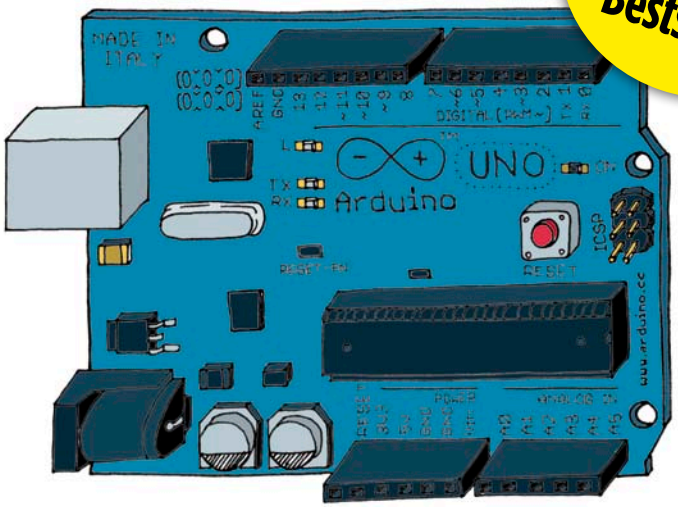
O'REILLY®

3. Auflage

Arduino für Einsteiger

Aktualisierter und erweiterter Bestseller



Die OPEN-SOURCE-PLATTFORM für MAKER

Massimo Banzi, Mitbegründer von Arduino, & Michael Shiloh

Übersetzung von Tanja Feder

3. AUFLAGE

Arduino für Einsteiger

Massimo Banzi &
Michael Shiloh

O'REILLY®
Beijing · Cambridge · Farnham · Köln · Sebastopol · Tokyo

Kommentare und Fragen können Sie gerne an uns richten: :
O'Reilly Verlag
Balthasarstr. 81
50670 Köln
E-Mail: kommentar@oreilly.de

Die Originalausgabe erschien im Dezember 2014 unter dem Titel
Getting Started with Arduino, 3rd bei Maker Media, Inc.

Bibliografische Information Der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über *http://dnb.dnb.de* abrufbar.

Lektorat: Volker Bombien, Köln
Übersetzung: Tanja Feder, Bonn
Korrektorat: Sabine Krämer, Gütersloh
Umschlaggestaltung: Karen Montgomery, Boston & Michael Oreal, Köln
Produktion: Karin Driesen, Köln
Satz: le-tex publishing services GmbH, Leipzig, *www.le-tex.de*
Belichtung, Druck und buchbinderische Verarbeitung:
Druckerei Kösel, Krugzell; *www.koeselbuch.de*

ISBN: 978-3-95875-055-5

Dieses Buch ist auf 100% chlorfrei gebleichtem Papier gedruckt.

Inhaltsverzeichnis

Vorwort . . . . . **vii**

1/Einleitung **1**

An wen sich das Buch richtet **2**

Was ist Interaction Design? **2**

Was ist Physical Computing? **3**

2/Die Philosophie von Arduino **5**

Prototyping **5**

Tüfteln **6**

Patching **7**

Modifizieren von Schaltkreisen **9**

Keyboard-Hacks **11**

Wir lieben Elektroschrott **12**

Hacken von Spielzeug **13**

Kooperation **14**

3/Die Arduino-Plattform **15**

Die Arduino-Hardware **15**

Die Integrierte Entwicklungsumgebung (IDE) **18**

Die Installation von Arduino auf dem Computer **18**

Installieren der IDE: Macintosh **19**

Installieren der IDE: Windows **20**

4/Die wirklich ersten Schritte mit Arduino **23**

Der Aufbau eines interaktiven Gerätes **23**

Sensoren und Aktoren **24**

Eine LED zum Blinken bringen **24**

Reich mir den Parmesan **29**

Arduino ist nichts für Zauderer **29**

Wirkliche Tüftler schreiben Kommentare **30**

Der Code – Schritt für Schritt **30**

Was wir bauen werden **34**

Was ist Elektrizität? **34**

Steuerung einer LED mit einem Drucktaster **38**

Erläuterung der Funktionsweise 41
Ein Schaltkreis – 1.000 Verhaltensweisen 42

5/Erweiterter Input und Output 47
Der Einsatz anderer Ein/Aus-Sensoren 47
Steuerung von Licht mittels PWM 50
Einsatz eines Lichtsensors anstelle eines Drucktasters 58
Analoger Eingang 60
Der Einsatz anderer analoger Sensoren 63
Serielle Kommunikation 64
Der Umgang mit größeren Lasten 66
Komplexe Sensoren 67

6/Der Arduino Leonardo 69
Worin unterscheidet sich dieses Arduino-Board von anderen Arduino-Boards? 69
Weitere Unterschiede zwischen Arduino Leonardo und Arduino Uno 70
Beispiel für eine Tastaturnachricht mit dem Leonardo 71
Beispiel für eine Steuerung der Maustaste mit dem Leonardo 74
Weitere Unterschiede beim Leonardo 78

7/Kommunikation mit der Cloud 81
Planung 83
Der Code 84
Das Zusammenbauen des Schaltkreises 90
So funktioniert das Zusammenbauen 91

8/Ein System zur automatischen Gartenbewässerung 93
Planung 95
Testen der Echtzeituhr (RTC) 98
Testen der Relais 103
Der elektronische Schaltplan 106
Testen des Temperatur- und Feuchtigkeitssensors 117
Programmierung 120
Zusammenbau des Schaltkreises 139
Ideen zum Ausprobieren 168
Die Einkaufsliste für das Bewässerungsprojekt 169

9/Troubleshooting 171
Verständnis 171
Vereinfachung und Segmentierung 172
Ausschließen und Vergewissern 172

Testen des Board 172
Testen des Schaltkreises auf der Steckplatine 174
Das Isolieren von Problemen 176
Probleme beim Installieren von Treibern unter Windows 176
Probleme mit der IDE unter Windows 177
Identifizierung der Arduino-COM-Ports unter Windows 177
Andere Debugging-Techniken 178
So findest du Online-Hilfe 180

A/Die Steckplatine **183**

B/Das Lesen von Widerständen und Kondensatoren **187**

C/Arduino-Kurzreferenz **191**

D/Das Lesen von Schaltskizzen **207**

Index **211**

Vorwort

Der dritten Auflage von *Arduino für Einsteiger* wurden zwei neue Kapitel hinzugefügt: Bei Kapitel 8, *Ein System zur automatischen Gartenbewässerung*, auf Seite 93 handelt es sich um ein ehrgeiziges Projekt, in dem ein komplexerer Schaltkreis und ein komplizierteres Programm veranschaulicht werden. In diesem Kapitel werden auch Projektdesign, Testen und Konstruktion erläutert und Schaltpläne benutzt, die in Anhang D beschrieben werden.

Im zweiten neuen Kapitel, Kapitel 6, *Der Arduino Leonardo*, auf Seite 69, wird der Arduino Leonardo vorgestellt. Beim Leonardo handelt es sich um eine andere Art Arduino, weil der USB-Controller in der Software implementiert ist und sich nicht auf einem separaten Chip befindet, wie es vor dem Leonardo der Fall war. Dadurch kann das USB-Verhalten auf dem Board modifiziert werden.

Abgesehen von diesen neuen Kapitel hat es weitere Updates gegeben:

Die dritte Ausgabe ist für die Version 1.0.5 der Arduino-IDE geschrieben. Zwischenzeitlich ist die Arduino-IDE-Version 1.5 erschienen. In Vorwegnahme der Veröffentlichung von Version 1.5 haben wir die Unterschiede zwischen 1.0.5 und 1.5 aufgeführt.

Zahlreiche Vorschläge von Studenten und anderen Lesern wurden implementiert.

Michael

Vorwort zur 2. Auflage

Vor einigen Jahren stand ich vor einer sehr interessanten Herausforderung: Ich sollte Designern die einfachsten Grundlagen der Elektronik vermitteln, um sie in der Lage zu versetzen, interaktive Prototypen der Objekte, an denen sie gerade arbeiteten, herzustellen.

Ich folgte unterbewusst meinem Instinkt, Elektronik auf die Weise zu lehren, wie ich es von der Schule her kannte. Später wurde mir klar, dass das nicht so gut funktionierte, wie ich das gerne gehabt hätte, und ich erinnerte mich an die Stunden im Klassenzimmer, in denen jede Menge Theorie ohne praktischen Bezug auf mich eingeprasselt war und ich mich zu Tode gelangweilt hatte.

Eigentlich kannte ich die Elektronik zur Schulzeit bereits – und diese Kenntnisse hatte ich auf einem sehr empirischen Weg erworben: mit sehr wenig Theorie, aber mit viel praktischer Erfahrung.

Ich begann also, darüber nachzudenken, auf welche Weise ich Elektronik wirklich verstanden habe:

- Ich nahm alle elektronischen Geräte auseinander, die ich in die Finger bekommen konnte.
- So lernte ich langsam alle einzelnen Komponenten kennen.
- Ich begann, mit ihnen herumzubasteln, einige ihrer inneren Verbindungen zu verändern und dann zu beobachten, wie das Gerät reagierte, üblicherweise mit einer Art Explosion oder mit einer Rauchwolke.
- Ich baute Bausätze aus Beilagen von Elektrozeitschriften zusammen.
- Ich kombinierte von mir gehackte Geräte mit zweckentfremdeten Bausätze und anderen Schaltungen, die ich in Zeitschriften gefunden hatte, um aus ihnen etwas Neues herzustellen.

Als kleines Kind war ich fasziniert davon, herauszufinden, wie Dinge funktionieren, daher habe ich sie immer auseinandergenommen. Je mehr nicht benutzte elektronische Objekte, die ich irgendwo im Haus fand und in ihre Einzelteile zerlegte, desto größer wurde mein Interesse. Schließlich brachten die Leute alle möglichen Geräte zu mir, damit ich sie auseinandernehmen sollte. Meine größten Objekte zu dieser Zeit waren eine Geschirrspülmaschine und ein früher Computer aus einem Versicherungsbüro, der über einen Drucker, Elektronikkarten, Magnetkartenleser und viele andere Teile verfügte, deren komplette Zerlegung sich als äußerst interessant und knifflig erwies.

Nach umfangreichen Untersuchungen wusste ich, was elektronische Komponenten sind, und auch ungefähr, was sie tun. Obendrein stapelten sich in unserem Haus große Mengen elektronische Zeitschriften, die mein Vater irgendwann Anfang der 1970er gekauft haben musste. Ich habe Stunden damit verbracht, die Artikel zu lesen und mir die Schaltskizzen anzuschauen, ohne allerdings sonderlich viel zu begreifen.

Dieses immer wieder erneute Lesen der Artikel auf der äußerst hilfreichen Grundlage des Wissens, das ich durch das Zerlegen von Schaltungen erworben hatte, erwies sich als langsamer, wirkungsvoller Prozess.

Ein großer Durchbruch kam an einem Weihnachtstag, als mein Vater mir einen Bausatz schenkte, mit dem Teenagern Wissen über die Elektronik vermittelt werden sollte. Jede Komponente war in einem Plastikwürfel untergebracht, der magnetisch an den anderen Würfeln haften konnte, so dass eine Verbindung entstand. Oben auf diesen Würfeln war das jeweilige elektronische Symbol angeführt. Ich wusste noch wenig davon, dass dieses Spielzeug auch ein Meilenstein des Designs "Made in Germany" war, denn es war bereits in den 1960ern von Dieter Rams entwickelt worden.

Mit diesem neuen Tool konnte ich schnell Schaltkreise zusammenbauen, ausprobieren und mir dann das Resultat ansehen. Der Prototyping-Zyklus wurde dabei immer kürzer.

Danach baute ich Radios, Verstärker, Schaltkreise, die fürchterlichen Lärm oder auch schöne Töne produzierten, Regensensoren und kleine Roboter.

Ich habe lange nach einem englischen Begriff gesucht, der diese Arbeitsweise ohne speziellen Plan wiedergibt, bei der man einfach von einer bestimmten Idee ausgeht und bei einem völlig unerwarteten Resultat landet. Schließlich stieß ich auf das Wort Tinkering, das sich mit dem Begriff Tüfteln ins Deutsche übertragen ließe. Ich erkannte, dass dieser Begriff in vielen anderen Bereichen verwendet wurde, um eine Arbeitsweise zu beschreiben und Menschen zu porträtieren, die ausgetretene Pfade verlassen und Neuland erkundet hatten. Auch die französischen Regisseure, die die Nouvelle Vague begründeten, wurden im englischen Sprachraum als Tinkerer bezeichnet. Die beste Definition, die ich kenne, habe ich im Rahmen einer Ausstellung im Exploratorium in San Francisco (*www.exploratorium.edu/tinkering*) gefunden:

> Tinkering is what happens when you try something you don't quite know how to do, guided by whim, imagination, and curiosity. When you tinker, there are no instructions – but there are also no failures, no right or wrong ways of doing things. It's about figuring out how things work and reworking them.
>
> Contraptions, machines, wildly mismatched objects working in harmony – this is the stuff of tinkering.
>
> Tinkering is, at its most basic, a process that marries play and inquiry.

Von meinen früheren Experimenten wusste ich bereits, wie viel Erfahrung nötig ist, um einen Schaltkreis aus Basiskomponenten aufzubauen, der dann auch noch das tut, was du möchtest.

Ein weiterer Durchbruch erfolgte im Sommer 1982, in dem ich mit meinen Eltern in London viele Stunden lang das Science Museum besichtigte. Hier war gerade ein neuer Bereich eröffnet worden, der Computern gewidmet war und in dem angeleitete Experimente vorgestellt wurden. Indem ich ihnen folgte, lernte ich die Grundlagen der Binärmathematik und der Programmierung.

Dabei stellte ich fest, dass Ingenieure bei vielen Anwendungen keine Schaltkreise mehr aus Basiskomponenten bauten, sondern viele intelligente Möglichkeiten mittels Mikroprozessoren in ihre Produkte implementieren. Software ersparte dabei viele Arbeitsstunden beim elektronischen Design und ermöglichte kürzere Zyklen beim Tüfteln.

Nach der Rückkehr begann ich Geld zu sparen, weil ich mir einen Computer kaufen und das Programmieren lernen wollte.

Mein erstes und wichtigstes Objekt war ein brandneuer ZX81-Computer, mit dem ich eine Schweißmaschine steuerte. Das klingt sicher nicht nach einem besonders spannenden Projekt, aber es bestand ein gewisser Bedarf und für mich war es eine große Herausforderung, weil ich gerade erst das Programmieren erlernt hatte. Zu diesem Zeitpunkt wurde mir klar, dass das Schreiben von Code-Zeilen weniger zeitaufwendig ist als das Aufbauen komplexer Schaltungen.

Mehr als 20 Jahre später denke ich, dass es diese Erfahrung ist, die es mir ermöglicht, Menschen zu unterrichten, die sich nicht einmal daran erinnern, irgendeine Mathematikstunde besucht zu haben, und ihnen die gleiche Begeisterung für das Tüfteln und die entsprechenden Fähigkeiten zu vermitteln, die ich in meiner Jugend erworben und mir seitdem bewahrt habe.

Massimo

Danksagung von Massimo Banzi

Dieses Buch ist Ombretta gewidmet.

Danksagung von Michael Shiloh

Dieses Buch ist meinem Bruder und meinen Eltern gewidmet.

Zuallererst möchte ich mich bei Massimo dafür bedanken, dass er mich eingeladen hat, an der dritten Auflage dieses Buches mitzuarbeiten und überhaupt in die Welt des Arduino einzutauchen. Es war ein wirkliches Privileg und ein wahres Vergnügen, an diesem Projekt teilzuhaben zu können.

Als Nächstes möchte ich Brian Jepson für seine Beratung, seine Betreuung, seine Ermutigung und seine Unterstützung danken. Bei Frank Teng möchte ich mich dafür bedanken, dass er mich stets in der Spur hielt. Ein weiter Dank gebührt Kim Cofer und Nicole Shelby, die einen fantastischen Job im Lektorat und der Herstellung gemacht haben.

Bei meiner Tochter Yasmine möchte ich mich dafür bedanken, dass sie eine solch hohe Meinung von mir hat, und dafür, dass sie mich unentwegt unterstützt und mich ermutigt hat, meine Interessen konsequent zu verfolgen. Ich danke ihr auch dafür, dass sie mich für irgendwie cool hält, obwohl ich ihr Vater bin. Ohne ihre Unterstützung hätte ich diese Arbeit nie geschafft.

Zu guter Letzt möchte ich mich bei bei meiner Partnerin Judy Aime' Castro für die endlosen Stunden bedanken, die sie damit verbracht hat, mein Gekritzel in schöne Illustrationen umzuwandeln, für die Diskussionen, die wir zu verschiedenen Aspekten dieses Buches geführt haben, und für ihre grenzenlose Geduld mit mir.

In diesem Buch benutzte Konventionen

Die folgenden typografischen Konventionen werden in diesem Buch verwendet:

Kursiv
: Steht für neue Begriffe, URLs, E-Mail-Adressen, Dateinamen und Dateierweiterungen.

`Feste Breite`
: Programmlistings und Programmelemente im Fließtext, zum Beispiel Variablen oder Funktionsnamen, Datenbanken, Datentypen, Umgebungsvariablen, Anweisungen und Schlüsselwörter.

`Feste Breite Fettdruck`
: Befehle oder anderer Text, der genau so vom Anwender eingegeben werden sollte.

`Feste Breite kursiv`
: Text, der vom Anwender durch eigene Werte ersetzt werden sollte.

Dieses Symbol zeigt einen Tipp, eine Empfehlung oder eine allgemeine Anmerkung.

Dieses Symbol zeigt eine Warnung oder Vorsichtsmaßnahme.

1/Einleitung

Arduino ist eine Open-Source-Plattform für *Physical Computing*, mit der sich interaktive Objekte herstellen lassen. Dabei kann es sich um Stand-Alone-Objekte handeln oder um solche, die mit der Software auf deinem Computer interagieren. Arduino ist für Künstler, Designer und andere Leute gedacht, die Physical Computing in ihr Design integrieren möchten, ohne dafür erst Elektrotechnik studieren zu müssen.

Die Arduino-Hardware und -Software ist Open Source. Durch die Open-Source-Philosophie wird eine Community gefördert, die ihr Wissen großzügig teilt. Für Anfänger ist dies eine tolle Sache, denn Hilfe ist meistens online jederzeit verfügbar, und zwar auf vielen verschiedenen Niveaus und in einer verblüffenden Bandbreite an Themen. Projekte werden nicht nur in Form von Bildern vorgestellt, sondern du findest auch Anweisungen, wie du dein eigenes Projekt entwickelst oder das vorgestellte Objekt als Ausgangspunkt für den Einbau in einer abgeleiteten oder ähnlichen Version nutzen kannst.

Die Arduino-Software, bekannt als Integrierte Entwicklungsumgebung (Integrated Development Environment, IDE), ist frei und steht unter *www.arduino.cc* jedem zum Download bereit. Die Arduino-IDE basiert auf der Sprache Processing *http://www.processing.org*, die entwickelt wurde, um Künstlern zu helfen, Computer-Art zu entwickeln, ohne erst zu Software-Ingenieuren werden zu müssen. Die Arduio-IDE kann unter Windows, Macintosh und Linux betrieben werden.

Das Arduino-Board ist preiswert (etwa 40 Euro) und recht tolerant gegenüber typischen Anfängerfehlern. Wenn du etwas tust, wodurch die Hauptkomponente auf dem Arduino Uno beschädigt wird, kostet es weniger als 4 Euro, sie zu ersetzen.

Das Arduino-Projekt wurde in einer Lernumgebung entwickelt und ist ein sehr beliebtes Lehrwerkzeug. Aufgrund derselben Open-Source-Philosophie, auf der die Communitys basieren, in denen großzügig Informationen, Antworten und Projekte geteilt werden, werden auch Lehrmethoden, Lehrinhalte und andere Informationen geteilt. Bei Arduino gibt es eine spezielle Mailing-Liste (*http://bit.ly/1vKh0wb*, um Diskussionen unter Nutzern zu erleichtern, die daran interessiert sind, mithilfe von oder über Arduino zu unterrichten.

Da die Arduino-Hardware und -Software Open Source sind, kannst du das Arduino-Hardware-Design herunterladen und dein eigenes entwickeln oder es als Ausgangspunkt für eigene Projekte nutzen, deren Design auf Arduino

basiert (oder Arduino enthält), oder einfach um zu verstehen, wie Arduino funktioniert. Dasselbe gilt für die Software.

Dieses Buch soll Einsteigern ohne Vorkenntnisse helfen, erste Schritte mit dem Arduino zu wagen.

An wen sich das Buch richtet

Dieses Buch ist für die "ursprünglichen" Arduino-Benutzer geschrieben: Designer und Künstler. Daher werden Dinge in einer Weise erklärt, die einigen Ingenieuren möglicherweise die Haare zu Berge stehen lässt. Einer nannte die einleitenden Kapitel meines ersten Entwurfs sogar "fluff" (Staubfusseln). Das ist genau der Punkt, denn seien wir mal ehrlich: die meisten Ingenieure können anderen Ingenieuren und erst recht fachfremden Personen, das, was sie tun, nicht erklären. Wir wollen nun tief in diese "Staubfusseln" eintauchen.

Dieses Buch soll kein Lehrbuch für Elektronik und Programmierung sein, du wirst aber beim Lesen etwas über Elektronik und Programmierung lernen.

> Als Arduino allmählich beliebter wurde, konnte ich beobachten, wie Experimentatoren, Hobbybastler und alle Arten von Hackern schöne und verrückte Objekte herstellten. Ich erkannte, dass ihr selbst alle Künstler und Designer seid. Daher ist dieses Buch auch euch gewidmet.
>
> Massimo

Arduino basiert auf einer Diplomarbeit von Hernando Barragan, die er über die Wiring-Plattform schrieb, als er unter Casey Reas und mir am Interaction Design Institute (IDII) studierte.

Was ist Interaction Design?

Arduino wurde aus der Idee geboren, Interaction Design, eine Design-Disziplin, bei der das Prototyping im Zentrum der Methodik steht, zu vermitteln. Es gibt viele Definitionen für den Begriff Interaction Design, wir bevorzugen die folgende:

Interaction Design ist das Design einer beliebigen interaktiven Erfahrung.

In unserer heutigen Welt befasst sich Interaction Design mit dem Erzeugen bedeutsamer Erfahrungen zwischen uns (Menschen) und Objekten. Dies ist eine gute Methode, um das Entstehen von schönen und vielleicht auch kontroversen Erfahrungen im Umgang mit der Technik zu erschließen. Beim

Interaction Design erfolgt Design in einem Iterationsprozess, basierend auf Prototypen und mit ständig wachsender Präzision. Dieser Ansatz, der teilweise auch beim konventionellen Design zu finden ist, kann so erweitert werden, dass Prototyping in die Technologie eingebunden wird, speziell im Bereich Elektronik.

Der spezielle Bereich von Interaction Design, der bei Arduino zum Tragen kommt, ist das Physical Computing (oder auch Physical Interaction Design).

Was ist Physical Computing?

Beim Physical Computing wird Elektronik verwendet, um Prototypen von neuen Arbeitsmaterialien für Designer und Künstler herzustellen. Dies umfasst auch das Design von interaktiven Objekten, die über Sensoren und Aktoren, die mittels einer vorgegebenen Verhaltensweise gesteuert werden, mit den Menschen kommunizieren können. Diese Verhaltensweise ist als Software implementiert, die in einem Mikrocontroller (ein kleiner Computer auf einem einzelnen Chip) ausgeführt wird.

Früher musste man beim Einsatz von Elektronik ständig Ingenieure hinzuziehen, weil Schaltkreise aus kleinen Bauteilen zusammengesetzt wurden. Dadurch wurden kreative Menschen daran gehindert, mit dem Medium direkt zu experimentieren. Die meistens Tools waren für Ingenieure gedacht und setzten erhebliche Kenntnisse voraus.

In den letzten Jahren wurden Mikrocontroller billiger und einfacher in der Handhabung. Gleichzeitig wurden sie schneller und leistungsfähiger, wodurch die Herstellung besserer (und einfacherer) Tools für die Entwicklung ermöglicht wurde.

Der Fortschritt, den wir mit dem Arduino erlangten, bestand darin, dass diese neuen Tools Neulingen näher gebracht wurden, so dass es ihnen möglich wurde, nach nur einem zwei- oder dreitägigen Workshop bereits beeindruckende Dinge zu bauen. Mit Arduino können sich Designer und Künstler die Grundlagen von Elektronik und Sensoren sehr schnell aneignen und ohne große Investitionen mit dem Bau von Prototypen beginnen.

2/Die Philosophie von Arduino

Die Philosophie von Arduino besteht darin, Design zu erstellen, anstatt darüber zu sprechen. Sie besteht in einem andauernden Suchen nach schnelleren und leistungsstärkeren Möglichkeiten, um bessere Prototypen zu bauen. Wir haben viele Prototyping-Techniken erkundet und so quasi mit unseren Händen neue Denkansätze geschaffen.

Das klassische Engineering beruht auf einem strikten Prozess, der von A nach B führt; bei Arduino besteht der Spaß in der Möglichkeit, auf diesem Weg verlorenzugehen und stattdessen bei C zu landen.

Dies ist der Prozess des Tüftelns, den wir so liebgewonnen haben – grenzenlos mit einem Medium herumexperimentieren und dabei das Unerwartete entdecken. Bei dieser Suche nach Wegen, bessere Prototypen herzustellen, haben wir auch eine Reihe von Software-Paketen ausgewählt, die einen Prozess der ständigen Veränderung des Software- oder des Hardware-Mediums ermöglichen.

In den nächsten Abschnitten werden einige philosophische Aspekte, Ereignisse und Pioniere vorgestellt, durch die die Philosophie von Arduino inspiriert wurde.

Prototyping

Prototyping ist das Herzstück der Arduino-Philosophie: Wir stellen Dinge her und bauen Objekte, die mit anderen Objekten, Menschen oder Netzwerken interagieren. Wir sind bestrebt, einen einfacheren und schnelleren Weg für das Prototyping zu finden, der außerdem möglichst kostengünstig sein soll.

Viele Einsteiger gehen zunächst mit der Vorstellung an Elektronik heran, dass sie lernen müssen, alles von Grund auf selbst zu bauen. Das ist reine Energieverschwendung: Was du wirklich möchtest, ist die Bestätigung, dass etwas sehr schnell funktioniert, so dass du selbst motiviert bist, den nächsten Schritt zu unternehmen, oder dass du sogar jemand anderen motivierst, entsprechend großzügig in dich zu investieren.

Daher haben wir das *opportunistische Prototyping* entwickelt: Warum sollten wir Zeit und Energie darauf verschwenden, Dinge von Grund auf zu bau-

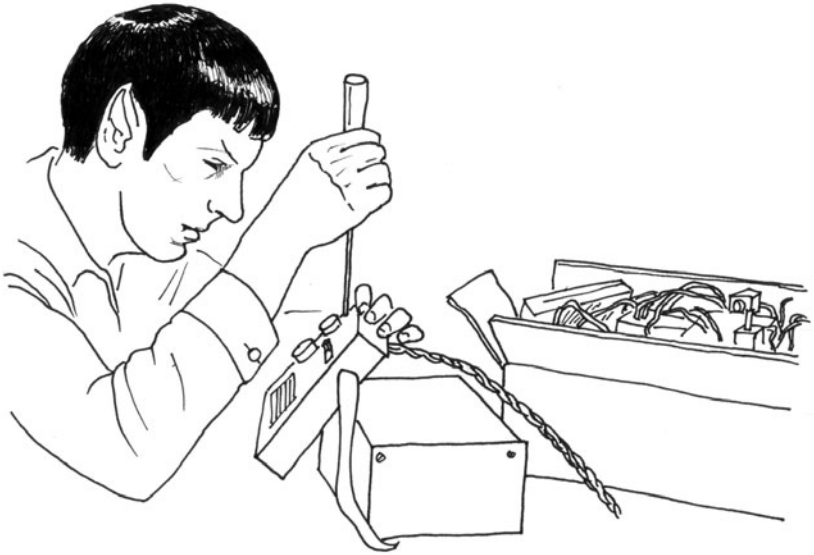
en, ein Prozess, der viel Zeit und tiefgehendes technisches Wissen erfordert, wenn wir fertige Geräte hacken und so die harte Arbeit nutzen können, die von großen Unternehmen und fähigen Ingenieuren bereits getan wurde?

Unser Held ist James Dyson, der 5.127 Prototypen seines Vakuumstaubsaugers baute, bevor er mit dem Resultat zufrieden war (*www.dyson.co.uk/*).

Tüfteln

Wir glauben, dass es essenziell ist, mit Technologie herumzuexperimentieren und verschiedene Möglichkeiten direkt mit der Hard- oder Software auszuprobieren – manchmal, ohne dabei ein wirklich definiertes Ziel zu haben.

Das Verwerten von bereits vorhandener Technologie ist eine der besten Möglichkeiten beim Tüfteln. Durch das Sammeln und Hacken von billigem Spielzeug und alten ausgemusterten Geräten lassen sich tolle Resultate erzielen.

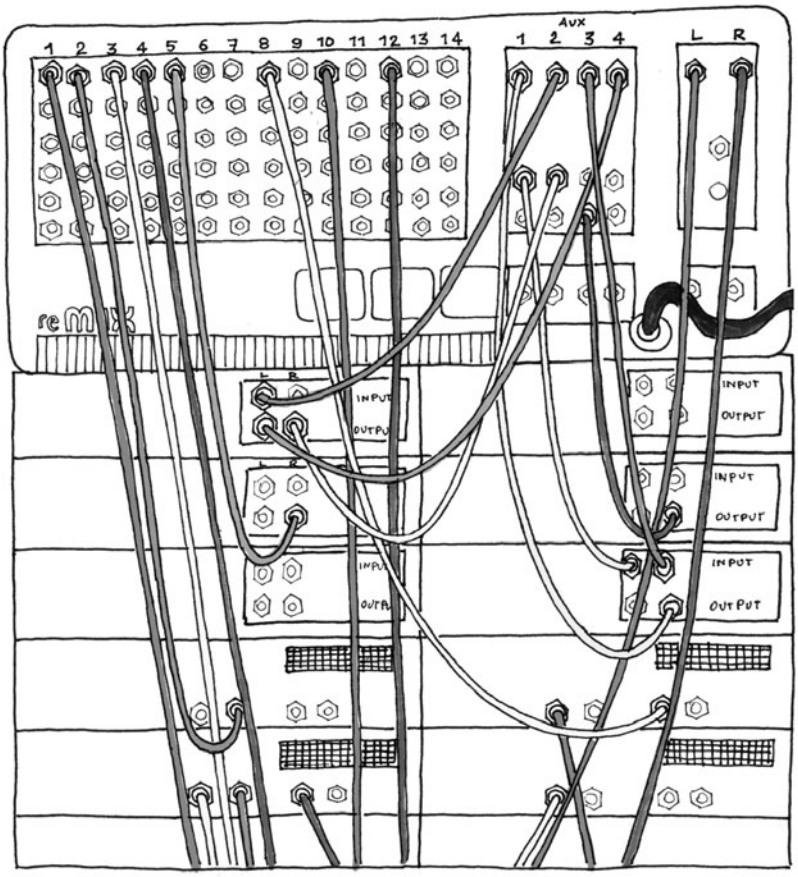


Patching

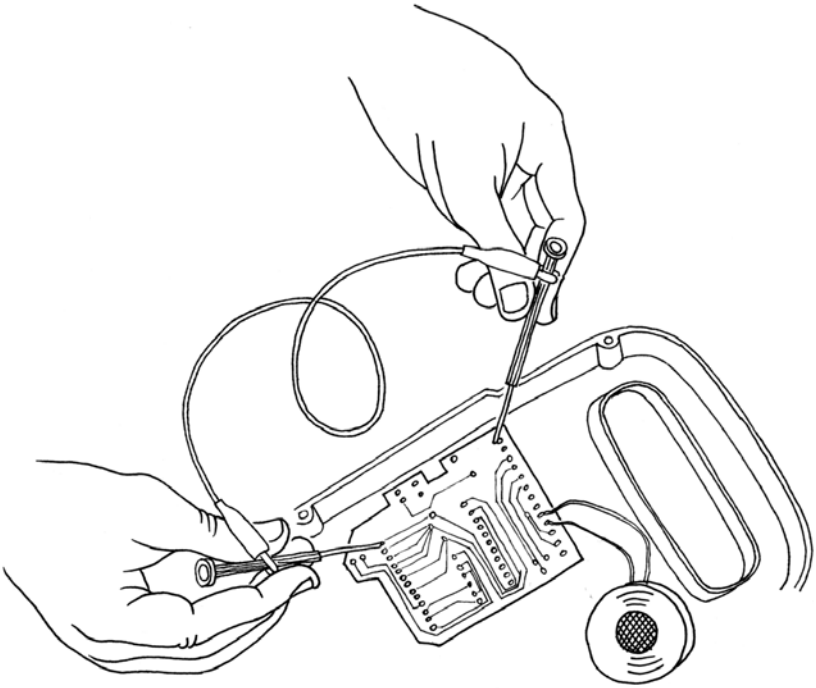
> Ich war immer fasziniert vom Baukastenprinzip und von der Möglichkeit, durch das Verbinden einfacher Geräte komplexe Systeme zu schaffen. Diese Methode wird sehr gut durch Robert Moog und seine analogen Synthesizer repräsentiert. Musiker erzeugten Sounds, indem sie verschiedene Module mit Kabeln zusammensteckten und so unzählige Kombinationen herstellten. Durch diesen Ansatz hatten Synthesizer oft das Aussehen alter Telefon-Switchboards, die allerdings mit zahlreichen Finessen ausgestattet waren und so eine perfekte Plattform für Soundexperimente und musikalische Innovationen darstellten. Moog beschrieb dies als einen Prozess zwischen Beobachten und Entdecken. Ich bin sicher, dass die Musiker zu Beginn nicht wussten, wozu die Hunderten von Knöpfen dienten, aber sie experimentierten unaufhörlich und entwickelten ihren Stil ständig weiter, ohne Unterbrechung dieses Prozesses.
>
> Massimo

Möglichst wenige Unterbrechungen im Prozess zu haben, ist sehr wichtig für die Kreativität – je nahtloser der Prozess ist, desto besser funktioniert das Tüfteln.

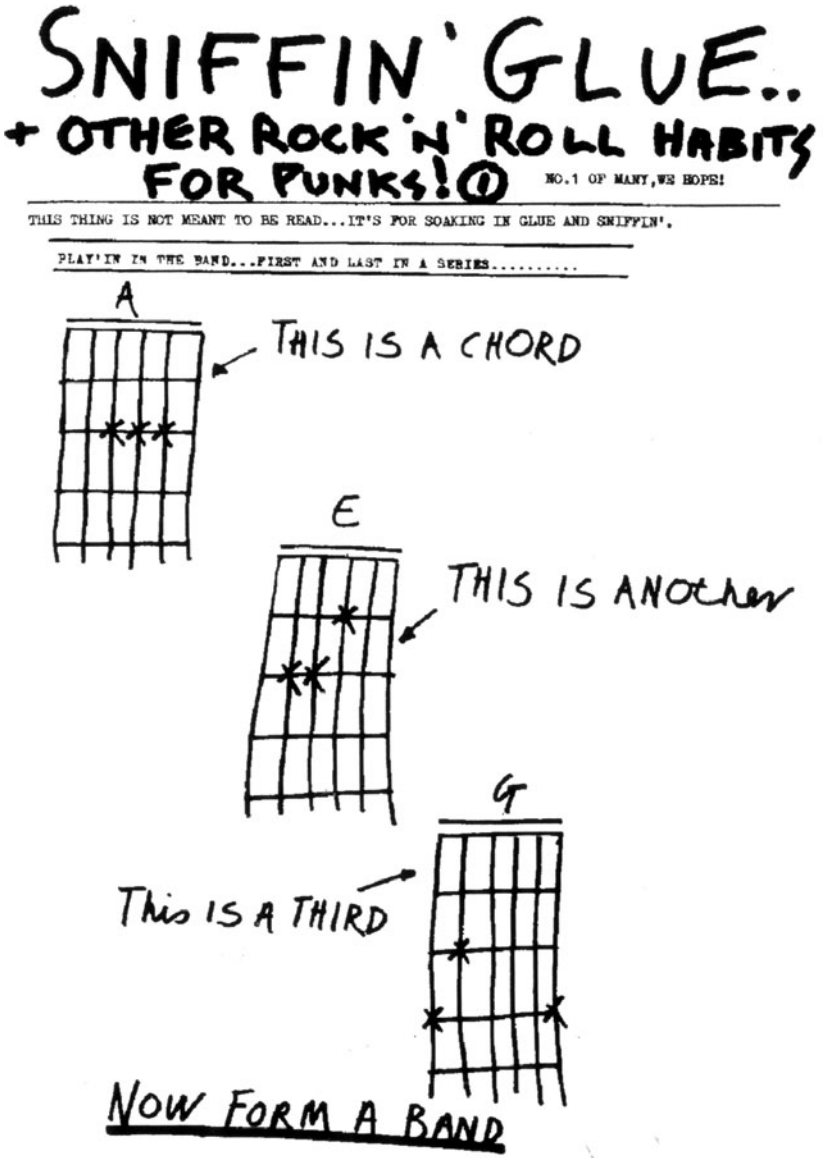
Diese Technik wurde in Form von Entwicklungsumgebungen für das visuelle Programmieren wie Max, Pure Data oder VVVV in die Welt der Software übertragen. Diese Tools lassen sich als Behälter für verschiedene Funktionen visualisieren, die sie bereitstellen. Indem sie diese Behälter miteinander verbinden, stellen die Nutzer dann *Patches* her. Diese Umgebung bietet dem Nutzer die Möglichkeit, mit der Programmierung zu experimentieren, ohne dabei ständig den Zyklus aus Programmeingabe, Kompilierung – verdammt, da ist ein Fehler –, Fehlerbehebung, Kompilierung und schließlich Programmausführung zu unterbrechen. Wenn du also eher visuell orientiert bist, empfehle ich dir, solche Entwicklungsumgebungen auszuprobieren.


1 2 3 4 5 6 7 8 9 10 11 12 13 14
Aux
1 2 3 4
L R
remix
INPUT
OUTPUT
INPUT
OUTPUT
INPUT
OUTPUT
INPUT
OUTPUT

Modifizieren von Schaltkreisen

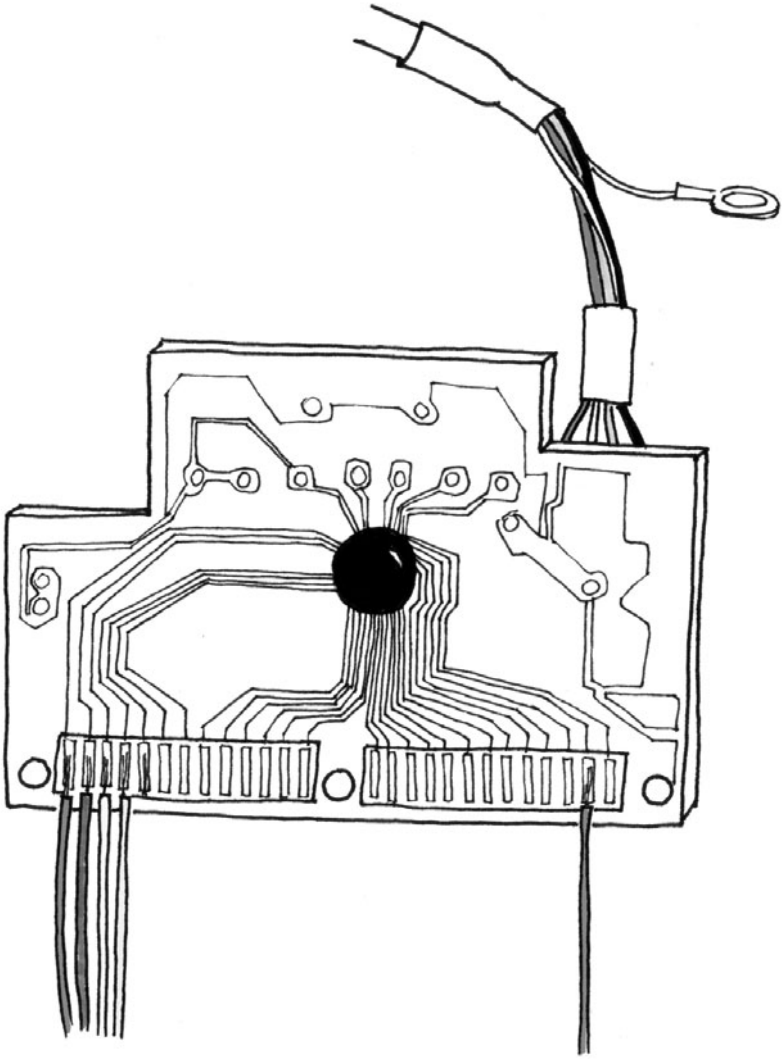


Modifizieren von Schaltkreisen ist eine der interessantesten Formen des Tüftelns. Es handelt sich um das kreative Kurzschließen von akustischen Geräten, die mit einer Niederspannungsbatterie betrieben werden, z. B. Pedale für Gitarreneffekte, Kinderspielzeug und kleine Synthesizer, um neue Musikinstrumente und Schallgeber zu kreieren. Das Herzstück dieses Prozesses ist die Kunst des Zufalls. Es begann 1966, als Reed Ghazala zufällig einen Spielzeugverstärker an einem Metallobjekt seiner Schreibtischschublade kurzschloss, woraus eine Flut ungewöhnlicher Töne resultierte. Das Modifizieren von Schaltkreisen ist eine hervorragende Möglichkeit, durch das Zweckentfremden oder Modifizieren von Technologie die wildesten Geräte zu erschaffen, ohne dass dazu notwendigerweise ein theoretisches Verständnis ihrer Funktion erforderlich wäre.



Es ist ein wenig wie beim Fanzine *Sniffin Glue*, aus dem hier ein Auszug zu sehen ist: Während der Punk-Ära reichte die Kenntnis von drei Gitarren-Akkorden aus, um eine Band zu gründen. Lasst nicht zu, dass Experten in einem bestimmten Bereich euch vermitteln, dass ihr niemals zu ihnen gehören werdet. Ignoriert sie und überrascht sie dann.

Keyboard-Hacks



Nach mehr als 60 Jahren sind Computertastaturen immer noch der am weitesten verbreitete Weg, mit dem Computer zu interagieren. Alex Pentland, der akademische Leiter des MIT Media Laboratory, bemerkte einst: "Ent-