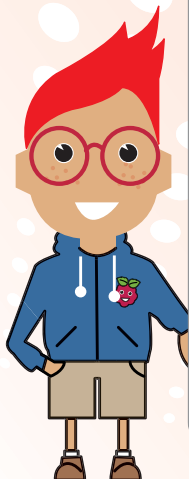


DIE PROGRAMME ZUM PAKET

Am besten baust du die Programme anhand der Abbildungen am Bildschirm selbst zusammen. Du kannst sie aber auch unter www.buch.cd herunterladen. Entpacke in diesem Fall die ZIP-Datei aus dem Download in ein Verzeichnis auf der Festplatte. Klicke dann oben links in Snap4Arduino auf das Dateisymbol und wähle **Importieren**, um die Programme, die im XML-Format vorliegen, in Snap4Arduino zu importieren. Einmal importierte Programme stehen danach in der eigenen Bibliothek, die über den Menüpunkt **Öffnen** erreichbar ist, zur Verfügung.



Dieses erste Programm findest du als **01led01** bei den Downloads.

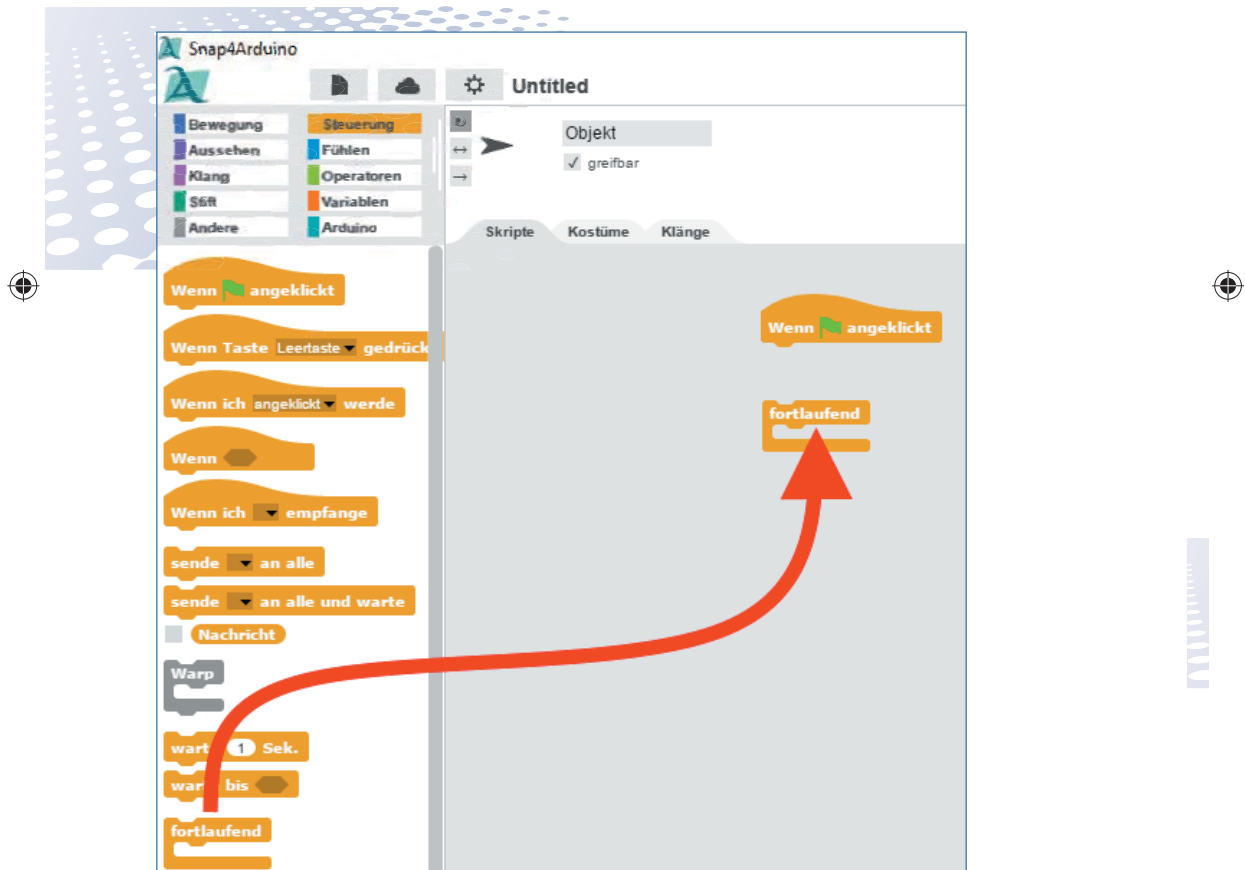


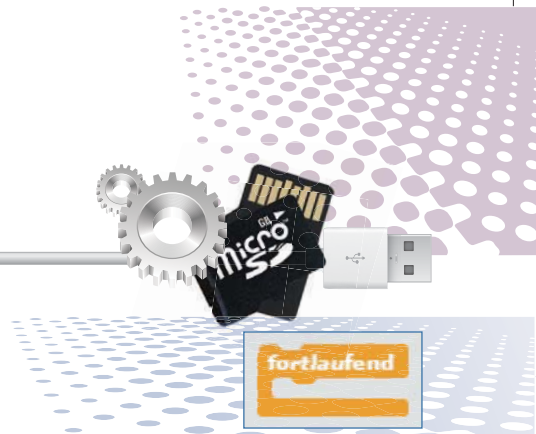
3 Wechselblinklicht



Klicke oben links auf das gelbe Symbol **Steuerung**. Dann werden in der Blockpalette links die Blöcke zur Steuerung angezeigt.

Ziehe die Blöcke, die du brauchst, einfach aus der Blockpalette links in das Programmfenster in der Mitte. Wenn du einen neuen Block dicht





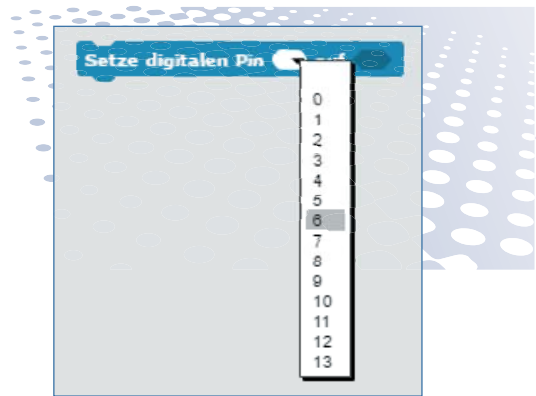
unter einen anderen schiebst, rastet er ein und verbindet sich automatisch.

Einen nicht mehr benötigten Block kannst du jederzeit von den anderen Blöcken abtrennen und wieder zurück auf die Blockpalette ziehen.

Um einen digitalen Pin des Nano ein- oder auszuschalten, verwendet man den Block **Setze digitalen Pin... auf...** von der Palette **Arduino**. Wähle im Listenfeld den Pin 6 aus.

SO FUNKTIONIERT DAS PROGRAMM

Das Programm startet wie die meisten Snap4Arduino-Programme mit dem Block **Wenn grünes Fähnchen angeklickt**, der auf der Blockpalette **Steuerung** zu finden ist. Der Block ist oben rund, passt also unter keinen anderen Block. Er muss immer als Erstes gesetzt werden.



Eine **fortlaufend**-Schleife sorgt dafür, dass die beiden LEDs abwechselnd endlos blinken, und zwar so lange, bis du auf das rote Stopp-Symbol oben rechts in Snap4Arduino klickst.

Jeder digitale Pin kann den Logikwert **wahr** oder **falsch** annehmen. Dabei steht **wahr** für **eingeschaltet** und **falsch** für **ausgeschaltet**. Auf der Palette **Operatoren** ist ein Block mit einem Schalter zu finden, der durch einfaches Anklicken auf den Wert **wahr** oder **falsch** gesetzt werden kann.



3 Wechselblinklicht



Ziehe den Block **wahr** in das passende Feld ganz rechts im Block **Setze digitalen Pin... auf...** und ziehe den ganzen Block dann in die **fortlaufend**-Schleife.



DEZIMALPUNKT STATT KOMMA

Snap4Arduino verwendet wie viele amerikanische Programme den Punkt als Dezimaltrennzeichen, nicht das in Deutschland übliche Komma. Eine Hundertstelsekunde Wartezeit schreibt man also **0.01** und nicht **0,01**.



Nachdem die LED an Pin 6 eingeschaltet ist, wartet das Programm eine Hundertstelsekunde. Solche sogenannten „Timeouts“ (oder auf Deutsch „Auszeiten“) baut man immer dann ein, wenn Programme direkt mit Hardware kommunizieren. Sie verhindern, einfach ausgedrückt, dass sich ein Programm „überschlägt“ und irgendein Hardwareereignis nicht mehr mitbekommt. Zwischen dem Setzen zweier Pins sollte immer eine minimale Wartezeit eingebaut werden.



Ziehe dazu einen Block **warte... Sek** von der Blockpalette **Steuerung** unter den Block **Setze digitalen Pin... auf...**

innerhalb der **fortlaufend**-Schleife und trage im Zahlenfeld den Wert **0.01** ein.



Immer wenn die LED am Pin 6 leuchtet, soll die andere LED am Pin 9 ausgeschaltet sein. Ziehe dazu einen weiteren Block **Setze digitalen Pin...**



auf... in die Schleife. Wähle in diesem Block den Pin 9 aus und ziehe ganz rechts in das Feld einen Logikwert **falsch** von der Blockpalette **Operatoren**.

Danach wird auf die gleiche Weise die LED am Pin 9 ein- und die am Pin 6 ausgeschaltet. Nach einer weiteren halben Sekunde wiederholt sich die Schleife und schaltet die LED am Pin 6 ein und die am Pin 9 aus.



Danach soll das Programm eine halbe Sekunde warten, bis die LED am Pin 6 wieder ausgeschaltet und dafür die LED am Pin 9 eingeschaltet wird. Füge dazu einen **warte...Sek** Block ein und schreibe in das Textfeld **0.5**.

Wenn du das Programm aus den Blöcken fertig zusammengebaut hast und die LEDs am Arduino angeschlossen sind, klicke auf das grüne Fähnchen rechts oben. Damit wird das Programm gestartet. Die LEDs blinken.



3 Wechselblinklicht

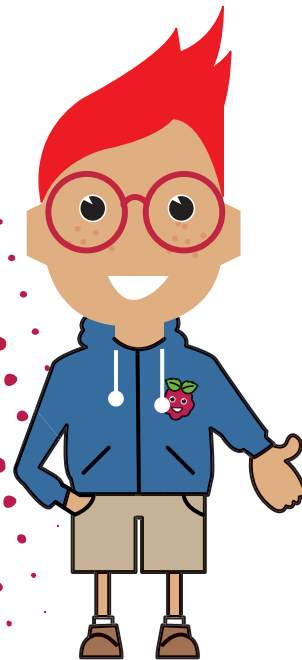


Mit einem Klick auf das rote Stopp-Symbol oben rechts kannst du das Programm wieder anhalten.



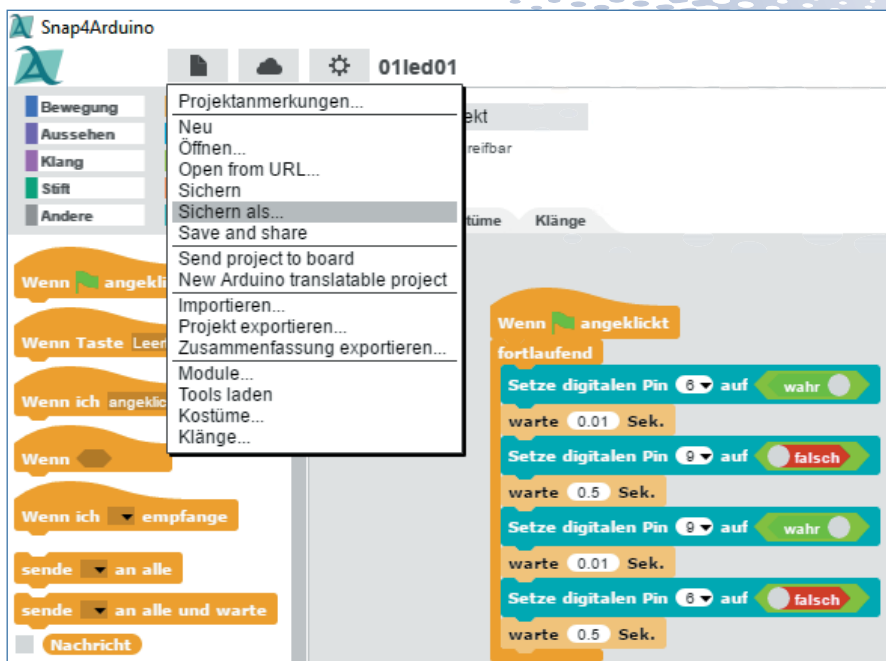
TIPP

Soll die LED schneller blinken, verkürze die Zeiten in den beiden *warte...* *Sek*-Blöcken innerhalb der Schleife. Soll sie langsamer blinken, verlängere die Wartezeiten.





Vergiss nicht, das fertige Programm mit dem Menüpunkt **Sichern als** im Dateimenü zu speichern, um es später wiederverwenden zu können.



4 Fußgängerampel



Einzelne LEDs ein- und wieder auszuschalten mag im ersten Moment ganz spannend sein, aber dafür braucht man eigentlich keinen Computer. Eine Ampel mit ihrem typischen Lichtwechsel von Grün über Gelb nach Rot und dann über die Lichtkombination Rot-Gelb wieder zu Grün ist da schon viel spannender.

Das nächste Experiment stellt eine einfache Ampelschaltung mit Fußgängerampel dar, die während der

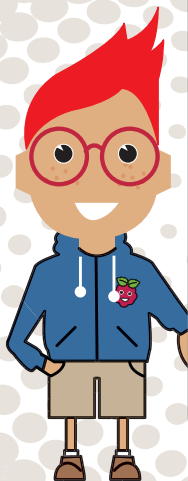
Rotphase der Verkehrsampel eine Grünphase für Fußgänger anzeigt.

Dieses Experiment zeigt außerdem, wie du den Nano mit selbstgebauten Kontakten aus Knete steuern kannst. Probiere dieses Experiment in aller Ruhe aus, bevor du dich an die komplizierteren Versuche heranwagst. Denn hier lernst du eine Menge Tricks, die du später noch brauchen wirst.

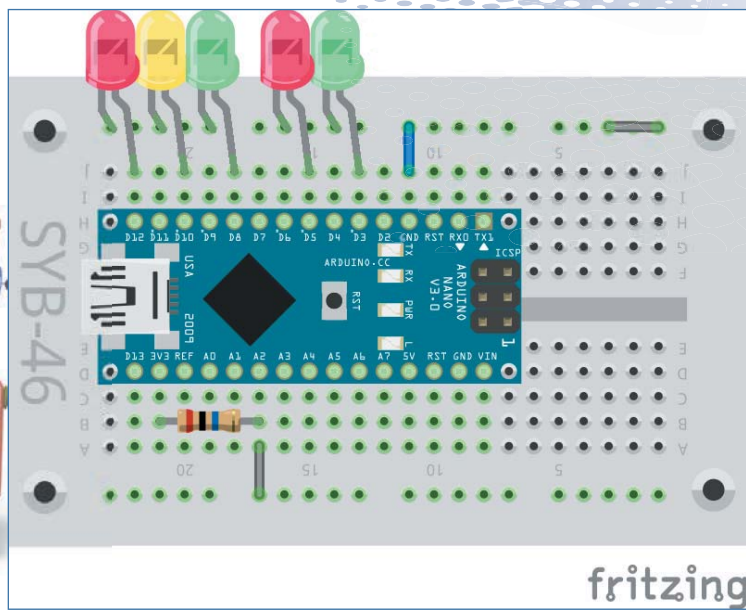
Baue zunächst die abgebildete Schaltung auf einem Steckbrett auf. Eine der Kontaktleisten an den langen Seiten des Steckbretts ist wie beim letzten Mal mit der Masseleitung des Nano verbunden, die andere Seite verwenden wir für den Knetekontakt.

BAUTEILE

- 1x Nano
- 1x Steckbrett
- 2x LED rot (mit Vorwiderstand)
- 1x LED gelb (mit Vorwiderstand)
- 2x LED grün (mit Vorwiderstand)
- 1x 20-MO Ω -Widerstand (rot-schwarz-blau)
- 2x blanke Drahtbrücke
- 1x isolierte Drahtbrücke
- 2x Knetekontakt



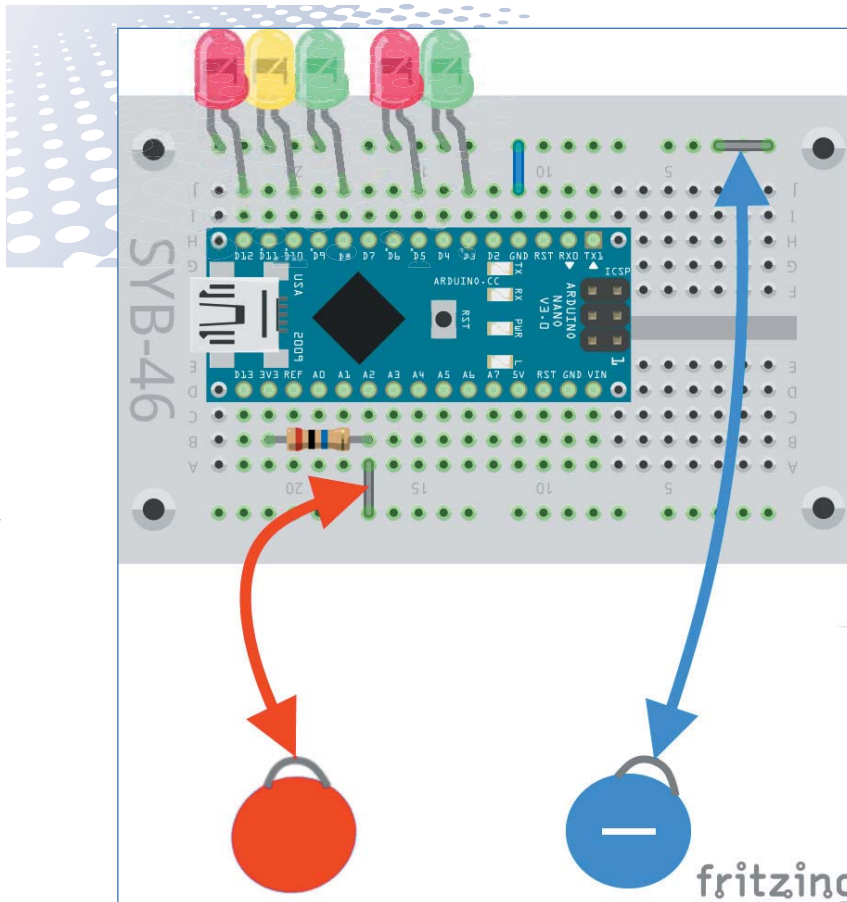
Der 20-MO Ω -Widerstand zwischen 3,3 V und dem Pin A2 ist für den Knetekontakt. Wir verwenden in diesem Experiment zwei Knetekontakte: Einer kommt an die Masseleitung, der andere an den analogen Pin A2.



KABELFARBEN

Üblicherweise werden in der Elektrotechnik rote Kabel für die Verbindung mit dem Pluspol der Schaltung und schwarze oder blaue Kabel für die Verbindung mit der Masseleitung verwendet. Um leichter erklären zu können, welcher Knetekontakt gemeint ist, verwenden wir in der Abbildung dementsprechend rote und blaue Knete. Der Massekontakt ist zusätzlich mit einem Minuszeichen gekennzeichnet.

4 Fußgängerampel



Beim Berühren des Knetkontakts startet der typische Ampelzyklus einer Fußgängerampel. Im Ruhezustand leuchtet die Fußgängerampel rot, die Verkehrsampel grün.

HIGH ODER LOW?

Ein digitaler Pin, der vom Programm ein- oder ausgeschaltet werden kann, wird wie im ersten Experiment als Ausgang bezeichnet, da der Nano hier ein Signal ausgibt, nämlich



falsch für ausgeschaltet oder **wahr** für eingeschaltet.

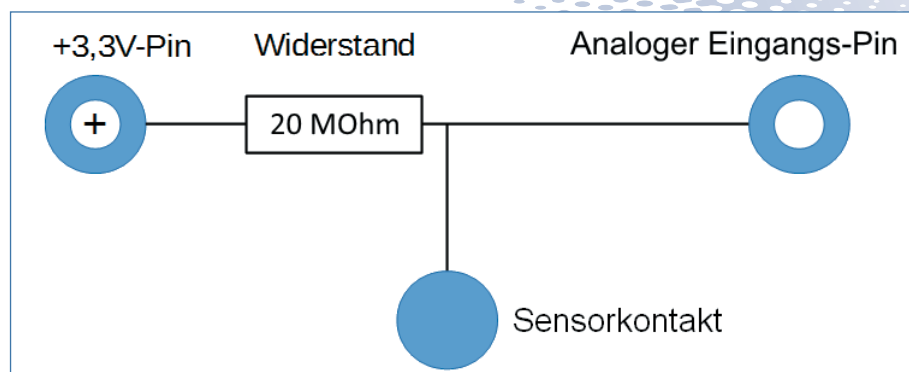
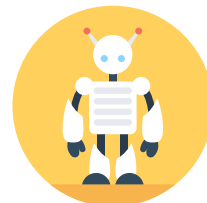
Umgekehrt kann ein digitaler Pin auch als Eingang benutzt werden, der dem Programm ein **falsch** oder **wahr** bzw. eine 0 oder eine 1 schickt. Dies hängt davon ab, welches Signal am Eingang anliegt. Man unterscheidet in der Digitalelektronik zwischen **Low**-Signalen (= niedrig, falsch, 0) und **High**-Signalen (= hoch, wahr, 1). Dabei gilt:

- **Low**: Der Eingang ist mit der Masse verbunden.
- **High**: Am Eingang liegt eine positive Spannung an.

DER TRICK MIT DEN KNETEKONTAKTEN

Das Prinzip der Knetkontakte ist einfach: Der verwendete Pin ist über einen extrem hohen Widerstand (20 MOhm) mit +3,3 V verbunden, sodass ein schwaches, aber eindeutig als **High** definiertes Signal am Pin anliegt.

Wenn du mit einer Hand den blauen Knetkontakt an der Masseleitung anfasst, ist dein Körper „geerdet“, also mit der Masse verbunden.

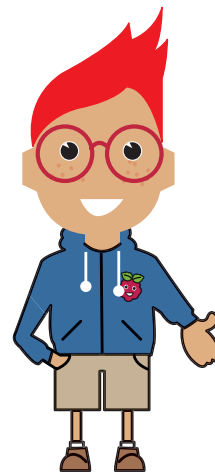


4 Fußgängerampel



ACHTUNG!

Berühre nie eine Steckdose oder ein nicht isoliertes Kabel, das am 230-V-Stromnetz angeschlossen ist! Auch diesen Strom würdest du, da du über deine Füße immer mit der Masseleitung Erde verbunden bist, direkt dorthin leiten – und das ist lebensgefährlich! Das wäre vergleichbar mit einem Blitzeinschlag in deinem Körper. Ein Gewitter ist nichts anderes als ein Kurzschluss zwischen positiv geladenen Wolken und der Erde, der durch die feuchte Luft herbeigeführt wird.

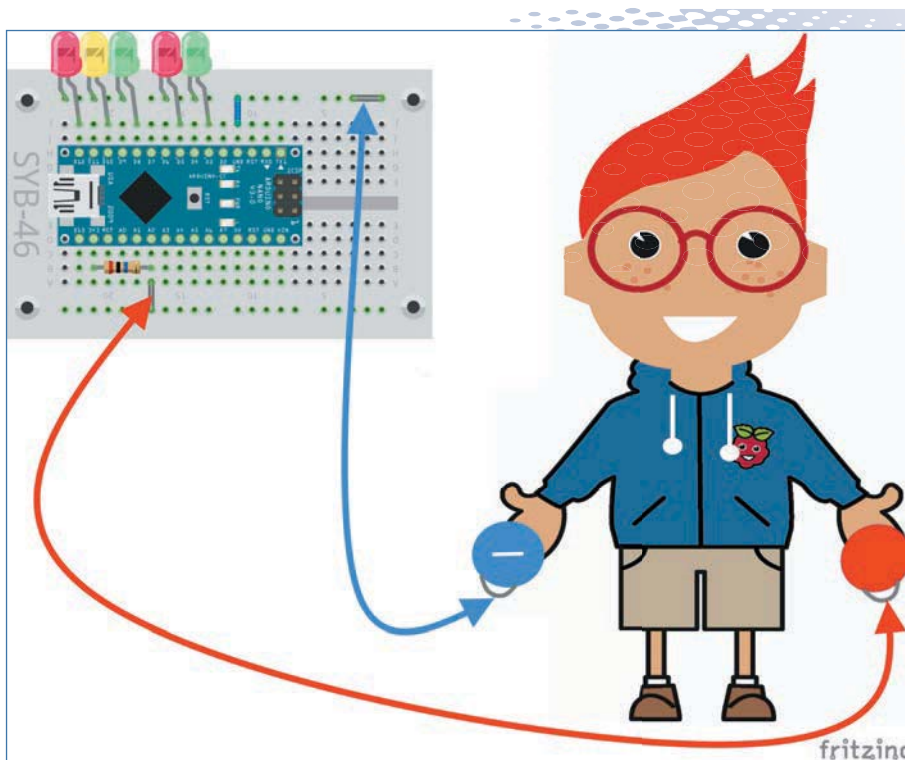




Fasst du gleichzeitig mit der anderen Hand den roten Knetkontakt am Pin A2 an, wird das schwache **High**-Signal dort von dem deutlich stärkeren **Low**-Signal der Hand überlagert und zieht den entsprechenden Pin auf **Low**.

ES GEHT LOS

Nachdem du die Schaltung aufgebaut und die Knetkontakte angeschlossen hast, starte das Programm **02ampel** und klicke auf das grüne Fähnchen.



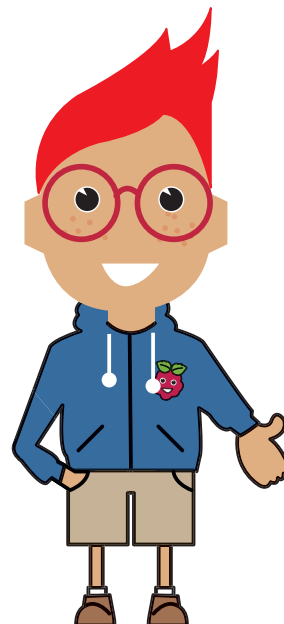
4 Fußgängerampel



```
Wenn angeklickt
  Setze digitalen Pin 12 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 10 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 5 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 3 auf falsch
  warte 0.01 Sek.
  fortlaufend
  falls lies analogen Pin 2 < 200
    Setze digitalen Pin 8 auf falsch
    warte 0.01 Sek.
    Setze digitalen Pin 10 auf wahr
    warte 0.5 Sek.
    Setze digitalen Pin 10 auf falsch
    warte 0.01 Sek.
    Setze digitalen Pin 12 auf wahr
    warte 0.5 Sek.
    Setze digitalen Pin 5 auf falsch
    warte 0.01 Sek.
    Setze digitalen Pin 3 auf wahr
    warte 3 Sek.
    Setze digitalen Pin 3 auf falsch
    warte 0.01 Sek.
    Setze digitalen Pin 5 auf wahr
    warte 0.5 Sek.
    Setze digitalen Pin 10 auf wahr
    warte 0.5 Sek.
    Setze digitalen Pin 12 auf falsch
    warte 0.01 Sek.
    Setze digitalen Pin 10 auf falsch
    warte 0.01 Sek.
    Setze digitalen Pin 8 auf wahr
    warte 3 Sek.
```

Klicke auf das grüne Fähnchen, um das Programm zu starten. Die Verkehrsampel leuchtet grün, die Fußgängerampel rot – genau so, wie es echte Ampeln stundenlang tun, solange kein Fußgänger kommt und auf den Knopf drückt.

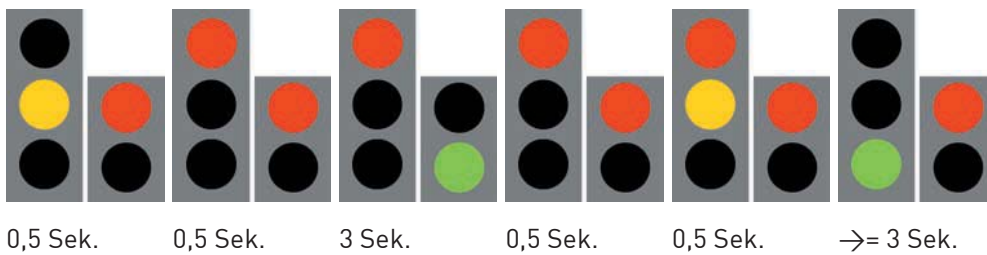
Nimm den blauen Knetekontakt an der Masseleitung in die Hand. Berühre zusätzlich den roten Knetekontakt. Jetzt startet der Ampelzyklus, der in unserem Programm wie auch bei einer echten Ampel aus sechs unterschiedlichen Lichtmus-





tern besteht, die unterschiedlich lange leuchten.

müssen die Autos auch einmal fahren dürfen. In unserer Modellampel sind

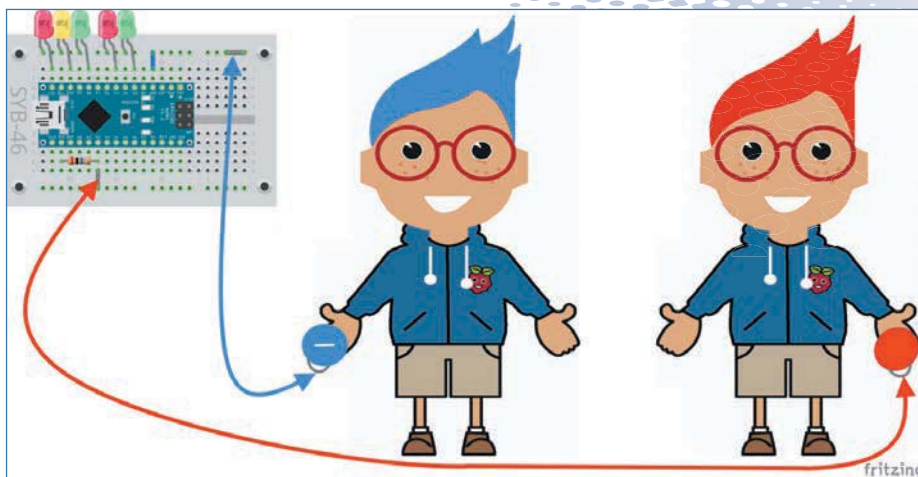


Mit dem letzten Lichtmuster – Fußgänger rot, Verkehrsampel grün – erreicht die Ampel wieder den Standardzustand. Das Programm muss allerdings dafür sorgen, dass auch dieser immer für eine Mindestzeit eingehalten wird. Selbst wenn ständig Fußgänger auf den Knopf drücken,

das 3 Sekunden, bei einer wirklichen Ampel natürlich deutlich mehr.

ZWEI PERSONEN

Du kannst den Stromkreis auch über mehrere Personen schließen. Nimm



4 Fußgängerampel

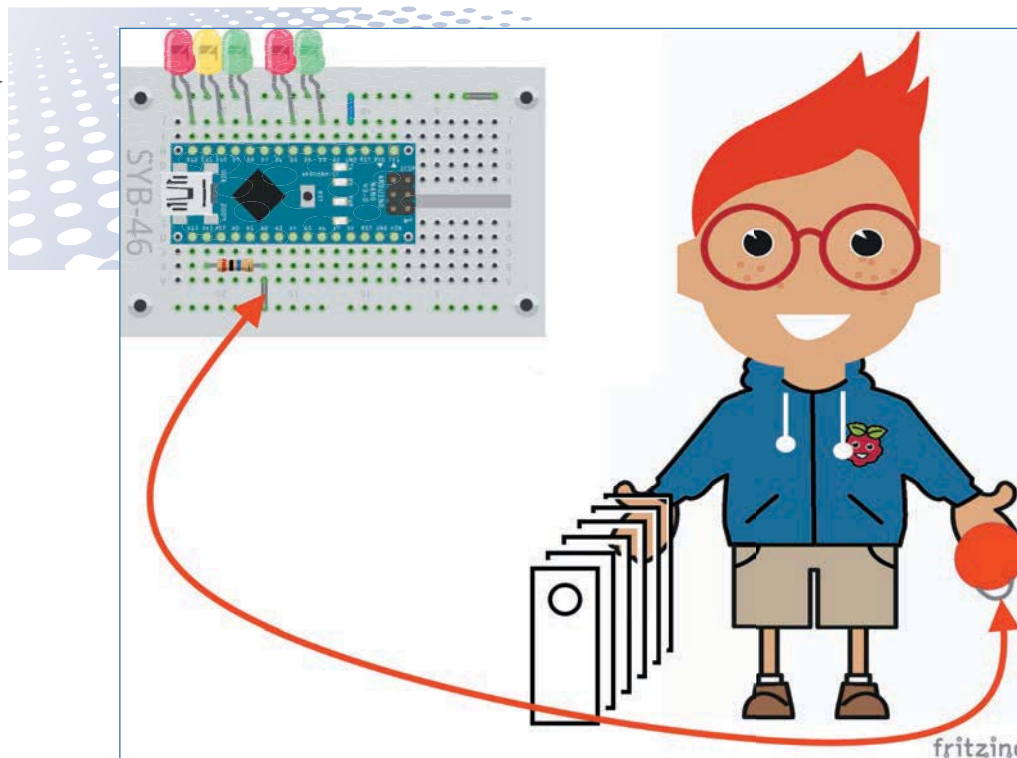


den blauen Knetekontakt in eine Hand und bitte eine andere Person, den roten Knetekontakt in die Hand zu nehmen. Noch passiert gar nichts.

Wenn ihr euch an die Hand nehmt, startet der Ampelzyklus. Der Stromkreis ist über eure Hände geschlossen.

ES GEHT AUCH OHNE MASSEKONTAKT

Nimm einmal nur den roten Knetekontakt in die Hand, ohne über den blauen Knetekontakt mit der Masseleitung verbunden zu sein. Bewege dich jetzt etwas, stehe auf, setze dich hin oder berühre mit der anderen Hand ein Möbelstück. Die Ampel wird immer mal wieder starten.





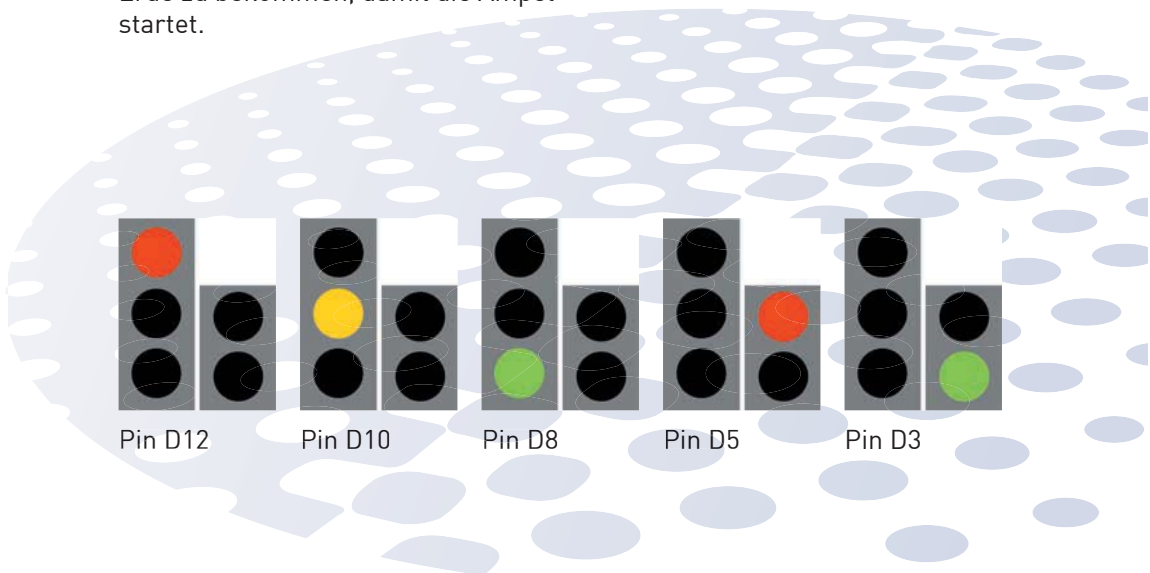
Das bedeutet, dass der Nano-Pin mit der Masse verbunden ist – aber wie? Ein Mensch ist über seine Füße fast immer mit der Masse der Erde verbunden. Wie hoch allerdings der Widerstand zwischen deiner Hand und der Erde ist, hängt von vielen Dingen ab, vor allem davon, was du für Schuhe anhast und auf welchem Fußboden du stehst. Barfuß im nassen Gras ist die Verbindung zur Masse der Erde am besten, aber auch auf Steinfußboden funktioniert es meistens gut. Du kannst auch mit der freien Hand ein geerdetes Metallteil wie zum Beispiel einen Heizkörper oder einen Wasserhahn berühren, um guten Kontakt zur Erde zu bekommen, damit die Ampel startet.

SO FUNKTIONIERT DAS PROGRAMM

Wenn du auf das grüne Fähnchen klickst, wird die Grundstellung der Ampel eingeschaltet: Grün für die Autos, Rot für die Fußgänger. Die anderen drei LEDs werden ausgeschaltet. Das Ausschalten ist hier am Anfang eigentlich nicht nötig. Es dient nur dazu, das Programm auf jeden Fall mit einem klar definierten Zustand zu starten.



Die Abbildung zeigt, welcher Pin welche LED steuert.

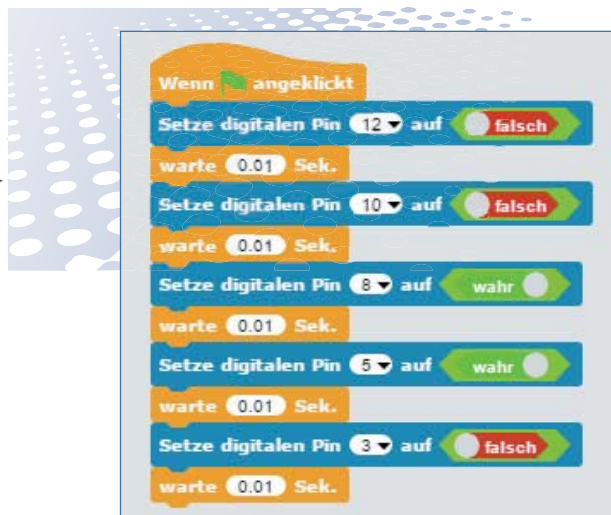


4 Fußgängerampel



Jetzt beginnt wie im letzten Programm eine **fortlaufend**-Schleife. Innerhalb der Schleife wird in einem **falls**-Block der Knetekontakt abgefragt. Die Blöcke innerhalb des **falls**-Blocks werden immer nur dann ausgeführt, wenn die Abfrage im Titelbalken dieses Blocks ein wahres Ergebnis liefert.

Ist das Ergebnis der Abfrage hingegen falsch, passiert gar nichts, und die Blöcke unterhalb des **falls**-Blocks werden ausgeführt. In unserem Fall startet dann die **fortlaufend**-Schleife neu.



Für die Abfrage selbst ist im **falls**-Block ein längliches Feld mit spitzen Enden vorgesehen. Hier muss ein Block aus der grünen Blockpalette **Operatoren** eingefügt werden. Ziehe den Block mit dem Kleiner-als-Zeichen auf das Platzhalterfeld im Block **falls**.



Dieser Operator ist immer dann wahr, wenn der Wert links vom Zeichen kleiner ist als der Wert rechts davon.

Ziehe also in das linke Feld des **Kleiner-als**-Operators einen Block **lies analogen Pin...** von der Blockpalette

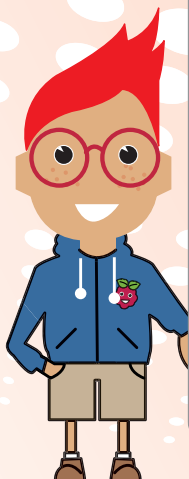




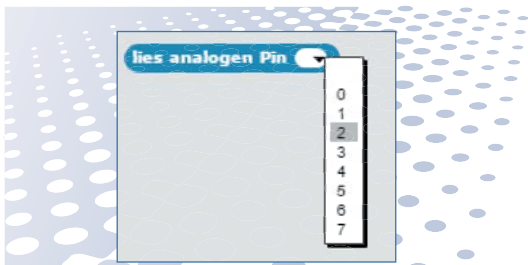
ANALOGE UND DIGITALE NANO-PINS

Da Snap4Arduino die im Nano eingebauten Pulldown-Widerstände immer einschaltet, werden digitale Eingänge immer auf 0 gezogen. Sie haben auch ohne Berührung einen Low-Pegel. Deshalb nutzen wir zur Abfrage der Knetkontakte einen analogen Eingangspin. Diese liefern Zahlenwerte zwischen 0 und 1023, je nach anliegendem Eingangspegel. Bei niedrigen Werten liegt der Eingangspegel näher an dem Massewert von 0 V, hohe Werte stehen für Eingangspegel, die dem logischen High, also +3,3 V, näher kommen. Auch die analogen Eingänge dürfen nicht mit höheren Spannungen als +3,3 V angesteuert werden. Ein Wert von 200 hat sich als guter Grenzwert zwischen berührtem und nicht berührtem Knetkontakt erwiesen.

Die analogen Pins befinden sich auf der den digitalen Pins gegenüberliegenden Seite der Platine und werden dort mit **A0...A5** bezeichnet. Im Block *lies analogen Pin...* werden nur die Nummern **0** bis **7** angegeben. Analoge Pins können nur als Eingänge, nicht als Ausgänge verwendet werden.



Arduino und wähle im Listenfeld den **Pin 2** aus.



Schreibe in das rechte Feld des **Kleiner-als**-Operators den Wert **200** und ziehe dann den ganzen **Kleiner-als**-Operator in das Feld im Titel des **falls...**-Blocks.



4 Fußgängerampel



```
fortlaufend
falls lies analogen Pin 2 < 200
  Setze digitalen Pin 8 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 10 auf wahr
  warte 0.5 Sek.
  Setze digitalen Pin 10 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 12 auf wahr
  warte 0.5 Sek.
  Setze digitalen Pin 5 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 3 auf wahr
  warte 3 Sek.
  Setze digitalen Pin 3 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 5 auf wahr
  warte 0.5 Sek.
  Setze digitalen Pin 10 auf wahr
  warte 0.5 Sek.
  Setze digitalen Pin 12 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 10 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf wahr
  warte 3 Sek.
```

Wenn diese Abfrage **wahr** ergibt, der analoge Pin also einen Wert kleiner als 200 liefert, weil der Knetkontakt berührt wird, werden nacheinander die unterschiedlichen Lichtmuster der Ampel geschaltet. Die Ampel wartet jedes Mal 0,5 Sekunden bis zum nächsten Lichtmuster. Während die Fußgängerampel grün leuchtet, wird 3 Sekunden gewartet. Werden zwei LEDs scheinbar gleichzeitig geschaltet, ist wie im letzten Experiment eine Auszeit von einer Hundertstelsekunde im Programm eingebaut.





WENN ES NICHT FUNKTIONIERT

In manchen Fällen kann es passieren, dass die Ampel angeht, auch wenn du gar keinen der Knetkontakte berührst. Dies kann verschiedene Ursachen haben, zum Beispiel, dass die Tischplatte oder eine andere Unterlage, auf der die Knetkontakte liegen, leitfähig ist und so den Stromkreis schließt. Wenn du den roten Knetkontakt am isolierten Kabel hochhebst, ohne die Knete oder eine blanke Drahtstelle zu berühren, sollte alles wieder ruhig sein.

Selbst geringe statische Elektrizitätsfelder, wie sie zum Beispiel durch kunststoffbeschichtete Tischplatten oder Teppichböden entstehen, können schon zum Kontakt führen.

In seltenen Fällen kann es vorkommen, dass die Ampel angeht, selbst wenn der rote Knetkontakt frei in der Luft hängt oder du ihn nicht einmal angeschlossen hast. Das kann durch Störfelder – oft auch als Elektromog bezeichnet – passieren. Solche elektrischen Felder entstehen durch Handys, Computer, Fernseher, Mikrowellen und viele andere elektrische Geräte in der unmittelbaren Umgebung. Sie können bei bestimmten Wetterlagen aber auch auf natürliche Weise entstehen. Sogar besondere Gesteinsformationen im Erdboden unter dir können elektromagnetische Unregelmäßigkeiten verursachen, die sich bei den geringen Strömen, mit denen unsere Knetkontakte arbeiten, schon bemerkbar machen.

Sollte das Programm nicht zuverlässig funktionieren, probiere als Grenzwerte zwischen **berührt** und **nicht berührt** statt des vorgegebenen Werts **200** andere Werte zwischen etwa **100** und **300** aus.



5 LED-Würfel



Typische Spielwürfel mit einem bis sechs Augen kennt jeder, und vermutlich hat auch jeder welche zu Hause. Wesentlich cooler ist ein elektronisch gesteuerter Würfel, der mit einer Steuerung über einen Knetkontakt die Augen aufleuchten lässt – aber nicht einfach ein bis sechs LEDs in einer Reihe, sondern so, wie die Augen auf dem Spielwürfel angeordnet sind.

Spielwürfel haben Augen in der typischen quadratischen Anordnung, wozu man sieben LEDs braucht, die in vier Gruppen gesteuert werden. Für die Ansteuerung der LEDs brau-

chen wir nur vier statt sieben Pins, da Würfel zur Darstellung gerader Zahlen die Augen paarweise nutzen. Die mittlere LED ist die, die fest auf der Nano-Platine verbaut und mit dem Pin 13 verbunden ist. Somit brauchen wir sogar nur drei externe Pins.

Baue die Schaltung aus sechs LEDs mit eingebauten Vorwiderständen und zwei Knetkontakten auf. In der Schaltung werden beide seitlichen Schienen des Steckbretts für Masseleitungen benötigt. Daher sind die Knetkontakte diesmal etwas anders angeschlossen.





Würfelzahl	Pin 13	Pin 5	Pin 8	Pin 2
1	Blue			
2		Red		
3	Blue		Green	
4		Red	Green	
5	Blue	Red	Green	
6		Red	Green	Yellow

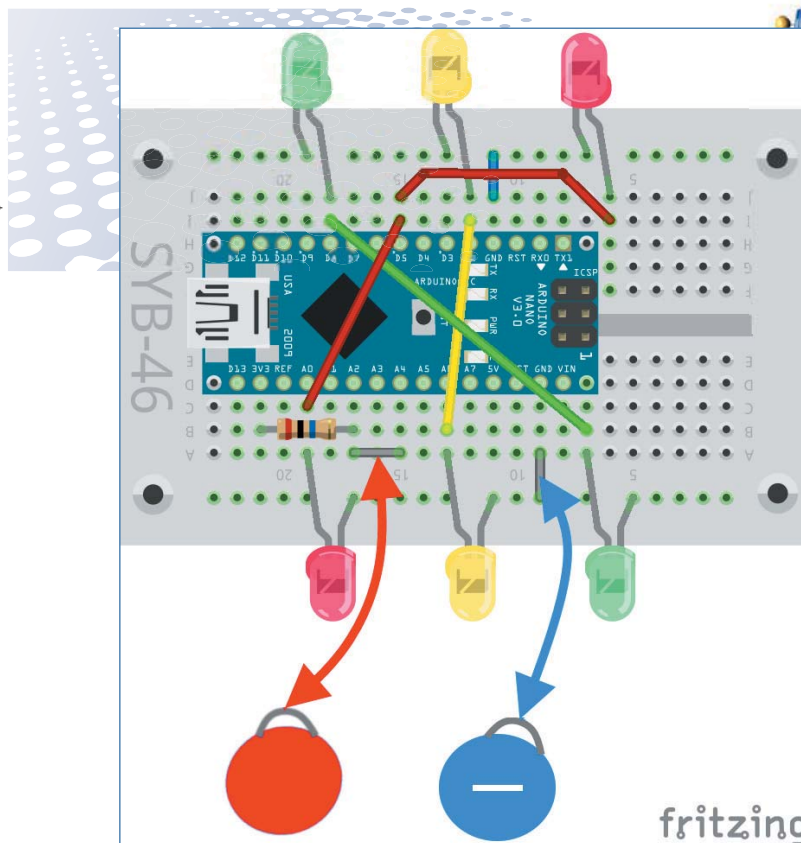


5 LED-Würfel



Die Schaltung sieht auf den ersten Blick deutlich komplizierter aus als die vorherigen. Immerhin werden neben dem Nano noch sieben

weitere elektronische Bauteile und zusätzlich diverse Drähte auf dem Steckbrett aufgebaut.





Nachdem du die Schaltung aufgebaut und die Knetekontakte angeschlossen hast, starte das Programm **03wuerfel01**.

```

Wenn angeklickt
fortlaufend
falls Les analogen Pin 2 < 200
  warte 0.01 Sek.
  Setze digitalen Pin 2 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 3 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 13 auf falsch
  warte 0.1 Sek.
  setze w auf Zufallszahl von 1 bis 6
  falls w = 1
    Setze digitalen Pin 13 auf wahr
  falls w = 2
    Setze digitalen Pin 8 auf wahr
  falls w = 3
    Setze digitalen Pin 5 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 13 auf wahr
  falls w = 4
    Setze digitalen Pin 5 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 13 auf wahr
  falls w = 5
    Setze digitalen Pin 5 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 13 auf wahr
  falls w = 6
    Setze digitalen Pin 2 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 5 auf wahr
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf wahr
  
```

5 LED-Würfel



Alle LEDs sind am Anfang aus. Berühre nun kurz die beiden Knetkontakte, um zu würfeln. Wenn du wieder loslässt, wird die gewürfelte Zahl so lange angezeigt, bis du die Knetkontakte wieder berührst.

Endlosschleife, die darauf wartet, dass du den Knetkontakt berührst.

Wenn der analoge Eingang einen Wert kleiner als **200** hat, wird der Knetkontakt berührt und die Anweisungen innerhalb des **falls**-Blocks werden ausgeführt. Am Anfang sind die LEDs alle ausgeschaltet, aber später bleibt ein angezeigtes Würfelergebnis so lange bestehen, bis du den Knetkontakt wieder berührst.

SO FUNKTIONIERT DAS PROGRAMM

Wenn du auf das grüne Fähnchen klickst, startet sofort die

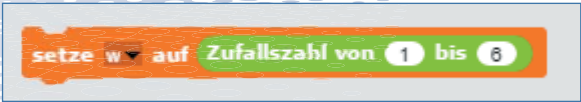


```
Wenn  angeklickt
fortlaufend
falls lies analogen Pin 2 < 200
  warte 0.01 Sek.
  Setze digitalen Pin 2 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 5 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 13 auf falsch
  warte 0.1 Sek.
```



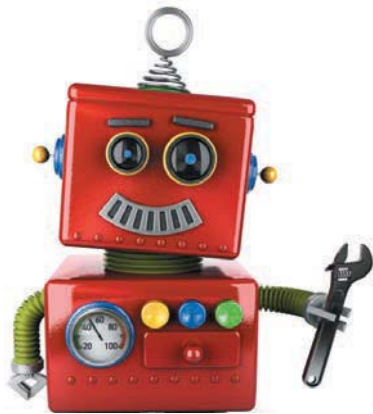
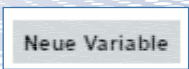
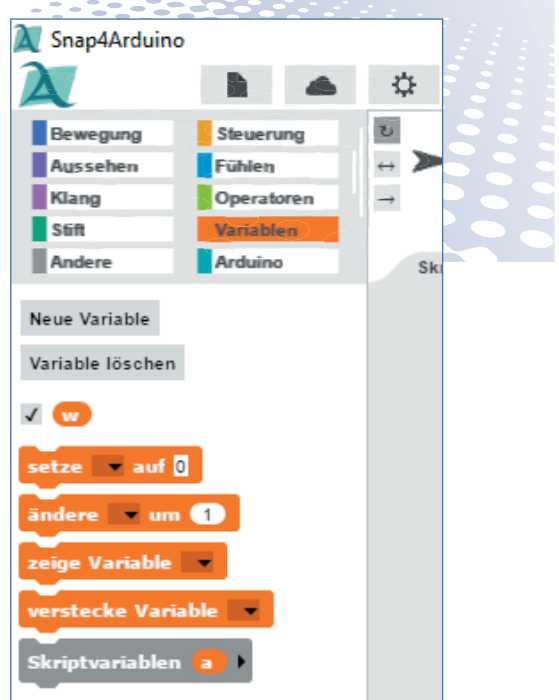
Danach wird eine zufällige Zahl zwischen 1 und 6 erzeugt und in der Variablen **w** gespeichert. Variablen sind kleine Speicherplätze, in denen man sich, während ein Programm

Gib dann der Variablen einen Namen, wir verwenden einfach **w** (wie Würfel). In der Blockpalette werden verschiedene Blöcke zur Arbeit mit Variablen angezeigt.



läuft, eine Zahl oder irgendetwas anderes merken kann. Wenn das Programm beendet wird, werden die Variablenspeicher automatisch geleert.

Variablen müssen in Snap4Arduino erst einmal angelegt werden, bevor man sie benutzen kann. Klicke in der Blockpalette oben auf das orangefarbene Symbol **Variablen** und dann auf **Neue Variable**.



5 LED-Würfel

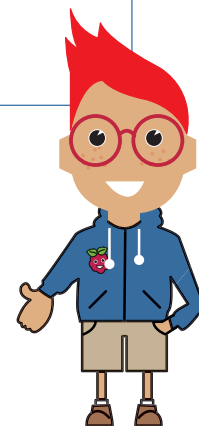
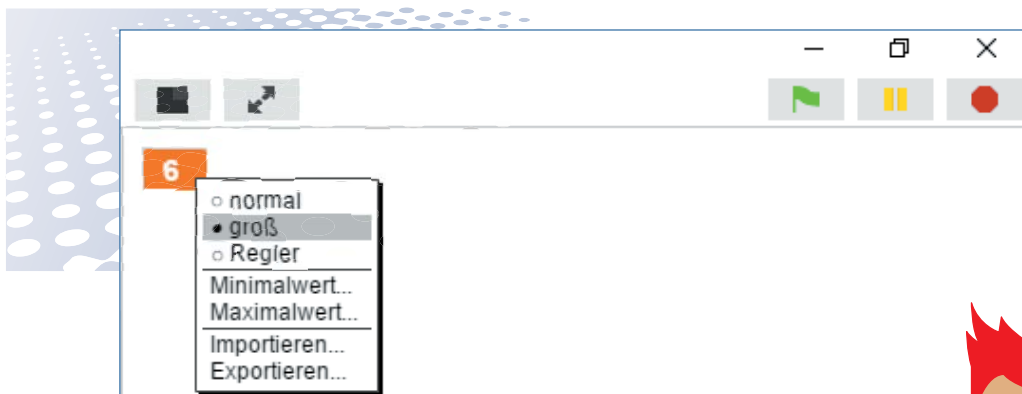


Wenn du den Schalter links neben der Variable **w** einschaltest, wird diese Variable automatisch auf der Bühne in einem kleinen orange-farbenen Feld angezeigt. So siehst du immer die gewürfelte Zahl und kannst leicht kontrollieren, ob die LEDs funktionieren. Dieses Zahlenfeld ist sehr klein. Klicke mit der rechten Maustaste darauf und wähle aus dem Menü **groß**. Jetzt ist die Zahl besser zu erkennen.

Ziehe nun den Block **setze...auf** in das Programmfenster. Wähle im Listenfeld die Variable **w** aus.



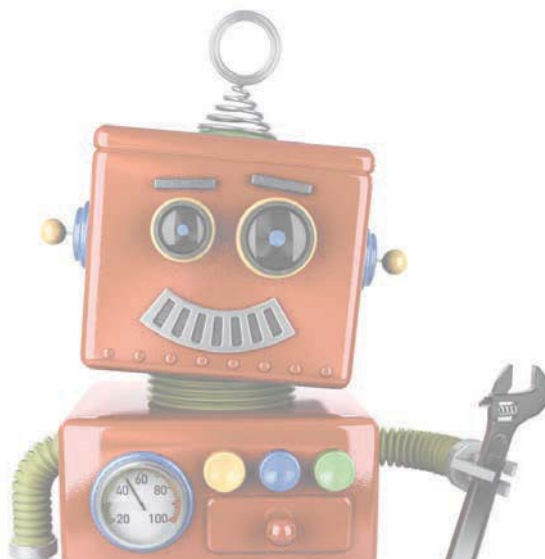
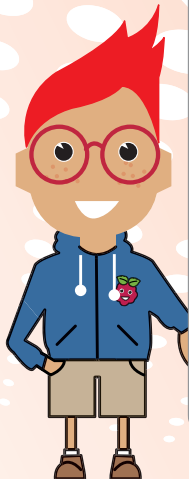
Ziehe dann aus der grünen Blockpalette der Operatoren den Block **Zufallszahl von...bis...** auf das Zahlenfeld im orangefarbenen Block **setze...auf**. Trage in die beiden Zahlenfelder eine **1** und eine **6** ein, da die Zufallszahl in diesem Bereich liegen soll.





WIE ENTSTEHEN EIGENTLICH ZUFALLSZAHLEN?

Man denkt vielleicht, in einem Programm könne nichts zufällig passieren, alles sei geplant. Wie kann ein Programm dann in der Lage sein, zufällige Zahlen zu generieren? Teilt man eine große Primzahl durch irgendeinen Wert, ergeben sich ab der x-ten Nachkommastelle Zahlen, die kaum noch vorhersehbar sind und sich auch ohne jede Regelmäßigkeit ändern, wenn man den Divisor regelmäßig erhöht. Dieses Ergebnis ist zwar scheinbar zufällig, lässt sich aber durch ein identisches Programm oder mehrfachen Aufruf des gleichen Programms jederzeit reproduzieren. Nimmt man aber eine aus einigen dieser Ziffern zusammengebaute Zahl und teilt sie wiederum durch eine Zahl, die sich aus der aktuellen Uhrzeitsekunde oder dem Inhalt einer beliebigen Speicherstelle des Rechners ergibt, kommt ein Ergebnis heraus, das sich nicht reproduzieren lässt. Aus diesem Grund wird es als Zufallszahl bezeichnet.

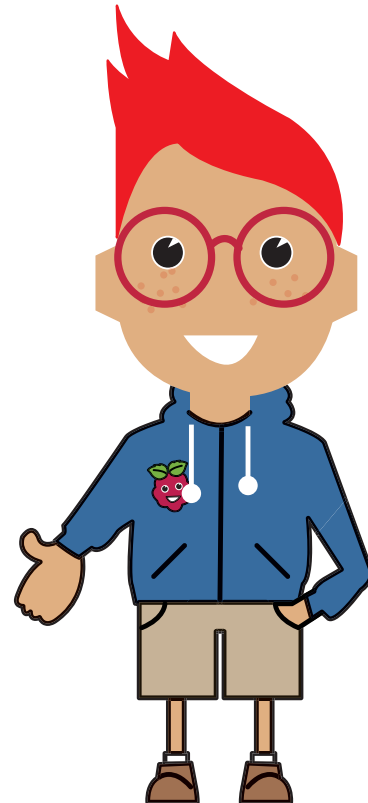


5 LED-Würfel



Baue den Block in das Programm ein. Dieses sollte nun so aussehen:

```
Wenn angeklickt  
fortlaufend  
falls lies analogen Pin 2 < 200  
  warte 0.01 Sek.  
  Setze digitalen Pin 2 auf falsch  
  warte 0.01 Sek.  
  Setze digitalen Pin 5 auf falsch  
  warte 0.01 Sek.  
  Setze digitalen Pin 8 auf falsch  
  warte 0.01 Sek.  
  Setze digitalen Pin 13 auf falsch  
  warte 0.1 Sek.  
  setze w auf Zufallszahl von 1 bis 6
```



Nachdem die Zahl gewürfelt wurde, folgen sechs *falls*-Blöcke für jeden möglichen Würfelwert.

```
falls w = 1  
  Setze digitalen Pin 13 auf wahr
```



Jeder dieser Blöcke schaltet, wenn eine bestimmte Zahl gewürfelt wurde, die entsprechende Kombination von LEDs ein.



Ziehe den grünen **=**-Block in das Abfragefeld des **falls**-Blocks.



Ziehe dann den Block der Variable **w** aus der Blockpalette **Variablen** in das erste der beiden kleinen weißen Felder im grünen Block. Schreibe in das zweite Feld eine **1**.

Jetzt wird der Block innerhalb der Klammer immer dann abgearbeitet, wenn das Würfelergebnis eine **1** ist. Setze innerhalb des **falls**-Blocks einen Block **setze digitalen Pin... auf wahr** und wähle dort den Pin 13 aus, um die mittlere LED einzuschalten – die 1 auf dem Würfel.

Mit einem Rechtsklick auf den **falls**-Block kannst du diesen duplizieren. Du brauchst nur noch die Würfelergebnisse und die dazu passenden LEDs zu ändern. Werden mehrere LEDs eingeschaltet, setze dazwischen eine kurze Wartezeit von 0,01 Sekunden. Nach der letzten eingeschalteten LED ist keine Wartezeit nötig, da die Endlosschleife automatisch wartet, bis du wieder den Knetekontakt berührst.



Nach den sechs **falls**-Abfragen startet die Endlosschleife einen neuen Durchlauf. Je nach ermitteltem Sensorwert wird die Würfelaktion gestartet oder nicht. Solange der Knetekontakt nicht berührt wird, hat der digitale Pin einen deutlich größeren Wert als **200**, und die Endlosschleife zeigt weiterhin einfach nur das zuletzt gewürfelte Ergebnis an.

5 LED-Würfel



LED-WÜRFEL MIT ECHTEM WÜRFELEFFEKT

Ein wirklicher Würfel zeigt nicht sofort die endgültige Augenzahl an, sondern rollt noch eine kurze Zeit, in der man Würfelergebnisse sieht, die sich dann doch nicht bewahrheiten. Das Programm *03wuerfel02* simuliert das Rollen, indem der Würfel erst ein paar andere Würfelergebnisse mit immer längeren kurzen Pausen dazwischen anzeigt, bevor das endgültige Ergebnis erscheint.



```
Wenn ... angeklickt
fortlaufend
falls Les analogen Pin < 200
setze n auf 0.1
wiederhole 4 mal
  Setze digitalen Pin 2 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 5 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 8 auf falsch
  warte 0.01 Sek.
  Setze digitalen Pin 13 auf falsch
  warte 0.1 Sek.
  setze w auf Zufallszahl von 1 bis 6
  falls w = 1
    Setze digitalen Pin 13 auf wahr
  falls w = 2
    Setze digitalen Pin 8 auf wahr
  falls w = 3
    Setze digitalen Pin 5 auf wahr
    warte 0.01 Sek.
    Setze digitalen Pin 13 auf wahr
  falls w = 4
    Setze digitalen Pin 5 auf wahr
    warte 0.01 Sek.
    Setze digitalen Pin 8 auf wahr
  falls w = 5
    Setze digitalen Pin 5 auf wahr
    warte 0.01 Sek.
    Setze digitalen Pin 8 auf wahr
    warte 0.01 Sek.
    Setze digitalen Pin 13 auf wahr
  falls w = 6
    Setze digitalen Pin 2 auf wahr
    warte 0.01 Sek.
    Setze digitalen Pin 5 auf wahr
    warte 0.01 Sek.
    Setze digitalen Pin 8 auf wahr
  warte n Sek.
  ändere n um 0.2
```



Das Programm nutzt den gleichen Würfelmechanismus, würfelt aber nach dem Berühren des Knetkontakts nicht nur einmal, sondern viermal hintereinander, wobei die Wartezeiten vor dem Löschen des Ergebnisses jedes Mal um 0,2 Sekunden länger werden.

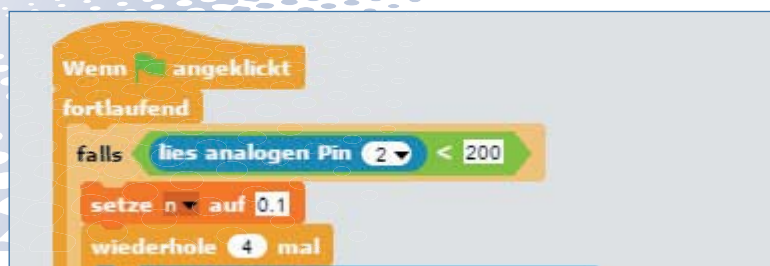
Am Ende jedes Durchlaufs wartet das Programm für die in der Variablen *n* gespeicherte Zeit, bevor der nächste Schleifendurchlauf wieder würfelt. Nach der Wartezeit wird die nächste Wartezeit um 0,2 Sekunden verlängert.

Dazu wird zu Beginn direkt nach dem Berühren des



Knetkontakts eine neue Variable *n* auf 0,1 Sekunden gesetzt. Dies ist die Wartezeit zwischen den beiden ersten Ergebnissen. Anschließend startet eine *wiederhole...mal*-Schleife, die viermal läuft und den bereits bekannten Würfelmechanismus enthält.

Der Block *ändere... um ...* erhöht den Wert einer Variablen um einen bestimmten Wert, ohne dass extra eine Berechnung durchgeführt werden muss. Gibt man im Zahlenfeld eine negative Zahl an, wird der Wert der Variablen entsprechend verringert.



6 LED dimmen



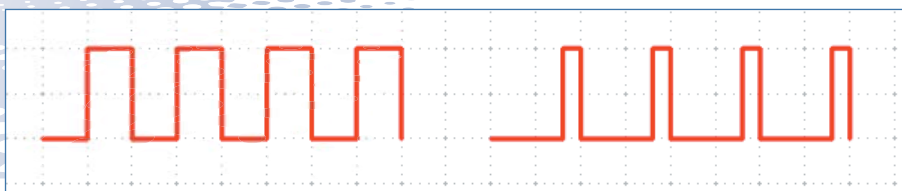
LEDs sind typische Bauteile zur Ausgabe von Signalen in der Digital-elektronik. Sie können zwei verschiedene Zustände annehmen, nämlich ein und aus, 1 und 0 oder **HIGH** und **LOW**. Das Gleiche gilt für die als Ausgänge definierten digitalen Pins. Demnach wäre es theoretisch nicht möglich, eine LED zu dimmen.

Mit einem Trick ist es dennoch möglich, die Helligkeit einer LED an einem digitalen Pin zu regeln. Lässt man eine LED schnell genug blinken, nimmt das

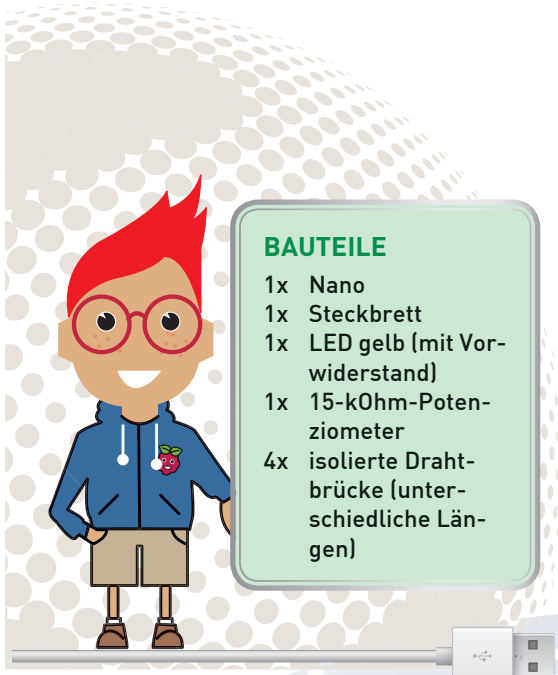
menschliche Auge das nicht mehr als Blinken wahr. Diese als Pulsweitenmodulation (PWM) bezeichnete Technik erzeugt ein pulsieren-

PINS FÜR PWM-SIGNALE

Nur die Pins 3, 5, 6, 9, 10 und 11 auf dem Nano können für Pulsweitenmodulation verwendet werden.



Links: Tastverhältnis 50 %, rechts: Tastverhältnis 20 %



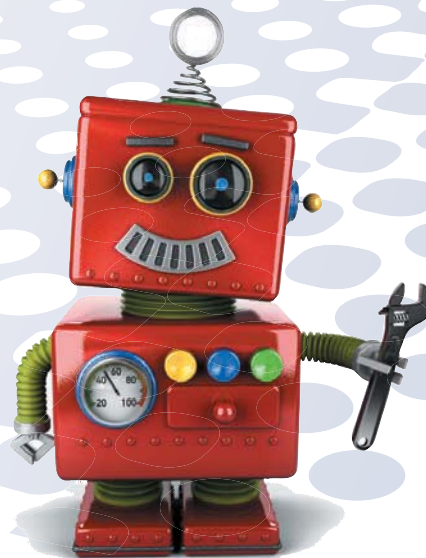
BAUTEILE

- 1x Nano
- 1x Steckbrett
- 1x LED gelb (mit Vorwiderstand)
- 1x 15-kOhm-Potenzimeter
- 4x isolierte Drahtbrücke (unterschiedliche Längen)



des Signal, das sich in sehr kurzen Abständen ein- und ausschaltet. Die Spannung des Signals bleibt immer gleich, nur das Verhältnis zwischen Level **LOW** (0 V) und Level **HIGH** (+3,3 V) wird verändert. Das sogenannte Tastverhältnis gibt die Dauer des eingeschalteten Zustands im Verhältnis zur Gesamtdauer eines Schaltzyklus an.

Je kleiner das Tastverhältnis, desto kürzer ist die Leuchtzeit der LED im Verhältnis zur gesamten Zeit. Dadurch wirkt die LED dunkler als eine permanent eingeschaltete LED.

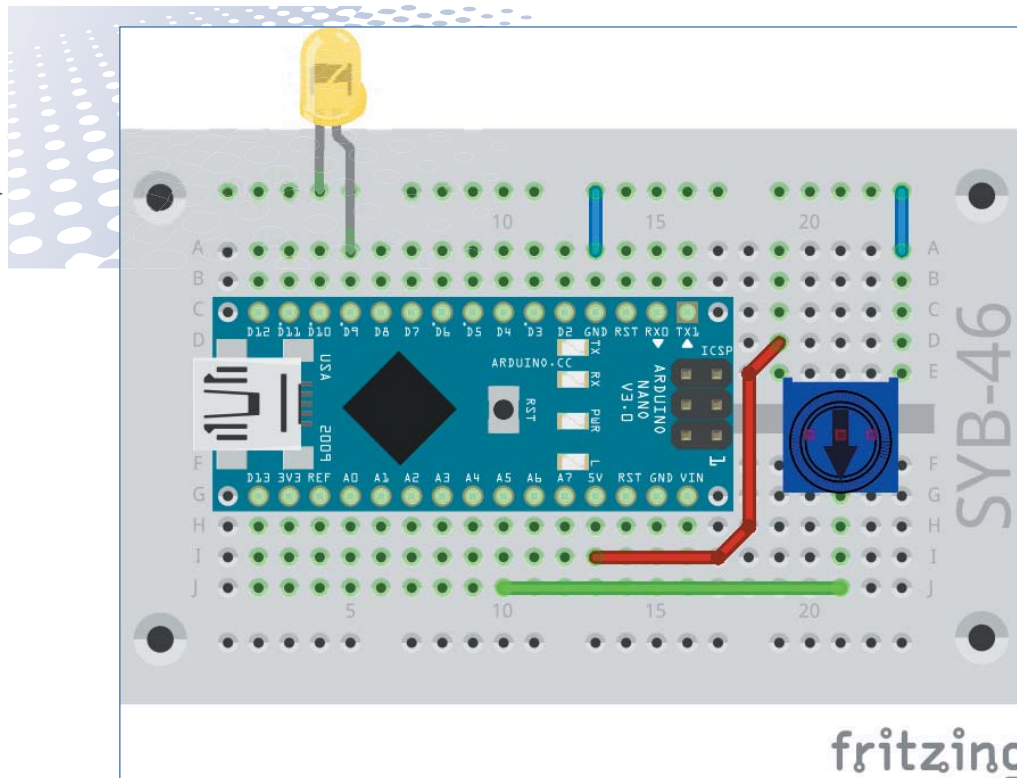


6 LED dimmen



Um eine LED wirklich wie eine Wohnzimmerlampe zu dimmen, verwenden wir das Potenziometer aus dem Paket. Ein Potenziometer ist ein einstellbarer Widerstand, mit dem sich ein Spannungsteiler bauen lässt, der

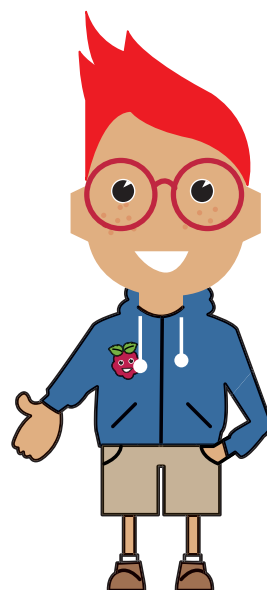
eine beliebige Spannung zwischen 0 V und der verwendeten Ausgangsspannung liefern kann. Dazu werden die beiden äußeren Anschlüsse des Potenziometers mit 0 V und +5 V verbunden, der mittlere Anschluss, der





ACHTUNG!

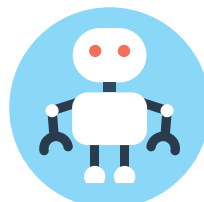
Dieses Experiment zeigt einen der wenigen Fälle, in denen die +5-V-Leitung des Nano verwendet wird. Die meisten Projekte nutzen nur +3,3 V, da die digitalen Eingänge, im Gegensatz zu den analogen Eingängen, nur mit 3,3 V belastet werden dürfen.



mit dem drehbaren Schleifer verbunden ist, wird an einen analogen Eingang des Nano angeschlossen.

Die mit dem Potenziometer eingestellte Spannung kann aber nicht

direkt an die LED weitergegeben werden, sondern muss in einen digitalen PWM-Wert umgerechnet werden.



6 LED dimmen

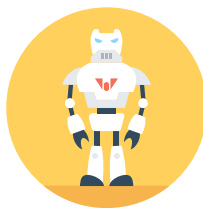
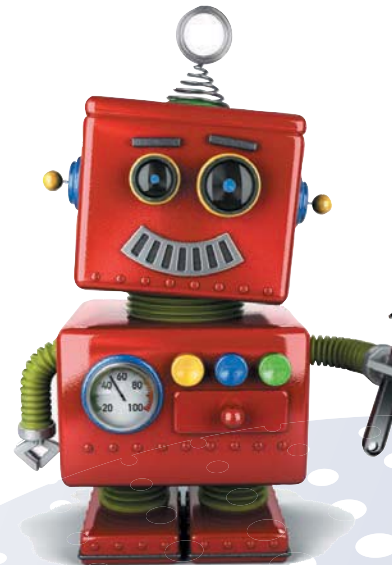


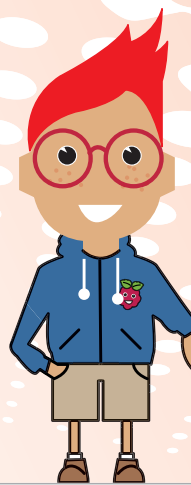
SO FUNKTIONIERT DAS PROGRAMM

Wenn du auf das grüne Fähnchen klickst, startet die Endlosschleife, die den Wert des analogen Pins 5 ausliest und durch 4 teilt. Das Ergebnis wird in der Variablen **x** gespeichert und auf der Bühne angezeigt.



Anschließend wird der PWM-Pin 9 auf diesen Wert gesetzt. Damit leuchtet die angeschlossene LED je nach Einstellung des Potenziometers unterschiedlich hell.

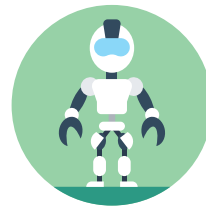




PWM-WERTE UND ANALOGE EINGANGSWERTE

Die analogen Eingänge des Nano werten einen analogen Spannungswert aus und liefern abhängig von der anliegenden Spannung digitale Werte zwischen **0** und **1023**. Dabei steht **0** für 0 V und **1023** für +5 V Spannung am jeweiligen Pin.

PWM-Werte müssen aber zwischen **0** und **255** liegen. Um einen Wert zwischen **0** und **1023** in einen Wert zwischen **0** und **255** umzurechnen, teilt man ihn einfach durch **4**.



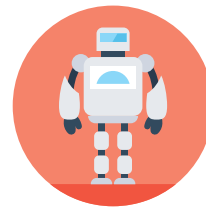
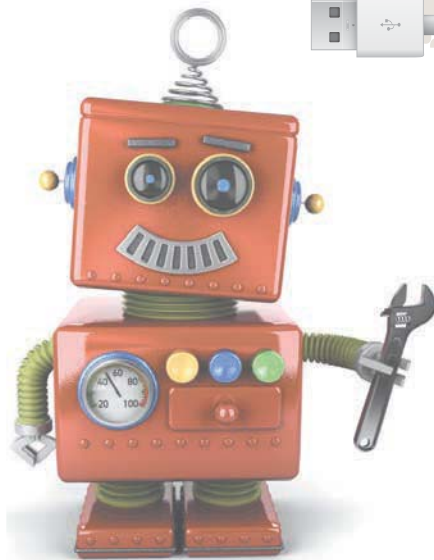
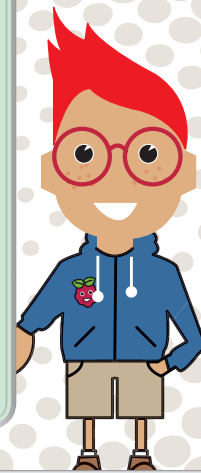
7 Pegelanzeige

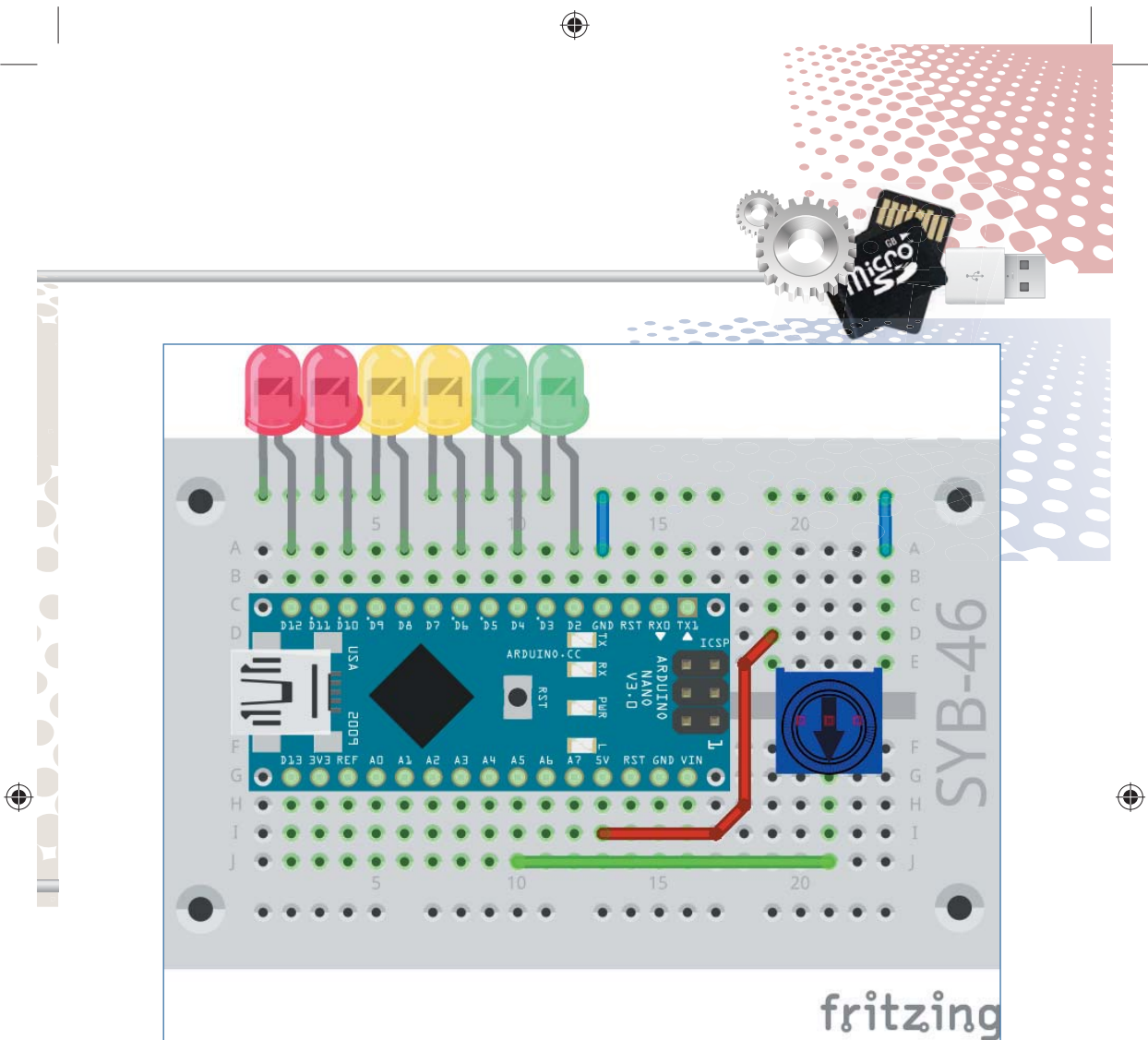


Wie man mit analogen Werten die Helligkeit einer LED steuert, hast du bereits ausprobiert – um einen analogen Wert aber wirklich zu visualisieren, ist diese Methode nicht aussagekräftig genug. Helligkeiten werden sehr subjektiv wahrgenommen und hängen auch stark von der Umgebung und den verwendeten LEDs ab. An einer Pegelanzeige lassen sich analoge Werte viel deutlicher ablesen.

BAUTEILE

- 1x Nano
- 1x Steckbrett
- 2x LED rot (mit Vorwiderstand)
- 2x LED gelb (mit Vorwiderstand)
- 2x LED grün (mit Vorwiderstand)
- 1x 15-kOhm-Potenzio-
meter
- 4x isolierte Drahtbrücke (unterschiedliche Längen)





Die LEDs sind mit den digitalen Pins 2, 4, 6, 8, 10 und 12 verbunden, das Potenziometer verwendet wie im letzten Projekt den analogen Eingang A5.

Das Programm **05pegel** lässt je nach Einstellung des Potenziometers unterschiedlich viele LEDs leuchten. Ähnliche Anzeigen werden zum Beispiel bei Lautstärkereglern an HiFi-Anlagen verwendet.

7 Pegelanzeige



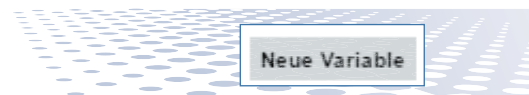
```
Wenn angeklickt
  setze x auf Liste 2 4 8 8 10 12
  fortlaufend
    setze p auf
    lies analogen Pin 5 / 1024 x Länge von x + 1
    setze i auf 1
    wiederhole Länge von x mal
      falls i < p
        Setze digitalen Pin Element i von x auf wahr
      sonst
        Setze digitalen Pin Element i von x auf falsch
    warte 0.01 Sek.
    ändere i um 1
```

SO FUNKTIONIERT DAS PROGRAMM

Die Nummern der für die LEDs verwendeten Pins sind in einer Liste gespeichert. Das hat den Vorteil, dass ein Programm mithilfe mathematischer Formeln auf beliebige

Werte der Liste zugreifen kann, die einfach nur durchnummeriert sind.

Listen werden in Snap4Arduino in Variablen gespeichert. Lege also als Erstes eine Variable **x** an.



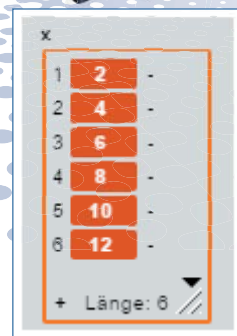
Ziehe den roten Block **Liste** in einen **Setze...auf...**-Block, um in der Variablen eine Liste anzulegen.



Klicke fünfmal hintereinander auf den Pfeil nach rechts, der sich ganz rechts im Block **Liste** befindet. Damit legst du insgesamt sechs Felder für die sechs LEDs in dieser Liste an. Schreibe in diese sechs Felder die Nummern der digitalen Pins, an denen die LEDs angeschlossen sind.

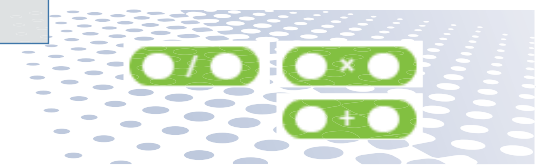


Die Liste wird zu Beginn des Programms angelegt und in der Variablen **x** gespeichert. Die Liste wird auf der Bühne in einem eigenen Listenfeld angezeigt.



Danach startet eine **fortlaufend**-Schleife, die fortlaufend den analogen Pin 5 abfragt. Dessen Wert, der zwischen 0 und 1023 liegen kann, wird durch 1024 geteilt und anschließend

mit der Anzahl der LEDs plus 1 multipliziert. Auf diese Weise erhalten wir einen Wert zwischen 0 und 6, um damit null bis sechs LEDs einzuschalten.



Diese Berechnungsformel wird aus drei grünen Operatoren für Division, Multiplikation und Addition zusammengesetzt.



7 Pegelanzeige



Der Block **Länge von...** gibt die Anzahl der Elemente in einer Liste an. Natürlich hätten wir den Wert des analogen Pins auch direkt durch 146 teilen können, um einen Zahlenwert zwischen 0 und 6 zu erhalten. Die ausführliche Berechnungsformel über die Länge der Liste hat aber den Vorteil, dass man, ohne das Programm zu verändern, weitere LEDs anschließen und der Liste hinzufügen kann, um die Anzeige zu verlängern. Umgekehrt können auch LEDs weggenommen und die Liste verkürzt werden.

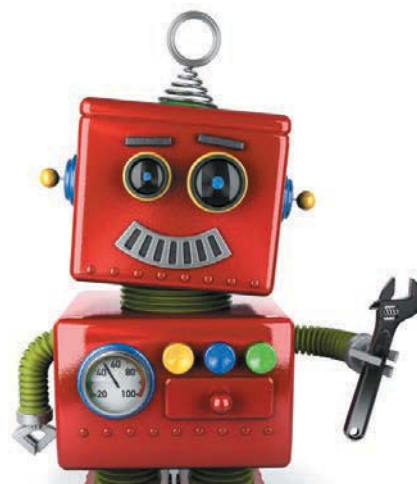
Danach wird eine neue Variable **i** auf 1 gesetzt. Programmierer verwenden den Variablennamen **i** gerne für programminterne Zählvariablen. Deshalb bezeichnen wir diesen Zähler hier auch als **i**. Mithilfe dieses Zählers schaltet das Programm eine LED nach der anderen in der Pegelanzeige ein – natürlich abhängig vom Wert, der auf dem Potenziometer eingestellt ist.

setze **i** auf 1

setze **p** auf
lies analogen Pin 5 / 1024 × Länge von x + 1

Das Ergebnis der Berechnung wird in der Variablen **p** gespeichert und gibt an, wie viele LEDs leuchten sollen.

Es folgt eine Schleife, die für jede LED einmal durchläuft und die

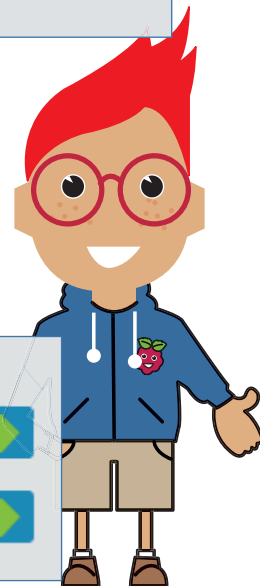




anhand des Potenziometerwerts festlegt, ob die jeweilige LED ein- oder ausgeschaltet wird.

```
Wenn angeklickt
  setze x auf Liste 2 4 6 8 10 12
  fortlaufend
    setze p auf
    lies analogen Pin 5 / 1024 x Länge von x + 1
    setze i auf 1
    wiederhole Länge von x mal
```

Eine *falls...sonst...*-Abfrage innerhalb der Schleife prüft, ob der errechnete Wert *p* größer ist als die aktuelle Nummer der LED, die in der Variablen *i* gespeichert ist. Wenn ja, wird diese LED eingeschaltet.



```
falls i < p
  Setze digitalen Pin Element i von x auf wahr
sonst
  Setze digitalen Pin Element i von x auf falsch
```

7 Pegelanzeige



Hier zeigt sich der große Vorteil von Listen gegenüber einzelnen Variablen: Mit dem Block **Element... von...** kann jedes Element einer Liste einfach über seine Nummer angesprochen werden.



In unserem Fall wird die LED mit der Nummer *i* ein- oder ausgeschaltet. Dazu verwenden wir den bereits bekannten **Setze digitalen Pin... auf ...**-Block.

Unabhängig davon, ob die aktuelle LED in einem Schleifendurchlauf ein- oder ausgeschaltet wird, folgt eine kurze Wartezeit von 0,01 Sekunden, da danach die Schleife gleich wieder startet und die nächste LED der Pegelanzeige einschaltet.



Zum Schluss wird die Variable *i*, die Nummer der LED, um 1 erhöht, und im nächsten Schleifendurchlauf wird geprüft, ob sie immer noch kleiner als der aus dem analogen Eingabewert errechnete Wert *p* ist. Abhängig

davon, wird auch diese LED ein- oder ausgeschaltet.



Nachdem alle sechs LEDs auf diese Weise ein- oder ausgeschaltet wurden, startet die Hauptschleife des Programms neu und fragt erneut den aktuell auf dem Potenziometer eingestellten Wert ab. Dieser wird ausgewertet und auf der Pegelanzeige dargestellt.

Das Ganze wiederholt sich, bis du auf das rote Stopp-Symbol klickst.

