

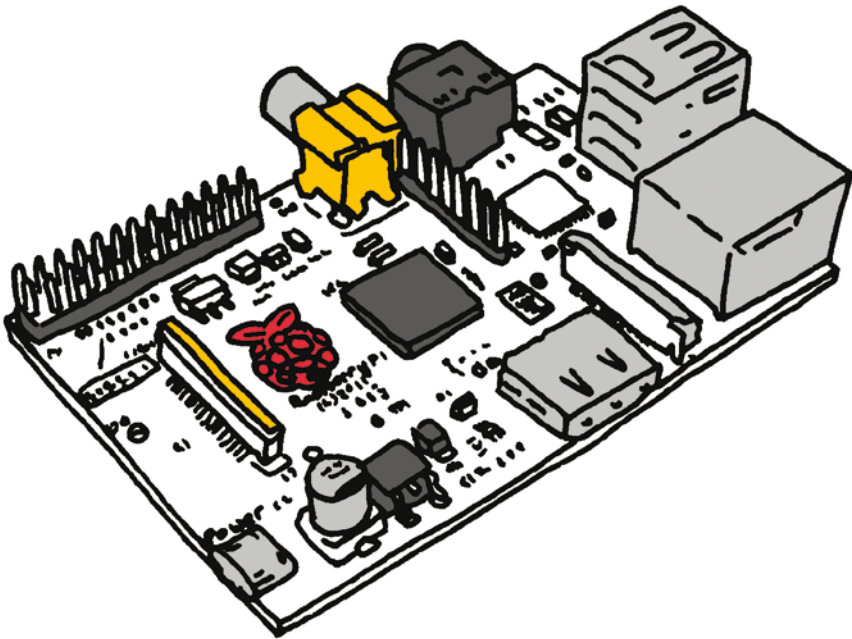


**DER
MINI-PC,
DER ALLES
KANN**

Raspberry Pi für Einsteiger

Matt Richardson & Shawn Wallace

Übersetzung von Thomas Demmig



Raspberry Pi für Einsteiger

**Matt Richardson &
Shawn Wallace**

Deutsche Übersetzung von
Thomas Demmig

O'REILLY®

Beijing · Cambridge · Farnham · Köln · Sebastopol · Tokyo

Die Informationen in diesem Buch wurden mit größter Sorgfalt erarbeitet. Dennoch können Fehler nicht vollständig ausgeschlossen werden. Verlag, Autoren und Übersetzer übernehmen keine juristische Verantwortung oder irgendeine Haftung für eventuell verbliebene Fehler und deren Folgen.

Alle Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt und sind möglicherweise eingetragene Warenzeichen. Der Verlag richtet sich im Wesentlichen nach den Schreibweisen der Hersteller. Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Alle Rechte vorbehalten einschließlich der Vervielfältigung, Übersetzung, Mikroverfilmung sowie Einspeicherung und Verarbeitung in elektronischen Systemen.

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

Copyright der deutschen Ausgabe:
© 2013 O'Reilly Verlag GmbH & Co. KG

Die Originalausgabe erschien 2012 unter dem Titel
Getting Started with Raspberry Pi
bei O'Reilly 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: Thomas Demmig, Frankfurt
Korrektur: Dr. Dorothee Leidig, Freiburg
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: Media-Print, Paderborn

ISBN 978-3-95561-147-7

Dieses Buch ist auf 100% chlorfrei gebleichtem Papier gedruckt.

Inhaltsverzeichnis

Vorwort	ix
1/Fahrt aufnehmen	1
Eine Tour über das Board	2
Das passende Zubehör	6
Das Gehäuse	8
Wählen Sie eine Distribution aus	9
Die SD-Karte flashen	11
Booten	12
Konfigurieren Sie Ihren Pi	14
Herunterfahren	16
Fehlersuche	16
Weitere Informationen	17
2/Linux auf dem Raspberry Pi	19
Arbeiten an der Befehlszeile	22
Dateien und das Dateisystem	23
Weitere Linux-Befehle	26
Prozesse	29
Sudo und Berechtigungen	29
Das Netzwerk	31
/etc	32
Datum und Uhrzeit setzen	33
Neue Software installieren	33
Weitere Informationen	34
3/Python auf dem Pi	35
Hallo Python	36
Ein bisschen mehr Python	39
Objekte und Module	41
Noch mehr Module	44
Fehlersuche	45
Weitere Informationen	46

4/Animation und Multimedia in Python	47
Hallo Pygame	47
Pygame-Surfaces	49
Auf Surfaces zeichnen	50
Ereignisse und Eingaben	51
Sprites	54
Klänge abspielen	55
Videos wiedergeben	57
Weitere Informationen	58
5/Scratch auf dem Pi	59
Hallo Scratch	59
Die Bühne	63
Noch zwei Dinge über Sprites	64
Ein größeres Beispiel: Astral Trespassers	66
Scratch und die reale Welt	71
Ihre Programme veröffentlichen	73
Weitere Informationen	74
6/Arduino und der Pi	75
Arduino in Raspbian installieren	76
Den seriellen Port herausfinden	78
Serielle Kommunikation	78
Weitere Informationen	82
7/Ein- und Ausgänge	83
Eingangs- und Ausgangsanschlüsse nutzen	85
Digitale Ausgabe: Eine LED zum Leuchten bringen	86
Digitaler Eingang: Einen Taster auslesen	90
Projekt: Cron-Lampenzeitschaltuhr	93
Befehle skripten	94
Eine Lampe anschließen	95
Befehle zeitgesteuert über cron ausführen	96
Mehr zu Cron	97
Weitere Informationen	98
8/Ein- und Ausgänge mit Python programmieren	99
GPIO in Python installieren und testen	99
Eine LED blinken lassen	102
Einen Taster auslesen	104
Projekt: Ein einfaches Soundboard	106
Weitere Informationen	109

9/Webcams einsetzen	111
Webcams testen	112
Installation und Test von SimpleCV	113
Ein Bild anzeigen	114
Ein Bild verändern	116
Zugriff auf die Webcam	118
Gesichtserkennung	120
Projekt: Raspberry Pi Photobooth	121
Weitere Informationen	124
10/Python und das Internet	125
Daten von einem Webserver laden	125
Die Wettervorhersage abfragen	127
Serving Pi (ein Webserver sein)	131
Grundlagen zu Flask	132
Das Web mit der realen Welt verbinden	135
Projekt: WebLamp	137
Weitere Informationen	141
A/Ein SD-Karten-Image schreiben	143
B/Vollständiger Code von Astral Trespassers	147
C/Analoger Input	153
Index	159

Vorwort

Es ist verständlich, dass die Leute bei der ersten Ankündigung des Raspberry Pi skeptisch waren. Ein kreditkartengroßer Computer für 35 Euro schien zu schön, um wahr zu sein. Das ist der Grund, warum der Raspberry Pi dann für so großes Entzücken gesorgt hat, als er wirklich verfügbar war.

Die Nachfrage war größer als das Angebot und die Wartelisten für diesen Minicomputer waren sehr lang. Aber abgesehen vom Preis — was ist an dem Raspberry Pi dran, dass die Leute so scharf darauf sind? Bevor wir uns anschauen, warum der Raspberry Pi so toll ist, wollen wir erklären, für wen er gedacht ist.

Eben Upton und seine Kollegen an der University of Cambridge stellten fest, dass die Studenten, die heutzutage Informatik studieren wollen, nicht mehr die Fertigkeiten mitbringen, die in den 1990er Jahren noch verbreitet waren. Sie führten das — neben anderen Faktoren — auf das „Aufkommen von Home-PCs und Spielekonsolen zurück, die die Amigas, BBC Micros, Spectrum ZX und Commodore 64er verdrängten, die die Generation zuvor noch nutzte, um das Programmieren zu lernen.“ Da der Computer für alle Mitglieder einer Familie wichtig wurde, hielt dies die Jüngeren davon ab, mit ihm herumzuexperimentieren und ein für alle so wichtiges Werkzeug eventuell lahmzulegen. Aber die neuen Prozessoren für Mobiltelefone und Tablets sind immer günstiger und gleichzeitig leistungsfähiger geworden. Das bereitete den Weg für den Raspberry Pi als supergünstigen, aber dennoch gut „bearbeitbaren“ Rechner. Wie der Vater von Linux, Linus Torvalds, einmal in einem Interview mit BBC News sagte: Mit dem Raspberry Pi ist es nun möglich, „Fehler zu machen“.

Was können Sie mit ihm machen?

Eine der tollen Seiten des Raspberry Pi ist, dass es nicht nur einen Weg gibt, ihn einzusetzen. Ob Sie mit ihm nur Videos sehen oder im Internet surfen wollen oder ob Sie mit ihm Hacken, Lernen und Basteln wollen — der Raspberry Pi ist eine flexible Plattform, mit der man Spaß haben kann, die nütz-

lich ist und die ein Experimentierumfeld bietet. Hier nur ein paar der Möglichkeiten, den Raspberry Pi einzusetzen:

Ein universell einsetzbarer Computer

Denken Sie daran: Der Raspberry Pi ist ein Computer, den Sie tatsächlich auch als Computer nutzen können. Nachdem Sie ihn in Kapitel 1 zum Laufen gebracht haben, können Sie sich dazu entscheiden, ihn mit einer grafischen Benutzeroberfläche und einem Webbrowser laufen zu lassen — was heutzutage ein häufiger Einsatzzweck für Computer ist. Aber abgesehen vom Webzugriff haben Sie eine umfangreiche Auswahl an freier Software, wie zum Beispiel die LibreOffice (<http://www.libreoffice.org/>)-Suite, mit der Sie Dokumente und Tabellenblätter auch ohne Internet-Verbindung bearbeiten können.

Programmieren lernen

Da der Raspberry Pi als Lernwerkzeug gedacht ist, mit dem Kinder experimentieren sollen, enthält er bereits Interpreter und Compiler für viele verschiedene Programmiersprachen. Für Anfänger gibt es Scratch, eine grafische Programmiersprache vom MIT, die wir in Kapitel 5 behandeln werden. Wollen Sie lieber direkt Code schreiben, können Sie mit der Programmiersprache Python einsteigen. Deren Grundlagen erklären wir in Kapitel 3. Und Sie sind nicht nur auf Scratch und Python beschränkt. Sie können Programme für Ihren Raspberry Pi in Sprachen wie C, Ruby, Java oder Perl schreiben.

Projektplattform

Der Raspberry Pi unterscheidet sich von einem normalen Computer nicht nur durch seinen Preis und seine Größe, sondern auch durch seine Fähigkeit, gut in elektronische Projekte eingebunden werden zu können. Ab Kapitel 7 zeigen wir Ihnen, wie Sie mit dem Raspberry Pi LEDs steuern und Leitungen schalten können. Zudem lernen Sie dort, wie Sie den Status von Schaltern und Tasten ermitteln.

Raspberry Pi für Bastler

Als Bastler können wir unter einer ganzen Reihe von Plattformen auswählen, wenn wir technologiebasierte Projekte bauen wollen. In letzter Zeit haben Entwicklerboards mit Mikrocontroller an Beliebtheit gewonnen, wie zum Beispiel der Arduino, da man sehr einfach mit ihnen arbeiten kann. Aber *System-on-a-Chip*-Plattformen wie der Raspberry Pi unterscheiden sich von den klassischen Mikrocontrollern in vielerlei Hinsicht. Tatsächlich hat der Raspberry Pi mehr mit Ihrem Computer gemeinsam als mit einem Arduino.

Wir wollen damit nicht sagen, dass ein Raspberry Pi besser ist als ein klassischer Mikrocontroller — er ist nur anders. Wollen Sie zum Beispiel ein einfaches Thermostat bauen, fahren Sie aus Gründen der Einfachheit vermutlich besser mit einem Arduino Uno oder einem ähnlichen Mikrocontroller. Aber

wenn Sie per Web aus der Ferne auf das Thermostat zugreifen wollen, um seine Einstellungen zu ändern und die Logdateien mit den Temperaturdaten herunterzuladen, sollten Sie eher darüber nachdenken, einen Raspberry Pi einzusetzen.

Die richtige Wahl hängt sehr von den Anforderungen Ihres Projekts ab, und tatsächlich müssen Sie sich auch gar nicht für einen von beiden entscheiden. In Kapitel 6 zeigen wir Ihnen, wie Sie den Raspberry Pi nutzen können, um den Arduino zu programmieren, und wie beide miteinander kommunizieren.

Mit dem Lesen dieses Buches werden Sie ein besseres Verständnis für die Stärken des Raspberry Pi und seine Einsatzmöglichkeiten für Bastler erhalten.

Das ist noch nicht alles!

Sie können so viel mit dem Raspberry Pi anstellen — wir können das alles gar nicht in einem Buch unterbringen. Hier noch zwei Einsatzmöglichkeiten:

Media Center

Da der Raspberry Pi sowohl HDMI- als auch Composite-Videoausgänge besitzt, lässt er sich einfach an Fernseher anschließen. Er besitzt auch ausreichend CPU-Leistung, um HD-Videos im Vollbild abzuspielen. Um das zu unterstützen, haben die Beteiligten des Open Source Media Player XBMC (<http://xbmc.org/>) ihr Projekt auf den Raspberry Pi portiert. XBMC kann viele verschiedene Medienformate abspielen und seine Benutzeroberfläche ist mit großen Buttons und Texten ausgestattet, so dass sie sich auch leicht vom Sofa aus bedienen lässt. XBMC verwandelt den Raspberry Pi in eine umfassend konfigurierbare Komponente für ein Home Entertainment Center.

„Bare Metal“ Computer-Hacking

Die meisten Leute, die Computerprogramme schreiben, schreiben Code, der innerhalb eines Betriebssystems läuft, wie zum Beispiel Windows, Mac OS oder — beim Raspberry Pi — Linux. Aber was wäre, wenn Sie Code schreiben könnten, der ohne ein Betriebssystem direkt auf dem Prozessor läuft? Sie könnten sich sogar Ihr eigenes Betriebssystem schreiben, wenn Sie daran Spaß haben. Das Computer Laboratory der University of Cambridge hat einen kostenlosen Online-Kurs (<http://www.cl.cam.ac.uk/freshers/raspberrypi/tutorials/os/>) veröffentlicht, in dem Sie Schritt für Schritt erfahren, wie Sie mit Assembly-Code Ihr eigenes Betriebssystem erstellen.

Linux und Raspberry Pi

Auf einem klassischen Computer läuft ein Betriebssystem, wie zum Beispiel Windows, OS X oder Linux. Es wird gestartet, wenn Sie Ihren Computer starten, und es ermöglicht Ihren Anwendungen Zugriff auf die Hardware-Funktionen Ihres PCs. Schreiben Sie zum Beispiel eine Anwendung, die auf das Internet zugreift, können Sie dazu die Funktionen des Betriebssystems nutzen. Sie müssen nicht jede mögliche Ethernet- oder WLAN-Hardware kennen und eigens ansprechen, die es in der Welt gibt.

Wie jeder andere Computer nutzt auch der Raspberry Pi ein Betriebssystem — standardmäßig *Raspbian*, eine Linux-Variante. Linux ist für den Raspberry Pi die perfekte Wahl, weil es frei und Open Source ist. Der Preis für die Plattform wird dadurch gering gehalten und gleichzeitig hat man mehr Eingriffsmöglichkeiten. Sie sind aber nicht auf Raspbian beschränkt, es gibt viele andere Linux-*Distributionen*, die Sie einsetzen können. Es gibt sogar ein paar Nicht-Linux-Varianten. In diesem Buch werden wir aber die Standard-Raspbian-Distribution einsetzen, die auf der Download-Seite des Raspbian Pi (<http://www.raspberrypi.org/downloads>) verfügbar ist.

Sind Sie mit Linux nicht vertraut, brauchen Sie sich keine Sorgen zu machen. In Kapitel 2 werden Sie die Grundlagen kennenlernen, mit denen Sie die Themen im Buch bearbeiten können.

Was andere mit dem Raspberry Pi gemacht haben

Wenn Sie mit einer spannenden neuen Technologie arbeiten, können Sie sich manchmal nicht entscheiden, was Sie damit anstellen wollen. Wenn Sie noch unschlüssig sind, gibt es zum Glück keinen Mangel an Ideen für interessante und kreative Raspberry-Pi-Projekte, von denen Sie sich inspirieren lassen können. Als Herausgeber von MAKE haben wir viele tolle Anwendungen für den Raspberry Pi kennengelernt, von denen wir einige unserer Favoriten vorstellen wollen.

Spielautomaten-Sofatisch

(<http://www.instructables.com/id/Coffee-Table-Pi/>)

Der Instructables-User grahamgelding hat eine Schritt-für-Schritt-Anleitung veröffentlicht, wie man einen Sofatisch so umbaut, dass er als Basis für einen Spielautomaten-Emulator mit dem Raspberry Pi dient. Damit die Spiele auf dem Pi laufen, hat er den MAME (Multiple Arcade Machine Emulator) genutzt – ein Open-Source-Softwareprojekt, mit dem Sie klassische Automaten Spiele auf modernen Computern laufen lassen können. Im Tisch ist ein 24-Zoll-LCD-Bildschirm eingebaut, der mit dem Raspberry Pi per HDMI verbunden ist, dazu klassische Joystick-Knöpfe und der eigentliche Joystick, die mit den GPIO-Pins des Pi verbunden sind und als Eingabequelle dienen.

RasPod (<https://github.com/lionaneesh/RasPod>)

Aneesh Dogra, ein Teenager aus Indien, war auf einem der vorderen Plätze beim 2012 Summer Coding Contest der Raspberry Pi Foundation. Er hat RasPod gebaut, ein über das Web steuerbarer MP3-Player auf Basis des Raspberry Pi. Mit Hilfe von Python und einem Web-Framework namens Tornado können Sie sich mit dem RasPod aus der Ferne am Raspberry Pi anmelden, um die Musik abzuspielen und wieder zu stoppen, die Lautstärke anzupassen, Lieder auszuwählen und Playlists zu erstellen. Die Musik kommt aus einer der Audio-Buchsen des Raspberry Pi, so dass Sie ein Paar PC-Lautsprecher oder Ihre Stereoanlage nutzen können, um der Musik zu lauschen.

Raspberry-Pi-Supercomputer (http://www.southampton.ac.uk/mediacentre/features/raspberry_pi_supercomputer.shtml)

Viele Supercomputer sind aus Clustern mit Standard-PCs aufgebaut, die miteinander verknüpft sind. Die Rechen-Jobs werden unter allen Prozessoren aufgeteilt. Eine Gruppe von Informatikern an der University of Southampton hat 64 Raspberry Pis miteinander verbunden, um einen günstigen Supercomputer aufzubauen. Seine Leistung reicht natürlich nicht annähernd an die heutiger Top-Supercomputer heran, aber er zeigt die Prinzipien bei der Entwicklung solcher Systeme. Das Tüpfelchen auf dem i ist, dass das Racksystem für sämtliche Raspberry Pis mit den Legosteinen des sechsjährigen Sohnes des Gruppenleiters gebaut wurde.

Wenn Sie mit Ihrem Raspberry Pi etwas Interessantes tun, lassen Sie es uns wissen. Sie können Ihre Projekte bei der Redaktion des MAKE einreichen, indem Sie das Formular auf [makezine.com](http://blog.makezine.com/contribute/) (<http://blog.makezine.com/contribute/>) ausfüllen.

In diesem Buch genutzte 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 weist auf einen Tipp, einen Vorschlag oder eine allgemeine Anmerkung hin.



Dieses Symbol weist auf eine Warnung hin.

Wie Sie uns erreichen

Bitte wenden Sie sich mit Fragen und Anmerkungen zu diesem Buch an den Verlag:

MAKE
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

MAKE bringt eine wachsende Gemeinde einfallsreicher Leute zusammen, versorgt sie mit Informationen und inspiriert sie. Diese Leute arbeiten in der Garage, im Hof oder im Keller an spannenden Projekten. MAKE baut auf Ihrem Recht auf, jegliche Technologie anzupassen, zu hacken oder zu Ihren Gunsten zu verändern. Rund um MAKE wächst eine Kultur und Community, die daran glaubt, dass wir selbst, unsere Umgebung und unser Bildungssystem besser werden kann – ja sogar unsere ganze Welt. Es ist viel mehr als eine Leserschaft – es ist eine weltweite Bewegung, die MAKE anführt. Wir nennen sie „Maker Movement“.

Mehr Informationen über MAKE finden Sie online:

MAKE Magazine: <http://makezine.com/magazine/>
Maker Faire: <http://makerfaire.com>
Makezine.com: <http://makezine.com>
Maker Shed: <http://makershed.com/>

Wir haben eine (englischsprachige) Webseite zu diesem Buch, auf der Sie Korrekturen von Fehlern, Beispiele und weitere Informationen finden. Sie erreichen sie über

<http://shop.oreilly.com/product/0636920023371.do>

Kommentare und technische Fragen zum Buch können Sie in einer E-Mail an folgende Adresse schicken:

E-Mail: bookquestions@oreilly.com

Danksagungen

Wir möchten uns bei ein paar Leuten bedanken, die uns Wissen, Unterstützung, Rat und Rückmeldung zu *Raspberry Pi für Einsteiger* gaben:

Brian Jepson
Marc de Vinck
Eben Upton
Tom Igoe
Clay Shirky
John Schimmel
Phillip Torrone
Limor Fried
Kevin Townsend
Ali Sajjadi
Andrew Rossi

1/Fahrt aufnehmen

Ein paar Begriffe tauchen immer wieder auf, wenn sich Leute über den Raspberry Pi unterhalten: klein, günstig, hackbar, für die Ausbildung geeignet. Eines ist er aber nicht: aus dem Stand lauffähig. Es ist nicht so schwierig, ihn in einen Fernseher einzustöpseln und etwas angezeigt zu bekommen. Aber der Pi ist kein Consumer-Gerät. Abhängig von Ihren Zielen werden Sie eine Reihe von Entscheidungen zur Peripherie und Software Ihres Raspberry Pi treffen müssen.

Natürlich brauchen Sie als Allererstes den Raspberry Pi selbst. Wahrscheinlich haben Sie jetzt schon einen, aber wenn nicht, hat die Raspberry Pi Foundation Vereinbarungen mit ein paar Herstellern getroffen, bei denen Sie einen Pi direkt zum üblichen Preis von 35 bis 40 Euro kaufen können. Es sind:

Premier Farnell/Element 14

(<http://www.element14.com/community/groups/raspberry-pi/>)

Ein britischer Elektronikhändler mit vielen Vertretungen auf der ganzen Welt (auch in Deutschland)

RS Components (<http://www.rs-components.com/rasberrypi>)

Ein weiterer aus Großbritannien stammender Elektronikhändler

Der niedrige Preis des Raspberry Pi ist natürlich ein zentraler Bestandteil der ganzen Geschichte. Es ist ungewöhnlich, dass normale Privatanutzer direkt bei einem Distributor wenige Geräte zu dem gleichen Preis kaufen können wie große Reseller. Einige potenzielle Reseller waren vom Preis in den ersten Ankündigungen verwirrt – sie konnten sich nicht vorstellen, damit noch Gewinn zu machen. Das ist der Grund, warum manche Reseller noch etwas auf die 35 Euro aufschlagen (meist bis 40 Euro). Die Allgemeinheit kann immer noch direkt bei den erwähnten Distributoren kaufen, aber die Retailer und Reseller liefern häufig schneller.

Genug Mikroelektronik-Klatsch – lassen Sie uns das Board des Raspberry Pi endlich genauer anschauen.

Eine Tour über das Board

Lassen Sie uns kurz aufzählen, was Sie im Karton mit dem Raspberry Pi finden.

Man macht es sich zu einfach, wenn man den Raspberry Pi als ein Mikrocontroller-Entwicklungsboard wie den Arduino oder als Laptop-Ersatz ansieht. Tatsächlich handelt es sich eher um die Innereien eines mobilen Geräts mit vielen Bastler-freundlichen Anschlussmöglichkeiten für die verschiedenen Ports und Funktionen. In Abbildung 1-1 sehen Sie alle unten beschriebenen Teile des Boards.

- A. *Der Prozessor* Das Herz des Raspberry Pi enthält den gleichen Prozessor wie das iPhone 3G oder der Kindle 2, Sie können den Raspberry Pi leistungstechnisch also durchaus mit diesen Geräten vergleichen. Es handelt sich um ein 32-Bit-System mit 700 MHz, basierend auf der ARM11-Architektur. ARM-Chips gibt es in verschiedenen Architekturen mit unterschiedlich vielen Cores konfiguriert, um eine Reihe von Leistungsebenen (und Preisstufen) anzusprechen. Das Modell B besitzt 512 MB RAM, das Modell A 256 MB. (Die erste Charge des Modells B hat auch nur 256 MB.)
- B. *Der Secure Digital (SD-)Kartenslot* Sie werden bemerken, dass es keine Festplatte auf dem Pi gibt – alles wird auf einer SD-Karte gespeichert. Ein Grund für ein schützendes Gehäuse für Ihren Pi ist, dass die Kontakte auf dem SD-Socket eventuell nicht mehr funktionieren, wenn die SD-Karte unabsichtlich verbogen wird.
- C. *Der USB-Port* Beim Modell B gibt es zwei USB-2.0-Ports, während das Modell A nur einen besitzt. Einige der frühen Raspberry Pis waren bei der möglichen Stromzufuhr am USB-Port beschränkt. Manche USB-Geräte können bis zu 500 mA ziehen. Das erste Pi-Board hat nur etwa 100 mA bereitgestellt, neuere Versionen erfüllen hier die USB-2.0-Spezifikation vollständig. Um Ihr Board zu überprüfen, kontrollieren Sie, ob es zwei *Polyfuses* (selbstrückstellende Sicherungen) besitzt, die den Strom begrenzen (siehe Abbildung 1-2). Auf jeden Fall ist es keine allzu gute Idee, Ihr Mobiltelefon mit dem Pi laden zu wollen. Verwenden Sie einen externen Hub mit eigener Stromversorgung, wenn Sie ein USB-Gerät mit mehr Strombedarf nutzen wollen.
- D. *Ethernet-Port* Das Modell B besitzt einen Standard-RJ45-Ethernetport. Das Modell A hat den nicht, kann aber über einen USB-Ethernet-Adapter mit einem Netzwerk verbunden werden (der Port auf Modell B ist eigentlich auch nicht mehr als ein USB-Ethernet-Adapter auf dem Board selbst). Eine andere Möglichkeit ist eine WLAN-Verbindung über einen USB-Stick.
- E. *HDMI-Anschluss* Der HDMI-Port bietet eine digitale Video- und Audio-Ausgabe. Es werden 14 verschiedene Video-Auflösungen unterstützt und das HDMI-Signal kann nach DVI (von vielen Monitoren genutzt),

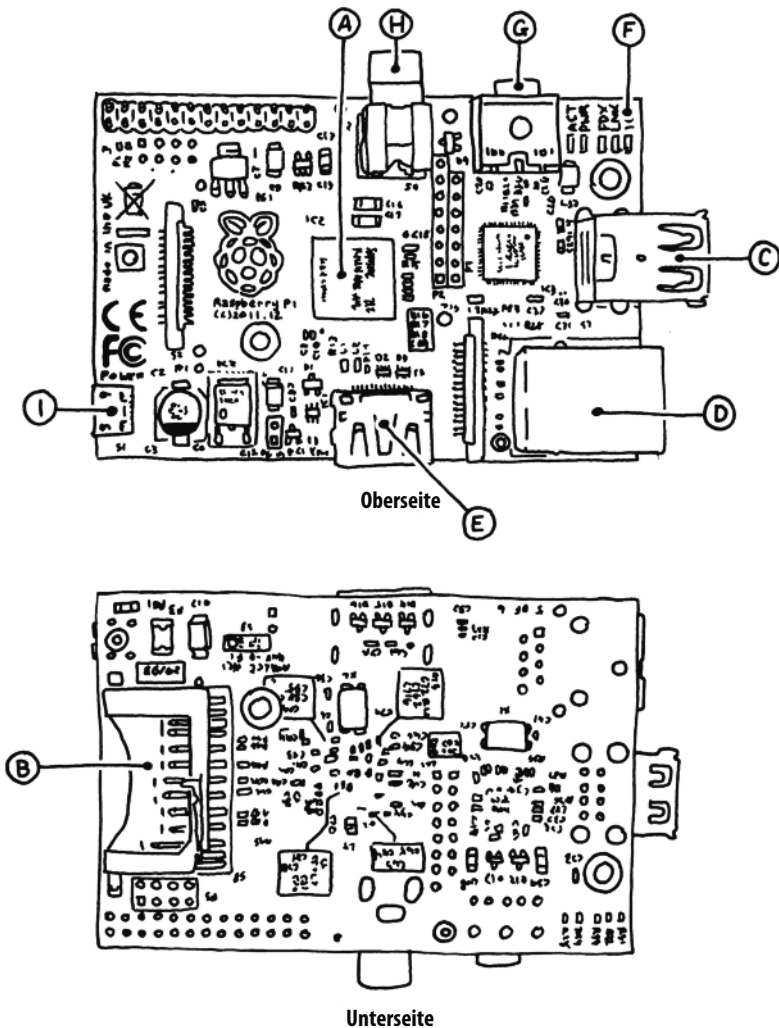


Abbildung 1-1. Die Platine des Raspberry Pi

Composite (analoges Videosignal über den gelben Cinch-Stecker) oder SCART (ein europäischer Standard für das Verbinden von Audio- und Video-Geräten) umgewandelt werden.

F. *Status-LEDs* Der Pi besitzt fünf LEDs, die eine visuelle Rückmeldung ermöglichen (siehe Tabelle 1-1).



Abbildung 1-2. Einige der älteren Boards besaßen Polyfuses (links), um den USB-Port zu schützen. Bei manchen Boards wurden diese Sicherungen durch Jumper ersetzt (Mitte), und bei der neuesten Version des Modells B sind sie ganz verschwunden und der Platz wurde für eine Montagebohrung genutzt (rechts).

Tabelle 1-1. Die fünf Status-LEDs

ACT	grün	Leuchtet, wenn auf die SD-Karte zugegriffen wird (auf älteren Boards mit OK gekennzeichnet)
PWR	rot	Verbunden mit dem 3,3 V-Anschluss
FDX	grün	Leuchtet, wenn der Netzwerk-Adapter im Full-Duplex-Modus läuft
LNK	grün	Netzwerk-Aktivität
100	gelb	Ist an, wenn die Netzwerk-Verbindung mit 100 Mbps läuft (auf manchen älteren Boards steht fälschlicherweise 10M)

- G. *Analoger Audio-Ausgang* Dies ist ein Standard-3,5-mm-Ausgang, der für eine hohe Impedanz gedacht ist (zum Beispiel für Aktiv-Lautsprecher). Kopfhörer oder nicht verstärkte Lautsprecher werden nicht sehr gut klingen. Aktuell ist die Qualität der analogen Audioausgabe deutlich geringer als die HDMI-Audioausgabe, die Sie durch das Anschließen eines Fernsehers über HDMI erhalten. Zum Teil ist dafür die Audiotreiber-Software verantwortlich, die noch weiterentwickelt wird.
- H. *Composite-Video-Ausgang* Dies ist eine Standard-Cinch-Buchse für Composite-NTSC- oder -PAL-Videosignale. Dieses Videoformat ist verglichen mit HDMI sehr schlecht aufgelöst. Besitzen Sie einen Fernseher oder Monitor mit HDMI-Anschluss, sollten Sie auf jeden Fall den digitalen Ausgang wählen.
- I. *Stromversorgung* So ziemlich das Erste, was Ihnen am Pi auffallen wird, ist, dass er keinen Netzschalter besitzt. Diese MicroUSB-Buchse wird genutzt, um die Platine mit Strom zu versorgen (das ist kein zusätzlicher USB-Port, sondern dient nur der Stromversorgung). MicroUSB wurde gewählt, weil die Buchse günstig ist und USB-Stromversorgungen leicht erhältlich sind.

Abbildung 1-3 zeigt alle Input/Output- (I/O-) und Stromversorgungs-Pins auf dem Raspberry Pi, die im Folgenden beschrieben werden.

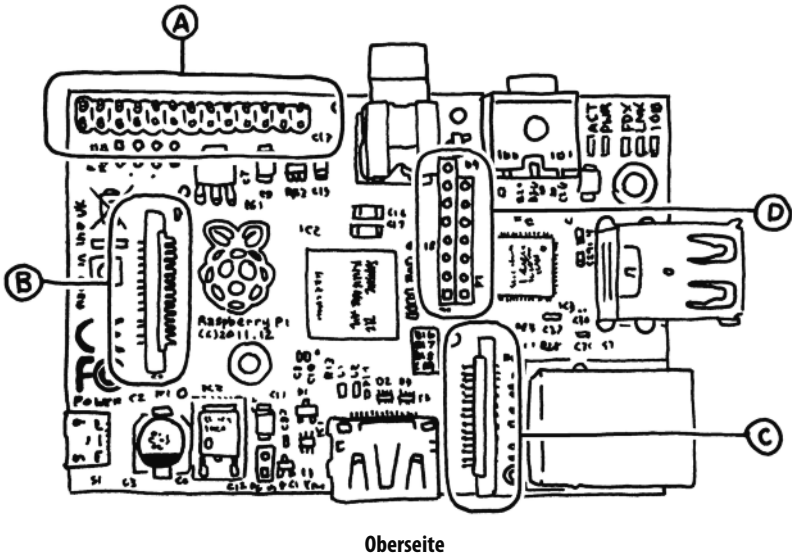


Abbildung 1-3. Die Pins und Steckleisten des Raspberry Pi

- A. *General Purpose Input and Output (GPIO) und andere Pins* In Kapitel 7 und Kapitel 8 erfahren Sie, wie Sie diese Pins nutzen können, um Taster und Schalter auszulesen und Aktuatoren wie LEDs, Relais oder Motoren zu steuern.
- B. *Der Display-Serial-Interface-(DSI-)Anschluss* Hier kann man ein 15-adriges Flachbandkabel anschließen, um den Pi mit einem LCD- oder OLED-Bildschirm zu verbinden.
- C. *Der Camera-Serial-Interface-(CSI-)Anschluss.* Über diesen Port kann man ein Kameramodul direkt mit dem Board verbinden.
- D. *P2- und P3-Stiftleiste.* Diese zwei Leisten sind die JTAG-Teststiftleisten für den Broadcom-Chip (P2) und den LAN9512-Netzwerkchip (P3). Da das Chipset von Broadcom proprietär ist, werden diese Stiftleisten für Sie nicht wirklich von Nutzen sein.



Im Herbst 2012 hat die Raspberry Pi Foundation eine neue Version des Boards veröffentlicht, auf dem sich ein paar neue Hardware-Features befinden – unter anderem zwei 2,5 mm große Befestigungslöcher und eine Stiftleiste für einen Reset-Schalter. Es gibt zudem eine nicht angeschlossene 2x4-Stift-

leiste in der Nähe der GPIO-Pins, die für Clock- und Audio-Boards von Fremdherstellern gedacht ist, die direkt auf der Hauptplatine befestigt werden sollen.

Das passende Zubehör

Nachdem Sie nun gelernt haben, wo sich was auf dem Board befindet, müssen Sie noch ein paar Dinge über das passende Zubehör wissen (manches ist in Abbildung 1-4 zu sehen), das Sie mit Ihrem Pi nutzen können. Es gibt eine ganze Reihe von Starter Kits, die gut ausgestattet sind. Allerdings gibt es auch ein paar Stolpermöglichkeiten. Auf jeden Fall gibt es eine Liste unterstützter Peripheriegeräte (http://elinux.org/RPi_VerifiedPeripherals) auf dem Haupt-Wiki.

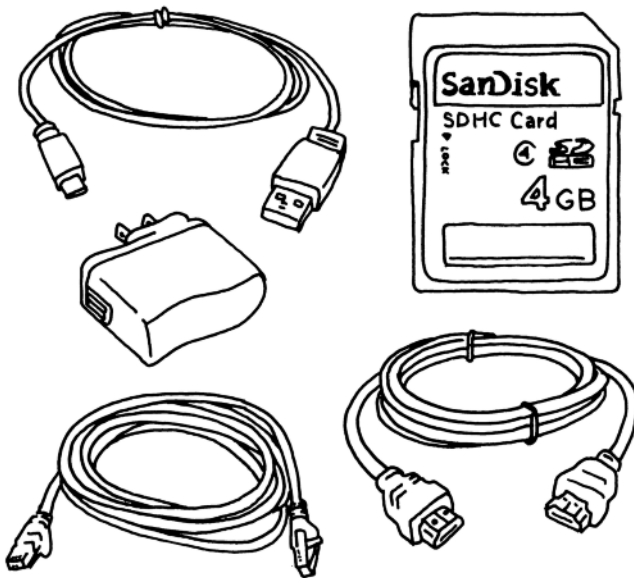


Abbildung 1-4. Die wichtigsten Peripherie-Geräte: Eine MicroUSB-Stromversorgung, Kabel und eine SD-Karte. Sie benötigen mindestens eine Class-4-SD-Karte mit 4 GB (es ist okay, MicroSD-Karten mit einem Adapter einzusetzen). No-Name-SD-Karten sind bekannt für ihre sehr variierende Qualität, suchen Sie sich daher lieber eine Markenkarte (siehe http://elinux.org/RPi_VerifiedPeripherals#SD_cards).

- A. *Stromversorgung* Dies ist das wichtigste Zubehör für Sie. Sie sollten einen MicroUSB-Adapter nutzen, der 5 V und mindestens 700 mA liefern kann (für das Modell A reichen 500 mA). Ein Ladegerät für ein Mobiltelefon reicht hier nicht, da der typische Vertreter dieser Gattung nur 400 mA oder weniger bietet. Werfen Sie aber im Zweifelsfall einen Blick auf die Angaben des Ladegeräts. Ein Pi, der mit zu wenig Strom versorgt wird, scheint meist noch zu laufen, ist aber instabil und liefert unvorhersehbare Fehler.
-



In der aktuellen Version des Pi-Boards ist es möglich, den Pi von einem USB-Hub aus mit Strom zu versorgen. Allerdings sind hier die Schutzmechanismen nicht sehr ausgeprägt, daher sollte man diesen Weg eher vermeiden. Das gilt insbesondere dann, wenn Sie elektronische Basteleien mit dem Board anstellen und dabei unabsichtlich Kurzschlüsse verursachen, die viel Strom ziehen können.

- B. *Eine SD-Karte* Sie brauchen mindestens 4 GB, und es sollte eine Class-4-Karte sein. Class-4-Karten können mindestens 4 MB/s übertragen. Einige der älteren Raspberry-Pi-Boards hatten Probleme mit Class 6 oder höher, die schneller übertragen können, aber nicht so stabil sind. Eine MicroSD-Karte in einem Adapter lässt sich ebenfalls nutzen.
- C. *Ein HDMI-Kabel* Für die Verbindung mit einem Monitor benötigen Sie solch ein Kabel oder einen passenden Adapter für einen DVI-Monitor. Sie können den Pi auch ohne Display laufen lassen (siehe weiter unten). HDMI-Kabel gibt es in einer sehr großen Preisspanne. Wenn das Kabel zum Monitor nur ein bis zwei Meter lang sein muss, müssen Sie nicht mehr als 5 Euro dafür ausgeben. Bei längeren Kabeln sollten Sie auf jeden Fall ein qualitativ hochwertiges nehmen.
- D. *Ethernet-Kabel* Bei Ihnen zu Hause gibt es vermutlich längst nicht mehr so viele Ethernet-Buchsen wie noch vor fünf Jahren – mittlerweile ist doch vieles schon drahtlos. Daher nervt Sie der kabelgebundene Anschluss vielleicht ein wenig. In Abschnitt „Headless laufen“ (S. 32) werden ein paar Alternativen zum direkt mit einem Hub oder einer Netzwerkdose verbundenen Ethernet-Kabel aufgeführt.

Wollen Sie mit Ihrem Raspberry Pi noch viel mehr machen, gibt es weiteres interessantes Zubehör und Addons, über die wir in Kapitel 5 sprechen. Schauen Sie sich auch ein paar der folgenden Addons an (unter http://elinux.org/RPi_VerifiedPeripherals erhalten Sie eine Liste der Peripheriegeräte, die bekanntermaßen funktionieren):

Ein USB-Hub mit Stromversorgung

Es wird ein USB-2.0-Hub empfohlen.

Kühlkörper

Ein Kühlkörper ist ein kleines Stück Metall, oft mit Lamellen, um eine große Oberfläche zu erhalten und die Wärme besser ableiten zu können. Kühlkörper lassen sich an Chips anbringen, die heiß werden. Das Chipset des Pi ist für einen mobilen Einsatz entworfen, daher ist meist kein Kühlkörper nötig. Aber wie wir später noch sehen werden, gibt es Fälle, in denen Sie den Pi mit einer höheren Geschwindigkeit laufen oder über längere Zeit intensiv rechnen lassen wollen. Dann erhitzt sich der Chip etwas. Manche Leute haben berichtet, dass sich auch der Netzwerk-Chip erwärmen kann.

Echtzeituhr

Eventuell wollen Sie einen Echtzeituhr-Chip einsetzen (wie den DS1307), um korrekte Zeiten zu protokollieren, wenn der Pi offline ist.

Kameramodul

Anfang 2013 wird ein offizielles 5-Megapixel-Kameramodul für den Raspberry Pi verfügbar sein. Bis dahin können Sie eine USB-Webcam nutzen (in Kapitel 9 finden Sie ein vollständiges Beispiel).

LCD-Display

Viele LCDs lassen sich mit ein paar Verbindungen zur GPIO-Steckerleiste anschließen. LCDs, die die DSI-Schnittstelle nutzen, werden 2013 verfügbar sein.

WLAN-USB-Stick

Viele WLAN-USB-Sticks arbeiten mit dem Pi zusammen. Halten Sie am besten Ausschau nach einem, der nicht so viel Strom benötigt.

Laptop-Dock

Eine Reihe von Leuten hat Laptop-Docks angepasst, die für Mobiltelefone gedacht sind (wie das Atrix Lapdock), um als Anzeige und Halter für den Raspberry Pi zu dienen.

Das Gehäuse

Sie werden sehr bald ein Gehäuse für Ihren Raspberry Pi haben wollen. Die steifen Kabel auf allen Seiten sorgen dafür, dass das Board nur schwer flach hinzulegen ist, und manche Komponenten (wie zum Beispiel der SD-Kartenslot) können schon beim normalen Einsatz mechanisch beschädigt werden.

Der Pi besitzt sechs Schichten mit Leiterbahnen, die die diversen Komponenten miteinander verbinden – im Gegensatz zu einfachen Mikrocontroller-Platinen, die die Leiterbahnen nur oben und unten haben. Neben den beiden Oberflächen sind vier zusätzliche Schichten mit Leiterbahnen wie bei einem Sandwich vorhanden. Wird das Board zu sehr gebogen, können manche dieser Verbindungen so brechen, dass es fast unmöglich ist, den Fehler zu finden. Die Lösung: ein Gehäuse.

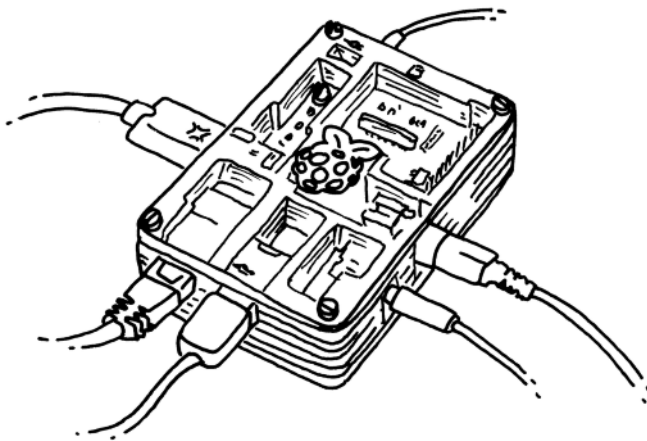


Abbildung 1-5. *Das farbenfrohe Pibow-Gehäuse*

Es gibt eine ganze Reihe fertiger Gehäuse, aber auch Gehäusedesigns, die man sich herunterladen und mit einem Laser-Cutter oder einem 3D-Drucker selbst herstellen kann. Vermeiden Sie am besten Gehäuse, bei denen bruchempfindliches Acrylglas rechtwinklig zusammengesetzt ist. Das Schichtsystem von Pibow (<http://pibow.com/>) (Abbildung 1-5) ist da eine farbenfrohe Alternative.

Wir brauchen es vermutlich nicht wirklich zu erwähnen, aber es ist einer der offensichtlichen Fehler, der einem gelegentlich widerfährt: Legen Sie Ihren Raspberry Pi nicht auf eine leitende Oberfläche. Drehen Sie das Board einmal um und schauen Sie sich den Boden an – es gibt hier eine ganze Reihe Komponenten und viele Lötstellen, die sich problemlos kurzschließen lassen. Dies ist ein weiterer Grund, warum Sie Ihrem Pi ein Gehäuse spendieren sollten!

Wählen Sie eine Distribution aus

Der Raspberry Pi nutzt Linux als Betriebssystem. Technisch gesehen ist Linux nur der Kernel, aber ein Betriebssystem ist mehr als das: Es handelt sich um eine Zusammenstellung von Treibern, Diensten und Anwendungen. Im Laufe der Jahre wurden eine ganze Reihe von Linux-Distributionen oder -Variationen entwickelt, die auf dem Pi laufen. Einige der bekanntesten auf Desktop-Computern sind Ubuntu, Debian, Fedora und Arch. Jede hat ihre eigene Community mit Anwendern und ist für bestimmte Programme besonders gut geeignet.

Da der Pi auf einem Chipset für mobile Geräte aufbaut, hat er andere Anforderungen an die Software als ein Desktop-Computer. Der Broadcom-

Prozessor besitzt ein paar spezielle Eigenschaften, die besondere „Binary Blob“-Gerätetreiber erfordern. Dazu kommt Code, der sich so nicht in einer Standard-Linux-Distribution findet. Und während die meisten Desktop-Computer über RAM im Gigabyte-Bereich und hunderte Gigabyte Festplattenplatz verfügen, ist der Pi in beidem beschränkt. Daher wurden spezielle Linux-Distributionen entwickelt, die den Pi als Zielpattform haben. Zu den am weitesten verbreiteten gehören:

Raspbian (<http://raspbrian.org>)

Die „offiziell empfohlene“ Distribution der Foundation, basierend auf Debian (siehe Abbildung 1-6). Beachten Sie, dass raspbian.org (<http://raspbrian.org>) eine Community-Site ist, die nicht von der Foundation betrieben wird. Wenn Sie nach der offiziellen Distribution suchen, schauen Sie auf der Downloads (<http://raspberrypi.org>)-Seite unter raspberrypi.org vorbei.

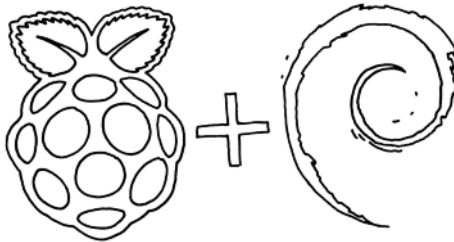


Abbildung 1-6. *Raspberry Pi + Debian = Raspbian*

Adafruit Raspberry Pi Educational Linux (Occidentalis)

(<http://learn.adafruit.com/adafruit-raspberry-pi-educational-linux-distro>)

Dies ist die Raspbian-basierte Distribution von Adafruit, die Tools und Treiber enthält, mit denen der Elektronikunterricht erleichtert wird.

Arch Linux (<http://www.archlinux.org/>)

Arch Linux ist speziell für ARM-basierte Computer entwickelt, daher wurde der Pi schon sehr frühzeitig unterstützt.

Xbian (<http://xbian.org/>)

Dies ist eine Raspbian-basierte Distribution für Anwender, die den Raspberry Pi als Media Center nutzen wollen (siehe auch OpenELEC (<http://openelec.tv/>) und Raspbmc (<http://www.raspbmc.com>)).

QtOnPi (<http://qt-project.org/wiki/Qt-RaspberryPi>)

Eine Distribution, die auf dem Qt-5-Framework basiert.

In diesem Buch werden wir uns auf die offizielle Raspbian-Distribution beschränken.

Die SD-Karte flashen

Viele Händler verkaufen SD-Karten mit einem vorinstallierten Betriebssystem. Das ist für manche vielleicht die beste Einstiegsmöglichkeit. Auch wenn nicht das neueste Release darauf zu finden ist, können Sie leicht upgraden, sobald der Pi einmal gestartet wurde und mit dem Internet verbunden ist.

Raspbian besitzt auch einen Network Installer (<http://www.raspbian.org/RaspbianInstaller>). Um ihn zu verwenden, müssen Sie die Installer-Dateien auf eine SD-Karte kopieren (formatiert als FAT32, was bei diesen Karten üblich ist) und dann den Pi mit der eingesteckten Karte booten. Damit das funktioniert, müssen Sie aber mit dem Internet verbunden sein.

Zuerst müssen Sie Raspbian von der Downloads-Seite von [raspberrypi.org](http://www.raspberrypi.org) (<http://www.raspberrypi.org/downloads>) herunterladen. Das Betriebssystem wird als Disk Image bereitgestellt, bei dem es sich um eine 1:1-Repräsentation der Daten handelt, die auf die SD-Karte geschrieben werden sollten.

Beachten Sie, dass Sie das Disk Image nicht einfach auf die SD-Karte ziehen können – Sie müssen eine bitweise Kopie des Image vornehmen. Dazu brauchen Sie einen Kartenleser (ein ganz billiger reicht) und ein Disk-Image-Tool. Die notwendigen Schritte sind je nach Betriebssystem unterschiedlich. Entpacken Sie die Image-Datei (Sie sollten dann eine *.img*-Datei besitzen) und folgen Sie den Anweisungen aus Anhang A.

Schnellere Downloads mit BitTorrent

Sie werden auf der Download-Site einen Hinweis sehen, möglichst eine Torrent-Datei zu nutzen, um Raspbian so effizient wie möglich herunterzuladen. Die Torrent-Datei ist ein dezentralisierter Weg, Dateien verteilt vorzuhalten. Das Herunterladen kann dadurch deutlich schneller geschehen, weil Sie immer nur Teile der Datei von vielen verschiedenen anderen Torrent-Clients holen statt von einem zentralen Server. Wollen Sie diesen Weg gehen, benötigen Sie einen BitTorrent-Client.

Ein paar beliebte BitTorrent-Clients sind:

- Vuze (<http://www.vuze.com/>): Integrierte Torrent-Suche und Download.
- Miro (<http://www.getmiro.com/>): Open-Source-Musik- und Video-Player, der auch Torrents verarbeitet.
- MLDonkey (http://mldonkey.sourceforge.net/Main_Page): Filesharing-Tool nur für Windows und Linux.
- Transmission (<http://www.transmissionbt.com/>): Schlanker Client für Mac und Linux, wird auch in Embedded-Systemen genutzt.

Booten

Mit diesen Schritten booten Sie Ihren Raspberry Pi das erste Mal:

1. Stecken Sie die SD-Karte in den Kartenschacht.
2. Stecken Sie eine USB-Tastatur und eine Maus ein. Beim Modell A müssen Sie beides in einen Hub mit eigener Stromversorgung stecken, den Sie dann mit dem Pi verbinden.
3. Verbinden Sie den Pi über ein HDMI-Kabel mit Ihrem Fernseher oder Monitor. Achten Sie darauf, dass der Monitor eingeschaltet ist.
4. Stecken Sie die Stromversorgung ein. Ganz allgemein sollten Sie darauf achten, dass alles angeschlossen ist, bevor Sie den Pi als Letztes mit Strom versorgen.

Online gehen

Sie haben verschiedene Möglichkeiten, sich mit dem Internet zu verbinden. Haben Sie einfachen Zugriff auf einen Router oder Switch (oder eine Ethernet-Dose, die mit einem Router verbunden ist), nutzen Sie einfach ein Standard-Ethernet-Kabel. Haben Sie einen USB-WLAN-Stick, können Sie sich drahtlos verbinden; auf dem Desktop gibt es ein Icon, über das Sie Ihre WLAN-Verbindung einrichten können. Es funktionieren nicht alle WLAN-Sticks, werfen Sie am besten einen Blick auf die Liste mit den geprüften Peripherie-Geräten (http://elinux.org/RPi_VerifiedPeripherals).

Haben Sie einen Laptop in der Nähe oder lassen Sie den Pi ohne Monitor laufen, können Sie den WLAN-Anschluss Ihres Laptops auch durch den Pi nutzen (Abbildung 1-7). Auf dem Mac ist das ganz einfach: Aktivieren Sie in Ihren Sharing-Einstellungen einfach Internet Sharing und verbinden Sie dann den Pi über ein Ethernet-Kabel mit dem Mac. Unter Windows aktivieren Sie »Anderen Benutzern im Netzwerk gestatten, diese Verbindung des Computers als Internetverbindung zu verwenden« in Ihren Einstellungen zur „Gemeinsamen Nutzung der Internetverbindung“. Der Pi sollte automatisch eine IP-Adresse erhalten, wenn er sich verbindet und online geht.

Für einen Windows-PC benötigen Sie vermutlich ein Cross-Over-Kabel, während bei Apple-Hardware ein beliebiges Ethernet-Kabel reicht, weil der Kabeltyp dort automatisch erkannt wird.

Wenn alles klappt, sollten Sie einen Haufen Startup-Logeinträge auf Ihrem Bildschirm sehen. Wenn nicht, blättern Sie gleich weiter zum Abschnitt „Fehlerbehebung“ am Ende dieses Kapitels. Die Logeinträge zeigen alle Prozesse, die beim Booten des Pi gestartet werden. Die Netzwerk-Schnittstelle wird initialisiert und die angeschlossenen USB-Geräte werden erkannt und protokolliert. Sie können sich diese Logeinträge anzeigen lassen, indem Sie nach dem Anmelden an der Befehlszeile `dmesg` tippen.

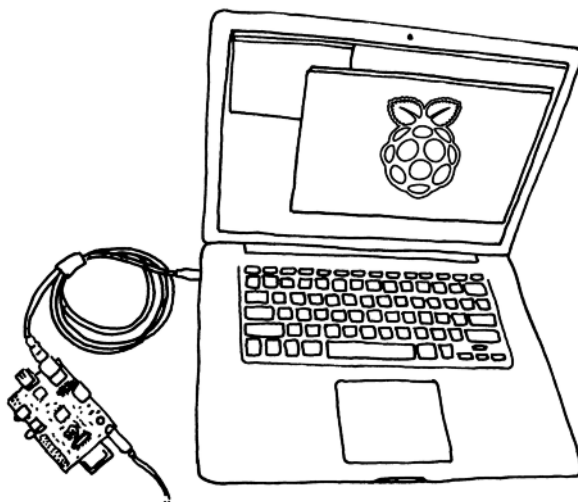


Abbildung 1-7. Es ist sehr praktisch, die WLAN-Verbindung Ihres Laptops mit dem Pi zu teilen. Sie können den Pi auch ohne eigenen Monitor laufen lassen (siehe „Headless laufen“ (S. 32)), was hilft, wenn Sie Ihren Raspberry Pi irgendwo im Einsatz haben.

Beim allerersten Booten wird sich das raspi-config Tool bei Ihnen melden (siehe Abbildung 1-8). Es gibt ein paar wichtige Einstellungen, die Sie hier vornehmen müssen, denn es ist recht wahrscheinlich, dass Ihr Raspberry Pi nicht automatisch so arbeitet, wie Sie es sich vorstellen. Müssen Sie dieses Konfigurations-Tool später nochmals aufrufen, erreichen Sie es über die Eingabe folgender Anweisung an der Befehlszeile:

```
sudo raspi-config
```

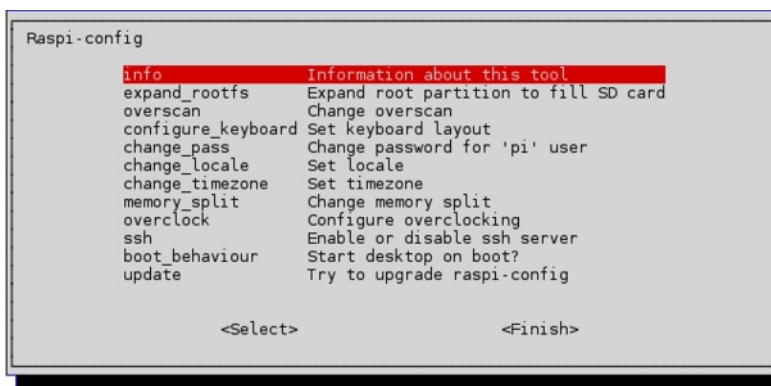


Abbildung 1-8. Das Menü des raspi-config Tools

Konfigurieren Sie Ihren Pi

Als Nächstes wollen wir Ihnen zeigen, welche Konfigurationsmöglichkeiten entscheidend sind und welche Sie sich auch später anschauen können. Beim Setzen der Optionen nutzen Sie im Tool die Cursortasten nach oben und unten, die Leertaste zum Auswählen und die Tabulatortaste zum Ändern von Feldern beziehungsweise zum Bewegen des Cursors zu den Schaltern unten im Fenster. Lassen Sie uns die Menüoptionen der Reihe nach vorstellen:

Expand rootfs

Sie sollten diese Option immer auswählen – damit wird das Dateisystem so erweitert, dass die gesamte SD-Karte genutzt wird.

Overscan

Lassen Sie die Overscan-Option zunächst deaktiviert. Haben Sie einen HD-Monitor, kann es sein, dass Text über den Rand des Bildschirms hinausläuft. Um das zu beheben, aktivieren Sie den Overscan und ändern die Werte so, dass das Bild vollständig auf den Monitor passt. Mit den Werten korrigiert die Anzeigesoftware die Darstellung – mit positiven Werten wird das Bild verkleinert, mit negativen vergrößert (falls schwarze Balken am Rand zu sehen sind).

Keyboard

Als Tastatur wird standardmäßig von einer generischen, englischen Version ausgegangen. Sollen die Tasten das tun, was auf ihnen steht, werden Sie sehr wahrscheinlich einen anderen Tastatortyp und ein Mapping auswählen wollen. Zum Glück ist die Tastaturliste sehr robust. Beachten Sie, dass Ihre Landeseinstellungen ebenfalls das Tastaturverhalten beeinflussen können.

Password

Es ist ratsam, das Standardpasswort von *raspberrypi* auf einen sichereren Wert zu ändern.

Change Locale

Außerhalb von Großbritannien sollten Sie Ihr Locale so ändern, dass Ihre Sprache und Ihre Zeichenkodierung angepasst wird. Standardeinstellung ist Englisch (UK) mit einer Standard-UTF-8-Zeichenkodierung (`en_GB.UTF-8`).

Change timezone

Sie werden vermutlich die richtige Zeitzone einstellen wollen.

Memory split

Mit dieser Option legen Sie fest, wie viel Speicher von CPU und GPU genutzt wird. Belassen Sie es zunächst bei den Standardwerten.

Overclock

Mit dieser Option können Sie den Prozessor schneller als 700 MHz laufen lassen. Beim ersten Booten sollten Sie die Standardeinstellung beibehalten oder *Medium* oder *Modest* ausprobieren. Sie können auch später noch an dieser Einstellung schrauben (im Turbomodus kann der Prozessor mit 1000 MHz laufen).

SSH

Diese Option aktiviert den Secure-Shell (ssh)-Server, mit dem Sie sich am Raspberry Pi aus der Ferne über das Netzwerk anmelden können. Das ist ausgesprochen praktisch, daher sollten Sie diese Option aktivieren.

Desktop Behavior

Mit dieser Option wird dafür gesorgt, dass nach dem Booten direkt die grafische Benutzeroberfläche aufgerufen wird. Standardmäßig ist sie schon auf *Yes* gesetzt. Wählen Sie *No*, erhalten Sie nach dem Booten die Befehlszeile und Sie müssen sich hier anmelden und die GUI per Hand starten:

```
raspberrypi login: pi
Password: raspberry
pi@raspberrypi ~ $ startx
```

Befinden Sie sich auf der grafischen Oberfläche, wird die Befehlszeile verschwinden. Sie können aber ein Terminal-Programm öffnen, um wieder eine Befehlszeile zu erhalten. Klicken Sie das Desktop-Menü links unten an und wählen Sie Accessories→LXTerminal.

Update

Sind Sie mit dem Internet verbunden, können Sie das config-Tool schließlich über diese Option aktualisieren. Führen Sie beim ersten Mal kein Update für das Betriebssystem durch – Sie werden in Kapitel 2 noch andere Wege dafür kennenlernen.

Sind Sie fertig, wählen Sie *Finish*. Dann landen Sie wieder auf der Befehlszeile. Geben Sie nun ein:

```
pi@raspberrypi ~ $ sudo reboot
```

Ihr Pi wird dann mit den neuen Einstellungen neu gestartet. Wenn alles klappt (und wenn Sie die Option gewählt haben, direkt die grafische Benutzeroberfläche anzuzeigen), sollten Sie den Openbox Window Manager sehen, der auf der Lightweight X11 Desktop Environment (LXDE) läuft. Dann sind Sie im Spiel!

Herunterfahren

Der Raspberry Pi hat keinen Ausschalter (auch wenn es auf neueren Boards Anschlüsse für einen Reset-Schalter gibt). Um den Rechner herunterzufahren, muss man das Logout-Menü auf der grafischen Benutzeroberfläche wählen und dann dort auf Shutdown klicken.

Sie können auch die Befehlszeile verwenden, indem Sie folgende Anweisung eingeben:

```
pi@raspberrypi ~ $ sudo shutdown -h now
```

Achten Sie darauf, den Pi sauber herunterzufahren (und nicht einfach nur den Stecker zu ziehen). Ansonsten kann es in seltenen Fällen passieren, dass die Daten auf der SD-Karte zerstört werden.

Fehlersuche

Wenn die Dinge nicht so laufen, wie sie sollten, gibt es ein paar häufig vorkommende Fehler, die man eventuell gemacht hat, oder Schritte, die vergessen wurden. Achten Sie darauf, alle folgenden Punkte zu prüfen:

- Befindet sich die SD-Karte im Slot und ist sie auch nicht verkantet? Verwenden Sie den richtigen Typ SD-Karte?
- Wurde das Disk Image korrekt auf die Karte geschrieben? Versuchen Sie, das Schreiben mit einem anderen Kartenleser zu wiederholen.
- Ist der Schreibschutz der SD-Karte aktiv? Dabei handelt es sich um einen kleinen Schiebeschalter, der sich leicht einmal in der falschen Position befinden kann.
- Prüfen Sie die Integrität Ihres verwendeten Disk Images. Das können Sie durch das Ausführen eines Checksum-Tools mit Secure Hash Algorithm (SHA) für das Image erreichen. Vergleichen Sie das Ergebnis mit den 40 Zeichen, die auf der Download-Seite veröffentlicht wurden.
- Startet der Pi von alleine neu oder hat er andere Probleme, während er läuft? Prüfen Sie die Stromversorgung – ein mit zu wenig Strom versorgtes Board kann laufen, sich aber unzuverlässig verhalten.
- Erhalten Sie beim Start eine Kernel Panic? Das ist das Äquivalent zum Blue Screen of Death von Windows – meist verursacht durch ein Problem mit einem Gerät am USB-Hub. Stöpseln Sie USB-Geräte aus und versuchen Sie dann, den Pi neu zu starten.

Wenn all das nicht hilft, schauen Sie auf die Troubleshooting-Seite des Raspberry Hub Wiki (http://elinux.org/R-Pi_Troubleshooting). Dort finden sich Lösungen für alle möglichen Probleme, die schon einmal aufgetreten sind.



Welches Board besitzen Sie?

Wenn Sie per E-Mail oder in einem Forum um Hilfe bitten, ist es für den Helfer sinnvoll, die genaue Betriebssystem- und Board-Version zu kennen. Um die Betriebssystem-Version herauszufinden, öffnen Sie das LXTerminal und geben Folgendes ein:

```
cat /proc/version
```

Für die Board-Version geben Sie ein:

```
cat /proc/cpuinfo
```

Weitere Informationen

The Raspberry Pi Hub (http://elinux.org/RPi_Hub)

Ein umfangreiches Wiki mit Informationen zur Hardware und Konfiguration des Pi, von elinux.org gehostet.

Liste mit geprüften Peripheriegeräten

(http://elinux.org/RPi_VerifiedPeripherals)

Die umfassende Liste mit Peripheriegeräten, die bekanntermaßen mit dem Raspberry Pi funktionieren.

2/Linux auf dem Raspberry Pi

Wollen Sie Ihren Raspberry Pi richtig nutzen, müssen Sie ein bisschen Linux lernen. Ziel dieses Kapitels ist es, Ihnen einen Überblick über das Betriebssystem zu geben und Ihnen genug Wissen zu vermitteln, damit Sie mit dem Dateisystem zurechtkommen, Pakete an der Befehlszeile oder im GUI installieren können und die wichtigsten Tools kennen, die Sie tagtäglich benötigen.

Raspbian bringt die grafische Benutzeroberfläche Lightweight X11 Desktop Environment (LXDE) mit. Dabei handelt es sich um eine abgespeckte Desktop-Umgebung für das X Window System, das schon seit den 80er Jahren Grundlage für die GUI von Unix- und Linux-Rechnern ist. Manche der Tools auf dem Desktop und im Menü werden direkt mit LXDE zusammen ausgeliefert (zum Beispiel der Texteditor Leafpad und die Shell LXTerminal).

Auf LXDE aufbauend läuft Openbox, ein *Window Manager*, der sich um das Aussehen und Verhalten der Fenster und Menüs kümmert. Wollen Sie das Erscheinungsbild Ihres Desktops anpassen, nutzen Sie dafür die Openbox-Konfigurationstools (klicken Sie links unten auf den Desktop und wählen Sie dann Other→Openbox Configuration Manager). Anders als bei OS X oder Windows ist es recht einfach, Ihre Desktop-Umgebung umfassend zu konfigurieren oder alternative Window Manager zu installieren. Einige der anderen Distributionen für den Raspberry Pi besitzen andere Umgebungen, die für spezielle Anwendungen optimiert sind, wie zum Beispiel Settop-Boxen, Telefonsysteme oder Netzwerk-Firewalls. Unter http://elinux.org/RPi_Distributions finden Sie eine Übersicht.

Der Dateimanager

Wollen Sie Dateien nicht an der Befehlszeile verschieben und kopieren (dazu gleich mehr), wählen Sie den File Manager aus dem Accessories-Menü. Sie können dann so mit dem Dateisystem arbeiten, wie Sie es vermutlich auf einer grafischen Benutzeroberfläche gewohnt sind.

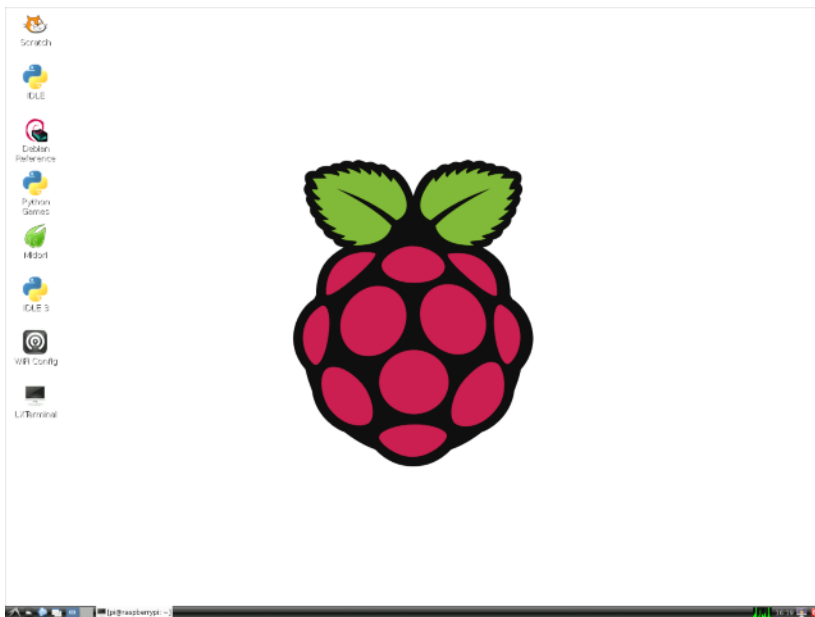


Abbildung 2-1. *Der grafische Desktop*

Der Webbrowser

Der Standard-Webbrowser ist Midori (<http://twotoasts.de/index.php/midori/>), der dafür entworfen wurde, mit begrenzten Ressourcen arbeiten zu können. Man vergisst heutzutage leicht, wie viel Arbeit Webbrowser erledigen. Da Raspbian eine sehr leichtgewichtige Betriebssystem-Distribution sein soll, gibt es eine Reihe nicht vorhandener Features, die Sie vielleicht von einem Webbrowser erwarten. So sind zum Beispiel Flash und das Java-Plugin nicht installiert (und somit funktioniert YouTube nicht) und Midori unterstützt keine HTML5-Videos. Wir werden später sehen, wie Sie neue Software installieren können (zum Beispiel Java). Schauen Sie sich einmal die Tools und Menü-Elemente im Pulldown-Menü oben rechts im Fenster an (siehe Abbildung 2-2). Es gibt eine Reihe von Browser-Alternativen — besonders erwähnt seien hier NetSurf (<http://www.netsurf-browser.org/>) und Dillo (<http://www.dillo.org/>).

Video und Audio

Für die Multimedia-Wiedergabe gibt es den omxplayer, der aktuell noch ein wenig experimentell wirkt. Er steht nur als Befehlszeilen-Tool zur Verfügung. Der omxplayer ist für eine direkte Zusammenarbeit mit der Graphics Processing Unit (GPU) des Prozessors ausgelegt — andere freie Software wie VLC und mPlayer spielen nicht so gut mit der GPU zusammen.

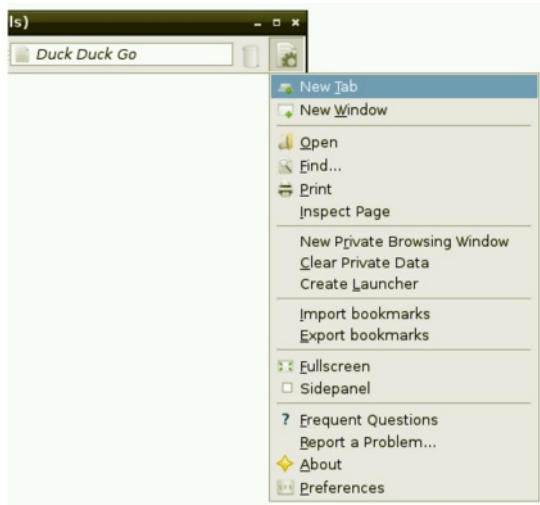


Abbildung 2-2. Das Pulldown-Menü im Webbrowser.



Um den Preis niedrig zu halten, sind bestimmte Video-Lizenzen nicht beim Raspberry Pi enthalten. Wollen Sie aufgezeichnete Fernsehsendungen oder DVDs im MPEG-2-Format (oder Microsofts VC-1-Format) sehen, müssen Sie einen Lizenzschlüssel im Onlineshop der Foundation (<http://raspberrypi.com/>) erwerben. Eine Lizenz für das Kodieren und Dekodieren von H.264 (MPEG-4) steht dagegen frei zur Verfügung.

Texteditor

Leafpad (<http://wiki.lxde.org/en/Leafpad>) ist der Standard-Texteditor, den Sie auch im Hauptmenü finden. Es gibt dazu auch Nano (<http://www.nano-editor.org/>), bei dem es sich um einen leicht zu erlernenden, allerdings auch nur sehr rudimentären Texteditor handelt. Klassische Unix-Texteditoren wie vim oder emacs sind nicht standardmäßig installiert, können aber leicht hinzugefügt werden (siehe Abschnitt 2.0).

Copy and Paste

Copy and Paste funktioniert zwischen den Anwendungen recht gut, auch wenn Sie bald merken werden, dass manche Programme leider nicht mitspielen. Hat Ihre Maus eine dritte, mittlere Taste, können Sie Text ganz normal auswählen, indem Sie ihn selektieren (mit der linken Maustaste anklicken und die Auswahl größer ziehen) und dann durch einen Druck mit der mittleren Maustaste einfügen, sofern sich der Mauszeiger über dem Zielfenster befindet.

Die Shell

Für eine ganze Reihe von Aufgaben müssen Sie zur Befehlszeile wechseln und dort Anweisungen ausführen. Das Programm LXTerminal ermöglicht diesen Zugriff zur Befehlszeile oder Shell. Standard-Shell bei Raspbian ist die Bourne Again Shell (bash (<http://www.gnu.org/software/bash/manual/bashref.html>)), die auf Linux-Systemen sehr verbreitet ist. Ein Alternative ist dash (http://en.wikipedia.org/wiki/Debian_Almquist_shell). Die Shells können Sie über das Programm-Menü oder die Anweisung chsh ändern.

Arbeiten an der Befehlszeile

Wenn es Ihnen hilft, können Sie sich einen Einsatz an der Befehlszeile wie das Spielen eines Textadventures vorstellen, nur dass Sie mit Dateien und dem Dateisystem arbeiten statt mit Monstern und Labyrinthen. Wenn Ihnen diese Metapher nicht hilft, sorgen Sie sich nicht – alle Befehle und Konzepte in diesem Abschnitt gehören zum Linux-Standard und lohnen ein Erlernen auf jeden Fall.

Bevor Sie beginnen, öffnen Sie das Programm LXTerminal (Abbildung 2-3). Es gibt zwei wichtige Tipps, um das Leben in der Shell einfacher zu machen: *Autovervollständigen* und *Befehlspuffer*. Häufig brauchen Sie nur die ersten paar Zeichen eines Befehls oder eines Dateinamens einzugeben und können dann die Tabulator-Taste drücken. Die Shell wird versuchen, den String basierend auf den Dateien im aktuellen Verzeichnis oder den Programmen in Verzeichnissen wie `/bin` oder `/usr/bin/` zu ermitteln. Drücken Sie an der Befehlszeile die Cursortaste nach oben, können Sie den Befehlspuffer durch-

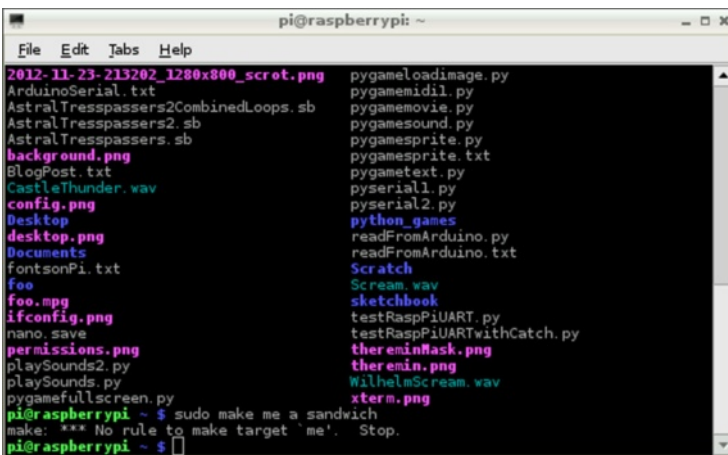


Abbildung 2-3. LXTerminal ermöglicht Ihnen den Zugriff auf die Befehlszeile (oder Shell)

laufen und sich vorher angegebene Befehle anzeigen lassen. Das ist nützlich, wenn Sie ein Zeichen in einer langen Befehlskette falsch eingegeben haben.

Dateien und das Dateisystem

Tabelle 2-1 auf der folgenden Seite enthält die wichtigsten Verzeichnisse im Dateisystem. Die meisten folgen dem Linux-Standard für das Ablegen von Dateien, ein paar sind spezifisch für den Raspberry Pi. Im Verzeichnis `/sys` können Sie auf die Hardware des Raspberry Pi zugreifen.

Das aktuelle Verzeichnis wird jeweils vor dem Command Prompt angezeigt. Unter Linux besitzt Ihr Home-Verzeichnis eine Kurzform: die Tilde (`~`). Wenn Sie das LXTerminal öffnen, landen Sie in Ihrem Home-Verzeichnis und Ihr Prompt sieht wie folgt aus:

```
pi@raspberrypi ~ $
```

Dies ist eine Erläuterung für dieses Prompt:

```
pi@❶raspberrypi❷ ~❸ $❹
```

- ❶ Ihr Benutzername (pi), gefolgt vom At-Symbol (@).
- ❷ Der Name Ihres Computers (raspberrypi ist der Standard-Hostname).
- ❸ Das *aktuelle Arbeitsverzeichnis* der Shell. Sie beginnen immer in Ihrem Home-Verzeichnis (`~`).
- ❹ Dies ist das *Shell Prompt*. Text, den Sie eintippen, erscheint rechts davon. Drücken Sie die Eingabetaste, um eine eingegebene Anweisung auszuführen.

Mit dem Befehl `cd` (Change Directory) können Sie sich im Dateisystem bewegen. Die folgenden zwei Befehle haben den gleichen Effekt (den Wechsel zum Home-Verzeichnis) für den Anwender pi:

```
cd /home/pi/  
cd ~
```

Beginnt der Verzeichnispfad, mit einem Schrägstrich wird er als absoluter Pfad auf das Verzeichnis interpretiert. Ansonsten wird die Verzeichnisangabe als relativ zum aktuellen Arbeitsverzeichnis betrachtet. Sie können auch `.` und `..` nutzen, um sich auf das aktuelle beziehungsweise das übergeordnete Verzeichnis zu beziehen. Um zum Beispiel an den „Anfang“ des Dateisystems zu springen:

```
pi@raspberrypi ~ $ cd ..  
pi@raspberrypi /home $ cd ..
```

Tabelle 2-1. Einige der wichtigsten Verzeichnisse im Raspbian-Dateisystem

Verzeichnis	Beschreibung
/	
/bin	Programme und Befehle, die alle Anwender nutzen können
/boot	Alle Dateien, die beim Booten benötigt werden
/dev	Spezielle Dateien, die die Geräte Ihres Systems repräsentieren
/etc	Konfigurationsdateien
/etc/init.d	Skripten, um Dienste zu starten
/etc/X11	X11-Konfigurationsdateien
/home	Home-Verzeichnisse der Anwender
/home/pi	Home-Verzeichnis des Anwenders pi
/lib	Kernel-Module/Treiber
/media	Mountpunkte für entfernbare Medien
/proc	Ein virtuelles Verzeichnis mit Informationen über laufende Prozesse und das Betriebssystem
/sbin	Programme zur Systemwartung
/sys	Ein spezielles Verzeichnis auf dem Raspberry Pi, das die Hardwaregeräte repräsentiert
/tmp	Ein Verzeichnis für Programme, um temporäre Daten zu erzeugen
/usr	Programme und Daten, die von allen Anwendern genutzt werden können
/usr/bin	Die meisten Programme des Betriebssystems finden sich hier
/usr/games	Ja, Spiele
/usr/lib	Bibliotheken für die gebräuchlichen Programme
/usr/local	Software, die für diesen Rechner spezifisch ist, kann hierhin kopiert werden
/usr/sbin	Weitere Programme zur Systemverwaltung
/usr/share	Dinge, die von Anwendungen gemeinsam genutzt werden – zum Beispiel Icons oder Schriften
/usr/src	Linux ist Open Source – hier finden Sie den Quellcode!
/var	System-Logs und Spooldateien
/var/backups	Backup-Kopien aller lebenswichtigen Systemdateien
/var/cache	Programme, die Daten cachen (wie zum Beispiel apt-get oder ein Webbrowser), legen die Daten hier ab
/var/log	Alle Systemlogs und individuellen Service-Logs
/var/mail	Die Benutzer-E-Mails werden hier abgelegt, wenn das System für E-Mails eingerichtet ist
/var/spool	Daten, die auf eine Verarbeitung warten (zum Beispiel eintreffende E-Mails oder Druckaufträge)

Sie erreichen dies auch über den absoluten Pfad /:

```
pi@raspberrypi ~ $ cd /
```

Sind Sie zu einem Verzeichnis gewechselt, verwenden Sie den Befehl `ls`, um die dort vorhandenen Dateien aufzuführen:

```
pi@raspberrypi / $ ls
bin  dev  home  lost+found  mnt  proc  run  selinux  sys  usr
boot  etc  lib  media      opt  root  sbin  srv      tmp  var
```

Für die meisten Befehle gibt es zusätzliche Parameter oder *Schalter*, mit denen man das Verhalten des Befehls beeinflussen kann. So führt zum Beispiel der Schalter `-l` zu einer ausführlicheren Ausgabe mit Dateigröße, -datum und -berechtigungen:

```
pi@raspberrypi ~ $ ls -l
total 8
drwxr-xr-x 2 pi pi 4096 Oct 12 14:26 Desktop
drwxrwxr-x 2 pi pi 4096 Jul 20 14:07 python_games
```

Der Schalter `-la` gibt alle Dateien aus, auch die unsichtbaren:

```
pi@raspberrypi ~ $ ls -la
total 80
drwxr-xr-x 11 pi pi 4096 Oct 12 14:26 .
drwxr-xr-x 3 root root 4096 Sep 18 07:48 ..
-rw----- 1 pi pi 25 Sep 18 09:22 .bash_history
-rw-r--r-- 1 pi pi 220 Sep 18 07:48 .bash_logout
-rw-r--r-- 1 pi pi 3243 Sep 18 07:48 .bashrc
drwxr-xr-x 6 pi pi 4096 Sep 19 01:19 .cache
drwxr-xr-x 9 pi pi 4096 Oct 12 12:57 .config
drwx----- 3 pi pi 4096 Sep 18 09:24 .dbus
drwxr-xr-x 2 pi pi 4096 Oct 12 14:26 Desktop
-rw-r--r-- 1 pi pi 36 Sep 18 09:35 .dmrc
drwx----- 2 pi pi 4096 Sep 18 09:24 .gvfs
drwxr-xr-x 2 pi pi 4096 Oct 12 12:53 .idlerc
-rw----- 1 pi pi 35 Sep 18 12:11 .lessshst
drwx----- 3 pi pi 4096 Sep 19 01:19 .local
-rw-r--r-- 1 pi pi 675 Sep 18 07:48 .profile
drwxrwxr-x 2 pi pi 4096 Jul 20 14:07 python_games
drwx----- 4 pi pi 4096 Oct 12 12:57 .thumbnails
-rw----- 1 pi pi 56 Sep 18 09:35 .xauthority
-rw----- 1 pi pi 300 Oct 12 12:57 .xsession-errors
-rw----- 1 pi pi 1391 Sep 18 09:35 .xsession-errors.old
```

Mit dem Befehl `mv` benennen Sie eine Datei um. Der Befehl `touch` kann genutzt werden, um eine leere Dummy-Datei zu erzeugen:

```

pi@raspberrypi ~ $ touch foo
pi@raspberrypi ~ $ ls
foo  Desktop  python_games
pi@raspberrypi ~ $ mv foo baz
pi@raspberrypi ~ $ ls
baz  Desktop  python_games

```

Entfernen Sie eine Datei mit `rm`. Um ein Verzeichnis zu entfernen, können Sie `rmdir` nutzen, wenn es leer ist, oder `rm -r`, wenn sich noch Dateien (oder andere Verzeichnisse) darin befinden. `-r` ist ein Parameter für den Befehl `rm`, der dafür sorgt, dass rekursiv alles im Verzeichnis gelöscht wird.

Wollen Sie herausfinden, welche Parameter es für einen bestimmten Befehl gibt, verwenden Sie `man` (oder Sie greifen auf die oft vorhandene Option `--help` zurück):

```

pi@raspberrypi ~ $ man curl
pi@raspberrypi ~ $ rm --help

```

Um ein neues Verzeichnis zu erstellen, rufen Sie `mkdir` auf. Um alle Dateien eines Verzeichnisses in einer einzelnen Datei zusammenzufassen, verwenden Sie den Befehl `tar`, der ursprünglich für Band-Archive gedacht war. Viele Dateien oder Quelltexte werden als `tar`-Dateien angeboten, meist noch mit dem Befehl `gzip` komprimiert. Versuchen Sie einmal Folgendes:

```

pi@raspberrypi ~ $ mkdir myDir
pi@raspberrypi ~ $ cd myDir
pi@raspberrypi ~ $ touch foo bar baz
pi@raspberrypi ~ $ cd ..
pi@raspberrypi ~ $ tar -cf myDir.tar myDir
pi@raspberrypi ~ $ gzip myDir.tar

```

Jetzt haben Sie ein `.tar.gz`-Archiv dieses Verzeichnisses, das per E-Mail oder über das Internet weitergegeben werden kann.

Weitere Linux-Befehle

Einer der Gründe, dass Linux (und Unix) so erfolgreich ist, ist das wichtigste Entwurfsziel: ein sehr kompliziertes System aus kleinen, einfachen, modularen Teilen aufbauen, die miteinander verknüpft werden können. Sie müssen dabei ein bisschen über zwei Elemente dieses Puzzles wissen: *Pipes* und *Redirection*.

Pipes sind einfach eine Möglichkeit, zwei Programme so zu verketten, dass die Ausgabe des einen Programms als Eingabe des anderen Programms dient. Alle Linux-Programme können Daten von *Standard Input* lesen (häufig als *stdin* bezeichnet), Daten nach *Standard Output* schreiben (*stdout*) und Fehlermeldungen an *Standard Error* ausgeben (*stderr*). Mit einer Pipe

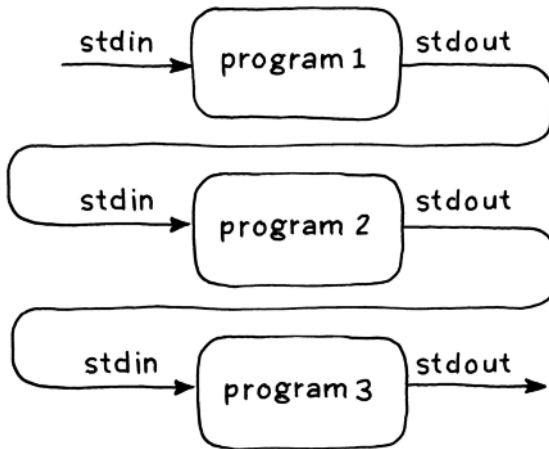


Abbildung 2-4. Pipes sind eine Möglichkeit, kleinere Programme zu kombinieren, damit sie größere Aufgaben erledigen können

können Sie stdout des einen Programms mit stdin eines anderen Programms kombinieren (Abbildung 2-4). Nutzen Sie dazu den Operator `|` wie im folgenden Beispiel:

```
pi@raspberrypi ~ $ ls -la | less
```

(Drücken Sie `q`, um das Programm `less` wieder zu verlassen.)

Versuchen Sie jetzt einmal Folgendes:

```
pi@raspberrypi ~ $ sudo cat /boot/kernel.img | aplay
```

Sie werden die Lautstärke vermutlich etwas herunterdrehen wollen, denn diese Anweisungen lesen das Kernel Image ein und geben all die Einsen und Nullen an den Audio Player aus. So klingt Ihr Kernel!

In einigen der in diesem Buch noch folgenden Beispiele werden wir auch eine *Redirection* nutzen, bei der ein Befehl ausgeführt und die stdout-Ausgabe in eine Datei umgeleitet wird. Wie Sie später sehen werden, werden viele Dinge in Linux als normale Dateien behandelt (wie zum Beispiel die General Purpose Input und Output Pins des Pi), daher kann die Redirection sehr praktisch sein. Um die Ausgabe eines Programms umzuleiten, verwenden Sie den Operator `>`:

```
pi@raspberrypi ~ $ ls > directoryListing.txt
```

Besondere Steuertasten

Neben den Tasten zum Autovervollständigen (Tabulator-Taste) und Befehlspuffer (Pfeil nach oben), die schon weiter oben erwähnt wurden, gibt es noch ein paar weitere Steuertasten, die Sie in der Shell brauchen werden. Dazu gehören unter anderem:

Strg-C

Beendet das aktuell laufende Programm. Funktioniert eventuell nicht mit jedem interaktiven Programm wie zum Beispiel Texteditoren.

Strg-D

Beendet die Shell. Sie müssen dies an der Befehlszeile selbst eingeben (und es darf nichts hinter dem \$ stehen, bevor Sie Strg-D drücken).

Strg-A

Bewegt den Cursor an den Zeilenanfang.

Strg-E

Bewegt den Cursor an das Zeilenende.

Es gibt noch weitere Tastenkürzel, aber diese hier sind die wichtigsten, die Sie jeden Tag einsetzen werden.

Manchmal wollen Sie den Inhalt einer Datei auf dem Bildschirm ausgeben. Handelt es sich um eine Textdatei und wollen Sie sie Seite für Seite lesen, verwenden Sie `less`:

```
pi@raspberrypi ~ $ ls > flob.txt
pi@raspberrypi ~ $ less flob.txt
```

Wollen Sie nur den gesamten Inhalt einer Datei an Standard Output schicken, verwenden Sie `cat` (Kurzform von *concatenate*). Das kann sehr nützlich sein, wenn Sie eine Datei an ein anderes Programm verfüttern oder sie woandershin umleiten wollen.

Dies entspricht zum Beispiel dem Kopieren einer Datei in eine andere mit einem neuen Namen (die dritte Zeile verkettet die beiden ersten Dateien):

```
pi@raspberrypi ~ $ ls > wibble.txt
pi@raspberrypi ~ $ cat wibble.txt > wobble.txt
pi@raspberrypi ~ $ cat wibble.txt wobble.txt > wobble.txt
```

Um sich nur die letzten paar Zeilen einer Datei anzeigen zu lassen (wie zum Beispiel die neuesten Einträge einer Logdatei), nutzen Sie `tail` (für den Anfang setzten Sie `head` ein). Suchen Sie in einer oder mehreren Dateien nach einem String, verwenden Sie das altherwürdige `grep`:

```
pi@raspberrypi ~ $ grep Puzzle */*
```

Grep ist ein leistungsfähiges Tool, da es mit den umfangreichen *regulären Ausdrücken* arbeitet. Reguläre Ausdrücke sind manchmal schwierig zu lesen und leisten sicherlich ihren Beitrag dazu, dass Linux für Neueinsteiger kompliziert wirkt.

Prozesse

Jedes Programm auf dem Pi läuft in einem eigenen Prozess, daher laufen eigentlich jederzeit dutzende Prozesse. Beim ersten Booten starten etwa 75 Prozesse, von denen jeder eine andere Aufgabe hat oder einen bestimmten Dienst betreut. Um all diese Prozesse aufgelistet zu bekommen, nutzen Sie das Programm `top`, mit dem auch die CPU-Last und der Speicherverbrauch angezeigt werden. Mit `top` sehen Sie, welche Prozesse die meisten Ressourcen belegen, mit dem Befehl `ps` erhalten Sie eine Liste aller Prozesse und ihrer ID-Nummern, zum Beispiel über:

```
pi@raspberrypi ~ $ ps -aux | less
```

Manchmal wollen Sie einen aus dem Ruder laufenden oder nicht mehr reagierenden Prozess stoppen. Dazu nutzen Sie zuerst `ps`, um seine ID zu finden, und dann `kill`, um ihn zu beenden.

```
pi@raspberrypi ~ $ kill 95689
```

Bei manchen Systemprozessen haben Sie nicht die Berechtigung, ihn zu beenden (allerdings werden Sie gleich `sudo` kennenlernen).

Sudo und Berechtigungen

Linux ist ein Mehrbenutzer-Betriebssystem. Allgemeine Regel ist, dass jeder seine eigenen Dateien besitzt und sie in seinem Bereich des Dateisystems erstellen, bearbeiten und löschen kann. Der Root- (oder Super-)User kann alle Dateien im Dateisystem ändern, was der Grund dafür ist, sich möglichst nicht immer als root anzumelden, sondern nur bei Bedarf.



Als Benutzer `pi` können Sie im System nicht viel Schaden anrichten. Als Superuser können Sie allerdings durchaus Chaos verursachen – unabsichtlich oder auch geplant. Seien Sie vorsichtig, wenn Sie `sudo` nutzen, insbesondere, wenn Sie Dateien verschieben oder löschen. Wenn Ihre Aktionen richtig in die Hose gegangen sind, können Sie allerdings immer noch ein neues SD-Karten-Image erzeugen (siehe Anhang A).

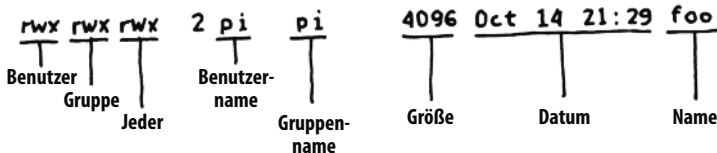


Abbildung 2-5. Dateiberechtigungen für den Besitzer, die Gruppe und jedermann

Es gibt ein paar Tools wie `sudo`, die es dem Anwender ermöglichen, als Superuser tätig zu sein. Dann kann er Software installieren, ohne sich den Gefahren (und der Verantwortung) auszusetzen, als `root` angemeldet zu sein. Sie werden `sudo` recht viel nutzen, wenn Sie direkt mit der Hardware interagieren oder wenn Sie an der systemweiten Konfiguration schrauben.

Jede Datei gehört einem bestimmten Anwender und einer bestimmten Gruppe. Mit den Befehlen `chown` und `chgrp` ändern Sie den Besitzer oder die Gruppe einer Datei. Beachten Sie, dass Sie `root`-User sein müssen, um einen dieser Befehle einsetzen zu können:

```
pi@raspberrypi ~ $ sudo chown pi garply.txt
pi@raspberrypi ~ $ sudo chgrp staff plugh.txt
```

Jede Datei besitzt zudem einen Satz *Berechtigungen*, die zeigen, ob eine Datei gelesen, geschrieben oder ausgeführt werden kann. Diese Berechtigungen können für den Besitzer der Datei, die Gruppe oder jedermann gesetzt werden (siehe Abbildung 2-5).

Sie setzen die einzelnen Berechtigungen mit dem Befehl `chmod`. Die Schalter für `chmod` sind in Tabelle 2-2 zusammengefasst.

Tabelle 2-2. Die Schalter, die mit `chmod` genutzt werden können

u	Benutzer (User)
g	Gruppe
o	Andere, die nicht in der Gruppe sind (Other)
a	Alle/Jedermann
r	Leseberechtigung (Read)
w	Schreibberechtigung (Write)
x	Ausführberechtigung (eXecute)
+	Berechtigung hinzufügen
-	Berechtigung entfernen

Hier ein paar Beispiele, wie Sie diese Schalter kombinieren können:

```

chmod u+rx,g-rwx,o-rwx wibble.txt ❶
chmod g+wx wobble.txt ❷
chmod -rx,+r wubble.txt ❸

```

- ❶ Nur der Besitzer darf lesen, schreiben und ausführen
- ❷ Berechtigung zum Lesen und Schreiben für die gesamte Gruppe hinzufügen
- ❸ Nur Lesen für jeden hinzufügen

Die einzige Hürde, die Ihre Dateien vor anderen Leuten schützt, ist Ihr Passwort – wählen Sie daher möglichst ein sicheres. Mit dem Befehl `passwd` ändern Sie es. Setzen Sie ihn vor allem ein, wenn Sie Ihren Pi ins Netzwerk hängen.

Das Netzwerk

Hängen Sie im Netz, gibt es eine Reihe von Linux-Tools, die Sie immer wieder einsetzen werden. Für eine Fehlersuche bei der Internetverbindung greifen Sie auf `ifconfig` zurück, mit dem Sie alle Ihre Netzwerk-Schnittstellen und die zugehörigen IP-Adressen anzeigen können (siehe Abbildung 2-6).

```

pi@raspberrypi ~ $ ifconfig
eth0: flags=4096<UP,BROADCAST,MULTICAST> mtu=1500 Metric:1
    inet addr:192.168.2.2 Bcast:192.168.2.255 Mask:255.255.255.0
    RX packets:7632 errors:0 dropped:0 overruns:0 frame:0
    TX packets:7050 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:558086 (545.0 KiB) TX bytes:1680110 (1.6 MiB)

lo:
    Link encap:Local Loopback
    inet addr:127.0.0.1 Mask:255.0.0.0
    UP LOOPBACK RUNNING MTU:16436 Metric:1
    RX packets:14 errors:0 dropped:0 overruns:0 frame:0
    TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:0
    RX bytes:1432 (1.3 KiB) TX bytes:1432 (1.3 KiB)

```

Abbildung 2-6. Der Befehl `ifconfig` liefert Ihnen Informationen über all Ihre Netzwerk-Schnittstellen zurück.

Der Befehl `ping` ist das einfachste Tool, um herauszufinden, warum die Netzwerkverbindung nicht funktioniert. Sie können mit `ping` (denken Sie an den Sonar-Ping eines U-Boots) testen, ob es eine Hin- und Rückverbindung zwischen zwei IP-Adressen im Netzwerk oder Internet gibt. Beachten Sie, dass viele Websites Ping-Anfragen blockieren, daher kann es passieren, dass Sie mehrere Sites anpingen müssen, um eine Verbindung sinnvoll testen zu können:

```

ping yahoo.com
ping altavista.com
ping lycos.com
ping netscape.com ❶

```

- ❶ Hier gibt es einen Fehler.

Um sich remote (und sicher, mit verschlüsselten Passwörtern) an einem anderen Computer anzumelden, können Sie die Secure Shell (ssh) nutzen. Der Computer am anderen Ende muss dazu einen SSH-Server laufen lassen, aber der SSH-Client wird in Raspbian schon mitgeliefert. Tatsächlich ist dies eine sehr praktische Möglichkeit, auf Ihrem Raspberry Pi ohne Monitor oder Tastatur zu arbeiten (siehe weiter unten in Headless laufen).

Zu SSH gehört auch das Programm sftp, mit dem Sie auf sicherem Weg Dateien von einem Computer zu einem anderen transportieren können. Abgerundet wird das Ganze durch scp, mit dem Sie Dateien von einem Computer über das Netzwerk oder Internet auf einen anderen kopieren. Bei all diesen Programmen wird der Secure Sockets Layer (SSL) genutzt, um Dateien mit verschlüsselten Anmeldeinformationen zu übertragen. Diese Tools sind allesamt ausgereifte und erprobte Linux-Tools.

Headless laufen

Wollen Sie auf dem Raspberry Pi arbeiten, ohne Monitor, Tastatur und Maus anzuschließen (also *Headless* laufen), gibt es dazu einige Möglichkeiten. Wenn Sie nur Zugriff auf die Befehlszeile brauchen, können Sie den Raspberry Pi einfach mit dem Netzwerk verbinden und dann einen SSH-Client aufrufen, um sich mit ihm zu verbinden (Benutzername: pi, Passwort: raspberry). Dazu können Sie das Terminal-Tools auf dem Mac nutzen, PuTTY (<http://www.chiark.greenend.org.uk/~sgtatham/putty/>) unter Windows oder Linux oder ssh unter Linux. Der SSH-Server auf dem Raspberry Pi ist standardmäßig aktiv (rufen Sie `raspi-config` nochmals auf, wenn er aus irgendeinem Grund nicht beim Starten mit hochfährt).

Ein anderer Weg, sich mit dem Pi über eine Netzwerk-Verbindung zu unterhalten, ist das Ausführen eines Virtual Network Computing Servers (VNC-Server) auf dem Pi und das Verbinden mit ihm über einen VNC-Client. Vorteil ist, dass Sie eine vollständige grafische Benutzeroberfläche in einem Fenster auf Ihrem Laptop oder Desktop laufen lassen können. Dies ist eine sehr gute Lösung für eine portable Entwicklungsumgebung. Auf dem Raspberry Pi Hub (http://elinux.org/RPi_VNC_Server) finden Sie umfassende Informationen zum Installieren von TightVNC (<http://www.tightvnc.com/>), einem schlanken VNC-Server.

/etc

Das Verzeichnis `/etc` enthält alle systemweiten Konfigurationsdateien und Start-Skripten. Als Sie die Konfigurations-Skripten nach dem ersten Booten Ihres Pis ausführten, wurden eine ganze Reihe von Werten in den diversen Dateien im Verzeichnis `/etc` verändert. Sie benötigen Superuser-Kräfte mit `sudo`, um Dateien in `/etc` zu bearbeiten – lesen Sie in einem Tutorial etwas

über das Bearbeiten einer Konfigurationsdatei, verwenden Sie einen Texteditor und rufen Sie ihn mit `sudo` auf:

```
pi@raspberrypi ~ $ sudo nano /etc/hosts
```

Datum und Uhrzeit setzen

Ein typischer Laptop oder Desktop besitzt zusätzliche Hardware und eine Backup-Batterie (meist eine Knopfzelle), um das aktuelle Datum und die Uhrzeit zu speichern. Der Raspberry Pi besitzt das nicht, aber Raspbian ist so konfiguriert, dass er seine Uhrzeit und das Datum automatisch mit einem Network Time Protocol (NTP) Server synchronisiert, wenn er mit einem Netzwerk verbunden ist.

Es kann für manche Anwendungen sehr wichtig sein, die korrekte Uhrzeit zu kennen (siehe das Beispiel in Kapitel 7, wo mit `cron` eine Lampe angesteuert wird). Um Datum und Uhrzeit manuell zu setzen, nutzen Sie das Programm `date`:

```
$ sudo date --set="Sun Nov 18 1:55:16 EDT 2012"
```

Neue Software installieren

Einer der Bereiche, in denen Linux anderen Betriebssystemen haushoch überlegen ist, ist das Software-Paketmanagement. *Paketmanager* kümmern sich um das Herunterladen und Installieren von Software und sie achten auch darauf, dass Abhängigkeiten aufgelöst und notwendige Komponenten ebenfalls heruntergeladen werden. Auch hier wird der modulare Ansatz verfolgt und viele Softwarepakete bauen unter Linux auf anderen Softwareelementen auf. Der Paketmanager hält Ordnung und die Paketmanager unter Linux sind dabei ausgesprochen robust.

Raspbian bringt zunächst recht wenig Software mit, daher werden Sie ziemlich bald neue Programme herunterladen und installieren wollen. Alle Beispiele in diesem Buch verwenden dafür die Befehlszeile, da es die flexibelste und schnellste Möglichkeit ist, Software zu installieren.

Das Programm `apt-get` mit dem Schalter `-install` wird genutzt, um Software herunterzuladen. `apt-get` lädt auch alle anderen Softwareelemente herunter, die Ihr Paket erfordert, so dass Sie sich nicht selbst um die Abhängigkeiten kümmern müssen. Software muss mit Superuser-Berechtigungen installiert werden, verwenden Sie also immer `sudo` (diese Anweisung installiert den Texteditor Emacs):

```
pi@raspberrypi ~ $ sudo apt-get install emacs
```



Einen Screenshot machen

Eines der ersten Dinge, die wir herausfinden mussten, als wir dieses Buch schrieben, war, wie man auf dem Pi einen Screenshot macht. Wir haben ein Programm namens `scrot` gefunden (eine Kurzform für SCReenshOT). Eine andere Option ist, das GNU Image Manipulation Program (Gimp) oder ImageMagick zu installieren, aber `scrot` war für uns ausreichend. Zum Installieren geben Sie Folgendes ein:

```
sudo apt-get install scrot
```

Weitere Informationen

Es gibt viel mehr zu Linux zu sagen und viele Orte, mehr darüber zu lernen. Ein paar gute Einstiegspunkte dafür sind:

Linux – kurz & gut (<http://www.oreilly.de/catalog/linuxpg2ger>)

So praktisch wie eine Kurzübersicht.

Linux in a Nutshell (<http://www.oreilly.de/catalog/linuxnut4ger>)

Detaillierter, aber immer noch eher eine Kurzübersicht.

The Debian Wiki (<http://wiki.debian.org/FrontPage>)

Raspbian basiert auf Debian, daher sind viele Informationen zu Debian auch für Raspbian hilfreich.

The Jargon File (<http://catb.org/jargon/>)

Auch als *New Hacker's Dictionary* veröffentlicht. Diese Sammlung von Definitionen und Geschichten ist Pflichtlektüre, wenn man sich mit der Unix/Linux-Subkultur befasst.

3/Python auf dem Pi

Python ist eine tolle erste Programmiersprache — sie ist klar strukturiert und man kommt schnell zu Ergebnissen. Wichtiger ist noch, dass es viele andere Anwender gibt, die man bei Problemen fragen und mit denen man Code teilen kann.

Guido van Rossum hat Python entworfen und sehr schnell ihre Eignung als erste Programmiersprache erkannt. 1999 hat van Rossum ein viel gelesenes Proposal namens Computer Programming for Everybody (<http://www.python.org/doc/essays/cp4e.html>) geschrieben, in dem sich eine Vision für ein ambitioniertes Programm zum Programmieren Lernen mit Python an Grund- und Mittelschulen findet. Mehr als ein Jahrzehnt später sieht es so aus, als ob sich diese Vision mit dem Raspberry Pi erfüllt.

Python ist eine *Interpreter-Sprache*. Sie können also ein Programm oder Skript schreiben und es direkt ausführen lassen, statt es erst in Maschinencode zu kompilieren. Mit Interpreter-Sprachen kann man schneller Programme schreiben, und Sie haben ein paar angenehme Nebeneffekte. So müssen Sie zum Beispiel in Python dem Computer nicht explizit mitteilen, ob es sich bei einer Variablen um eine Zahl, eine Liste oder einen String handelt. Der Interpreter ermittelt den Datentyp selbst, wenn Sie das Skript ausführen.

Der Python-Interpreter kann auf zwei Arten ausgeführt werden: als interaktive Shell, um einzelne Befehle zu starten, oder als Befehlszeilenprogramm, um Skripten laufen zu lassen. Die integrierte Entwicklungsumgebung (Integrated Development Environment, IDE), die für Python auf dem Raspberry Pi zur Verfügung steht, nennt sich IDLE (siehe Abbildung 3-1).

Das große Python-Versionsrätsel

Es gibt zwei Versionen der IDLE, weil es zwei Python-Versionen auf dem Pi gibt. Das ist gängige Praxis (allerdings auch ein bisschen verwirrend). Zum Zeitpunkt der Entstehung dieses Buches ist Python 3 die neueste Version, aber es gab beim Wechsel von Version 2 auf Version 3 Änderungen an der Sprache, so dass sie nicht abwärtskompatibel ist. Und obwohl Python 3 schon seit ein paar Jahren im Einsatz ist, hat es eine ganze Zeit gedauert,

bis sich diese neue Version auch verbreitet hat. Viele Pakete, die von Anwendern kommen, sind immer noch nicht auf Python 3 aktualisiert. Wenn Sie nach Python-Dokumentationen suchen, wird es noch verwirrender — achten Sie darauf, die richtige Version zu nutzen!

Die Beispiele in diesem Buch funktionieren sowohl mit Python 2.7 oder 3.X; wenn es nicht der Fall ist, wird explizit darauf hingewiesen.

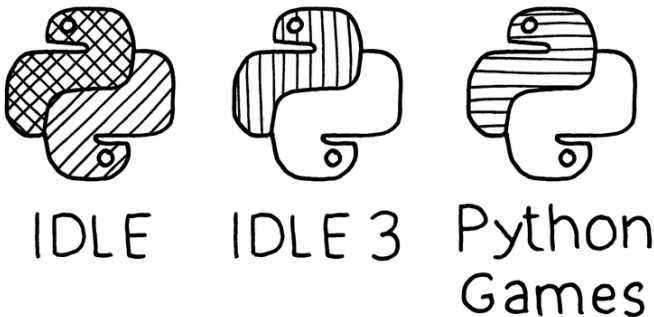


Abbildung 3-1. Python-Optionen auf dem Raspbian Desktop: Die IDLE-Umgebung für Python 2.0 (links), IDLE für Python 3.0 (Mitte) und eine Sammlung von Beispiel-Spielen, die mit Pygame implementiert wurden (rechts, von <http://inventwithpython.com>)

Hallo Python

Am besten steigt man in Python ein, indem man direkt loslegt. Sie können zwar einen beliebigen Texteditor nutzen, um Skripten zu schreiben, aber wir werden mit der IDE einsteigen. Öffnen Sie die Anwendung IDLE 3. Klicken Sie doppelt auf das IDLE-3-Symbol auf dem Desktop oder klicken Sie auf das Desktop-Menü unten links und wählen Sie dann Programming→IDLE 3.

IDLE kann ein paar Sekunden brauchen, um hochzufahren, aber wenn es geladen ist, sehen Sie ein Fenster mit der interaktiven Shell. Das dreifache Größer-Zeichen ist der interaktive Prompt — wenn Sie es sehen, heißt das, dass der Interpreter auf Ihre Eingabe wartet. Geben Sie dort ein:

```
>>> print("Saluton Mondo!")
```

Drücken Sie nun die Eingabetaste. Python führt diese Anweisung aus, und Sie werden das Ergebnis im Shell-Fenster sehen. Beachten Sie, dass der Be-

fehl `print()` einer derjenigen ist, die sich mit Python 3.0 geändert haben. Wenn Sie einen Syntax-Fehler erhalten, achten Sie darauf, dass Sie in der Version 3.0 von IDLE arbeiten.

Sie können die Shell als eine Art Taschenrechner nutzen, um Anweisungen auszuprobieren oder Berechnungen durchzuführen:

```
>>> 3+4+5
12
```

Stellen Sie sich die Anweisungen, die in der interaktiven Shell ausgeführt werden, als ein Programm vor, das Sie Zeile für Zeile laufen lassen. Sie können Variablen einrichten oder Module importieren:

```
>>> import math
>>> (1 + math.sqrt(5)) / 2
1.618033988749895
```

Die `import`-Anweisung stellt Ihrem Programm alle mathematischen Funktionen von Python zur Verfügung (mehr zu Modulen in „Objekte und Module“ (S. 41)). Um eine Variable einzurichten, nutzen Sie den Zuweisungsoperator (`=`):

```
>>> import math
>>> radius = 200
>>> radius * 2 * math.pi
125.66370614359173
```

Wollen Sie alle Variablen abräumen und in einem sauberen Status neu beginnen, wählen Sie `Shell→Restart Shell`. Sie können auch die interaktive Shell verwenden, um Informationen über den Einsatz einer bestimmten Anweisung, eines Moduls oder anderer Python-Themen zu erhalten. Dazu greifen Sie auf den Befehl `help()` zurück:

```
help("print")
```

Um eine Liste aller verfügbaren Themen zu erhalten, probieren Sie einmal Folgendes aus:

```
help("topics")
help("keywords")
help("modules")
```

Der Python-Interpreter ist ideal zum Austesten von Anweisungen oder einfachen Operationen. Häufig werden Sie aber Ihr Python-Skript wie eine eigenständige Anwendung ausführen wollen. Um ein neues Python-Programm aufzusetzen, wählen Sie `File→New Window`, dann bietet Ihnen IDLE ein Fenster zur Skript-Bearbeitung (siehe Abbildung 3-2).

Geben Sie einmal eine Codezeile ein und rufen Sie dann `Run→Run Module` auf. Sie werden eine Warnung erhalten: „Source Must Be Saved OK

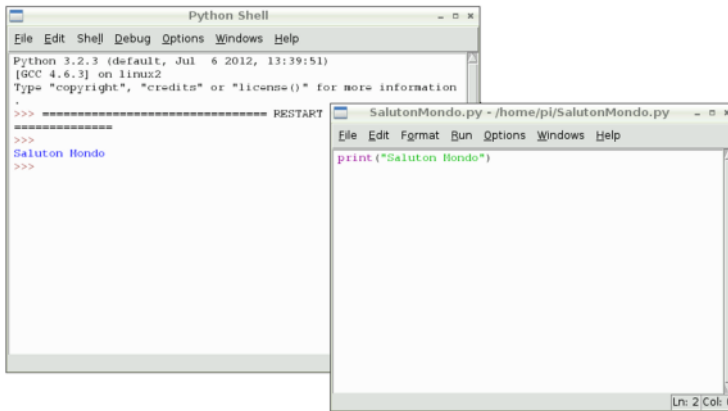


Abbildung 3-2. Die interaktive Shell IDLE (links) und ein Editor-Fenster (rechts)

To Save?". Sichern Sie Ihr Skript in Ihrem Home-Verzeichnis als *SalutonMondo.py* und sehen Sie dann, wie es in der Shell ausgeführt wird.

Manchmal werden Sie die IDLE-Umgebung nicht nutzen wollen. Um ein Skript von der Befehlszeile auszuführen, öffnen Sie das LXTerminal und tippen Sie:

```
python SalutonMondo.py
```

Das sind schon alle grundlegenden Mechanismen, die Sie kennen müssen, um mit der Umgebung vertraut zu sein. Als Nächstes müssen Sie noch die Sprache erlernen.

Befehlszeile oder IDLE?

Ihnen wird vielleicht auffallen, dass die Ausgabe von IDLE beim Ausführen von Beispielcode, der Text in der Shell ausgibt, sehr langsam ist. Für einen Vergleich können Sie IDLE und ein neues LXTerminal parallel offen haben. Sichern Sie das weiter unten stehende übernächste Skript in IDLE als *CountEvens.py* und geben Sie an der Befehlszeile Folgendes ein:

```
python CountEvens.py
```

Bevor Sie das tun, lassen Sie das gleiche Skript in IDLE noch per Run Module laufen — es benötigt einen Vorsprung. Sie werden schnell merken, wie stark die IDE das Skript auf dem Pi ausbremst, um seine Arbeit erledigen zu können. Die Beispiele weiter unten im Buch werden alle an der Befehlszeile ausgeführt, Sie können aber IDLE natürlich trotzdem als Editor nutzen.

Ein bisschen mehr Python

Wenn Sie sich Python aus der Arduino-Welt nähern, sind Sie es gewohnt, Programme (in Arduino *Sketches* genannt, in Python eher *Skripten*) in einem `setup/loop`-Format zu schreiben, wobei `setup()` eine Funktion ist, die nur einmal ausgeführt wird, während die Funktion `loop()` wieder und wieder läuft. Das folgende Beispiel zeigt, wie man das in Python erreichen kann. Wählen Sie an der Shell in IDLE 3 den Befehl `New Window` und geben Sie das Folgende ein:

```
# Setup
n = 0

# Loop
while True:
    n = n + 1
    # % ist der Modulo-Operator
    if ((n % 2) == 0):
        print(n)
```

Wählen Sie `Run Module` und geben Sie Ihrem Skript einen Namen (wie zum Beispiel *EvenIntegers.py*). Während es läuft, sollten alle geraden Zahlen ausgegeben werden (drücken Sie aber irgendwann `Strg-C`, um die Ausgabe abubrechen, sonst läuft es in alle Ewigkeit).



Im obigen Beispiel ist Ihnen vielleicht aufgefallen, dass jede Einrück-Stufe aus vier Leerzeichen besteht, nicht aus einem Tabulator-Zeichen. (Sie können in IDLE allerdings `Tab` drücken, es werden dann vier Leerzeichen eingefügt.) In Python ist das Einrücken nicht nur eine Lesehilfe, sondern es hat auch eine strukturelle Bedeutung. Dies ist einer der größten Fallstricke für Einsteiger oder beim Kopieren und Einfügen von Code. Wir sind allerdings trotzdem der Meinung, dass das verpflichtende Einrücken Python zu einer sehr gut lesbaren Sprache macht. In den *Python Style Guidelines* (<http://www.python.org/dev/peps/pep-0008/#indentation>) finden Sie Tipps zum Schreiben von lesbarem Code.

Es ist wichtig, auf Ihre Leerzeichen zu achten. Python ist eine sehr strukturierte Sprache, bei der Whitespace (also Leerzeichen, Tabs und so weiter) die Struktur festlegen. Im nächsten Beispiel ist alles, was unter der `loop()`-Funktion um eine Ebene eingerückt ist, Teil dieser Funktion. Das Ende einer Schleife wird durch die Zeile bestimmt, in der die Einrückung wieder eine Ebene zurückgesetzt ist (oder durch das Ende der Datei). Das unterscheidet Python von Sprachen wie C, bei denen Blöcke durch geschweifte Klammern oder andere Kennzeichen abgeschlossen werden.

Verwenden Sie Funktionen, um Codeabschnitte in einen Codeblock umzuwandeln, der an anderer Stelle in Ihrem Skript aufgerufen werden kann. Um das vorige Beispiel mit Funktionen umzuschreiben, nutzen Sie den folgenden Code (wenn Sie es ausführen wollen, speichern Sie es als *Count-Evens.py*):

```
# Globale Variablen deklarieren
n = 0 ❶

# Setup-Funktion
def setup(): ❷
    global n
    n = 100

def loop(): ❸
    global n
    n = n + 1
    if ((n % 2) == 0):
        print(n)

# Main ❹
setup()
while True:
    loop()
```

In diesem Beispiel wird jede gerade Zahl ab 102 ausgegeben. So funktioniert das Programm:

- ❶ Als Erstes wird die Variable *n* als globale Variable definiert, die in allen Blöcken des Skripts genutzt werden kann.
- ❷ Hier wird die Funktion *setup()* definiert (aber noch nicht ausgeführt).
- ❸ Genauso findet sich hier die Definition der Funktion *loop()*.
- ❹ Im Haupt-Codeblock wird *setup()* einmalig aufgerufen und danach *loop()*.

Die Verwendung des Schlüsselworts *global* in der ersten Zeile jeder Funktion ist wichtig – sie weist den Interpreter an, die globale Variable *n* zu verwenden und keine zweite (lokale, oder für diese Funktion *private*) Variable *n* zu erzeugen, die nur hier genutzt werden kann.

Dieses Tutorial ist zu kurz für eine vollständige Python-Referenz. Um die Sprache wirklich zu erlernen, schauen Sie sich vielleicht einmal Programmieren lernen mit Python (<http://www.oreilly.de/catalog/thinkpythonger>) und Python – kurz & gut (<http://www.oreilly.de/catalog/pythonpr4ger>) an. Der Rest dieses Kapitels wird Sie mit genug Wissen ausstatten, dass Sie die späteren Beispiele verstehen und ausführen können. Dabei lernen Sie die grundlegenden Features und verfügbaren Module kennen. Kapitel 4 behandelt das Framework PyGame – eine gute Möglichkeit, Multimedia-Programme auf dem Pi zu erstellen.

Objekte und Module

Sie müssen die Syntax-Grundlagen der Objekte und Module verstehen, um die Beispiele in diesem Buch nutzen zu können. Python ist eine aufgeräumte Sprache mit nur 34 reservierten *Schlüsselwörtern* (siehe Tabelle 3-1). Diese Schlüsselwörter bilden den Kern der Sprache, mit dem Sie die Struktur und den Ablauf in Ihrem Skript definieren. So gut wie alles, was kein Schlüsselwort ist, können Sie sich als ein *Objekt* vorstellen. Dabei handelt es sich um eine Kombination aus Daten und Verhaltensweisen mit einem Namen. Sie können die Daten eines Objekts ändern, Informationen darüber auslesen und sogar andere Objekte beeinflussen.

Tabelle 3-1. *Python besitzt nur 34 reservierte Schlüsselwörter*

Bedingte Anweisungen	Schleifen	Eingebaute Funktionen	Klassen, Module, Funktionen	Fehlerbehandlung
if	for	print	class	try
else	in	pass	def	except
elif	while	del	global	finally
not	break		lambda	raise
or	as		nonlocal	assert
and	continue		yield	with
is			import	
True			return	
False			from	
None				

In Python sind Strings, Listen, Funktionen, Module und selbst Zahlen Objekte. Ein Python-Objekt kann man sich als gekapselte Sammlung von Attributen und Modulen vorstellen. Sie erhalten Zugriff auf diese Attribute und Methoden über eine einfache Punkt-Syntax. Geben Sie zum Beispiel den folgenden Code am interaktiven Shellprompt ein, um ein String-Objekt zu erstellen und die Methode aufzurufen, mit der es bei sich selbst den Wortanfang in Großbuchstaben umwandelt:

```
>>> myString = "quux"  
>>> myString.capitalize()  
'Quux'
```

Oder nutzen Sie `reverse()`, um die Reihenfolge einer Liste umzukehren:

```
>>> myList = ['a', 'man', 'a', 'plan', 'a', 'canal']
>>> myList.reverse()
>>> print(myList)
['canal', 'a', 'plan', 'a', 'man', 'a']
```



Sowohl String als auch List sind eingebaute Module der *Standard-Bibliothek*, die jedem Python-Programm zur Verfügung stehen. In den beiden Modulen finden sich sehr viele Funktionen für den Umgang mit Strings und Listen, wie zum Beispiel `capitalize()` und `reverse()`.

Manche der Module der Standard-Bibliothek sind nicht eingebaut und Sie müssen Python mit der Anweisung `import` explizit mitteilen, dass Sie sie nutzen wollen. Um das Modul `time` aus der Standard-Bibliothek zu nutzen und Zugriff auf hilfreiche Funktionen für den Umgang mit Zeitwerten und Timestamps zu erhalten, nutzen Sie:

```
import time
```

Sie werden auch einmal die Anweisung `import as` sehen, um das Modul in Ihrem Programm umzubenennen:

```
import time as myTime
```

Dazu kommt noch `from import`, um ausgewählte Funktionen aus einem Modul zu laden:

```
from time import clock
```

Hier ein kurzes Beispiel für ein Python-Skript, das die Module `time` und `datetime` aus der Standard-Bibliothek nutzt, um in jeder Sekunde die aktuelle Uhrzeit auszugeben:

```
from datetime import datetime
from time import sleep

while True:
    now = str(datetime.now())
    print(now)
    sleep(1)
```

Die Funktion `sleep` stoppt die Ausführung des Programms für eine Sekunde. Beim Ausführen des Codes wird Ihnen auffallen, dass sich die Zeit auf Dauer ein wenig verschiebt. Das hat zwei Gründe:

1. Der Code berücksichtigt nicht, dass das Berechnen der aktuellen Uhrzeit auch Zeit kostet (daher wären 0,9 Sekunden Wartezeit die bessere Wahl).
2. Andere Prozesse nutzen ebenfalls die CPU und nehmen Ihrem Programm Rechenzeit weg. Daran sollten Sie immer denken: Wenn Sie auf dem Raspberry Pi programmieren, arbeiten Sie nicht in einer *Real Time*-Umgebung.

Verwenden Sie die Funktion `sleep()`, werden Sie bemerken, dass sie eine Genauigkeit von ungefähr 5 ms besitzt.

Als Nächstes wollen wir das Beispiel anpassen, eine Textdatei öffnen und regelmäßig Daten protokollieren. Wenn es um Textdateien geht, ist alles ein String. Mit der Funktion `str()` wandeln Sie Zahlen in Strings um (mit `int()` können Sie den umgekehrten Weg gehen).

```
from datetime import datetime
from time import sleep
import random

log = open("log.txt", "w")

while i in range(5):
    now = str(datetime.now())
    # Zufallszahlen im Bereich von 0 bis 1024 erzeugen
    data = random.randint(0, 1024)
    log.write(now + " " + str(data) + "\n")
    print(".")
    sleep(.9)
log.flush()
log.close()
```



Achten Sie bei einer echten Anwendung, die Daten protokolliert, darauf, dass der Raspberry Pi Datum und Uhrzeit korrekt eingestellt hat (siehe Kapitel 1).

Hier ein weiteres Beispiel (*ReadFile.py*), bei dem ein Dateiname als Argument eingelesen wird (in der Shell starten Sie das Programm mit `python3 ReadFile.py Dateiname`). Das Programm öffnet die Datei, liest jede Zeile als String ein und gibt ihn aus. Beachten Sie, dass `print()` wie `println()` in an-

deren Sprachen funktioniert – die Funktion fügt bei der Ausgabe des Strings einen Zeilenumbruch aus. Mit dem Argument `end` bei `print()` wird dieser Zeilenumbruch verhindert.

```
# Eine Datei als Befehlszeilenparameter öffnen und einlesen
import sys

if (len(sys.argv) != 2):
    print("Usage: python ReadFile.py filename")
    sys.exit()

scriptname = sys.argv[0]
filename = sys.argv[1]

file = open(filename, "r")
lines = file.readlines()
file.close()

for line in lines:
    print(line)
```

Noch mehr Module

Einer der Gründe für die Beliebtheit von Python ist die große Zahl an von Anwendern bereitgestellten Modulen, die auf der Standard-Bibliothek aufbauen. Der Python Package Index (PyPI) (<http://pypi.python.org/pypi>) ist die umfassendste Liste mit Paketen (oder Modulen). Manche der beliebtesten Module, die auch für den Raspberry Pi sehr nützlich sein können, sind in Tabelle 3-2 aufgeführt. Sie werden einige dieser Module später noch verwenden, insbesondere das Modul GPIO für den Zugriff auf die Ein- und Ausgabepins des Raspberry Pi.

Um eines dieser Module zu nutzen, müssen Sie den Code herunterladen, das Paket konfigurieren und installieren. Das Modul `serial` kann zum Beispiel auf diesem Weg installiert werden:

```
sudo apt-get install python-serial
```

Wurde ein Paket von seinem Ersteller auf dem üblichen Weg (mit Pythons `distutils`-Tool) aus seinen Modulen gebaut, müssen Sie es nur herunterladen, entpacken und das Folgende eingeben:

```
python setup.py install
```

Schauen Sie sich auch einmal den Pip Package Installer (<http://www.pip-installer.org>) an, ein Tool, mit dem das Installieren von Paketen von PyPI sehr einfach ist.

Tabelle 3-2. Pakete, die für Pi-Anwender besonders interessant sind

Modul	Beschreibung	URL	Paketname
RPi.GPIO	Zugriff auf die GPIO-Pins	http://code.google.com/p/raspberry-gpio-python/	python-rpi.gpio
Pygame	Gaming-Framework	http://pygame.org	python-pygame
SimpleCV	Einfache API für Computer Vision	http://simplecv.org/	Kein Paket
Scipy	Wissenschaftliches Rechnen	http://www.scipy.org/	python-scipy
Numpy	Die numerischen Grundlagen zu Scipy	http://numpy.scipy.org/	python-numpy
Flask	Microframework für Webentwicklung	http://flask.pocoo.org/	python-flask
Requests	"HTTP für Menschen"	http://ocs.python-requests.org	python-requests
PIL	Bildverarbeitung	http://www.pythonware.com/products/pil/	python-imaging
wxPython	GUI-Framework	http://wxpython.org	python-wxgtk2.8
PySerial	Zugriff auf den seriellen Port	http://pyserial.sourceforge.net/	python-serial
pyUSB	FTDI-USB-Schnittstelle	http://bleyer.org/pyusb	Kein Paket

Fehlersuche

Sie werden mit Ihrem Code ganz unvermeidlich in Probleme laufen und einen Fehler suchen müssen. Der IDLE Interactive Mode kann hier sehr hilfreich sein – das Debug-Menü bietet eine Reihe von Tools, mit denen Sie herausbekommen können, was Ihr Code tatsächlich tut. Sie haben auch die Möglichkeit, alle Ihre Variablen anzeigen zu lassen und den Code Zeile für Zeile ausführen zu lassen.

Syntax-Fehler lassen sich am einfachsten beheben – normalerweise handelt es sich nur um einen Tippfehler oder ein falsches Verständnis der Programmiersprache. *Semantische Fehler* – bei denen das Programm an sich korrekt ist, aber nicht das tut, was es soll – können schwieriger zu finden sein. Hier kann der Debugger sehr hilfreich sein. Es braucht Jahre, effektives Debug-

gen zu erlernen, aber hier eine kurze Checkliste, die Sie bei der Arbeit mit Python auf dem Pi berücksichtigen sollten:

- Nutzen Sie `print()`, um anzuzeigen, dass das Programm einen bestimmten Punkt erreicht hat.
- Mit `print()` können Sie auch die Werte von Variablen ausgeben lassen.
- Achten Sie ganz besonders auf Whitespace, um sicherzustellen, dass die Blöcke so definiert sind, wie Sie sich das vorgestellt haben.
- Denken Sie beim Debuggen von Syntax-Fehlern daran, dass der tatsächliche Fehler weit vor der Stelle liegen kann, die der Interpreter meldet.
- Prüfen Sie noch einmal alle Ihre globalen und lokalen Variablen.
- Achten Sie auf korrekt schließende Klammerpaare.
- Schauen Sie sich die Reihenfolge der Operationen bei Rechnungen an – wenn Sie sich nicht sicher sind, nutzen Sie Klammern. So liefern zum Beispiel $3 + 4 * 2$ und $(3 + 4) * 2$ verschiedene Ergebnisse.

Nachdem Sie sich mit Python vertraut gemacht und Erfahrung gesammelt haben, sollten Sie einmal einen Blick auf die Module `code` und `logging` werfen – dort finden Sie weitere Debugging-Tools.

Weitere Informationen

Es gibt noch viel mehr zu Python zu sagen. Hier ein paar nützliche Ressourcen:

Programmieren lernen mit Python von Allen Downey

(<http://www.oreilly.de/catalog/thinkpythonger>)

Dies ist ein klares und recht kompaktes Buch zum Programmieren Lernen (in Python).

Python – kurz & gut (<http://www.oreilly.de/catalog/pythonpr4ger>)

Weil es manchmal besser ist, ein Buch durchzublättern, als sich durch ein Dutzend Posts bei Stack Overflow durchzuklicken.

Stack Overflow (<http://stackoverflow.com/>)

Stack Overflow ist trotzdem eine ausgezeichnete Quelle kollektiven Wissens. Das funktioniert besonders gut, wenn Sie nach einer bestimmten Lösung oder Fehlermeldung suchen – es kann sehr gut sein, dass jemand anderes das gleiche Problem hatte.

Learn Python the Hard Way (<http://learnpythonthehardway.org/>)

Ein tolles Buch und eine tolle Online-Ressource – lesen Sie zumindest die Einführung „The Hard Way Is Easier.“

4/Animation und Multimedia in Python

Pygame ist ein schlankes Framework, mit dem man in Python einfache Spiele erstellen kann. Sie können es sich auch als ein Tool zur allgemeinen Multimedia-Programmierung vorstellen — es ist sehr gut dazu geeignet, Grafiken auf dem Bildschirm anzuzeigen, Klänge abzuspielen oder mit Tastatur- und Maus-Ereignissen umzugehen.

Pygame ist ein Software-Wrapper für eine andere Bibliothek namens Simple DirectMedia Layer (SDL). SDL kümmert sich um die ganzen Low-Level-Zugriffe auf Tastatur, Maus, Audio- und Videotreiber des Pi. Pygame vereinfacht SDL noch weiter.

Der Fokus dieses Kapitels liegt mehr auf den einfachen Multimedia-Fähigkeiten von Pygame und weniger auf Erklärungen, wie man Spiele programmiert. Ressourcen dazu finden Sie am Ende des Kapitels.

Hallo Pygame

Pygame ist auf Ihrem Raspberry Pi schon vorinstalliert. Es gibt eine Version von Pygame, die mit Python 3.0 zusammenarbeitet, aber die auf dem Raspberry mitgelieferte Version funktioniert nur mit Version 2.7. Sie können entweder zur Nicht-3.0-Version von IDLE wechseln (klicken Sie einfach doppelt auf das IDLE-Symbol statt auf das IDLE-3-Symbol) oder Ihr Skript einfach an der Befehlszeile ausführen — dann wird der 2.7-Interpreter genutzt.

Das folgende Beispiel zeigt die minimal notwendigen Schritte, um ein Pygame-Programm zu erzeugen. Es wird in einem neuen Fenster ein roter Kreis gezeichnet:

```
import pygame ❶  
  
width = 640 ❷  
height = 480  
radius = 100
```

```

fill = 1

pygame.init() ❸

window = pygame.display.set_mode((width, height)) ❹
window.fill(pygame.Color(255, 255, 255)) ❺

while True: ❻
    pygame.draw.circle(window, ❼
                        pygame.Color(255, 0, 0),
                        (width/2, height/2),
                        radius, fill)
    pygame.display.update() ❽

```

- ❶ Das Modul `pygame` wird importiert, womit alle Pygame-Objekte und -Funktionen verfügbar sind.
- ❷ Globale Variablen einrichten.
- ❸ Es wird immer zuerst die Funktion `init()` aufgerufen, die die zum Initialisieren der Pygame-Submodule notwendigen Schritte durchführt.
- ❹ Das Fenster wird eingerichtet. Dabei handelt es sich um ein *Surface*-Objekt von Pygame — ein Bereich, auf dem gezeichnet werden kann.
- ❺ Das Fenster wird mit Weiß gefüllt.
- ❻ Unendliche Schleife. Sie können sich jede Iteration dieser Schleife als einzelnen Frame einer unendlichen Animation vorstellen.
- ❼ Ein gefüllter roter Kreis wird in der Mitte des Fensters gezeichnet.
- ❽ Alle Zeichenbefehle in der Schleife bilden einen Animationsframe und werden in einen verborgenen Puffer gezeichnet. Sind Sie damit fertig, rufen Sie `display.update()` auf, um das Bild auf dem Bildschirm zu aktualisieren.



In allen diesen Beispielen müssen Sie an der Befehlszeile Strg-C drücken, um das Skript zu stoppen, wenn Sie fertig sind. Wollen Sie den Close-Button im Pygame-Fenster aktivieren, fügen Sie den folgenden Code am Ende der `while`-Schleife ein:

```

while True:
    if pygame.QUIT in [e.type for e in pygame.event.get()]:
        break

```

Damit wird das Ereignis abgefangen, wenn der Button gedrückt wird. Mehr über Ereignisse finden Sie in „Ereignisse und Eingaben“ (S. 51).

Pygame besteht aus einer Reihe von Submodulen und Objekten — die Grundlagen ihres Einsatzes werden im Rest dieses Kapitels beschrieben.

Pygame-Surfaces

Ein Pygame-Surface ist einfach ein rechteckiges Bild. Surfaces werden kombiniert und übereinandergelegt, um die Szenen in einem Frame des Spiels oder der Animation zu erstellen. Die Pixel eines Surface werden als Folge von drei 8-Bit-RGB-Werten dargestellt, zum Beispiel steht (0, 255, 0) für Grün. Fügen Sie eine vierte Zahl hinzu, wird auch die Transparenz mit abgebildet: (0, 255, 0, 127) ist zu 50% transparent.

Das Display-Window ist das Basis-Surface, auf dem alle anderen Fenster gezeichnet werden. Das Modul `pygame.display` kontrolliert und liefert Informationen über das Display-Window. Mit der Funktion `set_mode()` erzeugen Sie ein neues Display-Window und mit der Funktion `update()` wird jeder Frame neu gezeichnet.

Um ein Bild aus einer Datei in ein Surface zu laden, das angezeigt werden soll, verwenden Sie die Funktion `load()` aus dem Modul `pygame.image`. Haben Sie einmal ein Display-Surface, können Sie weitere Bilder hinzufügen, indem Sie ein neues Surface-Objekt anlegen und die Funktion `blit()` nutzen, um es mit dem bestehenden zu kombinieren:

```
import pygame

pygame.init()
screen = pygame.display.set_mode((450, 450))
background = pygame.image.load("background.png") ❶
background.convert_alpha() ❷
screen.blit(background, (0, 0)) ❸
while True:
    pygame.display.update()
```

- ❶ Lädt ein Hintergrundbild, das sich im gleichen Verzeichnis wie das Python-Programm befinden muss.
- ❷ Die Funktion `convert_alpha()` wandelt das Format des Surface um, damit es dem aktuellen Display entspricht. Das ist nicht erforderlich, beschleunigt aber das Kombinieren der Bilder.
- ❸ Standardmäßig ist das Bildschirm-Surface schwarz. Die Funktion `blit()` kombiniert das Background-Surface mit einem schwarzen Hintergrund.

Hier ein Beispiel für das Kombinieren zweier Bilder (siehe Abbildung 4-1):

```
import pygame

pygame.init()
screen = pygame.display.set_mode((450, 450))
background = pygame.image.load("background.png").convert_alpha()
theremin = pygame.image.load("theremin.png").convert_alpha()
screen.blit(background, (0, 0))
screen.blit(theremin, (135, 50))
while True:
    pygame.display.update()
```



Abbildung 4-1. Zwei Bilder per blit kombinieren

Das Modul `pygame.transform` stellt Funktionen zum Rotieren und Skalieren von Surfaces bereit. Auf die einzelnen Pixel eines Surface können Sie mit den Funktionen des Moduls `pygame.surfarray` zugreifen.

Surfaces sind beim Erstellen immer rechteckige Bilder; auch wenn die transparenten Bereiche des vorigen Beispiel-Surface so kombiniert wurden, dass es nicht mehr rechteckig aussah, hat es trotzdem diese Form. Soll ein Surface nicht rechteckig sein (zum Beispiel für eine pixelgenaue Kollisionserkennung), können Sie mit dem Modul `pygame.mask` eine Maske eines anderen Surface setzen. Masken werden detaillierter in den Ressourcen zur Spieleprogrammierung am Ende des Kapitels beschrieben.

Auf Surfaces zeichnen

Sie haben weiter oben ein Beispiel für die Pygame-Funktion zum Zeichnen von Kreisen kennengelernt. Das Modul `pygame.draw` stellt viele weitere Methoden für Rechtecke, Linien, Kreisbögen und Ellipsen bereit. Mit dem Modul `pygame.gfxdraw` gibt es eine andere Möglichkeit, Formen zu zeichnen. Es stellt ein paar mehr Optionen bereit, wird aber als experimentell beschrieben, und die API ändert sich eventuell noch.

Um Text auszugeben, erstellen Sie zunächst ein neues *Font*-Objekt (aus dem Modul `pygame.font`), dann nutzen Sie dieses Objekt, um einen Font zu laden und den Text zu rendern. Um eine Liste aller auf Ihrem Raspberry Pi verfügbaren Schriften zu erhalten, gibt es die Funktion `pygame.font.get_fonts()`:

```
import pygame

pygame.init()
for fontname in pygame.font.get_fonts():
    print fontname
```

Wie Sie sehen, gibt es in der Basis-Distribution von Raspbian eine Handvoll Schriften. Nutzen Sie das *SysFont*-Objekt, um Text mit der Schriftart *freese-rif* zu rendern:

```
import pygame

pygame.init()
screen = pygame.display.set_mode((725, 92))
font = pygame.font.SysFont("freeserif", 72, bold = 1)
textSurface = font.render("1 Theremin pro Kind!", 1,
                           pygame.Color(255, 255, 255))
screen.blit(textSurface, (10, 10))
while True:
    pygame.display.update()
```



Wollen Sie mehr Schriften bekommen, nutzen Sie die beiden folgenden Pakete:

```
sudo apt-get install ttf-mscorefonts-installer
sudo apt-get install ttf-liberation
```

Ereignisse und Eingaben

In Pygame werden vom Anwender ausgelöste Ereignisse, wie zum Beispiel das Drücken einer Taste oder das Klicken und Bewegen der Maus, als *Event*-Objekt gefangen und in die *Event Queue* gestellt, wo sie warten, bis Ihr Programm etwas damit macht. Das Modul `pygame.event` bietet Funktionen für das Verarbeiten dieser Events, die sich seit Ihrem letzten Blick in die Queue angesammelt haben. Sie können sogar Ihre eigenen Ereignis-Typen definieren, um ein Messaging-System zu implementieren. Im Folgenden finden Sie ein einfaches Beispiel, in dem das Programm mit dem roten Kreis erweitert wird: In jedem Frame wird der Kreis an der aktuellen Mausposition neu gezeichnet, wobei der Radius umso größer wird, je näher sich der Kreis am Rand des Fensters befindet (siehe Abbildung 4-2).

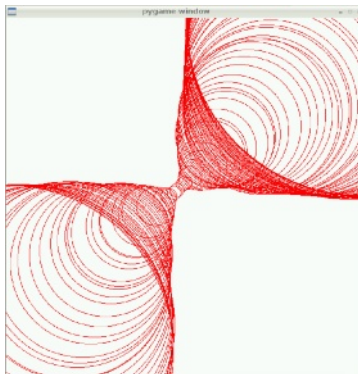


Abbildung 4-2. Ausgabe des Pygame-Events-Beispiels

```

import pygame
from pygame.locals import * ❶

width, height = 640, 640
radius = 0
mouseX, mouseY = 0, 0 ❷

pygame.init()
window = pygame.display.set_mode((width, height))
window.fill(pygame.Color(255, 255, 255))

fps = pygame.time.Clock() ❸

while True: ❹
    for event in pygame.event.get(): ❺
        if event.type == MOUSEMOTION: ❻
            mouseX, mouseY = event.pos
        if event.type == MOUSEBUTTONDOWN: ❼
            window.fill(pygame.Color(255, 255, 255))
            radius = (abs(width/2 - mouseX)+abs(height/2 - mouseY))/2 + 1 ❸
            pygame.draw.circle(window, ❾
                               pygame.Color(255, 0, 0),
                               (mouseX, mouseY),
                               radius, 1)
            pygame.display.update()
            fps.tick(30) ❿

```

- ❶ Das Modul `pygame.locals` definiert eine Reihe von Konstanten wie `MOUSEMOTION`. Durch das Importieren können wir sie nutzen, ohne ihnen immer ein `pygame.` voranstellen zu müssen.
- ❷ Variablen, um die Mauskoordinaten zu sichern.
- ❸ Diese Funktion initialisiert ein Objekt, das wir als Frame-Zähler verwenden werden. Mit dieser Variablen `fps` (Frames per Second) können Sie in jedem Frame eine bestimmte Zeit warten, um eine gleichmäßige Framerate zu erhalten.
- ❹ Unendliche Schleife; jedes Durchlaufen der Schleife ist ein Frame.
- ❺ Durchlaufen der Event Queue. Bei jedem Durchlauf findet sich in der Variable `event` das nächste Event der Queue.
- ❻ Handelt es sich beim Ereignis um eine Mausbewegung, wird die Variable aktualisiert, die die Mausposition speichert.
- ❼ Handelt es sich beim Ereignis um einen Mausklick, wird der Bildschirm geleert.
- ❸ Variiere den Radius in Abhängigkeit vom Abstand der Mausposition vom Mittelpunkt.
- ❾ Zeichne den Kreis.
- ❿ Warte, so dass die Framerate 30 fps ist.

Andere Module für den Umgang mit Ereignissen und Benutzereingaben sind:

pygame.time

Ein Modul, um mit Zeiten zu arbeiten.

pygame.mouse

Ein Modul, um Informationen über die Maus zu erhalten.

pygame.key

Ein Modul, um Informationen über die Tastatur zu erhalten. Dazu sind hier viele Konstanten für die einzelnen Tasten enthalten.

pygame.joystick

Ein Modul für die Arbeit mit einem Joystick.

Den gesamten Bildschirm einnehmen

Um Ihr Programm in einem Fenster laufen zu lassen, das den gesamten Bildschirm einnimmt, nutzen Sie den Schalter `pygame.FULLSCREEN`, wenn Sie den Display Mode setzen. Im Full Screen Mode sollten Sie darauf achten, eine Möglichkeit zum Verlassen des Skripts anzubieten, da Sie es mit Strg-C nicht mehr beenden können:

```
import pygame
import random
from time import sleep

running = True
pygame.init()
screen=pygame.display.set_mode((0,0), pygame.FULLSCREEN)
while running:
    pygame.draw.circle(
        screen,
        pygame.Color(int(random.random()*255),
                      int(random.random()*255),
                      int(random.random()*255)),
        (int(random.random()*1500),
         int(random.random()*1500)),
        int(random.random()*500), 0)
    pygame.display.update()
    sleep(.1)
    for event in pygame.event.get():
        if event.type == pygame.KEYDOWN:
            running = False
pygame.quit()
```

Das Skript wird beendet, wenn eine beliebige Taste gedrückt wird.

Sprites

Die meisten beweglichen und steuerbaren grafischen Elemente eines Spiels werden als *Sprites* realisiert werden. Das Modul `pygame.sprite` stellt alle grundlegenden Funktionen zum Zeichnen von Sprites und zur Kollisionserkennung bereit. Sprites können auch gruppiert werden, um sie gemeinsam steuern und aktualisieren zu können. Ein vollständiges, lauffähiges Spiel, das Sprites einsetzt, würde allerdings den Rahmen dieses Buches sprengen (der Abschnitt „Weitere Informationen“ (S. 58) verweist dafür auf weitere Ressourcen).

Sprites werden am besten dann eingesetzt, wenn Sie viele Bildschirm-Elemente erzeugen, die zum Großteil den gleichen Code nutzen. Hier ein Beispiel für das Erstellen und Aktualisieren mehrerer Sprites; als Ergebnis erhält man zwei Bälle, die von den Seiten des Bildschirms zurückspringen. Sie können weitere Bälle hinzufügen, indem Sie neue Sprites mit anderen Startkoordinaten, eigener Startrichtung und -geschwindigkeit erzeugen.

```
import pygame

class Ball(pygame.sprite.Sprite): ❶

    def __init__(self, x, y, xdir, ydir, speed): ❷
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.Surface([20, 20])
        self.image.fill(pygame.Color(255, 255, 255))
        pygame.draw.circle(self.image,
                           pygame.Color(255,0,0),
                           (10,10), 10, 0)
        self.rect = self.image.get_rect()
        self.x, self.y = x, y ❸
        self.xdir, self.ydir = xdir, ydir
        self.speed = speed

    def update(self): ❹
        self.x = self.x + (self.xdir * self.speed)
        self.y = self.y + (self.ydir * self.speed)
        if (self.x < 10) | (self.x > 490):
            self.xdir = self.xdir * -1
        if (self.y < 10) | (self.y > 490):
            self.ydir = self.ydir * -1
        self.rect.center = (self.x, self.y)

pygame.init()
fps = pygame.time.Clock()
window = pygame.display.set_mode((500, 500))
ball = Ball(100, 250, 1, 1, 5) ❺
ball2 = Ball(400, 10, -1, -1, 8)

while True:
    ball.update() ❻
    ball2.update()
    window.fill(pygame.Color(255,255, 255))
```

```

window.blit(ball.image, ball.rect) ❶
window.blit(ball2.image, ball2.rect)
pygame.display.update()
fps.tick(30)

```

- ❶ Die Anweisung `class` erzeugt ein neues Objekt (einen Ball) basierend auf dem Sprite-Objekt. Sie können dann Ihre eigenen Funktionen zum Zeichnen des Sprites und zum Verhalten beim Aktualisieren definieren.
- ❷ Diese Funktion wird aufgerufen, wenn die Funktion `Ball()` zum Erzeugen eines neuen Balls aufgerufen wird.
- ❸ Diese Variablen werden als Teil jeder *Instanz* von `Ball` definiert. Jeder Ball hat seinen eigenen Satz von Variablen.
- ❹ Die Funktion `update()` wird für jeden Frame aufgerufen. Die gespeicherte Position wird abhängig von der Richtung und der Geschwindigkeit des Balls neu berechnet und dann wird geprüft, ob sich der Ball einem Rand nähert. In diesem Fall wird die Richtung für die entsprechende Achse umgekehrt.
- ❺ Es werden zwei Bälle mit unterschiedlicher Position, Richtung und Geschwindigkeit erzeugt.
- ❻ Der Ball wird mit einer neuen Position versehen – basierend auf der letzten Position, Richtung und Geschwindigkeit.
- ❼ Zeichne den Ball an seiner aktuellen Position.

Klänge abspielen

In Pygame können Sie mit dem Modul `pygame.mixer` Audiodateien laden und abspielen. Alternativ gibt es auch das Modul `pygame.midi`, um MIDI-Events an andere Software auf dem Pi oder an MIDI-Hardware zu schicken, die am USB-Port angeschlossen ist. Das folgende Beispiel gibt eine WAV-Audiodatei wieder (die Sie vom Internet Archive (<http://archive.org/details/WilhelmScreamSample>) herunterladen können).

```

import pygame.mixer
from time import sleep

pygame.mixer.init(48000, -16, 1, 1024)

sound = pygame.mixer.Sound("WilhelmScream.wav")
channel1A = pygame.mixer.Channel(1)
channel1A.play(sound)
sleep(2.0)

```

Eine Audiodatei (im WAV-Format) wird geladen und mit einem Kanal verbunden. Der Mixer gibt jeden Kanal in einem eigenen Prozess wieder, daher können mehrere Klänge gleichzeitig abgespielt werden. Das `sleep()` am Ende ist notwendig, damit der Hauptprozess nicht endet, bevor die Klänge zu Ende gespielt wurden.

Der Ansatz über den Mixer zum Wiedergeben von Audiodateien wird in Kapitel 8 im Simple Soundboard-Beispiel noch vertieft. Die MIDI-Fähigkeiten von Pygame bieten verblüffende Möglichkeiten, so können Sie zum Beispiel recht einfach einen eigenen MIDI-Controller erstellen. Aktuell ist die Anzahl der Software-Synthesizer, die auf dem Raspbian laufen, noch sehr übersichtlich und die analoge Audioausgabe ist nicht so gut wie die HDMI-Audioausgabe. Das wird aber besser werden, und Sie können auch immer ein externes USB-MIDI-Gerät einsetzen. Um Informationen über MIDI-Geräte zu finden, die an Ihrem Raspberry Pi angeschlossen sind, nutzen Sie:

```
import pygame
import pygame.midi ❶

pygame.init()
pygame.midi.init() ❷
for id in range(pygame.midi.get_count()): ❸
    print pygame.midi.get_device_info(id) ❹
```

- ❶ Das midi-Modul wird nicht vom Standard-Modul pygame importiert.
- ❷ Es muss auch eigenständig initialisiert werden.
- ❸ Die Funktion `get_count()` gibt die Anzahl an MIDI-fähigen Geräten zurück, die mit dem Pi entweder über USB, als Software-Synthesizer oder über andere virtuelle MIDI-Geräte verbunden sind.
- ❹ Gibt Informationen zum Gerät aus.

Stecken Sie ein externes MIDI-Keyboard an, erhalten Sie so etwas wie die folgende Ausgabe:

```
('ALSA', 'Midi Through Port-0', 0, 1, 0)
('ALSA', 'Midi Through Port-0', 1, 0, 0)
('ALSA', 'USB Uno MIDI Interface MIDI 1', 0, 1, 0)
('ALSA', 'USB Uno MIDI Interface MIDI 1', 1, 0, 0)
```

Der erste String zeigt, dass Sie die Advanced Linux Sound Architecture (ALSA) nutzen, der zweite beschreibt jeden einzelnen MIDI-Port. Die letzten drei Zahlen geben an, ob es sich beim Port um ein Input- oder Output-Gerät handelt und ob es geöffnet ist oder nicht. Sie können das Beispiel als vier verschiedene MIDI-Ports von 0 bis 3 betrachten:

Midi Through Port-0 (0, 1, 0)

Port 0, ein Output-Port, mit dem Software-Synthesizer angesprochen werden, die eventuell auf Ihrem Pi laufen.

Midi Through Port-0 (1, 0, 0)

Port 1, ein Input-Port, der MIDI-Befehle von Software-Controllern annimmt, die eventuell auf Ihrem Pi laufen.

USB Uno MIDI Interface MIDI 1 (0, 1, 0)

Port 2, ein Output-Port zu einer externen USB-MIDI-Schnittstelle, die mit einem Keyboard verbunden ist.

USB Uno MIDI Interface MIDI 1 (1, 0, 0)

Port 3, ein Input-Port von einer externen USB-MIDI-Schnittstelle, die mit einem Keyboard verbunden ist.

So können Sie nun ein externes Keyboard an einen USB-Anschluss stecken und es über Pygame steuern, zum Beispiel:

```
import pygame
import pygame.midi
from time import sleep

instrument = 0 ❶
note = 74
volume = 127

pygame.init()
pygame.midi.init()

port = 2 ❷
midiOutput = pygame.midi.Output(port, 0)
midiOutput.set_instrument(instrument)
for note in range(0, 127):
    midiOutput.note_on(note, volume) ❸
    sleep(.25)
    midiOutput.note_off(note, volume)
del midiOutput
pygame.midi.quit()
```

- ❶ Dies sind MIDI-Werte, meist im Bereich von 0 bis 127.
- ❷ Öffne Port 2, der mit dem USB-MIDI-Keyboard verbundene Output-Port.
- ❸ Verschicke eine Note, warte, dann schalte die Note wieder ab.

Es ist offensichtlich, dass der Raspberry Pi großes Potenzial im Musikumfeld hat.

Videos wiedergeben

Pygame kann auch dazu genutzt werden, Videos mit dem Modul `pygame.movie` wiederzugeben. Die Videodateien müssen dazu im MPEG1-Format kodiert sein. Haben Sie ein Video in einem anderen Format, probieren Sie das `ffmpeg`-Tool aus, um es in das gewünschte Format zu konvertieren (zuerst müssen Sie `sudo apt-get install ffmpeg` ausführen). Um ein Video abzuspielen, erstellen Sie einfach ein neues `Movie`-Objekt und rufen dann die Funktion `play()` auf:

```

import pygame
from time import sleep
pygame.init()
screen = pygame.display.set_mode((320,240))
movie = pygame.movie.Movie("foo.mpg")
movie.play()
while True:
    if not(movie.get_busy()):
        print("rewind")
        movie.rewind()
        movie.play()
    if pygame.QUIT in [e.type for e in pygame.event.get()]:
        break

```

Besitzt das Video eine Audiospur, müssen Sie den Audiomixer von Pygame schließen, bevor Sie den Film abspielen. Damit er sich schließt, fügen Sie vor dem Abspielen diese Codezeile ein:

```
pygame.mixer.quit()
```



Noch mehr Beispiele

Es gibt ein ganzes Pygame-Modul mit umfangreicheren Beispielsprogrammen: `pygame.examples`. Sie finden den Quellcode für diese Beispiele im Verzeichnis `/usr/share/pyshared/pygame/examples`.

Weitere Informationen

Offizielle Pygame-Dokumentation (<http://www.pygame.org/docs/>)

Die offizielle Dokumentation ist an manchen Stellen noch ein wenig mager geraten, aber hoffentlich werden Sie mit ihr klarkommen, wenn Sie dieses Kapitel gelesen haben.

Making Games with Python & Pygame (<http://inventwithpython.com/>)

and Invent Your Own Computer Games with Python

(<http://inventwithpython.com/>) von Al Sweigart

Zwei Bücher unter der Creative Commons-Lizenz. Die Spiele, die in diesen Büchern entwickelt wurden, werden mit dem Raspberry Pi ausgeliefert.

5/Scratch auf dem Pi

Scratch wurde von der „MIT Media Lab’s Life-long Kindergarten“-Gruppe entwickelt, um einen neuen Weg beim Programmieren lernen für junge Leute gehen zu können. Programme werden aus farbenfrohen Blöcken zusammengestellt, von denen jeder eine Operation durchführt. Mit diesen Blöcken werden die Syntax-Probleme umgangen, die viele Neueinsteiger bei textbasierten Programmiersprachen ins Straucheln bringen.

Zu sagen, dass Scratch keine mächtige Programmiersprache ist, trifft nicht den Punkt: Es handelt sich um eine angenehme Umgebung, um Dinge schnell zu erstellen und umzusetzen. Ein junger Programmierer sieht, wie die Blöcke bei der Ausführung hervorgehoben werden, zudem lassen sie sich zur Laufzeit mit direkten Auswirkungen ändern.

Wie Sie sehen werden, drehen sich alle Scratch-Programme um das Verändern von Objekten oder Sprites auf einer *Bühne*. Es gibt eine große Community mit Scratch-Anwendern, und die Fähigkeit, Sprites und Code mit anderen auszutauschen, ist direkt in die Plattform integriert.

Hallo Scratch

Um zu zeigen, wie einfach man in die Programmierung mit Scratch einsteigt, werden wir mit einem sehr simplen „Katze in der Kiste“-Programm beginnen. Öffnen Sie Scratch, sehen Sie ein einzelnes Fenster mit einer Reihe von Bereichen. Die Aufgabe jedes Bereichs ist in Abbildung 5-1 beschrieben. Sie können Scratch durch einen Doppelklick auf sein Desktop-Symbol oder durch das Anklicken des Desktop-Menüs unten links und die Wahl von Education→Scratch starten.

Jedes Scratch-Programm enthält *Sprites*, die auf der *Bühne* miteinander interagieren. Sprites werden durch *Skripten* gesteuert, die aus einer Reihe von Blöcken bestehen, die wiederum aus der *Block-Palette* gezogen werden. Wenn Sie ein neues Dokument öffnen, erhalten Sie das Standard-Sprite mit der Scratch-Katze. Wollen Sie die Katze nicht haben, können Sie

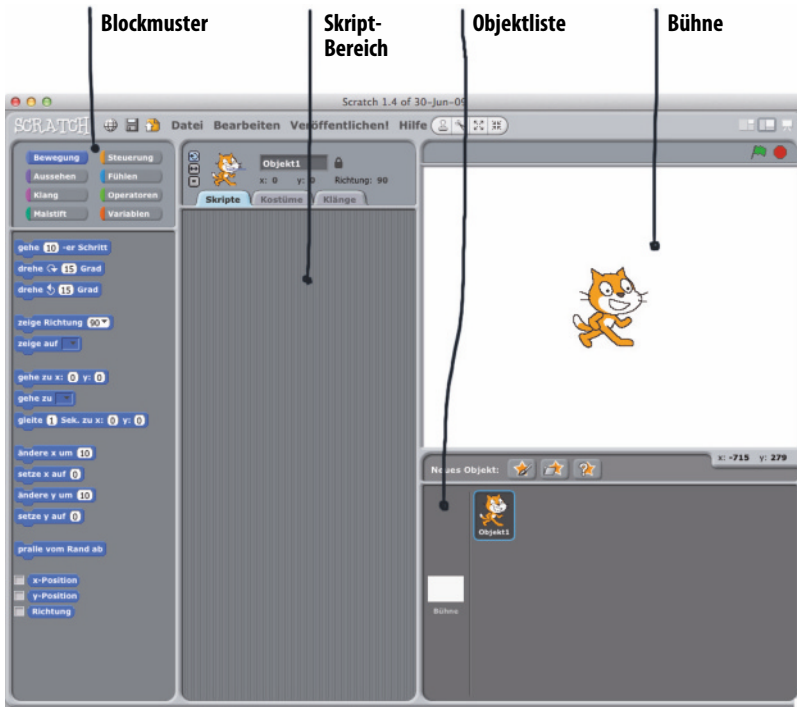


Abbildung 5-1. Die Scratch-Umgebung. Alles befindet sich in einem Fenster, das Bereiche für die Block-Palette, Skripten und Kostüme, die Sprite-Liste und die Bühne enthält.

sie löschen (durch einen Klick mit der rechten Maustaste auf den Sprite in der *Sprite-Liste*) und Ihr eigenes Objekt zeichnen oder sich ein zufälliges Überraschungs-Sprite schnappen, das jemand anderes aus der Scratch-Community gezeichnet hat.

Der *Skripte-Bereich* zeigt das Skript, das zum aktuell ausgewählten Sprite gehört. Wenn Sie auf die Katze in der Sprite-Liste klicken, erhalten Sie einen leeren Skript-Bereich für das Sprite. Hier können Sie an einer beliebigen Stelle ein Skript erstellen. Erzeugen Sie ein anderes Sprite, müssen Sie es anklicken, um sein Skript anzuzeigen.

Skripten entstehen, indem Sie Blöcke aus der Palette in den Skript-Bereich ziehen. Viele Blöcke können andere Blöcke enthalten. Wie sie zusammengehören, sehen Sie an ihrer Form. Es gibt in Scratch drei Arten von Blöcken (siehe Abbildung 5-2):



Abbildung 5-2. Es gibt in Scratch drei Arten von Blöcken: Hut-Blöcke (links), stapelbare Blöcke (Mitte) und Reporter-Blöcke (rechts).

Hut-Blöcke

Der Block *Wenn grüne Flagge angeklickt* ist ein Beispiel für einen Hut-Block, der einen Stapel mit Blöcken anführt und auf ein Ereignis wartet.

Stapelbare Blöcke

Blöcke mit einer Ausbuchtung oben oder einem Zapfen unten lassen sich mit anderen stapelbaren Blöcken verbinden. Die Blöcke werden von oben nach unten ausgeführt.

Reporter-Blöcke

Reporter-Blöcke haben abgerundete oder spitze Ecken und passen in den Eingabebereich anderer Blöcke. Reporter-Blöcke können Variablen sein oder Informationen bereitstellen, wie zum Beispiel die Mauskoordinaten oder bestimmte Bedingungen.

Beginnen Sie ein neues Skript durch Auswahl der Block-Palette „Steuerung“ und ziehen Sie diesen Block in den Skript-Bereich:



Als Nächstes setzen Sie den stapelbaren Block auf *wiederhole fortlaufend*, der alle in ihm enthaltenen Befehle so lange ausführt, bis das Skript gestoppt wird:



Üben Sie ein bisschen den Umgang mit den Blöcken. Ziehen Sie im Skript-Bereich an einem Block, nehmen Sie auch immer alle folgenden Blöcke mit. Um einen einzelnen Block herauszunehmen, müssen Sie oft eine ganze Gruppe von Blöcken wegziehen und dann das eigentliche Kindelement neh-

men. Sie können auch mit der rechten Maustaste auf einen Block klicken, um ihn (und seine Kinder) zu duplizieren und Hilfe zu erhalten.

Als Nächstes wechseln Sie zur Bewegungs-Palette und ziehen einen *drehe*-Block direkt in die Schleife.



Jetzt können Sie schon auf die grüne Flagge oben rechts über der Bühne klicken und damit das Skript ausführen. Einer der tollen Aspekte an Scratch ist, dass Sie zwischendurch Änderungen vornehmen können und direkt die Auswirkungen davon sehen. Damit lässt sich Ihr Programm einfach debuggen.



Wenn Sie auf die grüne Flagge klicken, wird eine Nachricht an alle Skripten im Projekt geschickt, mit der Ausführung zu beginnen (das Stopp-Zeichen schickt ein Stopp-Signal).

Wenn Sie sich den *drehe*-Block genauer anschauen, sehen Sie, dass der Drehwinkel in einem abgerundeten Feld steht. Sie können den Winkel direkt verändern oder ihn durch einen anderen Block der gleichen Form ersetzen. Durch ein paar Änderungen können Sie so dafür sorgen, dass sich das Sprite zufällig bewegt.

Als Erstes ersetzen Sie den Standardwert durch den Block *Zufallszahl* aus der Operatoren-Palette. Nachdem Sie den Block eingefügt haben, ändern Sie die Werte, so dass eine Zufallszahl zwischen -10 und 10 gewählt wird.



Nun fügen Sie einen *gehe*-Block aus der Bewegungs-Palette ein, wodurch das Sprite um eine gewisse Anzahl Pixel in die aktuelle Blickrichtung verschoben wird. Die Richtung wird in dem Info-Bereich über den Skripten als

Zahl und als blaue Linie angezeigt. Sobald Sie den Block absetzen, wird das Sprite mit einer Zufallsbewegung beginnen.



Um das Sprite schließlich auf der Bühne zu halten, fügen Sie einen Block *pralle vom Rand ab* aus der Bewegungs-Palette ein:



Das ist alles, was Sie zum Schreiben eines Scratch-Skripts benötigen!

Die Bühne

Die Bühne ist der Bereich oben rechts, in dem alle Ihre Sprites ihre Aktionen und Interaktionen durchführen. Wie Sprites kann auch die Bühne Skripten haben, die ihr Erscheinungsbild oder Verhalten beeinflussen. Sie können auf dem *Hintergründe*-Tab des Skript-Bereichs auch den Hintergrund der Bühne bemalen.

Das Koordinatensystem ist in Scratch ein bisschen anders als bei anderen Multimedia-Umgebungen wie Pygame. Es folgt mehr dem klassischen mathematischen Modell, wo der Ursprung (0, 0) tatsächlich auch die Mitte der Bühne ist. Der sichtbare Teil der Bühne reicht von (-240, 180) bis (240, -180), wie Sie in Abbildung 5-3 sehen. Sprites können auch von der Bühne wandern, wenn Sie sie lassen; wählen Sie ein Sprite aus, sehen Sie dessen aktuelle Position oberhalb des Skript-Bereichs und die Position des Cursors unterhalb der Bühne.

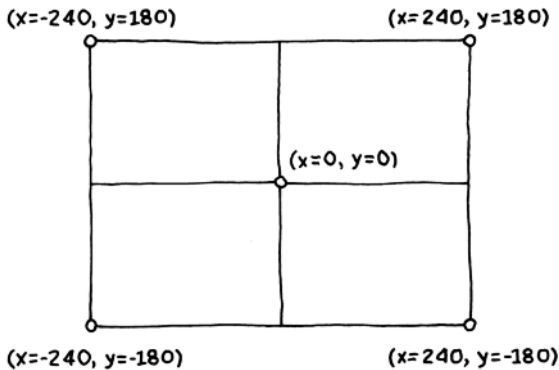
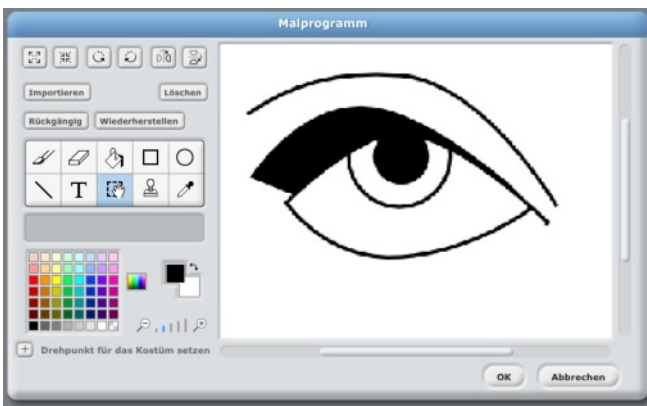


Abbildung 5-3. Das Koordinatensystem von Scratch mit dem Ursprung im Mittelpunkt der Bühne

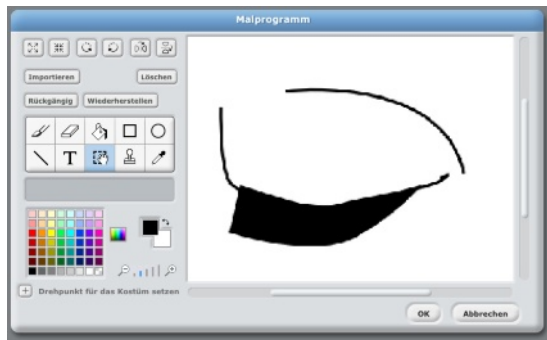
Noch zwei Dinge über Sprites

Es gibt zwei zusätzliche Tabs im Skript-Bereich, die Sie nutzen werden: *Kostüme* und *Klänge*.

Kostüme sind eine Sammlung von Bildern, in die sich das Sprite verwandeln kann. Kostüme können genutzt werden, um Animationen in einer Schleife zu erzeugen oder unterschiedliche Zustände des Sprite darzustellen (zum Beispiel ein explodierendes Raumschiff). Stellen Sie sich vor, Sie erstellen ein neues Sprite als offenes Auge. Dann können wir es mit Kostümen blinken lassen. Wählen Sie *Neues Objekt malen* (das Symbol rechts vom Text „Neues Objekt“ im Statusbereich) und zeichnen Sie etwas wie das folgende Bild. Klicken Sie dann auf OK:



Wählen Sie den *Kostüme*-Tab, dort sehen Sie das gerade gezeichnete Bild. Ändern Sie den Namen des Kostüms in *offen*. Klicken Sie auf den *Kopieren*-Button, um ein neues Kostüm zu erstellen. Klicken Sie dann auf *Bearbeiten*, um die Pupille zu löschen und ein geschlossenes Augenlid zu zeichnen. Dann klicken Sie auf OK und ändern den Namen dieses Kostüms in *geschlossen*:



Als Nächstes wählen Sie den *Skripte*-Tab und ziehen diese Blöcke zusammen, um ein Skript zu erstellen, das jede Sekunde das Auge blinzeln lässt:



Schließlich können Sie über den *Klänge*-Tab ein Sprite noch mit Klangeffekten versehen. Drücken Sie einfach auf *Aufnehmen*, um zum eingebauten Aufzeichnungstool zu gelangen. Sie benötigen eine externe USB-Soundkarte oder ein Mikrofon, um Klänge aufzuzeichnen. Wenn Sie dem Sondereffekt einen Namen geben, können Sie ihn über die Blöcke *spiele Klang* oder *spiele Klang ganz* wiedergeben.

Ein größeres Beispiel: Astral Trespassers

Im Spiel Astral Trespassers muss der Spieler auf eintreffende Alien-Raum-schiffe schießen. Das Spiel ist vorbei, wenn ein Alien die Verteidigungswaffe des Spielers trifft. Dieses einfache Spiel wird zeigen, wie die einzelnen Elemente zusammenwirken.

Wählen Sie zuerst Datei→Neu, dann speichern Sie das neue Spiel über Datei→Speichern als. Als Nächstes müssen Sie ein paar Sprites erstellen. Klicken Sie mit gedrückter Strg-Taste oder mit der rechten Maustaste auf die Katze in der Sprite-Liste und wählen Sie im Menü *Löschen*. Dann erstellen Sie neue Sprites, indem Sie sie selbst zeichnen oder aus einer Bilddatei laden. Ich habe die fünf Sprites in Abbildung 5-4 gezeichnet und dann als PNG gescannt.

Wenn Sie die Sprites zur Bühne hinzufügen, werden sie vermutlich nicht die Größe haben, die Sie sich vorstellen. Nutzen Sie daher die Tools *Objekt vergrößern* und *Objekt schrumpfen* direkt oberhalb der Bühne und klicken Sie das Sprite an, um es zu vergrößern oder zu verkleinern. Sie werden die Sprites auch noch mit den Namen aus Abbildung 5-4 versehen wollen.

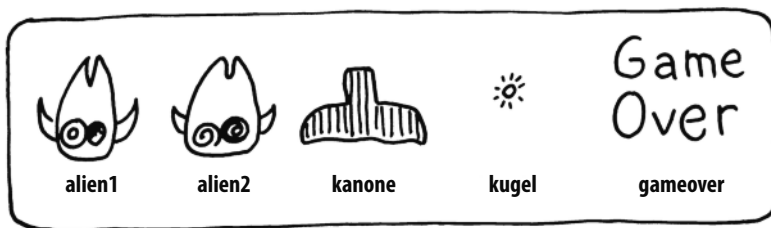


Abbildung 5-4. Die fünf Sprites für Astral Trespassers.

Die Aliens und die Kanone besitzen jeweils zwei Zustände – normal und explodiert. Wir zeigen das, indem wir für jedem Zustand ein Kostüm erzeugen (siehe Abbildung 5-5).

Jedes Sprite wird sein eigenes Skript erhalten, über das sein Verhalten im Spiel definiert ist. Um den Überblick besser behalten zu können, wird jede Aktion ein komplett eigenes Skript sein, das parallel zu den anderen Skripten läuft.



Anhang B enthält die Skripten für jedes Sprite und zeigt auch, wie Sie mehrfach wiederholende Kugeln nutzen können.



Abbildung 5-5. Erstellen Sie zwei Kostüme für jedes Sprite – eines für den normalen Status und eines für eine Explosion. Benennen Sie die Kostüme wie in diesem Bild angegeben.

Beginnen wir mit dem ersten Alien. Wählen Sie das Sprite aus und ziehen Sie folgende Blöcke in den Skript-Bereich:



Dieser Block bewegt das Sprite an seine Ausgangsposition und stellt sicher, dass es sichtbar und nicht explodiert ist.

Als Nächstes ziehen Sie die folgenden Blöcke an einer anderen Stelle im Skript-Bereich zusammen:



Wenn Sie auf die grüne Flagge drücken, sehen Sie, dass es mehr oder weniger das „Katze in der Kiste“-Beispiel von weiter oben im Kapitel ist. Da es hier drei Sprites gibt, müssen Sie sich überlegen, was passieren soll, wenn die Sprites miteinander kollidieren. Stoßen zwei Aliens zusammen, gibt es die Möglichkeit, sie um 90° zu drehen, damit sie sich nicht überlappen:



Trifft die Kugel das Alien, soll es explodieren:



Die Explosion erreicht man, indem man zum *Explosion*-Kostüm wechselt, für kurze Zeit einen Spezialeffekt anwendet und das Sprite dann verbirgt, womit es von der Bühne verschwindet. Nach ein paar Sekunden wird das Alien an einer zufälligen Position im oberen Bildschirmbereich wieder auftauchen.

Für die Aliens ist das alles an Skripten, was wir benötigen. Haben Sie mehrere Sprites, die das Gleiche (oder zumindest etwas Ähnliches) tun sollen, können Sie Code zwischen den Sprites kopieren, indem Sie einen Block mit der rechten Maustaste anklicken und dann *Duplizieren* wählen. Ziehen Sie den duplizierten Block zum zweiten Alien und wiederholen Sie dies für alle Block-Abschnitte. Wählen Sie nun das zweite Alien und ändern Sie die Startposition und den Kollisionstest:



Als Nächstes wählen Sie das Kanonen-Sprite aus. Bewegen Sie es zuerst an seine Position und stellen Sie sicher, dass es sichtbar und nicht explodiert ist:



Die Kanone kann sich nur nach links oder rechts bewegen, was durch die Cursortasten gesteuert wird. So sieht der Code aus, mit dem auf die Tastendrücke reagiert wird:



Prüfen Sie eine Kollision mit einem der Aliens über den *wird berührt?*-Test aus der Fühlen-Palette. Wenn das der Fall ist, lassen Sie die Kanone so explodieren wie die Aliens. Nach der Explosion verschickt die Kanone eine Nachricht an alle anderen Sprites, dass das Spiel vorbei ist. Diese Nachricht wird vom „Game Over“-Sprite aufgenommen, das dann alle Skripten stoppt.



Jetzt zum Kugel-Sprite. Fügen Sie diese Codeblöcke ein. Die Kugel wird verborgen, bis die Leertaste gedrückt wurde:



Mit jedem Druck auf die Leertaste wird die Kugel zur aktuellen Position der Kanone verschoben, sichtbar gemacht und dann die Bühne emporgeschickt, bis sie den oberen Rand erreicht.

Hier die Blöcke, die sich um die Reaktion auf die Leertaste kümmern:



Das letzte Sprite ist die Game-Over-Nachricht. Seine Aufgabe ist einfach: Es verbirgt sich, bis es die *Game Over*-Nachricht erhält; dann erscheint es und stoppt alle Skripten:



Das ist das Ende des Spiels. Alle Blöcke zusammen sehen Sie in Anhang B.

Scratch und die reale Welt

Wenn Sie sich auf der Fühlen-Palette umschauen, werden Sie zwei interessante Blöcke bemerken: *Sensor* und *Wert von Sensor*. Diese Blöcke werden genutzt, um mit einem externen Sensor-Board namens PicoBoard zu interagieren (siehe Abbildung 5-6). Das PicoBoard besitzt einen Mikrocontroller, der die Sensoren ausliest und die Werte über eine USB-Verbindung an Scratch übermittelt.

Das PicoBoard bietet einen Taster, einen Schieberegler, einen Geräuschsensord und Anschlüsse für vier beliebige Analogeingänge. Mit speziellen Krokodilklemmen können Sie analoge Werte von vielen Quellen auslesen.

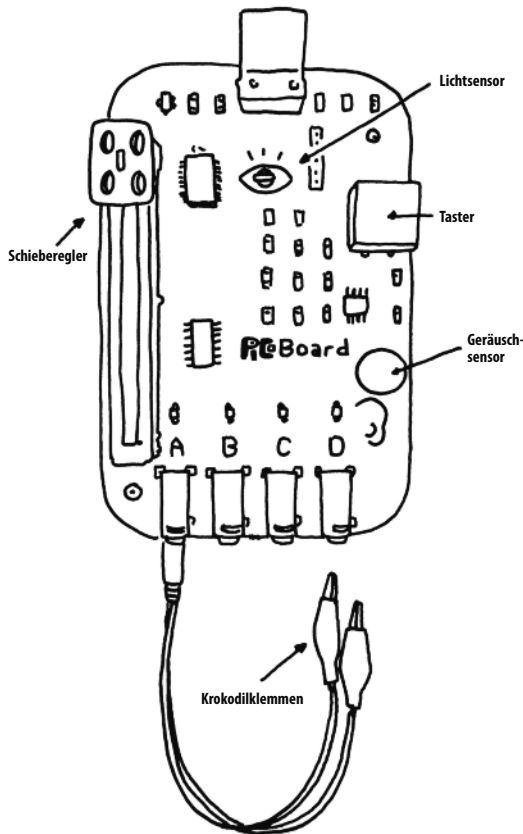


Abbildung 5-6. Das PicoBoard ist ein Sensor-Board, das für eine Zusammenarbeit mit Scratch entworfen wurde.

Das PicoBoard verschickt seine Sensor-Werte an Scratch über sein eigenes PicoBoard-Protokoll. Das S4A (<http://seaside.citilab.eu/scratch/arduino>)-Projekt (Scratch for Arduino) hat das gleiche Protokoll für Arduino implementiert, so dass Sie sich auch so mit der realen Welt verbinden können. Für S4A benötigen Sie eine spezielle Version von Scratch, die auf der Projektseite verfügbar ist.

Ihre Programme veröffentlichen

Einer der wirklich interessanten Aspekte von Scratch ist, dass es eine Community gibt, die direkt in das Programm mit eingebaut ist. Neben dem *Überraschungsobjekt*-Tool (mit dem zufällige Sprites von anderen Scratch-Anwendern geholt werden) gibt es auch ein Veröffentlichen-Feature, mit dem Ihre Programme zusammengepackt und auf die Scratch-Projektseite (<http://scratch.mit.edu/>) des MIT hochgeladen werden. Bei letzter Zählung gab es über 1,2 Millionen Scratch-Anwender, die über 2,8 Millionen Programme bereitgestellt haben.

Ein Grund für die vielen veröffentlichten Projekte ist, dass man es so einfach tun kann. Haben Sie einen Account unter scratch.mit.edu (<http://scratch.mit.edu/>) eingerichtet, wählen Sie einfach *Dieses Projekt im Internet veröffentlichen* im Veröffentlichen!-Menü. Es gibt eine Grenze von 10 MB, daher müssen Sie eventuell erst ein paar Ihrer Bilder oder Klänge komprimieren (siehe dazu die Optionen im Bearbeiten-Menü). Auf der Projektseite von Scratch werden Sie viele Beispiele dafür finden, was mit dieser Umgebung machbar ist.

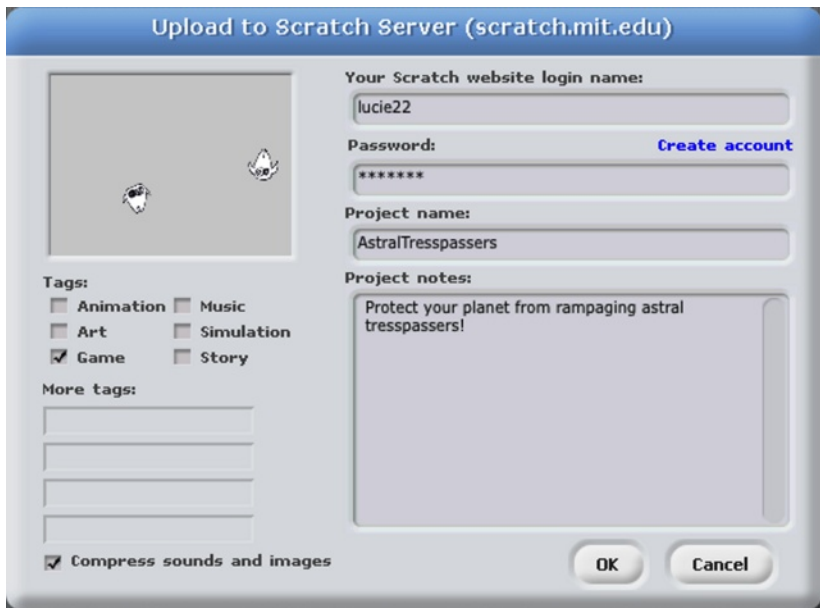


Abbildung 5-7. Haben Sie etwas Nettes geschaffen, veröffentlichen Sie es auf der Projektseite des MIT mit Hilfe des eingebauten Tools. Leider nutzt die Website Flash, daher werden Sie sie nicht direkt vom Raspberry Pi aufrufen können.

Weitere Informationen

Das zentrale Scratch-Wiki

(http://wiki.scratch.mit.edu/wiki/Scratch_Wiki)

Das Referenzmaterial zu Scratch vom MIT.

Die MIT-Scratch-Site (<http://scratch.mit.edu/>)

Dies ist die Scratch-Community-Site: Millionen von Projekten und registrierten Anwendern.

Das deutschsprachige Scratch-Wiki (<http://scratch-dach.info/>)

Viele Informationen rund um Scratch auf Deutsch.

6/Arduino und der Pi

Wie Sie in den nächsten Kapiteln sehen werden, können Sie die GPIO-Pins des Raspberry Pi nutzen, um Sensoren oder Dinge wie blinkende LEDs und Motoren anzuschließen. Haben Sie Erfahrungen mit dem Arduino, einer Mikrocontroller-Entwicklungsplattform, können Sie ihn auch zusammen mit dem Raspberry Pi einsetzen.

Als der Raspberry Pi das erste Mal angekündigt wurde, haben sich viele Leute gefragt, ob er ein Arduino-Killer werden würde. Für etwa den gleichen Preis können Sie viel mehr Prozessorleistung erhalten — warum sollte man da den Arduino nutzen, wenn man einen Pi haben kann. Es zeigte sich, dass sich die beiden Plattformen eher ergänzen und der Raspberry Pi ein ausgezeichnete Host für den Arduino ist. Es gibt ein paar weitere Fälle, in denen Sie Arduino und Pi zusammen nutzen wollen:

- Es gibt viele Bibliotheken und veröffentlichte Beispiele für den Arduino.
- Sie haben schon ein gut funktionierendes Arduino-Projekt und wollen es mit mehr Prozessorleistung ausstatten. Vielleicht haben Sie zum Beispiel einen MIDI-Controller, der mit einem Synthesizer verbunden ist, und wollen jetzt die Klänge direkt auf dem Pi erstellen lassen.
- Bei der Arbeit mit 5-V-Logikwerten. Der Pi arbeitet mit 3,3 V und seine Pins können nicht mit 5 V bedient werden.
- Sie experimentieren ein wenig außerhalb Ihres Wissensbereichs und haben Sorge, dass Sie einen Chip braten. Ich habe schon Schüler gesehen, die versuchten, Motoren direkt über einen Pin des Arduino zu steuern (versuchen Sie das nicht). Es war einfach, den zerstörten Mikrocontroller-Chip aus seinem Sockel zu nehmen und durch einen neuen zu ersetzen (was normalerweise weniger als 10 Euro kostet). Bei dem Raspberry Pi ist das nicht möglich.
- Bei einem Problem, das eine exakte Steuerung in Echtzeit erfordert, wie zum Beispiel für einen 3D-Drucker. Wie wir in Kapitel 3 gesehen haben, ist Raspbian kein Echtzeit-Betriebssystem, und die Programme können sich nicht unbedingt auf die gleiche Exaktheit bei den Instruktionen pro Zeiteinheit verlassen, wie bei einem Mikrocontroller.

Die Beispiele in diesem Abschnitt gehen davon aus, dass Sie zumindest die Grundlagen im Umgang mit dem Arduino-Board und der IDE kennen. Fehlen sie Ihnen, ist *Arduino für Einsteiger* (<http://www.oreilly.de/catalog/arduinoEinstger>) von Massimo Banzi ein sehr guter Ausgangspunkt. Die offiziellen Arduino-Tutorials (<http://arduino.cc/en/Tutorial/HomePage>) sind auch ziemlich gut und bieten viele Möglichkeiten, funktionierenden Code zu übernehmen.

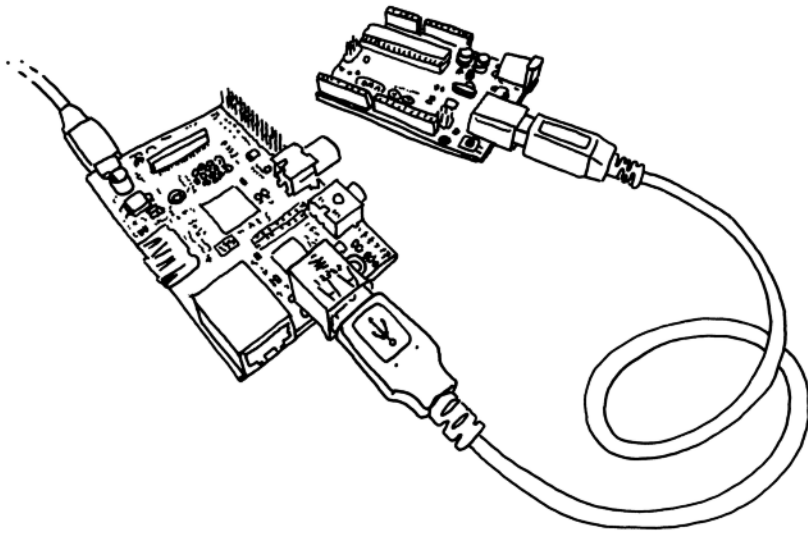


Abbildung 6-1. *Arduino und der Raspberry Pi sind beste Freunde.*

Arduino in Raspbian installieren

Um ein Arduino-Board zu programmieren, müssen Sie es über ein USB-Kabel mit einem Computer verbinden und dann ein Programm mit Hilfe der Arduino-IDE kompilieren und auf das Board flashen. Das geht mit einem beliebigen Rechner, Sie können aber auch Ihren Raspberry Pi als Host nutzen, um den Arduino zu programmieren.

Das Debuggen des Arduino geht mit dem Raspberry Pi schneller, dafür dauert das Kompilieren etwas länger als auf einem modernen Laptop oder Desktop-Rechner. Aber das ist nicht allzu schlimm und Sie werden feststellen, dass es nach dem allerersten Kompilieren auch wieder schneller geht, weil nur der Code neu kompiliert wird, der sich seit dem letzten Durchlauf geändert hat.

Zum Installieren der Arduino-IDE auf dem Raspberry Pi rufen Sie diese Befehle auf:

```
sudo apt-get update ❶  
sudo apt-get install arduino ❷
```

- ❶ Sicherstellen, dass die neueste Paketliste vorliegt
- ❷ Das Arduino-Paket herunterladen

Diese Anweisung installiert Java sowie eine Reihe weiterer Abhängigkeiten. Die Arduino-Umgebung erscheint im Abschnitt *Electronics* des Programmenüs (aber rufen Sie sie jetzt noch nicht auf).

Wenn Sie den Pi ohne Monitor laufen lassen, können Sie den Arduino einfach an einem der freien USB-Ports anschließen. Haben Sie keinen freien USB-Port mehr, können Sie auch einen an Ihrer Tastatur verwenden — sonst benötigen Sie einen USB-Hub. Die USB-Verbindung sollte genug Strom liefern können, um den Arduino zu versorgen, aber es ist auch nicht verkehrt, ihn direkt mit Strom zu beliefern.



Beachten Sie, dass Sie das USB-Kabel zum Arduino einstecken müssen, nachdem der Raspberry Pi gebootet wurden. Lassen Sie es während des Bootvorgangs stecken, kann es passieren, dass der Pi hängen bleibt, weil er alle Geräte am USB-Bus ausprobiert.

Wenn Sie die Arduino-IDE aufrufen, sucht sie alle USB-Geräte ab und erstellt eine Liste im Menü Tools→Serial Port. Um auf den seriellen Port zugreifen zu können, müssen Sie sicherstellen, dass der Benutzer *pi* dazu die Berechtigungen hat. Das erreichen Sie, indem Sie ihn den Gruppen *tty* und *dialout* hinzufügen. Dazu rufen Sie vor dem ersten Start der Arduino-IDE folgende Befehle auf:

```
sudo usermod❶ -a -G❷ tty pi  
sudo usermod -a -G dialout pi
```

- ❶ `usermod` ist ein Linux-Programm, um die Benutzer zu verwalten
- ❷ `-a -G` trägt den Benutzer (*pi*) in die angegebene Gruppe (*tty*, danach *dialout*) ein

Jetzt können Sie den Arduino starten. Klicken Sie auf Tools→Serial Port und wählen Sie den seriellen Port aus (sehr wahrscheinlich `/dev/ttyACM0`), dann klicken Sie auf Tools→Board und wählen die Art des Arduino Boards (zum Beispiel *Uno*). Wählen Sie File→Examples→01.Basics→Blink, um einen einfachen Beispiel-Sketch zu laden. Klicken Sie dann auf den Upload-Button in der Toolbar oder wählen Sie File→Upload, um den Sketch hochzuladen. Danach sollte die LED auf dem Arduino zu blinken beginnen.

Den seriellen Port herausfinden

Wenn aus irgendeinem Grund `/dev/ttyACM0` nicht funktioniert, müssen Sie ein bisschen nachforschen. Um herauszufinden, an welchem USB-seriellen Port der Arduino angesteckt ist, ohne das Menü durchzuprobieren, geben Sie folgenden Befehl an der Befehlszeile ein, wenn der Arduino noch nicht angeschlossen ist:

```
ls /dev/tty*
```

Stecken Sie den Arduino an und rufen Sie den Befehl erneut auf. Sie sollten dann sehen, was sich geändert hat. Auf meinem Raspberry Pi war zuerst `/dev/ttyAMA0` aufgeführt (der USB-Hub auf dem Board). Nach dem Einstecken des Arduino erschien zusätzlich noch `/dev/ttyACM0`.

Die Arbeit mit der IDE verbessern

Nach dem Einrichten der IDE ist Ihnen vielleicht aufgefallen, dass die Qualität der Standardschrift des Arduino-Editors nicht so toll ist. Sie können sie verbessern, indem Sie die Open-Source-Schriftart Inconsolata herunterladen. Zum Installieren geben Sie ein:

```
sudo apt-get install ttf-inconsolata
```

Dann bearbeiten Sie die Voreinstellungs-Datei für Arduino:

```
nano ~/.arduino/preferences.txt
```

und ändern die folgenden Zeilen:

```
editor.font=Inconsolata,medium,14  
editor.antialias=true
```

Nach dem Neustart der IDE wird der Editor die neue Schriftart nutzen.

Serielle Kommunikation

Um zwischen dem Raspberry Pi und dem Arduino über eine serielle Verbindung kommunizieren zu können, verwenden Sie auf Arduino-Seite die mitgelieferte Bibliothek *Serial* und das Python-Modul `pySerial` (<http://pyserial.sourceforge.net/>) für den Raspberry Pi. Zum Installieren des seriellen Moduls geben Sie ein:

```
sudo apt-get install python-serial python3-serial
```

Öffnen Sie die Arduino-IDE und laden Sie diesen Code auf den Arduino hoch:

```

void setup() {
    Serial.begin(9600);
}

void loop() {
    for (byte n = 0; n < 255; n++) {
        Serial.write(n);
        delay(50);
    }
}

```

Dieser Code zählt eine Variable hoch und schickt jede Zahl über die serielle Verbindung.



Beachten Sie, dass beim Arduino `Serial.write()` eine echte Zahl verschickt; der String "123" statt der Zahl 123 wird über `Serial.print()` übertragen.

Als Nächstes müssen Sie wissen, über welchen USB-seriellen Port der Arduino angeschlossen ist (siehe „Den seriellen Port herausfinden“ (S. 78)). Hier ist das Python-Skript dazu. Ist der Port nicht `/dev/ttyACM0`, ändern Sie den Wert von `port`. (In Kapitel 3 finden Sie mehr über Python.) Sichern Sie die Datei als `SerialEcho.py` und führen Sie sie mit `python SerialEcho.py` aus:

```

import serial

port = "/dev/ttyACM0"
serialFromArduino = serial.Serial(port,9600) ❶
serialFromArduino.flushInput() ❷
while True:
    if (serialFromArduino.inWaiting() > 0):
        input = serialFromArduino.read(1) ❸
        print(ord(input)) ❹

```

- ❶ Öffnet den seriellen Port, an dem der Arduino angeschlossen ist.
- ❷ Leert den Input-Puffer.
- ❸ Liest ein Byte aus dem seriellen Puffer.
- ❹ Wandelt das eingetroffene Byte mit `ord()` in eine „richtige“ Zahl um.



Sie werden nichts auf den Arduino hochladen können, wenn Python den seriellen Port geöffnet hat, daher achten Sie darauf, das Python-Programm mit Strg-C abzubrechen, bevor Sie den Sketch erneut hochladen. Bei einem Arduino Leonardo oder Arduino Micro ist das zwar möglich, aber dadurch wird die Verbindung zum Python-Skript unterbrochen, so dass Sie es sowieso neu starten müssen.

Der Arduino schickt eine Zahl an das Python-Skript, das die Zahl als String interpretiert. Die Variable `input` enthält das, was die Zeichentabelle an dieser Stelle in der ASCII-Tabelle (<http://de.wikipedia.org/wiki/ASCII>) definiert. Um besser zu verstehen, was da passiert, tauschen Sie die letzte Zeile des Python-Skripts einmal durch diese aus:

```
print(str(ord(input)) + " = ASCII-Zeichen " + input + ".")
```

Den seriellen Port als Argument mitgeben

Wollen Sie den Port als Argument an der Befehlszeile angeben, nutzen Sie das Modul `sys`, um das erste Argument auszulesen:

```
import serial, sys

if (len(sys.argv) != 2):
    print("Usage: python ReadSerial.py port")
    sys.exit()
port = sys.argv[1]
```

Danach können Sie das Programm wie folgt aufrufen: `python SerialEcho.py /dev/ttyACM0`.

Beim ersten, sehr einfachen Beispiel wurde nur ein einzelnes Byte verschickt. Das kann ausreichen, wenn Sie nur eine Folge von Ereignis-Codes vom Arduino verschicken wollen. Wenn zum Beispiel der linke Taster gedrückt wurde, wird eine 1 verschickt, beim rechten eine 2. Das reicht aber nur für 255 verschiedene Ereignisse – meist werden Sie beliebig große Zahlen oder Strings übertragen wollen. Wenn Sie analoge Sensoren auf dem Arduino auslesen, bewegen sich die Zahlen zum Beispiel im Bereich von 0 bis 1023.

Das Parsen beliebiger Zahlen, die Byte für Byte hereinkommen, kann in vielen Sprachen kniffliger sein, als Sie es sich vielleicht vorstellen. Aber der Weg, wie es Python und PySerial lösen, ist nahezu trivial. Als Beispiel können Sie den Arduino mit folgendem Code updaten, der von 0 bis 1023 zählt:

```
void setup() {
    Serial.begin(9600);
}

void loop() {
    for (int n = 0; n < 1024; n++)
        Serial.println(n, DEC);
    delay(50);
}
```

Der entscheidende Unterschied liegt hier im Befehl `println()`. Im vorigen Beispiel wurde die Funktion `Serial.write()` genutzt, um die reinen Zahlen über den seriellen Port auszugeben. Bei `println()` formatiert der Arduino die Zahl als Dezimalstring und verschickt die ASCII-Codes für den String. Statt also die Zahl 254 zu versenden, wird der String "254\r\n" übertragen. \r steht für einen Zeilenumbruch (Carriage Return) und \n für einen Zeilenvorschub (Newline; dies sind Konzepte, die aus der Schreibmaschinen-Ära in die Computerwelt übernommen wurden: Carriage Return bewegt den „Cursor“ an den Zeilenanfang, Newline beginnt eine neue Textzeile).

Auf der Python-Seite können Sie die Funktion `readline()` statt `read()` nutzen. Dann werden alle Zeichen bis (einschließlich) zum Zeilenumbruch und Zeilenvorschub gelesen. Python besitzt ein flexibles Set an Funktionen zum Konvertieren zwischen den verschiedenen Datentypen und Strings. Nutzen Sie hier einfach die Funktion `int()`, um den formatierten String in eine Integer-Zahl zu verwandeln:

```
import serial

port = "/dev/ttyACM0"
serialFromArduino = serial.Serial(port,9600)
serialFromArduino.flushInput()
while True:
    input = serialFromArduino.readline() ❶
    inputAsInteger = int(input) ❷
    print(inputAsInteger * 10) ❸
```

- ❶ Der gesamte String wird in die Variable `input` eingelesen.
- ❷ Die Variable wird in einen Integer-Wert gewandelt.
- ❸ Ausgabe des Werts, aber multipliziert mit 10, um zu zeigen, dass es sich wirklich um eine Zahl und keinen String handelt.

Es ist jetzt ganz einfach, dieses Beispiel so anzupassen, dass ein analoger Eingabewert ausgelesen und das Ergebnis verschickt wird – ändern Sie einfach die Schleife:

```
void setup() {
    Serial.begin(9600);
}

void loop() {
    int n = analogRead(A0);
    Serial.println(n, DEC);
    delay(100);
}
```

Wenn Sie jetzt noch das Python-Skript so ändern, dass Sie statt `inputAsInteger * 10` nur `inputAsInteger` ausgeben, sollten Sie Werte im 200er-Bereich erhalten, wenn nichts am analogen Pin 0 angeschlossen ist. Verbinden Sie den Pin mit GND, sollte der Wert 0 sein. Verbunden mit dem 3V3-Pin ergibt es einen Wert um 715 herum, beim 5-V-Pin um 1023.

Weitere Informationen

Haben Sie einmal die richtige Software gefunden, ist das Einrichten der Verbindung ziemlich einfach, und viele Projekte sehen in Bezug auf die Kommunikationsgrundlagen sehr ähnlich aus. Wie bei jeder Form der Kommunikation wird es meist erst dann schwieriger, wenn es über das „Hallo Welt“ hinausgeht. Sie müssen *Protokolle* aufsetzen (oder bestehende finden und implementieren), so dass sich beide Seiten verstehen. Die Details serieller Protokolle gehen über den Rahmen dieses Buches hinaus, aber es gibt viele tolle Beispiele, wie andere Leute die Probleme gelöst haben. Schauen Sie einmal im Abschnitt *Interfacing with Software* (<http://www.arduino.cc/playground/Main/InterfacingWithSoftware>) des Arduino Playground (<http://www.arduino.cc/playground/>) nach.

Firmata

Hans-Christoph Steiners *Firmata* (<http://arduino.cc/en/Reference/Firmata>) ist ein für viele Anwendungsfälle einsetzbares Protokoll, das es schon eine ganze Weile gibt. *Firmata* ist einfach und auch für Menschen lesbar. Es mag zwar nicht für alle Anwendungen perfekt sein, ist aber eine gute Ausgangsbasis.

MIDI

Dreht sich Ihr Projekt um Musik, sollten Sie sich den Einsatz von MIDI-Befehlen als serielles Protokoll überlegen. Da MIDI (mehr oder weniger) schon seriell ist, sollte es gut funktionieren.

Arduino-kompatible Raspberry Pi-Shields

Es gibt ein paar Boards (oder Shields) zu kaufen, die die GPIO-Pins des Raspberry Pi mit einem Arduino-kompatiblen Mikrocontroller verbinden. Das Shield A la Mode (<http://baldwisdom.com/projects/alamode/>) ist von WyoLum eine gute Lösung, die auch noch einige weitere Features wie eine Echtzeituhr enthält.

Über das Netzwerk kommunizieren

Schließlich können Sie die serielle Verbindung auch ganz verwerfen und mit dem Arduino über ein Netzwerk reden. Es gibt viele interessante Projekte, die das WebSocket (<http://www.websocket.org/>)-Protokoll zusammen mit der JavaScript-Plattform Node.js (<http://nodejs.org/>) nutzen. Das Noduino project (<http://semu.github.com/noduino/>) ist ein guter Ausgangspunkt für eine Entdeckungstour.

Die seriellen Pins auf dem Raspberry Pi nutzen

Der Raspberry Pi besitzt eine Reihe von Input- und Output-Pins, von denen zwei genutzt werden können, um serielle Daten unter Umgehen des USB-Ports zu verschicken und zu empfangen. Dazu müssen Sie sich erst mit den Informationen in Kapitel 8 befassen und sicherstellen, dass Sie einen Pegelumsetzer nutzen, um die 3,3-V-Pins des Raspberry Pi vor den 5-V-Pins des Arduino zu schützen.

Wollen Sie sich intensiver mit der Kommunikation von Geräten befassen, ist *Making Things Talk* (<http://www.oreilly.de/catalog/makingthingstalkger>) von Tom Igoe ein guter Ausgangspunkt.

7/Ein- und Ausgänge

Der Raspberry Pi ist zwar im Prinzip ein sehr günstiger Linux-Computer, aber es gibt ein paar Dinge, die ihn von Laptops und Desktop-Rechnern unterscheiden, die wir normalerweise zum Schreiben von E-Mails, Browsen im Web oder für die Textverarbeitung nutzen. Einer der Hauptunterschiede besteht darin, dass der Raspberry Pi ohne größere Probleme in Elektronik-Projekten eingesetzt werden kann, da er *General Purpose Input and Output*-Pins direkt auf dem Board besitzt (siehe Abbildung 7-1).

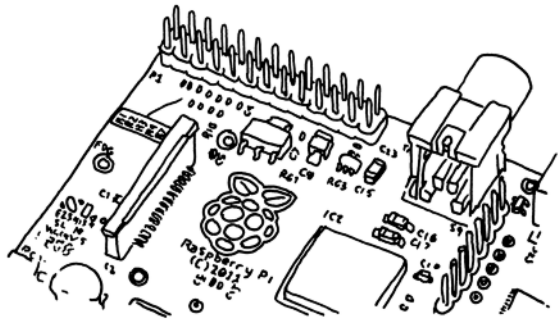


Abbildung 7-1. Die GPIO-Pins des Raspberry Pi

Auf diese GPIO-Pins können Sie zugreifen, um Hardware wie LEDs, Motoren und Relais anzusteuern – alles Beispiele für Ausgabe-Hardware. Bei Eingabe-Hardware kann der Pi den Status von Tastern, Schaltern und Stellknöpfen ermitteln oder Sensoren für Temperatur, Licht, Bewegung oder Entfernung auslesen (und viele mehr).



Einer der Nachteile des Raspberry Pi ist, dass es keine Möglichkeit gibt, *analoge Sensoren* direkt abzugreifen, wie zum Beispiel Licht- und Temperatursensoren. Dazu benötigt man einen Chip als *Analog-Digital-Wandler* oder *ADC* (Analog Digital Converter). In Anhang C erfahren Sie, wie Sie analoge Sensoren mit einem ADC auslesen.

Wenn man einen Computer mit GPIO-Pins besitzt, kann man Programme schreiben, die Eingabewerte lesen und die Ausgabe abhängig von vielen Bedingungen setzen – so einfach wie bei Programmen auf dem Desktop. Anders als bei einem klassischen Mikrocontroller-Board, das auch programmierbare GPIO-Pins besitzt, hat der Raspberry Pi ein paar zusätzliche Ein- und Ausgabemöglichkeiten wie Ihre Tastatur, Maus und den Monitor, aber auch den Ethernet-Port, der sowohl als Eingangs- als auch als Ausgangskanal dienen kann. Wenn Sie elektronische Projekte mit Mikrocontroller-Boards wie den Arduino kennen, können Sie beim Raspberry Pi ein paar mehr Eingangs- und Ausgangskanäle nutzen. Und sie sind direkt eingebaut – es sind keine zusätzlichen Schaltkreise notwendig, um sie einsetzen zu können.

Tastatur, Maus und Monitor sind nicht die einzigen Vorteile des Raspberry Pi gegenüber klassischen Mikrocontroller-Boards. Es gibt noch ein paar weitere wichtige Features, die Ihnen bei Ihren elektronischen Projekten hilfreich sein können:

Dateisystem

Die Möglichkeit, Daten in das Dateisystem von Linux zu schreiben und von dort zu lesen, macht einem bei vielen Projekten das Leben leichter. So können Sie zum Beispiel einen Temperatursensor an den Raspberry Pi anschließen und ihn einmal pro Sekunde auslesen. Jeder Wert kann an das Ende einer Log-Datei angefügt werden, die sich einfach herunterladen und in einem Grafikprogramm auswerten lässt. Man kann sie sogar direkt auf dem Raspberry Pi in eine Grafik umsetzen!

Linux-Tools

Zusammen mit der Linux-Distribution des Raspberry Pi gibt es einen Satz wichtiger Befehlszeilen-Tools, durch die Sie mit Dateien arbeiten, Prozesse steuern und viele verschiedene Aufgaben automatisieren können. Diese leistungsfähigen Tools stehen Ihnen für alle Ihre Projekte zur Verfügung. Und da es eine riesige Community mit Linux-Anwendern gibt, die auf diesen Tools aufbauen, ist Hilfe im Web meist nur eine Suche entfernt. Für eine allgemeine Linux-Hilfe finden Sie im Allgemeinen Antworten auf Stack Overflow (<http://stackoverflow.com>). Haben Sie eine Frage, die speziell den Raspberry Pi betrifft, probieren Sie es einmal im Raspberry Pi Forum (<http://www.raspberrypi.org/phpBB3/>) oder im Raspberry Pi-Bereich von Stack Overflow (<http://raspberrypi.stackexchange.com>).

Sprachen

Es gibt viele Programmiersprachen, und Embedded-Linux-Systeme wie der Raspberry Pi ermöglichen es Ihnen, die Sprache zu wählen, die Ihnen am vertrautesten ist. Die Beispiele in diesem Buch nutzen Shell-Scripting und Python, aber sie lassen sich auch leicht in Sprachen wie C, Java, Perl oder andere umsetzen.

Eingangs- und Ausgangsanschlüsse nutzen

Ein paar Dinge benötigen Sie, bevor Sie den Raspberry Pi für diese einfachen Ein-/Ausgabe-Tutorials nutzen können. Viele der Teile finden Sie in Ihrem lokalen Elektronikladen oder Sie bestellen Sie online bei Läden wie Watterott oder Adafruit. Hier ein paar der Teile:

- Lötfreie Steckplatine
- eine Sammlung verschiedener LEDs
- Kabel mit Steckern an beiden Enden
- Jumper-Kabel mit Buchse an einem und Stecker am anderen Ende. (Sie sind nicht so verbreitet wie solche mit Steckern an beiden Ende, aber Sie benötigen sie für das Verbinden der GPIO-Pins des Raspberry Pi mit der Steckplatine.)
- Druckschalter
- eine Sammlung verschiedener Widerstände

Um den Anschluss der Steckplatten-Komponenten an die Pins des Raspberry Pi zu vereinfachen, empfehlen wir das Pi Cobbler Breakout Kit von Adafruit. Damit sparen Sie sich die Jumper-Kabel mit Stecker und Buchse. Das Kit kommt als Bausatz, Sie müssen also die Teile selbst festlöten, aber das ist einfach und das Tutorial von Adafruit (<http://learn.adafruit.com/adafruit-pi-cobbler-kit/overview>) führt Sie Schritt für Schritt durch diese Aufgabe. Das Pi Cobbler Breakout Kit finden Sie auch zusammen mit den anderen oben aufgeführten Komponenten (abgesehen vom erwähnten Jumper-Wire) im Raspberry Pi Starter Kit von MAKE (<http://oreil.ly/pikit>). Das Pi Cobbler Breakout Kit ist auch in Deutschland erhältlich, beispielsweise über Farnell.

In Abbildung 7-2 haben wir jeden Pin entsprechend seiner Standard-GPIO-Signalnummer beschriftet (womit Sie den Pin in den Befehlen und in selbst geschriebenem Code ansprechen). Die unbeschrifteten Pins sind standardmäßig anderen Funktionen zugeordnet.

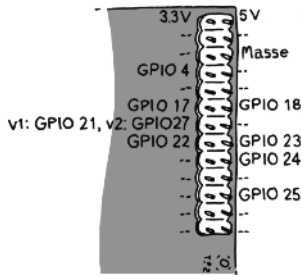


Abbildung 7-2. Die Standard-GPIO-Pins des Raspberry Pi. In neueren Versionen des Boards wurde der GPIO-Pin 21 durch Pin 27 ersetzt.



Ihnen ist vielleicht aufgefallen, dass einer der Pins in Abbildung 7-2 zwei verschiedene Nummern besitzt. In neueren Versionen des Boards wurde GPIO-Pin 21 zum Pin 27. Um die Version Ihres Boards herauszufinden, geben Sie an der Befehlszeile `cat /proc/cpuinfo` ein. Ist Ihre Versionsnummer 0002 oder 0003, haben Sie die erste Version des Boards. Bei einer höheren Nummer oder einem Buchstaben haben Sie eine neuere Version.

Digitale Ausgabe: Eine LED zum Leuchten bringen

Die einfachste Möglichkeit einer Ausgabe über die GPIO-Pins ist der Anschluss einer LED. Sie können dann die Befehlszeile unter Linux nutzen, um die LED an- und auszuschalten. Haben Sie einmal verstanden, wie diese Befehle funktionieren, sind Sie einen Schritt näher an der Möglichkeit, mit der LED anzuzeigen, ob Sie neue E-Mails haben, einen Schirm mitnehmen zu müssen, wenn Sie das Haus verlassen, oder ins Bett gehen sollten. Es ist auch sehr einfach, statt der LED ein Relais zu nutzen, um damit zum Beispiel eine Lampe anzusteuern.

Einstiegshilfe für Steckplatinen

Wenn Sie noch nie eine Steckplatine (Abbildung 7-3) genutzt haben, ist es vor allem wichtig zu wissen, welche Anschlüsse miteinander verbunden sind. Im folgenden Bild haben wir schattiert hervorgehoben, welche Anschlüsse auf einer typischen Steckplatine miteinander verbunden sind. Beachten Sie, dass die Stromversorgungsleitungen auf der linken Seite nicht mit denen auf der rechten Seite verbunden sind. Sie müssen hier zum Verbinden Jumper-Kabel mit Steckern an beiden Enden nutzen, wenn Sie auf beiden Seiten der Steckplatine Strom benötigen.

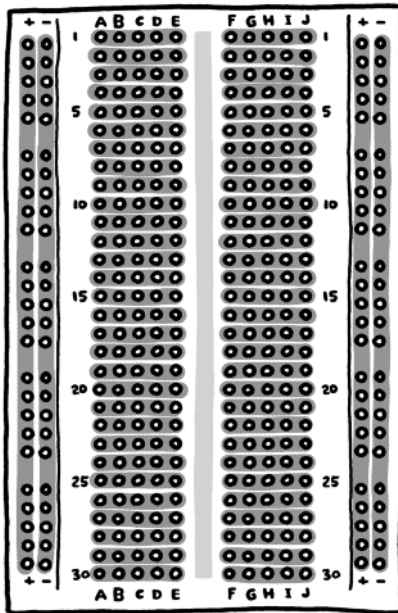


Abbildung 7-3. Steckplatine

1. Verbinden Sie mit einem Jumper-Kabel mit Stecker und Buchse Pin 25 auf dem Raspberry Pi mit der Steckplatine. In Abbildung 7-2 finden Sie die Position der Pins auf dem Pi.
2. Verbinden Sie mit einem weiteren Jumper-Kabel den Masse-Pin (Ground) des Raspberry Pi mit dem negativen Stromanschluss der Steckplatine.
3. Jetzt können Sie die LED anschließen (siehe Abbildung 7-4). Aber zuvor müssen Sie wissen, dass LEDs eine *Polarität* haben: Es ist entscheidend, welchen Draht der LED Sie wo anschließen. Von den beiden Drähten der LED ist der längere die Anode, die mit einem GPIO-Pin verbunden sein sollte. Der kürzere ist die Kathode, die mit Masse zu verbinden ist. Eine andere Möglichkeit der Unterscheidung ist ein Blick von oben auf die LED. Die flache Seite der LED zeigt die Kathode an, die mit Masse verbunden werden sollte. Stecken Sie die LED so in die Platine ein, dass die Anode mit dem Jumper-Kabel von Pin 25 verbunden ist, während die Kathode mit der Masse der Stromversorgung Kontakt hat.

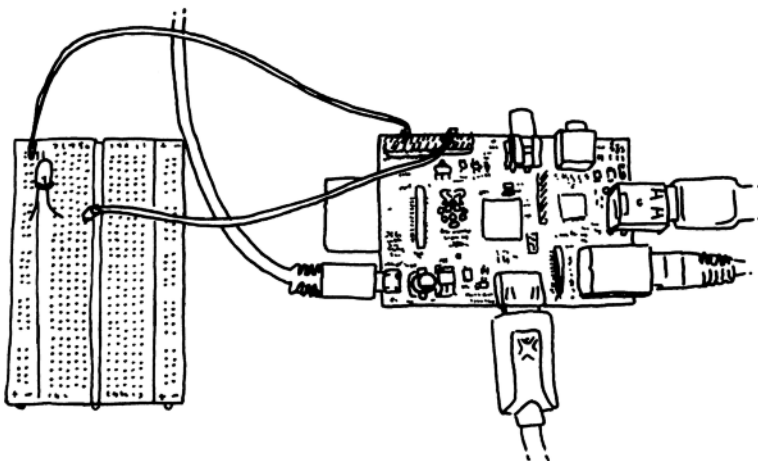


Abbildung 7-4. Eine LED mit dem Raspberry Pi verbinden

1. Schalten Sie Ihren Raspberry Pi mit angeschlossener Tastatur, Maus und Monitor ein und melden Sie sich an. Wenn die Befehlszeile erscheint, können Sie loslegen. Landen Sie in der X-Window-Umgebung, klicken Sie doppelt auf das LXTerminal-Symbol auf dem Desktop. Damit erhalten Sie auch ein Terminal-Fenster.
2. Um von der Befehlszeile auf die Ein- und Ausgangspins zugreifen zu können, müssen Sie die Befehle als Root ausführen – das *Superuser*-Konto des Raspberry Pi. Dazu geben Sie an der Befehlszeile **sudo su** ein und drücken die Eingabetaste:

```
pi@raspberrypi ~ $ sudo su
root@raspberrypi:/home/pi#
```

Der Prompt hat sich geändert und zeigt damit an, dass Sie folgende Befehle nun als Root ausführen.



Das root-Konto hat administrativen Zugriff auf alle Funktionen und Dateien des Systems. Es gibt so gut wie keinen Schutz davor, das Betriebssystem zu zerstören, wenn Sie die falschen Befehle eingeben. Seien Sie also vorsichtig, wenn Sie als Root Befehle ausführen. Haben Sie alles durcheinandergebracht, brauchen Sie aber nicht allzu besorgt zu sein, denn Sie können ja immer noch die SD-Karte mit einer sauberen Linux-Installation überschreiben. Wenn Sie Ihre Arbeit als Root erledigt haben, geben Sie `exit` ein, um wieder als Benutzer `pi` zu arbeiten.

3. Bevor Sie die LED an Pin 25 ein- und ausschalten können, müssen Sie den *Pin in den Userspace exportieren* (also den Pin auch außerhalb des Linux-Kernels benutzbar machen):

```
root@raspberrypi:/home/pi# echo 25 > /sys/class/gpio/export
```

Der Befehl `echo` schreibt die Nummer des gewünschten Pin (25) in die Datei `export` im Ordner `/sys/class/gpio`. Wenn Sie Pin-Nummern in diese spezielle Datei schreiben, wird ein neues Verzeichnis in `/sys/class/gpio` mit den Steuerdateien für den Pin erstellt. In diesem Fall wird ein neues Verzeichnis namens `/sys/class/gpio/gpio25` erzeugt.

4. Wechseln Sie mit `cd` in dieses Verzeichnis und geben Sie den Inhalt mit `ls` aus:

```
root@raspberrypi:/home/pi# cd /sys/class/gpio/gpio25
root@raspberrypi:/sys/class/gpio/gpio25# ls
active_low  direction  edge      power     subsystem uevent     value
```

Der Befehl `cd` steht für »Change Directory«. Damit wird das Arbeitsverzeichnis gewechselt, so dass Sie nicht bei jeder Datei den vollständigen Pfad angeben müssen. `ls` gibt die Dateien und Ordner innerhalb dieses Verzeichnisses aus. Mit zweien der Dateien in diesem Verzeichnis werden Sie arbeiten: `direction` und `value`.

5. Mit der Datei `direction` setzen Sie den Pin als Eingang (zum Beispiel für einen Taster) oder Ausgang (für eine LED). Da Sie eine LED an Pin 25 angeschlossen haben und Sie diese steuern wollen, werden Sie diesen Pin als Output definieren:

```
root@raspberrypi:/sys/class/gpio/gpio25# echo out > direction
```

6. Um die LED einzuschalten, nutzen Sie wieder `echo`, um die Zahl 1 in die Datei `value` zu schreiben:

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 1 > value
```

7. Nach Drücken der Eingabetaste wird die LED leuchten! Abschalten geht genauso einfach: Schreiben Sie mit `echo` eine Null in die `value`-Datei:

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 0 > value
```

Virtuelle Dateien

Die Dateien, mit denen Sie arbeiten, sind keine echten Dateien auf der SD-Karte des Raspberry Pi, sondern ein Teil des *virtuellen Dateisystems* von Linux. Dabei handelt es sich um ein System, mit dem man einfach auf Low-Level-Funktionen des Boards zugreifen kann. Sie könnten zum Beispiel die LED auch ein- und ausschalten, indem Sie in einen bestimmten Bereich des Speichers des Pi schreiben, aber dazu benötigen Sie mehr Code und müssen mehr Vorsicht walten lassen.

Wenn Sie also durch das Schreiben in eine Datei Ausgabekomponenten steuern können – wie prüfen Sie dann den Status von Eingangskomponenten? Wenn Sie »aus einer Datei lesen« vermutet haben, liegen Sie ganz richtig. Wir wollen das jetzt ausprobieren.

Digitaler Eingang: Einen Taster auslesen

Mit einfachen Tastschaltern wie dem in Abbildung 7-5 kann man wunderbar einen digitalen Eingang testen, und dazu passen sie auch noch perfekt auf eine Steckplatine.



Diese kleinen Taster werden in Elektronik-Projekten sehr gerne eingesetzt. Ein Verständnis für ihre Funktionsweise hilft beim Prototyping von Objekten sehr. Schauen Sie sich den Taster auf der Steckplatine an (siehe Abbildung 7-5): Die oberen beiden Anschlüsse sind immer miteinander verbunden, ebenso die unteren beiden. Drücken Sie den Taster, werden diese beiden Anschlusspärchen insgesamt miteinander verbunden.

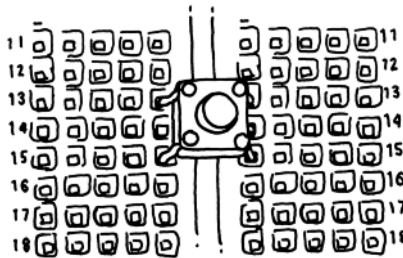


Abbildung 7-5. Taster

Lesen Sie einen digitalen Eingangswert auf einem Raspberry Pi aus, prüfen Sie damit, ob der Pin entweder mit 3,3 V oder mit Masse verbunden ist. Denken Sie daran, dass es einer dieser beiden Zustände sein muss. Ist der Pin mit einer anderen Spannung verbunden, werden Sie unerwartete Ergebnisse erhalten. Haben Sie einmal verstanden, wie ein digitaler Eingang mit einem Taster funktioniert, können Sie auch Komponenten wie magnetische Sicherheitsschalter, Arcade-Joysticks oder sogar Münzwerke von Verkaufsautomaten einsetzen.

1. Stecken Sie den Taster so auf die Steckplatine, dass seine Anschlüsse links und rechts des mittleren Bereichs verbunden sind.
2. Verbinden Sie Pin 24 des Raspberry Pi mit einem Jumper-Kabel mit einem der oberen Anschlüsse des Tasters.
3. Verbinden Sie den 3V3-Pin des Raspberry Pi mit der positiven Stromversorgung der Steckplatine.



Achten Sie darauf, den Taster mit dem 3V3-Pin und nicht mit dem 5V-Pin zu verbinden. Mehr als 3,3 V an einem Eingangspin werden Ihrem Raspberry Pi finalen Schaden zufügen.

4. Verbinden Sie einen der unteren Anschlüsse des Tasters mit der Stromversorgungsleiste. Wenn Sie jetzt den Taster drücken, werden die 3,3 V an Pin 24 anliegen.
5. Erinnern Sie sich daran, dass ein digitaler Eingangskanal immer an 3,3 V oder an Masse angeschlossen werden muss? Wenn Sie den Taster loslassen, ist Pin 24 weder mit dem einen noch mit dem anderen verbunden und *erdfrei*. Dies führt zu unvorhersagbaren Ergebnissen, daher wollen wir es beheben. Nutzen Sie einen 10K-Widerstand (mit dem Farbcode Braun/Schwarz/Orange/Silber oder Gold), um die Eingangsseite des Schalters mit der Masseleiste zu verbinden (die wiederum schon im vorigen Beispiel mit der Masse des Raspberry Pi verbunden wurde). Wird der Schalter nicht gedrückt, ist der Pin mit Masse verbunden.

Da elektrischer Strom immer den Weg des geringsten Widerstands zur Masse folgt, wandern die 3,3 V beim Drücken des Tasters zum Eingangspin des Raspberry Pi, der einen geringeren Widerstand als der 10K-Widerstand besitzt. Ist alles miteinander verbunden, sollte es so aussehen wie in Abbildung 7-6.

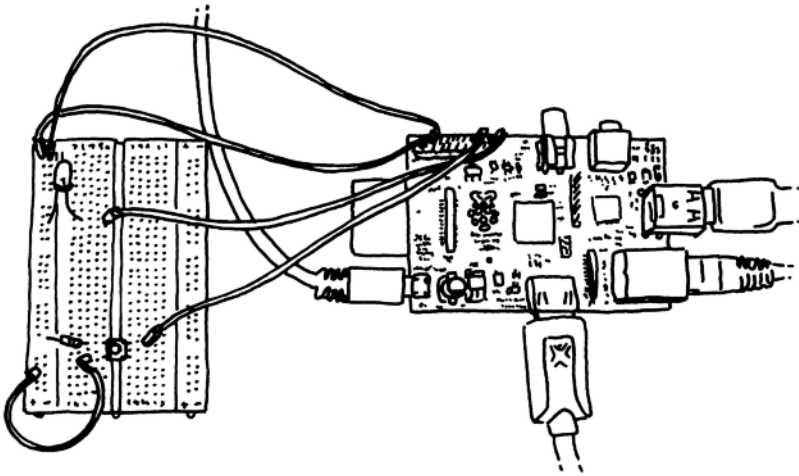


Abbildung 7-6. *Einen Taster mit dem Raspberry Pi verbinden*

6. Nachdem der Schaltkreis nun aufgebaut ist, wollen wir den Wert des Pins an der Befehlszeile auslesen. Arbeiten Sie noch nicht also Root, geben Sie **sudo su** ein.
7. Wie beim vorigen Beispiel müssen Sie den Eingangspin in den User-space importieren:

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 24 > /sys/class/gpio/export
```

8. Lassen Sie uns in das Verzeichnis wechseln, das durch die Export-Operation erzeugt wurde:

```
root@raspberrypi:/sys/class/gpio/gpio25# cd /sys/class/gpio/gpio24
```

9. Jetzt setzen wir die Ein- und Ausgaberichtung des Pins auf Input:

```
root@raspberrypi:/sys/class/gpio/gpio24# echo in > direction
```

10. Um den Wert des Pins auszulesen, verwenden Sie den Befehl cat, mit dem der Inhalt von Dateien im Terminal ausgegeben wird. Seinen Namen hat cat von der Möglichkeit, damit auch Dateien zu konkatenieren, also aneinanderzuhängen. Es kann aber eben auch den Inhalt einer Datei anzeigen.

```
root@raspberrypi:/sys/class/gpio/gpio24# cat value
0
```

11. Die Null zeigt, dass der Pin mit Masse verbunden ist. Drücken Sie nun den Taster und halten Sie ihn gedrückt, während Sie den Befehl erneut ausführen:

```
root@raspberrypi:/sys/class/gpio/gpio24# cat value
1
```

12. Wenn Sie die Eins sehen, haben Sie es richtig gemacht!



Um einen Befehl einfach ein zweites Mal auszuführen, blättern Sie mit der Cursortaste nach oben, bis Sie den gewünschten Befehl sehen, und drücken dann die Eingabetaste.

Nachdem Sie nun die Befehlszeile von Linux nutzen können, um eine LED zu steuern oder den Status eines Tasters auszulesen, wollen wir ein paar in Linux eingebaute Tools nutzen, um ein sehr einfaches Projekt rund um digitale Ein- und Ausgabe aufzusetzen.

Projekt: Cron-Lampenzeitschaltuhr

Gehen wir davon aus, dass Sie morgen früh auf eine lange Reise gehen und verhindern wollen, dass bei Ihnen eingebrochen wird. Eine Lampenzeitschaltuhr ist da sehr hilfreich, aber die Baumärkte sind nachts dummerweise geschlossen und Sie haben in der Frühe nicht mehr genug Zeit, bevor der Flieger geht. Aber Sie sind ja ein Raspberry Pi-Bastler und haben bestimmt ein paar weitere Zutaten in Ihrer Elektronikbox herumfliegen, nämlich:

- Raspberry Pi-Board
- Steckplatine
- Jumper-Kabel, Stecker/Buchse
- PowerSwitch Tail II-Relais
- Verbindungskabel

Mit diesen Dingen sowie *Shell-Skripten* und *cron* können Sie Ihre eigene programmierbare Zeitschaltuhr bauen.

Befehle skripten

Ein Shell-Skript ist eine Datei mit einer Folge von Befehlen (so wie die, die Sie beim Steuern und Auslesen der Pins genutzt haben). Schauen Sie sich das folgende Skript und die folgende Erläuterung an.

```
#!/bin/bash # ❶
echo Pin $1 exportieren. # ❷
echo $1 > /sys/class/gpio/export # ❸
echo Übertragung auf out setzen.
echo out > /sys/class/gpio/gpio$1/direction # ❹
echo Pin anschalten.
echo 1 > /sys/class/gpio/gpio$1/value
```

- ❶ Diese Zeile ist für alle Shell-Skripten notwendig.
- ❷ "\$1" verweist auf das erste Argument beim Aufruf von der Befehlszeile.
- ❸ Statt eine bestimmte Pin-Nummer zu exportieren, nutzt das Skript das erste Argument, das beim Aufruf mitgegeben wurde.
- ❹ Beachten Sie, dass auch hier das erste Argument als Pin-Nummer genutzt wird.

Sichern Sie dies als eine Textdatei mit dem Namen `on.sh` und sorgen Sie mit dem Befehl `chmod` dafür, dass sie ausgeführt werden kann:

```
root@raspberrypi:/home/pi# chmod +x on.sh
```



Sie müssen diese Befehle immer noch als Root ausführen. Geben Sie `sudo` ein, wenn Sie Fehler wie „Permission denied“ erhalten.

Ein Argument an der Befehlszeile ist eine Möglichkeit, Informationen an ein Programm oder Skript weiterzureichen. Dazu gibt man es nach dem Namen des Befehls an. Wenn Sie ein Shell-Skript schreiben, steht `$1` für das erste Argument, `$2` für das zweite und so weiter. Bei `on.sh` geben Sie die gewünschte Pin-Nummer zum Exportieren und Einschalten mit. Statt Pin 25 *hart kodiert* im Shell-Skript stehen zu haben, ist es universeller einsetzbar, wenn man auf den an der Befehlszeile angegebenen Pin verweist. Um Pin 25 zu exportieren und einzuschalten, können Sie jetzt die folgende Anweisung geben:

```
root@raspberrypi:/home/pi/# ./on.sh 25 ❶
Pin 25 exportieren.
Übertragung auf out setzen.
Pin anschalten.
```

- ❶ Das „`./`“ vor dem Dateinamen dient dazu, das Skript auszuführen, das im aktuellen Verzeichnis steht.

Ist die LED noch vom vorletzten Abschnitt an Pin 25 angeschlossen, sollte sie jetzt eingeschaltet sein. Erstellen wir ein zweites Skript namens `off.sh`, das die LED abschaltet:

```
#!/bin/bash
echo Pin ausschalten.
echo 0 > /sys/class/gpio/gpio$1/value
echo Export von Pin $1 zurücknehmen
echo $1 > /sys/class/gpio/unexport
```

Jetzt machen wir das Skript wieder ausführbar und lassen es laufen:

```
root@raspberrypi:/home/pi/temp# chmod +x off.sh
root@raspberrypi:/home/pi/temp# ./off.sh 25
Pin ausschalten.
Export von Pin 25 zurücknehmen
```

Funktioniert alles, sollte die LED wieder abgeschaltet sein.

Eine Lampe anschließen

Natürlich bietet so eine kleine LED nicht genug Licht, um Einbrecher über Ihre Anwesenheit zu täuschen. Daher wollen wir eine Lampe mit dem Raspberry Pi verbinden.

1. Entfernen Sie die LED, die mit Pin 25 verbunden ist.
2. Schließen Sie zwei Kabel an der Steckplatine an – eins muss mit dem Pin 25 des Pi verbunden sein, das andere mit dem Masse-Bus.
3. Schließen Sie das freie Ende des mit Pin 25 verbundenen Kabels am „+in“-Eingang des Power Switch Tail an.
4. Schließen Sie das freie Ende des mit dem Masse-Bus verbundenen Kabels am „-in“-Eingang des Power Switch Tail an. Vergleichen Sie Ihre Verdrahtung mit Abbildung 7-7.
5. Verbinden Sie den PowerSwitch Tail mit der Steckdose und einer Lampe. Achten Sie darauf, dass der Schalter der Lampe eingeschaltet ist.
6. Wenn Sie nun `./on.sh 25` aufrufen, sollte die Lampe eingeschaltet werden, mit `./off.sh 25` sollte sie wieder ausgehen!



Im Power Switch Tail gibt es ein paar Elektronikbauteile, die es Ihnen ermöglichen, Geräte wie eine Lampe oder einen Mixer, die mit normaler Haushaltsspannung betrieben werden, durch ein Signal mit geringer Spannung (zum Beispiel vom Raspberry Pi) anzusteuern. Der Klick, den Sie vom Power Switch Tail beim Ein- oder Ausschalten hören, ist das Relais – das zentrale Bauteil des Tails. Ein Relais funktioniert wie ein Schalter für Geräte mit höherer Spannung. Es schaltet ein oder aus – gesteuert von dem Signal vom Raspberry Pi.

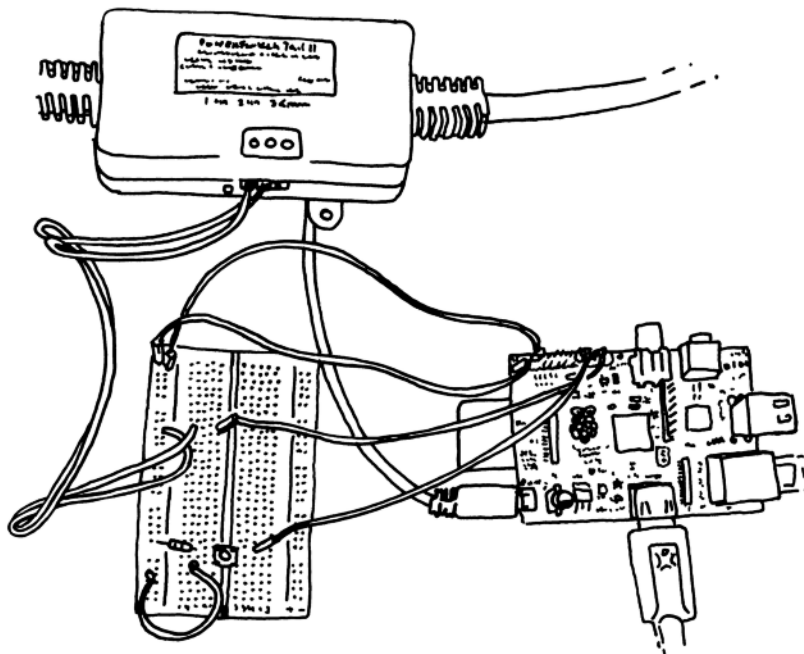


Abbildung 7-7. Einen PowerSwitch Tail II mit dem Raspberry Pi verbinden

Befehle zeitgesteuert über cron ausführen

Sie haben jetzt also ein paar Anweisungen in zwei einfachen Skripten verpackt, mit denen Sie einen Pin ein- oder ausschalten können. Und mit der Lampe, die über den Power Switch Tail mit dem Raspberry Pi verbunden ist, können Sie mit einem einzelnen Befehl Licht ein- oder ausschalten. Jetzt können Sie cron nutzen, um einen Zeitplan zu erstellen und das Licht zu verschiedenen Uhrzeiten ein- oder auszuschalten. cron ist der Job Scheduler von Linux. Damit können Sie dafür sorgen, dass Befehle zu bestimmten Uhrzeiten und an bestimmten Tagen oder mit einer bestimmten Regelmäßigkeit (zum Beispiel ein Mal pro Stunde) ausgeführt werden. Sie werden zwei Jobs einrichten: Einer wird das Licht um 20 Uhr einschalten, der andere wird es um 2 Uhr nachts ausschalten.



Wie bei anderen zeitabhängigen Programmen sollten Sie hier sicherstellen, dass Ihr Raspberry Pi das korrekte Datum und die richtige Uhrzeit kennt (siehe Kapitel 1).

Um diese Jobs hinzuzufügen, müssen Sie die cron-Tabelle bearbeiten (eine Liste mit Befehlen, die Linux zu bestimmten Uhrzeiten ausführt):

```
root@raspberrypi:/home/pi/# crontab -e
```

Damit wird ein Texteditor geöffnet, in dem Sie die cron-Tabelle bearbeiten können. Am Dateianfang stehen einige Informationen, wie man die cron-Tabelle bearbeitet. Mit den Cursorstasten scrollen Sie an das Dateiende und fügen diese beiden Einträge an:

```
0 20 * * * /home/pi/on.sh 25
0 2 * * * /home/pi/off.sh 25
```



cron ignoriert Zeilen, die mit dem Hash-Zeichen beginnen. Wollen Sie eine Zeile zeitweilig deaktivieren, ohne sie zu löschen, fügen Sie am Anfang der Zeile ein Hash-Zeichen ein.

Drücken Sie Strg-X, um das Programm zu beenden, tippen Sie y, um die Datei zu speichern, wenn Sie danach gefragt werden, und drücken Sie die Eingabetaste, um den vorgeschlagenen Dateinamen zu akzeptieren. Ist die Datei gesichert und sind Sie zur Befehlszeile zurückgekehrt, sollte der Text `installing new crontab` erscheinen, der besagt, dass die Änderungen, die Sie vorgenommen haben, von cron umgesetzt werden.

Mehr zu Cron

Mit Cron können Sie Aufgaben zu bestimmten Zeitpunkten oder in bestimmten Intervallen ausführen lassen. Es gibt fünf Zeitfelder (oder sechs, wenn Sie auch noch abhängig vom Jahr etwas ausführen wollen), die jeweils durch ein Leerzeichen getrennt sind. Danach folgt noch – wieder durch ein Leerzeichen getrennt – der auszuführende Befehl. Sternchen (Asteriske) geben bei einem Feld an, dass die Aufgabe in jeder Periode auszuführen ist. Zum Beispiel:

Tabelle 7-1. Cron-Eintrag für das Einschalten von Licht jeden Tag um 20.00 Uhr

0	20	*	*	*	/home/pi/on.sh 25
Minute (:00)	Stunde (20)	Jeden Tag	Jeden Monat	Jeden Wochentag	Pfad zur Anweisung

Nehmen wir an, dass die Lampe nur an Werktagen eingeschaltet werden soll. Dann sähe der crontab-Eintrag so aus:

Tabelle 7-2. Cron-Eintrag für das Einschalten von Licht an jedem Werktag um 20:00 Uhr

0	20	*	*	1-5	/home/pi/on.sh 25
Minute (:00)	Stunde (20)	Jeden Tag	Jeden Monat	Montag bis Freitag	Pfad zur Anweisung

Nehmen wir nun an, dass Sie ein Shell-Skript besitzen, das prüft, ob Sie neue Mails bekommen haben, und Ihnen dann eine Mail schickt. So lassen Sie das Skript alle fünf Minuten laufen:

Tabelle 7-3. Cron-Eintrag für das Prüfen von Mails alle fünf Minuten

* /5	*	*	*	*	/home/pi/checkMail.sh
Alle fünf Minuten	Jede Stunde	Jeden Tag	Jeden Monat	Jeden Wochentag	Pfad zur Anweisung

* /5 steht für eine Periode von fünf Minuten.

Wie Sie sehen, ist cron ein sehr leistungsfähiges Tool, mit dem Sie Aufgaben zu bestimmten Terminen und Uhrzeiten oder in bestimmten Intervallen ausführen lassen können.

Weitere Informationen

eLinux's Raspberry Pi GPIO Reference Page

(http://elinux.org/RPi_Low-level_peripherals)

Dies ist die umfassendste Referenz zu den GPIO-Pins des Raspberry Pi.

Adafruit: MCP230xx GPIO Expander on the Raspberry Pi

(<http://learn.adafruit.com/mcp230xx-gpio-expander-on-the-raspberry-pi>)

Haben Sie für Ihre Aufgabe nicht genug Pins zur Verfügung, bietet Adafruit den MCP23008-Chip mit acht weiteren GPIO-Pins oder den MCP23017 für 16 weitere GPIO-Pins.

8/Ein- und Ausgänge mit Python programmieren

Am Ende von Kapitel 7 haben Sie die GPIO-Pins des Raspberry Pi über ein Shell-Skript »programmiert«. In diesem Kapitel werden Sie lernen, wie Sie das Gleiche mit Python erreichen ... und ein bisschen mehr. Wie beim Shell-Skript können Sie mit Python auf die Pins zugreifen, indem Sie Code zum Lesen und Steuern der Pins schreiben.

Der Vorteil von Python gegenüber Shell-Skripten ist, dass man den Code besser schreiben und lesen kann. Es gibt sehr viele Python-Module, mit denen Sie komplexe Aufgaben mit wenigen Zeilen Code einfach lösen können. In Tabelle 3-2 sind ein paar Module aufgeführt, die für Ihre Projekte nützlich sein könnten. Und das Beste ist: Es gibt ein Python-Modul namens `raspberry-gpio-python` (<http://code.google.com/p/raspberry-gpio-python/>), mit dem Sie die GPIO-Pins problemlos auslesen und ansteuern können. Sie werden in diesem Kapitel lernen, wie Sie das Modul einsetzen.

GPIO in Python installieren und testen

In der neuesten Version der Raspbian-Distribution ist das GPIO-Modul schon installiert. Nutzen Sie eine ältere Version von Raspbian, müssen Sie es eventuell installieren. Um dies herauszufinden, nutzen Sie den interaktiven Interpreter von Python (wie Sie in Kapitel 3 gelernt haben, können Sie mit dem interaktiven Interpreter Python-Codezeilen eintippen, die dann direkt ausgewertet werden – im Gegensatz zum Schreiben von Code in einer Datei, die ausgeführt wird).

1. Rufen Sie den interaktiven Interpreter von Python am Terminal als Root auf. (Da `raspberry-gpio-python` Root-Zugriff benötigt, um die Pins auszulesen und zu steuern, müssen Sie den interaktiven Interpreter von Python mit dem `sudo`-Befehl als Root starten.)

```
pi@raspberrypi ~ $ sudo python
Python 2.7.3rc2 (default, May  6 2012, 20:02:25)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

2. Zeigt sich der Prompt >>>, versuchen Sie, dieses Modul zu importieren:

```
>>> import RPi.GPIO as GPIO
```

3. Erhalten Sie keinen Fehler, ist schon alles fertig.

Bekommen Sie dagegen einen Fehler angezeigt, wenn Sie versuchen, das GPIO-Modul zu importieren, können Sie die Installation in wenigen Schritten nachholen – dank apt-get, dem Paketmanager auf dem Raspberry Pi.

Ist raspberry-gpio-python bei Ihnen also noch nicht installiert, gehen Sie wie folgt vor:

1. Verlassen Sie den Interpreter (Drücken Sie Strg-D oder tippen Sie `exit()` und drücken die Eingabetaste), aktualisieren Sie den Paketindex von apt-get und starten Sie die Installation für *raspberry-gpio-python*:

```
>>> exit()
pi@raspberrypi ~ $ sudo apt-get update
pi@raspberrypi ~ $ sudo apt-get install python-rpi.gpio
```

2. Wenn das erledigt ist, wechseln Sie zurück zum interaktiven Interpreter von Python und importieren das Modul:

```
pi@raspberrypi ~ $ sudo python
Python 2.7.3rc2 (default, May  6 2012, 20:02:25)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import RPi.GPIO as GPIO
>>>
```



In diesem Kapitel nutzen wir Python 2.7 statt Python 3, da eines der von uns eingesetzten Module auf dem Raspberry Pi nur für Python 2.x vorliegt. Geben Sie an der Befehlszeile `python` ein, läuft aktuell standardmäßig Python 2.7. Dieses Verhalten kann sich in Zukunft ändern (Sie können Python 2.7 aber explizit ausführen, indem Sie statt `python` den Befehl `python2.7` aufrufen).

Ein wichtiger Unterschied zwischen den beiden Versionen ist die Ausgabe von Text an der Konsole. In Python 2.x nutzen Sie `print "Hello, World!"`, währendes in Python 3 `print("Hello, World!")` ist.

Erhalten Sie bei der Eingabe der Import-Anweisung keine Fehlermeldungen, können Sie das Modul nun das erste Mal ausprobieren.

1. Bevor Sie die Pins verwenden können, müssen Sie dem GPIO-Modul mitteilen, wie Ihr Code darauf verweisen wird. In Kapitel 7 hingen die von uns genutzten Pin-Nummern nicht mit ihrer Reihenfolge auf dem Board zusammen. Sie nutzten stattdessen die Signalnamen des Broadcom-Chips, der sich auf dem Board befindet. Bei diesem Python-Modul können Sie auswählen, wie Sie die Pins bezeichnen wollen. Um die Nummerierung anhand des Board-Layouts zu nutzen, verwenden Sie `GPIO.setmode(GPIO.BOARD)`. Lassen Sie uns aber bei der Pin-Nummerierung bleiben, die wir in Kapitel 7 eingesetzt haben (`GPIO.setmode(GPIO.BCM)`) – das nutzen auch der Pi Cobbler von Adafruit und ähnliche Breakout-Boards:

```
>>> GPIO.setmode(GPIO.BCM)
```

2. Setzen Sie die Verwendungsrichtung von Pin 25 auf Output:

```
>>> GPIO.setup(25, GPIO.OUT)
```

3. Schließen Sie eine LED am Pin 25 an, so wie Sie dies schon in Einstiegs-hilfe für Steckplatinen (S. 86) getan haben.
4. Schalten Sie die LED ein:

```
>>> GPIO.output(25, GPIO.HIGH)
```

5. Schalten Sie die LED aus:

```
>>> GPIO.output(25, GPIO.LOW)
```

6. Beenden Sie den interaktiven Interpreter von Python:

```
>>> exit()
pi@raspberrypi ~ $
```



In Kapitel 7 haben Sie gelernt, dass digitale Ein- und Ausgangssignale auf dem Raspberry Pi entweder 3,3 V haben oder mit Masse verbunden sein müssen. In der Digital-Elektronik bezeichnen wir diese Signale als high und low. Denken Sie daran, dass nicht jede verfügbare Hardware 3,3 V für das High-Signal nutzt – manche verwenden 1,8 V oder 5 V. Wenn Sie vorhaben, Ihren Raspberry Pi über seine GPIO-Pins mit digitaler Hardware zu verbinden, müssen Sie darauf achten, dass sie auch 3,3 V nutzt.

Diese Schritte haben Ihnen ungefähr gezeigt, wie Sie die GPIO-Pins mit Hilfe von Python-Anweisungen steuern können. So wie Sie ein Shell-Skript erstellt haben, um die Pins ein- und auszuschalten (siehe Kapitel 7), werden Sie nun ein Python-Skript schreiben, um die Pins auszulesen und zu setzen.

Eine LED blinken lassen

Um eine LED mit Python blinken zu lassen, werden Sie vor allem die Anweisungen nutzen, mit denen Sie schon im interaktiven Interpreter experimentiert haben. Bei den nächsten Schritten gehen wir davon aus, dass Sie mit der Desktop-Umgebung arbeiten, aber Sie dürfen auch gerne die Befehlszeile einsetzen und diese Python-Skripte dort schreiben und ausführen.

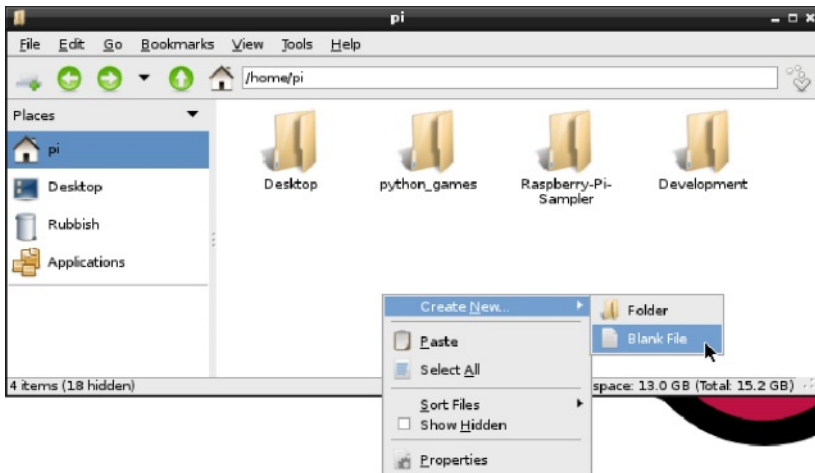


Abbildung 8-1. Eine neue Datei im Home-Verzeichnis erstellen

1. Öffnen Sie den File Manager, indem Sie sein Symbol in der Taskbar anklicken.
2. Achten Sie darauf, dass Sie sich im Home-Verzeichnis befinden – standardmäßig `/home/pi`. Wenn nicht, klicken Sie auf das Home-Symbol unter dem Places-Bereich.
3. Erstellen Sie in Ihrem Home-Verzeichnis eine Datei namens `blink.py`. Dazu klicken Sie mit der rechten Maustaste in das Fenster des Home-Verzeichnisses, wählen „Create New...“ und klicken dann auf „Blank File.“ Geben Sie der Datei den Namen `blink.py`.
4. Klicken Sie doppelt auf `blink.py`, um es im Standard-Texteditor Leafpad zu öffnen.

5. Geben Sie den folgenden Code ein und sichern Sie die Datei:

```
import RPi.GPIO as GPIO ❶
import time ❷

GPIO.setmode(GPIO.BCM) ❸
GPIO.setup(25, GPIO.OUT) ❹

while True: ❺
    GPIO.output(25, GPIO.HIGH) ❻
    time.sleep(1) ❼
    GPIO.output(25, GPIO.LOW) ❸
    time.sleep(1) ❹
```

- ❶ Den Code importieren, der zum Ansteuern der GPIO notwendig ist
- ❷ Den Code für die sleep-Funktion importieren
- ❸ Die Chips-Signalnummern nutzen
- ❹ Pin 25 als Ausgabe-Pin setzen
- ❺ Eine unendliche Schleife aus dem folgenden eingerückten Code erstellen
- ❻ Die LED einschalten
- ❼ Eine Sekunde warten
- ❸ Die LED ausschalten
- ❹ Eine Sekunde warten



Denken Sie daran, dass in Python das Einrücken entscheidend ist.

6. Öffnen Sie LXTerminal und rufen Sie dann die folgenden Befehle auf, um in Ihr Home-Verzeichnis zu wechseln und das Skript auszuführen:

```
pi@raspberrypi ~/Development $ cd ~
pi@raspberrypi ~ $ sudo python blink.py
```

- 7. Ihre LED sollte nun blinken!
- 8. Drücken Sie Strg+C, um das Skript zu stoppen und an die Befehlszeile zurückzukehren.

Versuchen Sie, das Skript zu verändern, um durch den Einsatz von Zahlen zwischen 0 und 1 in `time.sleep()` die LED schneller blinken zu lassen. Sie können auch versuchen, weitere LEDs anzuschließen und sie in Form eines Musters blinken zu lassen. Dabei können Sie die folgenden GPIO-Pins nutzen: 4, 17, 18, 21, 22, 23, 24 oder 25 (siehe Abbildung 7-2).

Einen Taster auslesen

Soll etwas passieren, wenn Sie einen Taster drücken, können Sie dies durch eine Technik namens *Polling* umsetzen. Beim Polling wird immer wieder auf eine bestimmte Bedingung geprüft. In diesem Fall wird kontrolliert, ob der Taster den Eingangs-Pin mit 3,3 V oder mit der Masse verbindet. Um mehr über Polling zu lernen, werden Sie ein neues Python-Skript erstellen, das Text auf dem Bildschirm ausgibt, sobald der Anwender einen Taster drückt.

1. Schließen Sie einen Taster wie in „Digitaler Eingang: Einen Taster auslesen“ (S. 90) an, wobei Sie Pin 24 als Eingangspin verwenden. Vergessen Sie nicht den Pulldown-Widerstand, der zwischen Masse und den Eingangspin gehört.
2. Erstellen Sie in Ihrem Home-Verzeichnis eine Datei namens *button.py* und öffnen Sie sie im Editor.
3. Geben Sie den folgenden Code ein:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(24, GPIO.IN) ❶

count = 0 ❷

while True:
    inputValue = GPIO.input(24) ❸
    if (inputValue == True): ❹
        count = count + 1 ❺
        print("Taster " + str(count) + " Mal gedrückt.") ❻
        time.sleep(.01) ❼
```

- ❶ Pin 24 als Input setzen
- ❷ Eine Variable namens count erstellen und 0 darin ablegen
- ❸ Den Wert von Pin 24 in inputValue ablegen
- ❹ Prüfen, ob dieser Wert True ist (der Taster also gedrückt ist)
- ❺ Wenn ja, Zähler erhöhen
- ❻ Text am Terminal ausgeben
- ❼ Kurz warten, aber anderen Programmen die Möglichkeit geben, selbst zu laufen, statt nur die Prozessorzeit zu verbraten

4. Wechseln Sie zurück zum LXTerminal und führen Sie das Skript aus:

```
pi@raspberrypi ~ $ sudo python button.py
```

5. Drücken Sie nun den Taster. Wenn Sie alles richtig gemacht haben, sehen Sie ein paar Zeilen lang „Der Taster wurde gedrückt.“ – jedes Mal, wenn Sie den Taster drücken.

Dieser Code prüft den Status des Tasters 100-mal pro Sekunde, darum wird auch immer mehr als einmal der Satz ausgegeben (sofern Sie nicht wirklich schnelle Finger haben). Die Python-Anweisung `time.sleep(.01)` steuert, wie oft der Taster abgefragt wird.

Aber warum prüfen wir den Taster nicht kontinuierlich? Wenn Sie die Anweisung `time.sleep(.01)` aus dem Programm entfernen, wird die Schleife unglaublich schnell und Sie erfahren viel früher, ob der Taster gedrückt wurde. Das hat aber ein paar Nachteile: Der Prozessor auf dem Board wird fortlaufend in Anspruch genommen, so dass andere Programme nur schwer weiterlaufen können und der Stromverbrauch des Raspberry Pi ansteigt. Da *button.py* sich den Rechner mit anderen Programmen teilen muss, müssen Sie darauf achten, dass das Skript nicht alle Ressourcen für sich beansprucht.

Nun fügen wir noch eine weitere Zeile Code ein, um das Erkennen eines einzelnen Tastendrucks zu verbessern:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(24, GPIO.IN)

count = 0

while True:
    inputValue = GPIO.input(24)
    if (inputValue == True):
        count = count + 1
        print("Taster " + str(count) + " Mal gedrückt.")
        time.sleep(.3) ❶
        time.sleep(.01)
```

❶ Hilft dabei, einen einzelnen Tasterdruck besser zu erkennen

Diese zusätzliche Codezeile hilft sicherzustellen, dass jeder Tasterdruck nur einmal registriert wird. Aber es ist noch keine perfekte Lösung. Versuchen Sie einmal, den Taster gedrückt zu halten. Weiterhin wird dreimal pro Sekunde der Zähler hochgesetzt. Versuchen Sie, den Taster ganz schnell hintereinander zu drücken. Dann werden nicht alle Tasterdrücke registriert, wenn man häufiger als dreimal pro Sekunde drückt.

Das sind die Herausforderungen, denen Sie sich gegenübersehen, wenn Sie den Status eines digitalen Eingangskanals pollen wollen. Eine Möglichkeit, dies zu umgehen, ist der Einsatz eines *Interrupts*, bei dem ein bestimmter Codeblock immer dann ausgeführt wird, wenn die Hardware eine Änderung des Pin-Status erkennt. Es gibt momentan eine experimentelle Unterstützung für Interrupts in `RPi.GPIO`, und Sie erfahren mehr über dieses Feature in der Dokumentation zur Bibliothek (<http://pypi.python.org/pypi/RPi.GPIO>).

Projekt: Ein einfaches Soundboard

Nachdem Sie nun wissen, wie Sie die Eingangspins eines Raspberry Pi auslesen, können Sie die Sound-Funktionen von Pygame nutzen, um ein Soundboard zu basteln. Mit einem Soundboard können Sie kurze Audio-Aufzeichnungen abspielen, wenn dessen Taster gedrückt werden. Um Ihr eigenes Soundboard zu bauen, benötigen Sie neben Ihrem Raspberry Pi noch folgende Bauteile:

- 3 Drucktaster
- Jumper-Kabel Stecker auf Buchse
- Standard-Jumper-Kabel oder passend gekürzter Schalt draht
- Steckplatine
- 3 Widerstände mit je 10 K Ω
- Computer-Lautsprecher oder einen HDMI-Monitor mit eingebauten Lautsprechern

Dazu brauchen Sie noch ein paar nicht komprimierte Sound-Dateien im .wav-Format. Zu Testzwecken gibt es auf dem Raspberry Pi schon ein paar solche Dateien, die Sie einsetzen können. Funktioniert das Soundboard erst einmal, ist es ein Leichtes, sie durch die eigentlich gewünschten Dateien zu ersetzen, auch wenn Sie sie vorher vielleicht erst ins .wav-Format umwandeln müssen. Fangen Sie aber zunächst mit dem Schaltkreis an:

1. Verbinden Sie den Masse-Pin des Raspberry Pi über ein Jumper-Kabel mit Stecker und Buchse mit der negativen Leiste auf der Steckplatine.
2. Verbinden Sie den 3,3-V-Pin des Raspberry Pi über ein Jumper-Kabel mit Stecker und Buchse mit der positiven Leiste auf der Steckplatine.
3. Stecken Sie die drei Drucktaster auf die Platine so, dass sie den mittleren Bereich überspannen.
4. Verbinden Sie durch Standard-Jumper-Kabel oder kurzen Schalt draht die positive Leiste der Steckplatine mit dem oberen Pin jedes Tasters.
5. Nun fügen Sie die Pulldown-Widerstände ein. Verbinden Sie den unteren Pin jedes Tasters mit einem 10K-Widerstand.
6. Verbinden Sie durch Jumper-Kabel mit Stecker und Buchse den unteren Pin jedes Tasters (den mit dem 10K-Widerstand) mit den GPIO-Pins des Raspberry Pi. Für dieses Projekt haben wir die Pins 23, 24 und 25 verwendet.

Abbildung 8-2 zeigt den vollständigen Aufbau. Wir haben dieses Diagramm mit Fritzing (<http://fritzing.org>) erstellt, einem Open-Source-Tool zum Entwerfen von Hardware-Designs.

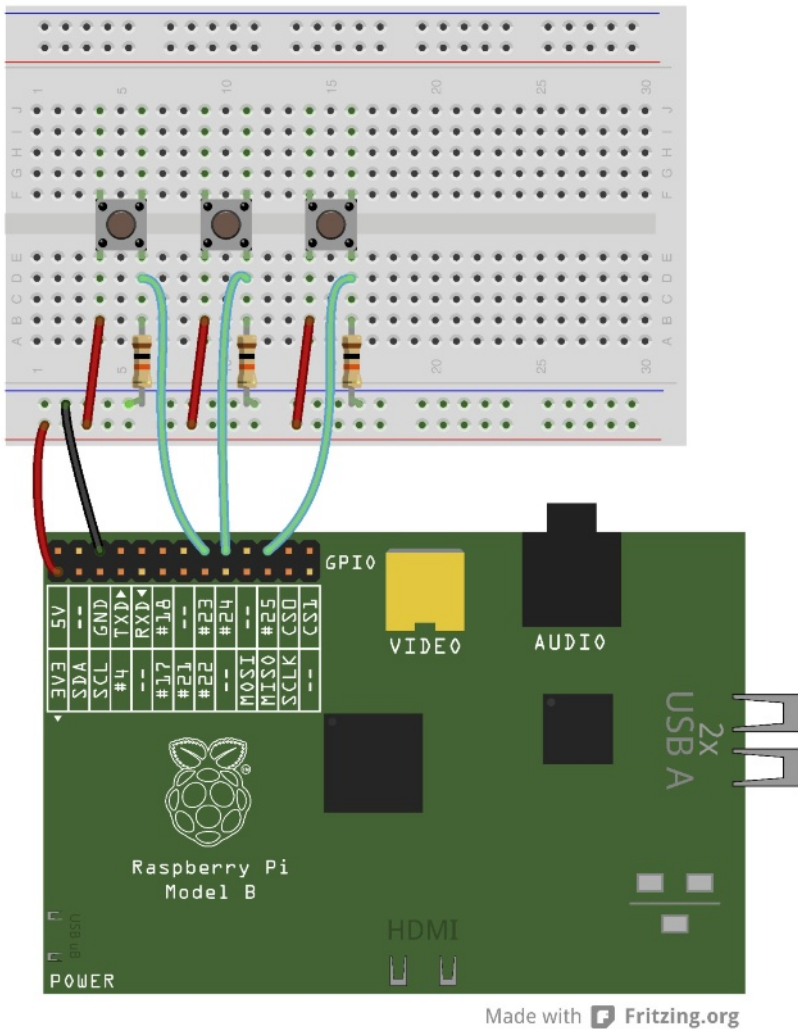


Abbildung 8-2. *Vollständiger Schaltkreis für das Soundboard-Projekt*

Nachdem Sie die Steckplatine aufgebaut haben, müssen wir uns noch um den Code kümmern.

1. Erstellen Sie in Ihrem Home-Verzeichnis ein neues Verzeichnis namens *soundboard*.
2. Öffnen Sie diesen Ordner und erstellen Sie dort eine Datei mit dem Namen *soundboard.py*.

3. Öffnen Sie *soundboard.py* und geben Sie folgenden Code ein:

```
import pygame.mixer
from time import sleep
import RPi.GPIO as GPIO
from sys import exit

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(25, GPIO.IN)

pygame.mixer.init(48000, -16, 1, 1024) ❶

soundA = pygame.mixer.Sound("/usr/share/sounds/alsa/Front_Center.wav") ❷
soundB = pygame.mixer.Sound("/usr/share/sounds/alsa/Front_Left.wav")
soundC = pygame.mixer.Sound("/usr/share/sounds/alsa/Front_Right.wav")

soundChannelA = pygame.mixer.Channel(1) ❸
soundChannelB = pygame.mixer.Channel(2)
soundChannelC = pygame.mixer.Channel(3)

print "Soundboard bereit" ❹

while True:
    try:
        if (GPIO.input(23) == True): ❺
            soundChannelA.play(soundA) ❻
        if (GPIO.input(24) == True):
            soundChannelB.play(soundB)
        if (GPIO.input(25) == True):
            soundChannelC.play(soundC)
        sleep(.01) ❼
    except KeyboardInterrupt: ❽
        exit()
```

- ❶ Den Mixer von Pygame initialisieren
- ❷ Die Klänge laden
- ❸ Drei Kanäle einrichten – einen für jeden Klang –, so dass wir parallel drei verschiedene Klänge abspielen können
- ❹ Den Anwender informieren, dass das Soundboard bereit ist (in Python-2-Syntax)
- ❺ Wenn der Pin high ist, die folgende Zeile ausführen
- ❻ Den Klang abspielen
- ❼ Den Prozessor nicht komplett in Beschlag nehmen, indem die Taster nicht häufiger als nötig abgefragt werden
- ❽ Das Skript sauber verlassen, wenn der Anwender Strg+C drückt, ohne die Traceback Message angezeigt zu bekommen

4. Wechseln Sie zur Befehlszeile und navigieren Sie zu dem Ordner, in dem Sie *soundboard.py* gespeichert haben. Führen Sie nun das Skript mit Python 2 aus:

```
pi@raspberrypi ~/soundboard $ sudo python soundboard.py
```

5. Nach der Anzeige von „Soundboard bereit“ drücken Sie auf die Taster, um die Audio-Samples abzuspielen.



Pygame steht zwar auch für Python 3 zur Verfügung, aber in der Standard-Installation für den Raspberry Pi ist es nur für Python 2 installiert.

Abhängig davon, wie Ihr Raspberry Pi eingerichtet ist, werden die Audio-Informationen über HDMI an Ihren Monitor übermittelt oder über die 3,5-mm-Audiobuchse auf dem Board ausgegeben. Um das zu ändern, verlassen Sie das Skript durch Strg+C und führen die folgende Befehlszeile aus, um die analoge Audioausgabe zu verwenden:

```
pi@raspberrypi ~/soundboard $ sudo amixer cset numid=3 1
```

Um die Audio-Daten über HDMI an den Monitor zu schicken, nutzen Sie:

```
pi@raspberrypi ~/soundboard $ sudo amixer cset numid=3 2
```

Die mitgelieferten Klänge sind natürlich nicht sehr spannend, aber Sie können sie durch beliebige eigene Audiodaten ersetzen: Applaus, Gelächter, Summer und Plings. Kopieren Sie sie in das Verzeichnis *soundboard* und passen Sie den Code so an, dass er auf sie zugreift. Wollen Sie mehr Klänge für Ihr Soundboard nutzen, fügen Sie weitere Taster hinzu und ändern Sie den Code entsprechend.

Weitere Informationen

RPi.GPIO (<http://code.google.com/p/raspberry-gpio-python/>)

Da die RPi.GPIO-Bibliothek immer noch aktiv entwickelt wird, sollten Sie auf der Homepage des Projekts nach den neuesten Updates Ausschau halten.

Using the MCP3008 ([http://learn.adafruit.com/](http://learn.adafruit.com/reading-a-analog-in-and-controlling-audio-volume-with-the-raspberry-pi/overview)

reading-a-analog-in-and-controlling-audio-volume-with-the-raspberry-pi/overview)

Adafruit besitzt ein ausgezeichnetes Tutorial zum Einsatz des MCP3008-Analog-Digital-Konverters, um Ihr Raspberry Pi-Projekt mit analogen Sensoren bestücken zu können.

9/Webcams einsetzen

Einer der Vorteile einer Plattform wie dem Raspberry Pi für Bastelprojekte ist, dass sie sehr viele verschiedene USB-Geräte unterstützt. Sie können nicht nur eine Tastatur und eine Maus anschließen, sondern auch Drucker, WLAN-Adapter, externe Festplatten, weitere Speicherkarten, Kameras und USB-Sticks. In diesem Kapitel werden wir ein paar Wege aufzeigen, wie Sie eine USB-Webcam bei Ihren Raspberry-Pi-Projekten einsetzen können.

Die Webcam ist zwar nicht so verbreitet wie Tastatur und Maus, aber moderne Computer besitzen sie fast schon als Standard. Zum Glück bedeutet das auch für uns, dass wir eine Webcam von einer bekannten Marke für 25 Euro bekommen können. Sie finden sogar Webcams für viel weniger Geld, wenn Sie eine eher unbekannte Marke nutzen. Und das Allerbeste ist: Viele USB-Webcams werden von Linux erkannt, ohne dass zusätzliche Treiber installiert werden müssen.

In Kapitel 1 ist Ihnen vielleicht aufgefallen, dass eine der Komponenten auf dem Board ein Anschluss für »The Camera Serial Interface« (CSI) ist (siehe Abbildung 9-1). Da der Broadcom-Chip auf dem Raspberry Pi für Mobiltelefone und Tablets gedacht ist, dient der CSI-Anschluss einem Hersteller mobiler Geräte dazu, eine Kamera anzuschließen. Leider sind CSI-Kameras für Endverbraucher normalerweise nicht so einfach erhältlich wie USB-Webcams. Aktuell entwickelt die Raspberry Pi Foundation eine Kamera, die den CSI-Anschluss auf dem Board nutzt. Bis diese Kamera zur Verfügung steht, empfehlen wir den Einsatz einer USB-Webcam (Abbildung 9-2).

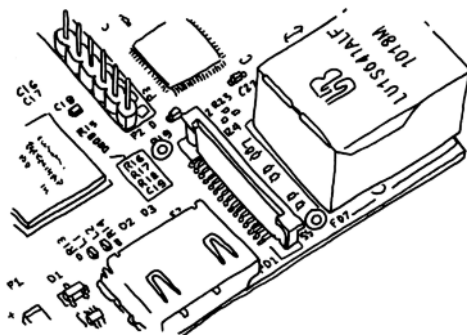


Abbildung 9-1. Das Camera Serial Interface des Raspberry Pi

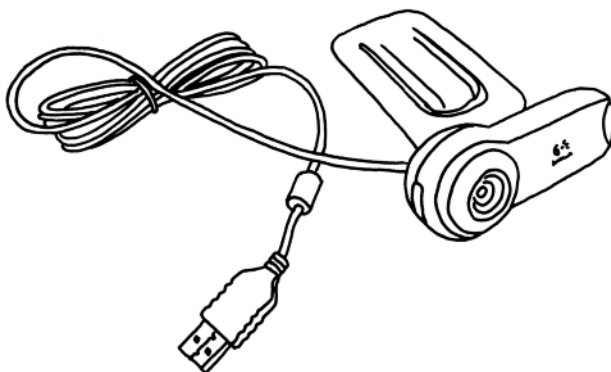


Abbildung 9-2. Eine typische USB-Webcam

Webcams testen

Bei all den verschiedenen Webcam-Modellen, die auf dem Markt sind, gibt es keine Garantie, dass eine Kamera ohne weiteres direkt funktioniert. Kaufen Sie eine Webcam für Ihren Raspberry Pi, schauen Sie online nach, ob andere schon mit dem entsprechenden Modell erfolgreich waren. Sie können auch den Webcam-Abschnitt der eLinux.org-Seite für Peripherie-Geräten (http://elinux.org/RPi_VerifiedPeripherals#USB_Webcams) nutzen, um Webcams zu finden, die mit dem Raspberry Pi zusammenarbeiten.

Beachten Sie, dass Sie einen USB-Hub mit eigener Stromversorgung an Ihren Raspberry Pi anschließen müssen, um Ihre Webcam neben Tastatur und Maus einsetzen zu können, da der Pi seine USB-Ports nur sehr eingeschränkt mit Strom versorgen kann. Ein USB-Hub mit eigener Stromversorgung stellt seinen angeschlossenen Geräten dagegen genug Strom zur Verfügung, so dass Sie nicht auf den Raspberry Pi zurückgreifen müssen.

Haben Sie eine Webcam, die Sie mit dem Raspberry Pi testen möchten, nutzen Sie `apt-get` im Terminal, um `luvcview` zu installieren – eine einfache Anwendung zum Anzeigen der Kameradaten:

```
pi@raspberrypi ~ $ sudo apt-get install luvcview
```

Nachdem `apt-get` die Installation abgeschlossen hat, rufen Sie die Anwendung auf, indem Sie in einem Terminal auf der Desktop-Oberfläche `luvcview` eingeben. Es öffnet sich ein Fenster, in dem die Daten der ersten Videoquelle angezeigt werden, die im Ordner `/dev` gefunden wird, wie zum Beispiel `/dev/video0`. Beachten Sie, dass die Frame-Größe im Terminal-Fenster ausgegeben wird. Wenn das Video ein wenig abgehackt erscheint, können Sie die Standard-Größe des Videos anpassen. Ist zum Beispiel die Standard-Videogröße `640x480`, schließen Sie `luvcview` und öffnen Sie es mit der halben Videogröße durch Eingabe folgenden Befehls:

```
pi@raspberrypi ~ $ luvcview -s 320x240
```

Erscheinen keine Videodaten, sollten Sie hier schon auf Fehlersuche gehen, bevor Sie weitermachen. Eine Möglichkeit, den Fehler herauszufinden, ist das Ab- und Wiederanstecken der Webcam und das Ausführen des Befehls `dmesg`, durch den Diagnosemeldungen ausgegeben werden, die vielleicht Hinweise geben können.

Installation und Test von SimpleCV

Um die Webcam per Python-Code ansprechen zu können, werden wir SimpleCV (Abbildung 9-3) nutzen, bei der es sich um eine umfangreiche Open-Source-Bibliothek zur Computer Vision handelt. Mit SimpleCV ist es sehr einfach, Bilder von der Webcam auszulesen und sie auf dem Bildschirm anzuzeigen oder als Datei abzuspeichern. Aber das Beste an SimpleCV sind seine Algorithmen zur Computer Vision, mit denen Sie ziemlich coole Dinge anstellen können. Neben grundlegenden Bildtransformationen können Sie auch Objekte in einem Bild oder auf einem Video finden, erkennen und verfolgen. Weiter unten in diesem Kapitel werden wir eine einfache Gesichtserkennung mit SimpleCV durchführen („Gesichtserkennung“ (S. 120)).

Um SimpleCV für Python zu installieren, müssen Sie erst die Bibliotheken installieren, auf denen es aufbaut. Dazu können Sie `apt-get` nutzen:

```
pi@raspberrypi ~ $ sudo apt-get install python-opencv python-scipy python-numpy python-pip
```



Abbildung 9-3. Logo von SimpleCV

Die Installation ist ziemlich umfangreich und es kann einige Zeit dauern, bis der Prozess abgeschlossen ist. Als Nächstes installieren Sie die eigentliche SimpleCV-Bibliothek:

```
pi@raspberrypi ~ $ sudo pip install https://github.com/ingenuitas/SimpleCV/  
zipball/master
```

Danach prüfen Sie, ob die Installation erfolgreich war, indem Sie den interaktiven Interpreter von Python aufrufen und die Bibliothek importieren:

```
pi@raspberrypi ~ $ python  
Python 2.7.3rc2 (default, May 6 2012, 20:02:25)  
[GCC 4.6.3] on linux2  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import SimpleCV  
>>>
```

Erhalten Sie beim Importieren der Bibliothek keine Fehler, können Sie mit Ihren Experimenten zur Computer Vision auf dem Raspberry Pi beginnen.

Ein Bild anzeigen

Bei vielen Beispielen in diesem Kapitel müssen Sie in der Desktop-Umgebung arbeiten, um Bilder auf dem Monitor anzeigen zu können. Sie können in IDLE arbeiten oder Ihren Code als .py-Dateien im Leafpad sichern und sie im Terminalfenster ausführen.

Wir werden ein paar SimpleCV-Grundlagen an Bilddateien demonstrieren und dann Bilder von der Webcam auslesen. Wenn Sie die Bilder von der Webcam haben, werden Sie die Gesichtserkennung ausprobieren können.

1. Erstellen Sie in Ihrem Home-Verzeichnis ein neues Verzeichnis namens *simplecv-test*.
2. Öffnen Sie Midori und suchen Sie nach für Sie interessanten Bildern. Wir haben uns dazu entschlossen, ein Bild mit Himbeeren von Wikipedia zu nehmen und es in *raspberries.jpg* umzubenennen.
3. Klicken Sie mit der rechten Maustaste auf das Bild und wählen Sie »Save Image«.
4. Sichern Sie das Bild im Ordner *simplecv-test*.
5. Öffnen Sie im File Manager (im Menü »Accessories«) den Ordner *simplecv-test* und klicken Sie mit der rechten Maustaste in den Ordner. Wählen Sie Create New → Blank File.
6. Geben Sie der Datei den Namen *image-display.py*.
7. Klicken Sie doppelt auf die neu erstellte Datei *image-display.py*, um sie in Leafpad zu öffnen.
8. Geben Sie den Code aus Beispiel 9-1 ein.
9. Sichern Sie die Datei *image-display.py* und führen Sie sie im Terminalfenster aus. Ist alles richtig eingegeben, sehen Sie ein Foto in einem neuen Fenster (siehe Abbildung 9-4). Sie können das Fenster selbst schließen oder im Terminal Strg-C drücken, um das Skript zu beenden.

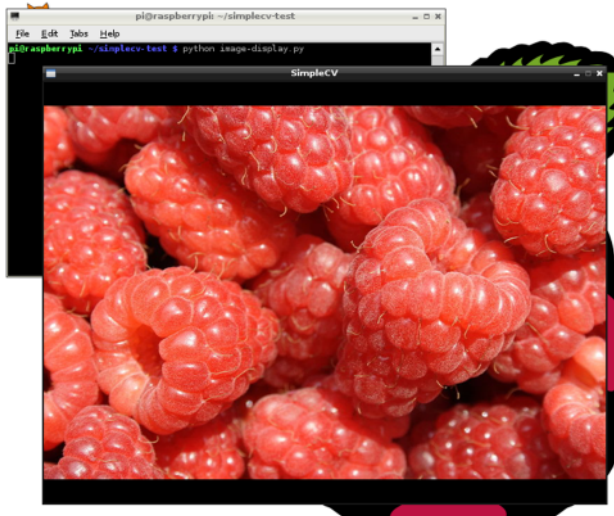


Abbildung 9-4. Das Himbeer-Foto in einem Fenster

Beispiel 9-1. Quellcode für *image-display.py*

```
from SimpleCV import Image, Display ❶
from time import sleep

myDisplay = Display() ❷

raspberryImage = Image("raspberries.jpg") ❸

raspberryImage.save(myDisplay) ❹

while not myDisplay.isDone(): ❺
    sleep(0.1)
```

- ❶ Importiert die Bild- und Anzeige-Funktionen von SimpleCV
- ❷ Erstellt ein neues Window-Objekt
- ❸ Lädt die Bilddatei *raspberries.jpg* als Objekt *image* in den Speicher
- ❹ Gibt das Bild im Fenster aus
- ❺ Verhindert, dass das Skript sich nach dem Anzeigen des Bildes direkt beendet

Ein Bild verändern

Nachdem Sie nun ein Bild in den Speicher laden und auf dem Monitor anzeigen können, wollen wir es vor der Anzeige ein wenig bearbeiten. Dabei wird nicht die Bilddatei selbst verändert, sondern nur die Kopie, die im Speicher vorliegt.

1. Sichern Sie die Datei *image-display.py* als Kopie unter dem neuen Namen *superimpose.py*.
2. Nehmen Sie die Erweiterungen im Code vor, die in Beispiel 9-2 zu sehen sind.
3. Sichern Sie die Datei und führen Sie sie an der Befehlszeile aus.
4. Sie sollten das gleiche Bild wie vorher sehen, dieses Mal aber überlagert mit dem Kasten und dem Text.

Beispiel 9-2. Quellcode für *superimpose.py*

```
from SimpleCV import Image, Display, DrawingLayer, Color ❶
from time import sleep

myDisplay = Display()

raspberryImage = Image("raspberries.jpg")

myDrawingLayer = DrawingLayer((raspberryImage.width, raspberryImage.height)) ❷
```

```

myDrawingLayer.rectangle((50,20), (250, 60), filled=True) ❸
myDrawingLayer.setFontSize(45)
myDrawingLayer.text("Raspberries!", (50, 20), color=Color.WHITE) ❹

raspberryImage.addDrawingLayer(myDrawingLayer) ❺
raspberryImage.applyLayers() ❻

raspberryImage.save(myDisplay)

while not myDisplay.isDone():
    sleep(0.1)

```

- ❶ Importiert neben den Bild- und Anzeigefunktionen aus dem vorigen Beispiel auch den Zeichen-Layer und Farbfunktionen von SimpleCV
- ❷ Erstellt einen neuen Zeichen-Layer mit der gleichen Größe wie das Bild
- ❸ Zeichnet auf dem Layer ein Rechteck mit den Koordinaten von 50, 20 bis 250, 60 und füllt es
- ❹ Schreibt auf dem Layer den Text »Raspberries!« an Position 50, 20 in weißer Farbe
- ❺ Fügt myDrawingLayer zu raspberryImage hinzu
- ❻ Verbindet die Layer, die raspberryImage hinzugefügt wurden (Abbildung 9-5 zeigt das neue Bild)

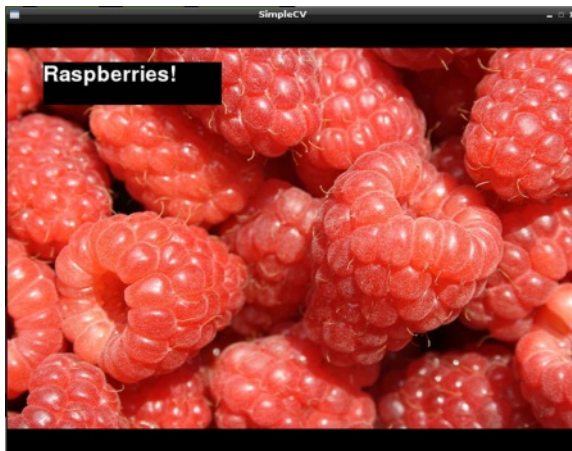


Abbildung 9-5. *Das veränderte Himbeer-Foto*

Statt das Bild auf dem Monitor anzuzeigen, können Sie Ihre Änderungen auch in einer Datei speichern. Beispiel 9-3 zeigt, wie der Code dafür aussehen kann.

Beispiel 9-3. Quellcode für *superimpose-save.py*

```
from SimpleCV import Image, DrawingLayer, Color
from time import sleep

raspberryImage = Image("raspberries.jpg")

myDrawingLayer = DrawingLayer((raspberryImage.width, raspberryImage.height))

myDrawingLayer.rectangle((50,20), (250, 60), filled=True)
myDrawingLayer.setFontSize(45)
myDrawingLayer.text("Raspberries!", (50, 20), color=Color.WHITE)

raspberryImage.addDrawingLayer(myDrawingLayer)
raspberryImage.applyLayers()

raspberryImage.save("raspberries-titled.jpg") ❶
```

❶ Speichert das veränderte Bild unter dem Namen *raspberries-titled.jpg*

Da dieser Code kein Fenster öffnet, können Sie ihn auch an der Befehlszeile ohne Desktop-Umgebung öffnen. Sie können den Code sogar so anpassen, dass Sie mit einem einzelnen Befehl sehr viele Bilder zum Beispiel mit einem Wasserzeichen versehen.

Und Sie sind nicht auf Text und Rechtecke beschränkt. Hier ein paar weitere Zeichenfunktionen, die Ihnen mit SimpleCV zur Verfügung stehen. Die vollständige Dokumentation finden Sie unter <http://simplecv.org/doc/SimpleCV.html#i/SimpleCV.DrawingLayer.DrawingLayer>

- Kreis
- Ellipse
- Linie
- Polygon
- Bezierkurve

Zugriff auf die Webcam

Glücklicherweise ist es nicht komplizierter, den Videostream einer Webcam in SimpleCV zu bringen, als Bilddateien zu laden. Versuchen Sie es – Sie können Ihren eigenen kleinen Webcam-Viewer schreiben.

1. Erstellen Sie eine neue Datei namens *basic-camera.py* und sichern Sie den Code aus Beispiel 9-4 darin.
2. Führen Sie das Skript mit angeschlossener Webcam aus. Es sollte ein Fenster erscheinen, in dem das zu sehen ist, was die Webcam aufnimmt (siehe Abbildung 9-6).
3. Um das Fenster zu schließen, drücken Sie im Terminal Strg-C.

Beispiel 9-4. Quellcode für basic-camera.py

```
from SimpleCV import Camera, Display
from time import sleep

myCamera = Camera(prop_set={'width': 320, 'height': 240}) ❶

myDisplay = Display(resolution=(320, 240)) ❷

while not myDisplay.isDone(): ❸
    myCamera.getImage().save(myDisplay) ❹
    sleep(.1) ❺
```

- ❶ Erstellt ein neues Camera-Objekt und setzt Höhe und Breite des Bildes aus Performance-Gründen auf 320x240
- ❷ Setzt die Größe des Fensters ebenfalls auf 320x240
- ❸ Durchläuft den eingerückten Code, bis das Fenster geschlossen wird
- ❹ Holt einen Frame von der Webcam und zeigt ihn im Fenster an
- ❺ Wartet eine zehntel Sekunde zwischen jedem Frame

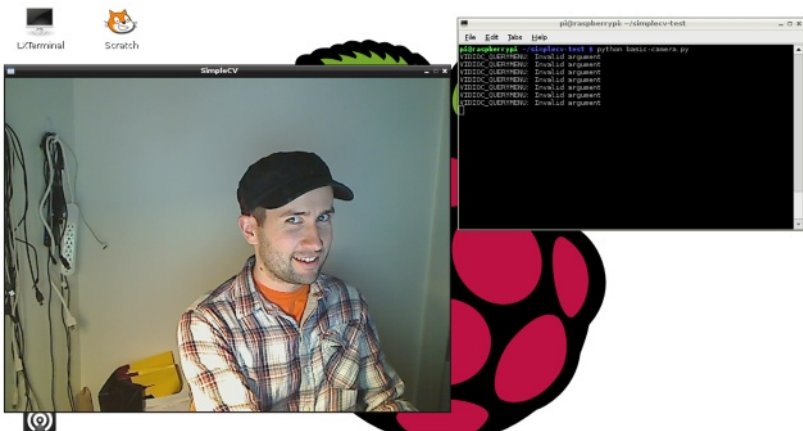


Abbildung 9-6. Ausgabe der Webcam-Aufnahme am Monitor

Sie können auch den Code aus den letzten beiden Beispielen kombinieren und ein Python-Skript erstellen, mit dem ein Bild von der Webcam geholt und als *.jpg*-Datei gesichert wird:

```
from SimpleCV import Camera
from time import sleep

myCamera = Camera(prop_set={'width':320, 'height': 240})

frame = myCamera.getImage()
frame.save("camera-output.jpg")
```

Gesichtserkennung

Eine der leistungsfähigen Funktionen, die SimpleCV mitbringt, nennt sich `findHaarFeatures`. Dabei handelt es sich um einen Algorithmus, mit dem Sie in einem Bild nach Mustern suchen können, die einem bestimmten Profil oder einer *Cascade* entsprechen. SimpleCV bringt schon ein paar Cascades mit, wie zum Beispiel Gesicht, Nase, Auge, Mund und ganzer Körper. Alternativ können Sie auch bei Bedarf Ihre eigene Cascade-Datei erzeugen. `findHaarFeatures` analysiert ein Bild auf Übereinstimmungen. Findet es mindestens eine, gibt die Funktion die Position dieser Übereinstimmungen im Bild zurück. Sie können also Objekte wie Autos, Tiere oder Personen in Bilddateien oder den Daten einer Webcam finden. Zum Ausprobieren von `findHaarFeatures` versuchen Sie es doch einmal mit einer einfachen Gesichtserkennung:

1. Erstellen Sie im Verzeichnis *simplecv-test* eine neue Datei mit dem Namen *face-detector.py*.
2. Geben Sie den Code aus Beispiel 9-5 ein.
3. Führen Sie das Skript an der Befehlszeile aus, während die Webcam angeschlossen ist und auf Gesichter zeigt.
4. Im Terminalfenster sehen Sie nun die Position der Gesichter, die `findHaarFeatures` findet. Bewegen Sie sich und beobachten Sie, wie sich diese Werte ändern. Versuchen Sie, ein Foto mit einem Gesicht in die Kamera zu halten, um herauszufinden, was dann passiert.

Beispiel 9-5. Quellcode für *face-detector.py*

```
from SimpleCV import Camera, Display
from time import sleep

myCamera = Camera(prop_set={'width':320, 'height': 240})

myDisplay = Display(resolution=(320, 240))

while not myDisplay.isDone():
    frame = myCamera.getImage()
    faces = frame.findHaarFeatures('face') ❶
    if faces: ❷
        for face in faces: ❸
            print "Gesicht bei: " + str(face.coordinates())
    else:
        print "Keine Gesichter erkannt."
    frame.save(myDisplay)
    sleep(.1)
```

- ❶ Sucht im Bildframe nach Gesichtern und speichert sie in einem `faces`-Objekt

- ② Wenn findHaarFeatures mindestens ein Gesicht erkennt, wird der folgende, eingerückte Code ausgeführt
- ③ Für jedes face in faces wird der folgende Code (die print-Anweisung) mit dem jeweiligen face ausgeführt

Wenn Sie sich auf dem Bildschirm sehen, aber trotzdem die Nachricht »Keine Gesichter erkannt« erhalten, versuchen Sie es mit ein paar möglichen Schritten zur Fehlerbehebung:

- Haben Sie genug Licht? Wenn Sie sich in einem dunklen Raum befinden, kann es für den Algorithmus schwer sein, Ihr Gesicht zu erkennen. Versuchen Sie es dann mit mehr Licht.
- Diese spezielle Haar-Cascade ist dazu gedacht, Gesichter zu finden, die normal ausgerichtet sind. Drehen Sie Ihren Kopf zu sehr oder ist die Kamera schräg angebracht, wird damit die Fähigkeit des Algorithmus zur Gesichtserkennung eingeschränkt.

Projekt: Raspberry Pi Photobooth

Sie können verschiedene Bibliotheken so kombinieren, dass Python ein mächtiges Tool für ziemlich komplexe Projekte wird. Mit der GPIO-Bibliothek, die Sie in Kapitel 8 kennengelernt haben, und SimpleCV können Sie Ihr eigenes Photobooth auf dem Raspberry Pi aufsetzen, das auf Ihrer nächsten Party bestimmt ein großer Hit wird (siehe Abbildung 9-7). Und mit der Funktion findHaarFeatures von SimpleCV können Sie Ihr Photobooth mit einem zusätzlichen Feature ausstatten – dem automatischen Überlagern von Gesichtern mit lustigen virtuellen Objekten wie Hüten, Monokeln und Bärten. Der Code dieses Projekts baut auf dem Mustacheinator-Projekt in Practical Computer Vision with SimpleCV (<http://shop.oreilly.com/product/0636920024057.do>) von Kurt Demaagd, Anthony Oliver, Nathan Oostendorp und Katherine Scott auf.



Abbildung 9-7. Ausgabe des Raspberry Pi Photobooth

Das brauchen Sie, um Ihren Raspberry Pi in einen Photobooth zu verwandeln:

- Eine USB-Webcam
- Einen Monitor
- Einen Taster – beliebiger Art
- Verbindungskabel in der gewünschten Länge
- 1 Widerstand, 10 K Ω

Bevor Sie loslegen, stellen Sie sicher, dass die Python-Bibliotheken RPi.GPIO und SimpleCV auf Ihrem Raspberry Pi installiert sind und auch funktionieren. In GPIO in Python installieren und testen (S. 99) und Installation und Test von SimpleCV (S. 113) finden Sie weitere Details.

1. Wie in Kapitel 8 verbinden Sie Pin 24 mit dem Taster. Eine Seite des Tasters sollte mit 3,3 V verbunden sein, die andere mit Pin 24. Vergessen Sie nicht den 10-K Ω -Pull-down-Widerstand zwischen Masse und der Seite des Tasters, der mit Pin 24 verbunden ist.
2. Finden Sie ein kleines Bild eines schwarzen Schnurrbarts auf einem weißen Hintergrund oder erstellen Sie eines und sichern Sie es als *mu-stache.png* in einem neuen Ordner auf Ihrem Raspberry Pi. Sie können auch unsere fertige Schnurrbart-Datei im Unterverzeichnis ch09 der Download-Daten nutzen (siehe „Wie Sie uns erreichen“ (S. xiv)).
3. Erstellen Sie in diesem Ordner eine neue Datei namens *photobooth.py*, geben Sie dort den Code aus Beispiel 9-6 ein und speichern Sie die Datei.

Beispiel 9-6. Quellcode für *photobooth.py*

```
from time import sleep, time
from SimpleCV import Camera, Image, Display
import RPi.GPIO as GPIO

myCamera = Camera(prop_set={'width':320, 'height': 240})
myDisplay = Display(resolution=(320, 240))
stache = Image("mustache.png")
stacheMask = \
    stache.createBinaryMask(color1=(0,0,0), color2=(254,254,254)) ❶
stacheMask = stacheMask.invert() ❷

GPIO.setmode(GPIO.BCM)
GPIO.setup(24, GPIO.IN)
```

```

def mustachify(frame): ❸
    faces = frame.findHaarFeatures('face')
    if faces:
        for face in faces:
            print "Gesicht bei: " + str(face.coordinates())
            myFace = face.crop() ❹
            noses = myFace.findHaarFeatures('nose')
            if noses:
                nose = noses.sortArea()[-1] ❺
                print "Nase bei: " + str(nose.coordinates())
                xmust = face.points[0][0] + nose.x - (stache.width/2) ❻
                ymust = face.points[0][1] + nose.y + (stache.height/3) ❼
            else:
                return frame ❽
            frame = frame.blit(stache, pos=(xmust, ymust), mask=stacheMask) ❾
        return frame ❿
    else:
        return frame ⓫

while not myDisplay.isDone():
    inputValue = GPIO.input(24)
    frame = myCamera.getImage()
    if inputValue == True:
        frame = mustachify(frame) ❿
        frame.save("mustache-" + str(time()) + ".jpg") ⓫
        frame = frame.flipHorizontal() ⓬
        frame.show()
        sleep(3) ⓭
    else:
        frame = frame.flipHorizontal() ⓬
        frame.save(myDisplay)
        sleep(.05)

```

- ❶ Erstellt eine Maske mit dem Schnurrbart, wobei alles, was nicht schwarz ist, transparent gemacht wird (die beiden Parameter color1 und color2 sind die Farben als RGB-Werte von 0 bis 255)
- ❷ Invertiert die Maske, so dass nur die schwarzen Pixel des Bildes angezeigt und alle anderen Pixel transparent werden
- ❸ Erstellt eine Funktion, die einen Frame übernimmt und einen Frame mit einem darübergelegten Schnurrbart zurückgibt, wenn sie Gesicht und Nase findet
- ❹ Erstellt ein Subimage des Gesichts, so dass die Suche nach einer Nase schneller abläuft
- ❺ Gibt es mehrere Kandidaten für eine Nase im Gesicht, wird die größte ausgewählt
- ❻ Setze die x-Koordinate des Barts
- ❼ Setze die y-Koordinate des Barts
- ❽ Wird keine Nase gefunden, wird nur der Frame zurückgegeben

- ⑨ Mit der blit-Funktion (Kurzform von BLock Image Transfer) wird der Schnurrbart über den Frame gelegt
- ⑩ Der »verschnurrbartete« Frame wird zurückgegeben
- ⑪ Wird kein Gesicht gefunden, wird nur der Frame zurückgegeben
- ⑫ Übergib den Frame an die Funktion `mustachify`
- ⑬ Sichere den Frame als JPG mit dem aktuellen Zeitstempel im Dateinamen
- ⑭ Vor der Anzeige des Bildes spiegele es entlang der vertikalen Achse, um es als Spiegelbild auszugeben
- ⑮ Zeige das gesicherte Bild für drei Sekunden auf dem Bildschirm an
- ⑯ Ist der Taster nicht gedrückt, spiegele das Live-Bild einfach und zeige es an

Jetzt können Sie das Ganze ausprobieren. Achten Sie darauf, dass Ihre Webcam angeschlossen ist. Dann wechseln Sie zum Terminal und rufen das Verzeichnis auf, in dem sich das Schnurrbart-Bild und *photobooth.py* befinden, und führen das Skript aus:

```
pi@raspberrypi ~ $ sudo photobooth.py
```

Die Ausgabe der Webcam wird auf dem Bildschirm erscheinen. Wird der Taster gedrückt, werden Gesichter erkannt, ein Schnurrbart wird hinzugefügt und das Bild gespeichert. (Alle Bilder landen in dem gleichen Ordner wie das Skript.)

Weitere Informationen

Practical Computer Vision with SimpleCV

(<http://shop.oreilly.com/product/0636920024057.do>)

Dieses Buch von Kurt Demaagd, Anthony Oliver, Nathan Oostendorp und Katherine Scott ist eine umfassende Anleitung zum Einsatz von SimpleCV. Es enthält viel Beispiel-Code und Projekte, um mehr über die Arbeit mit Bildern und Computer Vision in Python zu erlernen.

10/Python und das Internet

Python besitzt eine sehr aktive Entwickler-Community, die ihre Arbeit häufig als Open-Source-Bibliotheken zum Vereinfachen komplexer Aufgaben bereitstellt. Manche dieser Bibliotheken erleichtern uns den Anschluss unserer Projekte an das Internet, um zum Beispiel Daten über das Wetter zu ermitteln, eine E-Mail oder SMS zu verschicken, Trends auf Twitter zu folgen oder als Webserver zu dienen.

In diesem Kapitel werden wir uns ein paar Möglichkeiten für Internet-Projekte mit dem Raspberry Pi anschauen. Wir beginnen mit dem Lesen von Daten aus dem Internet, dann werden wir Ihnen zeigen, wie Sie Ihren eigenen Webserver auf dem Raspberry Pi erstellen.

Daten von einem Webserver laden

Wenn Sie eine Adresse in Ihren Webbrowser eingeben und die Eingabetaste drücken, ist Ihr Browser der *Client*. Er stellt eine Verbindung zum *Server* her, der mit einer Webseite reagiert. Natürlich muss ein Client kein Webbrowser sein, es kann sich auch um eine Mail-Anwendung, ein Wetter-Widget auf Ihrem Telefon oder Computer oder ein Spiel handeln, das Ihren Highscore an eine zentrale Bestenliste übermittelt. Im ersten Teil dieses Kapitels werden wir uns auf Projekte konzentrieren, in denen der Raspberry Pi als Client agiert. Der Code verbindet sich mit Internet-Servern, um Informationen zu erhalten. Zuvor müssen Sie aber noch eine beliebte Python-Bibliothek namens *Requests* installieren, mit der die Verbindungen zu den Webservern über das *Hypertext Transfer Protocol* (HTTP) hergestellt wird. So installieren Sie die Bibliothek über das Terminal:

```
pi@raspberrypi ~ $ sudo apt-get install python-requests
```

Um sicherzugehen, dass Requests installiert ist:

```
pi@raspberrypi ~ $ python
Python 2.7.3rc2 (default, May  6 2012, 20:02:25)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import requests
>>>
```

Erhalten Sie keine Fehlermeldung, wissen Sie, dass Requests sauber installiert und nun in diese Python-Session importiert wurde. Probieren Sie es jetzt aus:

```
>>> r = requests.get('http://www.google.com/')
>>>
```

Sie sind vielleicht zunächst etwas enttäuscht, weil nichts passiert zu sein scheint. Aber tatsächlich wurden alle Daten des Requests im Objekt `r` gespeichert. So können Sie den Statuscode ausgeben:

```
>>> r.status_code
200
```

Der HTTP-Statuscode 200 bedeutet, dass die Anfrage erfolgreich war. Es gibt noch ein paar weitere HTTP-Statuscodes, von denen die wichtigsten in Tabelle 10-1 zu finden sind.

Tabelle 10-1. Häufige HTTP-Statuscodes

Code	Bedeutung
200	OK
301	Dauerhaft verschoben
307	Zeitweise verschoben
401	Nicht autorisiert
404	Nicht gefunden
500	Server-Fehler

Wollen Sie den Inhalt der *Response* (was der Server Ihnen zurückgeschickt hat) ausgeben, nutzen Sie diese Anweisung:

```
>>> r.text
```

Hat alles wie gewünscht funktioniert, wird eine ganze Menge Text ausgegeben. Sie werden teilweise etwas lesen können, aber das meiste wird nur schwer verständlich sein. Dies ist der HTML-Code der Startseite von Google – normalerweise von einem Webbrowser interpretiert und gerendert.

Aber nicht alle HTTP-Requests sind dazu gedacht, von einem Webbrowser gerendert zu werden. Manchmal werden nur Daten übertragen – ohne weitere Informationen zur gewünschten Anzeigeform. Viele Sites stellen diese Datenprotokolle der Öffentlichkeit zur Verfügung, so dass wir mit ihnen Daten austauschen können, ohne einen Browser nutzen zu müssen. Egal ob sie öffentlich ist oder nicht – eine Datenprotokoll-Spezifikation wird meist als *Application Programming Interface* oder API bezeichnet. Mit APIs können verschiedene Software-Elemente miteinander kommunizieren und man kann mit ihnen Daten über das Internet von einer Site zu einer anderen übertragen.

Nehmen wir zum Beispiel an, dass Sie ein Projekt aufbauen wollen, das vor Ihrer Tür angebracht ist und Sie daran erinnert, einen Schirm mitzunehmen, wenn für den Tag Regen angesagt ist. Statt Ihre eigene Wetterstation einzurichten und herauszufinden, wie man Regen voraussagt, können Sie die Vorhersage von einer der vielen Wetter-APIs abholen, die zur Verfügung stehen.

Die Wettervorhersage abfragen

Um herauszufinden, ob es heute regnen wird oder nicht, werden wir Ihnen zeigen, wie Sie die API von Weather Underground (<http://www.wunderground.com/>) nutzen.



Denken Sie daran, dass nicht alle APIs gleich aufgebaut sind. Sie müssen sich die zugehörige Dokumentation anschauen, um herauszufinden, ob die API für Ihr Projekt passt. Zudem beschränken die meisten APIs die Anzahl an Anfragen, die Sie stellen dürfen – manche kosten sogar Geld. Häufig sind eine gewisse Zahl an Anfragen pro Tag kostenlos, was zum Experimentieren und für den persönlichen Gebrauch ausreicht.

1. Rufen Sie in einem Webbrowser die Homepage von Weather Underground (<http://www.wunderground.com/weather/api/>) auf und geben Sie Ihre Informationen an, um sich anzumelden.
2. Nach dem Anmelden mit Ihrem Account klicken Sie auf Key Settings.
3. Im Feld »Key ID« finden Sie einen langen String (siehe Abbildung 10-1). Dabei handelt es sich um eine eindeutige Kennung für Ihren Account, den Sie in jedem Request mitliefern müssen. Missbrauchen Sie den Server von Weather Underground, indem Sie zu viele Requests schicken, kann die Firma diesen API Key einfach abschalten und weitere Requests nicht beantworten, bis Sie für mehr Anfragemöglichkeiten zahlen.
4. Klicken Sie auf »Documentation« und dann auf »Forecast«, um herauszufinden, welche Daten Sie erhalten, wenn Sie einen Forecast Request abschicken. Am Fuß der Seite steht eine Beispiel-URL für die Vorhersage für San Francisco. In der URL muss sich Ihr API Key wiederfinden:

http://api.wunderground.com/api/YourAPIKey/forecast/q/CA/San_Francisco.json

The screenshot shows the Weather Underground API account page. The header includes the Wunderground logo and navigation links: Analytica, Key Settings, Documentation, Community, Sign Out, and a Twitter icon. The main content area is titled "Your Subscription: Status Developer" and shows a subscription status of "7b448398fcac51cc - Getting Started with Raspberry Pi". Below this, there's a section for "THE PERFECT DATA STORM" with a "STEP 1: Choose the plan that suits your needs." button. The "STRATUS PLAN" is selected, showing options like Geolookup, Autocomplete, Current conditions, 3-day forecast summary, Astronomy, and Almanac for today. The "CUMULUS PLAN" is also visible with a "10-day forecast summary" option. On the right, there's an "Edit API Key" form with fields for Key ID, Project Name, Company Website, Contact Phone, and Contact Email. The "Update" button is at the bottom of the form.

Abbildung 10-1. Der Weather Underground API Account von Matt mit dem API Key oben rechts

5. Um sicherzugehen, dass alles funktioniert, rufen Sie diese URL in Ihrem Webbrowser auf. Sie sollten die Vorhersagedaten in einem Format namens *JSON* (JavaScript Object Notation) sehen (Beispiel 10-1). Beachten Sie, wie die Daten hierarchisch strukturiert sind.



Auch wenn das J für JavaScript steht, wird JSON in vielen Programmiersprachen genutzt, insbesondere bei der Kommunikation über eine API.

6. Versuchen Sie nun in einem ersten Schritt, die lokale Voraussage in Textform aus dieser Datenstruktur auszulesen. Ändern Sie die URL so, dass Sie zu Ihrem Land und Ihrer Stadt passt, und stecken Sie sie in ein neues Python-Skript namens *text-forecast.py* (Beispiel 10-2).
7. Wie Sie im Code sehen, müssen wir das richtige Feld aus der hierarchischen Datenstruktur auslesen (siehe Beispiel 10-1). Die Text-Voraussage finden Sie in `forecast` → `txt_forecast` → `forecastday` → `0` (der erste Eintrag, bei dem es sich um die heutige Voraussage handelt) → `fct-text_metric`
8. Lassen Sie dieses Skript an der Befehlszeile laufen, sollte die Ausgabe Ihre lokale Wettervorhersage für heute enthalten, wie zum Beispiel: »Overcast in the morning, then mostly cloudy. High of 6C. Winds less than 5 km/h.«

Beispiel 10-1. Teil der JSON-Response von der Weather Underground-API

```
"response": {
  "version": "0.1",
  "termsOfService":
    "http://www.wunderground.com/weather/api/d/terms.html",
  "features": {
    "forecast": 1
  }
},
"forecast":{ ❶
  "txt_forecast": { ❷
    "date":"1:00 AM CET",
    "forecastday": [ ❸
      {
        "period":0,
        "icon":"mostlycloudy",
        "icon_url":"http://icons-ak.wxug.com/i/c/k/mostlycloudy.gif",
        "title":"Sunday",
        "fcttext":
          "Overcast in the morning, then mostly cloudy. High of 43F.
           Winds less than 5 mph.",
        "fcttext_metric":
          "Overcast in the morning, then mostly cloudy. High of 6C.
           Winds less than 5 km/h.", ❹
        "pop":"0"
      }
    ],
  }
}
```

- ❶ forecast ist das oberste Eltern-Element der Daten, die wir ansprechen wollen
- ❷ Innerhalb von forecast wollen wir die Text-Vorhersage haben
- ❸ In der Text-Vorhersage brauchen wir die täglichen Vorhersagen
- ❹ fcttext_metric, der Text der aktuellen Vorhersage

Beispiel 10-2. Quellcode für text-forecast.py

```
import requests
```

```
key = 'HIER IHR KEY' ❶
```

```
ApiUrl = \
```

```
    'http://api.wunderground.com/api/' + key + '/forecast/q/DE/Berlin.json'
```

```
r = requests.get(ApiUrl) ❷
```

```
forecast = r.json ❸
```

```
print forecast['forecast']['txt_forecast']['forecastday'][0]['fcttext_metric'] ❹
```

- ❶ Tragen Sie hier Ihren API Key ein
- ❷ Die Vorhersage von Weather Underground für Berlin holen (setzen Sie Ihr eigenes Land und Ihre gewünschte Stadt ein)
- ❸ Den Text aus der JSON-Response holen und ihn in ein Python-Dictionary-Objekt parsen
- ❹ In die Daten-Hierarchie greifen, um die Vorhersage für heute zu erhalten

Jetzt haben Sie ein Python-Skript, das die Text-Vorhersage bei Bedarf ausgibt. Aber wie kann Ihr Raspberry Pi herausfinden, ob es heute regnen soll könnte oder nicht? Sie könnten natürlich den Vorhersage-Text nach Wörtern wie »rain«, »drizzle«, »thunderstorms«, »showers« und so weiter parsen, aber es gibt eine bessere Möglichkeit. Eines der Felder in den Daten der Weather Underground API hat den Namen *pop*, was für *Probability of Precipitation* steht. Mit Werten zwischen 0 und 100% können Sie abschätzen, wie wahrscheinlich es ist, dass es regnet oder schneit.

Für einen Regen-Voraussage-Warner wollen wir davon ausgehen, dass jede Niederschlagswahrscheinlichkeit größer als 30% einen Ratschlag beinhalten sollte, den Regenschirm mitzunehmen.

1. Schließen Sie eine LED wie in Abbildung 7-4 an Pin 25 an.
2. Erstellen Sie eine neue Datei mit dem Namen *umbrella-indicator.py* und verwenden Sie den Code in Beispiel 10-3. Vergessen Sie nicht, Ihren eigenen API Key und den gewünschten Standort in die API-URL einzutragen.
3. Führen Sie das Skript mit der Anweisung
`sudo python umbrella-indicator.py` als Root aus.

Beispiel 10-3. Quellcode für *umbrella-indicator.py*

```
import requests
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(25, GPIO.OUT)

key = 'HIER IHR KEY' ❶
ApiUrl = \
    'http://api.wunderground.com/api/' + key + '/forecast/q/DE/Berlin.json'

while True:
    r = requests.get(ApiUrl)
    forecast = r.json
    popValue = forecast['forecast']['txt_forecast']['forecastday'][0]['pop'] ❷
    popValue = int(popValue) ❸

    if popValue >= 30: ❹
        GPIO.output(25, GPIO.HIGH)
    else: ❺
        GPIO.output(25, GPIO.LOW)

    time.sleep(180) # 3 Minuten ❻
```

- ❶ Wie schon oben tragen Sie hier Ihren API Key ein
- ❷ Die Niederschlagswahrscheinlichkeit für heute auslesen und in *popValue* ablegen

- ❸ popValue aus einem String in eine Zahl umwandeln, um sie besser vergleichen zu können
- ❹ Ist der Wert größer oder gleich 30, wird die LED angeschaltet
- ❺ Ansonsten wird die LED ausgeschaltet
- ❻ Drei Minuten bis zur nächsten Abfrage warten, damit das Skript die API-Grenze von 500 Abfragen pro Tag nicht überschreitet

Drücken Sie Strg-C, um das Programm zu beenden, wenn Sie fertig sind.

Die API von Weather Underground ist eine der vielen verschiedenen APIs, mit denen Sie experimentieren können. In Tabelle 10-2 finden Sie ein paar weitere Sites und Services, die APIs anbieten.

Tabelle 10-2. *Beliebte Application Programming Interfaces*

Site	API-Referenz-URL
Facebook	https://developers.facebook.com/
Flickr	http://www.flickr.com/services/api/
Foursquare	https://developer.foursquare.com/
Reddit	https://github.com/reddit/reddit/wiki/API
Twilio	http://www.twilio.com/
Twitter	https://dev.twitter.com/
YouTube	https://developers.google.com/youtube/

Serving Pi (ein Webserver sein)

Sie können mit dem Raspberry Pi nicht nur Daten über das Internet von Servern auslesen – Ihr Pi kann auch selbst als Server dienen. Es gibt viele verschiedene Webserver, die Sie auf dem Raspberry Pi installieren können. Klassische Webserver, wie Apache oder lighthttpd, stellen Clients Dateien von Ihrem Pi bereit. Meistens verschicken solche Server HTML-Dateien und Bilder, aus denen Webseiten aufgebaut werden, aber sie können auch Sound- und Videodateien, ausführbare Programme und vieles mehr anbieten.

Es gibt aber auch eine neue Art von Tools, die Programmiersprachen wie Python, Ruby und JavaScript so erweitern, dass Webserver entstehen, die den HTML-Code dynamisch erstellen, wenn sie HTTP-Requests von einem Webbrowser erhalten. Damit lassen sich wunderbar physikalische Ereignisse auslösen, Daten ablegen oder der Wert eines Sensors aus der Ferne über einen Webbrowser ermitteln. Sie können sogar Ihre eigene JSON-API für ein Elektronikprojekt schreiben!

Grundlagen zu Flask

Wir werden ein Python-Webframework namens Flask (<http://flask.pocoo.org/>) nutzen, um den Raspberry Pi in einen dynamischen Webserver zu verwandeln. Sie können mit Flask schon »out of the box« sehr viel erreichen, aber es unterstützt eine Reihe von Erweiterungen für die Benutzer-Authentifizierung, das Erstellen von Formularen und den Zugriff auf Datenbanken. Sie können zudem auch all die Standard-Python-Bibliotheken einsetzen, die im Netz verfügbar sind.

Um Flask zu installieren, muss pip installiert sein. Ist pip noch nicht installiert, lässt sich das leicht nachholen:

```
pi@raspberrypi ~ $ sudo apt-get install python-pip
```

Nach der Installation von pip können Sie damit Flask und seine abhängigen Module installieren:

```
pi@raspberrypi ~ $ sudo pip install flask
```

Um die Installation zu testen, erstellen Sie eine neue Datei namens *hello-flask.py* mit dem Code aus Beispiel 10-4. Machen Sie sich keine Sorgen, wenn Sie den Code noch unverständlich finden. Der wichtigste Teil des Codes ist der mit dem String »Hallo Welt!«

Beispiel 10-4. Source code for *hello-flask.py*

```
from flask import Flask
app = Flask(__name__) ❶

@app.route("/") ❷
def hello():
    return "Hallo Welt!" ❸

if __name__ == "__main__": ❹
    app.run(host='0.0.0.0', port=80, debug=True) ❺
```

- ❶ Ein Flask-Objekt mit dem Namen app erstellen
- ❷ Den folgenden Code ausführen, wenn jemand die Root-URL des Servers anfordert
- ❸ Den Text »Hallo Welt!« an den Client schicken
- ❹ Wenn das Skript direkt von der Befehlszeile aus aufgerufen wurde ...
- ❺ ... den Server an Port 80 lauschen lassen und Fehler protokollieren



Bevor Sie das Skript laufen lassen, müssen Sie die IP-Adresse Ihres Raspberry Pi kennen (siehe „Das Netzwerk“ (S. 31)). Alternativ können Sie `avahi-daemon` installieren (rufen Sie `sudo apt-get install avahi-daemon` an der Befehlszeile auf). Damit können Sie den Pi in Ihrem lokalen Netzwerk über die Adresse `http://raspberrypi.local` ansprechen. Greifen Sie auf den Webserver auf dem Raspberry Pi über einen Windows-Rechner zu, müssen Sie dort eventuell die Bonjour Services (<http://support.apple.com/kb/DL999>) installieren, damit alles funktioniert.

Jetzt können Sie den Server laufen lassen, was Sie als Root tun müssen:

```
pi@raspberrypi ~ $ sudo python hello-flask.py
* Running on http://0.0.0.0:80/
* Restarting with reloader
```

Geben Sie an einem anderen Computer, der sich im gleichen Netzwerk wie der Raspberry Pi befindet, die IP-Adresse des Pi in einen Webbrowser ein. Zeigt Ihr Browser »Hallo Welt!« an, wissen Sie, dass Sie ihn richtig konfiguriert haben. Vielleicht sind Ihnen auch die zusätzlichen Zeilen im Terminalfenster des Raspberry Pi aufgefallen:

```
10.0.1.100 - - [19/Nov/2012 00:31:31] "GET / HTTP/1.1" 200 -
10.0.1.100 - - [19/Nov/2012 00:31:31] "GET /favicon.ico HTTP/1.1" 404 -
```

In der ersten Zeile steht, dass der Webbrowser die Root-URL angefordert und unser Server den HTTP-Statuscode 200 für »OK« zurückgeliefert hat. Die zweite Zeile ist ein Request, den viele Webbrowser automatisch mitschicken, um ein kleines Symbol als *Favicon* zu erhalten, das sie neben der URL in der Adresszeile des Browsers anzeigen können. Unser Server besitzt keine Datei *favicon.ico*, daher wurde der Statuscode 404 zurückgegeben, der angibt, dass die URL nicht gefunden wurde.

Wollen Sie dem Browser eine Seite schicken, die in sauberem HTML formatiert ist, müssen Sie nicht den ganzen HTML-Code in Ihrem Python-Skript selbst schreiben. Flask nutzt eine Template Engine namens Jinja2 (<http://jinja.pocoo.org/docs/templates/>), so dass Sie separate HTML-Dateien mit Platzhaltern verwenden können, in die dynamische Daten eingefügt werden.

Läuft bei Ihnen immer noch *hello-flask.py*, drücken Sie Strg-C, um es abubrechen.

Um die Templates auszuprobieren, erstellen Sie eine neue Datei mit dem Namen *hello-template.py* und fügen den Code aus Beispiel 10-5 ein. Im Verzeichnis mit *hello-template.py* erstellen Sie ein Unterverzeichnis namens *templates*. In diesem Unterverzeichnis erstellen Sie eine Datei *main.html*

und fügen den Code aus Beispiel 10-6 ein. Alle Teile in doppelten geschweiften Klammern im HTML-Template werden als Variable interpretiert, die vom Python-Skript über die Funktion `render_template` übergeben werden.

Beispiel 10-5. Quellcode für *hello-template.py*

```
from flask import Flask, render_template
import datetime
app = Flask(__name__)

@app.route("/")
def hello():
    now = datetime.datetime.now() ❶
    timeString = now.strftime("%d.%m.%Y %H:%M") ❷
    templateData = {
        'title' : 'HALLO!',
        'time': timeString
    } ❸
    return render_template('main.html', **templateData) ❹

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)
```

- ❶ Aktuelle Uhrzeit holen und in `now` ablegen
- ❷ Einen formatierten String mit Datum und Uhrzeit aus dem Objekt `now` erstellen
- ❸ Ein *Dictionary* mit Variablen erstellen (eine Sammlung von *Schlüsseln*, wie `title`, die mit Werten wie `HALLO!` verbunden sind), um es an das Template zu übergeben
- ❹ Das Template *main.html* mit den Variablen aus dem Dictionary `templateData` an den Webbrowser zurückgeben

Beispiel 10-6. Quellcode für *templates/main.html*

```
<!DOCTYPE html>
<html>
  <head>
    <title>{{ title }}</title> ❶->
  </head>

  <body>
    <h1>Hallo, Welt!</h1>
    <h2>Auf dem Server ist es jetzt: {{ time }}</h2> ❷->
  </body>
</html>
```

- ❶ Die Variable `title` im HTML-Titel der Seite nutzen
- ❷ Die Variable `time` auf der Seite einsetzen

Wenn Sie jetzt *hello-template.py* ausführen (wie zuvor müssen Sie dazu `sudo` nutzen) und die IP-Adresse Ihres Raspberry Pi in Ihrem Webbrowser aufrufen, sollten Sie eine formatierte HTML-Seite mit dem Titel »HALLO!« und dem aktuellen Datum und der Uhrzeit des Pi erhalten.



Es ist sehr unwahrscheinlich, dass diese Seite außerhalb Ihres lokalen Netzwerks erreichbar ist. Wollen Sie die Seite auch außerhalb zur Verfügung stellen, müssen Sie Ihren Router für das Port Forwarding konfigurieren. Schauen Sie sich dazu die Dokumentation Ihres Routers an.

Das Web mit der realen Welt verbinden

Sie können Flask mit anderen Python-Bibliotheken zusammen einsetzen, um Ihre Site mit zusätzlicher Funktionalität zu versehen. So können Sie zum Beispiel mit dem Python-Modul RPi.GPIO (siehe Kapitel 8) eine Website aufbauen, die mit der realen Welt verbunden ist. Um das auszuprobieren, verbinden Sie drei Taster oder Schalter mit den Pins 23, 24 und 25 – so, wie Sie es beim Simple Soundboard-Projekt in Abbildung 8-2 gemacht haben.

Der folgende Code erweitert die Funktionalität von *hello-template.py*, kopieren Sie ihn daher in eine neue Datei mit dem Namen *hello-gpio.py*. Fügen Sie das RPi.GPIO-Modul und eine neue *Route* zum Auslesen der Taster hinzu, wie wir es in Angepasster Quellcode für *hello-gpio.py* gemacht haben. Die neue Route übernimmt eine Variable aus der angeforderten URL und nutzt sie, um herauszufinden, welcher Pin ausgelesen werden soll.

Sie sollten auch ein neues Template *pin.html* erstellen. Das unterscheidet sich nicht sehr von *main.html*, daher können Sie *main.html* nach *pin.html* kopieren und dann die entsprechenden Änderungen wie in Beispiel 10-7 vornehmen.

Angepasster Quellcode für *hello-gpio.py*

```
from flask import Flask, render_template
import datetime
import RPi.GPIO as GPIO

app = Flask(__name__)

GPIO.setmode(GPIO.BCM)

@app.route("/")
def hello():
    now = datetime.datetime.now()
    timeString = now.strftime("%Y-%m-%d %H:%M")
```

```

templateData = {
    'title' : 'HALLO!',
    'time': timeString
}
return render_template('main.html', **templateData)

@app.route("/readPin/<pin>") ❶
def readPin(pin):
    try: ❷
        GPIO.setup(int(pin), GPIO.IN) ❸
        if GPIO.input(int(pin)) == True: ❹
            response = "Pin Nummer " + pin + " ist High!"
        else: ❺
            response = "Pin Nummer " + pin + " ist Low!"
    except: ❻
        response = "Es gab einen Fehler beim Lesen von Pin " + pin + "."

    templateData = {
        'title' : 'Status von Pin' + pin,
        'response' : response
    }

    return render_template('pin.html', **templateData)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)

```

- ❶ Eine dynamische Route mit der Pin-Nummer als Variable hinzufügen
- ❷ Führt der folgende, eingerückte Code zu einer Exception, wird der Code im except-Block ausgeführt
- ❸ Übernimmt die Pin-Nummer aus der URL, wandelt sie in eine Zahl um und setzt sie als Input
- ❹ Ist der Pin High, wird der entsprechende Response-Text gesetzt
- ❺ Ansonsten wird der andere Response-Text gesetzt
- ❻ Gab es einen Fehler beim Lesen des Pins, wird die Response entsprechend gesetzt

Beispiel 10-7. Quellcode für templates/pin.html

```

<!DOCTYPE html>
<html>
  <head>
    <title>{{ title }}</title> ❶->
  </head>

  <body>
    <h1>Pin-Status</h1>
    <h2>{{ response }}</h2> ❷->
  </body>
</html>

```

- ❶ Fügt den Titel aus *hello-gpio.py* in den Seitentitel ein
- ❷ Setzt den Response von *hello-gpio.py* auf der Seite in HTML-Heading-Tags

Lassen Sie das obige Skript laufen, können Sie in Ihrem Webbrowser die IP-Adresse des Raspberry Pi eingeben. Dann sollten Sie weiterhin die Standard-Seite mit »Hallo Welt!« erhalten, die wir schon angelegt hatten. Hängen Sie nun aber an das Ende der URL `/readPin/24` an, um so etwas wie `http://10.0.1.103/readPin/24` zu erhalten. Es sollte eine Seite erscheinen, auf der angezeigt wird, dass der Pin Low ist. Halten Sie nun den Taster für Pin 24 gedrückt und aktualisieren Sie die Seite – er sollte nun als High angezeigt werden!

Probieren Sie auch die anderen Taster durch Ändern der URL aus. Das Tolle an diesem Code ist, dass wir die Funktion zum Lesen des Pins nur einmal schreiben mussten und auch die HTML-Seite nur einmal anzulegen war – aber trotzdem sieht es so aus, als ob es separate Webseiten für jeden der Pins gäbe!

Projekt: WebLamp

In Kapitel 7 haben wir Ihnen in „Projekt: Cron-Lampenzeitschaltuhr“ (S. 93) gezeigt, wie Sie den Raspberry Pi als einfachen Zeitschalter für eine Lampe nutzen. Nachdem Sie nun wissen, wie Sie Python und Flask einsetzen, können Sie den Status der Lampe auch über das Web beeinflussen. Dieses einfache Projekt soll als Ausgangspunkt für Sie dienen, damit Sie sehen, wie Sie Geräte über den Raspberry Pi mit dem Internet verbinden können.

Und wie das vorige Flask-Beispiel gezeigt hat, wie Sie den gleichen Code für mehrere Pins einsetzen können, werden Sie dieses Projekt so einrichten, dass Sie in Zukunft auch mehrere Geräte ansteuern können.

1. Das Hardware-Setup für dieses Projekt entspricht exakt dem aus „Projekt: Cron-Lampenzeitschaltuhr“ (S. 93). Alle dort aufgeführten Teile benötigen Sie auch hier.
2. Verbinden Sie den PowerSwitch Tail II mit Pin 25 – so wie beim Cron-Lampenschalter.
3. Haben Sie ein weiteres PowerSwitch-Tail- II-Relais, verbinden Sie es mit Pin 24, um ein zweites Gerät ansteuern zu können. Oder Sie schließen nur eine LED an Pin 24 an. Wir werden dies nutzen, um zu zeigen, wie mehrere Geräte über den gleichen Code gesteuert werden können.
4. Erstellen Sie ein neues Verzeichnis in Ihrem Home-Verzeichnis und geben Sie ihm den Namen *WebLamp*.
5. In *WebLamp* erstellen Sie eine Datei namens *weblamp.py* und fügen den Code aus Beispiel 10-8 ein.

6. Erstellen Sie in *WebLamp* ein neues Verzeichnis mit dem Namen *templates*.
7. Erstellen Sie in *templates* die Datei *main.html*. Den Quellcode für diese Datei finden Sie in Beispiel 10-9.

Wechseln Sie im Terminal ins Verzeichnis *WebLamp* und starten Sie den Server (achten Sie darauf, zuvor mit Strg-C alle anderen Flask-Server zu stoppen, die noch laufen):

```
pi@raspberrypi ~/WebLamp $ sudo python weblamp.py
```

Beispiel 10-8. Quellcode für *weblamp.py*

```
import RPi.GPIO as GPIO
from flask import Flask, render_template, request
app = Flask(__name__)

GPIO.setmode(GPIO.BCM)

pins = {
    24 : {'name' : 'Kaffeemaschine', 'state' : GPIO.LOW},
    25 : {'name' : 'Lampe', 'state' : GPIO.LOW}
} ❶

for pin in pins: ❷
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, GPIO.LOW)

@app.route("/")
def main():
    for pin in pins:
        pins[pin]['state'] = GPIO.input(pin) ❸
    templateData = {
        'pins' : pins ❹
    }
    return render_template('main.html', **templateData) ❺

@app.route("/<changePin>/<action>") ❻
def action(changePin, action):
    changePin = int(changePin) ❼
    deviceName = pins[changePin]['name'] ❽
    if action == "on": ❾
        GPIO.output(changePin, GPIO.HIGH) ❿
        message = deviceName + " eingeschaltet." ⓫
    if action == "off":
        GPIO.output(changePin, GPIO.LOW)
        message = deviceName + " ausgeschaltet."
    if action == "toggle":
        GPIO.output(changePin, not GPIO.input(changePin)) ⓬
        message = deviceName + " umgeschaltet."

    for pin in pins:
        pins[pin]['state'] = GPIO.input(pin) ⓭
```

```

templateData = {
    'message' : message,
    'pins' : pins
} ❷

return render_template('main.html', **templateData)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)

```

- ❶ Erstellt ein Dictionary namens pins, um dort Pin-Nummer, Name und Pin-Status abzulegen
- ❷ Setzt jeden Pin als Output und seinen Status auf Low
- ❸ Liest jeden Pin-Status und speichert ihn in Dictionary pins
- ❹ Steckt das Pin-Dictionary in das Dictionary für die Template-Daten
- ❺ Übergibt die Template-Daten an das Template *main.html* und gibt das Ergebnis an den Anwender zurück
- ❻ Die folgende Funktion wird ausgeführt, wenn jemand eine URL mit der Pin-Nummer und der entsprechenden Aktion anfordert
- ❼ Wandelt den Pin aus der URL in eine Zahl um
- ❽ Holt den Gerätenamen für den zu ändernden Pin
- ❾ Ist der Aktionsteil der URL »on«, wird der folgende eingerückte Code ausgeführt
- ❿ Setzt den Pin auf High
- ⓫ Sichert die Status-Nachricht, um sie an das Template übergeben zu können
- ⓬ Liest den Pin und setzt ihn auf den gegenteiligen Wert (also ein Umschalter)
- ⓭ Liest den Status jedes Pins und legt ihn in Dictionary pins ab
- ⓮ Steckt das Pin-Dictionary zusammen mit der Nachricht in das Template-Daten-Dictionary

Beispiel 10-9. Quellcode für templates/main.html

```

<!DOCTYPE html>
<html>
<head>
    <title>Aktueller Status</title>
</head>

<body>
    <h1>Geräteliste mit Status</h1>

    {% for pin in pins %} ❶->

```

```

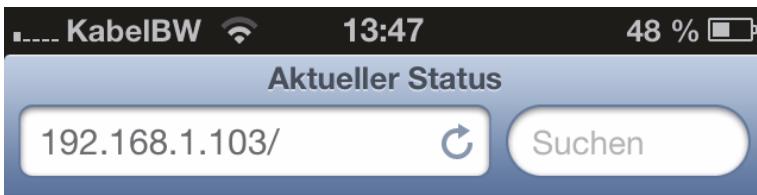
<p>The {{ pins[pin].name }} ❷->
{% if pins[pin].state == true %} ❸->
    ist eingeschaltet (<a href="{{pin}}/off">abschalten</a>)
{% else %} ❹->
    ist ausgeschaltet (<a href="{{pin}}/on">einschalten</a>)
{% endif %}
</p>
{% endfor %}

{% if message %} ❺->
<h2>{{ message }}</h2>
{% endif %}

</body>
</html>

```

- ❶ Durchläuft das Folgende für jeden Pin in Dictionary pins
- ❷ Gibt den Namen des Pins aus
- ❸ Ist der Pin High, wird ausgegeben, dass das Gerät an ist, dazu ein Link mit einer URL zum Abschalten
- ❹ Ansonsten wird ausgegeben, dass das Gerät aus ist, dazu ein Link mit einer URL zum Einschalten
- ❺ Wurde eine Nachricht an das Template übergeben, wird diese ausgegeben



Geräteliste mit Status

Kaffeemaschine ist eingeschaltet ([ausschalten](#))

Lampe ist ausgeschaltet ([einschalten](#))

Abbildung 10-2. Die Bedienoberfläche auf einem iPhone-Webbrowser

Schreiben Sie den Code auf diese Art und Weise, können Sie sehr einfach so viele Geräte hinzufügen, wie die Hardware unterstützt. Tragen Sie einfach die zusätzlichen Informationen über das Gerät in Dictionary pins ein. Wenn

Sie den Server neu starten, wird das neue Gerät in der Statusliste auftauchen und die Steuer-URLs funktionieren automatisch.

Es gibt noch ein weiteres tolles Feature: Wollen Sie den Status eines Geräts mit einem einzelnen Tipp auf Ihrem Smartphone ändern, können Sie ein Le-sezeichen für die Adresse *http://ipaddress/pin/toggle* anlegen. Diese URL prüft den aktuellen Status des Pins und schaltet ihn um.

Weitere Informationen

Requests (<http://docs.python-requests.org/en/latest/>)

Die Homepage von Requests enthält eine sehr umfangreiche Dokumentation mit leicht verständlichen Beispielen.

Flask (<http://flask.pocoo.org/>)

Es gibt noch viel mehr über Flask zu sagen. Die offizielle Site enthält das vollständige Feature-Set.

Flask Extensions (<http://flask.pocoo.org/extensions/>)

Flask-Erweiterungen erleichtern das Hinzufügen von Funktionalität für Ihre Site.

A/Ein SD-Karten-Image schreiben

Dieses Buch dreht sich zwar um das Raspbian-Betriebssystem, aber es gibt viele andere Distributionen, die auf dem Raspberry Pi laufen können. Bei jeder müssen Sie einfach nur das Disk Image herunterladen und dann (nicht ganz so einfach) dieses Image auf die SD-Karte kopieren. Im Folgenden erhalten Sie eine Beschreibung, wie Sie aus dem Image unter OS X, Windows und Linux eine SD-Karte erstellen.

Eine SD-Karte unter OS X schreiben

1. Öffnen Sie Ihr Terminal-Tool (Sie finden es unter */Applications/Utilities* beziehungsweise im Finder unter *Programme/Dienstprogramme*), um eine Befehlszeile zu erhalten.
2. Geben Sie *ohne* die Karte im Kartenleser Ihres Rechners den Befehl `df -h` ein. Das Programm `df` zeigt Ihnen Ihren freien Festplattenplatz, aber auch, welche Disk Volumes gemountet sind.
3. Nun stecken Sie die SD-Karte ein und rufen `df -h` erneut auf.
4. Schauen Sie sich die Liste der gemounteten Volumes an und ermitteln Sie, welches die SD-Karte ist, indem Sie die Liste mit der vorigen Ausgabe vergleichen. So wird auf Ihrem Computer vielleicht eine SD-Karte als */Volumes/Untitled* gemountet, während der Device-Name */dev/disk3s1* ist. Abhängig von der Konfiguration Ihres Rechners kann dieser Name aber anders sein. Namen werden beim Mounten von Geräten zugewiesen, daher erhalten Sie vielleicht eine größere Zahl, wenn andere Geräte oder Disk Images im Finder schon gemountet sind. Schreiben Sie sich den Device-Namen der Karte auf.
5. Um auf die Karte zu schreiben, müssen Sie sie erst unmounten. Dazu geben Sie `sudo diskutil unmount /dev/disk3s1` ein (natürlich mit dem Gerätenamen, den Sie sich vorher aufgeschrieben haben). Beachten

Sie, dass Sie die Befehlszeile oder das Festplatten-Dienstprogramm nehmen *müssen*, um die Karte zu unmounten. Wenn Sie sie einfach nur im Finder auswerfen, müssen Sie sie herausnehmen und wieder einstecken (und sie immer noch an der Befehlszeile oder mit dem Festplatten-Dienstprogramm unmounten). Lässt sich die Karte nicht unmounten, achten Sie darauf, dass alle Finder-Fenster geschlossen sind, die den Inhalt der Karte anzeigen.

6. Als Nächstes müssen Sie den Raw-Device-Namen der Karte herausfinden. Nehmen Sie Ihren Gerätenamen, ersetzen Sie `disk` durch `rdisk` und lassen Sie das `s1` weg (dabei handelt es sich um die Partitionsnummer). So lautet der Raw-Device-Name für das Gerät `/dev/disk3s1` zum Beispiel `/dev/rdisk3`.



Es ist sehr wichtig, dass Sie den Raw-Device-Namen korrekt ermitteln! Sie können Ihre Festplatte überschreiben und Daten verlieren, wenn Sie statt auf die SD-Karte auf die Festplatte schreiben. Nutzen Sie nochmals `df` für eine doppelte Sicherung, bevor Sie fortfahren.

-
7. Achten Sie darauf, dass das heruntergeladene Image entpackt ist und sich in Ihrem Home-Verzeichnis befindet. Sie werden das Unix-Tool `dd` einsetzen, um das Image Bit für Bit auf die SD-Karte zu kopieren. Im folgenden finden Sie die Anweisung – ersetzen Sie nur den Namen des Disk Image durch den der von Ihnen heruntergeladenen Datei und `/dev/rdisk3` durch den Raw-Device-Namen der SD-Karte aus Schritt 6. Sie erfahren mehr über die Befehlszeile in Kapitel 3, aber kurz gesagt weisen Sie `dd` an, als Root zu laufen und die Quelldatei (Input File, `if`) in die Zieldatei (Output File, `of`) zu kopieren.

```
sudo dd bs=1m if=~/.2012-09-18-wheezy-raspbian.img of=/dev/rdisk3
```

8. Es wird ein paar Minuten dauern, das gesamte Disk Image zu kopieren. Leider gibt `dd` keinerlei Rückmeldung über den aktuellen Stand – Sie müssen einfach warten. Nach Abschluss des Kopierens werden ein paar Statistikdaten ausgegeben. Werfen Sie die SD-Karte aus – Sie können sie jetzt in Ihrem Pi ausprobieren.

Eine SD-Karte unter Windows schreiben

1. Laden Sie das Programm Win32DiskImager (<https://launchpad.net/win32-image-writer>) herunter.
2. Stecken Sie die SD-Karte in Ihren Kartenleser und merken Sie sich den Laufwerksbuchstaben, der im Windows-Explorer erscheint.

3. Öffnen Sie Win32DiskImager und wählen Sie das Disk Image von Raspbian aus.
4. Wählen Sie den Laufwerksbuchstaben des SD-Kartenlesers und klicken Sie dann auf Write. Gibt es beim Beschreiben der Karte mit Win32DiskImager Probleme, versuchen Sie, sie im Windows-Explorer neu zu formatieren.
5. Werfen Sie die SD-Karte aus und stecken Sie sie in Ihren Raspberry Pi – fertig!

Eine SD-Karte unter Linux schreiben

Die Anweisungen für Linux entsprechen denen für den Mac:

1. Öffnen Sie eine neue Shell und geben Sie *ohne* die Karte im Kartenleser den Befehl `df -h` ein, um zu sehen, welche Disk Volumes gemountet sind.
2. Schieben Sie die SD-Karte ein und führen Sie `df -h` erneut aus.
3. Schauen Sie sich die Liste der gemounteten Volumes an und ermitteln Sie, welches die SD-Karte ist, indem Sie die Liste mit der vorigen Ausgabe vergleichen. Suchen Sie den Device-Namen heraus, der so ähnlich wie `/dev/sdd1` lauten sollte. Abhängig von der Konfiguration Ihres Rechners kann dieser Name aber anders sein. Schreiben Sie sich den Device-Namen der Karte auf.
4. Um auf die Karte zu schreiben, müssen Sie sie erst unmounten. Dazu geben Sie `umount /dev/sdd1` ein (mit dem Gerätenamen aus dem vorigen Schritt statt `/dev/sdd1`). Lässt sich die Karte nicht unmounten, prüfen Sie, ob es sich bei ihr um das Arbeitsverzeichnis in einer offenen Shell handelt.
5. Als Nächstes müssen Sie den Raw-Device-Namen der Karte ermitteln. Dabei handelt es sich um den Device-Namen ohne die Partitionsnummer. Ist Ihr Device-Name zum Beispiel `/dev/sdd1`, lautet der Raw-Device-Name `/dev/sdd`.



Es ist sehr wichtig, dass Sie den Raw-Device-Namen korrekt ermitteln! Sie können Ihre Festplatte überschreiben und Daten verlieren, wenn Sie statt auf die SD-Karte auf die Festplatte schreiben. Nutzen Sie nochmals `df` für eine doppelte Sicherung, bevor Sie fortfahren.

6. Achten Sie darauf, dass das heruntergeladene Image entpackt ist und sich in Ihrem Home-Verzeichnis befindet. Sie werden das Unix-Tool `dd` einsetzen, um das Image Bit für Bit auf die SD-Karte zu kopieren. Im Folgenden finden Sie die Anweisung – ersetzen Sie nur den Namen

des Disk Image durch den der von Ihnen heruntergeladenen Datei und /dev/sdd durch den Raw-Device-Namen der SD-Karte aus Schritt 5.

```
sudo dd bs=1M if=~/.2012-09-18-wheezy-raspbian.img of=/dev/sdd
```

Dieser Befehl weist dd an, als Root zu laufen und die Quelldatei (Input File, if) in die Zieldatei (Output File, of) zu kopieren.

7. Es wird ein paar Minuten dauern, das gesamte Disk Image zu kopieren. Leider gibt dd keinerlei Rückmeldung über den aktuellen Stand – Sie müssen einfach warten.

Nach Abschluss des Kopierens werden ein paar Statistikdaten ausgegeben. Sie könnten die SD-Karte jetzt auswerfen, aber um sicher zu sein, rufen Sie `sudo sync` auf, mit dem die Schreibpuffer des Dateisystems geleert werden.

8. Werfen Sie die Karte aus und stecken Sie sie in Ihren Raspberry Pi. Los geht's!

BerryBoot

Eine zweite Möglichkeit, das Betriebssystem auf eine SD-Karte zu bekommen, ist der Einsatz des BerryBoot-Tools (<http://www.berryterminal.com/doku.php/berryboot>). BerryBoot ist Teil des BerryTerminal Thin Client Project und ermöglicht es Ihnen, mehrere Betriebssysteme auf einer einzelnen Karte unterzubringen. Sie packen das BerryBoot Image auf eine SD-Karte, starten den Raspberry Pi und erhalten einen interaktiven Installer, mit dem Sie ein Betriebssystem aus einer Liste auswählen können. Sie müssen allerdings dazu mit einem Netzwerk verbunden sein.

B/Vollständiger Code von Astral Trespassers

Dieser Anhang enthält die vollständigen Scratch-Skripten für alle Sprites in Astral Trespassers (beschrieben in Kapitel 5). Die Beispiele sind auch dort vollständig, aber da es sich bei Scratch um eine visuelle Programmiersprache handelt, kann es hilfreich sein, alle zusammen an einem Ort zu finden. Die Kostüm-Tabs für jedes Sprite sind ebenfalls aufgeführt.

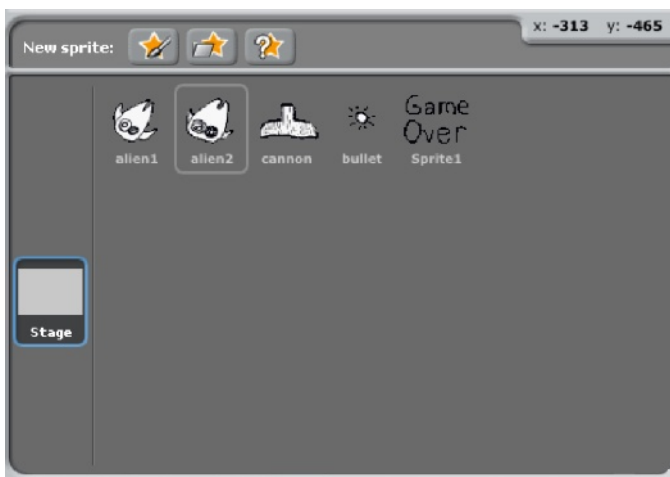


Abbildung B-1. Die Sprite-Liste und die fünf Sprites

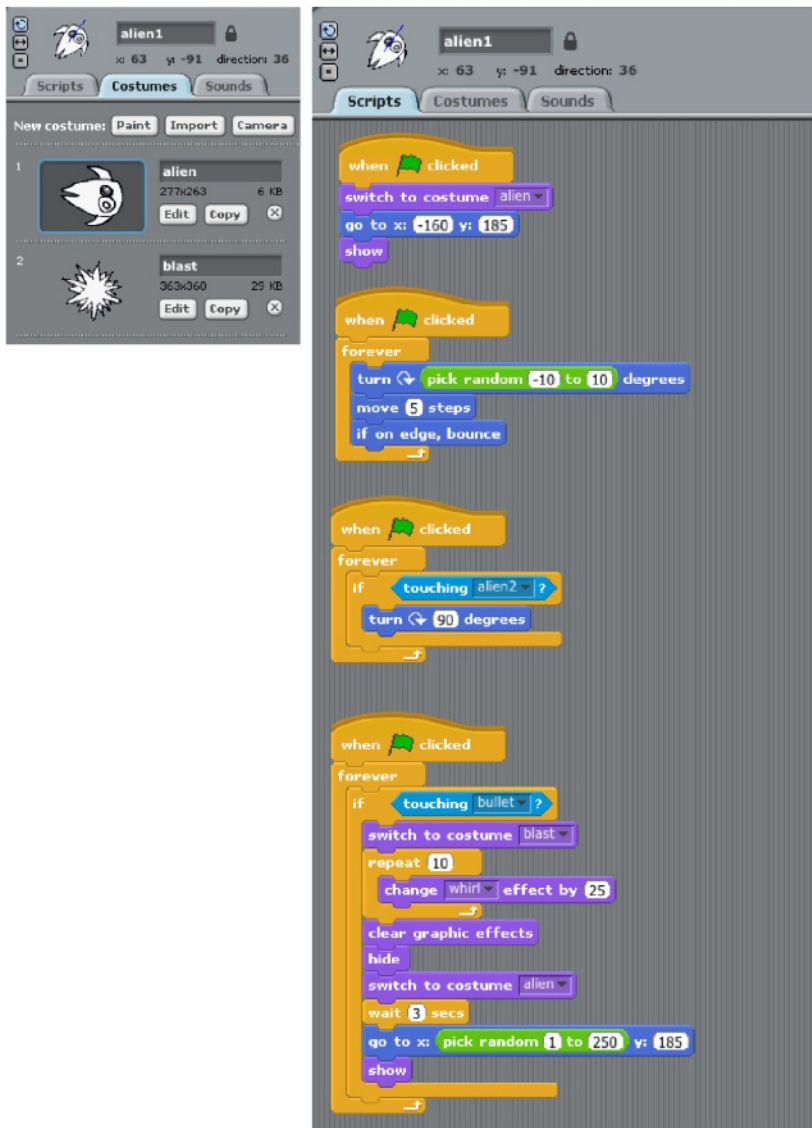


Abbildung B-2. Der Kostüm-Tab des Sprites alien1 (links) und das vollständige Skript (rechts)

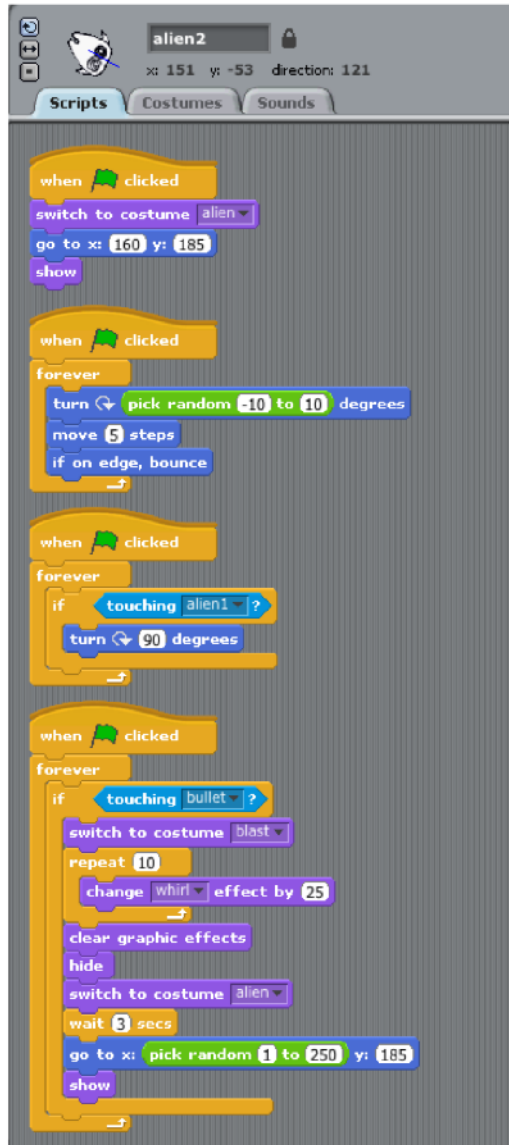
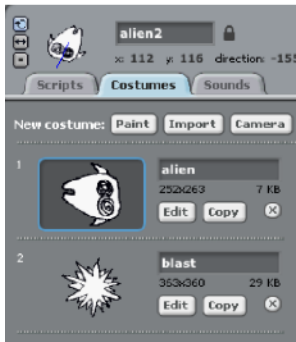


Abbildung B-3. Der Kostüm-Tab des Sprites alien2 und das vollständige Skript

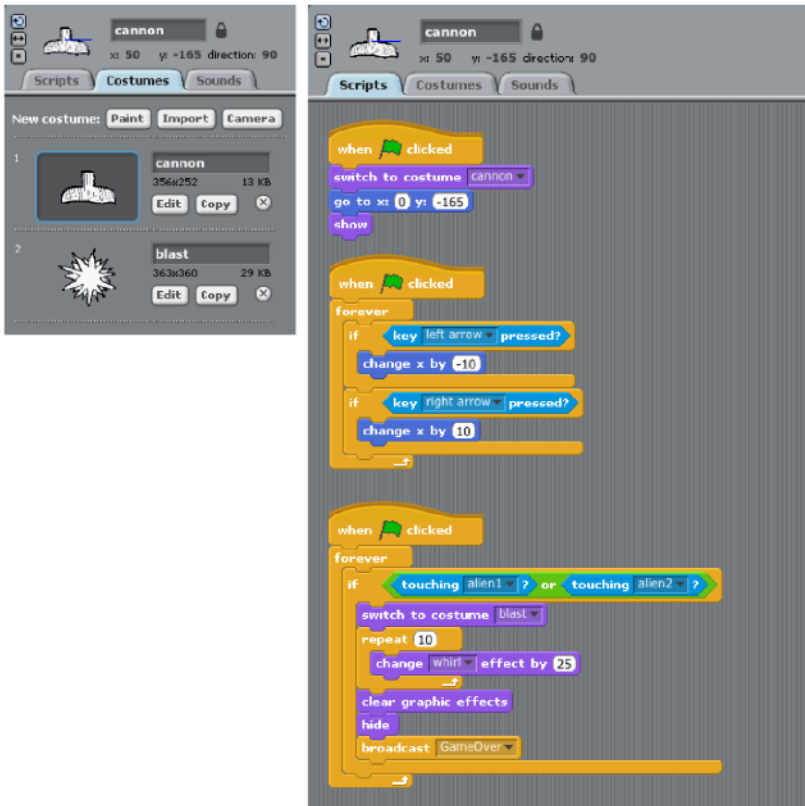


Abbildung B-4. Der Kostüm-Tab des Kanonen-Sprites und das vollständige Skript

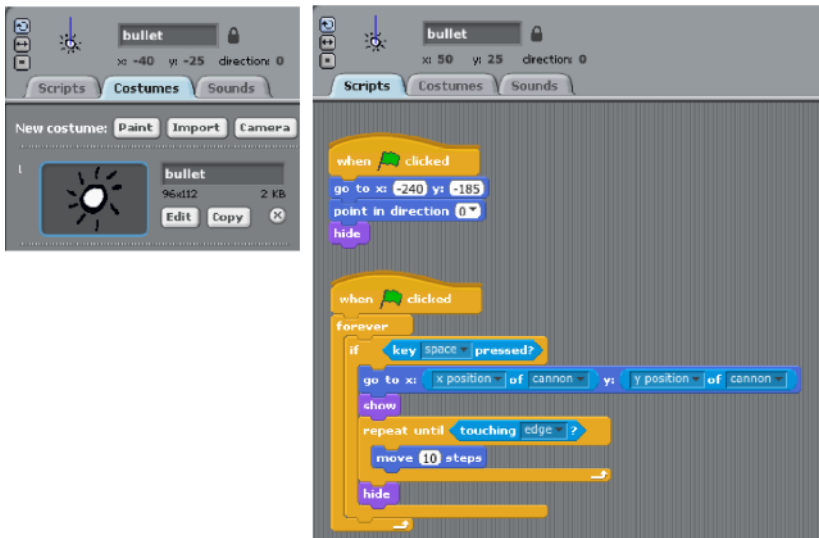


Abbildung B-5. Der Kostüm-Tab der Kanonenkugel und das vollständige Skript

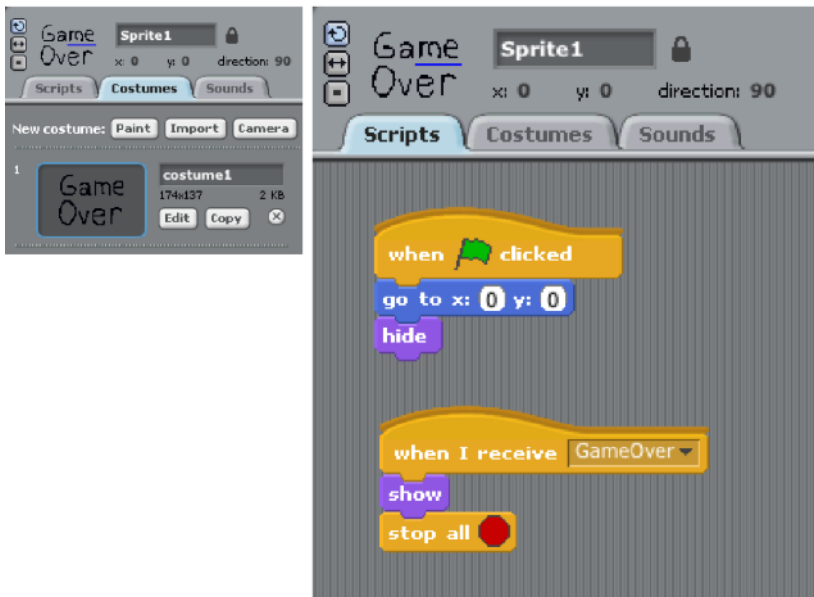


Abbildung B-6. Der Kostüm-Tab des gameover-Sprites und das vollständige Skript

C/Analoger Input

In diesem Buch haben Sie gelernt, wie Sie digitale Ein- und Ausgänge mit Tastern, Schaltern, LEDs und Relais nutzen können. Jede dieser Komponenten war entweder ein oder aus, aber nie irgendetwas dazwischen. Allerdings wollen Sie vielleicht auch einmal reale Dinge messen, die nicht notwendigerweise das eine oder das andere sind – wie Temperatur, Abstände, Helligkeit oder der Status eines Potenziometers.

Leider können Sie diese Art von Sensoren nicht direkt mit dem Raspberry Pi auslesen, da er nur digitale Input-Kanäle an Bord hat. Es gibt aber verschiedene Möglichkeiten, durch zusätzliche Komponenten analoge Sensoren über die digitalen Eingänge des Pi auszulesen.

Analog nach digital wandeln

Um von analog nach digital zu wandeln, zeigt Ihnen dieser Anhang, wie Sie einen ADC (*Analog to Digital Converter*) nutzen. Es gibt verschiedene Modelle, wir werden den ADS1015 von Texas Instruments nutzen. Der ADS1015 ist ein bisschen zu klein für Ihre Standard-Steckplatine, daher hat Adafruit Industries ein Breakout Board dafür aufgelegt (<http://www.adafruit.com/products/1083>, siehe Abbildung C-1). Haben Sie die Stiftleiste an das Breakout Board angelötet, können Sie mit diesem Chip in Ihrer Steckplatine experimentieren. Der Chip nutzt ein Protokoll namens I2C für das Übermitteln der analogen Werte. Zum Glück müssen wir das Protokoll nicht verstehen, um es einsetzen zu können. Adafruit stellt eine ausgezeichnete Open-Source-Python-Bibliothek bereit, um die Werte des ADS1015 und seines großen Bruders, des ADS1115, per I2C zu ermitteln.

Um das Breakout-Board des ADS1015 mit Ihrem Raspberry Pi zu verbinden:

1. Verbinden Sie den 3,3-V-Pin des Raspberry Pi mit dem positiven Anschluss der Steckplatine. In Abbildung 7-2 finden Sie die entsprechenden GPIO-Pins des Pi.

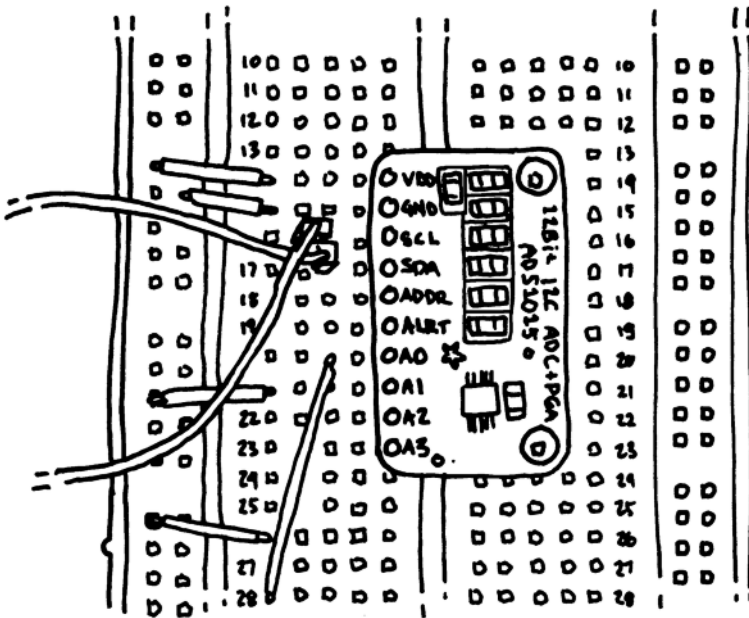


Abbildung C-1. Das Breakout-Board für den Analog-nach-Digital-Konverter ADS1015 von Adafruit

2. Verbinden Sie den Masse-Pin des Raspberry Pi mit der negativen Leiste der Steckplatine.
3. Stecken Sie den ADS1015 in die Steckplatine und nutzen Sie Jumper-Kabel, um dessen VDD-Pin mit der positiven Leiste und den GND-Pin mit der negativen Leiste zu verbinden.
4. Verbinden Sie den SCL-Pin des ADS1015 mit dem SCL-Pin des Raspberry Pi. Der SCL-Pin des Pi ist derjenige, der mit dem Masse-Pin auf der GPIO-Steckleiste zusammenliegt.
5. Verbinden Sie den SDA-Pin des ADS1015 mit dem SDA-Pin des Raspberry Pi. Der SDA-Pin liegt zwischen dem SCL-Pin und dem 3,3-V-Pin.

Jetzt müssen Sie einen analogen Sensor an den ADS1015 anschließen. Es gibt eine ganze Reihe von Möglichkeiten, aber für dieses kleine Experiment werden wir ein 2-K Ω -Potenziometer nutzen, um einen Drehregler für unseren Raspberry Pi zu haben. Ein *Potenzimeter* oder *Poti* ist im Prinzip nichts weiter als ein variabler Widerstand, der als Drehregler oder Schieberegler erhältlich ist. Um ein Potenziometer mit dem ADS1015 zu verbinden:

1. Stecken Sie das Potenziometer auf die Platine.

2. Das Poti besitzt drei Pins. Verbinden Sie den mittleren mit dem Pin A0 des ADS1015.
3. Verbinden Sie einen der äußeren Pins mit der positiven Leiste der Steckplatine. Es macht hier keinen Unterschied, welchen der Pins Sie nutzen.
4. Verbinden Sie den anderen äußeren Pin mit der negativen Leiste der Steckplatine.

Die Verbindungen sollten nun so aussehen wie in Abbildung C-2.

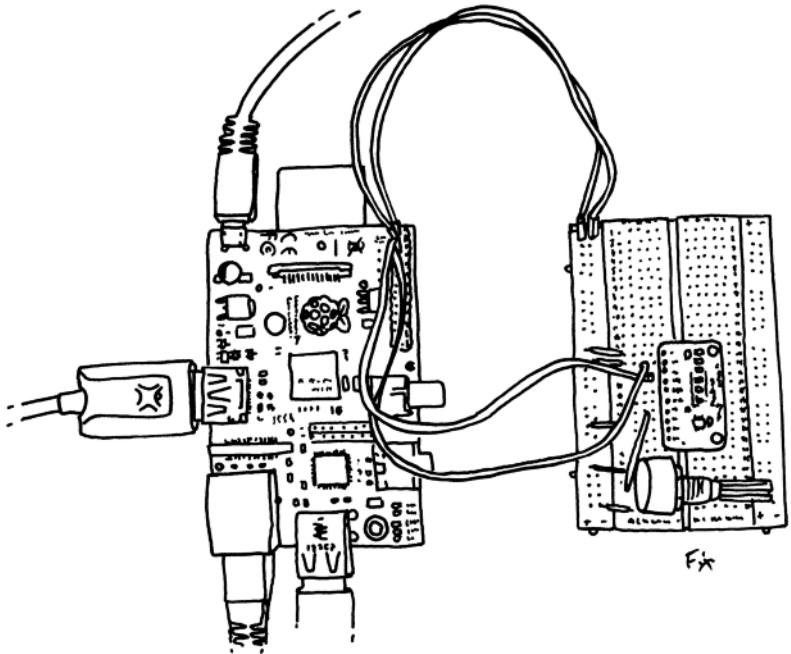


Abbildung C-2. Einsatz des ADS1015, um ein Potenziometer mit dem Raspberry Pi zu verbinden

Bevor Sie das Potenziometer auslesen können, müssen Sie das Protokoll I2C unterstützen und dafür ein paar Bibliotheken installieren:

1. Öffnen Sie an der Befehlszeile die Datei `raspi-blacklist.conf` als Root:

```
pi@raspberrypi ~ $ sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

Um I2C aus dieser Blacklist zu entfernen, setzen Sie ein Rautezeichen an den Anfang der Zeile `blacklist i2c-bcm2708`. Sie sollte nun so aussehen:

```
# blacklist spi and i2c by default (many users don't need them)

blacklist spi-bcm2708
#blacklist i2c-bcm2708
```

2. Drücken Sie Strg-X, um den Editor zu verlassen, und y, um die Datei zu speichern.

3. Öffnen Sie als Nächstes */etc/modules*:

```
pi@raspberrypi ~ $ sudo nano /etc/modules
```

4. Fügen Sie am Ende der Datei in einer eigenen Zeile *i2c-dev* hinzu. Die Datei sollte nun so aussehen:

```
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.

snd-bcm2835
i2c-dev
```

5. Drücken Sie Strg-X, um den Editor zu verlassen, und y, um die Datei zu speichern.

6. Aktualisieren Sie Ihre Paketliste:

```
pi@raspberrypi ~ $ sudo apt-get update
```

7. Installieren Sie die *i2c-tools* und *python-smbus*:

```
pi@raspberrypi ~ $ sudo apt-get install i2c-tools python-smbus
```

8. Starten Sie Ihren Raspberry Pi neu.

9. Testen Sie nach dem Neustart, ob der Raspberry Pi den ADS1015 erkennt. Auf Revision-1-Boards nutzen Sie:

```
pi@raspberrypi ~ $ sudo i2cdetect -y 0
```

Auf Revision-2-Boards nutzen Sie dagegen:

```
pi@raspberrypi ~ $ sudo i2cdetect -y 1
```

10. Wird das Board erkannt, sehen Sie die Zahl in der Tabelle:

```
      0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
40:  --  --  --  --  --  --  --  --  48  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
```

11. Nachdem Sie nun wissen, dass das Gerät angeschlossen ist und vom Pi erkannt wird, wollen wir das Potenziometer auslesen. Dazu laden Sie die Python-Bibliotheken für den Raspberry Pi vom Adafruits Code Repository in Ihren Home-Ordner. Geben Sie die folgende Anweisung an der Befehlszeile ein – alles in einer Zeile und ohne Leerzeichen in der URL:

```
wget https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code/archive/master.zip
```

12. Packen Sie die Datei aus:

```
pi@raspberrypi ~ $ unzip master.zip
```

13. Wechseln Sie in das Verzeichnis ADS1x15 der Bibliothek:

```
$ cd Adafruit-Raspberry-Pi-Python-Code-master/Adafruit_ADS1x15
```

14. Lassen Sie die Beispieldatei laufen:

```
$ sudo python ads1015_example.py
Channel 0 = 2.067 V
Channel 1 = 3.309 V
```

15. Drehen Sie das Potenziometer ganz in eine Richtung und lassen Sie das Programm erneut laufen. Achten Sie auf den geänderten Wert für Channel 0:

```
$ sudo python ads1015_example.py
Channel 0 = 3.306 V
Channel 1 = 3.309 V
```

16. Drehen Sie das Potenziometer nun ganz in die andere Richtung und führen Sie das Programm erneut aus:

```
$ sudo python ads1015_example.py
Channel 0 = 0.000 V
Channel 1 = 3.309 V
```

Wie Sie sehen, sorgt ein Drehen des Potenziometers für ein Ändern der Spannung des Kanals 0 des ADS1015. Der Code in *ads1015_example.py* rechnet ein bisschen, um den Spannungswert aus den Daten des ADC zu ermitteln. Bei Ihnen wird die Rechnung sicherlich ein bisschen anders aussehen – je nachdem, welche Art Sensor Sie nutzen wollen.

Erstellen Sie eine neue Datei im gleichen Verzeichnis mit dem folgenden Code:

```
from Adafruit_ADS1x15 import ADS1x15 ❶
from time import sleep
```

```

adc = ADS1x15() ❷

while True:
    result = adc.readADCSingleEnded(0) ❸
    print result
    sleep(.5)

```

- ❶ Importiert Adafruits ADS1x15-Bibliothek
- ❷ Erstellt ein neues ADS1x15-Objekt mit dem Namen adc.
- ❸ Liest Kanal A0 des ADS1015 aus und speichert den Wert in result.

Lassen Sie diesen Code als Root ausführen, werden zweimal pro Sekunde die Rohdaten der Messung ausgegeben. Durch Drehen am Poti werden die Werte steigen oder fallen.

Ist alles eingerichtet, erledigt die Adafruit-Bibliothek ADS1x15 die ganze Arbeit für Sie, und der Einsatz von analogen Sensoren ist dann sehr einfach. Wollen Sie zum Beispiel Ihr eigenes Pong-Spiel bauen, können Sie zwei Potenziometer auslesen und dann mit Pygame das Spiel auf dem Monitor ausgeben. Mehr Informationen zu Pygame finden Sie in Kapitel 4.

Adafruit's Educational Linux Distro

Adafruit Industries hat einen Fork der eigentlichen Linux-Distribution Raspbian/Wheezy erstellt und ihn mit zusätzlichen Treibern und Software für das einfachere elektronische Prototyping ausgestattet. Das Ergebnis ist die Adafruit Raspberry Pi Educational Linux Distro (<http://learn.adafruit.com/adafruit-raspberry-pi-educational-linux-distro>), Codename *Occidentalis* (der Artnamen der Schwarzen Himbeere).

Occidentalis ist ein interessantes Beispiel für das Open-Source-Modell im Einsatz – das Linux-Ökosystem besitzt eine lange Geschichte abgespaltenen Distributionen, die für eine bestimmte Anwendung hin optimiert wurden. Häufig sind diese Forks Grundlage für Verbesserungen, die dann auch wieder Eingang in andere Distributionen findet.

Viele der Verbesserungen und Ergänzungen in der Distribution dienen der Unterstützung häufig eingesetzter Sensoren und bestimmter Adafruit-Produkte, ohne weitere Software installieren zu müssen. Es gibt auch ein paar Verbesserungen sehr tief im System, die den Einsatz der Pulsweitenmodulation und das Ansteuern von Servomotoren direkt vom Raspberry Pi ermöglichen.

Auf der Overview-Seite (<http://learn.adafruit.com/adafruit-raspberry-pi-educational-linux-distro>) finden Sie eine Liste mit Features und in den Adafruit Tutorials (<http://learn.adafruit.com/category/raspberry-pi>) Beispiele für den Einsatz von Occidentalis.

Index

/etc, 32

A

A la Mode, 82
Adafruit Raspberry Pi
 Educational Linux, 10
Adafruit's Educational
 Linux Distro, 158
Analog nach digital, 153
Analoger Audio-Ausgang,
 4
Analoger Input, 153
API, 127
Arch Linux, 10
Arduino, 75
Arduino in Raspbian, 76
Arduino-IDE, 77
Arduino-Tutorial, 76
ASCII-Tabelle, 80
Astral Trespassers, 66
Ausgabe über die GPIO-
 Pins, 86
Autovervollständigen, 22

B

Backup-Batterie, 33
Bühne, 63
Befehlspuffer, 22
Befehlszeile, 22
Berechtigungen, 29, 30
Betriebssystem, 9
Bild anzeigen, 114
Bild verändern, 116
BitTorrent, 11
Booten, 12
Bourne Again Shell
 (BASH), 22
Breakout Board, 153
Broadcom-Chip, 111

C

Camera-Serial-Interface-
 (CSI-)Anschluss, 5

cd, 23
Change Locale, 14
Change timezone, 14
chmod, 30
Composite-Video-
 Ausgang, 4
Computer Vision, 113
Copy and Paste, 21
Cron, 97
Cron-Lampenzeit-
 schaltuhr, 93

D

dash, 22
Dateimanager, 19
Dateisystem, 84
Datum, 33
Desktop Behavior, 15
Diagramm, 106
Digitaler Eingang, 90
Display-Serial-Interface-
 (DSI-)Anschluss, 5

E

echo, 89
Echtzeituhr, 8
Ein- und Ausgänge, 83, 99
Element 14, 1
E-Mail, 125
Ethernet-Kabel, 7
Ethernet-Port, 2
Expand rootfs, 14

F

Fehlersuche, 16, 45
Firmata, 82
Flagge, 62
flashen, 11
Flask, 132
Framework, 47
Fritzing, 106
FULLSCREEN, 53

G

Gehäuse, 8
General Purpose Input
 and Output (GPIO), 5
General Purpose Input
 and Output-Pins, 83
Geräuschsensor, 72
Gesichtserkennung, 120
GPIO, 99

H

Hardware-Design, 106
HDMI-Kabel, 7
HDMI-Port, 2
HTTP-Requests, 127
HTTP-Statuscodes, 126
Hut-Blöcke, 61

I

IDLE, 35
ifconfig, 31
Internet, 125
Interrupt, 105
iPhone, 140

J

Java, 77

K

Kameramodul, 8
Kühlkörper, 8
Keyboard, 14
 Klänge, 55, 65
Konfigurieren, 14
Kostüme, 64

L

Lampenzeitschaltuhr, 93
Laptop-Dock, 8
LCD-Display, 8
Leafpad, 21
LED, 102

Lernwerkzeug, viii
LibreOffice, viii
Lightweight X11 Desktop
Environment (LXDE), 19
Linux, x
Linux-Distribution, 10
LXTerminal, 22

M

Mac OS X, 143
Masse, 91
Memory split, 14
MIDI, 82
Midori, 20
Modul, 41
Multimedia-Programmie-
rung, 47

N

Nano, 21
Network Installer, 11
Netzwerk, 31
Node.js, 82
Noduino project, 82

O

Objekt, 41
Overclock, 15
Overscan, 14

P

Password, 14
Photobooth, 121
Pi Cobbler Breakout Kit,
85
PicoBoard, 71
Pins, 5
Polling, 104
Power Switch Tail, 95
Premier Farnell, 1
Projektplattform, viii
Prozesse, 29
Prozessor, 2
Pygame, 47
Pygame-Surface, 49
pySerial, 78
Python, 35
Python Package Index
(PyPI), 44

Q

QtonPi, 10

R

Raspberry Pi Starter Kit,
85
Raspbian, x, 10
raspi-config Tool, 13
Reporter-Blöcke, 61
Requests, 126
root, 88
RPi.GPIO, 135
RS Components, 1

S

Schaltkreis, 107
Schiebereglер, 72
Schlüsselwort, 41
Schrägstrich, 23
Schriften, 50
Scratch, 59
Scratch-Umgebung, 60
Screenshot, 34
SD-Karte, 7
SD-Karten-Image, 143
Secure Digital, 2
Secure Sockets Layer
(SSL), 32
Sensor, 71
Sensor-Board, 71
Serielle Kommunikation,
78
sftp, 32
Shield, 82
SimpleCV, 113
Skripte-Bereich, 60
Skripten, 60
SMS, 125
Software installieren, 33
Soundboard, 106
Spielautomaten-
Sofatisch, x
Sprites, 54
SSH, 15
Stapelbare Blöcke, 61
Status-LED, 4
Steckleisten, 5
Steckplatine, 86
Steuertasten, 28

Stiftleiste, 5
Stromversorgung, 4, 7
Sudo, 29
Surface, 50
Syntax-Fehler, 45

T

Taster, 72, 90
Taster auslesen, 104
Tastschalter, 90
Texteditor, 21, 97
The Camera Serial Inter-
face (CSI), 111
Tilde (~), 23
Twitter, 125

U

Uhrzeit, 33
Update, 15
USB-Hub, 7
USB-Port, 2

V

Verzeichnis, 32
Verzeichnispfad, 23
Videoquelle, 113
Videos, 57
Videostream, 118
Virtuelle Datei, 90

W

Webbrowser, 20
Webcam, 111
WebLamp, 137
Webserver, 125, 131
WebSocket, 82
Wettervorhersage, 127
Windows, x
WLAN-Adapter, 111
WLAN-USB-Stick, 8
WyoLum, 82

X

Xbian, 10

Z

zeitgesteuert, 96
Zubehör, 6

Über die Autoren

Matt Richardson ist Technologie und Videoproduzent in Brooklyn. Er schreibt für das *MAKE*-Magazin und Makezine.com. Zudem ist er Eigentümer der Awesome Button Studios, einer Technologie-Beratungsfirma. Zu den Highlights seiner Arbeit gehört die Descriptive Camera – eine Kamera, die eine Textbeschreibung einer Szene statt eines Fotos ausgibt. Er hat auch The Enough Already erstellt, eine Arduino-Bastelei, um zu häufig vorkommende Promis im Fernsehen stummzuschalten. Matts Arbeiten wurden von *The New York Times*, *Wired* und *New York Magazine* erwähnt und fanden sich auch schon im The Nevada Museum of Art und auf der Santorini Biennale. Er betreibt aktuell ein Master-Studium im Interactive Telecommunications Program der New York University.

Shawn Wallace ist *Redakteur* bei Make Media und lebt in Providence (Rhode Island). Er ist auch Mitglied des Künstlerkollektivs Fluxama, das sich für neuartige iOS-Musikinstrumente wie Noisemusick und Doctor Om verantwortlich zeigt. Er hat bei Modern Device Open Hardware Kits entworfen und an der Fab Academy im Providence Fab Lab unterrichtet. Viele Jahre war er geschäftsführender Direktor des Kunstzentrums AS220 und Mitbegründer der SMT Computing Society.

Die Schriftart auf dem Cover und im Text ist BentonSans, die Überschriftenschrift ist Serifa und die Code-Schrift ist TheSansMonoCondensed.