Aaron Newcomb

Linux für Maker

Raspbian –
das Betriebssystem des
Raspberry Pi richtig verstehen
und effektiv nutzen

Make:

dpunkt.verlag

Aaron Newcomb ist Maker, seitdem er einen Schraubendreher in der Hand halten konnte, und verwendet Linux bereits seit 1997. Er war in der IT-Branche für Firmen wie New Relic, NetApp, Oracle, Sun Microsystems und Hewlett Packard tätig. Er moderiert viele Sendungen des Internet-TV-Senders TWit LLC mit, darunter FLOSS Weekly, All About Android, This Week in Google und The New Screen Savers. 2013 gründete er die gemeinnützige Organisation Benicia Makerspace, deren Präsident er ist, und wenn er neben der Arbeit noch Zeit dazu findet, auch noch deren Geschäftsführer.

Aaron Newcomb

Linux für Maker

Das Betriebssystem des Raspberry Pi richtig verstehen und effektiv nutzen

dpunkt.verlag

Aaron Newcomb

Lektorat: Dr. Michael Barabas
Übersetzung: Volker Haxsen
Copy-Editing: Ursula Zimpfer, Herrenberg
Fachliche Durchsicht: Maik Schmidt
Satz: Birgit Bäuerlein
Herstellung: Susanne Bröckelmann
Umschlaggestaltung: Helmut Kraus, www.exclam.de
Druck und Bindung: M.P. Media-Print Informationstechnologie GmbH, 33100 Paderborn

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

ISBN:
Print 978-3-86490-511-7
PDF 978-3-96088-359-3
ePub 978-3-96088-360-9
mobi 978-3-96088-361-6

1. Auflage 2018

dpunkt.verlag GmbH
Wieblinger Weg 17
69123 Heidelberg

Authorized German translation of the English edition of Linux for Makers ISBN 9781680451832

5 4 3 2 1 0

Vorwort

Als ich in meiner Heimatgemeinde meinen ersten »Makerspace« eröffnete, konnte ich einige interessante Lernstrategien beobachten. Während sich manche weigerten, sich eine neue Fertigkeit anzueignen, solange ihnen niemand beim Einstieg half, begannen andere sofort damit, auf eigene Faust herumzuprobieren, ohne eine Vorstellung davon zu haben, was sie da eigentlich machten. Letzteres führte in der Regel zu langsamen Fortschritten, bis wieder jemand kam, der ihnen mit einigen Hinweisen weiterhalf. Bei beiden Strategien beschleunigte also ein wenig Anschubhilfe den gesamten Lernprozess.

Zu lernen, wie man Linux für seine Bastelprojekte sinnvoll einsetzt, ist gewiss keine leichte Aufgabe. Viele behelfen sich damit, von einer Anleitung im Internet Befehle in die Kommandozeile von Linux zu kopieren, ohne zu verstehen, was sie im Einzelnen bewirken. Spätestens, wenn sie eigene Ideen umsetzen möchten, wird es frustrierend. Hinzu kommt, dass viele Maker vor dem Einsatz von Raspberry Pi oder vergleichbaren Einplatinencomputern zurückschrecken, da ihnen Linux sehr fremd ist und die in vielen Anleitungen angegebene Verwendung der Kommandozeile ihnen schwieriger erscheint, als eine grafische Benutzeroberfläche zu bedienen.

Dieses Buch hat zum Ziel, diese Ängste zu überwinden und eine solide Grundlage für weiteres Lernen und Entdecken zu liefern. Linux ist schlussendlich nur ein weiteres Werkzeug in Ihrer Werkzeugkiste als Maker. Linux mag sich zwar sehr von Ihren bisher gewohnten Betriebssystemen unterscheiden, doch – wie letztlich alle Werkzeuge – ist es nicht schwieriger als andere, sobald Sie wissen, wie man es benutzt. Linux ist so mächtig und vielseitig, dass Sie sich am Ende vielleicht sogar entschließen, es für Ihre täglichen Aufgaben am Computer zu nutzen.

Das quelloffene (Open-Source-)Betriebssystem Linux existiert bereits seit vielen Jahren und dient vor allem dem Betrieb von Servern und Webseiten. Den meisten Schülern und Makern kommt es das erste Mal in Form des Raspberry Pi oder vergleichbarer Einplatinencomputer (single board computer, SBC) wie BeagleBone Black oder Intel Galileo unter. Das Buch *Linux für Maker* ist das erste seiner Art, das Linux speziell Makern, und nicht Programmierern oder Administratoren, näherbringt. Durch die gewonnenen Linux-Kenntnisse können Maker spannende Projekte leichter umsetzen.

Weil dieses Buch auf die Makerszene von heute ausgerichtet ist, bezieht es sich auf die Linux-Distribution *Raspbian*, die auf dem Raspberry Pi am meisten verbreitet ist. Nichtsdestotrotz gelten die meisten der hier vorgestellten Konzepte auch für die anderen Linux-Distributionen; auf etwaige Abweichungen weise ich jeweils hin. Dieses Buch konzentriert sich auf die Grundprinzipien, die für Maker relevant sind, und lenkt nicht mit Einzelheiten ab, die für eigene Projekte keine Rolle spielen. Nach der Installation des Betriebssystems behandele ich die Grundfunktionen von Linux, die Bedienung über die Kommandozeile, die Steuerung von Geräten und verrate noch viele Tipps und Tricks, die Ihnen helfen, erfolgreicher voranzukommen.

Im ganzen Buch verstreut finden Sie Abschnitte, die mit »Probieren Sie es selbst« überschrieben sind. Damit können Sie das soeben Gelernte selbst in die Tat umsetzen und dabei weitere Möglichkeiten entdecken, neue Konzepte auszuprobieren. Illustrationen und Screenshots erläutern zusätzlich, was genau Sie auf dem Raspberry Pi zu sehen bekommen, wenn Sie Linux verwenden.

Alle diejenigen, die sich schon immer gefragt haben, wie das mit Linux alles angefangen hat, finden im Anhang A noch eine kurze historische Abhandlung.

Besondere Schreibweisen

Folgende Schreibweisen gelten einheitlich in diesem Buch:

Kursiv Neue Begriffe, URLs, E-Mail-Adressen, Dateinamen und Dateisuffixe

`Konstante Zeichenbreite`
Verwendung in Programmlistings sowie in Texten, die sich auf Programmelemente wie Variablen, Funktionsnamen, Datenarten, Umgebungsvariablen, Anweisungen oder Schlüsselwörter beziehen.

`Konstante Zeichenbreite, fettgedruckt`
Für Befehle oder anderen Text, der genau so vom Benutzer eingegeben werden soll.

`Konstante Zeichenbreite, kursiv`
Für Text, der vom Benutzer mit eigenen Werten bzw. solchen, die sich aus dem Kontext ergeben, ersetzt werden soll.


Dieses Zeichen steht für einen Tipp, einen Vorschlag oder eine allgemeine Bemerkung.


Dieses Zeichen weist auf eine Warnung hin und mahnt zur Vorsicht.

Kontaktmöglichkeiten

Fragen und Anmerkungen zu diesem Buch richten Sie bitte an den dpunkt.verlag unter *info@dpunkt.de*.

Wir haben für dieses Buch auch eine Website eingerichtet, auf der wir Fehlerkorrekturen auflisten, Beispiele liefern und zusätzliche Informationen bereitstellen: *http://www.dpunkt.de/maker_linux*.

Danksagungen

Ich möchte meiner Frau Jennifer und meinen Kindern Stephen, Olivia und James für deren Geduld danken, die sie bei der Verfassung dieses Buches hatten. Viele Nächte und Wochenenden hat dieses Buch zusätzlich zu meinem geschäftigen Alltagsleben erfordert und dabei haben sie mich wunderbar unterstützt. Vielen Dank vor allem an James, der mich 1997 an Linux herangeführt hat. Kennen Sie jemanden, dem Linux oder Raspberry Pi gefallen könnten? Erzählen Sie ihr oder ihm davon!

Für die Unterstützung durch meinen Lektor Patrick bin ich besonders dankbar, ebenso den Mitarbeitern von Maker Media und O'Reilly Media, die mich durch den gesamten Prozess des Schreibens, Bearbeitens und Korrigierens begleitet haben.

Ein großes Dankeschön geht noch an die, die ihre Zeit geopfert haben, dieses Buch durchzulesen, und wertvolle Hinweise geliefert haben: Robert Shaver, Christoph Zimmermann, Jim Kennon, Rashed Harun und Broedy Bowers.

Inhaltsübersicht

Inhaltsverzeichnis

1 Erste Schritte

Der Raspberry Pi ist ein Einplatinencomputer (single board computer, SBC), was – wie der Name schon sagt – bedeutet, dass es sich dabei um einen vollständigen Computer handelt, der auf einer einzigen Leiterplatte (printed circuit board, PCB) Platz findet. Wie die meisten SBCs kann man ihn nicht einfach auspacken, anschließen und loslegen. Er besteht aus den gleichen Grundbausteinen wie jeder andere Computer, den wir kennen: einem Hauptprozessor (central processing unit, CPU), einem Arbeitsspeicher, Grafik- und Audiochip sowie Netzwerkanschluss.

Was bei der Auslieferung fehlt, ist einzig das Speichermedium. Von unseren üblichen Computern sind wir meist Festplatten gewohnt, auf denen sowohl unser Betriebssystem als auch unsere Dateien gespeichert sind. Bei einem Raspberry Pi verwendet man SD-Speicherkarten als Hauptspeichermedium. Bevor wir uns also in unsere Abenteuer, die wir mit Linux in Angriff nehmen wollen, stürzen können, müssen wir unser gewünschtes Betriebssystem für unseren Raspberry Pi auf diese SD-Karte kopieren. Damit das optimal funktioniert, sollten wir eine SD-Karte (für den Raspberry Pi 3 eine microSD-Karte) verwenden, auf der mindestens 8 GB Speicherplatz frei ist.

Was ist ein Disk Image?

Bei einem *Disk Image* handelt es sich um eine einzige Datei, die den gesamten Inhalt eines Speichermediums zum jeweiligen Zeitpunkt enthält. Genau wie ein Foto, auf dem viele verschiedene Leute oder Objekte zu sehen sind, kann ein Disk Image viele verschiedene Partitionen, Verzeichnisse und Dateien enthalten.

Da an dieser Stelle leicht Missverständnisse entstehen können, teilen wir diesen Arbeitsschritt in mehrere Einzelschritte auf und führen sie der Reihe nach durch. Dazu gehören das Herunterladen eines komprimierten Disk Image aus dem Internet, das Dekomprimieren dieses Image auf dem eigenen Computer, das Kopieren dieses Image auf die SD-Karte und schließlich das Hochfahren Ihres Raspberry Pi. Sie werden schnell feststellen, dass diese Arbeitsschritte bei anderen SBCs genauso durchgeführt werden, auch wenn die Image-Dateien für das Betriebssystem jeweils unterschiedlich sind.

Im Verlaufe dieses Kapitels werde ich einige Konzepte erwähnen, die für Sie vielleicht noch neu sind, wie zum Beispiel Dateisysteme, Terminalemulation und die Kommandozeile. Keine Sorge, diese werden in Kapitel 2 noch ausführlich behandelt. Fürs Erste müssen wir Ihren Raspberry Pi lediglich zum Laufen bekommen, sodass Sie ihn verwenden können, wenn Sie dieses Buch weiter durcharbeiten. Für den Anfang benötigen Sie einen Desktop-Computer oder Laptop mit einer Internetverbindung.

1.1 Ein Disk Image auswählen und herunterladen

Der beste Ort für das Herunterladen der aktuellen Disk Images für den Raspberry Pi ist die Website der Raspberry Pi Foundation: *http://raspberrypi.org/downloads*. Wenn Sie das erste Mal auf diese Seite kommen, sehen Sie eine ganze Reihe von Disk Images, aus denen Sie auswählen können. Die beiden, die von der Raspberry Pi Foundation offiziell unterstützt werden, sind NOOBS und Raspbian. *NOOBS* ist im Grunde ein Installer für Raspbian und weitere Betriebssysteme, die auf dem Raspberry Pi betrieben werden können. NOOBS automatisiert gewissermaßen nur einige der Schritte, die in diesem Kapitel beschrieben werden, und ist daher kein eigentliches Betriebssystem, das man verwendet. Stattdessen wird das Betriebssystem im Verlauf der Installation heruntergeladen. *Raspbian* hingegen ist das eigentliche Betriebssystem, sodass man bei der Installation das Disk Image eigenhändig auf eine SD-Karte schreiben kann.

Ich lege Ihnen die Verwendung des Raspbian-Image aus mehreren Gründen ans Herz. Auch wenn die Installation von NOOBS weniger kompliziert erscheint, geht die direkte Installation von Raspbian tatsächlich einfacher und schneller. Außerdem lernt man durch die Installation von

Raspbian, wie man auch jedes andere Disk Image, das man eventuell später ausprobieren möchte, auf seinen Raspberry Pi bekommt. Immerhin geht es in diesem Buch um das Lernen von neuen Dingen. Zudem hat das Kopieren von Raspbian den Vorteil, dass die Festplatten- und Dateistruktur auf der SD-Karte standardisiert ist, was das Erstellen von Sicherungskopien von diesem Image etwas erleichtert. Solche Sicherungen sind dringend anzuraten, damit die eigene Arbeit nicht umsonst war, falls es durch Schäden an der SD-Karte oder dem Raspberry Pi zu Datenverlusten kommt.

Klicken Sie daher auf der Website auf den Link für Raspbian und laden Sie die Datei Raspbian Stretch herunter. Wenn Sie sich sicher sind, dass Sie die Desktop-Umgebung (siehe Kap. 3) nicht benötigen werden, können Sie die Variante Raspbian Stretch Lite herunterladen, die unter anderem durch das Fehlen der Desktop-Umgebung eine geringere Dateigröße hat. Speichern Sie die ZIP-Datei an einem Ort auf Ihrem Rechner, den Sie leicht wiederfinden, etwa in den Ordner *Downloads*. Während dieses Buches geschrieben wurde, hieß diese Datei *2017-09-07-raspbian-stretch.zip* und umfasste 1,6 GB.

1.2 Das Disk Image entpacken

Damit sie schneller heruntergeladen werden können, sind Disk Images stets komprimiert. Die Komprimierung wird im Prinzip durch einen Algorithmus bewerkstelligt, der leere Dateibereiche und doppelt vorhandene Informationen herausnimmt. Man denke beispielsweise an die vielen Leerzeichen zwischen den Wörtern eines Berichts oder eines Briefes an einen Freund. Deren Weglassen verkleinert das Produkt zwar deutlich, macht es aber umso schwerer zu lesen. Damit man also dieses Disk Image verwenden kann, muss man es zunächst dekomprimieren bzw. entpacken. Dazu benötigt man entweder eine eigenständige Software oder sie ist, wie inzwischen in den meisten Fällen, bereits auf dem Desktop-Computer oder Laptop vorhanden.

1.3 Windows

In den aktuellen Versionen von Windows ist die Software zum Dekomprimieren bereits im Datei-Explorer enthalten. Man öffnet sie und navigiert durch sein Dateisystem zu dem Ordner, in dem man die dekomprimierten Dateien gespeichert haben möchte und den man leicht wiederfindet. Dies kann durchaus derselbe Ordner sein, in dem man die ZIP-Datei ursprünglich gespeichert hat (siehe Abb. 1–1 für ein Beispiel, wie das aussehen kann).

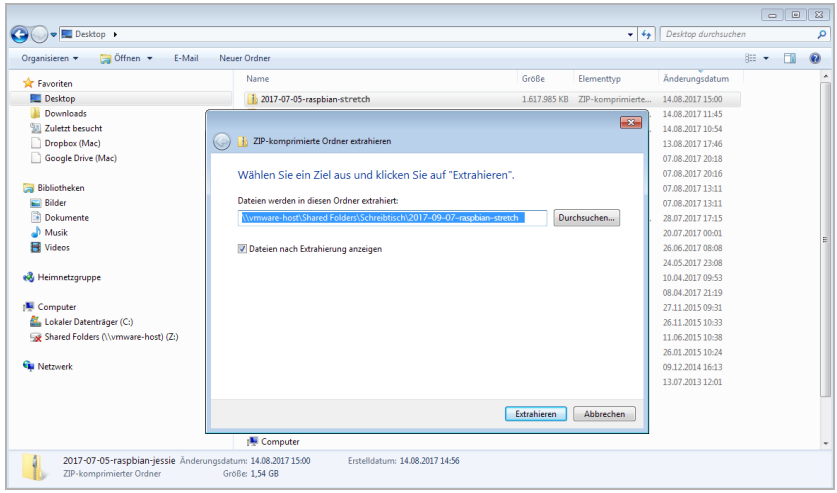


Abb. 1–1 Dekomprimierung des Disk Image mit Windows

1.4 macOS

Sobald Sie die ZIP-Datei mit der aktuellen Version von Safari heruntergeladen haben, wird sie automatisch in dem Ordner *Downloads* gespeichert und sofort automatisch dekomprimiert. Selbst wenn Sie mit einer älteren Version von macOS arbeiten, enthält sie das sogenannte Archivierungsprogramm, das sich sofort öffnet, sowie Sie auf die ZIP-Datei doppelklicken, und sie an Ort und Stelle entpackt (siehe Abb. 1–2).

Sobald der Vorgang abgeschlossen ist, sehen Sie die entpackte *.img*-Datei im selben Verzeichnis, in dem Sie die ZIP-Datei geöffnet haben.

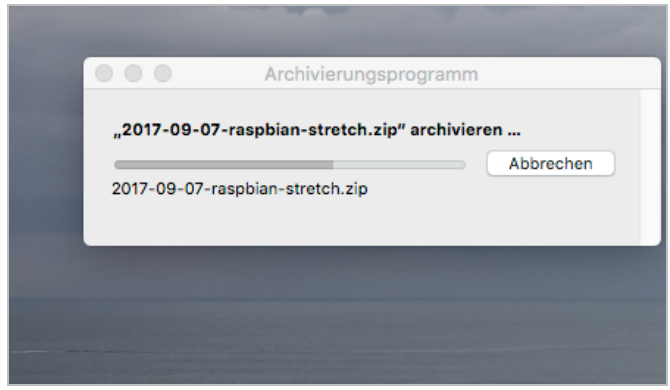


Abb. 1–2 Das Archivierungsprogramm in macOS in Aktion

1.5 Linux

Die meisten Linux-Distributionen enthalten ebenfalls Programme zum Dekomprimieren von gepackten Dateien. Vom Desktop aus können Sie den Dateibrowser öffnen und auf die von Ihnen heruntergeladene ZIP-Datei doppelklicken. Dadurch öffnet sich der Archivmanager, der Ihnen die Datei dekomprimiert (siehe Abb. 1–3).

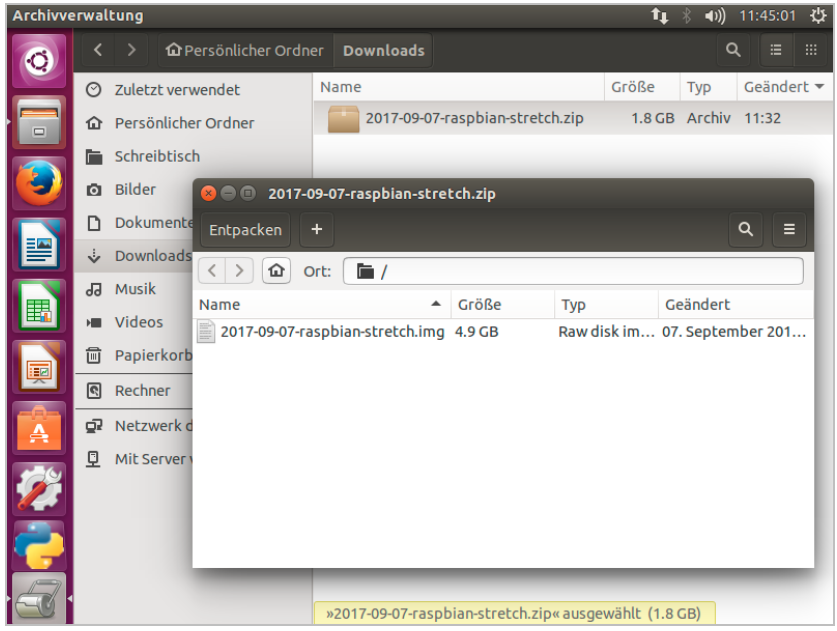


Abb. 1–3 Der Archivmanager von Linux

Sie können diese Aufgabe allerdings auch genauso gut von der Kommandozeile aus lösen, indem Sie im Terminal eingeben:

```
cd Downloads
unzip 2017-09-07-raspbian-stretch.zip
```

In diesem Fall wird davon ausgegangen, dass Sie Ihre ZIP-Datei im üblichen Verzeichnis heruntergeladen haben und auch der Dateiname stimmt. Nun brauchen Sie etwas Geduld, da es sich um eine große Datei handelt und das Entpacken entsprechend Zeit benötigt. In Abbildung 1–4 sehen Sie, wie das aussieht.

Kommandozeile noch unklar?

Sie fühlen sich unsicher wegen der Kommandozeile? Sie wird in diesem Buch eine große Rolle spielen. In den nächsten Kapiteln werden Sie mehr über sie erfahren.

```
anewcomb@anewcomb-VirtualBox ~ $ cd Downloads
anewcomb@anewcomb-VirtualBox ~ /Downloads $ unzip 2017-09-07-raspbian-stretch.zip
Archive:  2017-09-07-raspbian-stretch.zip
      Inflating:  2017-09-07-raspbian-stretch.img
anewcomb@anewcomb-VirtualBox ~ /Downloads $
```

Abb. 1–4 Dekomprimierung des Disk Image über die Kommandozeile von Linux

1.6 Das Disk Image auf eine SD-Karte kopieren

Vorsicht bei diesem Schritt, da durch ihn sämtliche zuvor auf dieser SD-Karte gespeicherten Dateien gelöscht werden.

1.7 Windows

Was das Schreiben eines Disk Image auf eine Speicherkarte angeht, ist Windows zurzeit das einfachste Betriebssystem. Allerdings muss man im Gegensatz zu macOS und Linux zuvor eine Software herunterladen. Öffnen Sie dazu Ihren Browser und laden Sie das Programm Win32 Disk Imager herunter (*https://sourceforge.net/projects/ win32diskimager/*).

Installieren Sie die Anwendung durch einen Doppelklick auf die heruntergeladene Datei. Nach erfolgter Installation legen Sie Ihre SD-Karte in den Computer ein und merken Sie sich den für diese Karte von Windows zugewiesenen Laufwerksbuchstaben. Öffnen Sie nun die soeben installierte Anwendung und überprüfen Sie als Erstes, ob der vom Programm ausgewählte Laufwerksbuchstabe stimmt, also tatsächlich die SD-Karte gemeint ist. Der Win32 Disk Imager wählt zwar in der Regel zuverlässig ausschließlich SD-Karten aus, doch ist es besser, hier auf Nummer sicher zu gehen, da man schließlich auf keinen Fall das falsche Laufwerk formatieren möchte.

Klicken Sie anschließend auf das blaue Ordnersymbol, um die im vorherigen Schritt entpackte Image-Datei auszuwählen. Abbildung 1–5 zeigt, wie das in etwa aussehen soll.

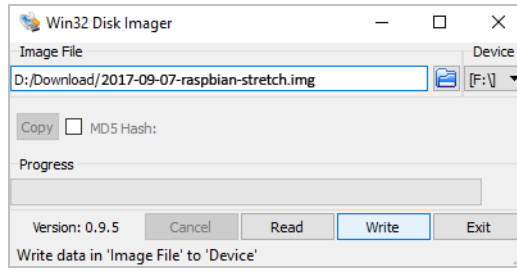


Abb. 1–5 Die Oberfläche des Win32 Disk Imager

Nachdem Sie die richtige Datei ausgewählt haben, klicken Sie auf den Button *Write*. Dadurch werden sämtliche Daten auf der SD-Karte überschrieben. Sobald der Vorgang beendet ist, schließen Sie Win32 Disk Imager. Öffnen Sie nun den Datei-Explorer, machen Sie auf den Laufwerksbuchstaben der SD-Karte einen Rechtsklick und wählen Sie *Auswerfen*. Gehen Sie beim Herausziehen einer SD-Karte stets auf diese Weise vor, um sicherzugehen, dass eventuelle Schreibvorgänge im Hintergrund abgeschlossen sind.

1.8 macOS

Ebenso wie Linux enthält macOS bereits sämtliche Software, die man benötigt, um ein Image auf eine SD-Karte zu schreiben. Allerdings muss man dazu die Kommandozeilen benutzen. Öffnen Sie dazu den Finder und klicken Sie auf Programme/Dienstprogramme/Terminal (siehe Abb. 1–6).

Legen Sie nun die SD-Karte ein und warten Sie ab, bis macOS sie erkannt hat. Geben Sie in der Kommandozeile des Terminals den Befehl `diskutil list` ein, um sich eine Liste sämtlicher mit Ihrem Mac verbundenen Laufwerke ausgeben zu lassen:

```
diskutil list
```

Finden Sie nun heraus, welches Laufwerk (nicht die Partition) Ihrer SD-Karte entspricht (zum Beispiel `disk1`, nicht aber `disk1s1`), wie in Abbildung 1–7 zu sehen.

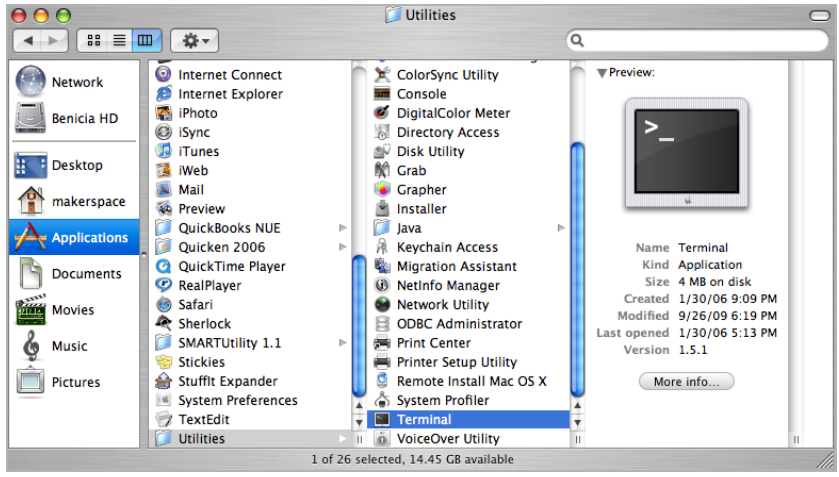


Abb. 1–6 Auffinden des Programms Terminal in macOS

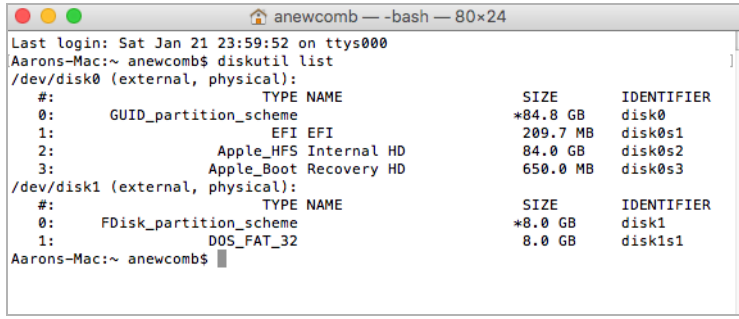


Abb. 1–7 Beispiel einer Ausgabe nach dem Befehl `diskutil list`

In diesem Fall hatte ich eine 64 GB SD-Karte eingelegt, die macOS als `disk1` erkannt hat.

Um die SD-Karte für das Herüberkopieren vorzubereiten, übergeben Sie die Option `unmountDisk` und den Namen der SD-Karte an `diskUtil` (siehe Abb. 1–8):

```
diskutil unmountDisk /dev/disk1
```

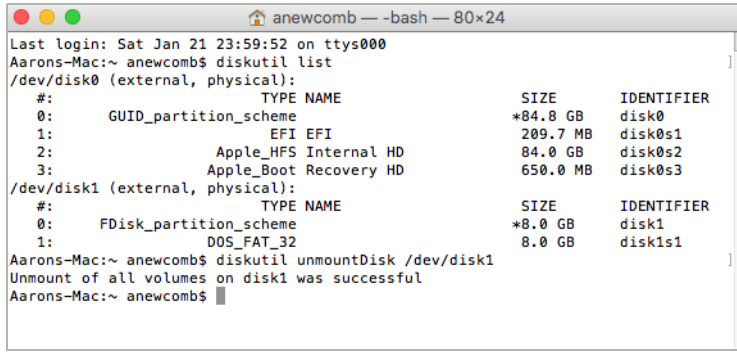


Abb. 1–8 Deaktivierung eines Laufwerks mit dem Befehl `diskutil list`

Nun geht es an das Herüberkopieren des Disk Image auf die SD-Karte. Dazu benutzen wir den Befehl *data duplicator*, also `dd`. Bitte achten Sie auf die richtige Laufwerksnummer, damit Sie nicht aus Versehen Ihr Systemlaufwerk überschreiben! Darüber hinaus benötigen Sie für diesen Befehl den *super user do* (`sudo`). Mit `sudo` kann ein normaler Benutzer auf sichere Weise einen Befehl ausführen, der normalerweise Administratorrechte erfordert:

```
sudo dd if=Desktop/2017-09-07-raspbian-stretch.img of=/dev/
rdisk1 bs=1m
```

Da Sie `sudo` verwendet haben, wird von Ihnen an dieser Stelle eine Passworteingabe verlangt.

Falls Sie GNU coreutils installiert haben, kann dies zu einer Fehlermeldung führen:

```
dd: invalid number '1m'
```

Kümmern Sie sich an dieser Stelle noch nicht darum, was dies alles zu bedeuten hat. Alles, was Sie nun tun müssen, ist, auf folgende Weise die Blockgröße in dem Bereich `bs=` des Befehls auf `1M` zu stellen:

```
sudo dd if=Desktop/2017-09-07-raspbian-stretch.img of=/dev/
rdisk1 bs=1M
```

Dies kann je nach Größe der Image-Datei einige Minuten in Anspruch nehmen. Um den Fortschritt dieses Prozesses zu untersuchen, können Sie Ctrl-T drücken und ein sogenanntes SIGINFO-Signal abgeben (siehe Abb. 1–9). Sollte dies nicht funktionieren, können Sie statt des Befehls `rdisk` den Befehl `disk` ausprobieren.

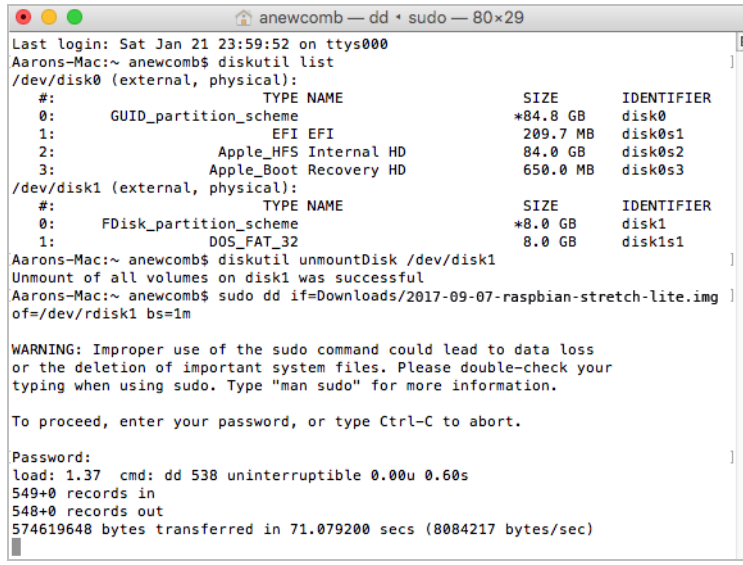


Abb. 1–9 Verwendung des `data duplicator` in macOS

In diesem Fall wurden mit dem Befehl `dd` 549 Blöcke von je 1 MB übertragen. Sobald der Vorgang abgeschlossen ist, zeigt das Terminal Ihnen wieder die Eingabezeile. Bevor Sie Ihre SD-Karte herausnehmen, geben Sie noch einen letzten Befehl ein:

```
sync
```

Auf diese Weise ist sichergestellt, dass alle Schreibvorgänge, die im Hintergrund auf die SD-Karte stattgefunden haben, tatsächlich abgeschlossen sind. Jetzt können Sie Ihre SD-Karte herausnehmen.

1.9 Linux

Bevor Sie in Linux Ihre SD-Karte in Ihren Computer stecken, geben Sie im Terminal folgenden Befehl ein:

```
sudo fdisk -l
```

Daraufhin werden Ihnen sämtliche physikalischen Laufwerke angezeigt, die mit Ihrem System verbunden sind. Notieren Sie sich die Laufwerksgrößen und deren Bezeichnung (siehe Abb. 1–10).

```
anewcomb@anewcomb-VirtualBox ~ $ sudo fdisk -l

Disk /dev/sda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders, total 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000c6e58

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048    37748735    18873344   83  Linux
/dev/sda2        37750782    41940991     2095105    5  Extended
/dev/sda5        37750784    41940991     2095104   82  Linux swap / Solaris
anewcomb@anewcomb-VirtualBox ~ $
```

Abb. 1–10 Auffinden der physikalischen Laufwerke mit `fdisk`

Stecken Sie nun Ihre SD-Karte in Ihren Linux-PC und warten Sie kurz ab, bis sie erkannt wurde. Mitunter aktiviert sie sich dabei selbsttätig und zeigt die darauf befindlichen Partitionen an. Nun geben Sie den Befehl nochmals ein und sehen dann das neue Laufwerk (siehe Abb. 1–11). Notieren Sie sich Laufwerksnamen und -größe. Der Laufwerksname ist im nächsten Schritt wichtig.

```
anewcomb@anewcomb-VirtualBox ~ $ sudo fdisk -l

Disk /dev/sda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders, total 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000c6e58

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048    37748735    18873344   83  Linux
/dev/sda2        37750782    41940991     2095105    5  Extended
/dev/sda5        37750784    41940991     2095104   82  Linux swap / Solaris

Disk /dev/sdb: 7985 MB, 7985954816 bytes
231 heads, 28 sectors/track, 2411 cylinders, total 15597568 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000cc086

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048    15597567     7797760    b  W95 FAT32
anewcomb@anewcomb-VirtualBox ~ $
```

Abb. 1–11 Auffinden der physikalischen Laufwerke mit `fdisk` (Fortsetzung)

Jetzt geben Sie die folgenden Befehle ein, um die Image-Datei auf Ihre SD-Karte zu schreiben. Achten Sie darauf, dass Sie tatsächlich den Laufwerksnamen Ihrer SD-Karte und nicht aus Versehen den Ihres Systemlaufwerks verwenden:

```
sudo unmount /dev/IhrSDKartenname*
```

Setzen Sie statt */IhrSDKartenname* den Laufwerksnamen Ihrer SD-Karte ein und fügen Sie hinten einen * an.

```
cd ~/Downloads

sudo dd if=2017-09-07-raspbian-stretch.img of=/dev/sdb bs=4M
```

Hier müssen Sie im Bereich `if =` des `dd`-Befehls den Dateinamen und im Bereich `of =` den Laufwerksnamen Ihrer SD-Karte einsetzen bzw. austauschen. Je nach Geschwindigkeit von Computer und SD-Karte kann die Bearbeitung des letzten Befehls bis zu 10 Minuten dauern. Sobald sie abgeschlossen ist, wartet das Terminal wieder auf eine Eingabe (siehe Abb. 1–12).

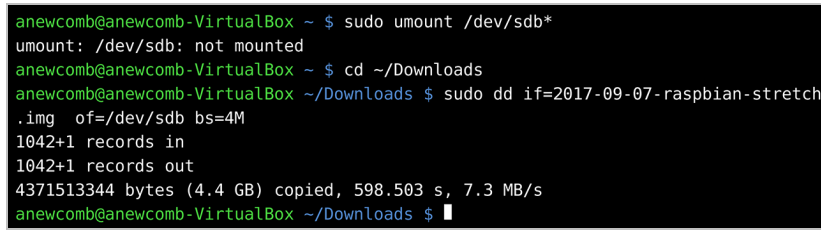


Abb. 1–12 Verwendung des `data duplicator` in Linux

Bevor Sie Ihre SD-Karte herausnehmen, geben Sie noch einen letzten Befehl ein:

```
sync
```

Auf diese Weise wird sichergestellt, dass alle im Hintergrund stattgefundenen Schreibvorgänge abgeschlossen sind. Jetzt können Sie Ihre SD-Karte entfernen.

1.10 Ihren Raspberry Pi das erste Mal hochfahren

Jetzt ist der magische Moment gekommen. Sobald Sie ein Netzgerät eingesteckt und mit der Steckdose verbunden haben und Ihr Monitor angeschlossen ist, können Sie Ihre SD-Karte in den Raspberry Pi einlegen. Stellen Sie sicher, dass Ihr Netzteil mindestens 2 A liefert – besser 2,5 A, die für den Raspi 3 empfohlene Größe. Diese Angabe finden Sie meist auf dem Netzteil aufgedruckt. Aktuelle Smartphone-Ladegeräte sollten funktionieren und auch der Anschluss an einen PC oder Laptop über USB reicht häufig aus.

Während Ihr Raspberry Pi hochfährt, sehen Sie, dass oben auf dem Bildschirm vier Raspberry-Pi-Logos zu sehen sind und jede Menge grüner Text auf schwarzem Grund über den Bildschirm läuft. Keine Sorge, das ist alles ganz normal. Linux ist lediglich dabei, das Betriebssystem hochzufahren und dabei einige Dienste zu starten (mehr darüber in Kap. 2).

Sobald Ihr Raspberry Pi hochgefahren ist, sollten Sie, bevor Sie mit dem Rest dieses Buchs fortfahren, einige Einstellungen verändern, die bei anderen SBCs mitunter nicht nötig sind. Bitte beachten Sie, dass die Entwickler des Betriebssystems dieses ständig aktualisieren. Zukünftige Versionen der Linux-Distribution Raspbian benötigen daher womöglich nicht mehr alle der hier vorgeschlagenen Änderungen.

1.11 Das Dateisystem erweitern

Bei der Erzeugung der Image-Datei wurde darauf geachtet, sie möglichst klein zu halten, damit sie schnell herunterzuladen ist. Wenn Sie Ihr System das erste Mal hochfahren, haben Sie daher insgesamt nur 4 GB Speicherplatz, von dem lediglich 700 MB frei verfügbar sind, obgleich Ihre SD-Karte sehr viel größer ist. Jetzt, wo das Image auf Ihre SD-Karte geladen ist, möchten Sie wahrscheinlich das Dateisystem erweitern, um den zusätzlichen Speicherplatz auch nutzen zu können. In der aktuellen Version von Raspbian geschieht dies beim ersten Hochfahren automatisch. Sie werden sehen, dass nach dem ersten Hochfahren des Systems es noch einmal neu gestartet wird und dabei die neue Größe des Dateisystems erkannt wird.

Bei früheren Versionen von Raspbian war es nötig, auf das Icon im Monitor zu klicken das wie ein dunkler Monitor aussah. Dadurch öffnete sich das Terminal, in dem wir Zugang zur Kommandozeile haben. In dem sich geöffneten Fenster musste man bei den älteren Versionen von Raspbian dann eingeben:

```
sudo raspi-config
```

Daraufhin öffnet sich das Programm Raspbian configuration, wie in Abbildung 1–13 zu sehen.

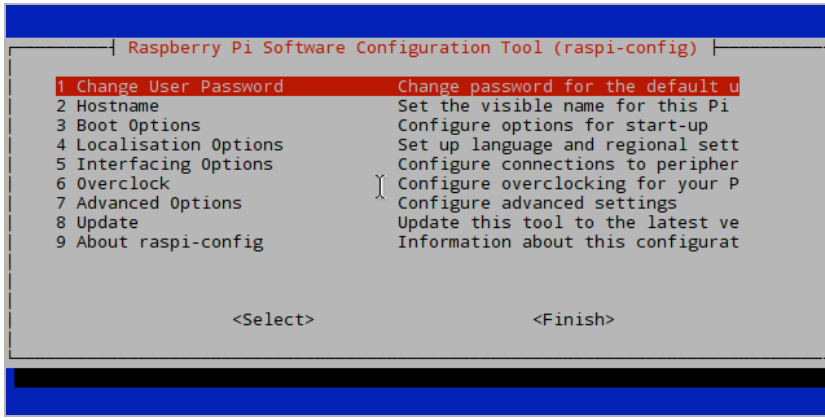


Abb. 1–13 Das Fenster `raspi-config`.

Bei markierter Optionen *Expand Filesystem* drücken Sie nun die Enter-Taste, woraufhin sich nach ein paar Sekunden ein neues Fenster öffnet, um Ihnen mitzuteilen, dass die Größe des Root-Dateisystems geändert wurde.

1.12 Ländereinstellungen ändern

Standardmäßig wird das Image von Raspbian mit den Einstellungen von Großbritannien ausgeliefert, da die Raspberry Pi Foundation dort beheimatet ist. Deshalb möchten Sie bestimmt Ihr Exemplar auf Ihre eigene Zeitzone, Tastaturlayout und Sprache einstellen. Glauben Sie mir, es kann sehr ärgerlich sein, wenn man auf seiner Tastatur ein Anführungszeichen eingibt und der Raspberry Pi dies als @-Symbol interpretiert.

Im selben Konfigurationsfenster wählen Sie mit der Pfeiltaste nach unten oder durch Eingabe der entsprechenden Zahl auf Ihrer Tastatur die *Localisation Options* aus. Wir ändern an dieser Stelle gleich alle drei Optionen in diesem Menü und beginnen mit der Einstellung des Landes. Drücken Sie hierzu wieder die Entertaste, dadurch kommen Sie in ein Fenster, das so ähnlich wie in Abbildung 1–14 aussieht.

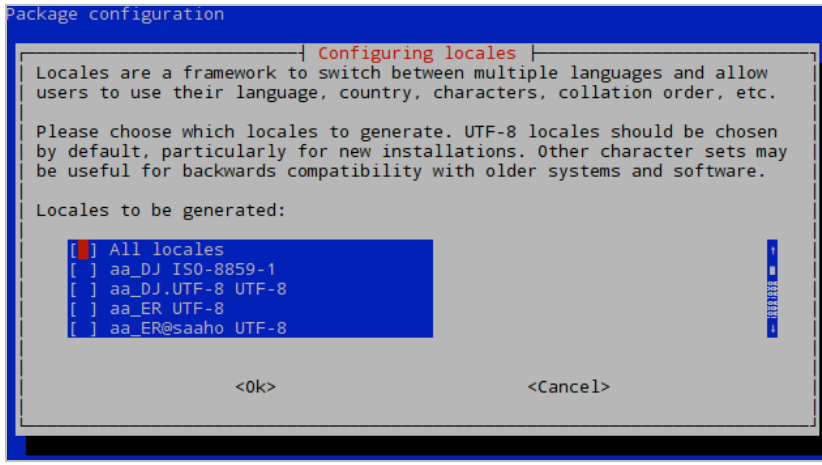


Abb. 1–14 Das Menü locales in raspi-config

Es mag an dieser Stelle verlockend sein, hier einfach »All locales« auszuwählen, statt seine Ländereinstellung zu suchen. Dies hätte allerdings zur Folge, dass Ihr Raspberry Pi nun sehr lange brauchen würde, um sämtliche Sprachpakete zu installieren, die sie fast alle niemals benötigen werden. Machen Sie sich daher besser die Mühe, die von Ihnen am meisten verwendete Sprache auszuwählen. Wie schon in der Beschreibung erwähnt, empfiehlt es sich, unter diesen diejenige auszuwählen, die mit UTF-8 endet. Für Deutschland wäre dies also *de_DE.UTF8*. Ist das gewünschte Sprachpaket mit dem roten Balken ausgewählt, drückt man die Leertaste. Ich werde mich in diesem Buch nicht mit der Erläuterung sämtlicher Ländercodes aufhalten können, doch wenn es Sie näher interessiert, können Sie eine vollständige Liste der Ländercodes online einsehen (*https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes*).

Sobald Sie Ihr Land ausgewählt haben, drücken Sie die Entertaste, woraufhin Sie in das nächste Fenster gelangen, in dem sie Ihre Standardländereinstellung auswählen. Hier wählen Sie mit der Pfeiltaste am besten wieder diejenige, die Sie vorher ausgesucht haben, und bestätigen Sie mit Enter. Jetzt macht sich Ihr Raspberry Pi daran, das entsprechende Sprachpaket zu installieren, um sich dann wieder mit dem Fenster von raspi-config zurückzumelden.

Dort geht es nun weiter mit der Einstellung Ihrer Zeitzone. Gehen Sie wieder in die *Localisation Options* und wählen Sie diesmal *Change Timezone*. Was Sie dann sehen, sollte Abbildung 1–15 entsprechen.

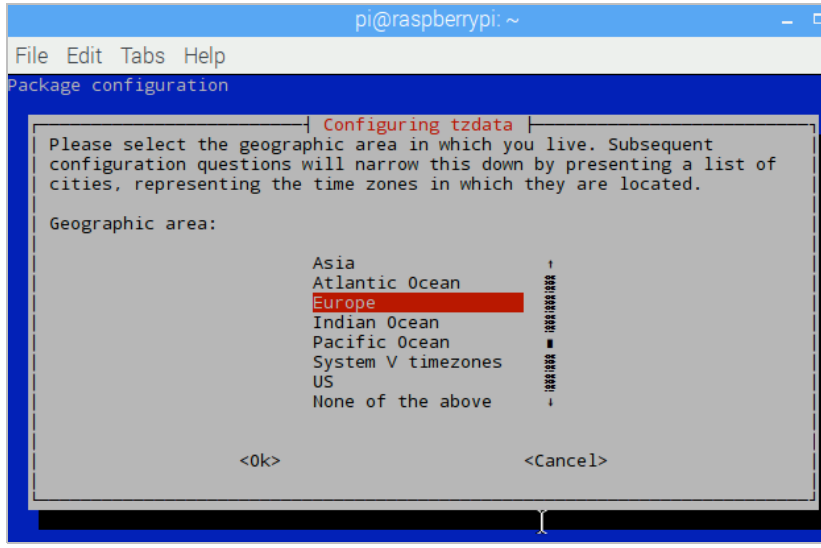


Abb. 1–15 Einstellung der Zeitzone in raspi-config

Wählen Sie hier wieder mithilfe der Pfeiltasten und der Entertaste Ihre Zeitzone aus. Für Deutschland wählen Sie zunächst Europa und tippen dann den Anfangsbuchstaben der für die Zeitzone maßgeblichen Stadt (z.B. B für Berlin) ein oder gehen mit den Pfeiltasten durch die (lange) Liste. Sind Sie bei der richtigen Stadt, wird nach Drücken der Entertaste die Zeitzone eingestellt und man kommt wieder auf die Seite mit allen Einstellungen.

Zu guter Letzt, und das ist vermutlich sogar am wichtigsten, wählen Sie das passende Tastaturlayout aus. Unter den *Localisation Options* gehen Sie nun zu *Change Keyboard Layout*. Dort erwartet Sie ein Fenster wie in Abbildung 1–16.

Mit den Pfeiltasten treffen Sie auch hier Ihre Auswahl. Meistens genügt es, in Europa das Layout Generic 105-key (Intl) PC auszuwählen. Nach Bestätigung mit der Entertaste kommt man in ein nächstes Fenster.

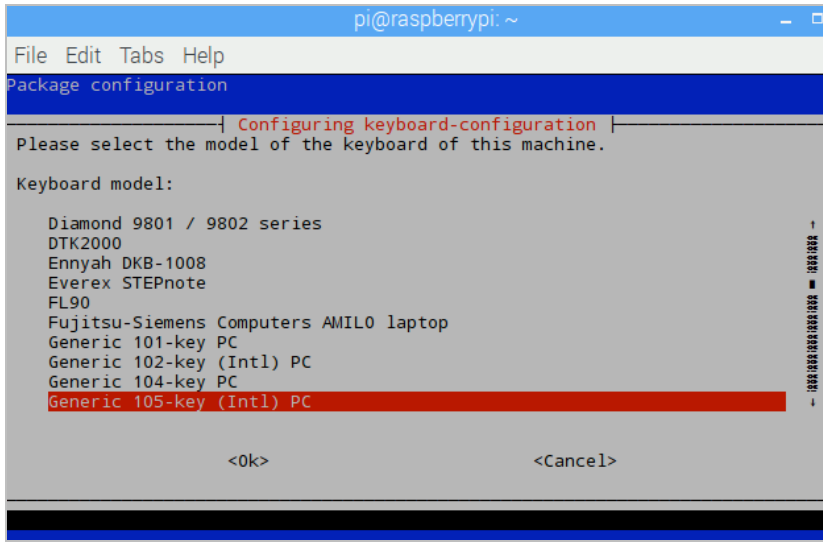


Abb. 1–16 Auswahl eines anderen Tastaturlayouts in raspi-config

Hier sind nun zunächst die britischen Layouts aufgeführt, die im Auslieferungszustand eingestellt sind. Lebt man nicht in Großbritannien, geht man hier also auf Other und drückt wieder Enter (siehe Abb. 1–17).

Im darauf folgenden Fenster wählt man dann sein Land und nach weiterem Drücken der Entertaste seine gewünschte Belegung. Ganz oben findet man ein allgemeines Layout, das in der Regel funktioniert, und wählt es wieder mit der Entertaste aus.

Noch haben wir es nicht ganz geschafft. Wählen Sie jetzt noch die jeweiligen Standardeinstellungen in den folgenden Fenstern, dann landen Sie wieder auf der Seite mit allen Einstellungen.

Drücken Sie nun die Tab- oder Pfeiltaste, um auf den Button *Finish* zu kommen und drücken dann wieder die Entertaste. Jetzt werden Sie gefragt, ob der Raspberry Pi neu starten soll, was Sie mit Wahl von *Yes* und der Entertaste bestätigen. Jetzt startet Ihr Raspberry Pi mit allen ausgewählten Einstellungen neu.

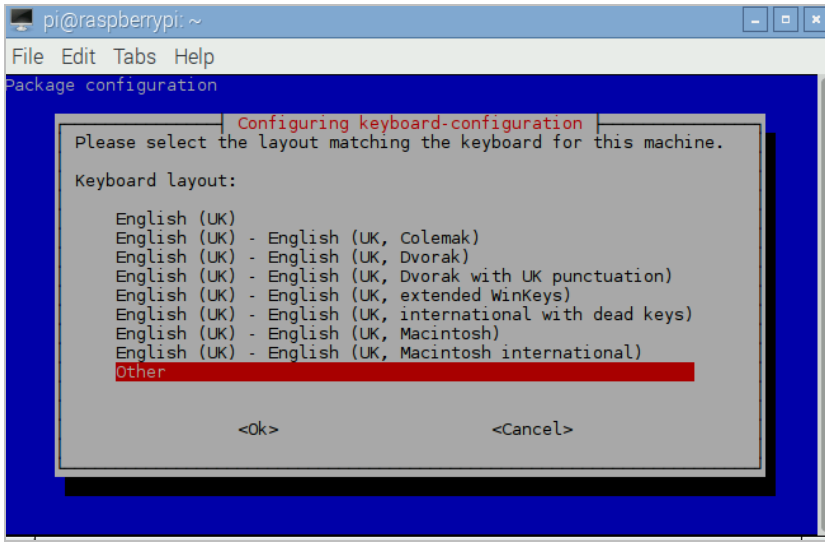


Abb. 1–17 Auswahl eines anderen Tastaturlayouts in `raspi-config` (Fortsetzung)

1.13 Das voreingestellte Passwort ändern

Bevor Sie es vergessen, sollten Sie an dieser Stelle immer das voreingestellte Passwort ändern. Möglicherweise können Sie sich nicht vorstellen, welchen Schaden solch ein kleines System in Ihrem Netzwerk oder gar dem Internet anrichten kann. Selbst die allerkleinsten Systeme können für die Ausführung und vor allem die Verbreitung von Schadsoftware eingespannt werden. Diese Gefahr ist vor allem dann gegeben, wenn Sie das voreingestellte Passwort nicht ändern, da es in aller Regel die erste Stelle des Angriffs darstellt.

Die Änderung des Passworts für den Benutzer »pi« lässt sich über die Kommandozeile einfach bewerkstelligen. Öffnen Sie dazu wie vorhin ein Terminalfenster und geben Sie folgenden Befehl ein:

```
passwd
```

Daraufhin werden Sie aufgefordert, das bis dahin existierende Passwort einzugeben, was nach der ersten Einrichtung »raspberry« lautet, und geben anschließend zweimal Ihr neues Passwort ein, um sicherzugehen, dass es stimmt (siehe Abb. 1–18). Selbstverständlich sollten Sie hier allereinfachste Kombinationen wie »Passwort« oder »12345678« meiden.

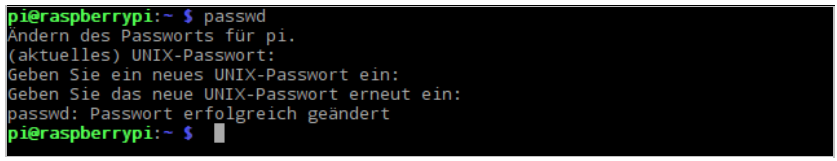


Abb. 1–18 Änderung des voreingestellten Passworts

Jetzt können Sie mit Ihrem Raspberry Pi richtig loslegen!

1.14 Warum dies für Maker wichtig ist

Sobald Sie den Raspberry Pi oder andere SBCs vermehrt für Ihre Projekte einspannen, werden Ihnen diese Prozesse immer leichter von der Hand gehen. Denken Sie daran, dass Sie für den Fall, dass Sie sich total verrannt haben, immer wieder mit einem neuen Disk Image auf Ihrer SD-Karte von vorn beginnen können. Bei vielen der heute erhältlichen SBCs wird bei der Einrichtung des Betriebssystems ähnlich verfahren, sodass Sie nun für spannende Entdeckungen auch mit den anderen hervorragenden Minirechnern gerüstet sind.

2 Grundprinzipien in Linux

Vermutlich denken Sie jetzt: »Prima! Los geht's! Ich möchte mein erstes Projekt starten.« Doch zunächst gibt es da noch eine Reihe von Prinzipien, nach denen das Betriebssystem Linux arbeitet und die es zu verinnerlichen gilt. Im Gegensatz zu einem Mikrocontroller wie dem Arduino ist die Verwendung von Linux in Ihren Projekten deutlich komplexer. Bei Linux handelt es sich schließlich um ein vollwertiges Betriebssystem mit unterschiedlichen Benutzern, Diensten, Dateisystemen und vielen weiteren Funktionen, die es für Maker so vielseitig und leistungsfähig machen (dagegen ist Arduino nur mit einem sehr beschränkten Befehlssatz ausgestattet).

2.1 Der Linux-Desktop

Die meisten Menschen benutzen ihren Computer über eine grafische Benutzeroberfläche (graphical user interface, GUI). Ganz gleich, ob Sie Windows, macOS, Android oder iOS verwenden (ja, mobile Geräte sind im Prinzip auch einfach kleine Computer), das GUI des Desktops ist der Ort, von wo aus Sie alles in Bewegung setzen. Bei Linux ist es genauso. Bis auf ganz wenige Ausnahmen werden die Linux-Distributionen mit einer Desktop-Umgebung ausgeliefert, um den Umgang mit den Programmen einfacher und effizienter zu gestalten. Wenn Sie sich im Internet bewegen, ein Dokument erstellen oder ein Foto bearbeiten wollen, ist der Desktop der richtige Ort dafür.

Kommandozeilen-Browser

Sie können einmal den *Browser Lynx* ausprobieren, um zu sehen wie es ist, über die Kommandozeile im Internet zu surfen.

Bei Einplatinencomputern (SBCs) wie dem Raspberry Pi, die mit Linux betrieben werden, verhält es sich genauso. In Abbildung 2–1 sieht man, wie der Standard-Desktop des Raspberry Pi zur Zeit der Verfassung dieses Buchs aussieht.

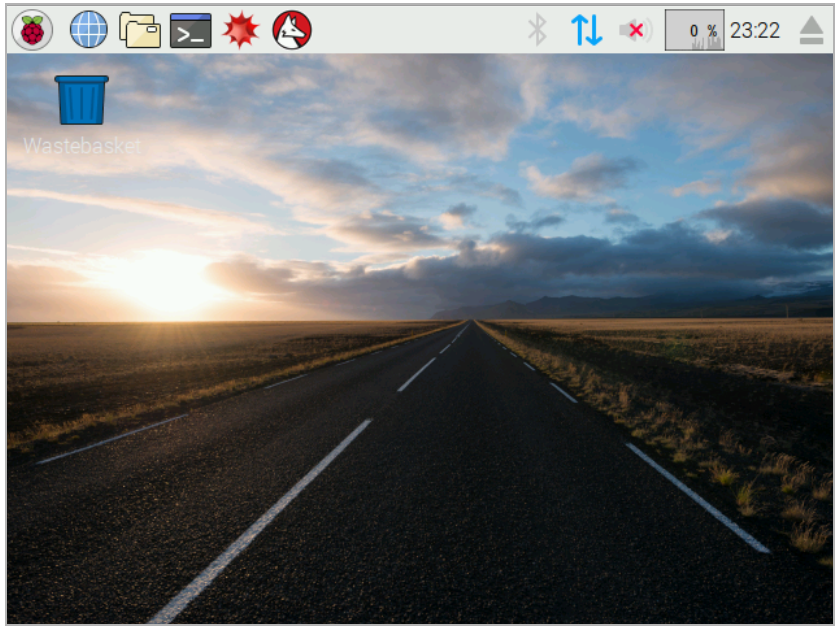


Abb. 2–1 Der Desktop in Raspbian

Ebenso wie es unterschiedliche Linux-Distributionen gibt (siehe Kap. 1), gibt es Linux-Desktops in diversen Ausführungen. Die von Raspbian standardmäßig verwendete heißt *Lightweight X11 Desktop Environment (LXDE)*. *Lightweight*, also leichtgewichtig, bedeutet hier, dass diese Version mit wenig Ressourcen auskommt und daher auf SBCs, die mit weniger leistungsfähigen CPUs und kleineren Arbeitsspeichern ausgestattet sind als moderne Desktop-PCs und Laptops, gut läuft. Weitere beliebte und verbreitete Desktop-Umgebungen unter Linux sind Xfce, Mate, Cinnamon und das von Ubuntu bis dato verwendete Unity. Alle diese Versionen bieten jeweils eigene Möglichkeiten und leicht unterschiedliche Herangehensweisen, mit der Desktop-Umgebung zu arbeiten. Die meisten Desktops besitzen eine Taskleiste mit einem Menü der installierten Programme sowie ein paar Verknüpfungen zu häufig verwendeten Programmen wie dem Browser oder dem Terminal. In der Regel haben sie

auch kleine Statusanzeigen, auf denen der Benutzer bestimmte Parameter wie etwa die Datenrate im Netzwerk, die CPU-Auslastung oder auch einfach die Uhrzeit ablesen kann. Um ein Programm vom Desktop aus zu öffnen, klicken Sie einfach auf die Verknüpfung in der Taskleiste oder auf den entsprechenden Button im Menü.

2.2 Das Terminal oder die Konsole

Falls Sie mit Ihrem Raspberry Pi lediglich im Internet surfen wollen, können Sie an dieser Stelle aufhören zu lesen, da Sie jetzt dazu bereits alles Nötige wissen. Da dieses Buch allerdings für Maker geschrieben ist und Maker einfach gerne Sachen bauen, gibt es mit dem Raspberry Pi und seinem Linux viel mehr zu entdecken als nur den Desktop. Dazu müssen Sie sich allerdings mit dem Terminal (das man gelegentlich auch als Konsole bezeichnet) vertraut machen und das Sie in Abbildung 2–2 sehen.

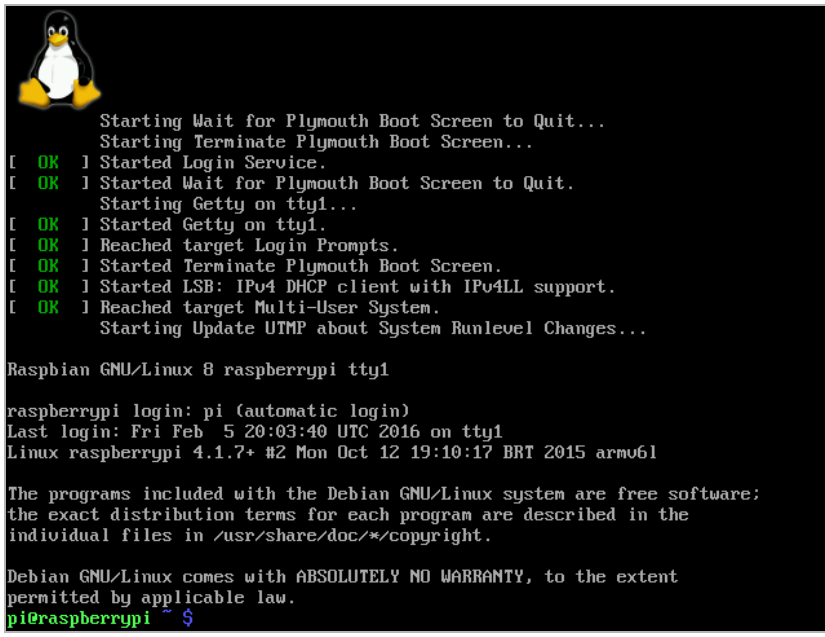


Abb. 2–2 Das Linux-Terminal auf einem Raspberry Pi

Wenn Sie sich schon an einer Raspberry-Pi-Anleitung im Internet entlang gehangelt oder Ihren Pi selbst anhand von Kapitel 1 eingerichtet haben,

wurden Sie ja bereits aufgefordert, einige Befehle im Terminal bzw. in der Konsole einzugeben.

Heißt es Terminal oder Konsole? Worin besteht der Unterschied?

Heutzutage werden diese Begriffe meist synonym verwendet, doch streng genommen ist das Terminal eine Methode, um mit dem Betriebssystem durch Texteingaben in Kontakt zu treten. Bei der Konsole hingegen handelt es sich um eine konkrete Anordnung von Hardware, wie etwa einer Tastatur, einer Maus und einem Monitor, von der man Informationen vom jeweiligen Programm bzw. der Benutzerumgebung erhält. In der Frühzeit der Computertechnik, noch vor dem Aufkommen der grafischen Benutzeroberfläche, aber nach den Lochkarten, war die Konsole die einzige Möglichkeit, direkt mit dem Computer zu kommunizieren. Man kann sich das so merken: Das Terminal erreicht man über die Konsole. Wie dem auch sei, beide Begriffe beziehen sich auf eine textbasierte Kommunikation mit dem Computer, um Programme laufen zu lassen.

Wenn man mit grafischen Benutzeroberflächen den Umgang mit Computern erlernt hat, scheint das Terminal eine archaisch anmutende und umständliche Methode zu sein, seine Aufgaben zu erledigen. Mit zunehmender Kenntnis und Sicherheit im Umgang mit der Konsole bei eigenen Projekten kann man allerdings zu dem Schluss kommen, dass man den Desktop eigentlich gar nicht mehr benötigt (dazu später mehr). Da das Terminal früher die einzige Methode des Computerzugangs war und es auch noch kein Kopieren und Einfügen (copy and paste) gab, haben die Programmierer damals zahlreiche Tastaturkürzel in ihre Produkte eingebaut, die zum Teil heute noch gebräuchlich sind. Dank dieser ist der heutige Einsatz des Terminals einfacher, als er zunächst erscheint.

Vom Desktop aus kann man das Terminal öffnen, indem man auf das entsprechende Icon in der Menüleiste klickt oder im Hauptmenü unter *Zubehör* den Punkt *LXTerminal* auswählt (siehe Abb. 2–3). Da wir gerade auf dem Desktop sind: Das Programm, das Sie soeben geöffnet haben, *emuliert* die Konsole eigentlich und wird deshalb auch als *Terminalemulator* bezeichnet. Wenn Sie also ohne Desktop-Umgebung arbeiten, sehen Sie als Erstes das Terminal, nachdem Sie Ihren Raspberry Pi oder einen anderen SBC gestartet haben. Denjenigen, die es noch nicht gewohnt sind, mit einem Linux-System zu arbeiten, sei gesagt, dass dieses mulmige Gefühl schon bald nachlassen wird.

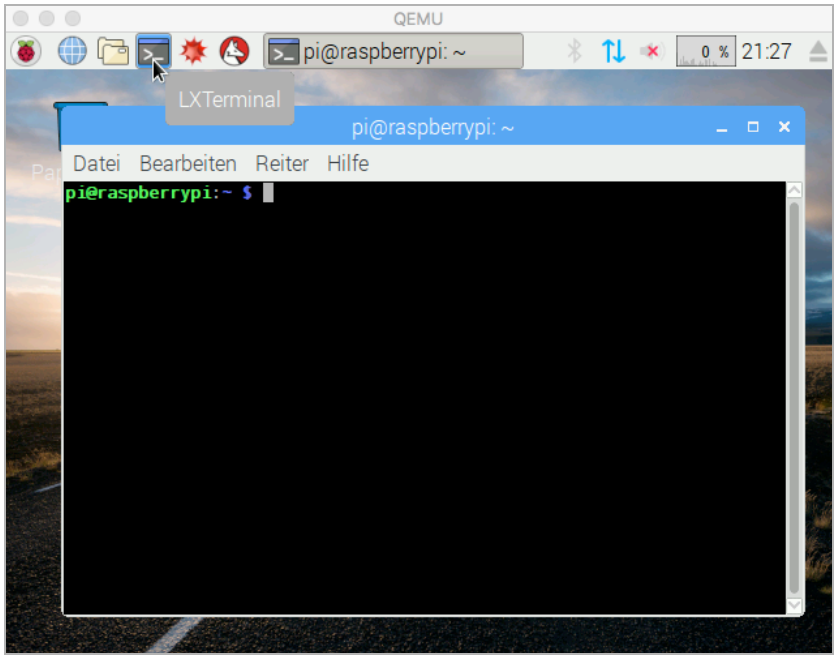


Abb. 2–3 Der Raspberry-Pi-Terminalemulator

2.3 Die Shell auf die Schnelle

Die *Shell* setzt Ihre Texteingaben in der Kommandozeile um, damit das Betriebssystem weiß, was es zu tun hat. Wenn Sie in der Kommandozeile z.B. den Befehl `ls` eingeben, weiß die Shell, wo sich das Programm befindet und wie man es richtig zum Laufen bringt. Die Shell bestimmt auch, wie die Konsolenoberfläche aussieht, und enthält zudem die vielen Tastaturkürzel, von denen wir bereits gesprochen haben. Die Shell kann man gewissermaßen als seinen persönlichen Butler innerhalb des Betriebssystems ansehen. Auf die für Maker wichtigsten Tastaturkürzel werde ich in Kapitel 4 eingehen.

Außerdem kann man ein Skript schreiben, das die Shell Zeile für Zeile ausführt. Man bezeichnet diese einfachen Textdateien mit einigen Befehlen darin, die von oben nach unten abgearbeitet werden, entsprechend als Shell-Skripts.

Mehr als nur eine Version der Shell

Es gibt viele verschiedene Shells, die in Linux laufen, doch in den meisten Distributionen, so auch in Raspbian, kommt die Bourne-Again Shell (bash) zum Einsatz. Sie wurde von Brian Fox für das GNU-Projekt geschrieben, um die Vorgängerversion (sh) von Steve Bourne[a] im Funktionsumfang zu erweitern. Der Grund für die starke Verbreitung von bash ist deren Leistungsfähigkeit. Sie hat Funktionen wie die Vervollständigung von Befehlszeilen und eine Verlaufsspeicherung der Befehle mit Suchfunktion, mit deren Hilfe man leichter findet, was man sucht, und man wiederholbare Befehle einfacher verwenden kann. Man kann bash auch einsetzen, um Skripte als eigenständige Programme laufen zu lassen. In vielen Fällen brauchen Sie in Sachen Linux außer der Shell eigentlich nichts anderes für Ihr Projekt.

a. *https://de.wikipedia.org/wiki/Bash_(Shell)*

2.4 Probieren Sie es selbst

Geben Sie im Terminal Folgendes ein:

```
echo Hallo Welt!
```

Drücken Sie nun die Entertaste und schauen Sie, was passiert.

Jetzt erstellen wir daraus ein Skript, das diesen Befehl zehnmal wiederholt, indem wir eingeben:

```
nano hallo.sh
```

Die Shell weist den Computer an, den Texteditor `nano` aufzurufen und die Datei *hallo.sh* zu bearbeiten. Falls diese Datei noch nicht existiert, wird sie von `nano` angelegt.

Sobald `nano` gestartet wurde, geben Sie ein:

```
#!/bin/bash
for i in `seq 1 10`;
do
echo Hallo Welt!
sleep 1
done
```

Backticks

Die in Fließtexten als Accent grave verwendeten `-Zeichen werden beim Programmieren als Backticks bezeichnet. Man darf sie nicht mit den einfachen Anführungszeichen (Hochkomma, Single Quotes) verwechseln. In Bash-Skripten wird der in Backticks eingeschlossene Code ausgeführt. Auf der deutschen Tastatur findet man sie meist rechts oben, links neben der Rückstelltaste (Backspace), wobei man gleichzeitig die Umschalttaste (Shift) drückt. In Abbildung 2–4 sieht man, wie es aussehen sollte, wenn man den Text im Editor innerhalb des Terminals eingegeben hat. Nachdem Sie den Editor wieder geschlossen haben, geben Sie in der Kommandozeile **seq 1 10** ein und beobachten, was passiert.

Drücken Sie nun Ctrl-X, dann J und abschließend die Entertaste, um das Skript zu speichern. Geben Sie nun ein:

```
sh hallo.sh
```

Drücken Sie erneut Enter. Glückwunsch! Sie haben soeben Ihr erstes Shell-Skript geschrieben.

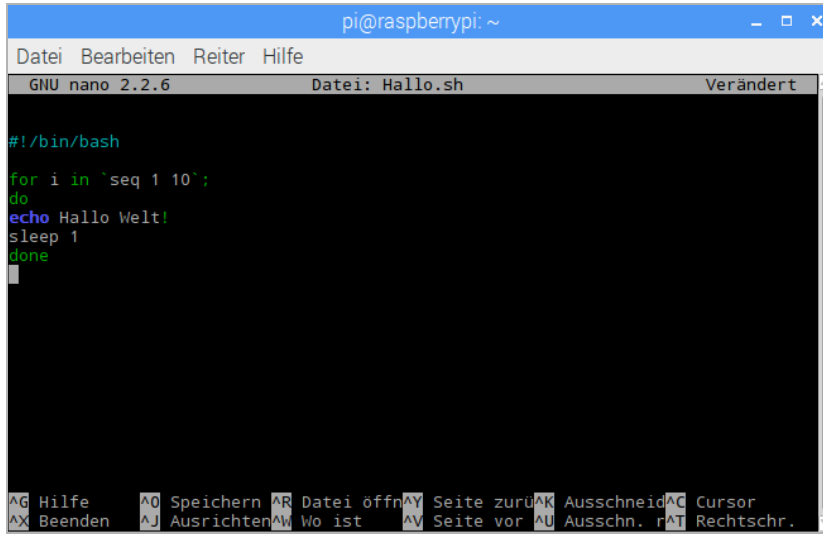


Abb. 2–4 Beispiel für ein Shell-Skript

2.5 Dateisysteme und -strukturen

Worin sich Linux grundsätzlich von anderen Betriebssystemen unterscheidet, ist der Ansatz bei den Dateisystemen. Mit dem Dateisystem bezeichnet man die Art und Weise, wie die Dateien auf dem Computer organisiert sind. Wie bei allen Unix-basierten Systemen bezieht sich fast alles, was man in Linux anstellen möchte, auf irgendeine Datei. Manchmal versteht sich das von selbst. Wenn Sie beispielsweise einen USB-Stick in Ihren Raspberry Pi einstecken, können Sie die Dateien über den Desktop oder das Terminal einsehen. Bei anderen Dingen ist das nicht ganz so einleuchtend. Schließen Sie etwa eine Maus oder eine Tastatur an Ihrem Raspberry Pi an, erzeugt Linux eine Datei, um sich auf dieses Gerät zu beziehen. Mithilfe dieser speziellen Dateien kommuniziert Linux mit diesen Geräten. Aus Dateien holt sich das System auch Informationen über die Hardwarekomponenten, aus denen der Computer besteht, wie etwa der CPU, dem Arbeitsspeicher und vielem anderen. Möchten Sie etwa die aktuelle CPU-Temperatur wissen, lesen Sie einfach die entsprechende Datei aus.

2.6 Probieren Sie es selbst

Geben Sie im Terminal ein:

```
more /sys/class/thermal/thermal_zone0/temp
```

Wenn Sie nun die Entertaste drücken, erscheint eine fünfstellige Zahl, die die CPU-Temperatur in Milligrad Celsius anzeigt. Bei dem Ergebnis

```
38470
```

läuft die CPU Ihres Raspberry Pi bei etwa 38 Grad Celsius.

Maker, die mit Peripherie-Geräten arbeiten und über ihre Programmierung aus dem Betriebssystem Informationen über bestimmte Computerkomponenten einholen möchten, müssen wissen, wie Linux mit Dateien umgeht. Ebenfalls wichtig ist, dass bestimmte Dateien nicht verändert, überschrieben oder gelöscht werden sollten, da das gesamte System dadurch zusammenbrechen könnte und man alles von vorne installieren muss. Glücklicherweise arbeitet Linux mit Benutzern und Zugriffsrechten (siehe Abschnitt 2.7), wodurch solche Fehler ausgeschlossen werden können.

Sie können die Dateien und Ordner einsehen, indem Sie im Desktop den Dateimanager aufrufen (Ordnersymbol in der Leiste oben) oder im Terminal Folgendes eingeben[1]:

```
ls /
```

Der Schrägstrich (Forward Slash) selbst steht für den obersten Ordner in der gesamten Hierarchie des Dateisystems, den Root-Ordner. In Windows würde das dem Laufwerk C: bzw. C:\ entsprechen.

Verwirrung bei Schrägstrichen

In Windows werden mit dem rückwärtsgewandten Schrägstrich (\) Ordner und Dateinamen in Pfadangaben getrennt. Bei Linux geschieht das mit dem vorwärtsgewandten Schrägstrich (/).

Die Struktur des Linux-Dateisystems

Nachfolgend ist die grundsätzliche Struktur des Root-Dateisystems eines Raspberry Pi aufgelistet. Die ***fett markierten Verzeichnisse*** enthalten wichtige Informationen bzw. Programmdaten, die nicht gelöscht oder verändert werden sollten, solange man sich nicht völlig sicher ist, was man dadurch auslöst:

`/`	Dies ist das Root-Dateisystem.
`/bin`	Für Linux essenzielle Programme befinden sich hier.
`/boot`	Hier befinden sich die Dateien, die Linux zum Hochfahren benötigt.
`/dev`	Die Gerätedateien findet man in diesem Verzeichnis.
`/etc`	Hier befinden sich viele Konfigurationsdateien für das Betriebssystem und andere Programme.
/home	Hier werden für die Homeverzeichnisse des Benutzers Unterverzeichnisse angelegt. Sind Sie als der voreingestellte Benutzer »pi« auf Ihrem Raspberry Pi eingeloggt, starten Sie im Terminal unter */home/pi*.
`/lib`	Dieser Ordner ist für Bibliotheken (libraries) und Dateien, die zur Unterstützung von Programmen dienen.
/media	Externe Datenträger erhalten hier ihr eigenes Verzeichnis, sobald sie an Ihrem Computer angeschlossen werden.

→

1) Achtung: Kleines L (l), nicht Ziffer 1; Leerzeichen vor dem Schrägstrich.

/mnt	An dieser Stelle werden andere Dateisysteme geöffnet. Meist ist dieser Ordner daher anfangs leer.
/opt	Optionale Software bzw. Programme werden manchmal in diesem Verzeichnis installiert.
/root	Hier handelt es sich um das Homeverzeichnis des »root«-Benutzers. Aus Sicherheitsgründen wird es von den anderen Benutzern getrennt verwaltet.
/run	Hier werden Informationen über das laufende System gespeichert.
/sbin	In diesem Ordner finden Sie wichtige Befehlsfolgen oder Programme, die »sicher« sind und zum Betrieb Root-Zugang erfordern.
/srv	Bestimmte serverspezifische Informationen werden manchmal hier abgelegt. Meist ist dies Verzeichnis anfangs leer und gelegentlich wird es von Web- und FTP-Servern genutzt.
/sys	Hier befinden sich Informationen über die mit dem System verbundenen Geräte.
/tmp	Temporäre Dateien werden oftmals hier angelegt und bei Neustart gelöscht.
/usr	In diesem Ordner werden zusätzliche Binaries und Programme gespeichert, die in der Regel allen Benutzern zugänglich sind. Manche von ihnen benötigen allerdings dennoch Root-Zugang, um lauffähig zu sein.
/var	Hier werden variable Daten und Dateien aufbewahrt. Dazu zählen beispielsweise Pufferdaten von Druckern, bevor sie an den Drucker gesandt werden, Spielstände und Logdateien.

2.7 Benutzer und Gruppen

Die Möglichkeit, mehrere Benutzer mit unterschiedlichen Profilen und Einstellungen verwalten zu können, ist bei den Betriebssystemen Windows, macOS und Android noch relativ neu. Doch weil Linux auf UNIX basiert, was ja ein Serverbetriebssystem ist, wurden seit jeher mehrere Benutzer und Benutzergruppen unterstützt. Dies ist vor allem vor dem Hintergrund sinnvoll, dass es seine Wurzeln in der Methode des Timesharings als Mehrbenutzersystem (siehe Kap. 1) hat. Bei den meisten Betriebssystemen von SBCs ist bereits ein Standardbenutzer angelegt.

Beim C.H.I.P. heißt er entsprechend »chip«. Für die üblichen Aufgaben ist es ratsam, diesen Standardbenutzer zu verwenden. Sie können allerdings weitere anlegen, wenn sie erforderlich sein sollten – darüber erfahren Sie mehr in Kapitel 6.

Die mit Linux ausgestatteten SBCs loggen Sie automatisch ein, wenn Sie das System im Desktop-Modus betreiben. Das Passwort brauchen Sie dann nur für besondere Befehle oder wenn Sie sich im Terminal einloggen müssen. Das voreingestellte Passwort für den Benutzer »pi« ist »raspberry«. Verwenden Sie einen anderen SBC, können Sie den voreingestellten Benutzer und dessen Passwort in der Regel der Kurzanleitung für den ersten Betrieb entnehmen. Installieren Sie Linux hingegen auf einem Laptop oder Desktop-Computer, werden Sie bereits während der Installation aufgefordert, einen Benutzer mit zugehörigem Passwort anzulegen.

In Linux lassen sich mehrere Benutzer auch zu *Gruppen* zusammenfassen. Dadurch wird die gleichzeitige Verwaltung mehrerer Benutzer erleichtert. Rechte, die man einer Gruppe zuteilt, gelten nämlich automatisch für alle Mitglieder dieser Gruppe. Als Maker wird man kaum in die Verlegenheit kommen, Gruppen in Linux anlegen zu müssen, außer wenn es nötig wird, als Teil einer Gruppe Zugriff auf ein Gerät oder Programm zu erhalten. Auch dazu erfahren Sie mehr in Kapitel 6.

Neben den Namen teilt Linux den Benutzern und den Gruppen auch eine Zahl zu. Die Verwaltung der Benutzer in Form einer Zahl ist für das System leichter als durch eine alphanumerische Zuordnung. Es handelt sich entsprechend um die Benutzeridentifikationsnummer (*user ID*, UID) und die Gruppenidentifikationsnummer (*group ID*, GID) . Was Sie allerdings im Terminal und anderen Programmen zu Gesicht bekommen ist fast ausnahmslos der alphanumerische Benutzername, damit er für uns Menschen leichter erkennbar ist.

Jede Linux-Version bringt einen speziellen Administrator bzw. Superbenutzer namens »root« mit. Dieser Benutzer hat in Linux die Rechte, fast alles zu verändern. Wie Sie sich vorstellen können, ist es daher wenig ratsam, als solcher ständig eingeloggt zu bleiben. Aus diesem Grund ist dieser Benutzer bei Linux-Versionen, die auf dem Raspberry Pi und anderen SBCs betrieben werden, standardmäßig inaktiviert. Falls man ihn wirklich einmal benötigen sollte, lässt er sich aktivieren, was allerdings nicht zu empfehlen ist. Mir ist es mehrfach passiert, dass ich als »root« eingeloggt

war und einen falschen Befehl eingegeben habe, der mir wichtige Dateien gelöscht oder etwas verändert hat, was ich nicht beabsichtigt hatte.

Als Maker, der SBCs für seine Projekte einsetzt, ist es zwar immer möglich, seine Speicherkarte neu einzurichten, doch die viele Arbeit, die man am SBC bereits investiert hat, will man ja nicht leichtfertig aufs Spiel setzen, bloß weil man als »root« eingeloggt einen schweren Fehler begangen hat. Schließlich kann es Stunden oder gar Tage dauern, bis man seinen bisherigen Projektstand wieder erreicht hat, insbesondere dann, wenn man keine Sicherungskopie besitzt. Gewohnheitsmäßig als »root« eingeloggt zu sein, sorgt schon deshalb für weniger Sicherheit, weil wir es uns gerne leicht machen. Mit der Zeit öffnet man Sicherheitslöcher und arbeitet mit einfach zu erratenden Passwörtern, die es anderen leicht machen, sich Root-Rechte zu verschaffen und unserem System und damit unserem Projekt ernsthaften Schaden zuzufügen und unsere Daten zu stehlen.

Bei den meisten Maker-Projekten ist Sicherheit kein großes Thema, doch sobald Ihr Projekt mit dem Internet verbunden ist oder anderweitig im öffentlichen Raum funktioniert, ist es wichtig, das System zu schützen. 2016 haben mehrere weltweite Störungen des Internets von sich reden gemacht, bei denen Geräte im »Internet der Dinge« (Internet of Things, IoT) von Hackern kompromittiert und für ein Botnet missbraucht wurden. Eine der Möglichkeiten, sein Projekt vor solchen Angriffen zu schützen, ist, unter Linux folgende goldene Regel zu befolgen: *Verwenden Sie niemals den Root-Benutzer, wenn Sie es nicht unbedingt müssen.*

2.8 Rechte und sudo

Vielleicht haben Sie sich schon gefragt, wie Linux darüber entscheidet, was ein Benutzer darf und was nicht bzw. auf welche Dateien er zugreifen darf. Eventuell haben Sie auch einmal versucht, einen Befehl oder ein Programm auszuführen und haben eine Meldung wie diese erhalten: »cannot remove file: permission denied« (Datei kann nicht entfernt werden: Zugriff verweigert). Diese Fehlermeldung bedeutet, dass der Benutzer nicht die erforderlichen Rechte besitzt, diese bestimmte Datei zu löschen. Da Linux die meisten Dinge in Form von Dateien verwaltet und alle diese Dateien jeweils mit Rechten versehen sind, ist es wichtig, zu verstehen, was Rechte sind und wie man sie einsetzt.

Jeder Datei und jedem Ordner sind die benötigten Rechte in Form einer Zahlenfolge zugewiesen. Diese Folge lässt sich in vier Gruppen aufteilen, die sich wiederum in Lese- (`r`), Schreib- (`w`) und Ausführungsrechte (`x`) für den Besitzer, die Gruppe und alle anderen Benutzer entsprechend unterteilen lassen. Wie das im Terminal aussieht, können Sie in Abbildung 2–5 sehen.

```
pi@raspberrypi:~ $ ls -l
insgesamt 40
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Desktop
drwxr-xr-x 6 pi pi 4096 Mai 31 00:14 Documents
drwxr-xr-x 2 pi pi 4096 Mai 31 00:27 Downloads
-rw-r--r-- 1 pi pi   69 Jun  2 10:18 Hallo.sh
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Music
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Pictures
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Public
drwxr-xr-x 2 pi pi 4096 Apr 10 11:52 python_games
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Templates
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Videos
pi@raspberrypi:~ $
```

Abb. 2–5 Beispiel einer Ausgabe nach dem Befehl `ls -l`

In Abbildung 2–6 habe ich beispielhaft aufgeschlüsselt, was diese Nummern und Buchstaben bedeuten, die in zwei Zeilen aus Abbildung 2–5 zu sehen sind.

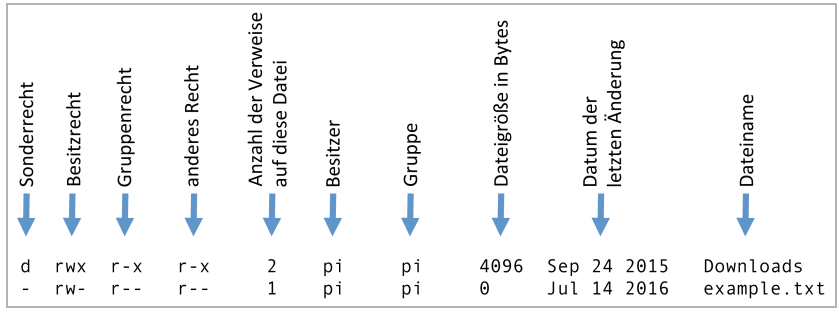


Abb. 2–6 Aufschlüsselung der Auflistung der Rechte in der Kommandozeile

Hier sind ein paar Dinge aufgelistet, die wir diesen Informationen entnehmen können:

1. Die Datei *Downloads* ist aufgrund ihrer Sonderrechte in Wirklichkeit ein Verzeichnis. Andere Möglichkeiten wären hier `l` für Link, `s`, um darauf hinzuweisen, dass die Datei nur mit Besitzrechten genutzt werden darf, oder `t`, um anzuzeigen, dass nur der Besitzer das Recht hat, die Datei zu löschen oder umzubenennen.

2. Der Besitzer dieser Dateien ist der Benutzer »pi«.
3. Die Gruppe, zu der diese Dateien gehören, heißt ebenfalls »pi«.
4. Der Benutzer »pi« hat Lese-, Schreib- und Ausführungsrechte für das Verzeichnis *Downloads*, für die Datei *Beispiel.txt* dagegen nur Lese- und Schreibrechte.
5. Mitglieder der Gruppe »pi« haben für die Datei *Beispiel.txt* lediglich Leserecht.
6. Alle weiteren Benutzer (mit Ausnahme von »root«) haben für die Datei *Beispiel.txt* ebenfalls nur Leserecht.

Rechte können auch als Zahlenfolge ausgedrückt werden. 777 bedeutet beispielsweise das Gleiche wie `rwx rwx rwx`. Programme verwenden zur Kennzeichnung der Rechte manchmal Zahlen, manchmal Buchstaben. Deshalb ist es hilfreich, von der einen auf die andere schließen zu können. Das geht folgendermaßen:

```
r = 4
w = 2
x = 1
```

Um also `rxw` ausdrücken zu wollen, addiert man einfach 4+2+1 und erhält somit 7. Eine Datei wie *Beispiel.txt*, die die Rechte `rw- r-- r --` erhalten soll, kann man also mit 644 kennzeichnen. Wie man genau Rechte und Besitzerschaft ändert, behandeln wir in Kapitel 6.

2.9 Probieren Sie es selbst

In der Konsole bzw. im Terminal geben Sie Folgendes ein:

```
ls -l
```

Daraufhin werden Ihnen alle (nicht versteckten) Dateien und Verzeichnisse Ihres jeweiligen Aufenthaltsorts in Ihrem Dateisystem angezeigt. Versuchen Sie nun herauszufinden, wie die Rechte für die aufgelisteten Dateien für die Benutzer und Gruppen beschaffen sind.

Gelegentlich müssen Sie ein Programm laufen lassen oder auf eine Datei zugreifen, für die Sie keine Rechte besitzen. Doch wie ich Ihnen schon gesagt habe, sollten Sie dazu trotzdem nicht als »root« eingeloggt sein. Jetzt sind Sie in der Zwickmühle! Glücklicherweise hat Linux für solche

Fälle vorgesorgt. Dieses eigens dafür geschaffene System heißt `sudo`, was für *super user do* steht und womit Sie Programme laufen lassen können, die zwar die Rechte von »root« erfordern, für die Sie aber nicht als solcher eingeloggt sein müssen. Dafür müssen Sie Ihrem Befehl nur `sudo` voranstellen. Der Zusatz `sudo` fungiert folgendermaßen als zusätzliche Sicherheitsebene:

- Das Benutzerpasswort wird abgefragt, selbst wenn der Benutzer bereits eingeloggt ist.
- Die Zugriffsmöglichkeit verfällt nach kurzer Zeit, danach muss man sich erneut identifizieren.
- Die Möglichkeiten von `sudo` können je nach Benutzer konfiguriert werden.
- Alle `sudo`-Befehle werden protokolliert.
- Das Passwort von »root« muss nicht mitgeteilt werden.
- Programme zum Knacken von Passwörtern haben keine Chance, das Passwort für »root« zu erraten, wenn der Zugang (root access) ohnehin inaktiviert ist.

2.10 Probieren Sie es selbst

Jetzt probieren wir einen Befehl aus, bei dem wir davon ausgehen, dass er nicht funktioniert. Geben Sie dazu im Terminal Folgendes ein:

```
apt-get update
```

Da Sie für einige Programme, die mit `apt-get` angesteuert werden, keine Rechte besitzen, wird dieser Befehl zu einer Fehlermeldung führen:

```
E: Unable to lock the administration directory (/var/lib/dpkg),
are you root?
```

Versuchen Sie es jetzt noch einmal, diesmal jedoch als »root«:

```
sudo apt-get update
```

Falls Sie mit dem Internet verbunden sind, sollten jetzt aufgrund des Befehls die aktuell verfügbaren Softwarepakete heruntergeladen werden. Mehr über `apt-get` erfahren Sie in Kapitel 4.

2.11 Dienste

In Linux gibt es einige Programme, die mit dem Hochfahren des Systems starten und im Hintergrund weiterlaufen. Solche Programme und die Prozesse, die diese starten, nennt man *Dienste*. Dazu gehören beispielsweise Webserver, die Netzwerkkonfiguration, das Filesharing, Programme zum Fernzugriff sowie wichtige Systemfunktionen. Die Dienste wurden geschaffen, damit die Benutzer nicht jedes einzelne Programm über die Kommandozeile starten müssen, um es benutzen zu können. Wenn Sie Ihren Raspberry Pi hochfahren, sehen Sie jede Menge Text über den Monitor laufen. Ein Großteil dieser Informationen betrifft Dienste, die beim Hochfahren gestartet werden. Sie können fast jedes Programm als Dienst laufen lassen. Wie man das macht, erkläre ich in Kapitel 6.

Im Regelfall müssen Sie sich um die Dienste nicht kümmern und Sie sollten auch nicht mit ihnen experimentieren, solange Sie nicht genau wissen, was Sie tun. Ein Beispiel für ungeschicktes Eingreifen in die Dienste könnte ein Raspberry Pi sein, den Sie, weil er Teil eines Roboters ist, ohne Monitor betreiben wollen. Während Sie von außen auf den Rechner zugreifen, kommt Ihnen plötzlich in den Sinn, den Netzwerkdienst zu beenden. Dadurch beenden Sie nicht nur sofort Ihren Zugriff, sondern Sie müssen anschließend Ihren Roboter auseinanderbauen, um sich direkt mit Ihrem Pi zu verbinden und den Dienst wieder zu starten. Im schlimmsten Fall müssen Sie die Stromzufuhr zu ihm unterbrechen, um ihn neu zu starten und sich erneut mit ihm verbinden zu können. Doch das ist meist keine gute Idee, da dies zu Datenverlusten auf Ihrer SD-Karte führen kann.

In Linux gibt es ein System zur Verwaltung der Dienste, das je nach Version unterschiedlich ist. Auf dem Raspberry Pi mit Raspbian Stretch und nachfolgenden Versionen heißt das System `systemd`, was für *system daemon* steht. Mit `systemd` werden nicht nur die Dienste, sondern auch viele andere Linux-Ressourcen kontrolliert. Deshalb ist es das Erste, was beim Hochfahren gestartet wird. Für jeden Dienst existiert ein Skript, das in einem speziellen Verzeichnis abgelegt wird, wenn das entsprechende Programm installiert wird. Dieses Skript bestimmt, was passiert, wenn der Dienst gestartet oder gestoppt wird.

2.12 Probieren Sie es selbst

Manchmal kann es erforderlich sein, einen Dienst neu zu starten, um zu sehen, ob er eigentlich läuft oder nicht. Wenn Sie sich einen Überblick verschaffen wollen, welche Dienste gerade aktiv sind, können Sie im Terminal `systemctl` (*systemd control*) eingeben:

```
systemctl
```

Drücken Sie die Leertaste, um die Liste Seite für Seite durchzugehen, und die Taste Q, um die Ansicht zu beenden. Wenn Sie wissen wollen, wie lange das Betriebssystem gebraucht hat, um hochzufahren, können Sie Folgendes eingeben:

```
systemd-analyze
```

Für die Information, wie lange jeder einzelne Dienst gebraucht hat, um zu starten, geben Sie dies ein:

```
systemd-analyze blame
```

Darüber hinaus gibt es Funktionen, mit denen man jeden einzelnen Dienst kontrollieren kann. Zum Nachschauen habe ich sie hier aufgelistet:

```
systemctl enable name . service

systemctl disable name . service

systemctl start name . service

systemctl stop name . service

systemctl restart name . service

systemctl status name . service

systemctl reload name . service
```

2.13 Prozesse

In einem Linux-System laufen stets mehrere Programme gleichzeitig. Damit Sie sich nicht gegenseitig stören, muss es eine Möglichkeit geben, den Überblick zu wahren und zu gewährleisten, dass unterstützende Programme laufen, auf die andere Programme angewiesen sind, um zu funktionieren. Dafür sorgen bei Linux die *Prozesse*. Wird also ein Programm

ausgeführt, legt Linux dafür einen Prozess an, der dafür steht, was das jeweilige Programm macht. Kurz gesagt, repräsentiert ein Prozess ein laufendes Programm. Jeder dieser Prozesse erhält Zugriff auf die Systemressourcen wie CPU, Arbeitsspeicher und Bibliotheken, damit das Programm korrekt funktioniert. Auf diese Weise kann das Betriebssystem auch verfolgen, was welches Programm gerade macht, damit das System sauber und ordentlich läuft. Zur einfacheren Zuordnung erhalten die Prozesse sogenannte Prozessidentifikationsnummern (process identification number, PID).

Bei Prozessen entstehen auf diese Weise Verwandtschaftsbeziehungen, da es immer ein Programm gibt, das ein anderes startet. Da `systemd` (auch `init`-Prozess genannt) das erste Programm ist, das gestartet wird, erhält es die PID mit der Ziffer 1. Sobald `systemd` einen Dienst oder ein Programm startet, bekommt dieser Prozess seine eigene PID, wobei die Elternidentifikationsnummer (parent PID, PPID) von 1 vom Betriebssystem vermerkt wird. Diese Eltern-Kind-Beziehung unter den Prozessen kann dabei helfen, Probleme zurückzuverfolgen, um deren Ursache zu finden. Wie man Prozesse verwaltet, zeige ich Ihnen in Kapitel 6.

2.14 Probieren Sie es selbst

Um zu sehen, welche Prozesse Ihr Benutzer für Ihre aktuelle Terminalsitzung gestartet hat, geben Sie den Befehl `ps` (für process, Prozess) ein:

```
ps
```

Da Sie vermutlich gerade nicht so viele Programme gleichzeitig laufen haben, wird diese Liste ziemlich kurz sein. Für die Gesamtheit aller aktuell laufenden Prozesse geben Sie Folgendes ein:

```
ps -ef
```

Nun erhalten Sie nicht nur eine Auflistung sämtlicher Prozesse, sondern auch der Benutzer, die sie starteten, die PPIDs und die Zeitpunkte der Starts (siehe Abb. 2–7).

```
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 Oct17 ?        00:00:15 /sbin/init
root         2     0  0 Oct17 ?        00:00:00 [kthreadd]
root         3     2  0 Oct17 ?        00:00:05 [ksoftirqd/0]
root         5     2  0 Oct17 ?        00:00:00 [kworker/0:0H]
root         7     2  0 Oct17 ?        00:00:00 [khelper]
root         8     2  0 Oct17 ?        00:00:00 [kdevtmpfs]
root         9     2  0 Oct17 ?        00:00:00 [netns]
root        10     2  0 Oct17 ?        00:00:00 [writeback]
root        11     2  0 Oct17 ?        00:00:00 [bioset]
root        12     2  0 Oct17 ?        00:00:00 [kblockd]
root        13     2  0 Oct17 ?        00:00:00 [rpciod]
root        15     2  0 Oct17 ?        00:00:00 [kswapd0]
root        16     2  0 Oct17 ?        00:00:00 [fsnotify_mark]
root        17     2  0 Oct17 ?        00:00:00 [nfsiod]
root        18     2  0 Oct17 ?        00:00:00 [kworker/u2:1]
root        23     2  0 Oct17 ?        00:00:00 [scsi_eh_0]
root        24     2  0 Oct17 ?        00:00:00 [scsi_tmf_0]
root        25     2  0 Oct17 ?        00:00:00 [kworker/0:1H]
root        26     2  0 Oct17 ?        00:00:00 [kworker/u2:2]
root        31     2  0 Oct17 ?        00:00:00 [kpsmoused]
root        33     2  0 Oct17 ?        00:00:00 [deferwq]
root        34     2  0 Oct17 ?        00:00:00 [jbd2/sda2-8]
root        35     2  0 Oct17 ?        00:00:00 [ext4-rsv-conver]
root        67     1  0 Oct17 ?        00:00:00 /lib/systemd/systemd-udevd
root        68     1  0 Oct17 ?        00:00:04 /lib/systemd/systemd-journald
root       395     1  0 Oct17 ?        00:00:00 dhclient -v -pf /run/dhclient.et
h0.pid -lf /var/lib/dhcp/dhclient.eth0.leases eth0
root       449     1  0 Oct17 ?        00:00:01 /usr/sbin/cron -f
```

Abb. 2–7 Beispiel für eine Ausgabe nach dem Befehl `ps -ef`

2.15 Warum dies für Maker wichtig ist

Es ist einfach so: Linux kann manchmal ganz schön verwirrend sein, vor allem, wenn man es noch nie benutzt hat. Um etwas zum Laufen zu bekommen, kann man manchmal eine Anleitung zurate ziehen, hat aber womöglich immer noch nicht verstanden, warum etwas funktioniert oder, falls nicht, wie man es für seine Zwecke so ändert, dass es doch funktioniert. Auch wenn Sie bereits mit Linux gearbeitet haben, hat dieses Kapitel vielleicht doch so manche Frage beantwortet, die Sie noch hatten. Mitten in einem Projekt steckenzubleiben, da einige Dinge nicht wie geplant funktionieren, ist nie lustig und kann zu Verzögerungen von Tagen führen, in denen Sie in Foren Nachforschungen anstellen und nach Antworten auf Ihre Fragen suchen. Haben Sie jedoch die Grundprinzipien von Linux einmal verstanden, haben Sie vielleicht schon eine Ahnung, wie Sie das Problem einkreisen können. Jetzt haben Sie auch eine solide Grundlage für die nächsten Kapitel, in denen wir tiefer in die Materie einsteigen werden.

3 Verwendung des Desktops

Auch wenn Sie den Desktop für Projekte mit Linux auf SBCs nicht benötigen, werde ich ihn hier behandeln, da er vermutlich das Erste sein wird, mit dem man als neuer Benutzer in Berührung kommt, wenn man seinen SBC das erste Mal hochfährt. Manche fühlen sich mit einer grafischen Benutzeroberfläche (GUI) auch einfach sicherer, deshalb gehe ich einige Punkte durch, die Linux betreffen und die jemandem, der Windows oder macOS gewohnt ist, nicht unbedingt klar sind.

3.1 Wann setzt man den Desktop ein?

Bei Projekten mit SBCs wird einem der Linux-Desktop an sich kaum nützen, es sei denn, man möchte die Grafik des Desktops direkt verwenden – etwa für ein Spiel oder ein GUI zur Interaktion der Benutzer. Trotz allem kann es auch für Maker Gründe geben, den Desktop einzusetzen.

Erstens kann es sein, dass Sie gar kein Projekt bauen wollen, sondern sich einfach einen preisgünstigen und stromsparenden Desktop-Computer wünschen. In diesem Buch gehe ich zwar grundsätzlich davon aus, dass Sie Ihren SBC als Teil eines Projektes wie etwa einen Roboter oder zum Auslesen von Sensordaten verwenden wollen, doch vielleicht möchten Sie ihn als eigenständigen Computer zum Surfen im Internet, für kleine Spiele oder für Office-Anwendungen einsetzen.

Zweitens kann es sein, dass Sie lieber über den Desktop als über die Kommandozeile arbeiten und dagegen ist im Prinzip nichts einzuwenden. Immerhin findet man im Desktop leichter wieder heraus, falls etwas komplett danebengegangen ist. Man kann über das Menü immer sein System neu starten oder eine zweite Terminalsitzung in einem zweiten Fenster starten, um etwas anderes zu machen, während ein Programm im ersten noch läuft.

Drittens haben Sie im Desktop Zugriff auf sämtliche Hilfsmittel, die das GUI Ihnen bietet, wie z. B. einen Webbrowser, mit dem Sie, ohne einen weiteren Rechner laufen zu lassen, aus dem Internet Informationen zu Ihrem Projekt oder einem Problem holen können. Dies ginge nicht, wenn Sie Ihren SBC nur über das Terminal betrieben.

3.2 Wann sollte man den Desktop nicht einsetzen?

Zunächst einmal sei angemerkt, dass SBCs für ihre Größe heutzutage recht leistungsfähige Rechner sind. Als ich ein Kind war, war die Vorstellung, dass ein voll funktionsfähiger Desktop-Computer locker auf eine Handfläche passen könnte, reine Science-Fiction. Trotzdem ist es immer noch so, dass jede Form von Grafikleistung einem SBC wie dem Raspberry Pi jede Menge Rechenleistung und Arbeitsspeicher abverlangt. Wenn Sie bereits mit einem Raspberry Pi gearbeitet haben, wissen Sie, dass der Browser nicht so flott läuft, wie Sie es von einem größeren Computer oder selbst Ihrem Smartphone gewohnt sind. Wie schwer Ihr Pi gerade arbeiten muss, können Sie der kleinen Anzeigegrafik oben rechts im Bildschirm entnehmen, in der die CPU-Auslastung angezeigt wird (siehe Abb. 3–1).

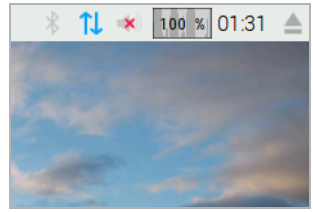


Abb. 3–1 Das Applet in der Taskleiste des Raspberry Pi mit der CPU-Auslastungsanzeige

Schon allein das Starten des Browsers kann die CPU-Auslastung deutlicher hochschnellen lassen, als man das von seinen anderen Computern gewohnt ist. Sollten Sie in diesem Moment noch ein Skript laufen lassen, das zeitkritische Geräte oder Prozesse steuert, kann schon allein dies Ihr Projekt beeinflussen. Der erste Grund also, auf den Desktop verzichten zu wollen, ist, die Ressourcen Ihres SBC zu schonen.

Ein weiterer Grundsatz, den es zu beachten gilt, besteht darin, alles so einfach wie nur irgend möglich zu halten. Jede Form von Komplexität sollte vermieden werden, es sei denn, sie ist absolut für Ihr Programm oder Ihren Prozess erforderlich. Je mehr »bewegliche Teile« ein System hat, desto größer ist die Wahrscheinlichkeit, dass etwas schiefläuft. Wenn ich beispielsweise die Motorhaube meines Pickups aus dem Jahre 1965 öffne, sehe ich nur wenige Teile. Der Großteil des Motorraums ist leer, sodass man einfach an alles herankommt. Falls etwas nicht funktioniert, findet man schnell die Ursache und behebt sie. Bei meinem Transporter aus dem Jahr 2014 ist das ganz anders. Der Motorraum ist voller Systemkomponenten, die voneinander abhängig sind, damit das Fahrzeug fährt. Daran zu arbeiten ist nicht nur wegen der Enge, sondern auch wegen der Komplexität schwierig, sodass man einen Experten benötigt, falls etwas nicht funktioniert. Der Betrieb des Desktops zieht ebenfalls jede Menge Prozesse und damit Komplexität nach sich, die für die meisten Projekte nicht erforderlich ist. Es kann zwar sein, dass bei Ihnen der Betrieb des Desktops keine Probleme bereitet, aber warum sollte man hier ein unnötiges Risiko eingehen?

Aus diesem Grund ist es ratsam, auf den Desktop Ihres SBC zu verzichten, wenn er nicht unbedingt notwendig ist. Im Übrigen läuft Linux in Großunternehmen standardmäßig ohne Desktop, damit eine potenzielle Störquelle wichtiger Geschäftsabläufe ausgeschlossen wird.

3.3 Im Desktop zurechtfinden

In den wesentlichen Aspekten ist der Desktop eines SBC dem von Windows und macOS sehr ähnlich. In der Regel gibt es einen Menü-Button von dem aus man Programme und Einstellungen erreicht. Dann gibt es noch einen Bereich, in dem man sieht, welche Anwendungen gerade laufen, sowie einen Anzeigebereich, wo man Datum, Uhrzeit oder Netzwerkaktivität ablesen kann. Der offensichtlichste Unterschied zu anderen GUIs besteht darin, dass in den meisten Linux-Systemen die Taskleiste oben statt unten platziert ist.

Manchmal ist dieses Panel vollständig unsichtbar und man muss das Menü durch einen Rechtsklick mit der Maus zum Vorschein bringen. In Abbildung 3–2 sieht mein ein Beispiel, wie das auf dem Raspberry Pi aussehen kann.

Abb. 3–2 Aufteilung des Desktops beim Raspberry Pi

1. Menü-Button
2. Anwendungsstartleiste
3. Laufende Anwendungen
4. Panel-Applets
5. Desktop-Bereich

3.4 Verbindung zum Netzwerk

Das Erste, was Sie vermutlich mit dem Desktop erledigen wollen, ist eine Netzwerkverbindung herzustellen, um ins Internet zu gelangen. Beim Anschließen eines Netzwerkkabels sollte dies automatisch geschehen. Für eine drahtlose Verbindung (auf dem Raspberry Pi 3 oder dem Pi Zero W – alle anderen benötigen ein zusätzliches WLAN-Dongle) klicken Sie auf das Applet mit dem WLAN-Symbol, wählen Ihre SSID und geben das Passwort ein.

3.5 Aussehen des Desktops ändern

Es gibt viele Möglichkeiten, das Aussehen des Linux-Desktops seinen Wünschen anzupassen. Im Folgenden schauen wir uns einige der üblichen Dinge an, die man gerne für seine Zwecke ändert. Ich werde hier nur auf den Desktop von Raspbian eingehen, doch bei anderen Desktop-Umgebungen sollten die Methoden genauso funktionieren.

Position des Panels ändern

Um das Panel von oben nach unten oder an eine der Seiten zu verlagern, machen Sie einfach irgendwo auf dem Panel einen Rechtklick und gehen in der sich öffnenden Dialogbox (siehe Abb. 3–3) in die Panel-Einstellungen. Geben Sie dort einfach ein, wo Sie das Panel haben möchten. Sie können darüber hinaus auch noch Größe, Zeichensatz und die Durchsichtigkeit einstellen. Außerdem wählen Sie dort, welche Panel-Applets geladen werden sollen.

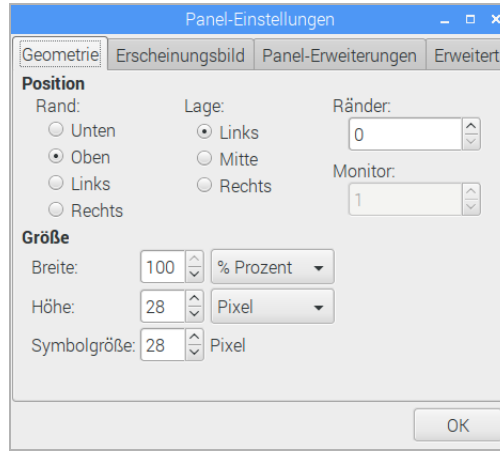


Abb. 3–3 Die Dialogbox für die Panel-Einstellungen

Hintergrundbild ändern

Ihnen gefällt das Aussehen Ihres Desktops nicht? Kein Problem, das geht ganz einfach zu ändern, indem Sie irgendwo auf den Desktop einen Rechtsklick machen und die Desktop-Einstellungen aus dem Kontext-

menü auswählen (siehe Abb. 3–4). Neben den voreingestellten Icons und dem Hintergrundbild können Sie dort auch die Schriften unter den Verknüpfungen auf dem Desktop ändern.

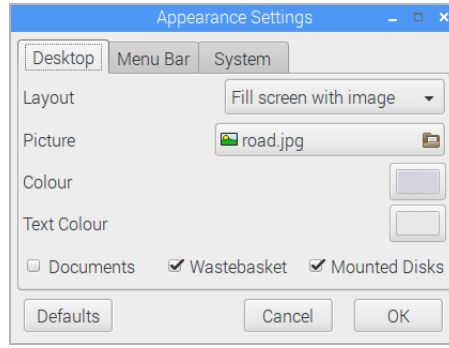


Abb. 3–4 Die Dialogbox mit den Desktop-Einstellungen[1]

Verknüpfungen in der Anwendungsstartleiste ändern

Die im Panel hinterlegten Verknüpfungen haben die Entwickler der jeweiligen Linux-Distribution erstellt. Zusammen mit dem Aufkommen neuer und interessanter Programme ändern sie sich auch gelegentlich. Im Regelfall finden Sie dort Verknüpfungen für den Browser, den Terminalemulator und den Dateimanager. In der Desktop-Umgebung lassen sie sich allerdings ganz einfach ändern. Sie brauchen dazu nur in den Bereich der Icons einen Rechtsklick zu machen und im Kontextmenü die Einstellungen (Settings) für die Anwendungsstartleiste auszuwählen (siehe Abb. 3–5). Links in der Box sind die Verknüpfungen aufgelistet, die sich aktuell in der Startleiste befinden, und rechts finden Sie alle verfügbaren Verknüpfungen. Hier können Sie nun durch Klicken auf eine Verknüpfung mithilfe der Buttons zum Hinzufügen und Entfernen die Anwendungen bestimmen, die Sie in der Startleiste vorfinden wollen. Sie können auch die Reihenfolge, in der sie dort angezeigt werden sollen, mit den Buttons für hoch oder herunter anpassen.

1) Es kann immer wieder vorkommen, dass Ihnen Dialogfenster auf Englisch begegnen, wenn die Übersetzung des Programms unvollständig ist.

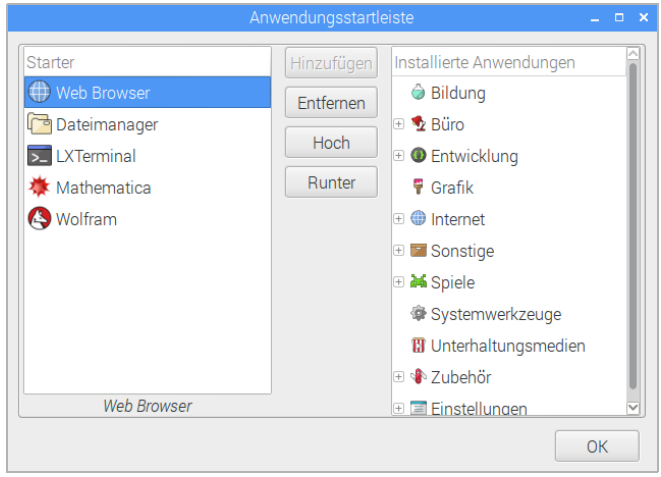


Abb. 3–5 Die Dialogbox für die Anwendungsstartleiste

In anderen Distributionen können Sie durch das Menü navigieren, auf die hinzuzufügende Anwendung einen Rechtsklick machen und »zum Panel hinzufügen« auswählen oder einfach das Icon in das Panel ziehen.

3.6 Verknüpfung auf dem Desktop anlegen

Auf dem Desktop eine Verknüpfung zu einer Anwendung anzulegen, die Sie bereits im Menü vorfinden, ist ebenfalls sehr einfach. Navigieren Sie dazu einfach durch das Menü zu der fraglichen Anwendung, machen einen Rechtsklick darauf und wählen »Dem Desktop hinzufügen«.

Befindet sich die Anwendung nicht im Menü oder handelt es sich um ein Skript, das Sie öfter starten wollen, bereitet das etwas mehr Umstände. Öffnen Sie dazu den mitgelieferten Texteditor, indem Sie ihn unter *Zubehör* auswählen. Tippen Sie dort folgende Informationen über die gewünschte Verknüpfung ein:

```
[Desktop Entry]
Name=irgendein Name
Comment=Klicken Sie hier, um das Programm zu starten
Icon=/usr/share/pixmaps/openbox.xpm
Exec=/Pfad/zu/Ihrem/Programm
Type=Application
Encoding=UTF-8
Terminal=false
Categories=None;
```

Haben Sie alles eingegeben, sollte das in etwa wie in Abbildung 3–6 aussehen. Soll die Verknüpfung auf ein Programm verweisen, das normalerweise im Terminal läuft, müssen Sie sicherstellen, dass der Wert für `Terminal` auf `true` steht.

Speichern Sie das Ganze im Desktop-Ordner unter Ihrem Homeverzeichnis. Geben Sie dem Dateinamen dabei unbedingt die Endung *.desktop*.

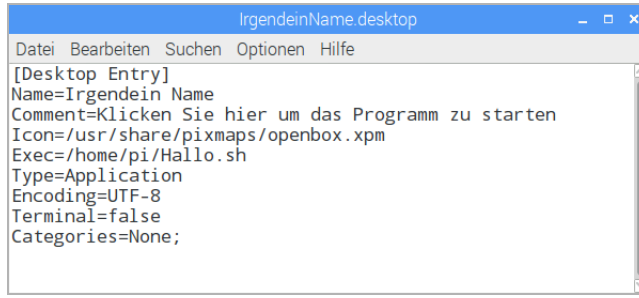


Abb. 3–6 Ein Beispiel für eine Konfigurationsdatei für eine Verknüpfung auf dem Desktop

In anderen Linux-Desktop-Umgebungen reicht es meist aus, einen Rechtsklick auf dem Desktop zu machen und »Verknüpfung anlegen« auszuwählen.

3.7 Probieren Sie es selbst

Versuchen Sie eine Desktop-Verknüpfung anzulegen, die das Shell-Skript *hallo.sh* startet, das wir in Kapitel 2 geschrieben haben. Falls Sie Hinweise benötigen, schauen Sie in Abbildung 3–6 nach, wie diese Datei auszusehen hat.

3.8 Warum dies für Maker wichtig ist

Die meisten Maker, die in Sachen Linux Neulinge sind, werden üblicherweise zunächst mit dem Desktop einsteigen. Das gibt Sicherheit und Vertrauen, um Dinge auszuprobieren und sich umzuschauen. Später wollen Sie den Desktop vielleicht lieber inaktivieren, um auf diese Weise wertvolle Systemressourcen Ihres SBC einzusparen. Wie man das macht, erfahren Sie in Kapitel 5.

4 Grundlagen der Kommandozeile

In diesem Kapitel möchte ich Ihnen einige Grundlagen vermitteln, wie man die Kommandozeile verwendet, ganz gleich, ob Sie vom Terminal(emulator) auf dem Desktop arbeiten oder ob Sie von außen über das Netzwerk Ihren Raspberry Pi ansteuern. Über die Jahre sind im Betriebssystem Linux immer mehr Kniffe und Tastaturkürzel eingebaut worden, um die Arbeit in der Kommandozeile schneller und einfacher zu machen. Schauen wir uns daher einige der wichtigsten Dinge an, die jeder Maker in der Kommandozeile können sollte.

4.1 Der Prompt

Unter dem Prompt versteht man die Stelle in der Kommandozeile, die einem zeigt, wo man sich gerade auf dem Bildschirm befindet. Man wird also aufgefordert (engl. to prompt: auffordern), etwas an einer bestimmten Stelle einzugeben. Solange man nicht im System eingeloggt ist, gibt es keinen Prompt. Geben Sie etwas im Terminal ein, ist es hinter dem Prompt zu sehen. Abbildung 4–1 zeigt, wie er auf dem Raspberry Pi unter den Werkseinstellungen aussieht.

Abb. 4–1 Der Prompt im Terminalemulator Bash auf einem Raspberry Pi

Der voreingestellte Prompt auf dem Raspberry Pi und den meisten anderen Linux-Versionen ist so gestaltet, dass er uns auf einen Blick gleich mehrere nützliche Informationen liefert. Wechseln wir dazu einmal in das Verzeichnis *Downloads* und betrachten uns die Bestandteile des Prompts im Einzelnen (siehe Abb. 4–2):

```
cd Downloads
```

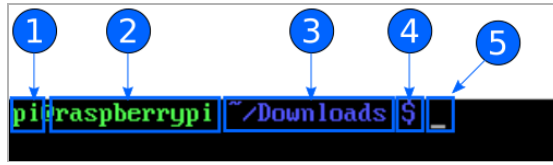


Abb. 4–2 Bestandteile des Prompts

1. Ihr Benutzername
2. Der Hostname vom Raspberry Pi
3. Ihr aktueller Standort im Dateisystem
4. Das Symbol für den Prompt. Sind Sie als »root« eingeloggt, wird es zu einem #.
5. Ein Cursor, der Ihnen anzeigt, wo der Text erscheint, den Sie eingeben. Der Cursor lässt sich mit den Pfeiltasten verschieben.

Sie sehen, dass der Prompt in Grün und Blau gehalten ist. Dies erleichtert Ihnen die Orientierung zwischen den unterschiedlichen Bestandteilen. Sobald Sie die Entertaste gedrückt haben, wird alles, was Sie eingegeben haben, und auch die Textausgabe nach oben gescrollt. Befinden Sie sich im Terminal in einer Desktop-Umgebung können Sie sich sowohl mit der Maus als auch mit dem Scrollbalken im Terminalfenster auf- und abbewegen. Befinden Sie sich nicht im Desktop, erledigen Sie das bei gedrückter Shift-Taste durch Drücken der Tasten für *Bild auf* und *Bild ab*. Das funktioniert in fast jeder Linux-Distribution. Bei der Befehlseingabe können Sie neben den Pfeiltasten auch die Tasten für *Home* und *End* zur Verschiebung des Cursors nutzen.

4.2 Probieren Sie es selbst

Die meisten Linux-Terminals stellen je nach Fenster- und Monitorgröße nur etwa 30 bis 40 Textzeilen im Fenster dar und scrollen dann nach oben weg. Dies führt dazu, dass Sie gelegentlich Teile der Ausgabe sehen möchten, die bereits aus dem Fenster gescrollt sind. Um das zu zeigen, wollen wir einmal einen der Befehle eingeben, die bekanntermaßen viel Ausgabetext zur Folge haben: Display message (`dmesg`) gibt alle Systemmitteilungen vom Linux-Kernel aus, seitdem das System hochgefahren wurde. Geben Sie diesen Befehl also im Terminal ein und drücken Sie die Entertaste:

```
dmesg
```

Die Ausgabe erfolgt so schnell, dass Sie keine Chance haben, alle Informationen mit dem Auge zu erfassen. Nutzen Sie daher die Tasten Shift+*Bild auf* oder Shift+*Bild ab*, um sich die Ausgabe seitenweise anzeigen zu lassen.

4.3 Sich im Dateisystem orientieren

Beim Einsatz von Linux ist es vor dem Ausführen eines Befehls wichtig, zu wissen, in welchem Verzeichnis man sich gerade befindet. Darüber hinaus ist es hilfreich, wenn man weiß, wie man von einem Ort im Dateisystem zu einem anderen springen kann und wie man Informationen über die Dateien bekommt, mit denen man arbeitet. Bevor Sie also irgendetwas anderes machen, sollten Sie sich unbedingt diese grundlegenden Techniken der Kommandozeile aneignen.

4.4 Zur Orientierung: pwd

Meist reicht schon ein Blick auf den Prompt, um zu sehen, in welchem Verzeichnis man sich gerade befindet. Manche Systeme jedoch bieten keinen solch auskunftsfreudigen Prompt und manchmal möchte man einfach sich vergewissern, wo man ist. Mit dem Befehl `pwd` (steht für *print working directory*; zeige aktuelles Verzeichnis an) können Sie herausfinden, wo Sie gerade sind. Sie bekommen dann eine Zeile mit dem Pfad zu Ihrem Verzeichnis angezeigt (siehe Abb. 4–3). Denken Sie daran, dass die höchste Ebene in der Hierarchie der Verzeichnisstruktur durch den Vorwärtsschrägstrich / angezeigt wird.

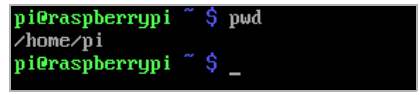


Abb. 4–3 Ein Beispiel für eine Ausgabe nach dem Befehl `pwd`

Der Abbildung 4–3 können wir also entnehmen, dass wir uns im Verzeichnis *pi* befinden, das ein Unterverzeichnis von *home* ist. Wenn man sich in sein System einloggt oder das Terminal öffnet, beginnt man normalerweise im Verzeichnis *home*. Alles, was man in der Kommandozeile tut (eine Datei anlegen, eine Datei löschen, ein Skript oder ein Programm ausführen), bezieht sich auf die Stelle im Dateisystem, an der man sich

gerade befindet. Möglicherweise haben Sie sich schon gefragt, warum der Prompt in Abbildung 4–3 eben nicht */home/pi* als aktuellen Ort anzeigt, sondern nur ~. Dies erklärt sich dadurch, dass ~ ein sogenannter Alias für das Homeverzeichnis des jeweiligen Benutzers ist. Mehr über Aliase erfahren Sie in Kapitel 6.

4.5 Das aktuelle Verzeichnis wechseln: cd

In Linux können Sie mit dem Befehl `cd` (*change directory*; wechsle das Verzeichnis) von einem Verzeichnis in ein anderes wechseln. Dazu geben Sie einfach `cd`, ein Leerzeichen und danach den Pfad zu Ihrem Zielverzeichnis ein. Beachten Sie dabei, dass es in Linux immer auf Groß- und Kleinschreibung ankommt. Mit dem Befehl

```
cd /home/pi/Downloads
```

kommen Sie beispielsweise von */home/pi* zu */home/pi/Downloads*. Sie können sich das so vorstellen, als ob Sie in einem großen Haus von Raum zu Raum springen würden. Selbst wenn das Haus noch so groß wäre, könnten Sie sich mit einem Sprung in jeden Raum katapultieren, solange Sie nur den Weg dorthin wissen. Der Ausgangspunkt, gewissermaßen die Eingangstür, ist dabei stets /. Wie gesagt, müssen Sie nicht immer von Ebene zu Ebene in der Hierarchie der Verzeichnisse wandern, sondern können überall hinspringen, wohin Sie gerade möchten. Geben Sie beispielsweise

```
cd /var/log
```

ein, kommen Sie von Ihrem beliebigen Aufenthaltsort im Dateisystem zu */var/log/*, also dorthin, wo Ihre Logdateien abgelegt werden. Der Befehl `cd` funktioniert also wie ein Teleporter, der Sie an jeden beliebigen Ort im Haus bringt. Wenn Sie diesen Vorgang mit dem mühseligen Durchhangeln durch die Dateistruktur in einem grafischen Dateimanager vergleichen, ist dies doch wesentlich einfacher und schneller – wenn auch nicht so intuitiv.

Dennoch kann die Eingabe des vollständigen Pfades zum Verzeichnis, in das Sie wechseln wollen, umständlich sein. Diesen vollständigen Pfad, der mit / beginnt und mit dem Zielort aufhört (z.B. */home/pi/Downloads*), nennt man auch *absoluter Pfad*. Wo es etwas Absolutes gibt, kann das Relative nicht weit sein: Mit den *relativen Pfaden* können Sie sich etwas

Eingabemühe ersparen. Beim vorherigen Beispiel sind wir von */home/pi* nach */home/pi/Downloads* gesprungen, indem wir Folgendes eingegeben haben:

```
cd /home/pi/Downloads
```

Sie können das Gleiche allerdings auch durch folgende Eingabe bewirken:

```
cd Downloads
```

Der Grund dafür liegt darin, dass *Downloads* ein Unterverzeichnis von */home/pi* ist und wir uns bereits in */home/pi* befinden. Beachten Sie, dass wir vor *Downloads* kein / gesetzt haben, da wir schließlich nicht in das allerhöchste Verzeichnis gelangen wollten. Stattdessen haben wir uns einfach in ein Unterverzeichnis bewegt, das *relativ* zu unserem aktuellen Verzeichnis ein Unterverzeichnis darstellt.

Mit relativen Pfadangaben können wir uns nicht nur in Unterverzeichnisse bewegen, sondern auch in das höhere, sogenannte Elternverzeichnis springen. Das geht mit dem Alias `..` (zwei Punkte). In Linux bezieht sich `.` (ein Punkt) in der Regel auf Ihren aktuellen Standort, `..` hingegen auf das Verzeichnis darüber, eben das Elternverzeichnis. Befinden Sie sich also gerade in */home/pi/Downloads* und möchten eine Ebene weiter nach oben in */home/pi,* geben Sie Folgendes ein:

```
cd ..
```

Möchten Sie gleich zwei Ebenen weiter hoch in das Verzeichnis */home*, geben Sie ein:

```
cd ../..
```

Sie erinnern sich, dass ich Ihnen gesagt habe, dass ~ als Alias bzw. Kürzel für das Homeverzeichnis des jeweiligen Benutzers steht. Der schnelle Weg dorthin zurück geht also auch immer folgendermaßen:

```
cd ~
```

Allerdings kommen Sie auch durch die alleinige Eingabe von `cd` ohne weitere Zusatzinformationen in das Homeverzeichnis:

```
cd
```

Sie möchten, von wo auch immer, in das Unterverzeichnis *Downloads* Ihres Homeverzeichnisses gelangen? Geben Sie dazu einfach Folgendes ein:

```
cd ~/Downloads
```

Dies entspricht der Eingabe:

```
cd /home/pi/Downloads
```

Sie ersetzen also einfach */home/pi* mit ~, wenn Sie dort hinwollen.

4.6 Inhalt eines Verzeichnisses anzeigen: ls

Haben Sie Ihr Zielverzeichnis erreicht, wollen Sie sicher wissen, was sich dort so befindet. Das können Sie mit dem Befehl `ls`, was für *list* steht. Dieser Befehl liefert Ihnen eine alphabetische Auflistung der Dateien und Verzeichnisse Ihres aktuellen Ortes (siehe Abb. 4–4).

Abb. 4–4 Ein Beispiel für die Ausgabe nach dem Befehl `ls`

Da wir einen Prompt mit Farbdarstellung verwenden, können wir den Unterschied zwischen Ordnern und Dateien schon allein farblich erkennen. In diesem Fall steht Blau für Ordner und Grau für die Dateien. Hängen wir dem Befehl `ls` die Zusatzinformation (eine sogenannte Option) `-l` an, erfahren wir weitere Details (siehe Abb. 4–5).

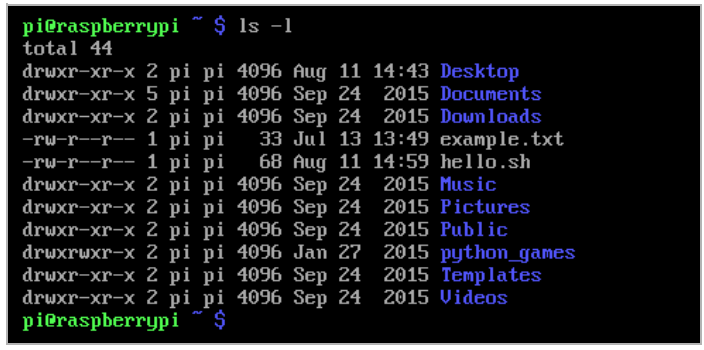


Abb. 4–5 Ein Beispiel für die Ausgabe nach dem Befehl `ls -l`

Was diese Informationen bedeuten, haben wir bereits in Kapitel 2 erfahren. In diesem Verzeichnis befinden sich allerdings mehr Dateien und Ordner, als hier aufgelistet sind. Linux versteckt nämlich bestimmte

Dateien, um einerseits den Zugriff zu erschweren, andererseits um für mehr Übersichtlichkeit zu sorgen, wenn man etwas sucht. In Linux haben versteckte Dateien und Verzeichnisse Namen, die mit einem Punkt beginnen. Möchte man sich diese versteckten Dateien anzeigen lassen, fügt man dem Befehl `ls` die Option `-a` hinzu (siehe Abb. 4–6).

Verketten von Optionen

Oftmals lassen sich mehrere Optionen in der Kommadozeile direkt hintereinanderhängen. Statt also `ls -l -a` können Sie einfach `ls -la` eingeben.

```
pi@raspberrypi ~ $ ls -la
total 116
drwxr-xr-x 20 pi   pi   4096 Sep  2 12:20 .
drwxr-xr-x  3 root root 4096 Sep 24  2015 ..
-rw-------  1 pi   pi   2279 Sep  9 12:51 .bash_history
-rw-r--r--  1 pi   pi   3243 Sep 24  2015 .bashrc
drwxr-xr-x  6 pi   pi   4096 Feb  3  2016 .cache
drwx------ 11 pi   pi   4096 Aug 11 17:08 .config
drwx------  3 pi   pi   4096 Sep 24  2015 .dbus
drwxr-xr-x  2 pi   pi   4096 Aug 11 14:43 Desktop
drwxr-xr-x  5 pi   pi   4096 Sep 24  2015 Documents
drwxr-xr-x  2 pi   pi   4096 Sep 24  2015 Downloads
-rw-r--r--  1 pi   pi     33 Jul 13 13:49 example.txt
drwxr-xr-x  2 pi   pi   4096 Feb  3  2016 .gstreamer-0.10
-rw-r--r--  1 pi   pi     68 Aug 11 14:59 hello.sh
drwx------  3 pi   pi   4096 Aug 11 12:49 .local
drwxr-xr-x  9 pi   pi   4096 Aug 11 12:47 .Mathematica
drwxr-xr-x  2 pi   pi   4096 Sep 24  2015 Music
drwxr-xr-x  2 pi   pi   4096 Sep 24  2015 Pictures
-rw-r--r--  1 pi   pi    675 Sep 24  2015 .profile
drwxr-xr-x  2 pi   pi   4096 Sep 24  2015 Public
drwxrwxr-x  2 pi   pi   4096 Jan 27  2015 python_games
drwxr-xr-x  2 pi   pi   4096 Sep 24  2015 Templates
drwxr-xr-x  3 pi   pi   4096 Sep 24  2015 .themes
drwx------  4 pi   pi   4096 Aug 11 14:09 .thumbnails
drwxr-xr-x  2 pi   pi   4096 Sep 24  2015 Videos
drwxr-xr-x  9 pi   pi   4096 Aug 11 12:47 .WolframEngine
-rw-------  1 pi   pi    105 Aug 31 00:47 .Xauthority
-rw-------  1 pi   pi   5295 Sep  2 12:20 .xsession-errors
pi@raspberrypi ~ $
```

Abb. 4–6 Die Ausgabe nach `ls -la`

Die Ausgabe nach `ls -la` liefert uns also eine detailreichere Aufstellung der Dateien inklusive der ansonsten versteckten Dateien und Verzeichnisse. Es gibt noch weitere nützliche Optionen des Befehls `ls`. Hier sind einige davon:

`-a` Listet alle Dateien inklusive der mit ».« beginnenden versteckten Dateien auf.

`-l` Listet alle Dateien in Langform auf und zeigt die dazugehörigen Rechte an.

`-lh` Listet alle Dateien in Langform mit besser lesbaren Dateigrößen auf.

`-s` Listet alle Dateien mit vorangestellter Größe auf.

`-r` Listet alle Dateien in umgekehrter Sortierreihenfolge auf.

`-R` Listet den Dateibaum auf.

`-S` Sortiert nach Dateigröße.

`-t` Sortiert nach Uhrzeit und Datum.

`-X` Sortiert nach Name der Dateiendungen.

Vielleicht wundern Sie sich, was »total 116« in der ersten Zeile der Ausgabe in Abbildung 4–6 zu bedeuten hat. Damit wird die gesamte Anzahl der Festplattenblöcke bezeichnet, die die durch den Befehl `ls -la` angezeigten Dateien belegen. Vielleicht ist das nicht besonders nützlich, aber interessant zu wissen.

4.7 Neue Dateien und Verzeichnisse anlegen: mkdir und touch

Manchmal möchten Sie vielleicht Ihre Dateien in Ordnern organisieren, die es noch nicht gibt. Um dazu einen neuen Ordner anzulegen, können Sie den Befehl `mkdir` verwenden, was für *make directory*, also »erzeuge Datei«, steht. Geben Sie also so etwas ein wie:

```
mkdir meinneuerOrdner
```

Um dann in diesem Unterverzeichnis eine Reihe neuer Ordner anzulegen, begeben Sie sich mit `cd` dort hinein und geben dann Folgendes ein:

```
mkdir Unterverzeichnis1
cd Unterverzeichnis1
mkdir Unterverzeichnis2
cd Unterverzeichnis2
mkdir Unterverzeichnis3
```

Alternativ können Sie mithilfe der Option `-p` alle drei gleichzeitig erzeugen:

```
mkdir -p Unterverzeichnis1/Unterverzeichnis2/Unterverzeichnis3
```

An Ihrem jeweiligen Standort können Sie auch Dateien erzeugen. Dazu gibt es mehrere Möglichkeiten. Im Regelfall werden die Dateien von den Anwendungen erzeugt, die Sie benutzen. Als Sie beispielsweise in Kapitel 2 Ihr erstes Shell-Skript erzeugt haben, haben Sie mithilfe von `nano` die Datei *hallo.sh* erstellt. Sie wurde in dem Moment angelegt, als Sie sie vor dem Beenden des Programms `nano` gespeichert haben. Mit dem Befehl `touch` können Sie allerdings auch leere Dateien anlegen. Diese weisen außer dem Dateinamen, den Rechten und dem Besitzer keine Daten auf. Um eine solche Leerdatei zu erzeugen, geben Sie einfach `touch` und den Namen der Datei an, die Sie erzeugen wollen:

```
touch Leerdatei
```

Für die Anlage von leeren Dateien kann es mehrere Gründe geben. Erstens kann es sein, dass Sie feststellen möchten, ob Sie an Ihrem aktuellen Ort im Dateisystem Schreibrechte besitzen. Können Sie mit `touch` Dateien anlegen, wissen Sie, dass Sie Schreibrechte haben. Zweitens möchten Sie vielleicht für ein komplexes Projekt wie eine Website oder ein Programm mit mehreren Konfigurations- und Logdateien eine Dateistruktur mit Platzhalterdateien aufbauen.

4.8 Dateien kopieren, verschieben und löschen: cp, mv und rm

Es kommt häufig vor, dass Sie Dateien von einem Ort an einen anderen verschieben oder komplett löschen müssen. Mit dem Befehl `cp` (für *copy*) legen Sie eine Kopie der Datei an. Mit dem Befehl `mv` (für *move*) bewegen, also verschieben Sie eine Datei. Die Befehle `cp` und `mv` funktionieren auf ähnliche Weise. Um eine Datei zu kopieren, geben Sie nach `cp` den Pfad des momentanen Orts der Datei ein, gefolgt vom Pfad des Zielorts. Zum Beispiel:

```
cp /home/pi/hallo.sh /home/pi/Downloads
```

Dadurch wird die Datei *hallo.sh* aus Ihrem Homeverzeichnis in das Unterverzeichnis *Download* kopiert. Da Sie sich bereits in Ihrem Homeverzeichnis befinden, können Sie das auch mit einem relativen Pfad folgendermaßen erledigen:

```
cp hallo.sh Downloads
```

Möchten Sie dagegen die Datei *hallo.sh* in das Verzeichnis *Documents* verschieben, geben Sie Folgendes ein:

```
mv hallo.sh Documents
```

Um eine Datei zu löschen, benutzen Sie den Befehl `rm` (remove; entfernen). *Verwenden Sie diesen Befehl mit äußerster Vorsicht!* In Linux gibt es keinen Papierkorb oder etwas Vergleichbares. Sobald Sie eine Datei über die Kommandozeile gelöscht haben, ist sie für immer verschwunden. Es gibt zwar Programme, mit deren Hilfe man im Einzelfall Dateien wiederherstellen kann, doch sind sie schwer zu bedienen und nicht in jedem Fall erfolgreich. Sind Sie sich dennoch sicher, eine Datei löschen zu wollen, geben Sie `rm`, gefolgt vom Dateinamen, ein:

```
rm hallo.sh
```

Möchten Sie gleich ein ganzes Verzeichnis inklusive aller Unterverzeichnisse löschen, können Sie dies mit der Option `-R` tun:

```
rm -R Ordnername
```

Auch hier ist äußerste Vorsicht angebracht. Es kommt in Foren immer wieder vor, dass Neulinge mit dem Befehl `sudo rm -fR /` ins Verderben gestürzt werden sollen: Dieser Befehl löscht das gesamte Dateisystem.

4.9 Probieren Sie es selbst

Lassen Sie uns jetzt einmal das Navigieren durch das Dateisystem unter Verwendung des Terminals üben. Beginnen Sie mit der Vergewisserung, wo Sie sich gerade befinden:

```
cd
```

Legen Sie ein Unterverzeichnis mit dem Namen *Unterordner* an:

```
mkdir Unterordner
```

Lassen Sie sich die Dateien in umgekehrter Reihenfolge anzeigen, damit Sie einfacher erkennen können, dass das Unterverzeichnis auch angelegt wurde:

```
ls -ltr
```

Wechseln Sie nun in dieses Unterverzeichnis:

```
cd Unterordner
```

Stellen Sie wieder sicher, dass Sie sich im neuen Unterverzeichnis befinden:

```
pwd
```

Erzeugen Sie nun eine Testdatei:

```
touch Testdatei
```

Vergewissern Sie sich, dass die neue Datei auch vorhanden ist:

```
ls -l
```

Jetzt erzeugen Sie eine Kopie dieser Datei:

```
cp Testdatei Testdatei2
```

Schauen Sie nach, ob die Kopie dieser Datei auch angelegt wurde:

```
ls -l
```

Löschen Sie jetzt die Originaldatei:

```
rm Testdatei
```

Wechseln Sie zurück in Ihr Homeverzeichnis:

```
cd ..
```

Jetzt löschen Sie den gesamten Unterordner inklusive der Kopie der Testdatei, die Sie erzeugt haben:

```
rm -R Unterordner
```

Vergewissern Sie sich zum Schluss noch, dass das Unterverzeichnis wirklich verschwunden ist:

```
ls -l
```

In Abbildung 4–7 sieht man ein Beispiel, wie das im Terminal aussehen könnte.

```
pi@raspberrypi:~ $ cd
pi@raspberrypi:~ $ mkdir Unterordner
pi@raspberrypi:~ $ ls -ltr
insgesamt 40
drwxr-xr-x 2 pi pi 4096 Apr 10 11:52 python_games
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Videos
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Templates
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Public
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Pictures
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Music
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Desktop
drwxr-xr-x 6 pi pi 4096 Jun 18 20:42 Documents
drwxr-xr-x 2 pi pi 4096 Jun 19 07:42 Downloads
drwxr-xr-x 2 pi pi 4096 Jun 19 07:50 Unterordner
pi@raspberrypi:~ $ cd Unterordner
pi@raspberrypi:~/Unterordner $ pwd
/home/pi/Unterordner
pi@raspberrypi:~/Unterordner $ touch Testdatei
pi@raspberrypi:~/Unterordner $ ls -l
insgesamt 0
-rw-r--r-- 1 pi pi 0 Jun 19 07:51 Testdatei
pi@raspberrypi:~/Unterordner $ cp Testdatei Testdatei2
pi@raspberrypi:~/Unterordner $ ls -l
insgesamt 0
-rw-r--r-- 1 pi pi 0 Jun 19 07:51 Testdatei
-rw-r--r-- 1 pi pi 0 Jun 19 07:51 Testdatei2
pi@raspberrypi:~/Unterordner $ rm Testdatei
pi@raspberrypi:~/Unterordner $ cd ..
pi@raspberrypi:~ $ rm -R Unterordner
pi@raspberrypi:~ $ ls -l
insgesamt 36
drwxr-xr-x 2 pi pi 4096 Apr 10 12:09 Desktop
drwxr-xr-x 6 pi pi 4096 Jun 18 20:42 Documents
```

Abb. 4–7 Typische Ausgabe der zuvor aufgelisteten Befehle

4.10 Hilfeholen auf Befehl: help, man und info

Es gibt Zehntausende Linux-Befehle und Programme, die sich von der Kommandozeile aus aufrufen lassen und die zudem meist auch noch mehrere Optionen bieten. Niemals könnten wir die alle in diesem Buch durchgehen. Glücklicherweise können wir leicht auf Quellen zugreifen, die uns bei Befehlen weiterhelfen, mit denen wir noch nicht vertraut sind. Diese Quellen sind sogar offline zugänglich, da sie entweder Teil des jeweiligen Programms sind oder automatisch mitinstalliert werden.

Den ersten der in der Überschrift aufgeführten Befehle kann man in der Regel einfach ausführen lassen, indem man dem Kommandonamen eine Hilfe-Option anhängt. Diese Option kann je nach Autor des Programms verschiedene Formen annehmen, doch meist ist es eine dieser vier:

- `-- help`
- `--h`
- `-help`
- `-h`

Die meisten Programmierer, die Hilfsprogramme in der Art von `ls` oder `mkdir` schreiben, legen dort einige hilfreiche Informationen zur Benutzung und die üblichen Optionen ab. Diese werden dann mit der entsprechenden Hilfe-Option angezeigt. Die meisten dieser Hilfsprogramme sagen einem sogar, wie man die Hilfe-Option anwendet, falls man sie nicht gleich korrekt eingibt (siehe Abb. 4–8).

```
pi@raspberrypi:~ $ mkdir -h
mkdir: Ungültige Option -- h
„mkdir --help“ liefert weitere Informationen.
pi@raspberrypi:~ $ mkdir --help
Aufruf: mkdir [OPTION]... VERZEICHNIS...
Erzeugen der/des Verzeichnisse(s), wenn sie noch nicht existieren.

Erforderliche Argumente für lange Optionen sind auch für kurze erforderlich.
  -m, --mode=MODUS  Zugriffsrechte setzen (wie chmod), nicht a=rwx - umask
  -p, --parents     kein Fehler, wenn vorhanden; übergeordnete
                      Verzeichnisse erzeugen, wenn notwendig
  -v, --verbose     für jedes angelegte Verzeichnis eine Meldung ausgeben
  -Z                   SELinux-Sicherheitskontext jedes erzeugten Verzeichnisses
                         auf den Standardtyp setzen
      --context[=KTXT] wie -Z, oder, wenn angegeben den SELinux- oder SMACK-
                         Kontext auf KTXT setzen
      --help     diese Hilfe anzeigen und beenden
      --version  Versionsinformation anzeigen und beenden

GNU coreutils Onlinehilfe: <http://www.gnu.org/software/coreutils/>
Melden Sie Übersetzungsfehler für mkdir an <translation-team-de@lists.sourceforg
e.net>
Full documentation at: <http://www.gnu.org/software/coreutils/mkdir>
or available locally via: info '(coreutils) mkdir invocation'
pi@raspberrypi:~ $
```

Abb. 4–8 Ausgabe des Hilfetextes für mkdir

Die zweite Hilfsquelle für Befehle in Linux ist `man` (für *manual*; Handbuch). Ja, es gibt tatsächlich ein eingebautes Handbuch, das mit jeder Linux-Installation mitgeliefert wird. Mit jedem hinzugefügten und mitinstallierten Programm oder Befehl kommen auch neue Seiten in dieser Anleitung hinzu, auf denen steht, wie man sie anwendet. Sie können `man` als eine Art persönliches Lexikon der Linux-Befehle und -Hilfsprogramme auffassen.

Sobald Sie in diesem Handbuch Informationen über einen bestimmten Befehl oder ein Programm aufrufen, bekommen Sie einen Handbucheintrag, auch *Manpage* genannt, angezeigt. Ist ein Programm Bestandteil von Linux oder kommt es von einer offiziellen Linux-Quelle, gibt es dafür aller Wahrscheinlichkeit nach einen Handbucheintrag. Ein solcher Handbucheintrag ist meist ausführlicher als die Ausgabe als Befehl mit der Hilfe-Option. Geben Sie dazu einfach `man`, gefolgt von dem Befehl, über den Sie etwas erfahren möchten, ein. In Abbildung 4–9 sehen Sie beispielhaft die Ausgabe für `man mkdir`.

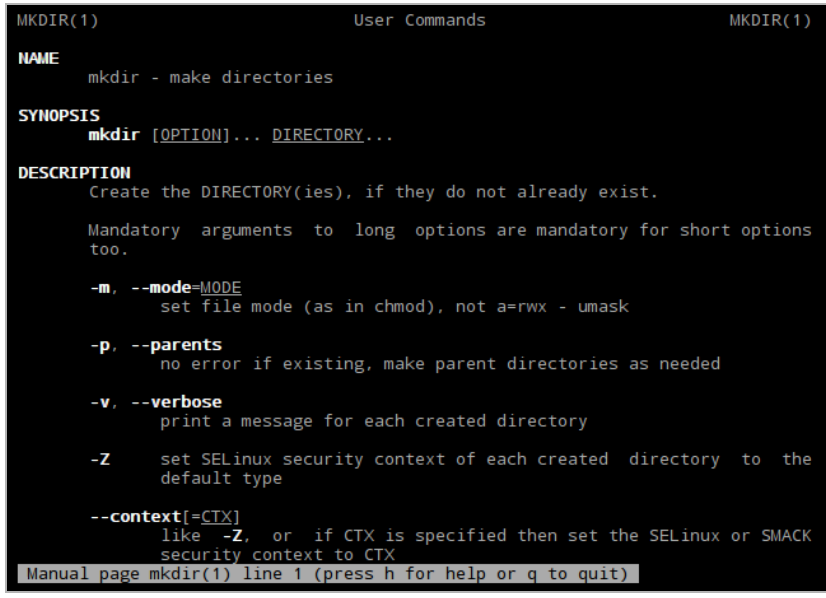


Abb. 4–9 Der Handbucheintrag für den Befehl `mkdir`

Sobald Sie den Handbucheintrag geöffnet haben, können Sie durch das Dokument navigieren, um die nötigen Informationen zu finden. Folgende Befehle werden Ihnen dies etwas erleichtern:

`h` Hilfeseite

Pfeiltaste nach unten oder *Entertaste*
: Eine Zeile herunter gehen

Pfeiltaste nach oben
: Eine Zeile nach oben rücken

`f`, *Leertaste* oder *Taste für Bild ab*
: Seitenweise nach unten bewegen

`b` oder *Taste für Bild auf*
: Seitenweise nach oben bewegen

`G` In die letzte Zeile der Datei springen

`g` In die erste Zeile der Datei springen

`/context`
: Suche nach »context«

`q` Beenden

Im Laufe der Zeit haben sich die Manpages im Aufbau zunehmend vereinheitlicht und enthalten inzwischen immer zumindest folgende Abschnitte:

Name
: Der Name des Befehls oder der Funktion, gefolgt von einer einzeiligen Beschreibung der Aufgabe.

Synopsis
: Eine ausformulierte Beschreibung, wie man den Befehl verwendet und welche Optionen es in der Kommandozeile gibt.

Description
: Eine Beschreibung, wie der Befehl funktioniert.

Examples
: Anwendungsbeispiele

See also
: Verwandte Befehle oder Funktionen

In Abbildung 4–9 sehen Sie in der ersten Zeile den Titel *MKDIR(1)*. Die (1) bezieht sich auf die Sektion innerhalb des Handbuchs, in dem sich diese Seite befindet. In diesem Falle in den *User Commands* (Benutzerkommandos). Bei manchen Programmen verteilen sich die Informationen auf mehrere Sektionen, dann wird dies unter See also angezeigt. Hier eine kurze Auflistung der Handbuchsektionen, damit Sie wissen, wofür die Nummern stehen:

1. User Commands (Benutzerkommandos)
2. System Calls (Systemaufrufe)
3. C Library Functions (Funktionen der Programmiersprache C)
4. Devices and Special Files (Konfigurationsdateien)
5. File Formats and Conventions (Dateiformate und Konventionen)
6. Games et al. (Spiele etc.)
7. Miscellanea (Diverses)
8. System Administration tools and Daemons (Kommandos zur Systemadministration)

Für die jeweiligen Linux-Distributionen werden die Handbuchsektionen nach deren spezifischen Bedürfnissen angepasst und enthalten oftmals zusätzliche Sektionen. Sobald Sie sehen, dass ein Befehl noch in einer anderen Sektion enthalten ist, und Sie sich die Ausführungen dort anschauen wollen, können Sie durch Angabe der Sektionsnummer vor dem Befehl, um den es geht, den dortigen Handbucheintrag aufrufen. Hier also zunächst der erste Handbucheintrag über die Benutzung von `printf`, einem Befehl zur Formatierung von Textausgaben, auf der Kommandozeile:

```
man printf
```

Für Informationen über die Verwendung dieses Befehls in der C-Programmierung geben Sie Folgendes ein:

```
man 3 printf
```

Die dritte Möglichkeit, sich in Linux Informationen zu beschaffen, besteht in dem Hilfsprogramm `info`. Wie bei `man` holt man sich mit `info`, gefolgt vom Befehl, um den es geht, die Informationen in Form einer Datei. In Abbildung 4–10 sehen Sie die Ausgabe nach dem Eintippen von `info mkdir`.

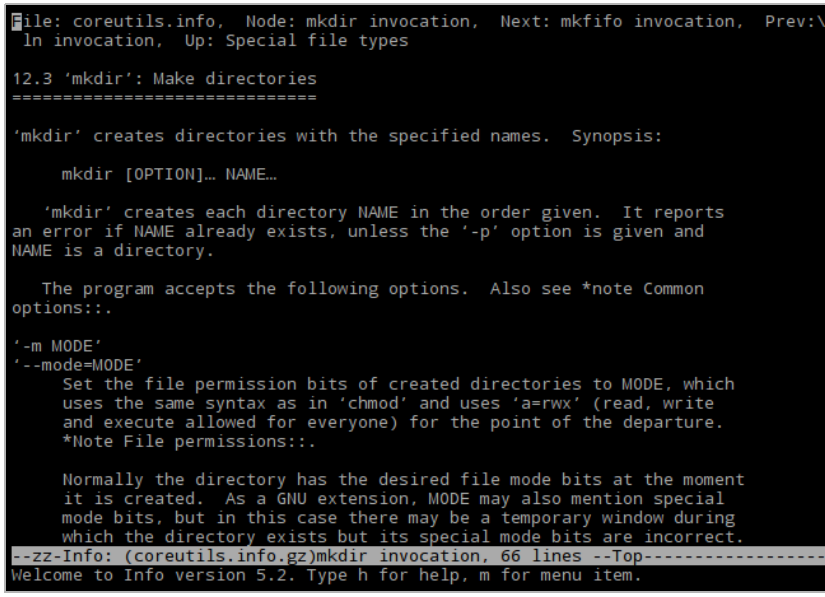


Abb. 4–10 Die Infoseite für den Befehl mkdir

Ganz oben auf der Seite können Sie erkennen, dass Sie sich in der Datei namens *coreutils.info* bewegen und in den Abschnitt (Sektion) über `mkdir` gesprungen sind. Darüber hinaus sehen Sie, dass der nächste Abschnitt von `mkfifo` und der vorhergende von `ln` handelt. Um sich innerhalb der Infodatei zu bewegen, können Sie folgende Tasten nutzen:

`h` Zugang zur Hilfeseite

`x` Schließen der jeweiligen Hilfeseite

`q` Alles beenden

`H` Aufrufen der Anleitung über die Hilfeseite

Pfeiltaste nach oben
: Eine Zeile nach oben rücken

Pfeiltaste nach unten
: Eine Zeile nach unten rücken

Entfernen-Taste
: Eine Seitenlänge zurückscrollen

Home-Taste
: Zum Anfang dieses Abschnitts

Ende-Taste
: Zum Ende dieses Abschnitts

Entertaste
: Dem Link unter dem Cursor folgen

`l` Zurück zum letzten Abschnitt dieser Seite

`[` Zum vorherigen Abschnitt dieses Dokuments

`]` Zum nächsten Abschnitt dieses Dokuments

`p` Zum vorherigen Info-Thema dieser Ebene

`n` Zum nächsten Info-Thema dieser Ebene

`u` Eine Ebene hinauf

`t` Zum obersten Abschnitt dieses Dokuments

`d` Zur Hauptübersicht (Directory node)

`s` Vorwärtssuche nach einer bestimmten Zeichenfolge

`{` Suche nach dem vorherigen Auftreten der gesuchten Zeichenfolge (Rückwärtssuche)

`}` Suche nach dem nächsten Auftreten der gesuchten Zeichenfolge

`i` Suche einer bestimmten Zeichenfolge im Index und anschließende Auswahl des ersten Abschnitts (Node), der gefunden wurde.

Das sind drei Möglichkeiten, wie man innerhalb von Linux Informationen über einen Befehl erhält. Ihre beste Informationsquelle außerhalb des Betriebssystems ist heutzutage die Internetrecherche mithilfe Ihrer bevorzugten Suchmaschine. Sobald Sie tiefer einsteigen wollen, um beispielsweise einen Webserver zu betreiben oder fortgeschrittenere Programme zu schreiben, können Sie sich natürlich auch ein gutes Buch kaufen.

4.11 Probieren Sie es selbst

Wir üben das Nachschlagen über Befehle über die Kommandozeile anhand von `pwd`:

```
man pwd
```

Bewegen Sie sich durch den Handbucheintrag mit den diversen Richtungstasten und, wenn Sie genug haben, beenden Sie den Vorgang mit der Taste Q. Machen Sie das Gleiche jetzt mit den anderen Befehlen, die wir in diesem Kapitel kennengelernt haben:

```
man cd

man ls

man mkdir

man touch

man cp

man mv

man rm
```

4.12 Sparen Sie sich etwas Tipparbeit

Wenn es Ihnen wir mir geht, kann es sein, dass Ihre Rechtschreib- und Tippfähigkeiten zu Wünschen übrig lassen. Nur allzu oft kommt es vor, dass ich 20 oder gar 30 Sekunden lang Befehlsfolgen mit jeder Menge Optionen eingebe und erst nach Drücken der Entertaste feststelle, dass ich mich vertippt habe und von vorne beginnen muss. Hinzu kommt, dass es mir schwerfällt, mich an die exakte Zeichenfolge eines Befehls zu erinnern. Glücklicherweise sind bei Linux einige Hilfsmittel eingebaut, die einem bei derlei Problemen zur Seite stehen.

Einen Befehl automatisch vervollständigen: Tab

Mithilfe der Tab(ulator)-Taste seiner Tastatur kann man sich die automatische Funktion zur Vervollständigung in der Shell zunutze machen. Hat man einen Befehl angefangen zu tippen, kann man ihn durch Druck auf die Tab-Taste ergänzen lassen. Je nach Kontext, was man gerade eingibt, kann man mit dieser Funktion sogar Dateinamen vervollständigen lassen.

Geben Sie beispielsweise *tou* ein und drücken dann die Tab-Taste, ergänzt die Shell die Eingabe, um daraus `touch` zu machen. Ergeben sich aus den ersten Buchstaben mehrere Optionen, passiert nach dem ersten Tab nichts. Beim zweiten Mal wird Ihnen eine Liste aller möglichen Befehle oder Dateinamen angezeigt, die für diese Anfangsbuchstaben infrage kommen. Gibt man also `mkd` ein und drückt zweimal Tab, bekommt man zwei Optionen präsentiert, die mit `mkd` anfangen: `mkdir` und `mkdosfs` (siehe Abb. 4–11).

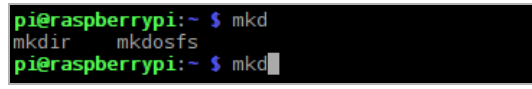
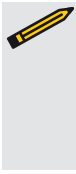


Abb. 4–11 Einsatz der Tab-Taste zur automatischen Vervollständigung von Befehlen

Wenn Sie vor dem Drücken von Tab mehr Zeichen eingeben, schließen Sie nach und nach weitere Optionen aus. Wenn nur noch eine Auswahlmöglichkeit bleibt, wird dieser Befehl oder Dateiname direkt vervollständigt. Diese Funktion bringt bei großen Befehlen und längeren Dateinamen eine echte Zeitersparnis. Auch werden Tippfehler vermieden, vor allem bei Befehlen, die man selten verwendet.



Tab nutzen: Ja oder Nein

Wenn Sie in Linux die normale Bash-Shell benutzen, kann die Tab-Funktion nicht sämtliche verfügbaren Optionen für einen Befehl kennen. Sie kennt lediglich den vollständigen Befehl und alle Dateinamen, die als Teil dieses Kommandos infrage kommen.

Nach einem vorherigen Befehl suchen: Pfeiltaste nach oben, Ctrl-R

Linux protokolliert alles, was man in der Kommandozeile eingibt. Im einfachsten Fall kann man mit der Pfeiltaste nach oben zurückscrollen. Liegt die Eingabe, nach der man sucht, sehr weit zurück, sucht man besser mithilfe des Tastaturkürzels Ctrl-R und gibt einige Zeichen daraus ein. Möchte man zum Beispiel nach der Stelle suchen, an der man zuletzt `nano` verwendet hat, drückt man Ctrl-R und gibt `nano` ein. Ganz gleich, ob bereits etwas vor dem Cursor steht, wenn man Ctrl-R drückt, funktioniert diese Suche. Der Prompt wechselt dann zu der Ausgabe (`reverse-i-`

search), gefolgt von den Buchstaben, die man bei seiner Suche im Protokoll der Kommandozeile eingegeben hat. Drücken Sie jetzt eine der Pfeiltasten oder die Home-, Ende- oder Tab-Taste, so ist die Suche beendet und Sie können das Kommando ändern, das Sie gesucht haben. Wenn Sie Ihre Suche fortsetzen wollen, drücken Sie einfach so oft erneut Ctrl-R, bis Sie Ihre Suche(n) beendet haben (siehe Abb. 4–12).

Abb. 4–12 Mit Ctrl-R im Protokoll der Kommandozeile suchen

4.13 Probieren Sie es selbst

Wechseln Sie in Ihr Homeverzeichnis und erzeugen Sie eine Datei, indem Sie Folgendes eingeben:

```
cd
tou <Tab>  Datei1
```

Wenn Sie jetzt die Tab-Taste drücken, sollte dadurch der Befehl touch vervollständigt werden. Wechseln Sie nun in das Verzeichnis *Downloads* durch folgende Eingabe:

```
cd D <Tab> <Tab>
```

Anschließend sollten Sie etwas Ähnliches wie in Abbildung 4–13 sehen.

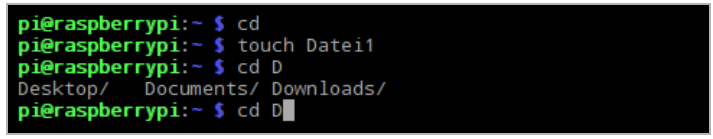


Abb. 4–13 Mit der Tab-Taste den Namen eines Verzeichnisses oder einer Datei automatisch vervollständigen

Wenn Sie jetzt die Buchstaben ow eingeben und anschließend wieder die Tab-Taste drücken, um den Pfad zu vervollständigen, gelangen Sie nach dem Drücken von Enter in den Ordner *Downloads*.

Jetzt erzeugen wir eine zweite Datei auf Grundlage des Kommandozeilenprotokolls. Drücken Sie wieder Ctrl-R und geben Sie wieder `tou` ein (siehe Abb. 4–14).

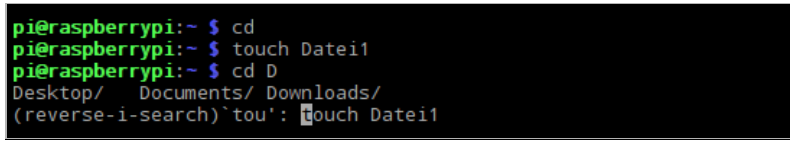


Abb. 4–14 Mithilfe des Kommandozeilenprotokolls einen zuvor verwendeten Befehl nachschauen

Nun drücken Sie die Ende-Taste, ändern *Datei1* in *Datei2* und beenden die Aufgabe mit der Enter-Taste. Jetzt haben Sie zwei Dateien erzeugt: eine in Ihrem Homeverzeichnis, die andere in *Downloads*. Nebenbei haben Sie sich auf diese Weise viel Tipparbeit erspart.

4.14 Über die Kommandozeile eine Netzwerkverbindung herstellen

Über die Kommandozeile eine drahtlose Netzwerkverbindung herzustellen ist zugegebenermaßen ziemlich kompliziert, vor allem, wenn man den Vorgang mit dem Anklicken in der Desktop-Umgebung vergleicht. Trotzdem kann es sehr wichtig werden, die Grundlagen zu kennen, wie man dabei vorgeht, falls einem nämlich nichts anderes übrig bleibt. Mir ist es öfters passiert, dass ich meine Projekte außer Haus, auf einem Maker-Treffen oder für ein Interview, präsentieren wollte und dann feststellen musste, dass die Netzwerkverbindung des SBC noch für den heimischen Router konfiguriert war. Sobald man sich mit einem Projekt, das mit einem Netzwerk verbunden ist, außer Haus begibt, sollte man unbedingt wissen, wie man das dortige Netzwerk konfiguriert, damit das Projekt wie vorgesehen funktioniert.

Leicht mitzunehmen

Für unterwegs sollten Sie sich überlegen, ob Sie Ihren SBC nicht über Ihr Smartphone oder ein anderes Gerät verbinden, sodass Sie in der Lage sind, ein eigenes drahtloses Netzwerk aufzubauen. Auf diese Weise ist Ihr Projekt vernetzt und kann, wo immer Sie einen mobilen Internetzugang haben, auch online gehen.

4.15 Netzwerkschnittstellen

Um das Netzwerk über die Kommandozeile konfigurieren zu können, muss man mehrere Dateien editieren. Zuvor muss man aber die Namen der Netzwerkschnittstellen ermitteln, denn die haben in modernen Linux-Distributionen, wie zum Beispiel Raspbian Stretch, keine festen Namen wie `eth0`, `wlan0` etc. mehr.

Diese Form der Namensgebung ist einigermaßen betagt und führt auf heutiger Hardware zu allerlei Problemen. Sowohl die Netzwerk-Hardware als auch die Software eines Rechners kann sich schnell mal ändern und ein Schnittstellen-Name wie `eth0` kann daher nach einem Neustart eine andere Bedeutung bekommen. Das kann zu Stabilitäts- und Sicherheitsproblemen führen.

Die Namen der Schnittstellen werden seit Debian Stretch aus diesen Gründen jetzt anders festgelegt und enthalten eine eindeutige Kennung. Sie beginnen bei drahtgebundenen Geräten mit »`enx`« und bei drahtlosen Geräten mit »`wlx`«. Dann folgt jeweils die MAC-Adresse des Geräts. Weil Raspbian auf Debian basiert, ist das auch auf dem Raspberry Pi der Fall. Einzig der Name `wlan0` blieb erhalten.

Die Namen der Schnittstellen zu ermitteln, ist recht einfach. Das Kommando `ifconfig` listet alle Schnittstellen mit ihrem Namen auf und es ist ab sofort eine gute Idee, sich die tatsächlichen Namen der Netzwerkschnittstellen anzusehen.

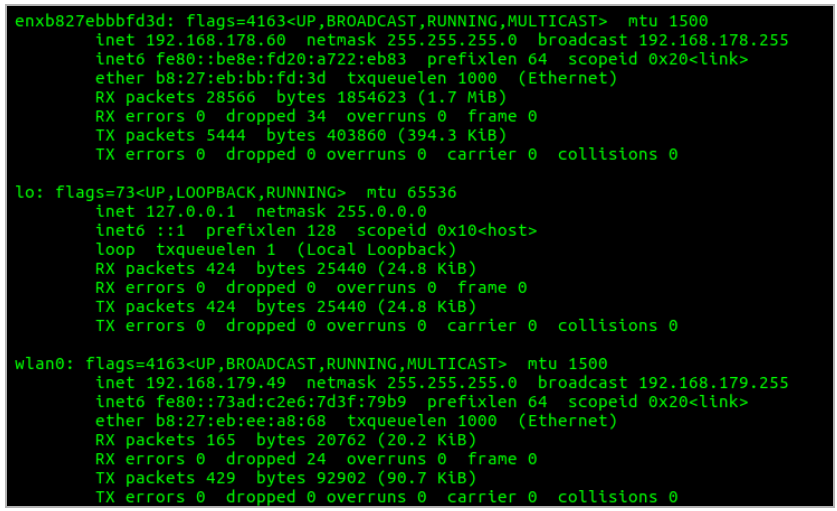


Abb. 4–15 Die Ausgabe von `ifconfig` auf einem Raspberry Pi 3

Bei einer Aktualisierung von Debian »Jessie« auf »Stretch« ändern sich die Namen der Netzwerkschnittstellen übrigens nicht. Das ist nur bei einer Neuinstallation der Fall. Um das alte Verhalten wiederherzustellen, muss man lediglich die Option »`net.ifnames=0`« am Ende der Datei */boot/cmdline.txt* ergänzen und das System neu starten.

4.16 Kabelgebundenes Ethernet

Das lokale Netzwerk ist so vorkonfiguriert, dass ihm automatisch per DHCP eine IP-Adresse zugeteilt wird. Um also eine kabelgebundene Ethernet-Verbindung herzustellen, müssen Sie nichts weiter unternehmen, als das Kabel einzustecken. Ihr Raspberry Pi stellt dann eine Verbindung zum Netzwerk her und ist innerhalb von wenigen Sekunden bereit, im Internet zu surfen.

4.17 Eine feste IP-Adresse zuweisen

Es kann vorkommen, dass Sie neben anderen Konfigurationsinformationen Ihrem SBC eine statische IP-Adresse zuweisen wollen, die sich nicht mehr ändert. Für die Verwaltung von IP-Adressen ist unter Raspbian der DHCP-Konfigurationsdienst verantwortlich und der wird im Wesentlichen über die Datei */etc/dhcpcd.conf* konfiguriert. Die folgende Abbildung zeigt, wie Sie Ihrem SBC ein statische IP-Adresse für eine Ethernet-Schnittstelle zuweisen können.

```
# A sample configuration for dhcpcd.
# See dhcpcd.conf(5) for details.

# Allow users of this group to interact with dhcpcd via the control socket.
#controlgroup wheel

# Inform the DHCP server of our hostname for DDNS.
hostname

# Use the hardware address of the interface for the Client ID.
clientid
# or
# Use the same DUID + IAID as set in DHCPv6 for DHCPv4 ClientID as per RFC4361.
# Some non-RFC compliant DHCP servers do not reply with this set.
# In this case, comment out duid and enable clientid above.
#duid

# Persist interface configuration when dhcpcd exits.
persistent

# Rapid commit support.
# Safe to enable by default because it requires the equivalent option set
# on the server to actually work.
option rapid_commit

# A list of options to request from the DHCP server.
option domain_name_servers, domain_name, domain_search, host_name
option classless_static_routes
# Most distributions have NTP support.
option ntp_servers
# Respect the network MTU. This is applied to DHCP routes.
option interface_mtu

# A ServerID is required by RFC2131.
require dhcp_server_identifier

# Generate Stable Private IPv6 Addresses instead of hardware based ones
slaac private

# Example static IP configuration:
interface enxb827ebbbfd3d
static ip_address=192.168.0.100/24
static routers=192.168.0.1
static domain_name_servers=192.168.0.1 8.8.8.8
```

Abb. 4–16 Die Datei dhcpcd.conf mit den Angaben für eine statische, drahtgebundene IP-Adresse

Beachten Sie die Änderungen, die wir unterhalb des Kommentars »Example static IP configuration« für die Schnittstelle `enxb827ebbbfd3d` vorgenommen haben. Das ist der Name meiner lokalen Ethernet-Schnittstelle und er muss in Ihrem Fall ein anderer sein. Er beginnt aber auch mit »`enx`« und sie können ihn mit einem Aufruf von `ifconfig` ermitteln.

Die Option `ip_address` legt die statische IP-Adresse fest, die dem Rechner ab sofort zugewiesen werden soll. Mit der Option `routers` kann die IP-Adresse des Routers angegeben werden und `domain_name_servers` legt fest, welche DNS-Server zur Adressauflösung dienen.

Es ist übrigens egal, ob man auf diese Weise einer Ethernet-Schnittstelle oder einer drahtlosen Schnittstelle eine statische IP-Adresse zuweist. Gäbe man für die Option `interface` zum Beispiel den Namen `wlan0` an, so würde die statische IP-Adresse für die WLAN-Anbindung gelten.

4.18 Drahtloses Netzwerk

Die Konfiguration einer drahtlosen Netzwerkverbindung erfordert auch Änderungen an einer Datei und in diesem Fall ist es */etc/wpa_supplicant/wpa_supplicant.conf* (siehe Abb. 4–17).

```
country=DE
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
  ssid="YourSSID"
  psk="YourPassword!"
}
```

Abb. 4–17 Die Datei *wpa_supplicant.conf* mit den Angaben für einen einfachen WLAN-Zugang

Im Auslieferungszustand enthält diese Datei nur die ersten drei Zeilen und die Option country hat den Wert GB. Der wurde hier schon auf DE gesetzt, weil der Computer in Deutschland betrieben wird.

Darüber hinaus gibt es einen Eintrag namens `network`, der den Zugang zu meinem WLAN-Netz konfiguriert, also die SSID (Netzwerkname) meines WLAN-Knotenpunkts (Access Point) und das dazugehörige Passwort enthält. Dies ist der bei Weitem einfachste Weg, über die Kommandozeile einen schnellen Zugang zum WLAN-Netz zu bekommen. Möchten Sie

darüber hinaus eine statische IP-Adresse für diese Schnittstelle zuweisen, können Sie die gleichen Änderungen vornehmen, wie sie oben für die kabelgebundene Schnittstelle skizziert sind.

Um mehr als ein drahtloses Netzwerk zu konfigurieren, muss man lediglich weitere network-Einträge zur Datei */etc/wpa_supplicant/wpa_supplicant.conf* hinzufügen. Die Priorität der Netzwerke kann man mit der Option `priority` steuern. Das Netz mit der höchsten Priorität wird dann bevorzugt, wenn es in Reichweite ist.

Es empfiehlt sich, den Rechner nach Änderungen an der Datei *wpa_supplicant.conf* neu zu starten. Zumindest in den ersten Versionen von Raspbian Stretch funktionierte die automatische Neukonfiguration nicht verlässlich.

4.19 Software installieren: apt

Das Installieren von Software in Linux unterscheidet sich aus Benutzersicht sehr von anderen Betriebssystemen. Da der Großteil der Software, die auf Linux betrieben wird, quelloffen und kostenlos ist, werden Softwarepakete (Packages) für die diversen Linux-Distributionen bereitgestellt. Das Herunterladen, Installieren und Entfernen von Paketen, die Verwaltung der eventuellen Abhängigkeiten von weiterer Software sowie die Aktualisierungen übernimmt der Softwarepaketmanager.

Open-Source-Software

Open-Source-Software, also quelloffene Software, unterscheidet sich von geschlossener, proprietärer Software in vielerlei Hinsicht. Zunächst kann jeder, wie das Wort offen schon sagt, den Quellcode einsehen und ändern. Daraus ergibt sich folglich, dass Beitragende aus der Programmierer-Community eingeladen sind, mitzuwirken. Gibt es ein Problem mit der Software, kann man dies entweder selbst beheben oder einen Fehlerbericht einreichen. Des Weiteren kann man völlig legal diese Software weitergeben, ohne sich lizenzrechtliche Probleme einzuhandeln. Dieses Weitergeben ist in der Regel sogar ausdrücklich erwünscht. Dies unterscheidet sich grundlegend von der Weitergabe anderer digitaler Inhalte (siehe Anhang A mit weiteren Informationen über die Geschichte der quelloffenen Software).

Da Raspbian, die verbreitetste auf Raspberry Pi genutzte Linux-Distribution, auf der Distribution Debian basiert, verwendet sie ebenfalls deren Paketmanager `apt`: *Advanced Package Tool*. In `apt` sind eine Reihe von Hilfsprogrammen enthalten, die diverse Aufgaben in Sachen Paketverwaltung erledigen. Das gängigste ist `apt-get`, das sämtliche Funktionen unter sich vereinigt, die nötig sind, um Software zu installieren. Suchen kann es allerdings nicht, dafür ist `apt-cache` zuständig. Da `apt` deutliche Änderungen im System vornehmen kann, muss man einige der Tools unter `apt` mit `sudo` starten.

4.20 Verwendung von apt-get update

Für Linux gibt es Tausende von Softwarepaketen, die auch noch regelmäßig aktualisiert werden. Wenn Sie häufiger die Softwareupdates für Ihren Raspberry Pi abfragen, werden Sie vielleicht schon festgestellt haben, dass täglich mehrfach Updates bereit liegen. Das heißt nun nicht, dass Sie jeden Tag Ihre Software auf den neuesten Stand bringen müssten. Die meisten Updates bringen nur kleine Leistungsverbesserungen und Fehlerbehebungen, die Sie vielleicht meist nicht bemerken. Es kommt jedoch vor, dass Updates Sicherheitslöcher stopfen und diese können tatsächlich sehr wichtig sein. Gut wäre es, das System mindestens einmal im Monat zu aktualisieren. Auch empfiehlt es sich, vor der Installation jeglicher Software das bisherige System zu aktualisieren, damit sichergestellt ist, dass die Softwaredatenbank auf Ihrem Raspberry Pi auf dem aktuellen Stand ist.

Um nach Updates zu schauen, geben Sie im Terminal Folgendes ein:

```
sudo apt-get update
```

Dadurch wird zunächst die Liste von Software aus den Quellen (Repositories) heruntergeladen, die im System vorkonfiguriert sind. Diese Liste wird dann durchgegangen und die Softwaredatenbank aktualisiert, damit sie die Informationen über die neuen und aktualisierten Softwarepakete erhält (siehe Abb. 4–18).

Abb. 4–18 Typische Ausgabe nach dem Befehl apt-get update

4.21 Verwendung von apt-get upgrade

Sobald Sie Ihre Softwaredatenbank aktualisiert haben, können Sie entweder Ihre Software upgraden, die Sie bereits installiert haben, oder neue Software installieren. Mit diesen *Upgrades* sind neue Versionen Ihrer installierten Software gemeint (was man sonst landläufig unter Updates versteht). Durch *Updates* werden lediglich die Aktualisierungen der Softwaredatenbank vorgenommen. Um also jetzt die neuen Softwareversionen zu bekommen, geben Sie Folgendes ein:

```
sudo apt-get upgrade
```

Das Erste, was bei einem Upgrade passiert, ist, dass `apt-get` die Liste der Pakete durchgeht und auf wechselseitige Abhängigkeiten untersucht. Da die Software in Linux quelloffen ist, kann sie modular aufgebaut sein. Hat also jemand bereits ein Programm geschrieben, das eine bestimme Funktion erfüllt, können andere Programme auf dieses zugreifen, sodass diese Funktion nicht von Grund auf neu entwickelt werden muss. Dadurch entstehen logischerweise Abhängigkeiten zwischen den Programmen, die entsprechend verwaltet werden müssen, damit die Programme auch nach Upgrades noch funktionieren.

Nachdem nun diese Abhängigkeiten geklärt sind, berechnet `apt-get`, welche Pakete aktualisiert werden müssen, und führt diese in einer Liste auf. Es zeigt auch das Gesamtvolumen der Downloads und den Festplattenspeicherplatz an, der nach Abschluss der Upgrades durch diese belegt wird (siehe Abb. 4–19).

```
pi@raspberrypi:~ $ sudo apt-get upgrade
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
Paketaktualisierung (Upgrade) wird berechnet... Fertig
Die folgenden Pakete werden aktualisiert (Upgrade):
  bluez dhcpcd5 gir1.2-gdkpixbuf-2.0 libbluetooth3 libgdk-pixbuf2.0-0
  libgdk-pixbuf2.0-common libservlet3.1-java lxpanel lxpanel-data
  lxplug-bluetooth lxplug-ejecter lxplug-network lxplug-volume pigpio
  python-jwt python-pigpio python3-jwt python3-pigpio raspi-copies-and-fills
  rpi-chromium-mods scratch2
21 aktualisiert, 0 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
Es müssen 80,4 MB an Archiven heruntergeladen werden.
Nach dieser Operation werden 1.156 kB Plattenplatz zusätzlich benutzt.
Möchten Sie fortfahren? [J/n] J
Holen:1 http://mirrordirector.raspbian.org/raspbian stretch/main armhf libgdk-pi
xbuf2.0-0 armhf 2.36.5-2+deb9u1 [152 kB]
Holen:2 http://archive.raspberrypi.org/debian stretch/ui armhf rpi-chromium-mods
 armhf 20170913 [9.197 kB]
Holen:3 http://mirrordirector.raspbian.org/raspbian stretch/main armhf libgdk-pi
xbuf2.0-common all 2.36.5-2+deb9u1 [310 kB]
Holen:4 http://mirrordirector.raspbian.org/raspbian stretch/main armhf gir1.2-gd
kpixbuf-2.0 armhf 2.36.5-2+deb9u1 [15,3 kB]
Holen:5 http://mirrordirector.raspbian.org/raspbian stretch/main armhf libservle
t3.1-java all 8.5.14-1+deb9u2 [393 kB]
Holen:6 http://mirrordirector.raspbian.org/raspbian stretch/main armhf python-jw
t all 1.4.2-1+deb9u1 [15,3 kB]
Holen:7 http://mirrordirector.raspbian.org/raspbian stretch/main armhf python3-j
wt all 1.4.2-1+deb9u1 [15,3 kB]
Holen:8 http://archive.raspberrypi.org/debian stretch/main armhf bluez armhf 5.4
3-2+rpt1+deb9u1 [722 kB]
```

Abb. 4–19 Beispiel für einen umfangreichen Upgrade-Vorgang nach `apt-get upgrade`

Ist das letzte Upgrade eine Weile her, kann diese Liste ziemlich lang sein. Wenn Sie irgendwelche Zweifel am Upgrade haben, können Sie mit `N` den Vorgang abbrechen. Ansonsten bestätigen Sie mit `J` und Enter.

Die Vorauswahl akzeptieren

Bei den textbasierten Hilfsprogrammen unter Linux gibt es häufig Mehrfachauswahlmöglichkeiten. Im Regelfall ist die vorgewählte Einstellung der sicherste Weg und in Großbuchstaben gekennzeichnet. Sie können Ihre Wahl dann eingeben oder einfach mit der Entertaste den vorgewählten Weg gehen.

Sie sollten sich an dieser Stelle allerdings auch bewusst sein, dass diese Schritte einige Zeit in Anspruch nehmen können, da die Pakete zunächst heruntergeladen und anschließend installiert werden müssen. Ob der gesamte Vorgang nun Minuten oder Stunden dauert, hängt nicht von der Anzahl der Upgrades (siehe Abb. 4–20), sondern vor allem auch von der Geschwindigkeit der Internetverbindung und der Geschwindigkeit des Raspberry Pi ab (langsamer geht es mit Raspberry Pi 1 oder Zero).

```
Holen:9 http://archive.raspberrypi.org/debian stretch/main armhf dhcpcd5 armhf 1
:6.11.5-1+rpt2 [135 kB]
Holen:10 http://archive.raspberrypi.org/debian stretch/main armhf libbluetooth3
armhf 5.43-2+rpt1+deb9u1 [92,4 kB]
Holen:11 http://archive.raspberrypi.org/debian stretch/ui armhf lxpanel armhf 0.
9.3-1+rpi4 [187 kB]
Holen:12 http://archive.raspberrypi.org/debian stretch/ui armhf lxpanel-data all
 0.9.3-1+rpi4 [1.050 kB]
Holen:13 http://archive.raspberrypi.org/debian stretch/ui armhf lxplug-bluetooth
 armhf 0.4 [15,9 kB]
Holen:14 http://archive.raspberrypi.org/debian stretch/ui armhf lxplug-ejecter a
rmhf 0.3 [9.076 B]
Holen:15 http://archive.raspberrypi.org/debian stretch/ui armhf lxplug-network a
rmhf 0.5 [28,4 kB]
Holen:16 http://archive.raspberrypi.org/debian stretch/ui armhf lxplug-volume ar
mhf 0.7 [15,5 kB]
Holen:17 http://archive.raspberrypi.org/debian stretch/main armhf pigpio armhf 1
.64-1 [193 kB]
Holen:18 http://archive.raspberrypi.org/debian stretch/main armhf python-pigpio
all 1.64-1 [33,5 kB]
Holen:19 http://archive.raspberrypi.org/debian stretch/main armhf python3-pigpio
 all 1.64-1 [33,5 kB]
Holen:20 http://archive.raspberrypi.org/debian stretch/main armhf raspi-copies-a
nd-fills armhf 0.6 [7.226 B]
Holen:21 http://archive.raspberrypi.org/debian stretch/main armhf scratch2 armhf
 0.17 [67,8 MB]
Es wurden 80,4 MB in 19 s geholt (4.219 kB/s).
Lese Changelogs... Fertig
Vorkonfiguration der Pakete ...
(Lese Datenbank ... 122683 Dateien und Verzeichnisse sind derzeit installiert.)
Vorbereitung zum Entpacken von .../00-rpi-chromium-mods_20170913_armhf.deb ...
```

Abb. 4–20 Beispiel für einen umfangreicheren Upgrade-Vorgang nach `apt-get upgrade` (Fortsetzung)

Ist das Terminalfenster nicht breit genug, um die gesamten Zeilen darzustellen, wird der Text beim Download der Pakete einfach umgebrochen. Sobald alle Pakete heruntergeladen sind, beginnt `apt-get` nach und nach mit dem Entpacken, Verarbeiten und Einrichten. Ist der gesamte Vorgang beendet, wird wieder der Prompt angezeigt (siehe Abb. 4–21).

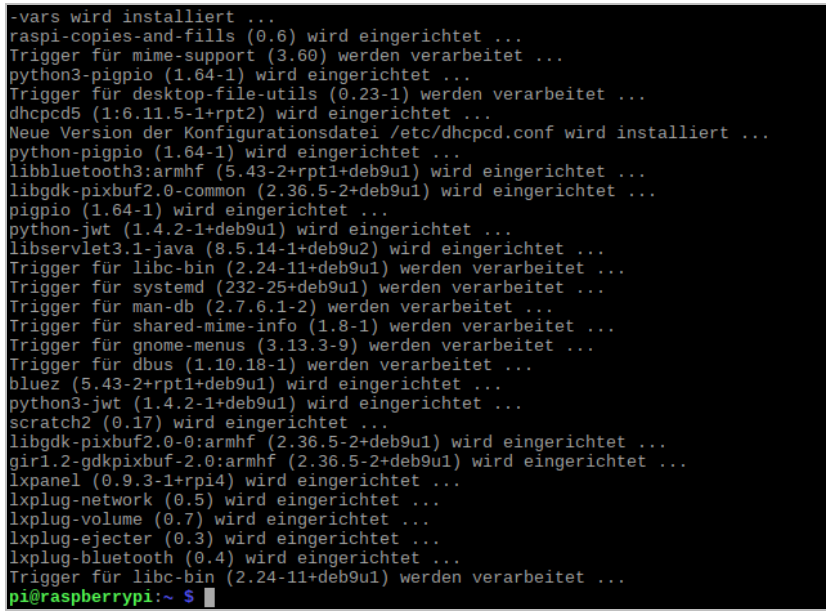


Abb. 4–21 Beispiel für einen umfangreicheren Upgrade-Vorgang nach `apt-get upgrade` (Fortsetzung)

Diese konkreten Upgrades haben mit dem aktuellen Raspberry Pi an einer schnellen Internetverbindung etwa 10 Minuten gedauert. Je häufiger Sie diese Upgrades durchführen, desto weniger Zeit nimmt der Durchlauf in Anspruch. Auch wenn es technisch eigentlich nicht erforderlich ist, so ist es dennoch ratsam, das System anschließend neu zu starten, vor allem, wenn es sich um ein größeres Upgrade handelt.

4.22 Verwendung von apt-cache

Bei der Vielzahl der für Linux erhältlichen Softwarepakete kann es schwierig sein, sich an deren Namen zu erinnern. Deshalb gibt es eine Suchfunktion, mit deren Hilfe man die Datenbank durchsuchen kann. Mit dem Hilfsprogramm `apt-cache` kann man nicht nur suchen, sondern auch nützliche Informationen über die einzelnen Pakete abrufen. Um also nach einem Paket zu suchen, gibt man Folgendes ein:

```
apt-cache search Suchbegriff
```

Das Programm `apt-cache` geht die Datenbank nach dem Suchbegriff durch und gibt alle Paketnamen und beschreibungen aus, in denen dieser Suchbegriff vorkommt (siehe Abb. 4–22).

```
pi@raspberrypi:~ $ apt-cache search gpio
ledmon - Enclosure LED Utilities
stm32flash - STM32 chip flashing utility using a serial bootloader
pigpio - Library for Raspberry Pi GPIO control
python-gpiozero - Simple API for controlling devices attached to the GPIO pins.
python-gpiozero-doc - Documentation for the gpiozero API
python-pigpio - Python module which talks to the pigpio daemon (Python 2)
python-rpi.gpio - Python GPIO module for Raspberry Pi
python-w1thermsensor - Python w1 therm sensor module (Python 2)
python3-gpiozero - Simple API for controlling devices attached to the GPIO pins.
python3-pigpio - Python module which talks to the pigpio daemon (Python 3)
python3-rpi.gpio - Python 3 GPIO module for Raspberry Pi
python3-w1thermsensor - Python w1 therm sensor module (Python 3)
raspi-gpio - Dump the state of the BCM270x GPIOs
wiringpi - The wiringPi libraries, headers and gpio command
pi@raspberrypi:~ $
```

Abb. 4–22 Beispielhafte Ausgabe nach Ausführung des Suchbefehls `apt-cache search` nach dem Begriff »gpio«

Sobald Sie das gesuchte Paket gefunden haben, möchten Sie eventuell mehr darüber erfahren. Dazu verwenden Sie das Kommando `show` und stellen den Paketnamen dahinter:

```
apt-cache show pigpio
```

Daraufhin wird Ihnen angezeigt, wer die Software entwickelt hat, welche Versionsnummer aktuell ist, wie groß der Speicherbedarf auf der Festplatte ist, die offizielle Website für das Paket, eine ausführliche Beschreibung und einiges mehr (siehe Abb. 4–23).

```
Package: pigpio
Version: 1.60-1
Architecture: armhf
Maintainer: Serge Schneider <serge@raspberrypi.org>
Installed-Size: 1267
Depends: libc6 (>= 2.17), init-system-helpers (>= 1.18~)
Homepage: http://abyz.co.uk/rpi/pigpio/
Priority: optional
Section: utils
Filename: pool/main/p/pigpio/pigpio_1.60-1_armhf.deb
Size: 196434
SHA256: bde6f198da866a7a8e70af94e52a22a5c1ddc1cfa690e7d1ad072af31ebc4799
SHA1: 16e18f48d17884f32ce272e39c4fecc1d25cf27c
MD5sum: 0eed990ab1cf1b3f4a161e75442ebf15
Description: Library for Raspberry Pi GPIO control
  Library for the Raspberry which allows control of the General Purpose Input
  Outputs (GPIO).
  .
  pigpio is written in C but may be used by other languages.
  In particular the pigpio daemon offers a socket and pipe interface to the
  underlying library.
Description-md5: f5f722d7007e62bb099c677a006cd65e

pi@raspberrypi:~ $
```

Abb. 4–23 Beispielhafte Ausgabe nach dem Befehl `apt-cache show`

4.23 Verwendung von apt-get install

Die Installation neuer Software geschieht über den Befehl `apt-get install`. Man kann mehrere Pakete gleichzeitig installieren, wobei `apt-get` sich um die Abhängigkeiten kümmert und gegebenenfalls mitinstalliert. Um also neue Software auf Ihrem Raspberry Pi zu installieren, brauchen Sie nur Folgendes einzugeben:

```
sudo apt-get update

sudo apt-get install Paketname Paketname Paketname
```

Anschließend sagt `apt-get` Ihnen, wie viel Daten heruntergeladen werden und wie viel Festplattenplatz durch sie belegt werden, wenn die Installation abgeschlossen ist. Wollen Sie nur ein einziges Paket installieren, wird es ohne weitere Rückfrage installiert (siehe Abb. 4–24).

Überspringen der Bestätigungsmeldung

Wenn Sie die Unterbrechungen durch die Bestätigungsmeldungen vermeiden wollen, können Sie mit der Option `-y` (Langform `-yes`) die interaktiven Fragen automatisch mit Ja beantworten (z.B. `sudo apt-get -y install` *`Paketname Paketname Paketname`*).

```
pi@raspberrypi:~ $ sudo apt-get install pigpio
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
Die folgenden NEUEN Pakete werden installiert:
  pigpio
0 aktualisiert, 1 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
Es müssen 196 kB an Archiven heruntergeladen werden.
Nach dieser Operation werden 1.297 kB Plattenplatz zusätzlich benutzt.
Holen: 1 http://archive.raspberrypi.org/debian/ jessie/main pigpio armhf 1.60-1
[196 kB]
Es wurden 196 kB in 0 s geholt (619 kB/s).
Vormals nicht ausgewähltes Paket pigpio wird gewählt.
(Lese Datenbank ... 112352 Dateien und Verzeichnisse sind derzeit installiert.)
Vorbereitung zum Entpacken von .../pigpio_1.60-1_armhf.deb ...
Entpacken von pigpio (1.60-1) ...
Trigger für man-db (2.7.0.2-5) werden verarbeitet ...
pigpio (1.60-1) wird eingerichtet ...
pi@raspberrypi:~ $
```

Abb. 4–24 Beispielhafte Ausgabe nach dem Befehl apt-get install

Wie Sie in der vorletzten Zeile sehen können, wurde auch `man` mit dem entsprechenden Handbucheintrag aktualisiert.

4.24 Verwendung von apt-get remove

Wie das Installieren von Software, so geht auch das Entfernen von Paketen recht einfach vonstatten. Geben Sie dazu nur Folgendes ein:

```
sudo apt-get remove Paketname Paketname Paketname
```

Durch `apt-get` werden nicht nur die Pakete, sondern auch die mit ihnen verbundenen Handbucheinträge entfernt (siehe Abb. 4–25).

```
pi@raspberrypi:~ $ sudo apt-get remove pigpio
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
Die folgenden Pakete werden ENTFERNT:
  nuscratch pigpio
0 aktualisiert, 0 neu installiert, 2 zu entfernen und 0 nicht aktualisiert.
Nach dieser Operation werden 47,3 MB Plattenplatz freigegeben.
Möchten Sie fortfahren? [J/n] J
(Lese Datenbank ... 112419 Dateien und Verzeichnisse sind derzeit installiert.)
Entfernen von nuscratch (20170112) ...
»Umleitung von /usr/bin/scratch zu /usr/bin/scratch.old durch nuscratch« wird en
tfernt
»Umleitung von /usr/bin/squeak zu /usr/bin/squeak.old durch nuscratch« wird entf
ernt
»Umleitung von /usr/share/scratch/locale/ja_HIRA.po zu /usr/share/scratch/locale
/ja_HIRA.po.old durch nuscratch« wird entfernt
»Umleitung von /usr/share/scratch/locale/ja.po zu /usr/share/scratch/locale/ja.p
o.old durch nuscratch« wird entfernt
Entfernen von pigpio (1.60-1) ...
Trigger für libc-bin (2.19-18+deb8u9) werden verarbeitet ...
Trigger für man-db (2.7.0.2-5) werden verarbeitet ...
pi@raspberrypi:~ $
```

Abb. 4–25 Beispiel für eine Ausgabe nach dem Befehl `apt-get remove`

Linux entfernt dadurch allerdings nicht sämtliche Abhängigkeiten. Nach Installation oder Upgrade von Software zeigt `apt-get` Ihnen eventuell Pakete an, die nicht mehr benötigt werden. Diese können Sie dann durch folgenden Befehl löschen:

```
sudo apt-get autoremove
```

Wieder zeigt Ihnen `apt-get` dann, wie viel Festplattenplatz dadurch frei wird, und fordert Ihre Bestätigung. Anschließend entfernt es alle Programme, die von anderen Softwarepaketen nicht mehr benötigt werden (siehe Abb. 4–26).

```
pi@raspberrypi ~ $ sudo apt-get autoremove
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  libasn1-8-heimdal libdrm-freedreno1 libdrm-nouveau2 libdrm-radeon1 libelf1
  libgssapi3-heimdal libhcrypto4-heimdal libheimbase1-heimdal
  pigpio
0 upgraded, 0 newly installed, 1 to remove and 11 not upgraded.
After this operation, 857 kB disk space will be freed.
Do you want to continue? [Y/n]
(Reading database ... 110787 files and directories currently installed.)
Removing pigpio (1.30-1) ...
Processing triggers for man-db (2.7.0.2-5) ...
pi@raspberrypi ~ $
```

Abb. 4–26 Beispiel für eine Ausgabe nach dem Befehl `apt-get autoremove`

4.25 Verwendung von apt-get dist-upgrade

Für das Upgrade der gesamten Linux-Distribution gibt es einen speziellen Befehl: `apt-get dist-upgrade`. Wie bei jedem anderen Installations- und Upgrade-Vorgang auch müssen Sie mit `apt-get update` Ihre Softwaredatenbank aktualisieren, bevor Sie ihn durchführen. Der Prozess ähnelt einem regelmäßigen Upgrade, kann aber etwas länger dauern, da mehr Pakete heruntergeladen und installiert werden müssen.

Sie sollten sich allerdings nicht genötigt fühlen, jedes Upgrade der Distribution mitzumachen, bloß weil es eines gibt. Wenn alles gut funktioniert, können Sie zufrieden sein, wie es gerade läuft. Die Distribution zu aktualisieren ist nämlich nicht wie ein Update der Softwareversion Ihres Smartphones. Nur sehr selten dürfte es vorkommen, dass dadurch eine Funktion hinzukommt, die Sie für Ihr Projekt dringend brauchen, damit es funktioniert. Ihre aktuelle Linux-Version sollte ein paar Jahre gut funktionieren und währenddessen regelmäßig mit Aktualisierungen versorgt werden. Um sicherzugehen, dass Sie die wichtigen Sicherheitsupdates nicht verpassen, reicht `apt-get update` statt `apt-get dist-upgrade` aus.

Denken Sie daran, dass das Upgrade Ihrer Distribution Pakete, die nicht unbedingt benötigt werden, nicht installiert. Von daher kann es sein, dass es einfacher ist, mit der aktuellen Version der Image-Datei alles neu aufzusetzen, wie wir es in Kapitel 1 getan haben. Sie sollten nur vorher Ihre persönlichen Daten sichern.

4.26 Konflikte beheben

Gelegentlich kann es vorkommen, dass Sie Fehlermeldungen wie »missing dependencies« oder »broken packages« bekommen, wenn `apt-get` versucht, Software zu installieren. Dies deutet in der Regel darauf hin, dass Sie eine Weile keine Aktualisierungen vorgenommen haben. Deshalb sollten Sie in solchen Fällen als Erstes dies probieren:

```
sudo apt-get update
sudo apt-get upgrade
```

Sowohl Softwaredatenbank als auch die Pakete selbst werden dadurch auf den neuesten Stand gebracht. Anschließend versuchen Sie erneut, Ihre Software zu installieren.

4.27 Probieren Sie es selbst

Installieren Sie die Hauptbibliothek *pigpio* und die dazugehörige Python-Bibliothek. Dazu müssen Sie zunächst die genauen Namen dieser Pakete herausfinden und nach Lektüre der Beschreibungen bestätigten, dass die gefundenen die richtigen sind. Der erste Schritt besteht wieder im Update der Softwaredatenbank. Danach schauen Sie die Paketnamen nach:

```
sudo apt-get update

sudo apt-cache search pigpio
```

Sie sollten daraufhin ein Ergebnis bekommen, das Ihnen alle Pakete zeigt, in denen *pigpio* im Titel oder in der Beschreibung vorkommt (siehe Abb. 4–27).

```
pi@raspberrypi:~ $ apt-cache search pigpio
pigpio - Library for Raspberry Pi GPIO control
python-pigpio - Python module which talks to the pigpio daemon (Python 2)
python3-pigpio - Python module which talks to the pigpio daemon (Python 3)
pi@raspberrypi:~ $
```

Abb. 4–27 Verwenden von apt-cache search für die Suche nach Softwarepaketen

Als Nächstes installieren Sie beide Pakete gleichzeitig, ohne dass Sie zwischendurch bestätigen müssen. Dabei gehen Sie davon aus, dass wir für eigene Programme Python 2 und nicht Python 3 verwenden werden. Geben Sie dazu folgenden Befehl ein, um die Software zu installieren (siehe Abb. 4–28):

```
sudo apt-get install -y pigpio python-pigpio
```

```
pi@raspberrypi:~ $ sudo apt-get install -y pigpio python-pigpio
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
python-pigpio ist schon die neueste Version.
Die folgenden NEUEN Pakete werden installiert:
  pigpio
0 aktualisiert, 1 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
Es müssen noch 0 B von 196 kB an Archiven heruntergeladen werden.
Nach dieser Operation werden 1.297 kB Plattenplatz zusätzlich benutzt.
Vormals nicht ausgewähltes Paket pigpio wird gewählt.
(Lese Datenbank ... 112352 Dateien und Verzeichnisse sind derzeit installiert.)
Vorbereitung zum Entpacken von .../pigpio_1.60-1_armhf.deb ...
Entpacken von pigpio (1.60-1) ...
Trigger für man-db (2.7.0.2-5) werden verarbeitet ...
pigpio (1.60-1) wird eingerichtet ...
pi@raspberrypi:~ $
```

Abb. 4–28 Installation von pigpio mit apt-get install

4.28 Neu starten und herunterfahren

Als ich Linux das erste Mal ohne Desktop-Umgebung betrieb, bekam ich ein Problem: Alles ging gut, bis ich das System herunterfahren musste. Da für mich Linux damals noch sehr neu war, wusste ich nicht, wie ich die Desktop-Umgebung wieder zum Laufen bekomme (mehr dazu in Kap. 5). Schlussendlich sah ich mich gezwungen, einfach die Stromzufuhr zu unterbrechen, was für diesen Zweck bei allen Computern keine gute Idee ist und eine ganz besonders schlechte bei einem Raspberry Pi.

Wie ich schon in Kapitel 2 geschrieben habe, wird in Linux alles innerhalb von Dateien geregelt, so auch der Zustand des Betriebssystems. Jede Zustandsänderung des Systems wird in solchen Dateien festgehalten. Wird im Moment der Aktualisierung einer solchen Datei die Stromzufuhr zum Raspberry Pi unterbrochen, wird diese Datei beschädigt und kann beim erneuten Hochfahren nicht ausgelesen werden. Je nachdem, um welche Datei es sich dabei handelt, können dadurch Dateien Ihres Projekts unbrauchbar werden oder, noch schlimmer, Ihr Raspberry Pi das Hochfahren komplett verweigern. Im Gegensatz zu anderen Computern wird dieses Problem noch dadurch verschärft, dass der Raspberry Pi zum Speichern eine SD-Karte verwendet und die Schreibgeschwindigkeit auf SD-Karten im Vergleich zu Festplatten und SSDs relativ gering ist.

Deshalb muss man wissen, wie man sein System über die Kommandozeile neu startet und herunterfährt. Der Befehl lautet für beide Aktionen `shutdown`. Er funktioniert bei fast allen Linux- und Unix-Systemen. Um ein Linux-System neu zu starten, geben Sie Folgendes ein:

```
sudo shutdown -r now
```

Um das Linux-System herunterzufahren, geben Sie ein:

```
sudo shutdown -h now
```

Beachten Sie, dass der einzige Unterschied zwischen diesen beiden Befehlen in der Angabe von `-r` zum Neustart und `-h` zum kompletten Herunterfahren liegt. (Unter Raspbian ist `sudo poweroff` zum Herunterfahren geläufiger. Zum Neustarten verwendet man `sudo reboot`.) Wenn Sie Ihr System herunterfahren, blinkt die LED auf der Platine zehnmal, sobald der Vorgang abgeschlossen ist. Danach kann man bedenkenlos das Stromkabel ziehen.

4.29 Warum dies für Maker wichtig ist

Grundlagenkenntnisse zur Verwendung der Kommandozeile erleichtern das Navigieren im Betriebssystem und ermöglichen es, sich mit dem Internet zu verbinden und Software zu installieren. Diese Vorgänge stellen das Minimum dar, mit dem man als Maker vertraut sein sollte, bevor man selbst loszieht, um ohne eine detaillierte Anleitung aus dem Internet tolle Projekte umzusetzen. Sie können natürlich noch tiefer in die Materie einsteigen und damit Ihre Freunde und Familie als Kommandozeilenzauberer beeindrucken, vor allem mit den Tipps und Tricks, die ich Ihnen in Kapitel 6 zeige.

5 Headless-Betrieb

In diesem Kapitel werde ich erläutern, wie man sich über das Netzwerk mit einem Raspberry Pi verbindet, ohne dass Tastatur, Maus oder Monitor angeschlossen sind, er also *headless* (kopflos) unter Linux läuft. Für jeden Maker ist es wichtig, sein Projekt von außen erreichen zu können. Dadurch sind Dinge mit einem SBC und Linux möglich, die beim permanenten Anschluss der üblichen Peripherie nicht zu machen wären. Der Headless-Betrieb eignet sich also hervorragend für alle mobilen Projekte oder wenn der SBC einfach still in der Ecke Daten sammelt oder liefert.

5.1 Den Desktop ausschalten

Ein Raspberry Pi, der headless betrieben wird, braucht keine aktive Desktop-Umgebung. Den Desktop auszuschalten geht relativ einfach und wird durch das Programm `raspi-config` erledigt. Geben Sie dazu zunächst Folgendes ein:

```
sudo raspi-config
```

Daraufhin öffnet sich die Anwendung zur Konfiguration des Raspberry Pi, wie in Abbildung 5–1 zu sehen. Bewegen Sie die Auswahl mit den Pfeiltasten, bis Sie bei den *Boot Options* sind, und drücken Sie dann die Entertaste.

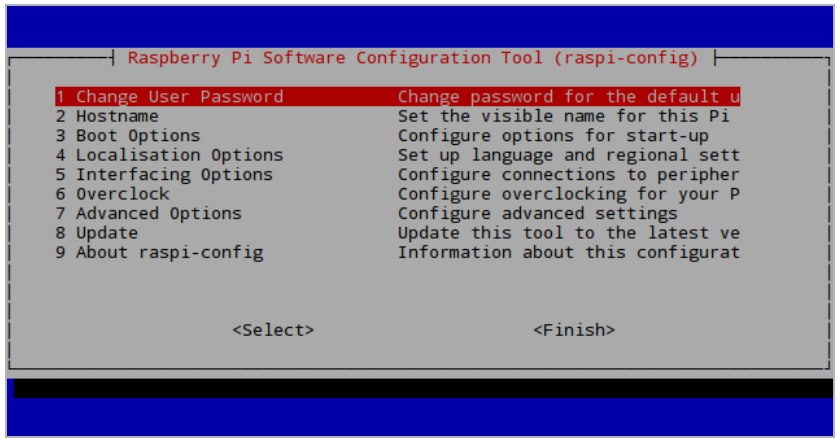


Abb. 5–1 Das Hauptmenü von raspi-config

Wählen Sie anschließend den Punkt *Desktop / CLI* und drücken Sie wieder Enter (siehe Abb. 5–2).

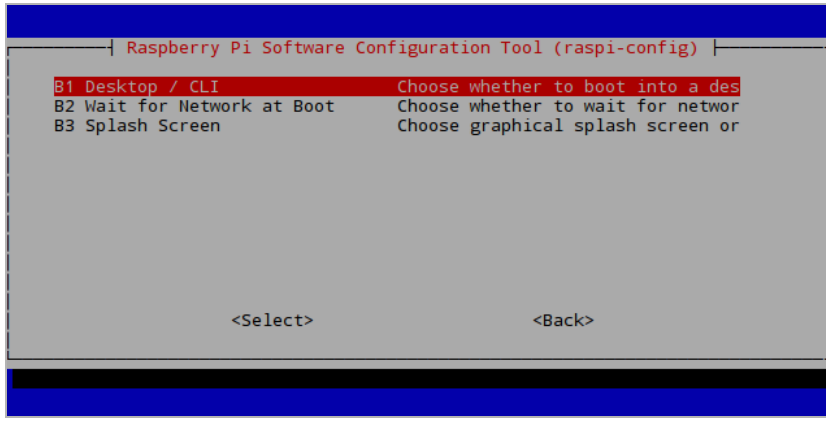


Abb. 5–2 Das Menü mit den Boot Options in raspi-config

Wenn Sie möchten, dass Sie sich einloggen müssen, sobald Sie das System hochfahren, wählen Sie *Console*. Anderenfalls wählen Sie *Console Autologin*, wodurch Sie dann als Benutzer »pi« automatisch eingeloggt werden (siehe Abb. 5–3). Sobald Sie Ihre Auswahl mit Enter getroffen haben, sehen Sie wieder das Hauptmenü von `raspi-config`.

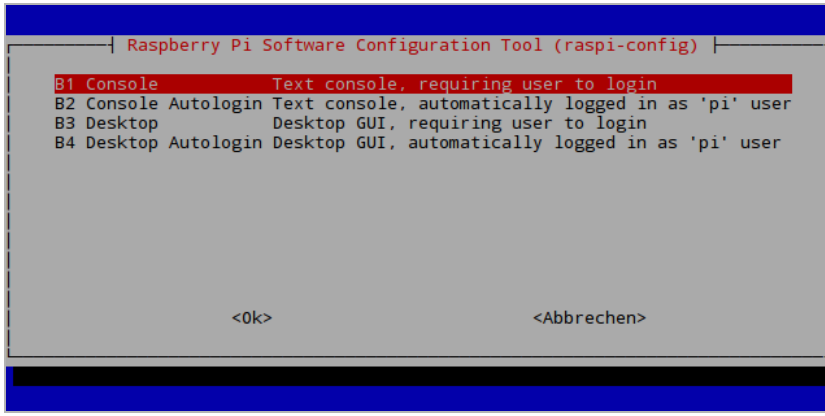


Abb. 5–3 Das Menü »Desktop / CLI« in raspi-config

Drücken Sie die Tab-Taste, um den Cursor auf *Finish* zu bewegen, und dann Enter, um `raspi-config` zu beenden. An dieser Stelle werden Sie gefragt, ob Sie jetzt oder später neu starten wollen. Wie auch immer Sie sich entscheiden, beim nächsten Mal werden Sie direkt mit dem Terminal starten und nicht mehr mit dem Desktop.

Möchten Sie aus irgendeinem Grund wieder in die Desktop-Umgebung wechseln, geben Sie in der Kommandozeile Folgendes ein:

```
startx
```

Wenn Sie wollen, dass Ihr System auch wieder automatisch mit dem Desktop startet, gehen Sie die aufgezeigten Schritte im Menü *Boot Options* in `raspi-config` erneut durch und wählen Sie eine der Desktop-Optionen.

5.2 Ihr System im Netzwerk finden

Um sich mit Ihrem Raspberry Pi verbinden zu können, müssen Sie dessen IP-Adresse herausfinden. Die IP-Adresse ist eine eindeutig identifizierbare Kennzeichnung jedes Computers in Ihrem Netzwerk. Die Adresse Ihres Raspberry Pi erfahren Sie im Gerät selbst, vom Router in Ihrem Heimnetzwerk oder einer App auf Ihrem Smartphone oder Tablet.

IP-Adresse über den Raspberry Pi

Am einfachsten finden Sie die IP-Adresse Ihres Raspberry Pi heraus, indem Sie sie aus der Kommandozeile oder vom Desktop beim Hochfahren ablesen. Sie sollte kurz vor dem Prompt am Schluss zu sehen sein (siehe Abb. 5–4). Natürlich müssen Sie dazu Tastatur, Maus und Monitor noch angeschlossen haben (noch kein Headless-Betrieb, aber evtl. schon ohne Desktop).

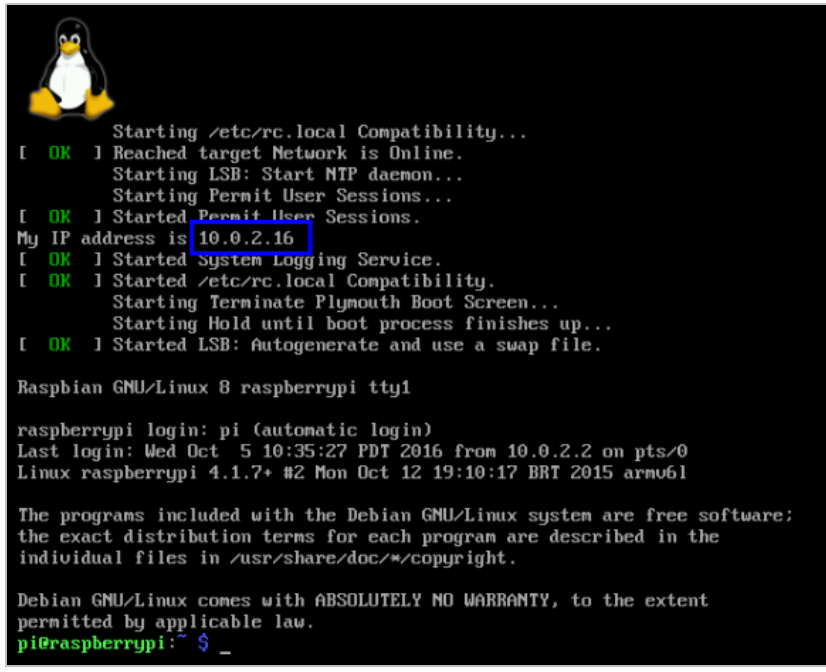


Abb. 5–4 Ablesen der IP-Adresse des Raspberry Pi beim Hochfahren

Ist der Raspberry Pi bereits hochgefahren und die IP-Adresse nicht mehr zu sehen, können Sie sie einfach mit dem Befehl `ip addr show` abfragen (siehe Abb. 5–5).

```
pi@raspberrypi:~ $ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOW
N group default qlen 1000
    link/ether b8:27:eb:f9:d1:85 brd ff:ff:ff:ff:ff:ff
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP g
roup default qlen 1000
    link/ether b8:27:eb:ac:84:d0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.150/24 brd 192.168.0.255 scope global wlan0
       valid_lft forever preferred_lft forever
    inet6 2a02:8071:286:a400:277c:6fa4:84f7:ea56/64 scope global mngtmpaddr nopr
efixroute dynamic
       valid_lft 559049sec preferred_lft 256649sec
    inet6 fe80::c76:9092:d829:7b4a/64 scope link
       valid_lft forever preferred_lft forever
pi@raspberrypi:~ $
```

Abb. 5–5 Abfragen der IP-Adresse mit dem Befehl `ip addr show`

Die für Sie richtige Adresse hängt von der Verbindungsart im Netzwerk ab. Sind Sie mit einem Ethernetkabel verbunden, ist es die im Abschnitt, der mit `enx` beginnt, bei drahtlosem Zugang die unter `wlan0`. Im vorliegenden Beispiel ist es `192.168.0.150` ohne Schrägstrich und allem anderen was noch dahinter steht.

IP-Adresse über den Router

Die meisten modernen Router zeigen Ihnen die verbundenen Geräte entweder als Liste oder wie bei mir als Grafik an. Loggen Sie sich dazu auf die interne Website Ihres Routers ein und begeben sich auf die entsprechende Konfigurationsseite, um diese Information einzusehen. Bei mir sieht das aus wie in Abbildung 5–6.

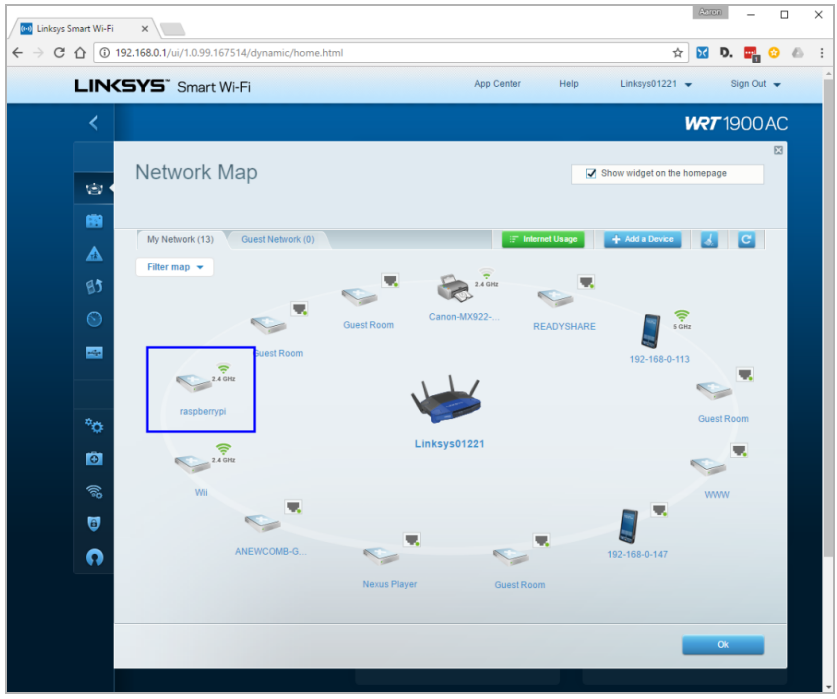


Abb. 5–6 Grafik mit den verbundenen Geräten eines Heimrouters

Durch Klicken auf das Bild, das mit Raspberry Pi beschriftet ist, kann ich dessen IP-Adresse ablesen, in diesem Fall 192.168.0.209 (siehe Abb. 5–7).

Abb. 5–7 Weitere Informationen über ein am Heimrouter angeschlossenes Gerät

IP-Adresse über das Smartphone

Sowohl für Android als auch für iOS gibt es zahlreiche Apps, die Ihr Netzwerk nach Geräten und deren IP-Adressen durchsuchen können. Ich persönlich verwende zurzeit die App »Fing – Network Scanner«, die man im Google Play Store und im App Store von Apple bekommen kann. Sobald die App installiert ist, brauchen Sie sie nur zu öffnen und eventuell den Button zum Aktualisieren der Seite aufzurufen, um dort Ihren Raspberry Pi in Ihrem Netzwerk zu sehen (siehe Abb. 5–8).

Abb. 5–8 Die App Fing zeigt die IP-Adresse des Raspberry Pi an.

5.3 Zugang zur Kommandozeile über SSH

Nur weil Ihr Projekt ohne Tastatur und Maus läuft, heißt das natürlich noch lange nicht, dass Sie keinen Zugriff darauf hätten. Immerhin müssen Sie gelegentlich Dateien hochladen, Einstellungen vornehmen und vor allem in der Lage sein, das System ordnungsgemäß herunterzufahren. Der beste Zugang zur Kommandozeile von außen geschieht über SSH.

SSH steht für *secure shell* (sichere Shell) und liefert, wie der Name schon sagt, einen sicheren Fernzugriff auf eine Shell. Darin unterscheidet sich SSH grundlegend vom Vorgänger Telnet (Teletype Network), der viele Jahre lang für diese Zwecke üblich war. Mit ihm wurden nämlich sämtliche Daten im Klartext übertragen, was eben auch für Benutzernamen und Passwörter galt. Bei SSH hingegen ist die Verbindung zum anderen System verschlüsselt, sodass sie von anderen im Netzwerk nicht mitgelesen werden kann.

Damit SSH funktioniert, sind zwei Komponenten Voraussetzung: ein SSH-Client auf Ihrem lokalen Computer und ein SSH-Server auf dem System, auf das zugegriffen werden soll. Der SSH-Server ist auf dem Raspberry Pi als Bestandteil der Linux-Distribution Raspbian bereits installiert. Aus Sicherheitsgründen ist dieser allerdings standardmäßig nicht in Betrieb. Wir schalten ihn über das Programm `raspi-config` ein (siehe Abb. 5–9):

```
sudo raspi-config
```

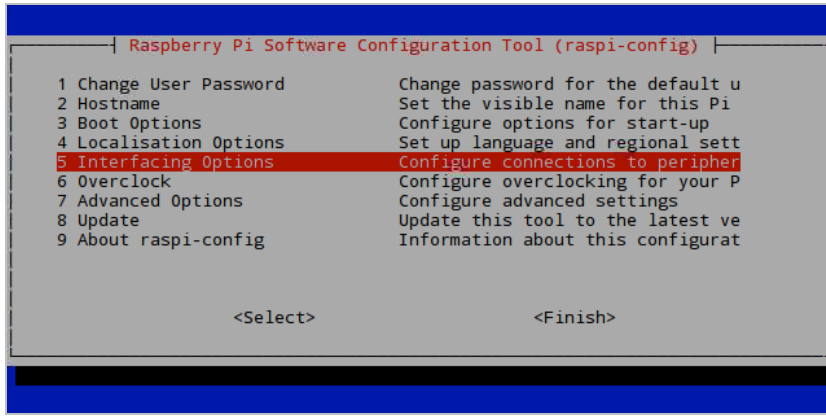


Abb. 5–9 Das Hauptmenü von raspi-config

Mit den Pfeiltasten bewegen Sie den roten Auswahlbalken in diesem Fall über die *Interfacing Options* und drücken dann Enter. Im nächsten Auswahlfenster gehen Sie mit den Pfeiltasten auf *SSH* und drücken dann wieder Enter (siehe Abb. 5–10).

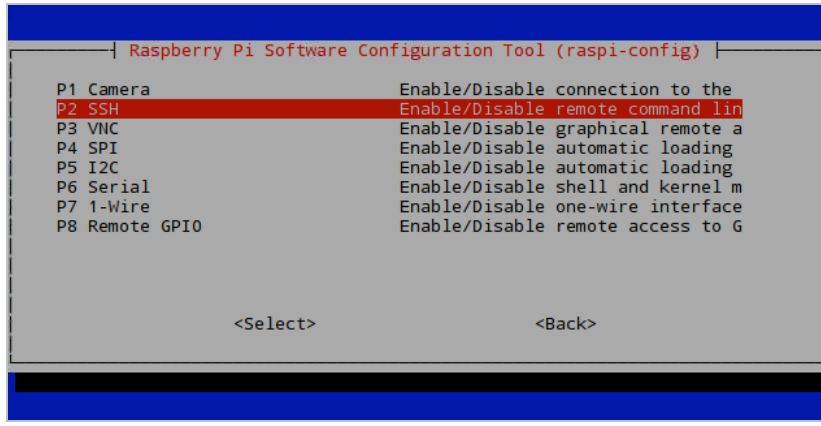


Abb. 5–10 Das Menü der Interfacing Options in raspi-config

Im sich darauf öffnenden Fenster werden Sie gefragt, ob Sie den SSH-Server aktivieren möchten. Sie wählen *Ja* und bestätigen die Aktivierung im nächsten Fenster mit *OK*, um wieder in das Hauptmenü von `raspi-config` zu gelangen.

Jetzt, wo der SSH-Server in Betrieb ist, müssen Sie auf Ihrem Computer noch einen SSH-Client installieren, der sich mit ihm verbindet. In den folgenden Abschnitten gebe ich Ihnen Empfehlungen, wie Sie Ihren Raspberry Pi von Windows, macOS, Linux und Android aus ansteuern können. Ich beschreibe die Installation der einzelnen SSH-Clients nicht, erkläre aber beispielhaft, wie man sich mit dem Raspberry Pi verbindet und was die wesentlichen Merkmale der unterschiedlichen Plattformen sind.

Windows

Unter Windows gibt es keinen mitgelieferten SSH-Client, sodass man selbst einen installieren muss. Der gängigste ist PuTTY, den man von *www.putty.org* herunterladen kann. Nach der Installation findet man ihn im Windows-Startmenü (siehe Abb. 5–11). (Es gibt auch die ausführbare Datei *PuTTY.exe* auf der Downloadseite.)

Welches PuTTY nehme ich?

Das Installationsprogramm installiert mehrere Programme. Das Programm mit dem Namen PuTTY ist der SSH-Client.

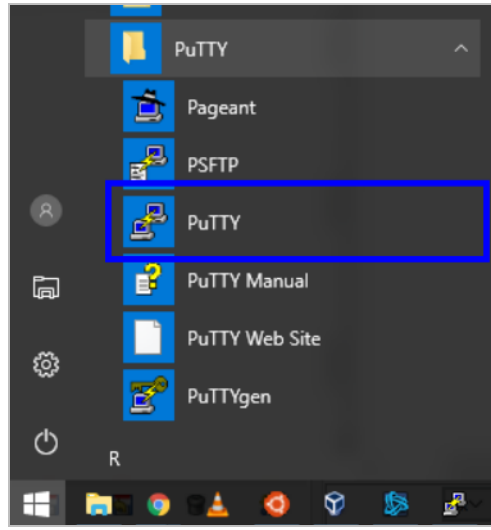


Abb. 5–11 Starten der Anwendung PuTTY unter Windows

Was ist TTY?

Die Buchstabenkombination TTY leitet sich von *TeleTYpe* ab und bezieht sich auf eine bestimmte Art und Weise, mit dem Computer zu kommunizieren. In den Anfängen der Computernutzung brauchte man einen Fernschreiber bzw. Teleprinter, um seine Daten einzugeben. Überbleibsel dieser Technologie finden sich noch in Linux und in Programmen für den Fernzugriff wie PuTTY.

Sobald Sie PuTTY geöffnet haben, geben Sie die IP-Adresse Ihres Raspberry Pi im Feld mit der Bezeichnung »Host Name (or IP address)« ein (siehe Abb. 5–12). Klicken Sie danach auf den Button *Open*, um Ihre SSH-Session zu starten.

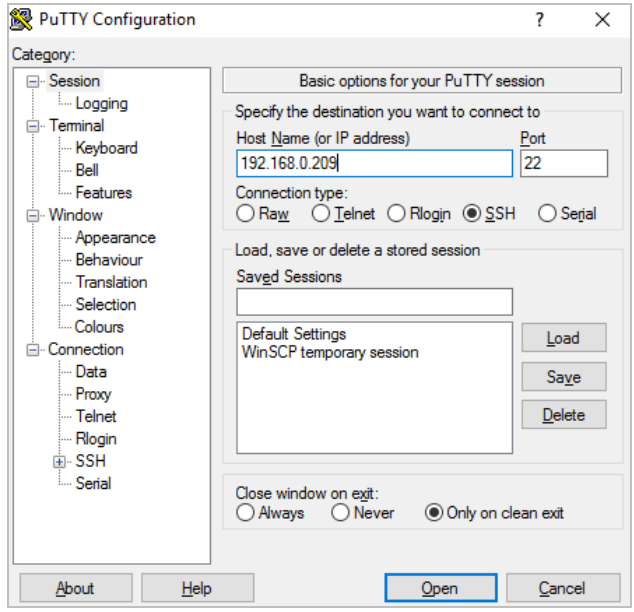


Abb. 5–12 Das Konfigurationsfester in PuTTY

Beim ersten Verbindungsversuch mit Ihrem Raspberry Pi werden Sie gebeten, den Schlüssel Ihres SSH-Servers zu akzeptieren (siehe Abb. 5–13). Klicken Sie einfach auf »Yes«.

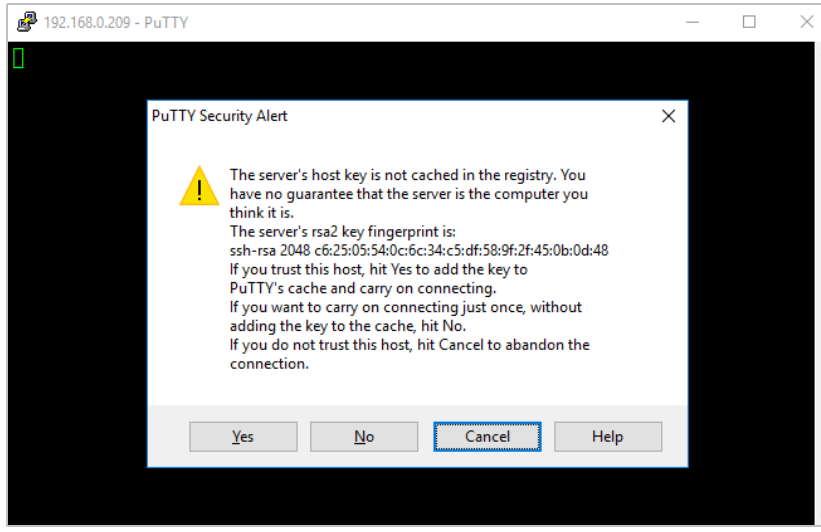


Abb. 5–13 Die Sicherheitswarnung in PuTTY beim ersten Verbindungsversuch

Anschließend öffnet PuTTY ein schwarzes Terminalfenster und verbindet sich mit Ihrem Raspberry Pi. Sobald dies geschehen ist, werden Sie gebeten, Benutzername und Passwort einzugeben (siehe Abb. 5–14). Haben Sie Ihr Passwort nicht schon vorher geändert, ist der voreingestellte Benutzername bei Raspbian »pi« und das Passwort »raspberry«. Schlagen Sie in Kapitel 2 nach, wie man das voreingestellte Passwort ändert.

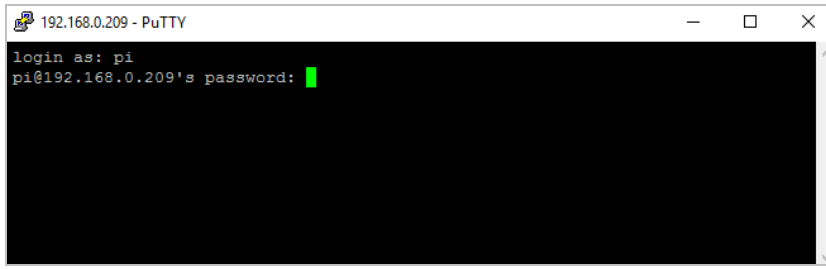


Abb. 5–14 Der Prompt zum Einloggen mit PuTTY

Es kann passieren, dass Sie eine Fehlermeldung bekommen: »Network error: Connection timed out«. Ihr Raspberry Pi ist dann über das Netzwerk nicht erreichbar. In diesem Fall schließen Sie einfach das Terminalfenster, überprüfen noch einmal die eingegebene IP-Adresse sowie die Netzwerkverbindung zum Raspberry Pi und versuchen es erneut.

Sobald Sie die Verbindung erfolgreich hergestellt haben, sehen Sie einen Prompt im Terminalfenster und können damit beginnen, Befehle einzugeben.

macOS

Da macOS auf UNIX basiert, ist dort passenderweise bereits ein SSH-Client installiert, den Sie direkt benutzen können. Um ihn zu finden, öffnen Sie mit dem Finder die Terminal-Anwendung unter Dienstprogramme (siehe Abb. 5–15).

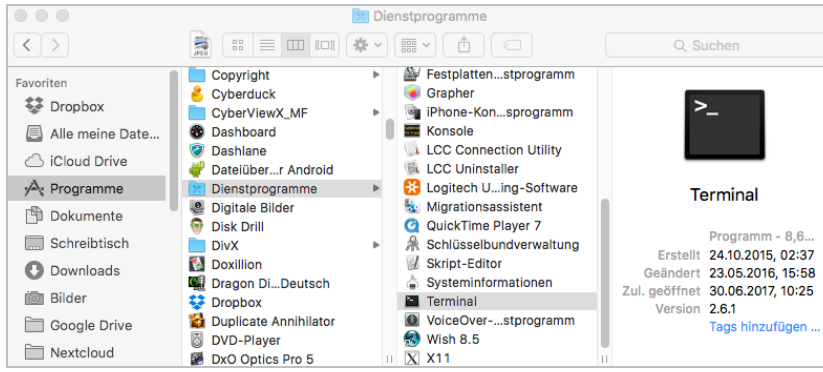


Abb. 5–15 Auffinden der Anwendung Terminal über den Finder von macOS

Nachdem Sie *Terminal* geöffnet haben, geben Sie nach dem Prompt `ssh Benutzername@IP-Adresse` ein, um sich mit Ihrem Raspberry Pi zu verbinden (siehe Abb. 5–16).

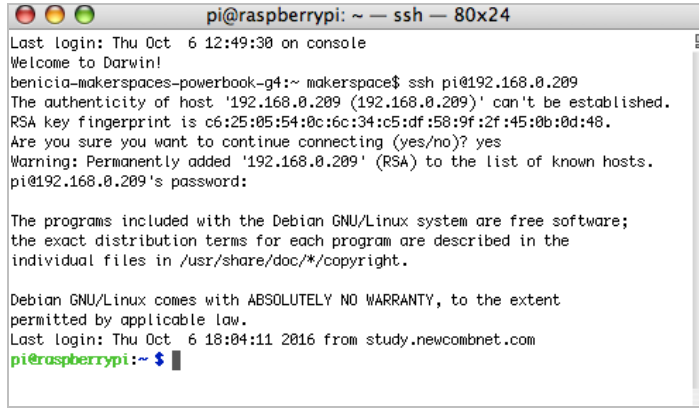


Abb. 5–16 SSH-Verbindung zu einem Raspberry Pi über das Terminal von macOS

Beim ersten Fernzugriff auf Ihr Gerät werden Sie gebeten, den Schlüssel des SSH-Servers zu akzeptieren. Geben Sie »yes« ein und drücken Sie dann die Entertaste, um weiterzukommen. Sobald Sie die Verbindung hergestellt haben, sehen Sie den Prompt zum Eingeben Ihrer Befehle.

Linux

In Linux ist bereits ein SSH-Client installiert. Im Terminalemulator gibt man lediglich `shh Benutzername@IP-Adresse` ein (siehe Abb. 5–17). Da ja auch der Raspberry Pi unter Linux läuft, kann man sogar mit einem Raspberry Pi auf einen anderen via `shh` zugreifen!

```
ssh pi@192.168.0.209
```

Abb. 5–17 Unter Linux mit ssh auf einen Raspberry Pi zugreifen

Wie bei macOS muss man beim ersten Zugriff auf das externe System den Schlüssel des SSH-Servers mit der Eingabe von »yes« und der Entertaste akzeptieren. Ist die Verbindung dann hergestellt, sieht man einen Prompt und kann seine Befehle eingeben.

Android/iOS

Für mobile Geräte gibt es ebenfalls zahlreiche SSH-Clients als Apps. Für Android empfehle ich persönlich den kostenlosen *ConnectBot*, bei dem man mehrere Verbindungen speichern kann (siehe Abb. 5–18). ConnectBot ist im Google Play Store erhältlich.

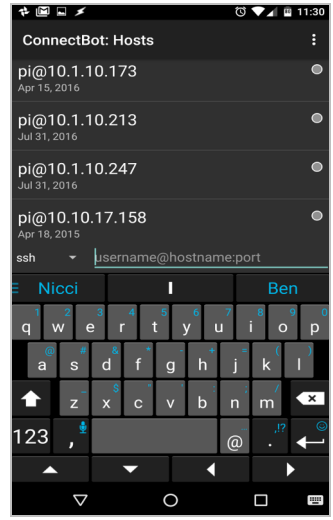


Abb. 5–18 Unter Android mit der App ConnectBot eine Verbindung zu einem Raspberry Pi herstellen

Es gibt zwar auch zahlreiche kostenlose SSH-Clients für iOS, doch auf dem iPhone ist mein persönlicher Favorit *Cathode*, eine App, die im Apple App Store für 5,49 € erhältlich ist.

Abb. 5–19 Mit einem iPhone mit der App Cathode eine Verbindung zu einem Raspberry Pi herstellen

5.4 Remote-Desktop-Verbindung mit VNC

Sie möchten sich mit Ihrem Raspberry verbinden und dabei lieber dessen Desktop bedienen als ihn nur über Kommandozeile erreichen? Kein Problem, dafür gibt es Lösungen in Form sogenannter VNC-Programme (Virtual Network Computing). Wie beim SSH-Zugang benötigt man auch hier zwei Komponenten: einen VNC-Server auf dem System, auf das man von außen zugreifen will, und einen VNC-Viewer auf dem lokalen Computer, von dem aus man sich auf das externe System einloggt. In der aktuellen Version von Raspbian sind sowohl VNC-Server als auch VCN-Viewer von RealVNC installiert. Auf die Installation werde ich nicht eingehen, jedoch auf das Einrichten der Programme, damit Sie den Linux-Desktop des Raspberry Pi fernsteuern können.

Raspberry Pi für VNC einrichten

Um den Desktop von außen sehen zu können, muss man den VNC-Server auf dem Raspberry Pi zunächst einrichten. Dies kann man über die Kommandozeile im lokalen Terminal oder von außen über `ssh` erledigen. Man muss nur folgenden Befehl eingeben, um die VNC-Serversoftware zu aktivieren (alternativ geht dies auch über `raspi-config`):

```
sudo systemctl enable vncserver-x11-serviced.service
```

Damit die VNC-Software tatsächlich läuft, geben Sie noch folgenden Befehl ein (siehe Abb. 5–20):

```
sudo systemctl start vncserver-x11-serviced.service
```

Ab jetzt startet der VNC-Server jedes Mal beim Hochfahren Ihres Raspberry Pi.

```
pi@raspberrypi:~ $ sudo systemctl enable vncserver-x11-serviced.service
Synchronizing state for vncserver-x11-serviced.service with sysvinit using update-rc.d...
Executing /usr/sbin/update-rc.d vncserver-x11-serviced defaults
Executing /usr/sbin/update-rc.d vncserver-x11-serviced enable
pi@raspberrypi:~ $ sudo systemctl start vncserver-x11-serviced.service
pi@raspberrypi:~ $
```

Abb. 5–20 Aktivieren und Starten des VNC-Servers auf dem Raspberry Pi

Der Raspberry Pi erkennt automatisch den angeschlossenen Monitor. Ist aber keiner mehr angeschlossen, wird standardmäßig die geringste Auflösung eingestellt, die ziemlich klein ist. Um also VNC ohne angeschlossenen Monitor sinnvoll nutzen zu können, muss man noch einige Konfigurationsänderungen vornehmen, um eine größere Monitorauflösung zu bekommen. Stellen Sie also eine Verbindung zu Ihrem Raspberry Pi her und ändern Sie die Datei */boot/config.txt* über die Kommandozeile:

```
sudo nano /boot/config.txt
```

Scrollen Sie in der Datei ganz nach unten und fügen Sie folgende Zeilen ein (siehe Abb. 5–21):

```
hdmi_force_hotplug=1
hdmi_ignore_edid=0xa5000080
hdmi_group=2
hdmi_mode=16
```

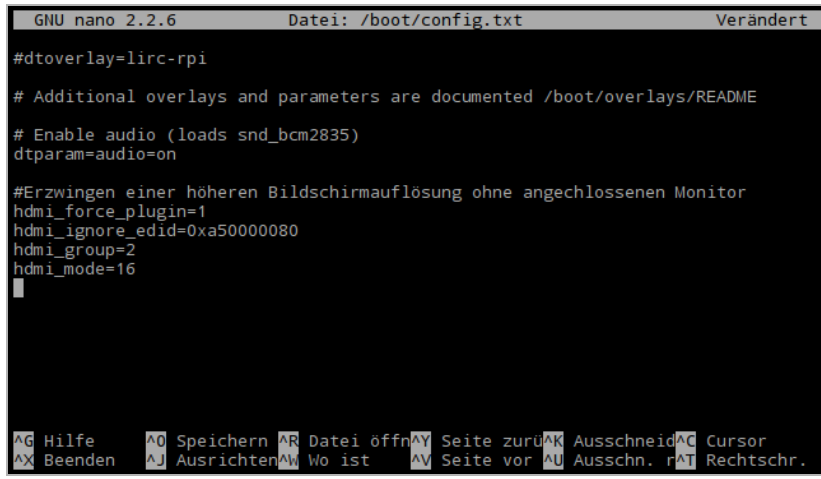


Abb. 5–21 Änderung der Datei config.txt auf dem Raspberry Pi

Die Zeile `hdmi_force_hotplug=1` sagt Ihrem Raspberry Pi, dass ein HDMI-Monitor angeschlossen ist, und `hdmi_mode=16` setzt dessen Auflösung auf 1024x768 Pixel bei einer Bildwiederholfrequenz von 60 Hz.

Drücken Sie jetzt Ctrl-X und anschließend `J`, um Ihre Änderungen zu speichern und `nano` zu beenden. Nach einem Neustart des Raspberry Pi haben Sie eine größere Arbeitsfläche, wenn Sie mit dem VNC-Viewer auf ihn zugreifen.

VNC unter Windows

Laden Sie sich die Software *VNC Connect* herunter: *www.realvnc.com*. Die Software besteht aus den beiden Komponenten VNC-Server und VNC-Viewer. Für die Installation des VNC-Servers müssen Sie sich registrieren und bekommen für die nicht kommerzielle Heimanwendung eine kostenlose Lizenz. Sie können aber den VNC-Viewer separat herunterladen oder bei der Installation von VNC Connect ausschließlich den Viewer installieren, sodass Sie ohne Registrierung auskommen. Für unsere Zwecke brauchen wir auf dem lokalen Computer nur den Viewer, um auf den Raspberry Pi (auf dem der VNC-Server bereits installiert ist) zuzugreifen. Nach der Installation können Sie den VNC-Viewer aus dem Startmenü starten (siehe Abb. 5–22).

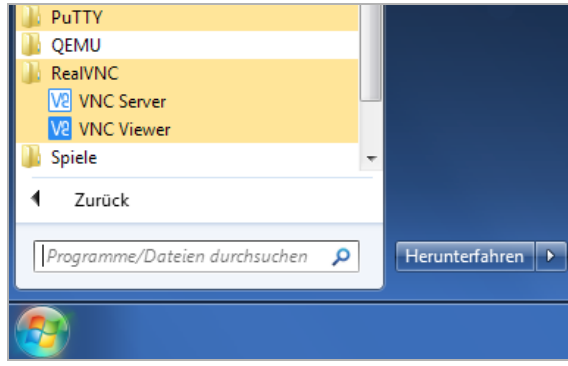


Abb. 5–22 Starten des VNC-Viewers unter Windows

Beim ersten Start des VNC-Viewers fragt er nach einer IP-Adresse eines Remote-Systems. Geben Sie hier die Ihres Raspberry Pi ein und verbinden Sie sich dadurch mit ihm (siehe Abb. 5–23).

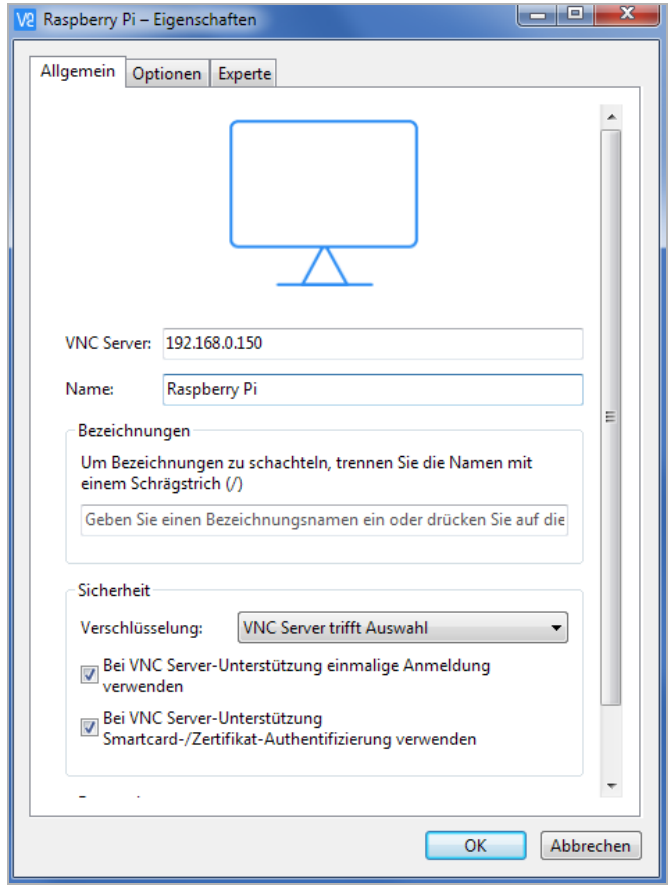


Abb. 5–23 Eingabe der IP-Adresse des Raspberry Pi, um sich über den VNC-Viewer mit ihm zu verbinden

Da dies der erste Zugriff ist, wird Ihnen die zum Server gehörende Signatur angezeigt und gefragt, ob Sie weitermachen wollen. Nachdem Sie zugestimmt haben, werden Benutzername und Passwort des Raspberry Pi abgefragt (siehe Abb. 5–24).

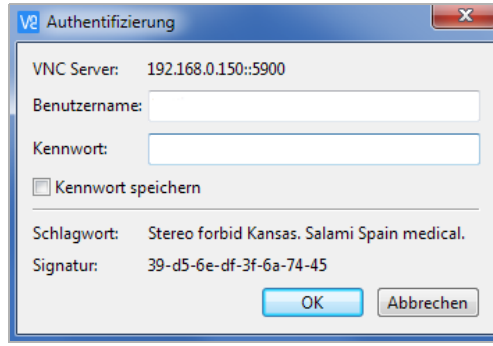


Abb. 5–24 Eingabe von Benutzername und Passwort des Raspberry Pi in den VNC-Viewer

Nach dem Klicken auf OK sollten Sie ein Fenster mit dem Desktop Ihres Raspberry Pi sehen.

VNC unter macOS

Laden Sie sich die Software *VNC Connect* herunter: *www.realvnc.com*. Die Software besteht aus den beiden Komponenten VNC-Server und VNC-Viewer. Für die Installation des VNC-Servers müssen Sie sich registrieren und bekommen für die nicht kommerzielle Heimanwendung eine kostenlose Lizenz. Sie können aber den VNC-Viewer separat herunterladen oder bei der Installation von VNC Connect ausschließlich den Viewer installieren, sodass Sie ohne Registrierung auskommen. Für unsere Zwecke brauchen wir auf dem lokalen Computer nur den Viewer, um auf den Raspberry Pi (auf dem der VNC-Server bereits installiert ist) zuzugreifen. Nach der Installation können Sie den VNC-Viewer über den Finder starten: Programme → RealVNC → VNC Viewer (siehe Abb. 5–25).

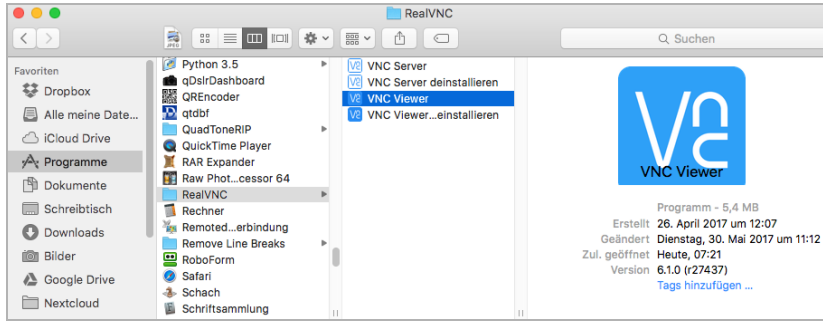


Abb. 5–25 Starten des VNC-Viewers unter macOS

Beim ersten Start des VNC-Viewers fragt er nach einer IP-Adresse eines Remote-Systems. Geben Sie hier die Ihres Raspberry Pi ein und verbinden Sie sich mit ihm (siehe Abb. 5–26).

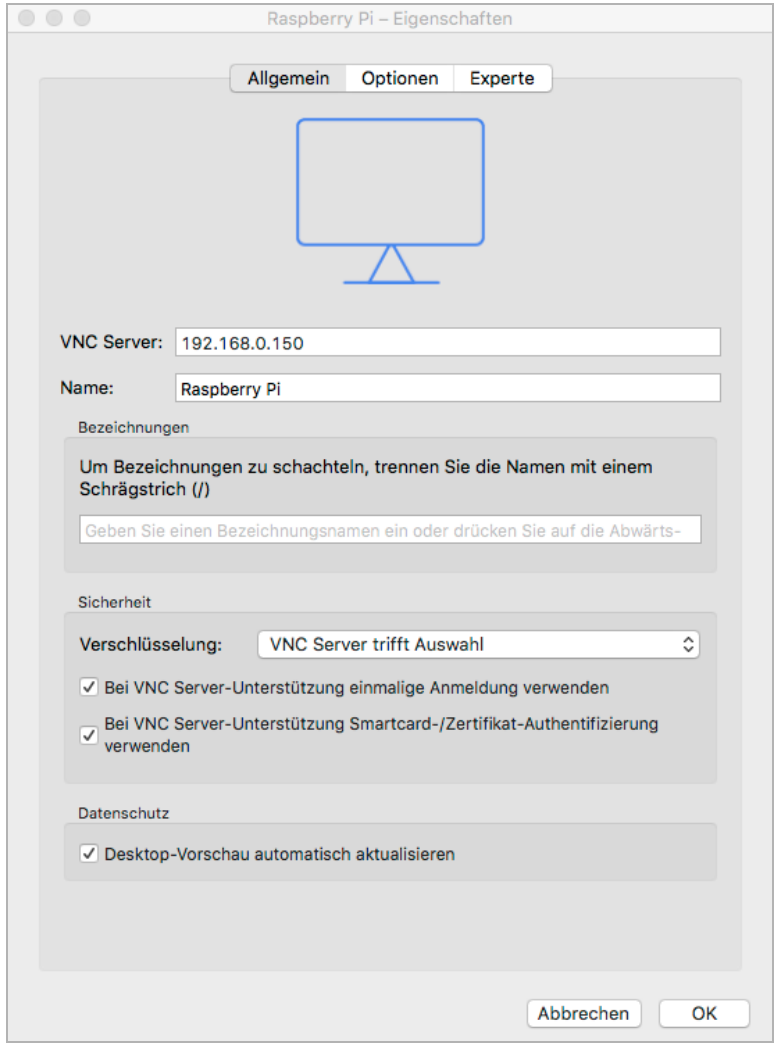


Abb. 5–26 Eingabe der IP-Adresse des Raspberry Pi, um sich über den VNC-Viewer mit ihm zu verbinden

Da dies der erste Zugriff ist, wird Ihnen die zum Server gehörende Signatur angezeigt und gefragt, ob Sie weitermachen wollen. Nachdem Sie zugestimmt haben, werden Benutzername und Passwort des Raspberry Pi abgefragt (siehe Abb. 5–27).

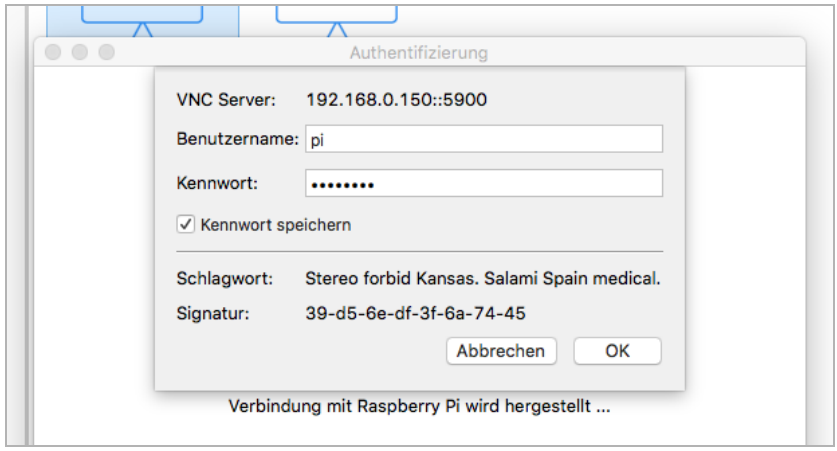


Abb. 5–27 Eingabe von Benutzername und Passwort des Raspberry Pi in den VNC-Viewer

Nach dem Klicken auf OK sollten Sie ein Fenster mit dem Desktop Ihres Raspberry Pi sehen.

VNC unter Linux

Laden Sie sich die Software *VNC Connect* herunter: *www.realvnc.com*. Für Linux gibt es dort viele Softwarepakete für diverse Distributionen, auch für Debian- und Red-Hat-basierte Distributionen. Es gibt aber auch ein allgemeines Linux-Installationspaket. Die Software besteht aus den beiden Komponenten VNC-Server und VNC-Viewer. Für die Installation des VNC-Servers müssen Sie sich registrieren und bekommen für die nicht kommerzielle Heimanwendung eine kostenlose Lizenz. Sie können aber den VNC-Viewer separat herunterladen oder bei der Installation von VNC Connect ausschließlich den Viewer installieren, sodass Sie ohne Registrierung auskommen. Für unsere Zwecke brauchen wir auf dem lokalen Computer nur den Viewer, um auf den Raspberry Pi (auf dem der VNC-Server bereits installiert ist) zuzugreifen. Nach der Installation können Sie den VNC-Viewer von der Taskleiste aus starten (siehe Abb. 5–28).

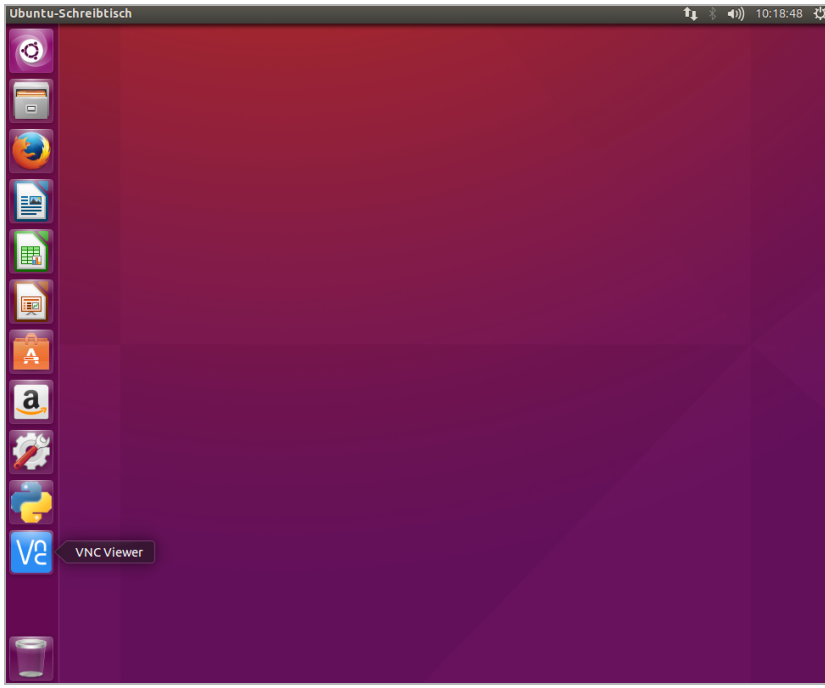


Abb. 5–28 Starten des VNC-Viewers unter der Linux-Distribution Ubuntu

Sie können den VNC-Viewer auch über die Kommandozeile starten:

```
vncviewer
```

Sobald Sie den VNC-Viewer gestartet haben, müssen Sie noch der Endbenutzerlizenz zustimmen, bevor Sie weitermachen können. Ab dort gleicht der Vorgang dem unter Windows und macOS. Geben Sie dann die IP-Adresse Ihres Raspberry Pi ein und klicken Sie auf OK (siehe Abb. 5–29).

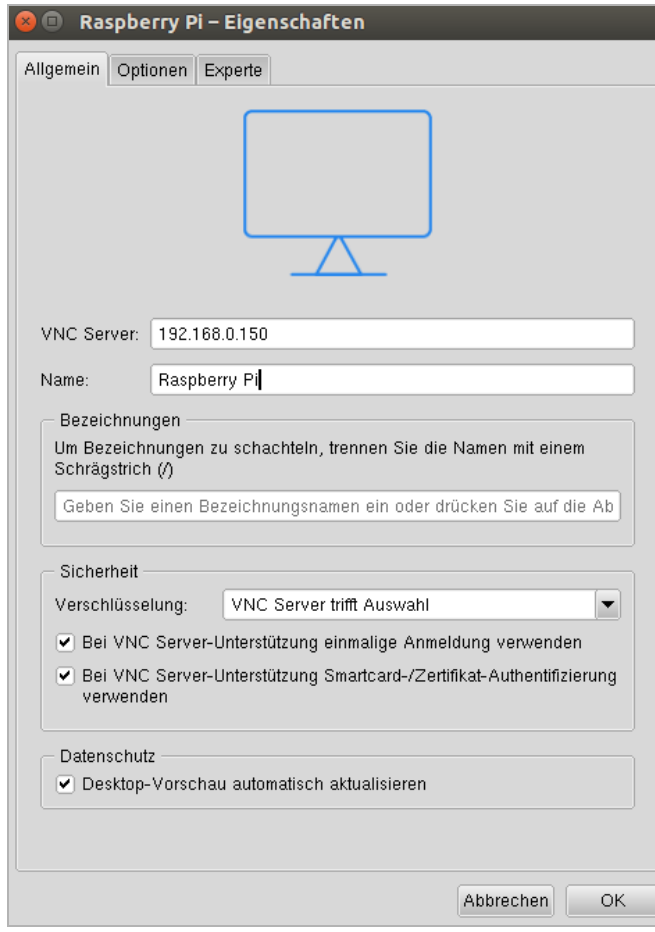


Abb. 5–29 Eingabe der IP-Adresse des Raspberry Pi, um sich über den VNC-Viewer mit ihm zu verbinden

Da dies der erste Zugriff ist, wird Ihnen die zum Server gehörende Signatur angezeigt und gefragt, ob Sie weitermachen wollen. Nachdem Sie zugestimmt haben, werden Benutzername und Passwort des Raspberry Pi abgefragt (siehe Abb. 5–30).

Nach dem Klicken auf OK sollten Sie ein Fenster mit dem Desktop Ihres Raspberry Pi sehen.

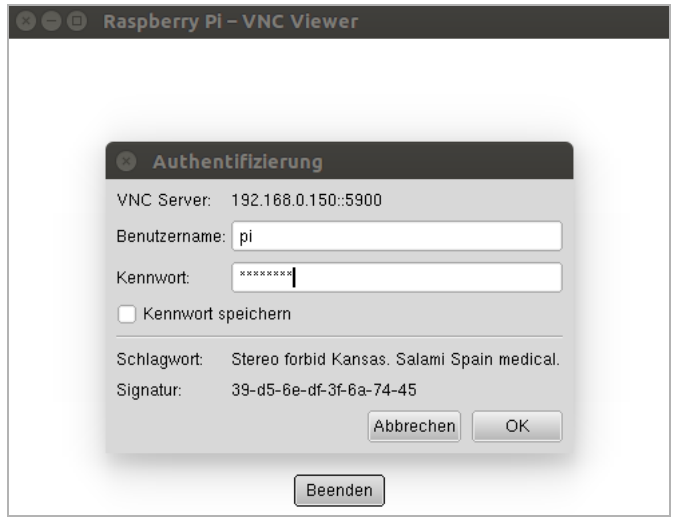


Abb. 5–30 Eingabe von Benutzername und Passwort des Raspberry Pi in den VNC-Viewer

VNC unter Android/iOS

Sie können den Desktop Ihres Raspberry Pi via VNC auch mit Ihrem Smartphone oder Tablet erreichen. Vom VNC-Viewer gibt es sowohl eine Android- (über den Google Play Store) als auch eine iOS-Version (Apple App Store). Der Verbindungsvorgang ist den Desktop-Versionen von VNC-Viewer sehr ähnlich, sodass Ihnen die Einrichtung leicht fallen dürfte. Schwieriger dürfte die Bedienung des Raspberry-Pi-Desktops über kleine Smartphone-Displays werden (siehe Abb. 5–31).

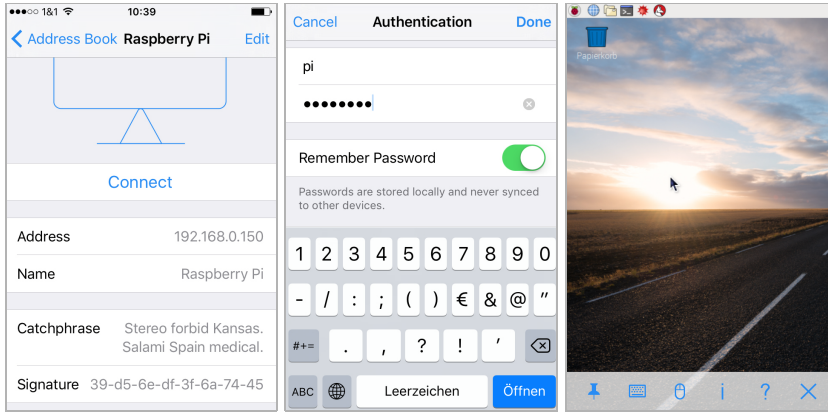


Abb. 5–31 Zugriff auf den Desktop des Raspberry Pi mit der mobilen App des VNC-Viewers

5.5 Dateiübertragungen: scp, sftp

Nachdem wir unseren Raspberry Pi über das Netzwerk bedienen können, bleibt noch die Frage, wie wir Dateien mit ihm austauschen können. Dazu gibt es bei allen SBCs mit Linux viele Möglichkeiten. Doch gerade die scheinbar simpelste, nämlich direkt von Windows oder macOS auf die SD-Karte zu kopieren, ist in Wirklichkeit nicht so einfach. Da die SD-Karte unter Linux für den Großteil der Datenspeicherung ein anderes Dateisystem benutzt, ist dieser Teil der Speicherkarte unsichtbar, wenn man die Karte einfach in ein Windows- oder macOS-System steckt. Man könnte zwar versuchen, ein Netzlaufwerk einzurichten oder Daten mit einem USB-Stick zu übertragen, doch ist das alles mühsam und relativ zeitaufwändig. Es wird noch schwieriger, wenn das System auch noch ohne Desktop und headless betrieben wird.

Aber es gibt einfache Lösungen, die bereits in Linux enthalten und stets zur Stelle sind, wenn es darum geht, ein paar Dateien über SSH sicher über das Netzwerk zu übertragen: *Secure Copy* (SCP) und *Secure File Transfer Protocol* (SFTP). Mit `scp` überträgt man am besten einzelne Dateien oder Ordner, wohingegen `sftp` auch neue Ordner anlegen und viele Dateien übertragen kann, wie man es vom normalen FTP kennt. So etwas geht zugegebenermaßen über eine grafische Benutzeroberfläche einfacher als über die Kommandozeile, doch zeige ich Ihnen beide Möglichkeiten, sofern sie zur Verfügung stehen.

Dateiübertragung mit VNC

Sie können auch über den VNC-Viewer von RealVNC Dateien auf Ihren Raspberry Pi übertragen, wenn er einen Desktop gestartet hat. Falls nicht, sind SCP und SFTP immer bereit, sodass es nützlich ist, auch diese Möglichkeiten zu kennen.

Dateiübertragung unter Windows

Unter Windows müssen Sie ein Programm zu Hilfe nehmen, um SCP oder SFTP nutzen zu können. Mit WinSCP haben Sie ein hervorragendes Programm zur Verfügung, das beides kann und dazu eine nette Drag-and-Drop-Oberfläche bietet. Sollte PuTTY bereits installiert sein, kann es auch dies starten, wenn es gebraucht wird. Sie können WinSCP kostenlos herunterladen: *www.winscp.net*.

Sobald sie es heruntergeladen und installiert haben, öffnen Sie es und wählen in den *Einstellungen* unter *Umgebung* eine *Oberfläche*. Ich bevorzuge die *Commander*-Oberfläche, da man mit ihr bequem Dateien zwischen lokalem Computer und Raspberry Pi hin- und herschieben kann. Als Erstes wird man allerdings mit einem Login-Fenster konfrontiert (Abb. 5–32).

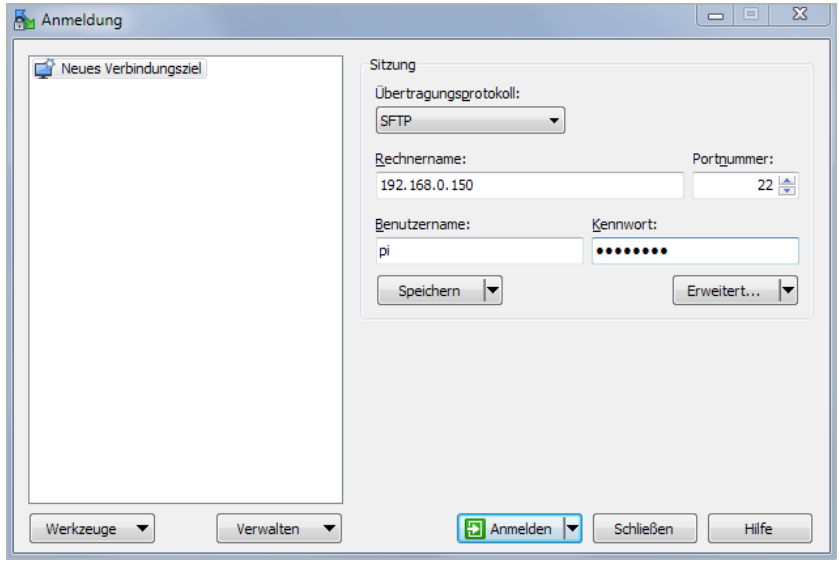


Abb. 5–32 Verbindung mit Raspberry Pi über WinSCP

Lassen Sie das Übertragungsprotokoll auf *SFTP* eingestellt (SFTP und das ältere SCP funktionieren in diesem Programm auf die gleiche Weise) und geben Sie die IP-Adresse Ihres Raspberry Pi in dem Kästchen *Rechnername* ein. Danach geben Sie *Benutzername* und *Kennwort* (Passwort) ein und klicken auf den Button *Anmelden*, um sich über das Netzwerk zu verbinden. Ist es das erste Mal, dass Sie sich mit Ihrem Raspberry Pi mit WinSCP verbinden, werden Sie gebeten, den Schlüssel des Servers zu akzeptieren. Sobald die Verbindung steht, sehen Sie die von Ihnen gewählte Oberfläche (siehe Abb. 5–33 mit der Oberfläche *Commander*).

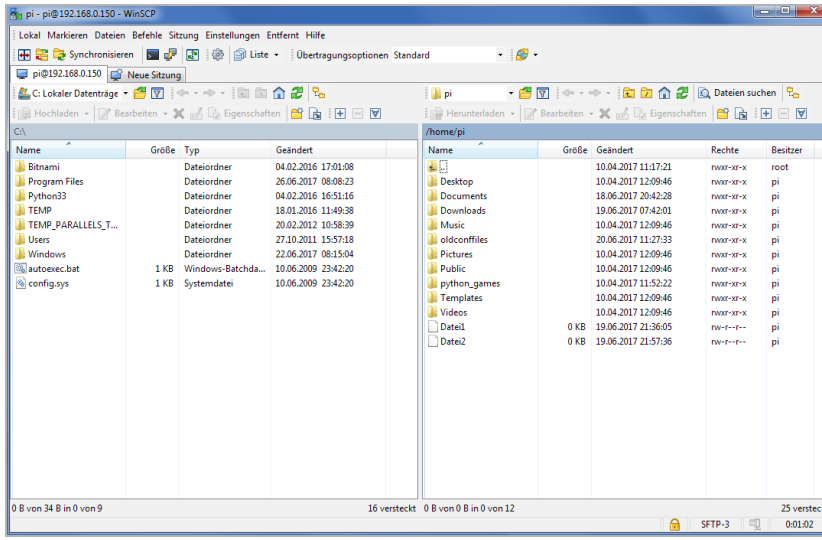


Abb. 5–33 Das Dateimanager-Fenster in WinSCP

Jetzt können Sie ganz einfach Dateien zwischen Ihrem Computer und Ihrem Raspberry Pi übertragen.

Dateiübertragung unter macOS

Für macOS gibt es viele Clients für die Dateiübertragung über SCP und SFTP. Einige von ihnen sind allerdings sehr umfangreich und entsprechend teuer. Es gibt im Apple App Store allerdings auch kostenlose Programme, die die einfachen Aufgaben des Dateitransfers zwischen Raspberry Pi und anderen SBCs und dem Mac zuverlässig erledigen. Eines davon in *Commander One*. Die kostenlose Variante von Commander One bietet eine übersichtliche Spaltenansicht der jeweiligen Dateisysteme, lässt sich unter einer Vielzahl von Übertragungsprotokollen betreiben und ist sehr bedienungsfreundlich. Man kann das Programm über die Entwickler-Website (*https://mac.eltima.com/file-manager.html*) herunterladen oder über den Apple App Store installieren.

Nach der Installation öffnen Sie es und sehen die Standardansicht. Um nun eine SFTP-Sitzung mit Ihrem Raspberry Pi zu öffnen, klicken Sie auf das Icon für den *Verbindungsmanager* (siehe Abb. 5–34).

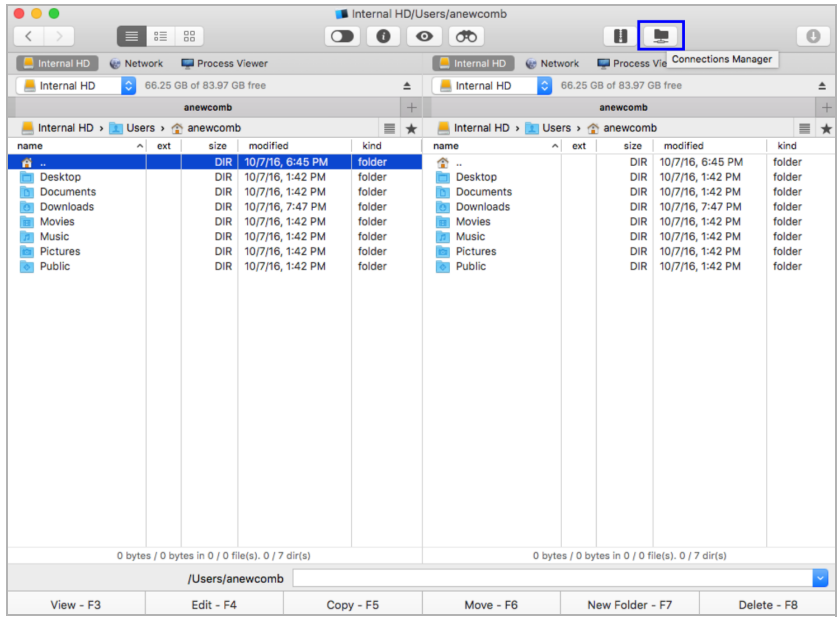


Abb. 5–34 Die Standardansicht des Dateimanagers in Commander One

Im Verbindungsmanager können Sie aus diversen Verbindungsprotokollen wählen. Entscheiden Sie sich hier durch Klicken auf den entsprechenden Button für SFTP (siehe Abb. 5–35).

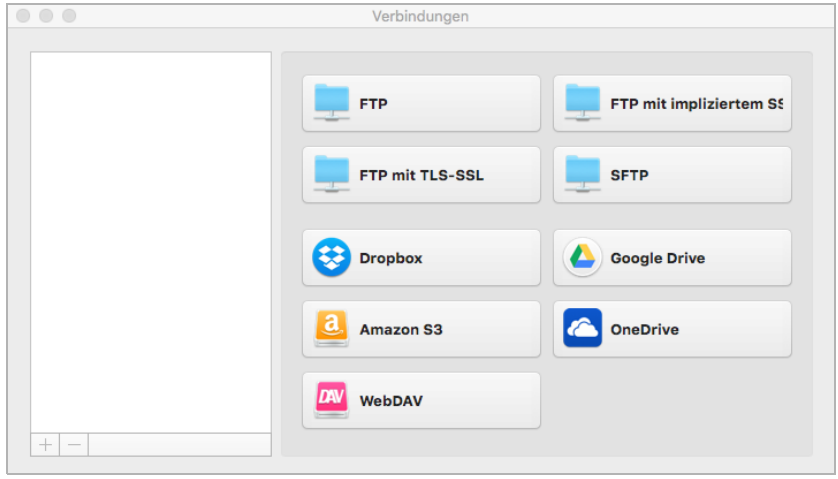


Abb. 5–35 Auswahl des Übertragungsprotokolls im Verbindungsmanager von Commander One

Im Fenster *Neue Verbindung* geben Sie dann bei *Server* die IP-Adresse Ihres Raspberry Pi, bei *Login* den Benutzernamen und dann noch das Passwort ein. Klicken Sie dann auf den Button *Verbinden* (siehe Abb. 5–36).

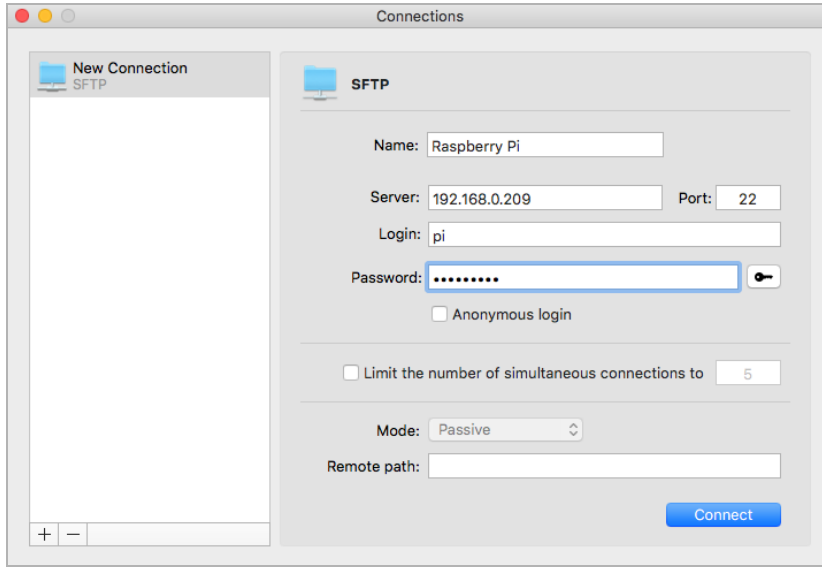


Abb. 5–36 Verbindungseinstellungen in Commander One

Eine der Spalten im Dateimanager wird durch den Inhalt des Dateisystems Ihres Raspberry Pi ersetzt. Zudem gibt es jetzt ganz oben einen Eintrag im Auswahlmenü über der Spalte. Jetzt können Sie ganz einfach Dateien zwischen Mac und Raspberry Pi bewegen. Falls Sie die Rechte einer Datei ändern müssen, können Sie dies per Rechtsklick und Auswahl von *Informationen* oder Ctrl-I ganz einfach erledigen (siehe Abb. 5–37).

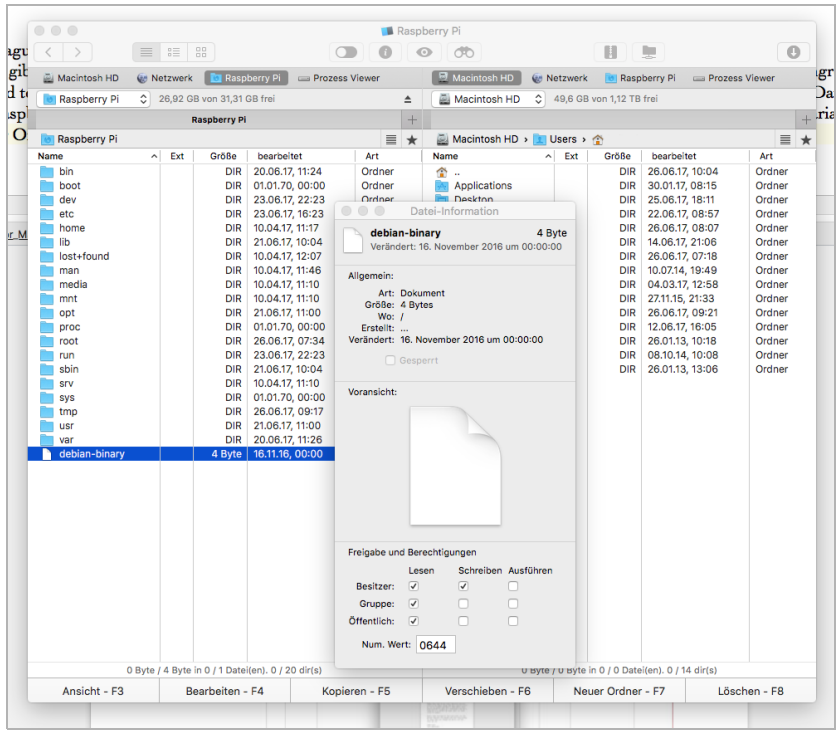


Abb. 5–37 Änderung der Rechte einer Datei in Commander One

Dateiübertragung unter Linux

In den meisten Linux-Distributionen ist innerhalb des Dateimanagers bereits die Funktion integriert, sich über das Netzwerk mit anderen Computern verbinden zu können. Unter den zahlreichen Distributionen ist hier Ubuntu gewählt, eine der zurzeit beliebtesten.

Öffnen Sie also den Dateimanager und gehen Sie in der linken Spalte auf *Mit Server verbinden* (siehe Abb. 5–38).

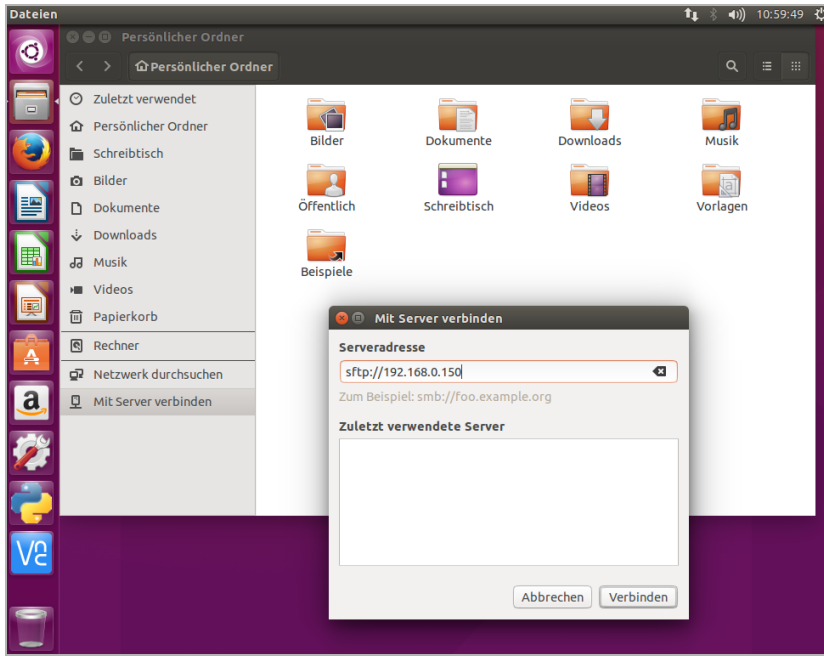


Abb. 5–38 Hinzufügen einer neuen Verbindung zu einem Server im Dateimanager von Ubuntu

Geben Sie dann unter *Serveradresse* Folgendes ein: sftp://IP-Adresse des Raspberry Pi (z.B. 192.168.0.150). Nach Klicken auf *Verbinden* werden Sie aufgefordert, Benutzername und Passwort einzugeben (siehe Abb. 5–39). Sobald Sie wieder auf *Verbinden* geklickt und die Verbindung bestätigt haben, sehen Sie das Dateisystem Ihres Raspberry Pi im Dateimanager. Ähnlich einem externen Laufwerk finden Sie ein Icon mit dieser Verbindung in der Spalte links und ein Symbol für Auswerfen, also das Beenden dieser Verbindung (siehe Abb. 5–40).

Abb. 5–39 Eintrag der Verbindungsdaten für den Raspberry Pi unter Ubuntu

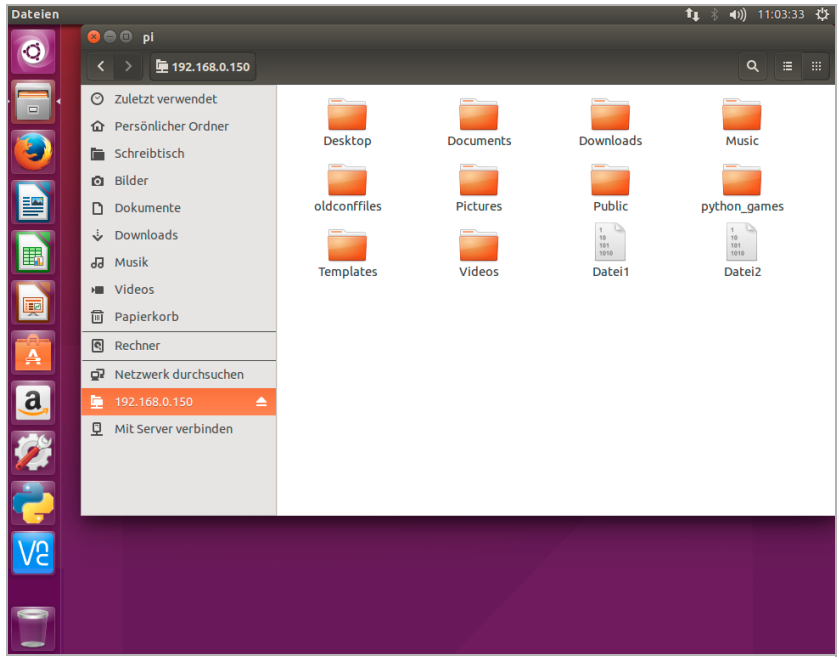


Abb. 5–40 Die Dateiansicht des Raspberry Pi nach erfolgreicher Verbindung unter Ubuntu

Dateiübertragung über die Kommandozeile: macOS und Linux

Sowohl unter macOS als auch unter Linux kann man über die mitgelieferten Terminalemulatoren SCP und/oder SFTP verwenden. Um schnell eine einzige Datei oder einen Ordner auf diese Weise auf einen verbundenen Raspberry Pi zu übertragen, ist SCP sehr gut geeignet. Da SFTP etwas komplizierter ist und zudem die Kenntnis der FTP-Befehle voraussetzt, wird es über die Kommandozeile selten eingesetzt, sodass ich mich hier auf die Beschreibung von SCP beschränke.

Öffnen Sie also in macOS oder Linux eine Terminalsitzung. Die Syntax des Befehls `scp` ähnelt der von `ssh`. Um also eine Datei in das Homeverzeichnis Ihres Raspberry Pi zu kopieren, geben Sie Folgendes ein:

```
scp meineDatei Benutzername@IP-Adresse:/home/pi
```

Dabei ist:

- `meineDatei`
  der Dateiname der zu übertragenden Datei (im Homeverzeichnis des lokalen Computers)
- `Benutzername`
  der Benutzername des Systems, auf das zugegriffen wird
- `IP-Adresse`
  die IP-Adresse des Systems, auf das zugegriffen wird
- `:/home/pi`
  das Zielverzeichnis auf dem Raspberry Pi

Achten Sie auf das Leerzeichen zwischen dem Dateinamen und dem Benutzernamen. Nachdem Sie Ihr Passwort eingegeben haben, wird Ihnen während des Kopiervorgangs der Fortschritt der Dateiübertragung angezeigt (siehe Abb. 5–41).

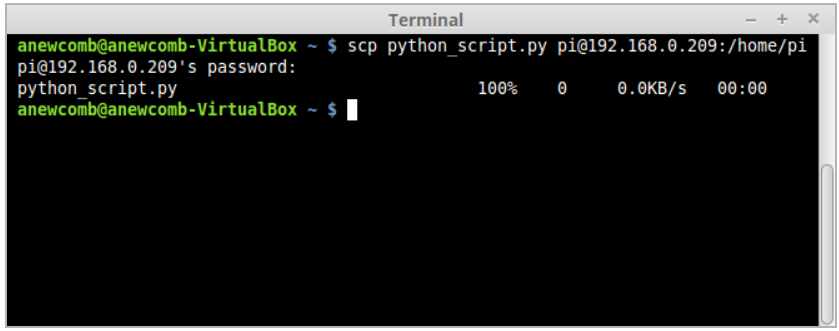


Abb. 5–41 Übertragung einer Datei mit dem Befehl scp über die Kommandozeile unter Linux

5.6 Warum dies für Maker wichtig ist

Ich habe festgestellt, dass es eher die Regel als die Ausnahme ist, dass Maker den Raspberry Pi oder einen anderen SBC für Projekte einsetzen und der permanente Anschluss von Monitor und Tastatur sehr unpraktisch wäre. Roboter, Überwachungskameras und LED-Leisten sind Projekte, die Spaß machen, aber am ehesten im Headless-Betrieb überzeugen. Zu wissen, wie man mit dem Raspberry Pi von außen kommuniziert und ihn steuert, eröffnet völlig neue Möglichkeiten, die es zu entdecken gilt.

6 Tipps und Tricks

Jetzt, wo Sie wissen, wie man die Kommandozeile von überall her einsetzt, werde ich Ihnen einige Funktionen erklären, die ein tüchtiger Maker kennen sollte, um seine Programme für sich arbeiten zu lassen. Die hier behandelten Themen kommen in Foren und in Gesprächen mit Linux-Neulingen immer wieder auf und gehören zu den absoluten Grundfertigkeiten eines Systemadministrators. Die Liste dieser Fertigkeiten ist nicht superlang, aber Sie können damit jede Menge Zeit bei Ihren Projekten sparen und Ihre Freunde mächtig beeindrucken, wenn Sie ihnen Ihr Können mit der Kommandozeile demonstrieren.

6.1 Hostnamen ändern

Der voreingestellte Hostname eines Raspberry Pi, auf dem Raspbian läuft, ist *raspberrypi*. Sobald man mehr als einen davon in seinem Netzwerk betreibt, kann man Schwierigkeiten bekommen, sie auseinanderzuhalten. Deshalb ist es ratsam, ihnen verschiedene Hostnamen zu geben. Die Änderung in jeden beliebigen Hostnamen geht in wenigen einfachen Schritten.

Keine Angst vor der Änderung!

Die Änderung des Hostnamens eines Raspberry Pi hat keinen Einfluss auf dessen IP-Adresse. Das einzige was sich wirklich ändert, ist der angezeigte Name, der in Netzwerkscannern wie Fing oder auf einer Netzwerkkonfigurationsseite wie der eines Routers zu sehen ist.

Als Erstes überprüfen wir den aktuellen Hostnamen durch den entsprechenden Befehl:

```
hostname
```

Ohne weitere Optionen gibt uns dieser Befehl den Hostnamen aus, ohne irgendetwas zu ändern (siehe Abb. 6–1).

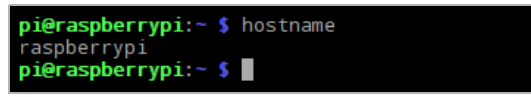


Abb. 6–1 Eingabe des Befehls hostname, um den Hostnamen abzufragen

Als Nächstes müssen wir die Datei *hosts* ändern. Diese spezielle Datei ist wie ein persönlicher Plan, den Linux verwendet, um Hostnamen den entsprechenden IP-Adressen zuzuordnen. Die hier abgelegten Informationen stehen über denen, die eventuell von anderen Geräten im Netzwerk kommen. Für die Änderung des Hostnamens müssen Sie nun alle Stellen, an denen `raspberrypi` steht, durch Ihren gewünschten Hostnamen ersetzen (siehe Abb. 6–2):

```
sudo nano /etc/hosts
```

In den Abbildungen 6–2 und 6–3 können Sie sehen, wie ich `raspberrypi` in `virtualpi` geändert habe.

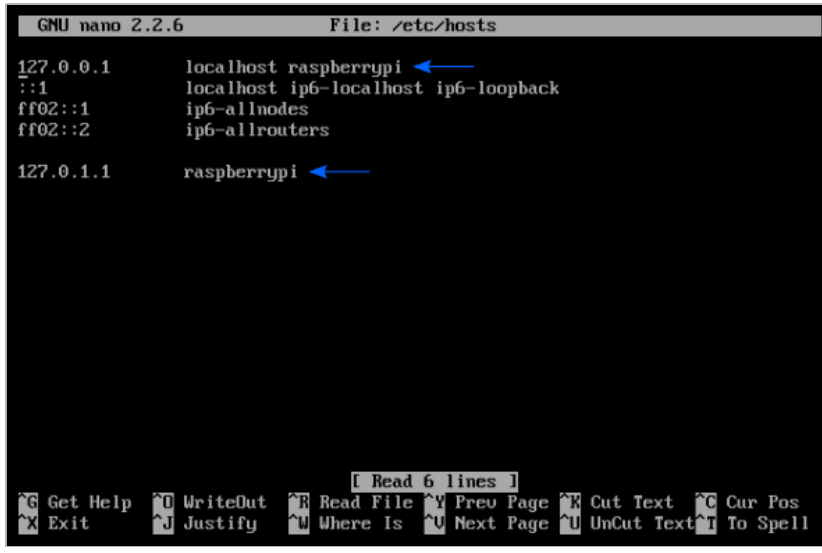


Abb. 6–2 Änderung der Datei hosts

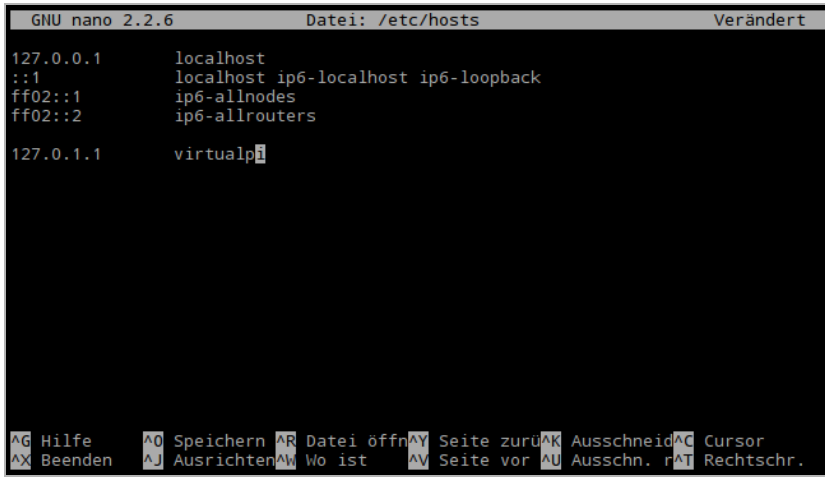


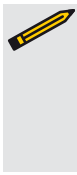
Abb. 6–3 Die Datei hosts mit neuem Hostnamen

Nach dem Speichern und Schließen von *hosts* besteht der letzte Schritt in der Änderung der Datei *hostname*. In dieser Datei steht einfach nur der Name Ihres Systems. Öffnen Sie sie wieder in `nano` und tauschen Sie auch hier `raspberrypi` mit dem in *hosts* verwendeten Namen aus (siehe Abb. 6–4):

```
sudo nano /etc/hostname
```

Abb. 6–4 Änderung der Datei hostname

Nach dem Speichern von *hostname* und Schließen von `nano`, starten Sie Ihren Raspberry Pi neu, damit wirklich alle Programme, die mit dem Hostnamen arbeiten, den neuen verwenden. Ab jetzt sollte Ihr Raspberry Pi in den Netzwerkübersichten unter dem neuen Namen erscheinen. Im Prompt sehen Sie ihn natürlich auch (siehe Abb. 6–5). Der Befehl `hostname` gibt den neuen Hostnamen selbstverständlich auch jederzeit aus.



Sorgen Sie für den Abgleich

Sollten die Namen in den Dateien *hosts* und *hostname* abweichen, können Sie Probleme bekommen, über das Netzwerk auf Ihr System zuzugreifen. Falls das einmal passieren sollte, müssen Sie Ihren Raspberry direkt an Tastatur und Monitor anschließen, um dies zu beheben.

Abb. 6–5 Der Hostname nach dem Befehl hostname und im Prompt

Anschließend überprüfen Sie, ob der neue Hostname in Ihrem Netzwerk angezeigt wird (Netzwerkscanner, Router-Website).

6.2 Skript beim Hochfahren starten: rc.local

Als Maker kommen Sie zwangsläufig in Situationen, dass Sie ein Programm oder Skript automatisch starten lassen wollen, sobald Ihr Raspberry Pi hochgefahren ist. Dies ist vor allem dann der Fall, wenn Sie ihn im Headless-Betrieb laufen lassen und Ihr Projekt zum Leben erweckt werden soll, sobald es eingeschaltet wurde. Am einfachsten lässt sich dies bewerkstelligen, indem Sie Ihr Skript oder Programm in die Datei *rc.local* einfügen, die sich im Verzeichnis */etc* befindet. Mit `sudo nano` fügen Sie dort eine Zeile ein, die Ihr Skript oder Programm automatisch starten lässt. Der Eintrag erfolgt direkt vor dem Statement `exit 0` (siehe Abb. 6–6):

```
sudo nano /etc/rc.local
```

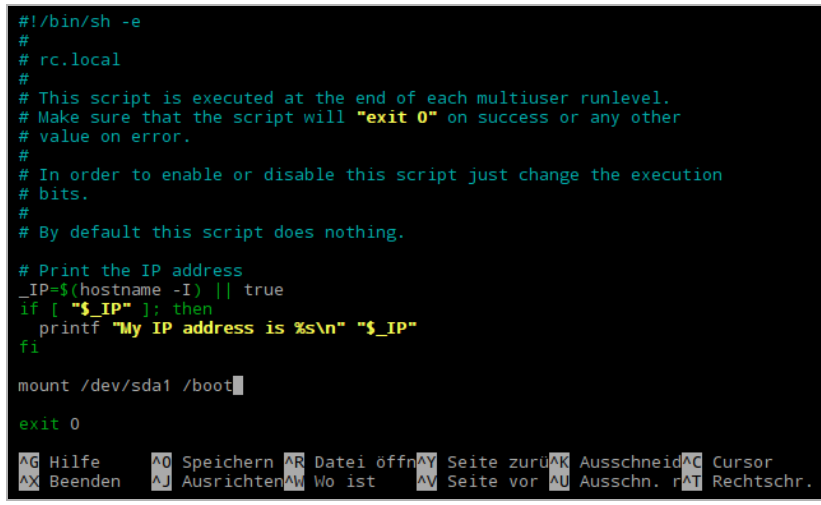


Abb. 6–6 Änderung der Datei rc-local

Kein sudo nötig

Das Skript *rc.local* läuft als »root« und benötigt daher keinen extra Befehl `sudo`, wie man ihn sonst verwenden müsste, wenn man als normaler Benutzer »pi« angemeldet ist. Allerdings muss der vollständige, also absolute Pfad zu Ihrem Skript oder Programm eingetragen werden.

6.3 Probieren Sie es selbst

Fügen Sie das Skript *hallo.sh* in *rc.local* ein, damit es beim Hochfahren gestartet wird. Öffnen Sie als Erstes also die Datei *rc.local*:

```
sudo nano /etc/rc.local
```

Nun fügen Sie den in Kapitel 2 verwendeten Befehl ein, um das Skript laufen zu lassen, aber mit dem vollständigen Pfad zur Datei (siehe Abb. 6–7).

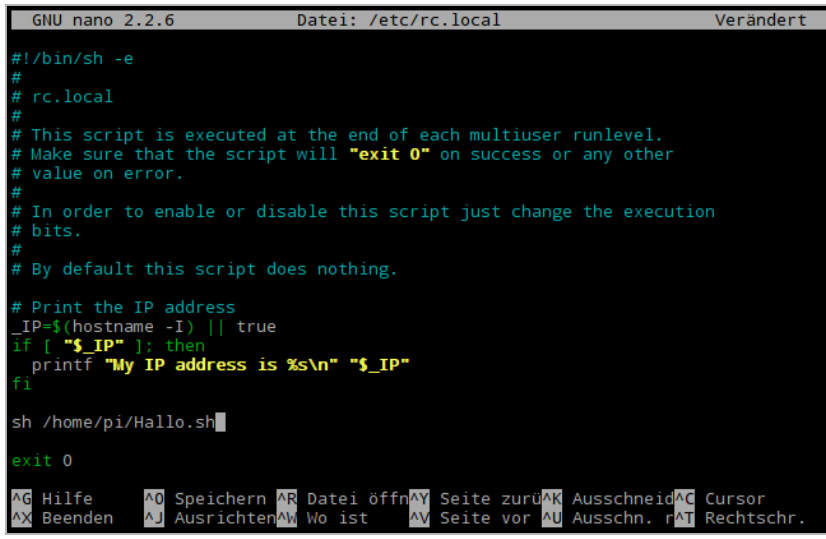


Abb. 6–7 Mithilfe der Datei rc.local ein Skript oder Programm nach dem Hochfahren starten lassen

Speichern Sie wieder die Datei, indem Sie `nano` mit Ctrl-X schließen, die Speicherung der Änderung mit `J` bestätigen und dann Enter drücken. Anschließend starten Sie Ihr System mit folgendem Befehl neu:

```
sudo shutdown -r now
```

Beim Hochfahren des Systems halten Sie nach den Ausgaben von »Hallo Welt!« Ausschau, die gegen Ende des Bootvorgangs erscheinen (siehe Abb. 6–8).

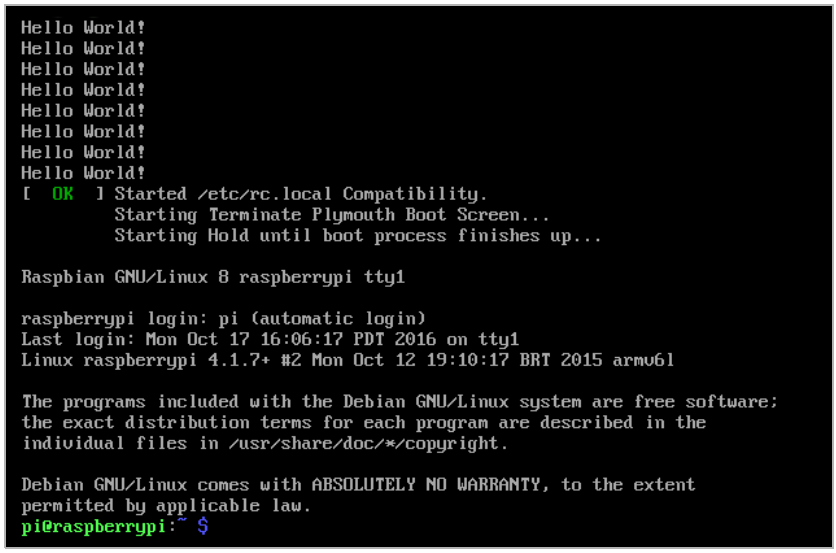


Abb. 6–8 Das Skript hallo.sh startet beim Hochfahren.

6.4 Aliase

In Linux sagen Aliase der Shell: »Wenn ich dies eingebe, sollst du eigentlich das tun.« Wenn Sie beispielsweise in der Kommandozeile Ihres Raspberry Pi `ls` eingeben, führt die Shell in Wirklichkeit `ls -- color=auto` aus. Dies liegt daran, dass die meisten Terminals heutzutage Farben unterstützen, doch standardmäßig schaltet `ls` die Farboption nicht ein. Jedes Mal `--color=auto` einzugeben, wäre äußerst lästig und deshalb gibt es einen Alias, der das übernimmt.

Die Aliase werden in Linux überwiegend in einem Skript namens *.bashrc* verwaltet. Jeder Benutzer hat seine eigene *.bashrc*-Datei in seinem Homeverzeichnis. Um also entweder einen eigenen Alias anzulegen oder einen existierenden zu ändern, muss man Änderungen an dieser Datei vornehmen. Dazu rufen wir sie im Terminal folgendermaßen auf:

```
nano .bashrc
```

Bei Änderungen an dieser Datei müssen Sie große Sorgfalt walten lassen, da sie jede Menge Einstellungen und Konfigurationsinformationen für Ihre Shell enthält. Scrollen Sie so weit nach unten, bis Sie den Abschnitt sehen, in dem die Aliase für Ihre Sitzung definiert sind (siehe Abb. 6–9).

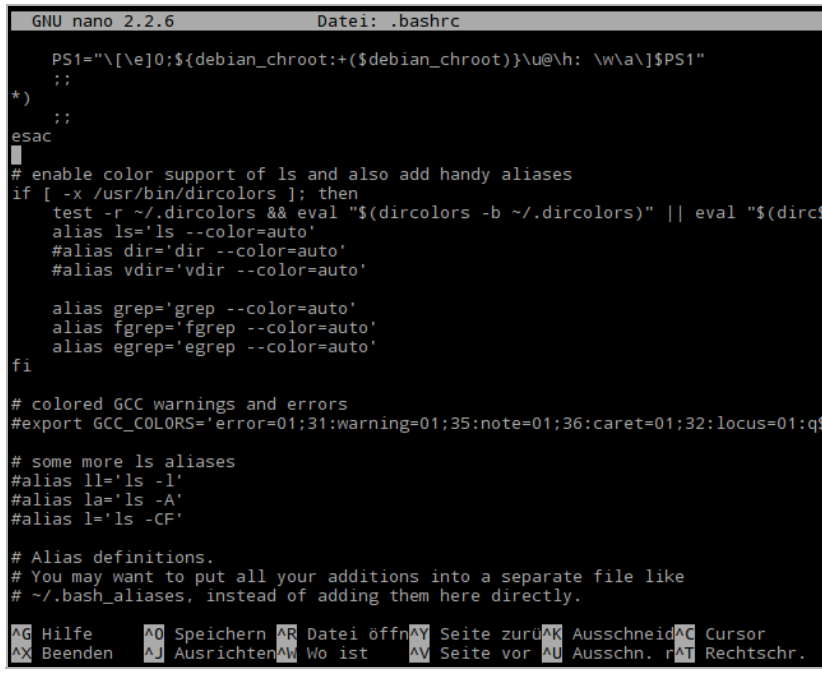


Abb. 6–9 Änderung der Datei .bashrc

Ein guter Ort, um seine Aliase zu platzieren, ist direkt nach der `fi`-Anweisung dieses Abschnitts, da so alles schön zusammensteht. Technische Gründe für die Platzierung gibt es jedoch nicht, weshalb Anweisungen in einer Datei wie dieser auch gerne ganz unten hineingeschrieben werden, damit man sie leicht findet. Die Syntax ist dafür umso wichtiger und so sollte man darauf achten, dass der neue Aliasname in einem Wort, also ohne Leerzeichen, geschrieben wird (z.B. »startedies« und nicht »starte dies«). Außerdem dürfen vor und nach dem Gleichheitszeichen (`=`) keine Leerzeichen stehen.

Abmelden, damit Änderungen wirksam werden

Die Datei *.bashrc* wird nur beim Login in das System ausgelesen, sodass man sich, um die Änderungen dieser Datei wirksam werden zu lassen, ab- und wieder anmelden muss, und zwar über die Kommandozeile mit dem Befehl `exit`.

6.5 Probieren Sie es selbst

Öffnen Sie die Datei *.bashrc* im Editor:

```
nano .bashrc
```

Fügen Sie einen Alias namens `lsbydate` hinzu, der die Ausgabe des Befehls `ls` nach dem Datum der letzten Änderung in aufsteigender Reihenfolge sortiert (siehe Abb. 6–10).

```
alias lsbydate='ls -ltr'
```

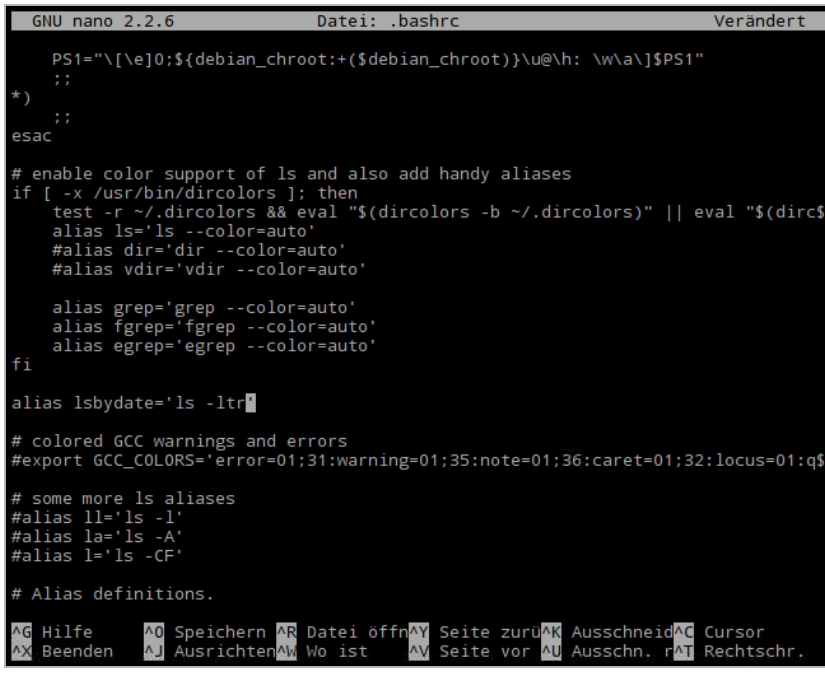


Abb. 6–10 Hinzufügen des Alias lsbydate zur Datei .bashrc

Speichern Sie die Datei durch Schließen des Editors mit Ctrl-X, dann J, dann Enter. Beenden Sie Ihre Terminalsitzung:

```
exit
```

Anschließend melden Sie sich wieder als der gleiche Benutzer an und geben Ihren Befehl ein. Jetzt sollten Sie die Dateien und Verzeichnisse Ihres aktuellen Verzeichnisses im Dateisystem nach Datum sortiert sehen (siehe Abb. 6–11).

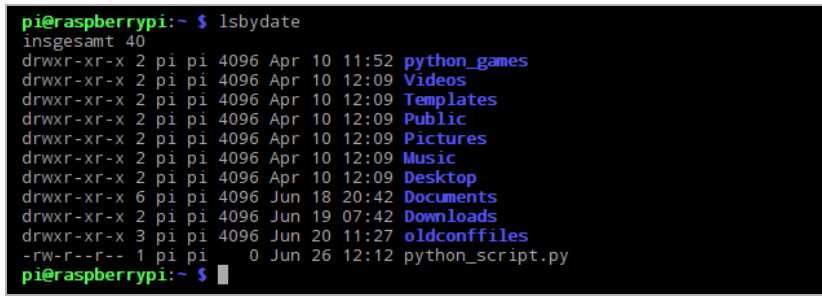


Abb. 6–11 Einsatz des neuen Alias lsbydate

6.6 Festplattenbelegung und Dateigrößen abfragen: df, du

Auch wenn man zusätzlichen Speicherplatz über die USB-Ports an seinem Raspberry Pi anschließen kann, so bleibt die SD-Karte doch das Hauptlaufwerk. Da ihr Speicherplatz sehr begrenzt ist, möchten Sie bestimmt wissen, wie viel davon schon belegt ist und wie viel noch frei ist. Mit dem Hilfsprogramm `df` (für *disk filesystem*) können Sie das ganz einfach herausfinden.

Das Programm `df` stellt eine Liste aller aktiven Laufwerke auf mit deren Gesamtgröße, belegtem und freiem Speicherplatz, prozentualem Anteil der Belegung und wo das Dateisystem aktiv ist. Die Dateisysteme, auf die es besonders zu achten gilt, sind die unter / und `/boot`, da sie den Hauptspeicherort und die Bootpartition darstellen.

Standardmäßig wird die Ausgabe von `df` in Kilobytes (1024 Bytes) dargestellt. Durch die Option `-h` ist die Ausgabe besser lesbar (siehe Abb. 6–12).

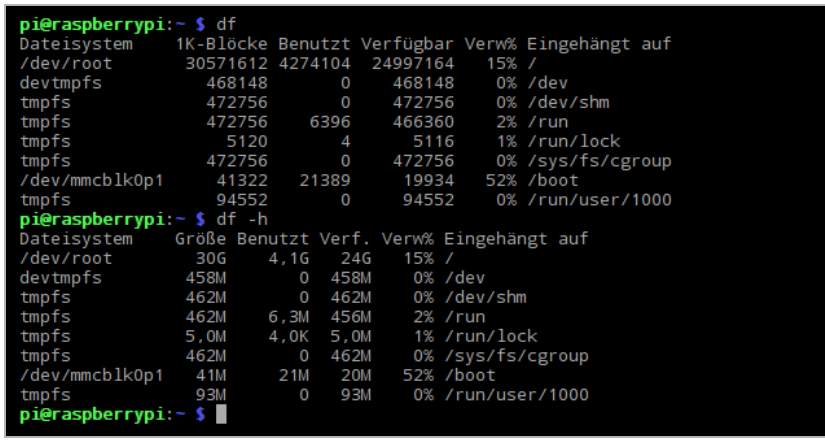


Abb. 6–12 Ausgabe nach dem Befehl `df`

Vielleicht möchten Sie auch noch wissen, wie viel Platz eine bestimmte Datei oder ein Verzeichnis in Ihrem Dateisystem in Anspruch nimmt. Dazu könnten Sie den Befehl `ls` eingeben, um eine Liste Ihrer Dateien zu bekommen, und die Speicherangaben addieren. Einfacher geht das jedoch mit dem Befehl `du` (*disk usage*). Ohne weitere Option zeigt `du` die Größe einer jeden Datei an, ausgehend vom momentanen Ort und dann weiter durch das gesamte Dateisystem, bis es nichts mehr anzuzeigen gibt.

Wie `df` werden bei `du` die Dateigrößen in Kilobytes (1024) angezeigt. Mit der Option `-h` werden auch diese in gut lesbarer Form dargestellt. Wenn dies zu unübersichtlich ist, kann man mit der Option `-d` die Anzahl der Unterverzeichnisse festlegen, deren Speicherbedarf ausgewiesen werden soll – ausgehend vom aktuellen Ort im Dateisystem. Möchte man also den Speicherplatz für sein aktuelles Verzeichnis wissen, übergibt man als Argument `-d 0` anzugeben. Sollen die Unterverzeichnisse der nächsten Ebene mit einbezogen werden, muss es `-d 1` heißen (siehe Abb. 6–13).

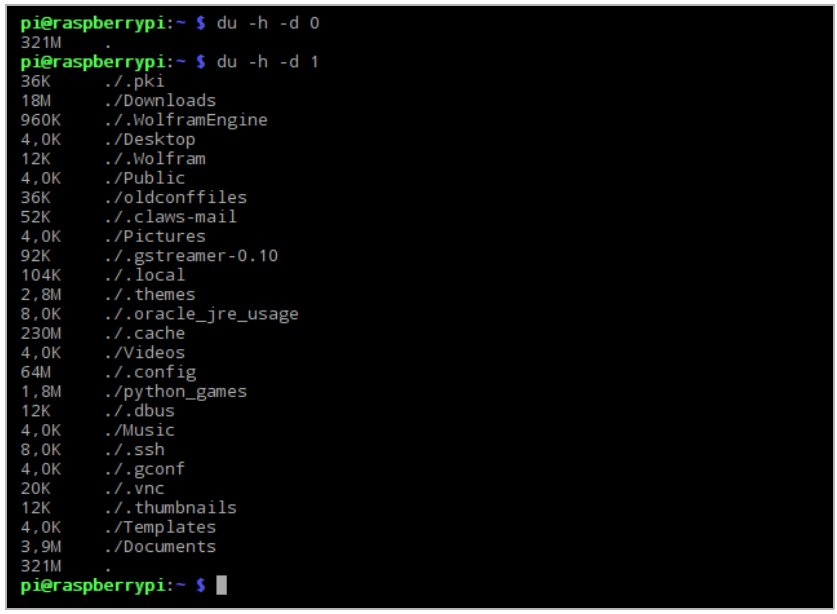


Abb. 6–13 Ausgabe nach dem Befehl du

In Abbildung 6–13 sieht man, dass der belegte Speicherplatz meines aktuellen Verzeichnisses (*/home/pi*) 172 MB beträgt, der des Unterverzeichnisses *Documents* 3,9 MB.

6.7 Systemauslastung überprüfen: top

Dem Thema Leistungsfähigkeit unter Linux sind ganze Bücher gewidmet, doch statt auf alles einzugehen, beschränke ich mich auf die Grundlagen, die jeder Maker kennen sollte. Hat man erst einmal einen Eindruck von der Vielzahl der Möglichkeiten von Projekten mit SBCs wie dem Raspberry, könnte die Verlockung groß sein, mit ihm sehr vieles gleichzeitig zu machen. Es stimmt ja auch, dass im Gegensatz zu Mikrocontrollern wie dem Arduino ein SBC fast gleichzeitig Sensoren auslesen, Motoren steuern und Twitter-Nachrichten versenden kann.

Wie mit allem, kann man es jedoch auch hier übertreiben, sodass Systemressourcen an ihre Grenzen stoßen. Dadurch läuft Ihr Projekt eventuell zu langsam oder funktioniert womöglich gar nicht mehr. Deshalb ist es wichtig, die Systemauslastung überwachen zu können, um eventuell unwichtige Funktionen im System abzuschalten, bevor sie Ihr Projekt aus-

bremsen. Zunächst können Sie ganz oben rechts im Desktop die aktuelle CPU-Auslastung erkennen (siehe Abb. 6–14).

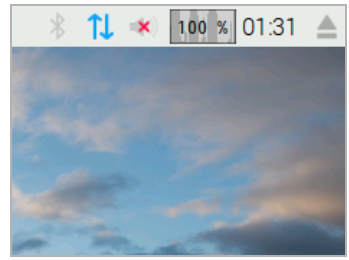


Abb. 6–14 Das Applet für die CPU-Auslastung im Desktop des Raspberry Pi

Leider zeigt dieses Applet nicht die Auslastung des Arbeitsspeichers oder der Datenströme an. Außerdem ist es nur sichtbar, wenn der Desktop läuft. Braucht man detaillierte Informationen zur Systemauslastung, hilft das Programm `top` (*table of processes*). Wie der Name schon verrät, listet es die aktuell laufenden Prozesse in tabellarischer Form auf und sortiert sie nach ihrer Ressourcenbeanspruchung. Standardmäßig wird die Darstellung der Prozesse alle drei Sekunden aktualisiert und dann nach deren CPU-Belastung absteigend sortiert. Da `top` jede Menge Informationen gleichzeitig anzeigt, schauen wir sie uns der Reihe nach an, damit Sie Ihr Projekt überwachen und ihm gegebenenfalls auf die Sprünge helfen können (siehe Abb. 6–15).

```
top - 19:38:30 up 3 days, 20:29,  2 users,  load average: 0.62, 0.20, 0.06
Tasks: 157 total,   1 running, 156 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.2 us,  0.6 sy,  0.0 ni, 98.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:    947740 total,   606868 used,   340872 free,    78044 buffers
KiB Swap:   102396 total,        0 used,   102396 free.   320672 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
  522 root      20   0   33412  17280  10408 S   2.6  1.8  51:47.17 vncserver-x11-c
19491 pi        20   0  416920 102676  51436 S   2.0 10.8   0:09.07 chromium-browse
  651 root      20   0  166780  52980  35516 S   1.0  5.6   3:20.74 Xorg
19523 pi        20   0    5112   2476   2092 R   1.0  0.3   0:00.22 top
   79 root     -51   0       0      0      0 S   0.3  0.0   1:42.23 irq/92-mmc1
   85 root      20   0       0      0      0 S   0.3  0.0   0:05.26 mmcqd/0
  673 root      20   0   10940   7204   6736 S   0.3  0.8   0:10.28 vncagent
 1104 pi        20   0   47220  19012  16060 S   0.3  2.0   0:03.57 lxterminal
19259 root      20   0       0      0      0 S   0.3  0.0   0:00.46 kworker/u8:1
    1 root      20   0   23892   3968   2740 S   0.0  0.4   0:11.32 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.12 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.92 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    7 root      20   0       0      0      0 S   0.0  0.0   0:54.26 rcu_sched
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S   0.0  0.0   0:00.10 migration/0
   10 root      rt   0       0      0      0 S   0.0  0.0   0:00.06 migration/1
```

Abb. 6–15 Ausgabe des Befehls top

Beginnen wir mit der obersten Zeile (siehe Abb. 6–16).

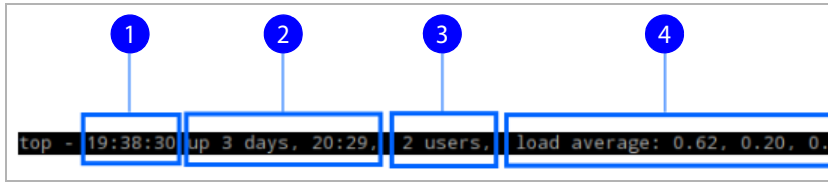


Abb. 6–16 Übersicht der Ausgabe nach dem Befehl top

1. Aktuelle Uhrzeit
2. Laufzeit seit letztem Neustart in Tagen, Stunden und Minuten
3. Anzahl angemeldeter Benutzer (bei laufendem Desktop normalerweise 2)
4. Durchschnittliche CPU-Auslastung der letzten Minute, 5 Minuten und 15 Minuten

Laufzeit

Die Informationen dieser Zeile können Sie sich auch mit dem Befehl `uptime` anzeigen lassen.

Schauen wir uns jetzt die Leiste `Tasks` an (siehe Abb. 6–17).

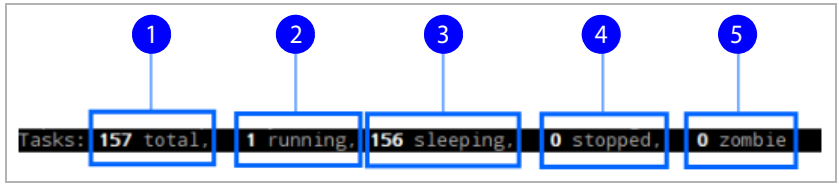


Abb. 6–17 Übersicht der Ausgabe nach dem Befehl `top` (Fortsetzung)

1. Gesamtzahl der Prozesse
2. Anzahl aktuell laufender Prozesse
3. Anzahl aktuell ruhender Prozesse
4. Anzahl der Prozesse, die angehalten wurden (dazu später mehr)
5. Anzahl eigentlich schon beendeter Prozesse, die aber noch auf die Beendigung anderer Prozesse warten

In Kapitel 2 haben wir bereits das Konzept von Eltern- und Kindprozessen angesprochen. Bei den *Zombie*-Prozessen handelt es sich um Kindprozesse, die ihre Aufgabe bereits erledigt haben, aber noch auf die Beendigung der Elternprozesse warten, bevor sie schließlich aus der Liste verschwinden. Schauen wir uns jetzt die Zeile `%CPU(s)` an (siehe Abb. 6–18).

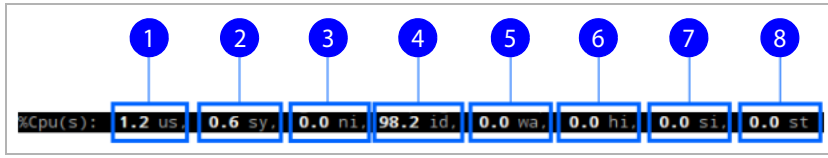


Abb. 6–18 Übersicht der Ausgabe nach dem Befehl top (Fortsetzung)

1. Prozentualer Zeitanteil, den die Gesamtheit der CPUs mit normalen Benutzerprozessen beschäftigt war.
2. Prozentualer Zeitanteil, den die CPUs mit System-Kernel-Prozessen beschäftigt waren.
3. Prozentualer Zeitanteil, den die CPUs mit priorisierten oder zurückgestuften Prozessen beschäftigt waren – in Linux auch gerne als *Niceness* bezeichnet (wird bei kleinen Systemen wie dem Raspberry Pi aber häufig nicht verwendet).
4. Prozentualer Zeitanteil, den die CPUs nichts getan haben, auch idle genannt.
5. Prozentualer Zeitanteil, den die CPUs gewartet haben, bis Ein-/Ausgaben beendet waren – ein Indikator für Arbeitsspeicherüberlastung oder langsamen Datenspeicher.
6. Prozentualer Zeitanteil, der mit der Behebung von Hardwareinterrupts (-unterbrechungen) verbracht wurde. Diese können vorkommen, wenn externe Geräte Informationen direkt zur CPU schicken müssen.
7. Prozentualer Zeitanteil, der mit der Behebung von Softwareinterrupts verbracht wurde – weniger wichtiger als die Hardwareinterrupts.
8. Dieser Wert gilt nur für virtuell laufende Systeme und steht für den prozentualen Zeitanteil, der dem System dadurch entging, dass das Hostsystem mit anderen Dingen beschäftigt war.

Falls Sie feststellen, dass Ihr Raspberry Pi zu langsam arbeitet, zeigt sich das normalerweise immer durch hohe Prozentzahlen der Werte Benutzer (*us*), System (*sy*) oder Warten (*wa*). Als Prozentwerte sollten alle Zahlen in der Reihe `%CPU(s)` natürlich zusammen 100 % ergeben. Gehen wir jetzt weiter zu den Zeilen `KiB Mem` und `KiB Swap` (siehe Abb. 6–19).

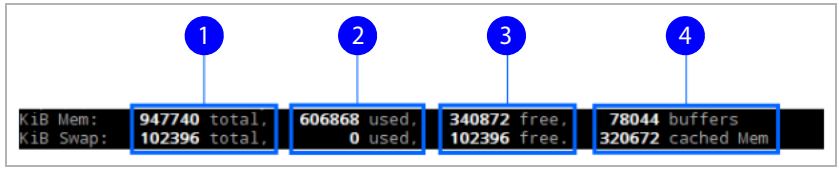


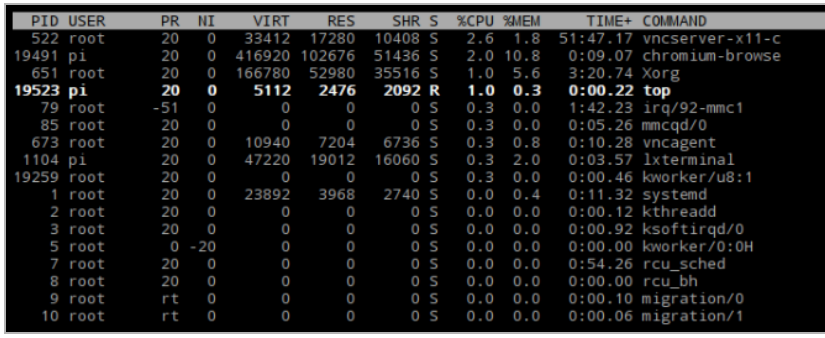
Abb. 6–19 Übersicht der Ausgabe nach dem Befehl top (Fortsetzung)

1. Gesamter verfügbarer Arbeitsspeicher inklusive Auslagerungsspeicher in Kilobytes (1024)
2. Umfang des verwendeten Arbeitsspeichers bzw. Auslagerungsspeichers
3. Menge des verfügbaren Arbeitsspeichers bzw. Auslagerungsspeichers
4. Umfang des Pufferspeichers und verwendeten Zwischenspeichers

Bei diesen beiden Zeilen geht es um unterschiedliche Formen von Arbeitsspeicher. Bei *Mem* handelt es sich um den echten Arbeitsspeicher, der in der Hardware fest verbaut ist. Bei meinem Raspberry Pi 3 sind das 1 GB RAM, weshalb sich die Größe des Gesamtspeichers in dem Bereich bewegen sollte. Der Auslagerungsspeicher *Swap* ist der virtuelle Speicher auf der Festplatte (in diesem Fall der SD-Karte), auf den zugegriffen wird, wenn der echte Arbeitsspeicher nicht ausreicht. Da die Zugriffsgeschwindigkeit auf die SD-Karte sehr viel geringer ist als die auf den Speicher auf der Platine, geht dies zulasten der Leistungsfähigkeit des Raspberry Pi.

Mit Zwischenspeicher (*Buffers*) ist der Speicher gemeint, den das System verwendet, wenn Dateisysteme aktiviert oder direkt angeschlossen sind. Um sich wiederholende Prozesse zu beschleunigen, werden darin einige Informationen über Dateien und Geräte zwischengespeichert. Die Daten werden gelesen, im Speicher abgelegt und bleiben dort so lange, wie schnell auf sie zugriffen werden muss. Später werden sie vom System automatisch gelöscht.

Worauf es hier im Wesentlichen ankommt ist der Umfang des verwendeten Speichers. Sollte bei Ihnen ständig der Speicher knapp werden, sollten Sie herausfinden, wodurch dies verursacht wird, und gegebenenfalls speicherhungrige Anwendungen schließen. Dabei können die Informationen aus dem letzten Abschnitt helfen (siehe Abb. 6–20).



PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
522	root	20	0	33412	17280	10408	S	2.6	1.8	51:47.17	vncserver-x11-c
19491	pi	20	0	416920	102676	51436	S	2.0	10.8	0:09.07	chromium-browse
651	root	20	0	166780	52980	35516	S	1.0	5.6	3:20.74	Xorg
19523	**pi**	**20**	**0**	**5112**	**2476**	**2092**	**R**	**1.0**	**0.3**	**0:00.22**	**top**
79	root	-51	0	0	0	0	S	0.3	0.0	1:42.23	irq/92-mmc1
85	root	20	0	0	0	0	S	0.3	0.0	0:05.26	mmcqd/0
673	root	20	0	10940	7204	6736	S	0.3	0.8	0:10.28	vncagent
1104	pi	20	0	47220	19012	16060	S	0.3	2.0	0:03.57	lxterminal
19259	root	20	0	0	0	0	S	0.3	0.0	0:00.46	kworker/u8:1
1	root	20	0	23892	3968	2740	S	0.0	0.4	0:11.32	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.12	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.92	ksoftirqd/0
5	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H
7	root	20	0	0	0	0	S	0.0	0.0	0:54.26	rcu_sched
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_bh
9	root	rt	0	0	0	0	S	0.0	0.0	0:00.10	migration/0
10	root	rt	0	0	0	0	S	0.0	0.0	0:00.06	migration/1

Abb. 6–20 Übersicht der Ausgabe nach dem Befehl top (Fortsetzung)

PID Die Prozess-ID

USER Der Besitzer (meist Starter) des Prozesses

PR Die aktuelle Priorität des Prozesses

NI Der *Nice*-Wert bzw. vom Benutzer zugewiesene Priorität

VIRT Der gesamte vom Prozess beanspruchte Speicherplatz

RES Der aktuell vom Prozess beanspruchte Speicherplatz

SHR Der dem Prozess zur Verfügung stehende gemeinsam genutzte Speicherplatz

S Der aktuelle Status des Prozesses, der durch folgende Buchstaben charakterisiert wird:

- D = ununterbrechbarer Schlaf
- R = läuft
- S = schläft
- T = auf Fehlersuche (traced) oder gestoppt
- Z = Zombie

%CPU
: Der prozentuale Anteil der Auslastung einer einzigen CPU, die von einem Prozess genutzt wird.

%MEM
: Der Anteil am gesamten Arbeitsspeicher, den dieser Prozess in Anspruch nimmt.

TIME Die Zeit in Hundertstelsekunden, die die CPUs mit diesem Prozess zugebracht haben.

COMMAND
: Der Befehl, der diesen Prozess ausgelöst hat. Das Pluszeichen deutet an, dass der Befehlsname zu lang ist, um in die Spalte zu passen.

Worauf es in diesem Abschnitt von `top` ankommt, sind die Prozesse, die viel CPU-Leistung und Arbeitsspeicher erfordern. Dafür sind die Spalten %CPU und %MEM relevant. Prozesse, die in dieser Hinsicht Probleme bereiten, erscheinen daher ganz schnell oben in der Liste. Aktuell laufende Prozesse werden dabei markiert. Die sich ständig aktualisierende Liste können Sie mit folgenden Groß- und Kleinbuchstaben beeinflussen:

`x` Markierung der aktuellen Spalte, nach der sortiert wird

`P` Sortierung nach %CPU (Voreinstellung)

`M` Sortierung nach %MEM

`N` Sortierung nach Prozess-ID

`T` Sortierung nach Laufzeit

< and >
: Umschalten zwischen den Spalten, nach denen sortiert wird

Pfeiltasten, Bild auf, Bild ab
: Nach links, rechts, oben und unten scrollen

`k` Abbrechen eines Prozesses (kill)

`h` Hilfe

`Q` Beenden

Es gibt noch jede Menge anderer Optionen in top. Sie finden sie auf der Hilfeseite und dem Handbucheintrag von `top`.

6.8 Probieren Sie es selbst

Finden Sie einmal heraus, wie viel Ressourcen eine bestimmte Anwendung benötigt, wenn sie startet und wenn sie in Betrieb ist. Am einfachsten geht das natürlich in der Desktop-Umgebung. Später werde ich Ihnen zeigen, wie man einen Prozess im Hintergrund ablaufen lässt, damit Sie dies auch über die Kommandozeile durchführen können. Öffnen Sie also das Terminal und starten Sie `top`:

```
top
```

Anschließend starten Sie eine andere Anwendung und schauen sich die Ausgabe von `top` an, um zu sehen, welche und wie viel Ressourcen in den ersten etwa 30 Sekunden benötigt werden. Als eine solche Anwendung eignet sich der Browser ganz gut. Beobachten Sie weiter, was passiert, wenn die Anwendung vollständig geladen ist und keine weiteren Aufgaben hat. Danach benutzen Sie die Anwendung (Öffnen einer Website, einer Datei etc.) und schauen, was mit der CPU-Auslastung und dem Speicherbedarf passiert.

6.9 Einen Prozess abbrechen: Ctrl-C, ps, kill

Alle Betriebssysteme haben Programme, die auf die eine oder andere Weise aus dem Ruder laufen können. Manchmal reagieren sie einfach nicht mehr oder man kann sie auf dem gewöhnlichen Weg nicht mehr beenden. Dies kann an Fehlern sowohl im Programm als auch im Betriebssystem liegen. Unabhängig davon, woran es letztlich liegt, sollten Sie wissen, wie man ein laufendes Programm abbricht, damit es nicht weiter Ressourcen blockiert oder den Computer lahmlegt. In Linux nennt man dieses erzwungene Abbrechen eines Programms oder Prozesses auch *Killing*. Bei vielen Programmen bleiben Abbruchvorgänge folgenlos, doch bei komplexeren Anwendungen können sie zu Programmfehlern führen, da ihnen möglicherweise die Gelegenheit genommen wird, Dateien oder Netzwerkverbindungen zu bereinigen. Hat Ihr Programm eine Funktion zur Beendigung (einen Button zum Schließen oder eine entsprechende Tastenfunktion), sollten Sie zunächst immer diese nutzen und den Programmabbruch immer nur als letzten Ausweg sehen.

Durch die Tastenkombination Ctrl-C senden Sie dem Programm, das gerade im Terminal läuft, ein Abbruchsignal. In der Regel wird das Programm dadurch sofort abgebrochen und man sieht anschließend wieder den Prompt. Bei einem Skript ist das der einfachste Weg, um anschließend weiterarbeiten zu können.

Bei Aufgaben oder Programmen, die automatisch oder durch einen anderen Benutzer gestartet wurden, muss man zunächst die PID des Prozesses ermitteln, den man abbrechen möchte. Dafür können Sie den Befehl `ps` (*process status*) verwenden. Ohne weitere Optionen zeigt Ihnen `ps` ausschließlich die vom aktuellen Benutzer gestarteten Prozesse. Für eine vollständige Liste aller laufenden Prozesse mit zusätzlichen Informationen fügen Sie die Option `-ef` an (siehe Abb. 6–21).

```
pi@raspberrypi:~ $ ps
  PID TTY          TIME CMD
 5309 pts/0    00:00:00 bash
 5319 pts/0    00:00:00 ps
pi@raspberrypi:~ $ ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 08:10 ?        00:00:02 /sbin/init splash
root         2     0  0 08:10 ?        00:00:00 [kthreadd]
root         3     2  0 08:10 ?        00:00:00 [ksoftirqd/0]
root         5     2  0 08:10 ?        00:00:00 [kworker/0:0H]
root         7     2  0 08:10 ?        00:00:03 [rcu_sched]
root         8     2  0 08:10 ?        00:00:00 [rcu_bh]
root         9     2  0 08:10 ?        00:00:00 [migration/0]
root        10     2  0 08:10 ?        00:00:00 [lru-add-drain]
root        11     2  0 08:10 ?        00:00:00 [cpuhp/0]
root        12     2  0 08:10 ?        00:00:00 [cpuhp/1]
root        13     2  0 08:10 ?        00:00:00 [migration/1]
root        14     2  0 08:10 ?        00:00:00 [ksoftirqd/1]
root        16     2  0 08:10 ?        00:00:00 [kworker/1:0H]
root        17     2  0 08:10 ?        00:00:00 [cpuhp/2]
root        18     2  0 08:10 ?        00:00:00 [migration/2]
root        19     2  0 08:10 ?        00:00:00 [ksoftirqd/2]
root        21     2  0 08:10 ?        00:00:00 [kworker/2:0H]
root        22     2  0 08:10 ?        00:00:00 [cpuhp/3]
```

Abb. 6–21 Die Ausgabe nach dem Befehl ps

Geben Sie den Befehl selbst einmal ein und Sie werden erkennen, dass dabei Hunderte von Prozessen aufgelistet werden können. Um darunter denjenigen zu finden, den Sie suchen, können Sie mithilfe von `grep` die Ausgabe eingrenzen (siehe Abb. 6–22). Wird der Befehl `grep` mit einem anderen Befehl kombiniert, werden nur die Zeilen ausgegeben, die einer bestimmten Zeichenfolge entsprechen (mehr über `grep` weiter hinten in diesem Kapitel):

```
ps -ef | grep gesuchteZeichenfolge
```

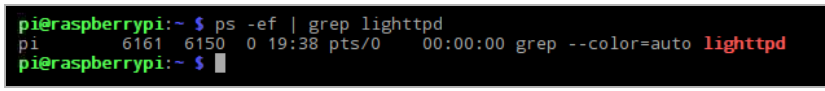


Abb. 6–22 Verwendung von grep mit dem Befehl `ps`

Wie Sie sehen, habe ich in diesem Fall nach `lighttpd` gesucht, einem Webserverprozess, der bei mir lief. Allerdings ist noch ein weiteres Suchergebnis angezeigt, und zwar das der Suche mit `grep` für `lighttpd` selbst. Auf dieses kommt es uns allerdings nicht an, sondern in diesem Falle nur auf PID 674.

Nehmen wir einmal an, dass sich mein Lighttpd-Webserver aus irgendeinem Grund aufgehängt hätte. Da es sich um einen Dienst handelt, hätte ich schon versucht, ihn mit dem richtigen Befehl anzuhalten (`sudo service lighttpd stop`), worauf er allerdings nicht reagiert hätte. Um also einen Prozess abzubrechen, kann man den Befehl `kill` benutzen. Der Befehl `kill` sendet ein Signal zum Abbruch an den jeweiligen Prozess, woraufhin dieser sofort stoppt. Je nachdem, wie man dieses Prozessende handhaben möchte, gibt es für `kill` diverse Optionen:

`kill PID`
: Normales Signal zum Beenden eines Prozesses

`kill -1 PID`
: Signal zum Neustart eines Prozesses

`kill -2 PID`
: Interrupt-Signal an einen Prozess; gleicher Effekt wie Ctrl-C

`kill -9 PID`
: Signal zum Abbrechen und sofortiger Beendigung des Prozesses

In diesem Fall möchte ich Lighttpd normal beenden. Da dieser Prozess nicht durch den aktuellen Benutzer gestartet wurde, muss ich dem `kill`-Befehl `sudo` voranstellen (siehe Abb. 6–23).

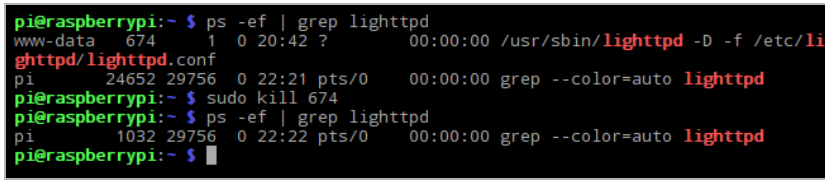


Abb. 6–23 Einsatz des Befehls `kill`

Nachdem Sie einem Prozess den Befehl `kill` gesandt haben, ist es ratsam, noch einmal `ps` laufen zu lassen, um sicherzugehen, dass der Prozess tatsächlich nicht mehr vorhanden ist. Reagiert ein Prozess weder auf `kill` allein noch auf die Option mit `-2`, können Sie als letzte Möglichkeit die Option `-9` wählen. Wie Sie aber in Abbildung 6–23 sehen, bekam ich bei der Nachkontrolle mit `grep` nach `kill` für den Prozess `lighttpd` nur den Suchbegriff selbst zurück. Der Prozess wurde also beendet.

6.10 Prozesse stoppen oder sie im Vorder- und Hintergrund ausführen: Ctrl-Z, &, fg

Manchmal kann es nützlich sein, einen Prozess anzuhalten, um dann etwas anderes zu machen und ihn anschließend weiterlaufen zu lassen. Es kann aber auch ganz nett sein, ein Programm gleich im Hintergrund zu starten, damit man es nicht die ganze Zeit beobachten muss. In Linux hat man zum einen Befehle zur Verfügung, mit denen man Prozesse anhalten kann, und zum anderen welche, mit denen man sie in den Hintergrund schickt, damit die Kommandozeile für weitere Eingaben frei bleibt.

Möchten Sie also ein Programm pausieren lassen, um es später weiterlaufen zu lassen, können Sie das Tastaturkürzel Ctrl-Z verwenden. Dadurch wird es unter Linux angehalten und in den Hintergrund geschickt und Sie sehen wieder den Prompt. Das im Hintergrund befindliche Programm wird jetzt keine weiteren Instruktionen verarbeiten, solange Sie es nicht zurück in den Vordergrund holen (siehe Abb. 6–24).

```
pi@raspberrypi:~ $ python loop.py
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
^Z
[1]+  Angehalten                python loop.py
pi@raspberrypi:~ $
```

Abb. 6–24 Mit Ctrl-Z einen Prozess anhalten

Wenn Sie das Programm weiterlaufen lassen wollen, können Sie dazu den Befehl `fg` verwenden. Dadurch kehrt es auf den Bildschirm zurück und macht da weiter, wo es aufgehört hat (siehe Abb. 6–25).

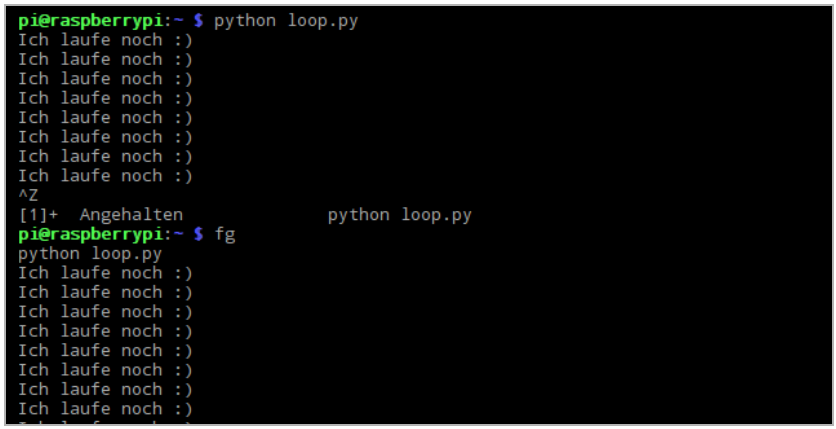


Abb. 6–25 Mit fg einen Prozess wieder in den Vordergrund holen

Möchten Sie ein Programm gleich im Hintergrund starten lassen, können Sie ihm das Zeichen `&` hinten anhängen. Dadurch wird das Programm gestartet und Ihnen gleich die PID angezeigt, falls Sie später noch einmal auf das Programm zurückgreifen müssen. Anschließend haben Sie wieder einen freien Prompt, um dort andere Aufgaben anzugehen. Gibt das im Hintergrund laufende Programm allerdings auf dem Bildschirm etwas aus, geschieht das auch weiterhin (siehe Abb. 6–26). Später zeige ich Ihnen noch, wie Sie diese Ausgabe auch woandershin verlagern können. Um das Programm in den Vordergrund zu holen, können Sie wieder den Befehl `fg` verwenden.

Abb. 6–26 Durch & ein Programm im Hintergrund laufen lassen

6.11 Probieren Sie es selbst

Erzeugen Sie ein Skript mit einer Schleife und üben Sie damit, es im Hintergrund laufen zu lassen und wieder in den Vordergrund zu holen. Finden Sie mithilfe von `ps` dessen PID heraus. Sobald Sie die PID kennen, können Sie das Programm mit dem Befehl `kill` beenden.

Legen Sie im Editor eine neue Datei namens *loop.py* an:

```
nano loop.py
```

Jetzt kopieren Sie folgenden Text in die leere Datei (achten Sie auf die Einhaltung der Einrückungen):

```
#!/usr/bin/python

import time

while True:

   print "Ich laufe noch :)"

   time.sleep(10)
```

Speichern Sie die Datei, indem Sie den Editor `nano` mit Ctrl-X beenden, dann J und Enter drücken. Lassen Sie das Programm mit folgendem Befehl laufen:

```
python loop.py
```

Das Skript gibt alle 10 Sekunden »`Ich laufe noch :)`« aus. Halten Sie es mit Ctrl-Z an. Jetzt sollten Sie wieder Ihren Prompt sehen, wo Sie weitere Befehle eingeben können. Holen Sie das Skript folgendermaßen wieder in den Vordergrund:

```
fg
```

Beenden Sie nun das Skript mit Ctrl-C. Starten Sie es danach wieder, diesmal allerdings im Hintergrund:

```
python loop.py &
```

Ermitteln Sie diesmal die PID des Skripts mithilfe von `ps` und `grep`:

```
ps -ef | grep loop
```

Beenden Sie den Prozess durch Angabe dessen PID.

```
kill PID
```

Dies sollte den Prozess beenden, damit er nicht mehr diese nervigen Mitteilungen ausgibt. :)

6.12 USB-Geräte finden: lsusb

Da der Raspberry Pi und die meisten anderen SBCs USB-Ports besitzen, kann man deren Funktionsumfang für seine Projekte beträchtlich erweitern. Tastatur, Maus, Audiogeräte, Bluetooth- und WLAN-Adapter, alles lässt sich via USB anschließen. Die meisten Linux-Distributionen unterstützen viele aktuelle und auch ältere USB-Geräte, ohne dass Sie irgendwelche Treiber dafür installieren müssten. Dank der harten Arbeit zahlloser Programmierer, die zum Linux-Kernel beigetragen haben, sind sie dort bereits enthalten.

Dies führt allerdings dazu, dass Sie keine hübsche Nachricht bekommen, dass Ihr Gerät vom System erkannt wurde, wie Sie das vielleicht von Windows kennen. Um eine vollständige Liste der von Ihrem System zurzeit erkannten USB-Geräte zu erhalten, können Sie den Befehl `lsusb` verwenden. Ähnlich wie bei `ls` werden Ihnen die USB-Geräte dann zusammen mit deren hexadezimalen Identifikationsnummern angezeigt. Generell ist es gut, diesen Befehl einzugeben, bevor Sie Ihr USB-Gerät einstecken. Wenn Sie nach dem Einstecken noch einmal `lsusb` eingeben, finden Sie das Gerät anschließend leichter in der Auflistung (siehe Abb. 6–27).

```
pi@raspberrypi:~ $ lsusb
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast
Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
pi@raspberrypi:~ $ lsusb
Bus 001 Device 004: ID 0d8c:013c C-Media Electronics, Inc. CM108 Audio Controlle
r
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast
Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
pi@raspberrypi:~ $
```

Abb. 6–27 Ausgabe nach dem Befehl lsusb

Wie Sie sehen, wurden bei mir schon vor dem Anschluss eines externen USB-Geräts durch `lsusb` USB-Geräte aufgelistet. Das liegt daran, dass einige Komponenten, wie der Ethernetadapter, intern mit dem USB-Bus verschaltet sind. Sobald ich mein USB-Mikrofon eingesteckt hatte und wieder `lsusb` eingab, wurde es mir als »C-Media Electronics, Inc. CM108 Audio Controller« angezeigt. Diese Art von Information kann Ihnen helfen, im Internet mehr über dieses Gerät zu erfahren. Es kann auch sein, dass von Ihnen geschriebene Programme die hexadezimale ID des Geräts benötigen, damit sie ordnungsgemäß laufen.

Wichtig zu beachten

Jedes Gerät, das Sie an den USB-Port Ihres Raspberry Pi anschließen, bedeutet zusätzlichen Strombedarf. Übersteigt dieser das verfügbare Maß, wird dies zum Absturz Ihres Systems führen, vor allem beim Hochfahren. Sorgen Sie also dafür, dass Ihr Netzgerät ausreichend Strom liefert, um neben dem Raspberry Pi noch die angeschlossenen Geräte ausreichend zu versorgen.

6.13 Ausgabe eines Skripts protokollieren: >, >>

Es gibt viele Gründe, die Ausgabe eines Skripts aufzeichnen zu wollen. Sammelt Ihr Projekt beispielsweise über längere Zeiträume hinweg Daten, möchten Sie diese vielleicht protokollieren, um sie später analysieren zu können. Es kann auch sein, dass zwischendurch Fehlermeldungen angezeigt werden, Sie aber keine Zeit haben, diese direkt zu lesen, bevor sie aus dem Blickfeld verschwinden. Wenn Sie gar mit *rc.local* ein Skript beim Hochfahren starten, läuft es als »root«, sodass Sie dessen Ausgabe beim Anmelden erst gar nicht zu Gesicht bekommen.

Für solche Fälle gibt es die Möglichkeit, die Ausgabe eines Skripts in eine Protokolldatei (Logdatei) schreiben zu lassen, damit Sie sich später deren Inhalt anschauen können. In Linux wird dazu einfach der Befehl, der das Skript startet, mit einem Größer-als-Zeichen versehen. Dadurch wird die Ausgabe umgeleitet.

`>` Leite die Ausgabe in eine neue Datei.

`>>` Füge die Ausgabe in eine vorhandene Datei ein.

`&>`, `&>>`
: Erzeuge bzw. ergänze eine Datei einschließlich der Fehlermeldungen.

Möchten Sie zum Beispiel die Ausgabe des Skripts *loop.py*, das wir vorhin erstellt haben, in eine neue Datei namens *loop.log* schreiben, können Sie das mit folgendem Befehl veranlassen:

```
python loop.py > loop.log
```

Während das Skript läuft, wird die normale Ausgabe (wie Print-Statements) in die Datei *loop.log* geschrieben. Nachdem das Skript ausgeführt wurde, können Sie den Inhalt dieser Protokolldatei in einem Texteditor oder einfach mithilfe des Befehls `more` einsehen (siehe Abb. 6–28).

```
pi@raspberrypi:~ $ python loop.py > loop.log
^CTraceback (most recent call last):
  File "loop.py", line 5, in <module>
    time.sleep(10)
KeyboardInterrupt
pi@raspberrypi:~ $ more loop.log
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
pi@raspberrypi:~ $
```

Abb. 6–28 Mit > die Ausgabe in eine neue Datei schreiben

Wenn Sie die weitere Ausgabe derselben Protokolldatei zuführen möchten, ohne dass sie überschrieben wird, nutzen Sie einfach zwei Größer-als-Zeichen (siehe Abb. 6–29):

```
python loop2.py >> loop.log
```

```
pi@raspberrypi:~ $ python loop2.py >> loop.log
^CTraceback (most recent call last):
  File "loop2.py", line 5, in <module>
    time.sleep(10)
KeyboardInterrupt
pi@raspberrypi:~ $ more loop.log
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
pi@raspberrypi:~ $
```

Abb. 6–29 Mit >> die Ausgabe dem Inhalt einer vorhandenen Datei hinzufügen

Wie Sie sehen, wurde die Ausgabe des zweiten Skripts der des ersten Skripts hinzugefügt. Die Traceback-Fehlermeldung wurde allerdings nicht mitprotokolliert. Sollen solche Fehlermeldung zusätzlich zur normalen Ausgabe aufgezeichnet werden, verwendet man `&>`, um den Inhalt der Protokolldatei zu überschreiben, oder `&>>`, um die komplette Ausgabe einer vorhandenen Datei hinzuzufügen (siehe Abb. 6–30).

```
python loop.py &>> loop.log
```

```
pi@raspberrypi:~ $ python loop.py &>> loop.log
^Cpi@raspberrypi:~ $ more loop.log
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Traceback (most recent call last):
  File "loop.py", line 5, in <module>
    time.sleep(10)
KeyboardInterrupt
pi@raspberrypi:~ $
```

Abb. 6–30 Mit &>> die Ausgabe inklusive der Fehlermeldungen dem Inhalt einer vorhandenen Datei hinzufügen

6.14 In der Ausgabe eines Befehls suchen: grep

Wie Sie bereits wissen, kann man mit dem Befehl `grep` in der Ausgabe des Befehls `ps` nach bestimmten Zeichenfolgen suchen. Man kann `grep` allerdings auch in der Ausgabe fast jedes anderen Befehls suchen lassen. Die Ursprünge von `grep` sind etwas geheimnisvoller als die anderer Linuxbefehle (`grep` bedeutet *globally search a regular expression and print*; suche global nach einem Begriff und gib ihn aus). Um `grep` zu verwenden, gibt man zunächst den Befehl ein, der die Ausgabe erzeugt, gibt dann einen senkrechten Strich | (genannt »Pipe«) ein, gefolgt von `grep`, und stellt dann die gesuchte Zeichenfolge, den Suchbegriff, dahinter. Haben Sie also beispielsweise eine Protokolldatei erzeugt, können Sie deren Inhalt mit `more` anzeigen lassen und mit `grep` kombinieren, um nur diejenigen Zeilen ausgeben zu lassen, die Ihren Suchbegriff enthalten (siehe Abb. 6–31).

```
more loop.log | grep wieder
```

```
pi@raspberrypi:~ $ more loop.log | grep wieder
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
pi@raspberrypi:~ $
```

Abb. 6–31 Mit grep sich nur die Zeilen ausgeben lassen, in denen »wieder« enthalten ist

Für `grep` gibt es viele nützliche Optionen. Meine Favoriten sind u.a.:

`-e` Gleichzeitige Suche nach mehreren Zeichenfolgen

`-i` Suche nach Groß- und Kleinbuchstaben gleichermaßen

`-c` Anzahl der Zeilen, in denen die Zeichenfolge vorkommt

Kombiniert man diese Optionen geschickt miteinander, wird `grep` zu einem sehr leistungsfähigen Werkzeug (siehe Abb. 6–32). Im entsprechenden Handbucheintrag finden sich viele weitere hilfreiche Optionen.

```
pi@raspberrypi:~ $ more loop.log | grep -e "l" -ie "i"
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe schon wieder :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Ich laufe noch :)
Traceback (most recent call last):
  File "loop.py", line 5, in <module>
    time.sleep(10)
KeyboardInterrupt
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
pi@raspberrypi:~ $ more loop.log | grep -ce "l" -ice "i"
34
pi@raspberrypi:~ $
```

Abb. 6–32 Suche nach mehreren Zeichen(folgen) gleichzeitig mit grep

In Abbildung 6–32 habe ich `grep` alle Zeilen einer Protokolldatei durchsuchen und ausgeben lassen, die entweder ein kleingeschriebenes *a* ODER ein groß- oder auch kleingeschriebenes *i* enthielten. Die gleiche Suche habe ich danach wiederholt, doch statt mir die entsprechenden Zeilen ausgeben zu lassen, habe ich mir deren Anzahl anzeigen lassen. Kombiniert man mehrere `grep`-Anweisungen miteinander, erreicht man eine UND-Verknüpfung, d.h., in den ausgegebenen Zeilen müssen alle Suchbegriffe vorkommen (siehe Abb. 6–33).

```
pi@raspberrypi:~ $ more loop.log | grep Hund | grep wieder
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
pi@raspberrypi:~ $
```

Abb. 6–33 Suche nach mehreren Zeichen(folgen) gleichzeitig mit `grep` (fortgesetzt)

6.15 Protokolldatei überwachen: tail

Manchmal kann es hilfreich sein, sich einfach nur die letzten Zeilen einer Protokolldatei anzusehen, um beispielsweise nachzuschauen, was kurz vor einem Programmabsturz oder einem Fehler passiert ist. Wie der Name vermuten lässt, zeigt `tail` ohne weitere Optionen die letzten 10 Zeilen einer Datei an (siehe Abb. 6–34).

```
pi@raspberrypi:~ $ tail loop.log
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
Der Hund laeuft wieder :)
pi@raspberrypi:~ $
```

Abb. 6–34 Mit tail die letzten paar Zeilen einer Datei ansehen

Für `tail` gibt es zwei besonders nützliche Optionen. Die eine ist `-n` *`Anzahl`*, mit der man sich jede beliebige Anzahl von Zeilen ausgeben lassen kann statt der üblichen zehn. Die andere ist `-f`, die zwar wieder die letzten 10 Zeilen einer Datei ausgibt, aber ständig die neu entstehenden hinzu-

fügt, während sie gerade in die Datei geschrieben werden. Auf diese Weise kann man die Entstehung der Protokolldatei in Echtzeit verfolgen.

6.16 Benutzer hinzufügen: adduser, addgroup

Irgendwann kann es vorkommen, dass Sie bei Ihrem Raspberry Pi einen neuen Benutzer anlegen wollen. Sei es, weil Sie jemand anderem Zugang zum System verschaffen wollen, ohne ihm das Passwort für den Benutzer »pi« und dessen Dateien und Einstellungen geben zu wollen, oder weil Sie Ihrem Projekt auf dem Computer einfach eine eigene Identität geben wollen. Da Linux von Grund auf als Mehrbenutzersystem ausgelegt wurde, ist das Anlegen eines neuen Benutzers unkompliziert. Mit `sudo` vorangestellt starten Sie dazu einfach das Hilfsprogramm `adduser`:

```
sudo adduser Benutzername
```

Anschließend folgen Sie den Anweisungen (siehe Abb. 6–35).

```
pi@raspberrypi:~ $ sudo adduser projekt
Lege Benutzer »projekt« an ...
Lege neue Gruppe »projekt« (1001) an ...
Lege neuen Benutzer »projekt« (1001) mit Gruppe »projekt« an ...
Erstelle Home-Verzeichnis »/home/projekt« ...
Kopiere Dateien aus »/etc/skel« ...
Geben Sie ein neues UNIX-Passwort ein:
Geben Sie das neue UNIX-Passwort erneut ein:
passwd: Passwort erfolgreich geändert
Benutzerinformationen für projekt werden geändert.
Geben Sie einen neuen Wert an oder drücken Sie ENTER für den Standardwert
        Vollständiger Name []:
        Zimmernummer []:
        Telefon geschäftlich []:
        Telefon privat []:
        Sonstiges []:
Sind die Informationen korrekt? [J/n] J
pi@raspberrypi:~ $
```

Abb. 6–35 Mit adduser in Linux einen neuen Benutzer anlegen

Ist der Vorgang abgeschlossen, haben Sie ein neues Benutzerkonto mit eigenem Homeverzeichnis und einer entsprechenden Benutzergruppe gleichen Namens. Möchten Sie Ihrem System lediglich eine neue Gruppe hinzufügen, geht das entsprechend mit dem Befehl `addgroup` (siehe Abb. 6–36).

```
pi@raspberrypi:~ $ sudo addgroup admins
Lege Gruppe »admins« (GID 1002) an ...
Fertig.
pi@raspberrypi:~ $
```

Abb. 6–36 Mit addgroup eine neue Gruppe in Linux anlegen

Mit dem Befehl `adduser` können Sie dieser neuen Gruppe neue Mitglieder zuführen. Dazu lassen Sie `adduser` den bereits existierenden Benutzernamen und den Gruppennamen folgen (siehe Abb. 6–37).

```
sudo adduser Benutzername Gruppenname
```

```
pi@raspberrypi:~ $ sudo addgroup admins
Lege Gruppe »admins« (GID 1002) an ...
Fertig.
pi@raspberrypi:~ $ sudo adduser pi admins
Füge Benutzer »pi« der Gruppe »admins« hinzu ...
Benutzer pi wird zur Gruppe admins hinzugefügt.
Fertig.
pi@raspberrypi:~ $ sudo adduser projekt admins
Füge Benutzer »projekt« der Gruppe »admins« hinzu ...
Benutzer projekt wird zur Gruppe admins hinzugefügt.
Fertig.
pi@raspberrypi:~ $
```

Abb. 6–37 Mit adduser einen vorhandenen Benutzer einer Gruppe hinzufügen

6.17 Besitzer und Rechte von Dateien ändern: chown, chmod

In Kapitel 2 habe ich erklärt, wie Rechte in Linux funktionieren. Lassen Sie uns jetzt schauen, wie man sie ändern kann. So etwas kann nötig werden, weil man zum Beispiel die Meldung »keine Berechtigung« bekommt, wenn man Skripte oder Befehle ausführen möchte, die auf Dateien anderer Benutzer zugreifen müssen. Es gibt auch Programme, die aus Sicherheitsgründen als separater Benutzer laufen möchten, sodass man die Besitzrechte von Dateien ändern muss, damit das Programm mit ihnen arbeiten kann. Auch Ihren eigenen Programmen oder Skripten müssen Sie Ausführungsrechte geben, damit sie ausgeführt werden können.

Warnung

Die Besitzrechte von Dateien zu ändern, auf die normalerweise nur der Root-Benutzer Zugriff hat, kann die Sicherheit Ihres gesamten Systems gefährden und es instabil werden lassen. Stattdessen sollten Sie die entsprechenden Befehle mit `sudo` ausführen.

Um das Besitzrecht einer Datei (für einen Benutzer und/oder Gruppe) zu ändern, verwenden Sie den Befehl `chown`. Wenn Sie nicht schon Schreibrechte für diese Datei haben, brauchen Sie `sudo`, um das Besitzrecht zu

ändern. Benutzer- und Gruppenbesitzrecht können gleichzeitig auf diese Weise geändert werden:

```
sudo chown Benutzer:Gruppe Dateiname
```

Möchten Sie lediglich das Besitzrecht auf Benutzerebene ändern, lassen Sie Doppelpunkt und Gruppennamen weg. Manchmal ist es auch ganz praktisch, wenn man die Besitzrechte aller Dateien eines Verzeichnisses gleichzeitig ändern kann. Dies geschieht mit Angabe der Option `-R` vor dem Benutzernamen:

```
sudo chown -R Benutzer Verzeichnis
```

Denken Sie daran, dass unter Linux für jeden Benutzer automatisch eine Gruppe gleichen Namens angelegt wird. Das mag anfangs etwas verwirren, ist aber ganz hilfreich, wenn man mehreren Benutzern gleichzeitig Besitzrechte zuteilen muss (siehe Abb. 6–38).

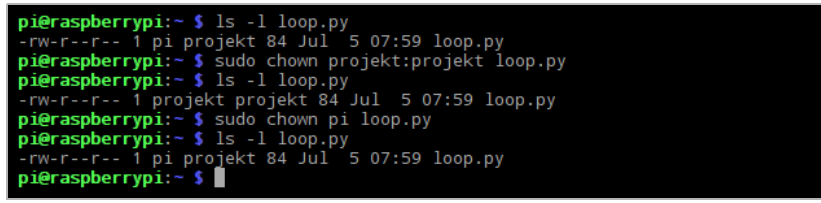


Abb. 6–38 Beispiele für die Verwendung des Befehls chown

Wie Sie in Abbildung 6–38 sehen, habe ich zunächst Benutzer- und Gruppenbesitzrecht der Datei *loop.py* für den Benutzer »projekt« und die Gruppe »projekt« geändert. Dann fiel mir ein, dass ich eigentlich noch wollte, dass die Besitzrechte der Datei beim Benutzer »pi« verbleiben, weshalb ich das Besitzrecht auf Benutzerebene auf »pi« übertrug. Nun können sowohl »pi« als auch »projekt« die Datei *loop.py* lesen und ausführen, doch nur »pi« kann sie überschreiben.

Wenn Sie eine neue Datei anlegen, wird sie mit den Rechten 644 bzw. `rw-r-- r--` ausgestattet. Dies bedeutet, dass der Besitzer die Datei lesen und verändern kann, jeder andere darf sie nur lesen. Gleichzeitig heißt das aber, dass ohne weiteres Zutun diese Datei niemand ausführen darf. Handelt es sich dabei aber beispielsweise um ein Skript, das im Rahmen Ihres Projektes ausgeführt werden soll, ist das ein Problem. Dieses Problem lösen Sie durch Änderung der Rechte mit `chmod` (*change file mode*). Wie bei `chown` brauchen Sie `sudo`, um die Dateirechte zu ändern.

Mit `chmod` können Sie die Rechte auf zweierlei Arten ändern. Die eine besteht in der numerischen Darstellung der Rechte, die Sie zuweisen möchten:

```
sudo chmod XXX Dateiname
```

XXX steht für die Zahlenangabe der Rechte (z.B. 644). Durch `+` und `-`, gefolgt von den Buchstaben `x`, `r` und/oder `w`, können Sie auf allen Ebenen der Besitzerschaft gleichzeitig Rechte hinzufügen oder wegnehmen. Um also beispielsweise allen Benutzern Ausführungsrechte für eine Datei zuzuteilen, können Sie Folgendes eingeben:

```
sudo chmod +x Dateiname
```

```
pi@raspberrypi:~ $ touch programm.py
pi@raspberrypi:~ $ ls -l programm.py
-rw-r--r-- 1 pi pi 0 Jul  6 09:17 programm.py
pi@raspberrypi:~ $ chmod +x programm.py
pi@raspberrypi:~ $ ls -l programm.py
-rwxr-xr-x 1 pi pi 0 Jul  6 09:17 programm.py
pi@raspberrypi:~ $ chmod 744 programm.py
pi@raspberrypi:~ $ ls -l programm.py
-rwxr--r-- 1 pi pi 0 Jul  6 09:17 programm.py
pi@raspberrypi:~ $
```

Abb. 6–39 Dateirechte mit `chmod` ändern

Wie bei `chown` auch, kann man mit der Option `-R` die Rechte ganzer Verzeichnisse ändern. Seien Sie dabei jedoch vorsichtig, da die Zuteilung der falschen Rechte an einer Datei leicht größere Sicherheitsprobleme für das ganze System nach sich ziehen kann.

6.18 Probieren Sie es selbst

Legen Sie eine neue Datei an und üben Sie das Ändern deren Besitzerschaft und Rechte.

Legen Sie mit `touch` eine neue Datei an:

```
touch programm.py
```

Lassen Sie sich mit `ls` die Rechte und Besitzerschaft anzeigen:

```
ls -l programm.py
```

Geben Sie der Datei den Gruppenbesitz »root«:

```
sudo chown pi:root programm.py
```

Geben Sie jetzt allen Benutzern Ausführungsrechte:

```
sudo chmod +x programm.py
```

Überprüfen Sie mit `ls` Ihre Änderungen:

```
ls -l programm.py
```

6.19 Mehrere Befehle gleichzeitig ausführen: &&, ||

Wenn ein Programm längere Zeit in der Kommandozeile läuft, kann man Däumchen drehen, während man auf den Bildschirm starrt und wartet, bis ein Programm oder Befehl abgearbeitet ist und man endlich weitermachen kann, abhängig davon, ob der Vorgang erfolgreich war oder nicht. In solchen Fällen kann es sehr hilfreich sein, zwei Befehle gleichzeitig ausführen zu können, damit man Zeit hat, zwischendurch einen Kaffee zu holen oder an seinem Buch weiter zu schreiben. Die Linux-Shell bietet dazu zwei Operatoren, die einem dies ermöglichen. Der erste wird als `&&` dargestellt und bedeutet in der Logik ein UND. Der zweite, `||`, bedeutet in der Logik ein ODER.

In der Kommandozeile funktioniert das folgendermaßen: Wenn man zwei durch `&&` getrennte Befehle eingibt, führt die Shell zuerst den linken aus und prüft, ob er erfolgreich ausgeführt wurde oder nicht. Falls ja, führt die Shell den Befehl auf der rechten Seite aus. Falls nein, wird der Befehl rechts nicht ausgeführt. Trennt man die Befehle mit `||`, passiert das Gegenteil: Der Befehl auf der rechten Seite wird nur ausgeführt, falls der linke aus irgendeinem Grund nicht erfolgreich durchgelaufen ist. Auf diese Weise lassen sie sich auch zu einer Aktion verknüpfen, deren Schritte vom Ergebnis des zuvor ausgeführten Befehls abhängen (siehe Abb. 6–40).

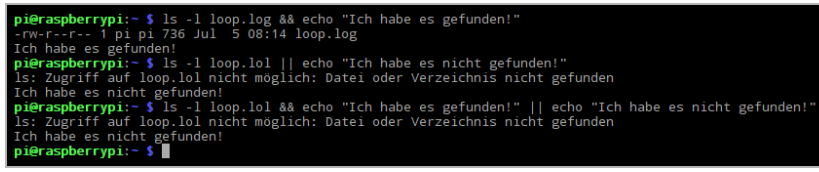


Abb. 6–40 Mit && mehrere Befehle nacheinander durchlaufen lassen

Diese Operatoren begegnen Ihnen in Skripts beim Hochfahren und in Shell-Skripts häufiger, deshalb ist es gut zu wissen, was sie tun, auch

wenn Sie selbst sie nicht oft nutzen. Ein einfaches Anwendungsbeispiel für den typischen Maker ist jedoch das Softwareupdate auf dem Raspberry Pi. Wie Sie sich aus Kapitel 4 bestimmt noch wissen, empfiehlt es sich, dazu stets zwei Befehle nacheinander einzugeben (`sudo apt-get update` und `sudo apt-get upgrade`). Mithilfe dieser Operatoren können wir diese Befehle jetzt einfach verknüpfen und etwas Zeit sparen:

```
sudo apt-get update && sudo apt-get upgrade
```

Möchten Sie einfach zwei oder mehr Befehle nacheinander durchlaufen lassen, ohne sich um das Ergebnis der einzelnen Befehle zu kümmern, können Sie sie auch mit einem Semikolon (`;`) voneinander trennen. Dies wird allerdings nicht empfohlen, da man sich dadurch jede Menge Probleme einhandeln kann. Beim Scheitern eines Befehls ist es in der Regel besser, anzuhalten und herauszufinden, woran es gelegen hat.

6.20 Eine weitere Terminalsitzung öffnen

Ganz gleich, ob Sie die Kommandozeile mit oder ohne Desktop verwenden, kann es vorkommen, dass sich ein Programm danebenbenimmt und einfriert, sodass Sie in der aktuellen Terminalsitzung weder Tastatur noch Maus bedienen können. In solch einer Lage weiß man oft nicht, ob das gesamte System oder nur die jeweilige Terminalsitzung abgestürzt ist. Statt nun einfach das Stromkabel zu ziehen (was wichtige Systemdateien beschädigen kann), können Sie mit einem Tastaturkürzel in eine andere Sitzung wechseln und das Problem von dort aus versuchen zu beheben.

Beim Hochfahren der meisten Linux-Distributionen werden nämlich mehrere virtuelle Terminalsitzungen im Hintergrund gestartet. Unter Linux heißen diese TTY1, TTY2 usw. Um nun eine andere Terminalsitzung anzuzeigen, drücken Sie gleichzeitig die Ctrl-, Alt- und eine der Funktionstasten von 1 bis 7. Die entsprechenden Terminalsitzungen unter Linux sind von TTY1 bis TTY7 durchnummeriert. Lassen Sie Ihr Terminal vom Desktop aus laufen, läuft es als TTY7, ohne Desktop als TTY1, sodass TTY2 bis TTY6 für zusätzliche Terminalsitzungen zur Verfügung stehen.

Arbeiten Sie also gerade in der Desktop-Umgebung im Terminal und es friert ein, drücken Sie Ctrl-Alt-F1, um zur Sitzung TTY1 zu wechseln. Um wieder in die ursprüngliche Sitzung zu wechseln, drücken Sie Ctrl-

Alt-F7. Booten Sie aus der Kommandozeile heraus, befinden Sie sich bereits in TTY1 und können durch Ctrl-Alt-F2 in eine neue Terminalsitzung wechseln.

Nur auf direktem Weg

Damit dies alles funktioniert, müssen Sie direkt mit Ihrem System verbunden sein. Greifen Sie via SSH oder VNC auf Ihr System zu, funktionieren diese Tastaturkürzel nicht. Sie funktionieren allerdings auch dann nicht mehr, wenn Ihr System völlig abgestürzt ist und auf keinerlei Tastatureingaben mehr reagiert.

6.21 Umgang mit langen Befehlen

Wie Sie mittlerweile mitbekommen haben, gibt es für manche Befehle und Programme jede Menge Optionen. Diese sind mitunter nicht nur sehr leistungsfähig, sondern können auch außerordentlich lang werden. Laufen Sie aus dem Terminalfenster, weiß man manchmal nicht, ob man sich eventuell verschrieben hat.

Doch in der Linux-Shell gibt es eine Möglichkeit, mit diesem Problem besser zurechtzukommen. Mit dem Maskierungszeichen (\, Escape, Backslash) und anschließendem Drücken der Entertaste kann man einen Zeilenwechsel innerhalb einer Programmzeile erzwingen, damit die Zeichen nicht aus dem Fenster laufen und alles besser lesbar bleibt. Man kann das Maskierungszeichen an jeder beliebigen Stelle setzen und so die Zeile trennen. Danach gibt man sein Kommando weiter ein, bis es vollständig ist. Solche Zeilenwechsel lassen sich innerhalb einer Programmzeile beliebig oft setzen, sodass man alles schön übersichtlich gestalten kann (siehe Abb. 6–41).

```
pi@raspberrypi:~ $ echo "Ein Gedicht würde man ohne Zeilentrennung nicht als sol
ches erkennen."
Ein Gedicht würde man ohne Zeilentrennung nicht als solches erkennen.
pi@raspberrypi:~ $ echo "Ein Gedicht würde man \
> ohne Zeilentrennung \
> nicht als solches \
> erkennen."
Ein Gedicht würde man ohne Zeilentrennung nicht als solches erkennen.
pi@raspberrypi:~ $
```

Abb. 6–41 Mit dem Zeichen \ einen Zeilenumbruch herbeiführen

Das Maskierungzeichen wurde eingeführt, damit man der Shell sagen kann, dass sie die normale Interpretation der nachfolgenden Zeichen maskieren soll, damit sie entweder anders oder direkt als Schriftzeichen interpretiert werden. Dazu ein Beispiel: Wollte man Anführungszeichen oben in einer Textausgabe wie bei `echo` verwenden, ginge das nicht direkt, weil damit der auszugebende Text eingeschlossen würde. Möchte man trotzdem, dass Anführungszeichen oben in der Ausgabe erscheinen, muss man diese mit dem Maskierungszeichen vor der Interpretation der Shell »verstecken« (siehe Abb. 6–42).

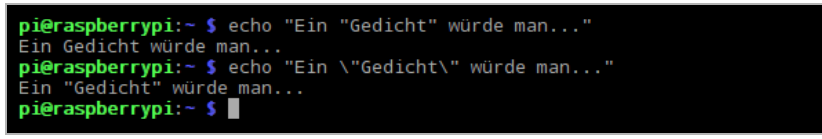


Abb. 6–42 Verwendung von \ als Maskierungszeichen

Im Internet sind sehr viele hervorragende Beispiele für die Verwendung des Maskierungszeichens zu finden. Denken Sie aber daran, dass je nach Programmierumgebung sich das Maskierungszeichen anders verhalten kann. Was also in der Linux-Shell funktioniert, kann in einer Programmiersprache wie Python oder Java ganz anders sein.

6.22 Nach Zeitplan arbeiten: cron

Skripte aus der Kommandozeile zu starten ist zwar gut und schön, doch bei einigen Projekten kann es nötig sein, ein Skript beispielsweise in festen Zeitinvallen zu starten. Typische Beispiele dafür sind regelmäßige Backup-Routinen oder Skripte, die alle zehn Minuten einen Temperatursensor auslesen. Für alle zeitbasierten Skripte kann man das Linux-Hilfsprogramm `cron` einsetzen.

Das vom griechischen Wort für Zeit, *Chronos,* abgeleitete cron stammt noch aus der Frühzeit von Unix. In Linux läuft `cron` stets im Hintergrund mit und schaut, ob es Zeit ist, irgendein Skript oder einen Befehl auszuführen. Jeder Benutzer kann `cron` individuell in der speziellen Datei *crontab* konfigurieren. Diese Datei lässt sich nicht direkt in einem normalen Texteditor öffnen. Stattdessen ruft der Benutzer *crontab* folgendermaßen auf:

```
crontab -e
```

Geschieht dies zum ersten Mal, fragt das System, welchen Editor man verwenden möchte (siehe Abb. 6–43).

```
pi@raspberrypi:~ $ crontab -e
no crontab for pi - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/ed
  2. /bin/nano        <---- easiest
  3. /usr/bin/vim.tiny

Choose 1-3 [2]: 2
```

Abb. 6–43 Entscheidung für einen Editor für `crontab`

Für die Erläuterungen zu *crontab* an dieser Stelle habe ich mich für `nano` entschieden. Sobald Sie einen Editor ausgewählt haben, wird Ihre Datei *crontab* geladen. In der noch nicht bearbeiteten Datei befindet sich etwas auskommentierter Text und ein Beispiel. Trotzdem mag dies zunächst verwirrend erscheinen. Dabei besteht jede zeitgesteuerte Aufgabe, jeder »Cronjob«, im Grunde aus einer einzigen Zeile: Minute, Stunde, Tag, Monat und Wochentag der Ausführung des Skripts oder Befehls, der ganz rechts steht. Sobald man verstanden hat, wie man einen Cronjob konfiguriert, hat man ein ziemlich flexibles System an der Hand. Schauen wir uns dazu in Abbildung 6–44 noch einmal gemeinsam an, wie diese Zeile aufgebaut ist.

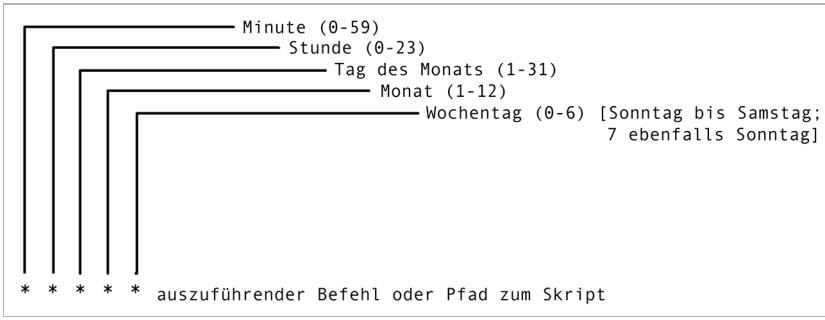


Abb. 6–44 Aufbau des Zeilenformats in der Datei `crontab`

Möchte ich also das Skript *hallo.sh* jeden Sonntag im Januar um genau 23:30 Uhr ausführen lassen, würde ich folgende Zeile in meine Datei *crontab* einfügen (siehe Abb. 6–45):

```
30 23 * 1 0 /home/pi/hallo.sh
```

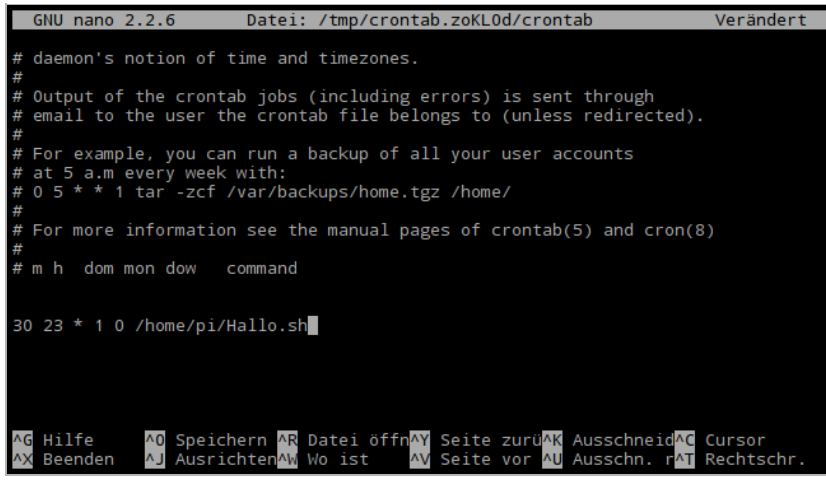
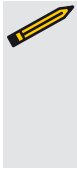


Abb. 6–45 Mit `cron` eine Aufgabe zeitlich planen

Beachten Sie, dass ich an der Position des Tags des Monats ein Sternchen (Asterisk) gesetzt habe. Das Sternchen steht für »beliebig«, d.h., der Tag des Monats war mir bei diesem Cronjob egal. Hätte ich die Ausführung meines Skripts auf Sonntage beschränken wollen, die auf den 5. eines jeden Monats fallen, hätte ich statt des Sternchens eine 5 eingetragen. Man kann eine bestimmte Position mit Schrägstrichen (/) auch in mehrere Zeitschritte unterteilen oder – durch Kommata getrennt – mehrere Werte an einer Position eintragen. Was die Angabe des auszuführenden Skripts angeht, empfiehlt es sich stets, den vollständigen (absoluten) Pfad einzutragen. Abbildung 6–46 zeigt noch weitere Cronjob-Beispielzeilen für die Datei *crontab*.

Abb. 6–46 Mehr Beispiele für die Verwendung von `cron`



Speichern nicht vergessen

Nachdem Sie Ihre Änderungen an *crontab* vorgenommen haben, stellen Sie sicher, dass Sie sie speichern. Auf diese Weise wird `cron` aktualisiert, sodass `cron` anschließend die Datei überprüft und die aufgelisteten Aufgaben ausführt.

6.23 Warum dies für Maker wichtig ist

Wenn Sie Ihre Projekte unter Linux bauen, wollen Sie zum Beispiel wissen, wie Sie die Auslastung Ihres Systems überwachen oder wie Sie neue Benutzer und Gruppen anlegen. Sie möchten die Rechte und den Eigentümer von Dateien verwalten oder Befehle und Programme automatisch ablaufen lassen. Einige dieser Aspekte kommen vielleicht bei jedem Projekt mehrfach vor, andere wiederum selten oder nie. Das erforderliche Wissen wird Ihnen aber in jedem Fall bei Problemen weiterhelfen. Sie werden Ihr Projekt auch schneller zum Abschluss bringen, sodass Sie mehr Freude daran haben, statt immer nur nach den Ursachen für Probleme zu suchen.

7 Interaktion mit der Außenwelt

Die meisten Maker wollen Projekte bauen, mit denen sie mit der Außenwelt über Sensoren, Motoren, Bauteile und Geräte in irgendeiner Form interagieren können. Die Steuerung solcher Geräte und Module geschieht in erster Linie über die Programmierung, doch auch vonseiten des Betriebssystems Linux müssen einige Voraussetzungen erfüllt sein, bevor man die Steuerung übernehmen kann. In diesem Kapitel erkläre ich Ihnen deshalb, wie man die GPIO-Pins (general-purpose input/output, Mehrzweck-Ein/Ausgabe-Kontaktstifte), das I^2C- (integrated circuit, integrierter Schaltkreis) und das SPI-Protokoll (serial peripheral interface, serielle periphere Schnittstelle) verwendet. Außerdem zeige ich Ihnen, wie man mit einem Arduino kommuniziert. Auch wenn ich, was das Programmieren betrifft, hier nicht tief in die Materie einsteigen kann, möchte ich Ihnen anhand einiger Beispiele in der Sprache Python die Grundzüge im Umgang mit diesen Schnittstellen demonstrieren.

7.1 GPIO

Eine der Möglichkeiten, ein externes Gerät oder Bauteil zu steuern, sind die GPIO-Pins, die sich auf der Platine des Raspberry Pi, aber auch auf vielen anderen SBCs und Mikrocontrollern befinden. Die 40er-Stiftleiste an der einen Kante des Raspberry Pi erlaubt einen einfachen Zugang zu den GPIO-Pins. Von diesen 40 Kontaktstiften haben allerdings nur 26 GPIO-Funktionalität. Die übrigen sind Spannungs- und Masse-Pins und solche, die nur von Aufsteckplatinen genutzt werden. Viele von ihnen lassen sich zudem für alternative Funktionen wie I^2C und SPI nutzen (siehe Abschnitt 7.2). Die Nummernbezeichnung der Pins entspricht daher nicht den GPIO-Nummern, sodass Sie beim Anschluss von Bauelementen an Ihren Raspberry Pi immer einen Plan zur Hand haben sollten (siehe Abb. 7–1).

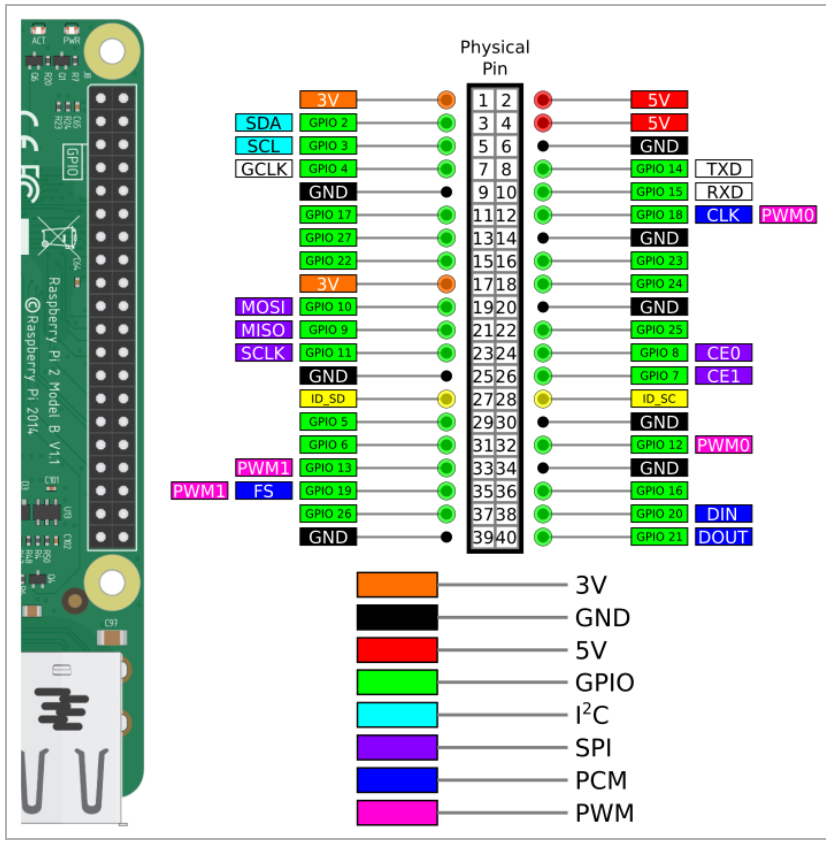


Abb. 7–1 Pinbelegung des Raspberry Pi B+

Wie Sie der Abbildung 7–1 entnehmen können, liegt GPIO 2 tatsächlich auf Pin 3. Würde man also ein Bauteil an Pin 2 anschließen, würde man dort 5 V hindurchschicken und es eventuell beschädigen. Welche Pins auf der Stiftleiste mit Zusatzfunktionen belegt sind, können Sie ebenfalls dem Plan entnehmen. Wenn Sie also den I²C-Bus aktivieren wollen, müssten Sie Ihr Bauteil an die Pins 3 und 5 anschließen, hätten dafür aber GPIO 2 und 3 nicht mehr zur Verfügung.

Zur Steuerung der GPIO-Pins gibt es für den Raspberry Pi mehrere Module bzw. Bibliotheken zur Auswahl. Eine der verbreitetsten ist die Python-Bibliothek *RPi.GPIO*. Diese einfache und nette Bibliothek wird bei vielen Programmierbeispielen, die man im Internet findet, eingesetzt. Trotzdem bevorzuge ich eine andere, und zwar *pigpio*, da sie auf dem

System als Dienst läuft und sowohl von Python oder C als auch von einem anderen Raspberry Pi im Netzwerk angesteuert werden kann.

Falls Sie im Verlauf von Kapitel 4 bereits mit `apt-get` *pigpio* installiert haben, können Sie den nächsten Schritt überspringen. Um *pigpio* zu installieren, müssen Sie mit `wget` die aktuelle Version herunterladen. Mit `wget` (»web get«) kann man über die Kommandozeile Dateien herunterladen, vorausgesetzt, man kennt die vollständige URL dieser Datei:

```
wget abyz.co.uk/rpi/pigpio/pigpio.zip
```

Entpacken Sie jetzt die ZIP-Datei mit `unzip`:

```
unzip pigpio.zip
```

Wechseln Sie nun in den Order, den Sie gerade entpackt haben:

```
cd PIGPIO
```

Anschließend kompilieren und installieren Sie die Software:

```
make -j4
sudo make install
```

Falls Sie noch nie Software kompiliert haben, wundern Sie sich wahrscheinlich über die fremdartige Bildschirmausgabe (siehe Abb. 7–2). Es handelt sich aber um eine normale Ausgabe vieler Befehle, die beim Aufbau der Software nacheinander durchlaufen.

```
strip --strip-unneeded libpigpiod_if.so
size     libpigpiod_if.so
   text    data     bss     dec     hex filename
  59271    4292   49244  112807   1b8a7 libpigpiod_if.so
gcc -shared -o libpigpiod_if2.so pigpiod_if2.o command.o
strip --strip-unneeded libpigpiod_if2.so
size     libpigpiod_if2.so
   text    data     bss     dec     hex filename
  69770    4292    1984   76046   1290e libpigpiod_if2.so
gcc -o x_pigpiod_if x_pigpiod_if.o -L. -lpigpiod_if -pthread -lrt
gcc -o pig2vcd pig2vcd.o
strip pig2vcd
gcc -o pigs pigs.o command.o
strip pigs
gcc -o x_pigpiod_if2 x_pigpiod_if2.o -L. -lpigpiod_if2 -pthread -lrt
gcc -shared -o libpigpio.so pigpio.o command.o
strip --strip-unneeded libpigpio.so
size     libpigpio.so
   text    data     bss     dec     hex filename
 248501    5336  598344  852181   d00d5 libpigpio.so
gcc -o x_pigpio x_pigpio.o -L. -lpigpio -pthread -lrt
gcc -o pigpiod pigpiod.o -L. -lpigpio -pthread -lrt
strip pigpiod
pi@raspberrypi:~/PIGPIO $ sudo make install
```

Abb. 7–2 Kompilieren des Programms pigpio mit dem Befehl make

Nach der Installation von *pigpio* können Sie den Dienst folgendermaßen im Hintergrund laufen lassen:

```
sudo pigpiod &
```

Wenn Sie möchten, dass er schon beim Hochfahren Ihres Raspberry Pi gestartet wird, können Sie die Zeile `pigpiod &` auch Ihrer *rc.local*-Datei hinzufügen. Da sie stets als »root« ausgeführt wird, ist `sudo` in diesem Fall nicht nötig.

Auf der Website von *pigpio* (*http://abyz.co.uk/rpi/pigpio/examples.html*) finden sich zahlreiche Codebeispiele in C, C++ und Python. Eines davon in Python gibt Ihnen Auskunft über den Status der einzelnen GPIO-Pins. Mit diesem Code können Sie auch überprüfen, ob der Dienst *pigpio* richtig funktioniert. Um dieses Skript zum Einsatz zu bringen, legen Sie als Erstes mit `nano` eine neue Datei für Ihr Skript an:

```
sudo nano gpio_status.py
```

Anschließend geben Sie folgenden Code direkt ein oder kopieren ihn über die Zwischenablage in die Datei:

```
#!/usr/bin/python

import time
import curses
import atexit
import pigpio

GPIOS=32
MODES=["INPUT", "OUTPUT", "ALT5", "ALT4", "ALT0", "ALT1",
"ALT2",
"ALT3"]

def cleanup():
   curses.nocbreak()
   curses.echo()
   curses.endwin()
   pi.stop()

pi = pigpio.pi()
stdscr = curses.initscr()
curses.noecho()
curses.cbreak()
atexit.register(cleanup)
cb = []
```

```
for g in range(GPIOS):
   cb.append(pi.callback(g, pigpio.EITHER_EDGE))

# disable gpio 28 as the PCM clock is swamping the system

cb[28].cancel()
stdscr.nodelay(1)
stdscr.addstr(0, 23, "Status of gpios 0-31", curses.A_REVERSE)

while True:
   for g in range(GPIOS):
      tally = cb[g].tally()
      mode = pi.get_mode(g)
      col = (g / 11) * 25
      row = (g % 11) + 2
      stdscr.addstr(row, col, "{:2}".format(g), curses.A_BOLD)
      stdscr.addstr("={} {:>6}: {:<10}".format(pi.read(g),
      MODES[mode], tally))
   stdscr.refresh()
   time.sleep(0.1)
   c = stdscr.getch()
   if c != curses.ERR:
      break
```

Danach speichern Sie die Datei, indem Sie Ctrl-X, dann J und zum Schluss Enter drücken. Jetzt müssen Sie der Datei noch Ausführungsrechte zuteilen, wie Sie es in Kapitel 6 gelernt haben:

```
chmod 755 gpio_status.py
```

Nun können Sie das entsprechende Kommando eingeben und die Ausgabe einsehen (siehe Abb. 7–3):

```
./gpio_status.py
```

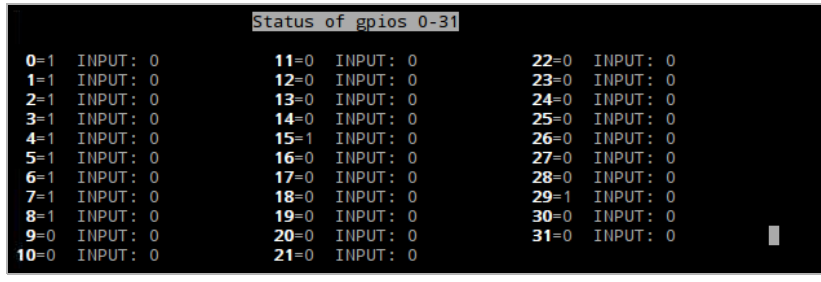


Abb. 7–3 Die Ausgabe von gpio_status.py

Jetzt können Sie den Status aller GPIO-Pins sehen, und zwar nicht nur die 26 Pins auf der 40er-Stiftleiste, sondern auch alle anderen. Wie Sie in Abbildung 7–3 erkennen können, sind die meisten GPIO-Pins als Input (Eingabe) definiert. GPIO 14 und 15 laufen unter der Zusatzfunktion `ALT0`, die sie in diesem Fall zu Senden-Empfangen-Pins serieller Daten macht (TXD und RXD).

Jetzt wollen wir diese Kenntnisse endlich anwenden. Mithilfe eines kleinen Python-Skripts wollen wir eine LED blinken lassen und ein Relais ein- und ausschalten. Zunächst zur LED: Verbinden Sie die Anode (+, längerer Pin) einer LED mit GPIO 18 (Pin Nr. 12) und die Kathode (-, kürzerer Pin) mit der Masse (GND) wie in Abbildung 7–4.

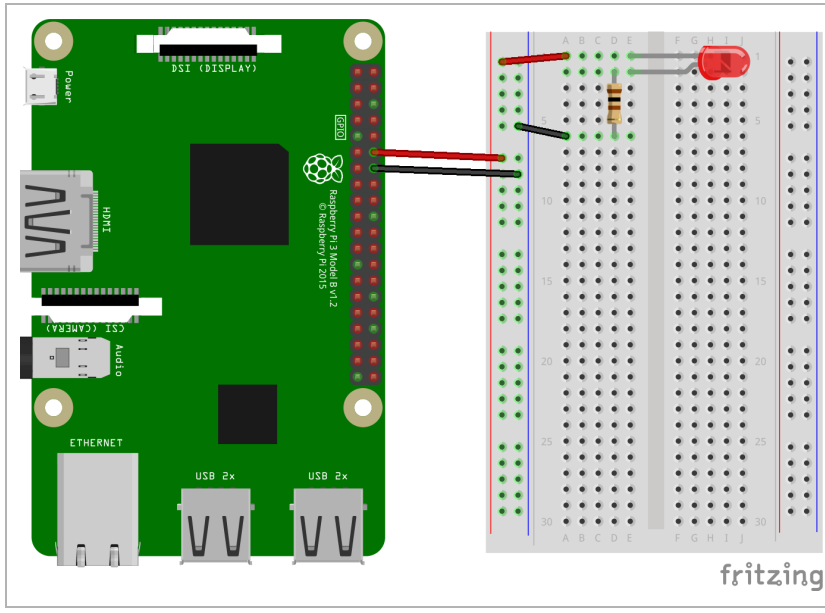


Abb. 7–4 Raspberry Pi mit einer LED an GPIO 18

Brauche ich einen Vorwiderstand?

Der Raspberry Pi schickt 3,3 V auf seine GPIO-Pins. Einige blaue oder weiße LEDs benötigen eine Flussspannung von 3,3 V, doch die mit anderen Farben wie Gelb, Rot und Grün werden bei niedrigeren Spannungen um die 2 V betrieben. Liegt der Spannungsbedarf Ihrer LED unter 3,3 V, müssen Sie zwischen der Anode (+)

→

Ihrer LED und dem GPIO-Pin einen Vorwiderstand einbauen. Suchen Sie daher in den Spezifikationen Ihrer LED zuerst die Flussspannung heraus. Auch wenn eine LED von geringerer Flussspannung ohne Widerstand funktionieren würde, so würde sie nicht lange durchhalten. Im Internet finden Sie zahlreiche gute Kalkulatoren wie beispielsweise diesen: *http://led.linear1.org/1led.wiz*. Mit ihnen lässt sich der passende Vorwiderstand leicht ermitteln.

Jetzt legen wir mit `nano` eine neue Datei für unser Python-Skript an:

```
nano gpio_blinken.py
```

```
#!/usr/bin/python

import pigpio
import time

pi = pigpio.pi ()
# GPIO 18 als Output definieren
pi.set_mode(18, pigpio.OUTPUT)
# Bei Relais sollte man den Ausgangszustand auf off stellen.
# Je nach Relais kann dies für den Pin eine Einstellung auf
# low oder high bedeuten.
pi.write(18, 1)

# Jetzt wechseln wir zwischen ein und aus
# und lassen dazwischen eine halbe Sekunde Pause.
while True:
   time.sleep(.5)
   pi.write(18, 0)
   time.sleep(.5)
   pi.write(18, 1)
```

Speichern und schließen Sie die Datei wieder mit Ctrl-X, J und Enter und geben ihr, wie Sie es in Kapitel 6 gelernt haben, Ausführungsrechte:

```
chmod 744 gpio_blinken.py
```

Jetzt können Sie das Kommando eingeben und die Wirkung überprüfen:

```
./gpio_blinken.py
```

Wenn alles korrekt verdrahtet ist, sollte Ihre LED jetzt in einer Schleife für jeweils eine halbe Sekunde leuchten und danach für eine halbe Sekunde wieder ausgehen.

Mit demselben Skript können Sie auch ein 5-V-Relais-Modul ein- und ausschalten. Dazu verbinden Sie das Relais zum einen mit demselben

GPIO-Pin (18) und dann den 5-V-Pin des Relais mit dem 5-V-Pin des Raspberry Pi wie in Abbildung 7–5 gezeigt.

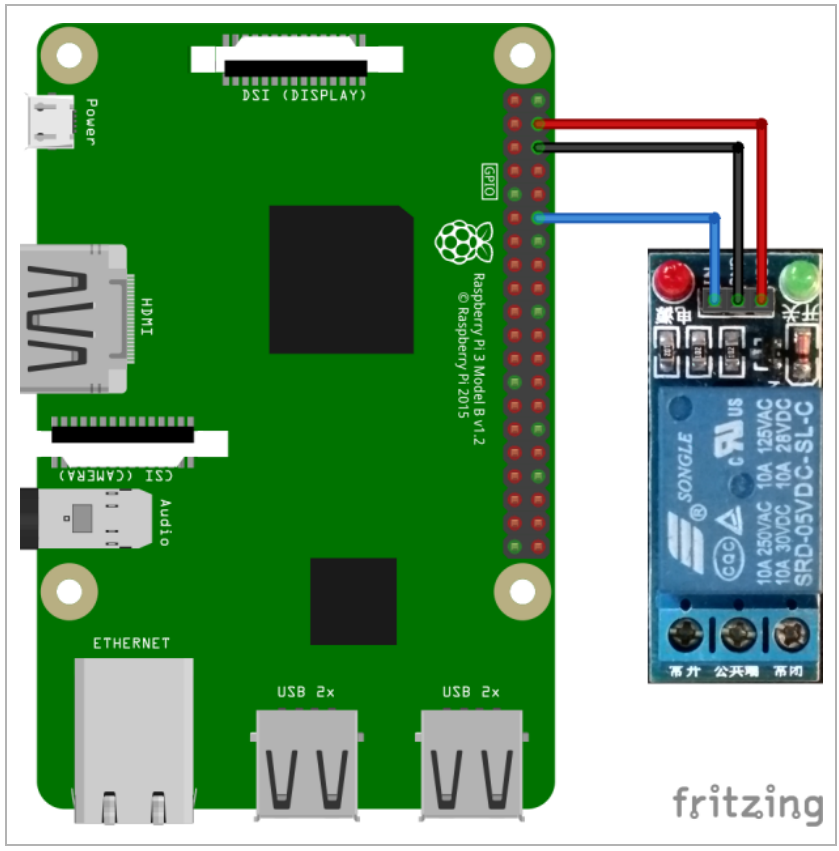


Abb. 7–5 Raspberry Pi mit angeschlossenem Relais

Mit dem komplett verdrahteten Relais-Modul hat man einen einfachen Schalter.

Achtung!

Lassen Sie im Umgang mit dem Netzstrom äußerste Vorsicht walten! Unterbrechen Sie bei der Arbeit an den Verbindungen auf jeden Fall den Stromkreis oder schalten Sie die Sicherung aus.

7.2 I²C und SPI

Bei I²C und SPI handelt es sich jeweils um serielle Datenübertragungsprotokolle, mit denen man Daten von einer Platine zur anderen hin und zurück überträgt. Für Ihren Raspberry Pi gibt es jede Menge vorgefertigter Module, die sich zur Kommunikation dieser Protokolle bedienen. I²C ist nicht so schnell wie SPI, hat jedoch den Vorteil, mit nur zwei Kabeln mehrere Bauteile verbinden zu können, da jedes Bauteil seine eigene Adresse bekommt. SPI ist zwar wesentlich schneller, benötigt aber vier oder mehr Kabel, wenn man mehr SPI-Bauteile mit demselben SBC verbinden möchte. Ich selbst musste noch nie mehr als ein Gerät zur gleichen Zeit anschließen und die Geschwindigkeit der Datenübertragung ist bei den meisten Projekten ohnehin nicht entscheidend. Finde ich also ein Zusatzmodul, das I²C unterstützt, bevorzuge ich es in aller Regel gegenüber einem mit SPI. Dennoch ist es gut, wenn man weiß, wie man beide anschließt.

Beide Protokolle müssen jeweils zuvor an Ihrem Raspberry Pi aktiviert werden. Erinnern Sie sich noch an die alternativen Funktionen der GPIO-Pins? Jetzt müssen wir tatsächlich einige ändern, damit sie I²C, SPI oder beide zugleich unterstützen. Denken Sie daran, dass Sie danach diese entsprechenden Pins nicht mehr als normale GPIO-Pins ansteuern können. Natürlich können Sie diese alternative Belegung später wieder rückgängig machen.

Um nun I²C oder SPI zu aktivieren, starten Sie `raspi-config` im Terminal (siehe Abb. 7–6):

```
sudo raspi-config
```

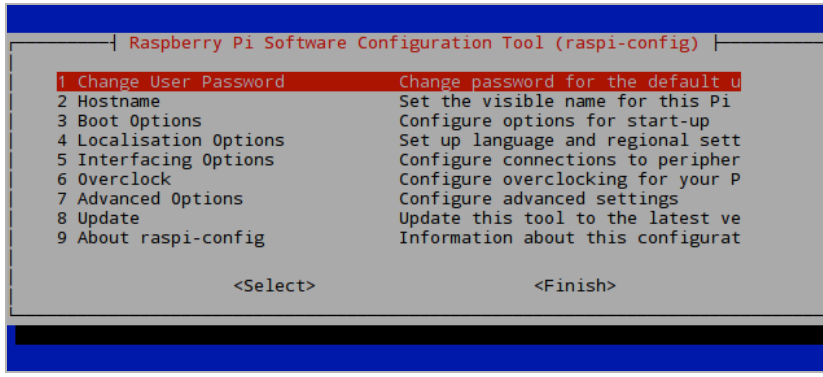


Abb. 7–6 Das Hauptfenster von raspi-config

Bewegen Sie den Cursor mit den Pfeiltasten auf die *Interfacing Options* und drücken Sie die Entertaste. Anschließend bewegen Sie den Cursor entweder auf SPI oder I²C und drücken wieder die Entertaste (siehe Abb. 7–7).

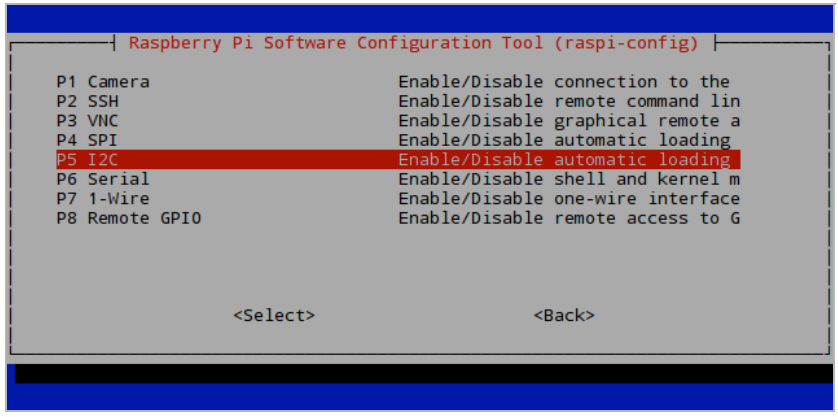


Abb. 7–7 Das Menü mit den Interfacing Options im Programm raspi-config

Anschließend lässt sich `raspi-config` noch einmal die Änderung zu I²C oder SPI von Ihnen bestätigen. Sobald Sie *Ja* ausgewählt und bestätigt haben, nimmt `raspi-config` die nötigen Änderungen vor und meldet anschließend, dass die Schnittstelle eingerichtet wurde. Kehren Sie mit Enter zum Hauptmenü zurück, wählen Sie dort *Finish* und verlassen Sie dadurch `raspi-config`.

Sollten Sie I²C mit einer älteren Raspbian-Version oder einer anderen Linux-Distribution verwenden, kann es sein, dass Sie einige Hilfsprogramme installieren müssen, die von den Bibliotheken benötigt werden, damit Sie Ihr Bauteil auch steuern können. Diese können Sie ganz einfach folgendermaßen installieren:

```
sudo apt-get update
sudo apt-get install i2c-tools
```

Der letzte Schritt besteht darin, Ihren Benutzer ein paar Gruppen hinzuzufügen, damit Sie auf jeden Fall über ausreichend Rechte verfügen, diverse Dateien und Programme lesen und ausführen zu dürfen, die für die Kommunikation mit Ihrem Bauteil nötig sind:

```
sudo usermod -a -G i2c pi
sudo usermod -a -G spi pi
sudo usermod -a -G gpio pi
```

Um ganz sicherzugehen, dass die SPI- oder I²C-Schnittstelle auch wirklich aktiviert und einsatzbereit ist, sollten Sie jetzt einen Neustart machen:

```
sudo shutdown -r now
```

Nun können Sie die Schnittstellen einsetzen.

Lernen Sie Ihr Bauteil gut kennen

Ihren Raspberry Pi für die Kommunikation über I²C oder SPI einzurichten ist das eine. Zu wissen, *was* in dieser Kommunikation stattfinden soll, ist ganz etwas anderes. Jedes Bauteil erfordert unterschiedliche Anweisungen, um dessen Möglichkeiten zu nutzen. Vor allem für Anfänger ist es wichtig, die für das Bauteil oder Modul zugeschnittene Bibliothek zu finden, da das die Programmierung später sehr erleichtert.

Aufgrund der Vielzahl an Bauteilen und Modulen, die sich an den Raspberry Pi anschließen lassen, kann ich niemals alle behandeln. Deshalb habe ich ein beispielhaftes Bauteil herausgesucht, um daran zu demonstrieren, was für die Steuerung nötig ist und wie die Schnittstellenprotokolle funktionieren. Bei dem Bauteil handelt es sich um ein kleines OLED-Display mit 128×64 Pixel. Es lässt sich an den Raspberry Pi anschließen und verwendet dabei das I²C-Protokoll. Man kann das Display dazu einsetzen, um Informationen wie das Wetter oder den Status eines laufenden Programms anzuzeigen. Ich werde mit Python darauf eine kleine Nachricht ausgeben, wenn mein Programm läuft.

Das Display läuft mit einem Treiber namens *SSD1306*, sodass ich ein Python-Modul ausfindig machen muss, das diesen Treiber unterstützt. Glücklicherweise gibt es für diese Displays zwei gute Python-Module. Das eine stammt von Adafruit (*http://bit.ly/2nkmlJ3*) und wurde auf das SSD1306-Display abgestimmt, das von dieser Firma vertrieben wird. Das andere stammt von Richard Hull (*http://bit.ly/2nhYfOe*) und unterstützt neben dem SSD1306 noch andere Display-Treiber. In diesem Beispiel verwende ich das von Richard Hull, da es dazu auch noch einige lustige Demoprogramme auf GitHub (*http://github.com/rm-hull/luma.oled*) gibt.

Als Erstes installieren wir die nötige Software und stellen sicher, dass sie auf dem aktuellen Stand ist:

```
sudo apt-get update
sudo apt-get install python-dev python-pip libfreetype6-dev
libjpeg8-dev libsdl2-dev
sudo pip install --upgrade luma.oled
```

Jetzt können wir mithilfe der Informationen über das Python-Modul von der Website (*http://bit.ly/2nhYfOe*) ein kleines Testprogramm schreiben. Legen Sie dazu mit `nano` eine neue Datei für dieses Python-Skript an:

```
nano ssd1306_beispiel.py
```

Geben Sie folgenden Code direkt ein oder kopieren Sie ihn über die Zwischenablage hinein:

```
#!/usr/bin/python

from luma.core.serial import i2c
from luma.oled.device import ssd1306, ssd1331, sh1106
from luma.core.render import canvas

# rev.1-Benutzer nehmen port=0
# fuer spi-Interface: spi(device=0, port=0)
serial = i2c(port=1, address=0X3C)

# gegebenenfalls gegen ssd1331(...)
# oder sh1106(...) austauschen

device = ssd1306(serial)
while True:
   with canvas(device) as draw:
      draw.rectangle(device.bounding_box, outline="white",
      fill="black")
      draw.text((30, 40), "Hallo Welt", fill="white")
```

Sichern und schließen Sie die Datei wieder mit Ctrl-X, dann J und Enter. Wie in Kapitel 6 gelernt, erteilen Sie der Datei Ausführungsrechte:

```
chmod 755 ssd1306_beispiel.py
```

Verbinden Sie Ihr SSD1306-Display jetzt mit Ihrem Raspberry Pi entsprechend der Abbildung 7–8.

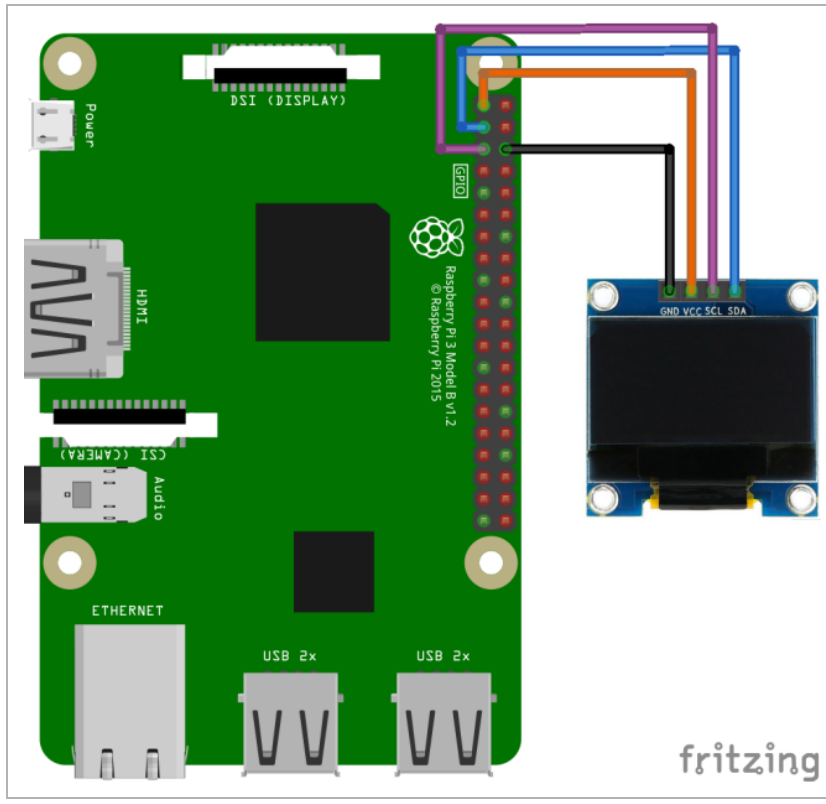


Abb. 7–8 Ein Raspberry Pi mit angeschlossenem SSD1306-Display

Jetzt können Sie das Kommando eingeben und die Ausgabe überprüfen:

```
./ssd1306_beispiel.py
```

Das OLED-Display sollte jetzt ein Rechteck mit »Hallo Welt« darin zeigen (siehe Abb. 7–9).

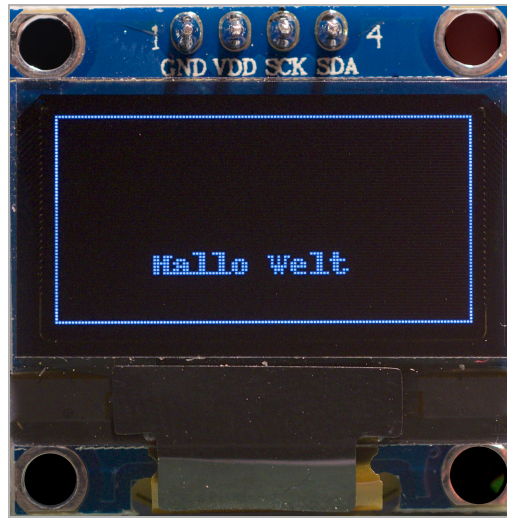


Abb. 7–9 Ein I²C-angesteuertes SSD1306-Modul, auf dem das Beispielskript ausgeführt wird

Wie ich schon sagte, hat der Entwickler dieses Softwarepakets, Richard Hull, noch ein paar weitere Beispielskripte, die man auf diesem Typ von Display laufen lassen kann. Haben Sie ein solches Display und möchten Sie die Skripte ausprobieren, müssen Sie als Erstes den entsprechenden Quellcode für das Python-Modul herunterladen:

```
wget https://github.com/rm-hull/luma.examples/archive/master.zip
```

Dann entpacken Sie die ZIP-Datei:

```
unzip master.zip
```

Wechseln Sie jetzt in das neu angelegte Verzeichnis *luma.examples-master/examples*:

```
cd luma.examples-master/examples
```

Jetzt können Sie eines der Beispielskripte laufen lassen und sich die Ausgabe anschauen (siehe Abb. 7–10):

```
./demo.py
```

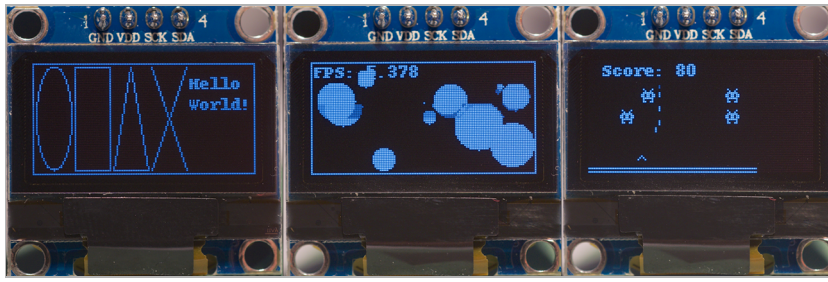


Abb. 7–10 Ausführen der Beispielskripte demo.py, bounce.py und invaders.pi auf einem SSD1306-Display

7.3 Probieren Sie es selbst

Suchen Sie sich ein Modul aus, das mit I^2C oder SPI arbeitet, und versuchen Sie es mit dem Raspberry Pi zum Laufen zu bekommen. Sie können ein Display, einen Temperatursensor, einen Beschleunigungssensor oder sonst etwas wählen. Finden Sie auf jeden Fall vorher heraus, ob dieses Bauteil oder Modul einen bestimmten Treiber oder Chipsatz verwendet, und nutzen Sie diese Informationen bei der Internetrecherche nach der passenden Programmbibliothek oder dem Modul für Ihr Gerät.

7.4 Verbindung zu einem Arduino

Manchmal ist es erforderlich, einen Raspberry Pi an einen Arduino anzuschließen, da man die Zuverlässigkeit oder die Verarbeitungsgeschwindigkeit erhöhen will. Es kann aber auch sein, dass Sie bei Aufgaben wie der Steuerung von Sensoren, Motoren etc. mit dem Arduino einfach geübter sind. Doch auch dann können Sie von der Flexibilität eines Raspberry Pi profitieren, indem Sie ihn gleichzeitig mit einem Arduino verwenden und mit ihm über das I^2C-Protokoll kommunizieren.

Überprüfen Sie Ihre Verbindungen

Die Signale des Arduinos gehen im Regelfall mit 5 V über die Pins, die des Raspberry Pi hingegen mit 3,3 V. Wenn Sie nun wie im folgenden Beispiel den I²C-Bus mit dem Raspberry Pi als Master benutzen, sollte das in Ordnung gehen. Doch aus Versehen einen Pin mit 5 V Spannung falsch zu verbinden, könnte den Raspberry Pi beschädigen. Deshalb sollten Sie Ihre Verbindungen immer noch einmal zusätzlich überprüfen. Wollen Sie dennoch 5-V-Signale an 3,3-V-Pins schicken, sollten Sie einen Pegelwandler (Level Shifter) zwischenschalten.

Bei dieser Übung werde ich Ihnen zeigen, wie man den Raspberry Pi als I²C-Master einsetzt, der Informationen abfragt, die der Arduino über I²C liefert. Als Erstes programmieren wir dazu den Arduino. Dies geschieht in der *Arduino IDE*, der kostenlosen Entwicklungsumgebung (Integrated Development Environment) für den Arduino (*http://www.arduino.org/downloads*). Dort machen wir einen neuen »Sketch« auf, geben folgenden Code ein und laden ihn anschließend auf den Arduino hoch:

```
#include <Wire.h>

void setup()
{
   Wire.begin(8);                // folgt i2c-Bus von Adresse #8
   Wire.onRequest(requestEvent); // Event anmelden
}

char str[17];
int x = 0;

void requestEvent() {
   sprintf(str, "Message %7d\n", x);
   if (++x > 9999999) x=0;
   Wire.write(str);              // sendet 16 Bytes
}

void loop() {
   delay(50);
}
```

Dieser Sketch weist den Arduino an, als »Slave« auf dem I²C-Bus zu fungieren und auf Anfragen mit dem Wort »Message« und einer Zahl zu antworten, die bei jeder Anfrage ansteigt. Dieses Szenario symbolisiert die Lieferung von Daten, die von einem Sensor oder einem anderen Bauteil stammen könnten, das mit einem Arduino verbunden ist.

Jetzt legen wir wieder eine Datei mit Python-Code auf dem Raspberry Pi an, die die Daten vom Arduino abfragt:

```
nano i2c_master.py
```

Geben Sie folgenden Code ein oder kopieren Sie ihn über die Zwischenablage hinein:

```
#!/usr/bin/python

import time
import pigpio

BUS=1
I2C_ADDR=8

pi = pigpio.pi()
#Oeffnen der Verbindung zum Slave
h = pi.i2c_open(BUS, I2C_ADDR)

while True:
   #allgemeine Abfrage ohne Aufzeichnung
   (c, d) = pi.i2c_read_device(h,16)
   if c >= 0:
      print d
   else:
      print "Keine Daten ..."
   time.sleep(.5)

pi.i2c_close(h)
pi.stop()
```

Speichern und schließen Sie die Datei wieder durch Ctrl-X, dann J und Enter. Wie in Kapitel 6 gelernt, geben Sie der Datei Ausführungsrechte:

```
chmod 755 i2c_master.py
```

Falls nicht schon geschehen, müssen Sie den Dienst `pigpio` im Hintergrund starten:

```
sudo pigpiod &
```

Jetzt stellen Sie anhand von Abbildung 7–11 die nötigen Verbindungen zwischen Raspberry Pi und dem Arduino her:

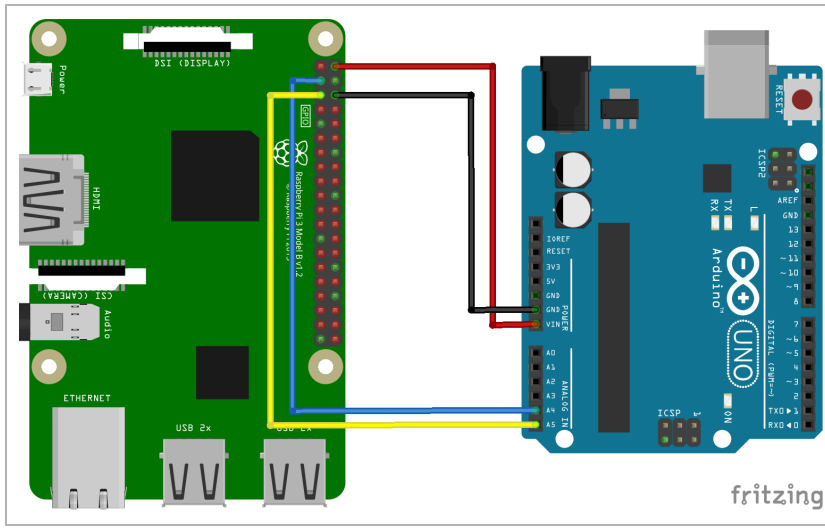


Abb. 7–11 Verbindungen eines Raspberry Pi als I^{2}C-Master mit einem Arduino als Slave

Jetzt können Sie das Kommando laufen lassen und die Ausgabe überprüfen:

```
./i2c_master.py
```

Das Python-Skript verwendet das zuvor von uns installierte Programm `pigpio`, um fortlaufend Anfragen an den Arduino zu richten, und die Antworten, die es erhält, auszugeben (siehe Abb. 7–12). Würde es sich um echte Daten handeln, könnten wir diese in unserem Programm verwenden. Empfängt das Skript keine Daten, gibt es »Keine Daten …« aus.

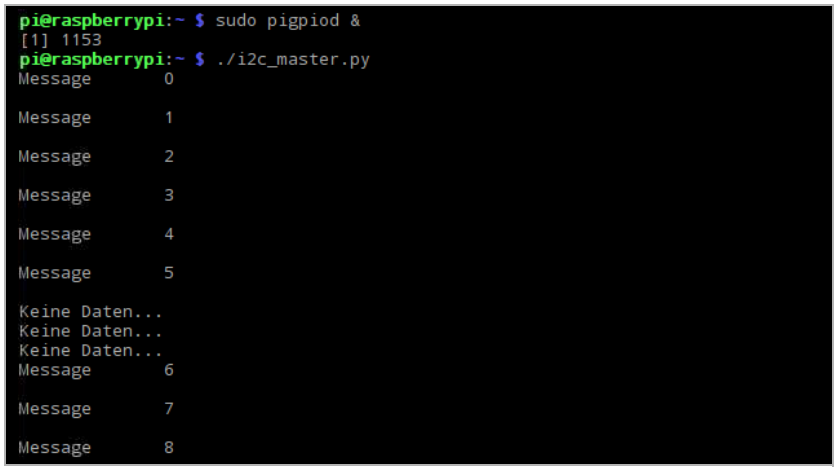


Abb. 7–12 Der Raspberry Pi erhält kontinuierlich Daten von einem Arduino (es sei denn, man unterbricht die Datenverbindung wie hier zwischenzeitlich).

7.5 Warum dies für Maker wichtig ist

Bei vielen Projekten von Makern interagieren Bauteile und Geräte mit SBCs wie dem Raspberry Pi. Zu wissen, wie man die GPIO-Pins zur Steuerung und Kommunikation verwendet und dabei mit Linux über die Kommandozeile arbeitet, hilft dabei, seine Projekte schneller fertigzustellen, und eröffnet neue Möglichkeiten der Kreativität. Da Protokolle wie I^2C und SPI verbreitete Standards sind, kann man lernen, mit einer Vielzahl von Modulen zu interagieren, und damit eine Funktionalität erzielen, wie sie mit dem Raspberry Pi allein nie möglich wäre.

8 Einsatz von Multimedia

Um seinem Projekt etwas Leben einzuhauchen, sind Audio und Video hervorragende Mittel. Vor Kurzem habe ich an einem Laser-Labyrinth gearbeitet, das ich zusammen mit einigen Kollegen auf diversen Maker-Messen und anderen Veranstaltungen in meiner Region aufbaue. Der 3×6 Meter große Raum ist voller Laserstrahlen, Rauch und Schaltern. Das ist zwar für sich genommen schon ganz toll, doch haben wir festgestellt, dass durch ein paar Start- und Stoppsounds zusammen mit etwas spannender Musik die Attraktion sehr gewinnt.

Ganz gleich, ob Sie nur eine einfache Türklingel oder ein hochkomplexes Videoangebot realisieren wollen, Sie müssen wissen, wie Sie Mediendaten in Ihr Projekt integrieren. Unter Linux gibt es zahlreiche Möglichkeiten, mit Multimedia zu arbeiten. In diesem Kapitel werde ich einige der Fallstricke erläutern, auf Dinge hinweisen, die es zu beachten gilt, und die effektivsten Möglichkeiten präsentieren, Multimedia in die Projekte zu integrieren.

8.1 Audio: HDMI oder analog

Möchte man Hörbares in seinem Projekt verwenden, muss man als Erstes festlegen, wo das Audiosignal ausgegeben werden soll. Standardmäßig versucht der Raspberry Pi das Audiosignal nämlich auf den HDMI-Anschluss zu schicken. Das geht zwar völlig in Ordnung, wenn man ein HDMI-fähiges Display in seinem Projekt verwendet, doch oftmals ist entweder gar kein Monitor angeschlossen oder dieser besitzt keine Lautsprecher. In solchen Fällen müssen Sie Ihrem Raspberry Pi zunächst mitteilen, dass er das (analoge) Audiosignal an die Audiobuchse schicken soll.

Dazu rufen wir über die Kommandozeile das Konfigurationsprogramm auf:

```
sudo raspi-config
```

Wir gelangen dadurch in das Konfigurationsprogramm des Raspberry Pi (siehe Abb. 8–1).

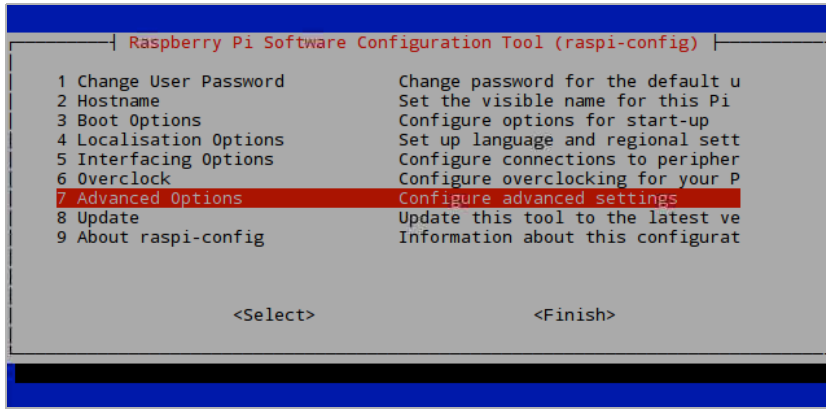


Abb. 8–1 Das Hauptmenü von `raspi-config`

Bewegen Sie den Cursor mit den Pfeiltasten auf *Advanced Options* und drücken Sie dann die Entertaste. Anschließend gehen Sie mit dem Cursor auf *Audio* und drücken wieder Enter (siehe Abb. 8–2).

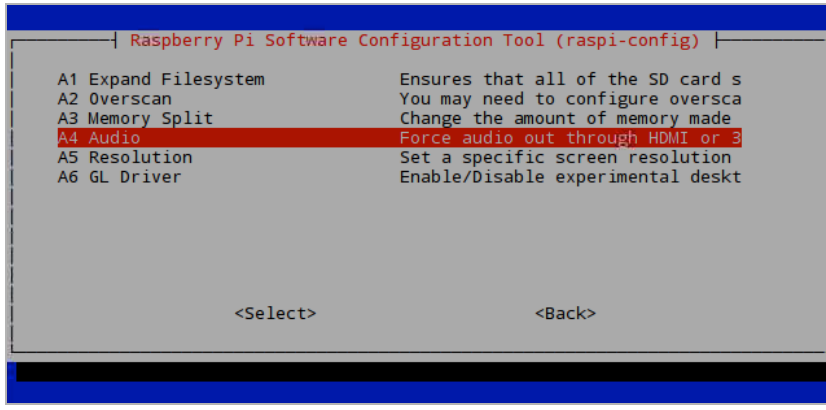


Abb. 8–2 Das Menü Advanced Options in `raspi-config`

Hier wählen Sie die Option *Force 3.5mm ('headphone') jack* und drücken Enter (siehe Abb. 8–3).

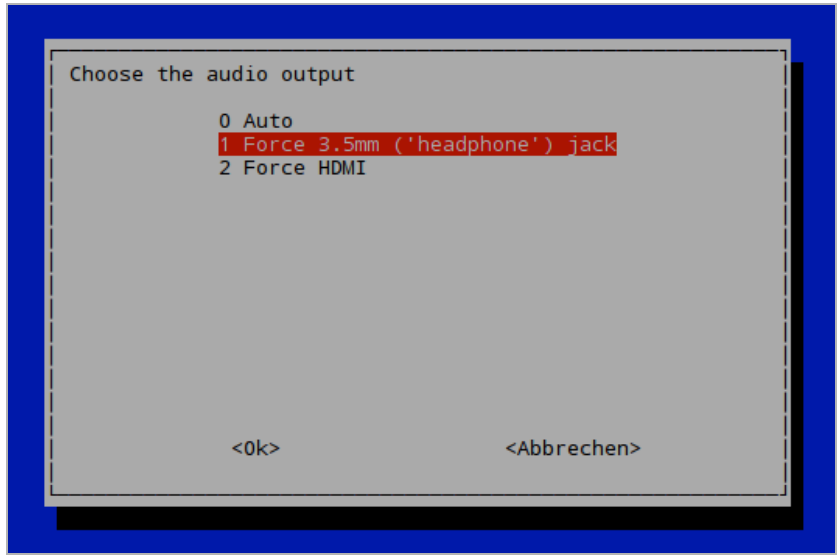


Abb. 8–3 Das Menü Audio in `raspi-config`

8.2 Audio- und Videodateien abspielen

Linux bietet viele Programme zum Abspielen von Audio- und Videodateien. Auf dem Raspberry Pi ist vor allem `omxplayer`, das mit der Distribution Raspbian mitgeliefert wird, eine gute Wahl. Der Hauptgrund für die Empfehlung von `omxplayer` ist dessen Einbeziehung des Grafikchips (GPU) bei der Decodierung. Dadurch belasten vor allem HD-Videos das System nicht allzu sehr. Außerdem kann `omxplayer` viele Mediendateiformate wie etwa MP3 und MP4 decodieren, sodass man für die jeweiligen Dateiformate nicht unterschiedliche Programme bemühen muss.

Ich werde oft gefragt, ob man `mplayer` oder `vlc` zur Videowiedergabe auf dem Raspberry Pi installieren soll. Ich rate meist davon ab, weil diese Hilfsprogramme zurzeit noch nicht in der Lage sind, die GPU mitzubenutzen und allein aus der Software heraus über die CPU laufen, sodass sie andere Programme dadurch zum Stillstand bringen können.

Um eine Mediendatei (Audio oder Video) mit `omxplayer` abzuspielen, geben Sie folgendes Kommando ein:

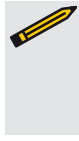
```
omxplayer -o [local / hdmi] Dateiname
```

Soll das Audiosignal analog aus dem 3,5-mm-Klinkenanschluss kommen, geben Sie hier `local` ein (ansonsten `hdmi` für ein HDMI-Display mit Lautsprechern). Leider hält sich `omxplayer` nicht immer an die in `raspi-config` festgelegte Audioausgabe, sodass es sich empfiehlt, sie hier noch einmal anzugeben. Außerdem können Sie an dieser Stelle die in `raspi-config` eingestellte Standardeinstellung der Audioausgabe im Einzelfall außer Kraft setzen.

Nach Eingabe des Kommandos werden einige Daten über die Codierung der Audio- oder Videodatei angezeigt und die Datei dann abgespielt. Danach kommt man zurück zum Prompt. Möchten Sie vorher abbrechen, drücken Sie die Taste Q (siehe Abb. 8–4).

```
pi@raspberrypi:~ $ omxplayer -o local tts.mp3
Audio codec mp3 channels 1 samplerate 24000 bitspersample 16
Subtitle count: 0, state: off, index: 1, delay: 0
have a nice day ;)
pi@raspberrypi:~ $ omxplayer -o local Nyan\ Cat.mp4
Video codec omx-h264 width 540 height 360 profile 578 fps 29.970030
Audio codec aac channels 2 samplerate 44100 bitspersample 16
Subtitle count: 0, state: off, index: 1, delay: 0
V:PortSettingsChanged: 540x360@29.97 interlace:0 deinterlace:0 anaglyph:0 par:1.
00 display:0 layer:0 alpha:255 aspectMode:0
Stopped at: 00:00:17
have a nice day ;)
pi@raspberrypi:~ $
```

Abb. 8–4 Mit omxplayer Audio- und Videodateien abspielen



Kein Bild?

Greifen Sie über SSH oder VNC auf Ihren Raspberry Pi zu, sehen Sie kein Videosignal, da das Videosignal über den HDMI- oder analogen Videoanschluss geht.

8.3 Lautstärkeregelung

Von der Kommandozeile aus gibt es zwei Arten der Lautstärkeregelung. Um die Lautstärke des gesamten Systems zu regeln, kann man das Programm `alsamixer` aufrufen. Nach diesem Befehl erhält man eine grafische Darstellung der Lautstärke des im Raspberry Pi verbauten Soundchips (siehe Abb. 8–5).

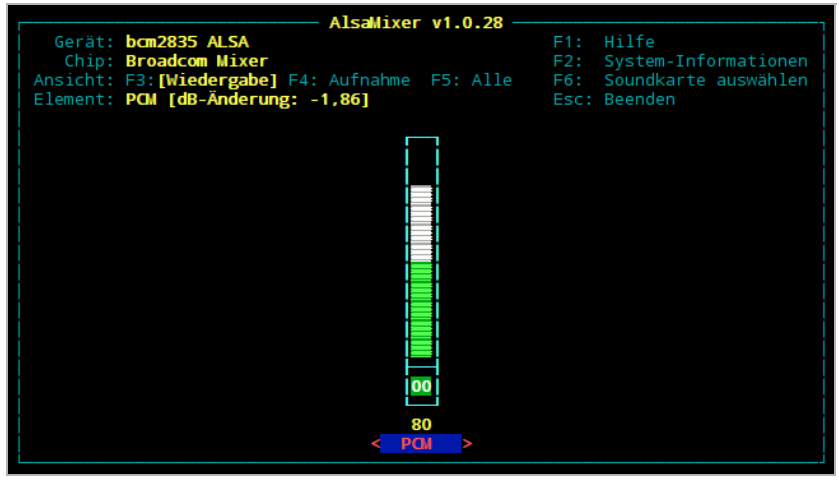


Abb. 8–5 Lautstärkeregelung auf dem Raspberry Pi mit alsamixer

Mit den Pfeiltasten für hoch und runter lässt sich die Lautstärke regeln und mit der Taste M stummschalten. Mit der Escapetaste verlässt man das Programm.

Die andere Möglichkeit der Lautstärkeregelung bietet `omxplayer` beim Abspielen einer Datei. Dies ist besonders dann hilfreich, wenn man eine einzelne Datei an das Lautstärkeniveau anderer anpassen muss. Dazu verwendet man wie zuvor `omxplayer`, fügt aber die Option `--vol` hinzu:

```
omxplayer -o [local / hdmi] -- vol <millibels>  Dateiname
```

Die *millibels* rangieren zwischen 500 (laut) und -4000 (sehr leise), wobei 0 den unveränderten Normalzustand darstellt. Werte oberhalb des Zahlenbereichs führen in der Regel zu starken Verzerrungen.

8.4 Mediendateien über ein Skript abspielen

Egal, ob Sie nun lieber in Python, Perl, Java, Go oder Ruby programmieren, es gibt viele Möglichkeiten, Mediendateien über ein Skript abspielen zu lassen. Die meisten Programmiersprachen können entsprechende Kommandos an das zugrunde liegende Betriebssystem senden. Dies ist zwar nicht unbedingt die beste Wahl, doch für kleine Projekte, die auf einem SBC laufen sollen, ist es häufig der einfachste Weg, um ordentliche Ergebnisse zu erzielen. Nach dem, was Sie bis jetzt gelernt haben, können Sie mithilfe dieser Befehle auf Ihrem Raspberry Pi Mediendateien über ein Skript abspielen lassen. Hier ein Beispiel in Python:

```
#/usr/bin/python

import os

os.system("omxplayer -o local Dateiname.mp3")
```

In diesem Fall führt die Methode `os.system` das Kommando innerhalb der Anführungsstriche im lokalen Betriebssystem aus, wodurch die MP3-Datei abgespielt wird.

8.5 Warum dies für Maker wichtig ist

Wenn man seine Projekte in die Tat umsetzt, sollte man auch an die Ästhetik denken. Oftmals lassen sich Klang- und Bildelemente integrieren. Zu wissen, wie man diese über die Kommandozeile abspielt, bringt nicht nur für Sie Leben in das Projekt, sondern auch für alle anderen, die es später benutzen.

9 Zugang zu Cloud-Diensten

Im Zeitalter des »Internets der Dinge« (Internet of Things, IoT) wird die Kommunikation mit Cloud-Diensten oder die Schaffung einer eigenen Cloud-Lösung mit Linux bei immer mehr Projekten zum wesentlichen Bestandteil. Vielleicht möchten auch Sie Ihre lokalen Dateien Ihres Raspberry Pi mit einem Cloud-Dienst synchronisieren oder eine Speicherlösung zu Hause realisieren, auf die die ganze Familie zugreifen kann. Oder Sie wollen von jedem Ort der Welt Ihre Gartenbewässerung aufdrehen können oder eine Nachricht erhalten, wenn die Beleuchtung zu Hause schon zu lange brennt. In diesem Kapitel möchte ich Ihnen den Weg zum Umgang mit Cloud-Lösungen unter Linux ebnen.

9.1 Cloud-Dienste über die Kommandozeile erreichen

Falls Sie, wie ich, schon seit einiger Zeit Ihre eigenen Programme schreiben, haben Sie sie bestimmt auch schon sowohl auf Ihrem lokalen Computer als auch zur Sicherheit in der Cloud gespeichert. Manchmal möchte ich auch zu Hause an einem Programm arbeiten und es dann später mit meinem Raspberry Pi ausprobieren. Ist mein Programm auch in der Cloud gespeichert, wird es mithilfe von Software auf meinen anderen lokalen Computer synchronisiert.

Von solchen Diensten, die die eigenen Dateien in der Cloud speichern, gibt es erstaunlich viele. Ich verwende für meine Zwecke Google Drive – Sie bevorzugen vielleicht eher Dropbox oder einen anderen Dienst. Zurzeit gibt es keinen nativen Google Drive-Client für Linux. Für Dropbox hingegen gibt es zwar einen Client mit grafischer Benutzeroberfläche, jedoch gibt es keinerlei Möglichkeit, programmiertechnisch auf Dropbox zuzugreifen. Das ist für die normale Synchronisation von Dateien zwar kein Problem, doch sobald man Dropbox in einen Cronjob z. B. im Rahmen eines Backup-Skripts einbinden möchte, stößt man an seine Gren-

zen. Deshalb zeige ich Ihnen trotz der Vielzahl an Softwarelösungen für die diversen Cloud-Dienste ein Tool, mit dem Sie auf Dateien vieler unterschiedlicher Dienste zugreifen können.

Das Kommandozeilenprogramm `rclone` wurde in der Programmiersprache Go geschrieben und kann mit vielen Cloud-Diensten interagieren und sowohl Dateien als auch ganze Ordner synchronisieren. Man kann damit Cloud-Speicherorte lokal mounten und sogar Cloud-Dienste untereinander synchronisieren (z.B. Google Drive zu Dropbox). Folgende Dienste werden zurzeit von `rclone` unterstützt:

- Amazon Drive
- Amazon S3
- Backblaze B2
- das lokale Dateisystem
- Dropbox
- Google Cloud Storage
- Google Drive
- Hubic
- Microsoft One Drive
- Openstack Swift/Rackspace Cloud Files/Memset Memstore
- Yandex Disk

Um `rclone` zu installieren, laden Sie zunächst die aktuelle Version mit `wget` herunter. Wie bereits erwähnt, steht `wget` für »web get« und wird für das Herunterladen von Dateien aus dem Internet über die Kommandozeile verwendet. Dazu muss man die vollständige URL zur Datei kennen:

```
wget http://downloads.rclone.org/rclone-current-linux-arm.zip
```

Mit dem Hilfsprogramm `unzip` entpacken Sie die Datei, sodass die Dateien in einem neu angelegten Ordner an Ihrem aktuellen Standort im Dateisystem abgelegt werden:

```
unzip rclone-current-linux-arm.zip
```

Wechseln Sie nun in diesen neuen Ordner, dessen Name aufgrund der Versionsnummer vom folgenden abweichen kann. Falls Sie ihn nicht sehen, können Sie ihn mit `ls` nachschauen oder die Funktion der automatischen Vervollständigung nutzen:

```
cd rclone-v1.36-linux-arm
```

Bevor Sie mit *rclone* arbeiten, müssen Sie es mit der Option `config` einrichten. In Abbildung 9–1 sehen Sie ein Beispiel dafür, wie das für Google Drive aussehen kann. Eine ausführliche Anleitung finden Sie auf der Website von `rclone` (*http://rclone.org*):

```
./rclone config
```

```
pi@raspberrypi:~/rclone-v1.36-linux-arm $ ./rclone config
2017/07/17 21:19:41 NOTICE: Config file "/home/pi/.config/rclone/rclone.conf" no
t found - using defaults
No remotes found - make a new one
n) New remote
r) Rename remote
c) Copy remote
s) Set configuration password
q) Quit config
n/r/c/s/q> n
name> gdrive
Type of storage to configure.
Choose a number from below, or type in your own value
 1 / Amazon Drive
   \ "amazon cloud drive"
 2 / Amazon S3 (also Dreamhost, Ceph, Minio)
   \ "s3"
 3 / Backblaze B2
   \ "b2"
 4 / Dropbox
   \ "dropbox"
 5 / Encrypt/Decrypt a remote
   \ "crypt"
 6 / Google Cloud Storage (this is not Google Drive)
   \ "google cloud storage"
 7 / Google Drive
   \ "drive"
 8 / Hubic
   \ "hubic"
 9 / Local Disk
   \ "local"
10 / Microsoft OneDrive
   \ "onedrive"
11 / Openstack Swift (Rackspace Cloud Files, Memset Memstore, OVH)
   \ "swift"
12 / SSH/SFTP Connection
```

Abb. 9–1 Beispiel für die Konfiguration von rclone für Google Drive

Nachdem das Konfigurationsskript gestartet ist, drücken Sie die Taste N, um einen neuen Cloud-Dienst (*remote service*) einzurichten. Anschließend geben Sie ihm eine aussagekräftige Bezeichnung. Wählen Sie nun den Dienst, den Sie einrichten wollen. Ich habe mich hier für Google Drive entschieden. Die abgefragten Informationen unterscheiden sich von Dienst zu Dienst. Es kann sein, dass Sie einen API-Key brauchen oder Benutzername und Passwort eingeben müssen, damit Sie sich mit dem gewünschten Cloud-Dienst verbinden können. Bei Google Drive kann man bei `client_id` und `client_secret` einfach mit der Entertaste weitergehen. Sobald das Skript bei *autoconfig* angekommen ist, können Sie die Taste N drücken. Läuft das Terminal auf dem Desktop, werden Sie zu

einem Browserfenster geführt, um den Zugriff von `rclone` auf Ihr Google-Konto zuzulassen. Anschließend kehren Sie in das Terminal zurück, wo bereits der Zugangsschlüssel (*token*) eingebaut wurde. Sie müssen nur noch mit *Yes* bestätigen und mit der Taste Q das Konfigurationsskript verlassen können (siehe Abb. 9–2).

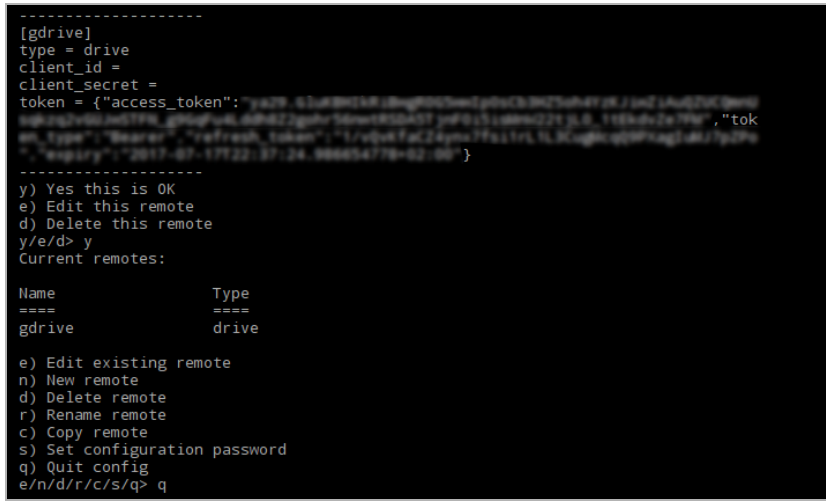


Abb. 9–2 Beispiel für die Konfiguration von rclone für Google Drive (Fortsetzung)

Jetzt können Sie mit `rclone` Dateien von Ihrem Speicherservice herunter- und hochladen. Die Dateien werden von `rclone` verglichen und es werden nur diejenigen übertragen, die sich geändert haben. Die Syntax von `rclone` ähnelt der von `cp`. Hier finden Sie eine Reihe nützlicher Kommandos mit `rclone`:

Dateien auflisten:

```
./rclone ls [Name des Cloud-Dienstes]:
./rclone ls [Name des Cloud-Dienstes]:[Verzeichnis]
```

Nur Ordner auflisten:

```
./rclone lsd [Name des Cloud-Dienstes]:
./rclone lsd [Name des Cloud-Dienstes]:[Verzeichnis]
```

Dateien von einem Ort zum anderen kopieren:

```
./rclone copy [Name des Cloud-Dienstes]:[Verzeichnis] [lokales
Verzeichnis]
./rclone copy [lokales Verzeichnis] [Name des Cloud-
Dienstes]:[Verzeichnis]
```

Im Beispiel in Abbildung 9–3 wollte ich einige Sounddateien aus meinem Google Drive auf meinen Raspberry Pi übertragen. Deshalb habe ich auf meinem Raspberry Pi ein Unterverzeichnis angelegt und die Dateien dann dort hineinkopiert.

```
pi@raspberrypi:~/rclone-v1.34-linux-arm $ mkdir Sounds
pi@raspberrypi:~/rclone-v1.34-linux-arm $ ./rclone copy gdrive:Sounds /home/pi/r
clone-v1.34-linux-arm/Sounds/
2016/12/08 21:54:16 Local file system at /home/pi/rclone-v1.34-linux-arm/Sounds:
 Waiting for checks to finish
2016/12/08 21:54:16 Local file system at /home/pi/rclone-v1.34-linux-arm/Sounds:
 Waiting for transfers to finish
2016/12/08 21:54:17
Transferred:   9.223 MBytes (1.591 MBytes/s)
Errors:                 0
Checks:                 0
Transferred:           29
Elapsed time:        5.7s
pi@raspberrypi:~/rclone-v1.34-linux-arm $
```

Abb. 9–3 Mit rclone Dateien aus der Cloud kopieren

9.2 IFTTT

Mithilfe des webbasierten Dienstes IFTTT (für *if this, then that*, wenn dies, dann das) kann man IoT-Geräte über die Cloud erreichen und aufgrund selbst eingestellter Kriterien Aktionen auslösen. Wenn beispielsweise ein an einem Raspberry Pi angeschlossener Temperatursensor einen bestimmten Wert überschreitet, können Sie sich eine E-Mail mit Datum und Uhrzeit dieses Ereignisses schicken lassen. Sie können auch über einen digitalen Assistenten wie Amazon Alexa oder Google Home durch den Raspberry Pi die Garagentür öffnen lassen. IFTTT sorgt in solchen Fällen für die Übermittlung dieses Signals. Solche Dinge sind natürlich auch ohne IFTTT möglich, doch mit IFTTT geht es viel einfacher, weil dort schon viele Dienste und Geräte vorkonfiguriert sind.

Damit Ihr System auch mit IFTTT zusammenarbeiten kann, müssen Sie entweder die ein- oder die ausgehende Kommunikation einrichten oder auch beide. Dazu können Sie das Programm `curl` im Terminal aufrufen und die Kommandos an IFTTT schicken. Das für Testzwecke hervorragend geeignete `curl` hat seinen Namen von *command-line URL*. Für aufwendigere Anwendungen können Sie einen Python-basierten Webserver laufen lassen, der die ein- und ausgehenden Nachrichten mit IFTTT austauscht.

Bevor Sie Ihren Raspberry Pi mit IFTTT einsetzen können, müssen Sie dort ein Konto anlegen und sich für den Maker-Service *Webhooks* registrieren. Im Internet finden sich viele Anleitungen, die einem dabei behilflich sind. Nachdem Sie sich beim Maker-Service registriert haben, gehen Sie in die Einstellungen (Settings) und notieren bzw. speichern Sie Ihre URL (siehe Abb. 9–4).

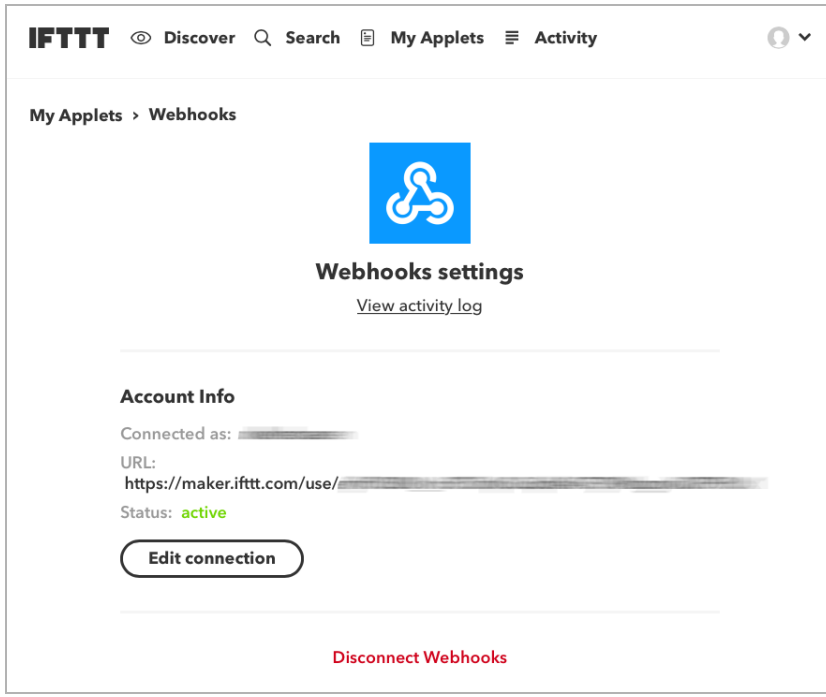


Abb. 9–4 Einstellungen von Webhooks bei IFTTT

Jetzt müssen Sie ein Applet erzeugen, das Sie von Ihrem Raspberry Pi aus auslösen (triggern) können. Gehen Sie dazu auf Ihr Profil und wählen Sie *New Applet*. Anschließend klicken Sie auf der nächsten Seite auf das blaue *this* und wählen *Webhooks* (siehe Abb. 9–5).

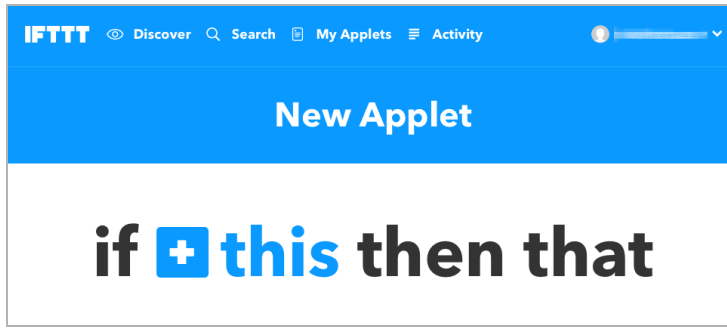


Abb. 9–5 Der erste Schritt beim Erzeugen des Applets bei IFTTT

Haben Sie *Webhooks* ausgewählt, gibt es dort nur einen Trigger (Auslöser) zur Auswahl (siehe Abb. 9–6).

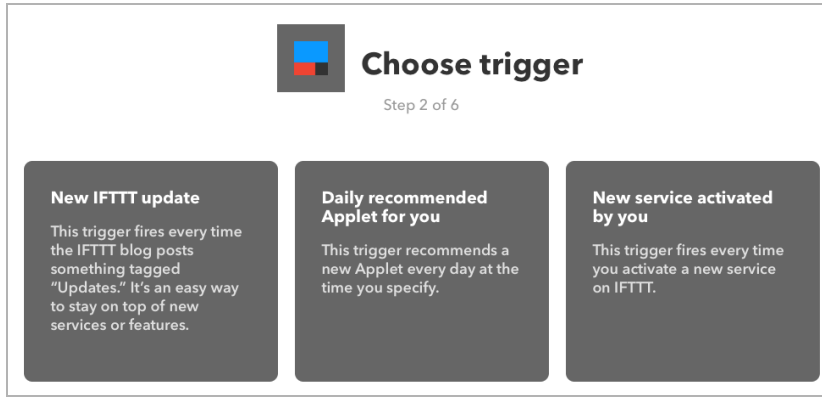


Abb. 9–6 Auswahl des Triggers (Auslösers) im Maker-Service Webhooks

Klicken Sie nun auf den Trigger und geben Sie ihm wie gefordert einen Namen. Ich habe ihn *knopfdruck* genannt. Danach sollen Sie durch Klicken auf das blaue *that* eine Aktion auswählen, die durch den Trigger ausgelöst werden soll (siehe Abb. 9–7).

Abb. 9–7 Auswahl der auszulösenden Aktion durch IFTTT

Diese Aktion findet statt, wenn der Raspberry Pi eine Webanfrage (web request) auslöst. Ich habe mich hier für eine E-Mail entschieden, die an mich gesendet werden soll. Für diesen Dienst gibt es nur einen Trigger. Klicken Sie deshalb einfach auf ihn und bestimmen Sie die Einzelheiten der E-Mail, die an Sie geschickt wird (siehe Abb. 9–8).

Nachdem Sie alles nach Ihren Wünschen eingerichtet haben, klicken Sie auf den Button *Create action*. Danach können Sie Ihr Applet begutachten und auf *Finish* klicken.

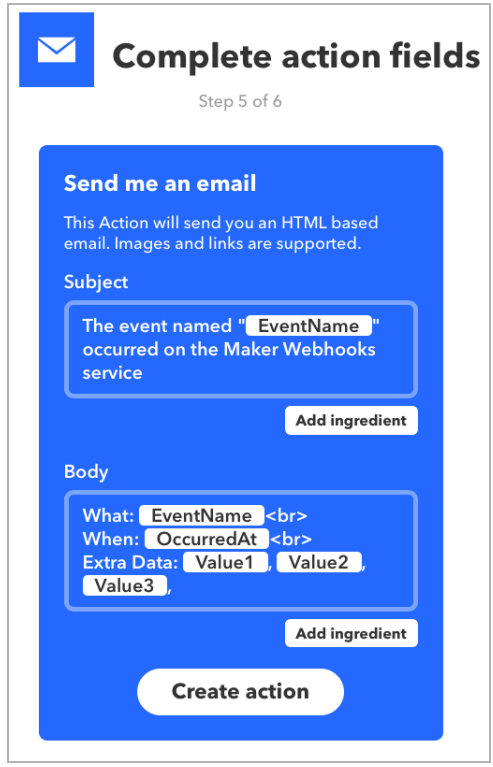


Abb. 9–8 Die Optionen für die Aktion des E-Mail-Dienstes

Um Ihr neues Applet von der Kommandozeile aus verwenden zu können, müssen Sie zuerst die URL eingeben, die Sie vorher im Browser nachgesehen haben (in der Dokumentation Ihres Maker-Service *Webhooks*). Ihr Schlüssel/Key steht ganz oben auf der Seite, auf der Sie auch nachlesen können, wie man das Applet von einem Browser aus nutzt, und sogar ein Beispiel mit `curl` für die Kommandozeile finden (siehe Abb. 9–9).

Abb. 9–9 URL in der Dokumentation von Webhooks bei IFTTT

Als einmalige Aktion können Sie das komplette Kommando mit `curl` im Terminal eingeben oder hineinkopieren, wobei Sie {event} einfach mit Ihrem eigenen Event-Namen austauschen. Stimmen alle Angaben, sollten Sie eine E-Mail erhalten, in der steht, dass das Event ausgelöst wurde (siehe Abb. 9–10).

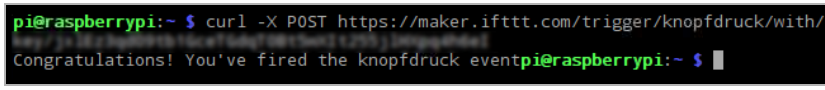


Abb. 9–10 IFTTT mit curl auslösen (triggern)

Das ist zwar alles gut und schön, wenn es sich um vom Raspberry Pi ausgehende Trigger handelt, doch eingehende Anfragen (Requests), mit denen man Dinge lokal durch externe Anfragen auslösen kann, können so nicht verarbeitet werden. Für diesen Zweck setzen wir einen einfachen Webserver auf, der sowohl auf Knopfdruck eine Anfrage losschicken kann als auch eine Anfrage in Form des Leuchtens einer LED empfangen kann. Selbstverständlich ließe sich diese Aufgabe auch mit viel komplexeren Webservern wie Apache oder gar Lighttpd erledigen, doch das würde für die Anforderungen der Mehrzahl der Benutzer weit über das Ziel hinausschießen. Wir setzen hier stattdessen Flask ein, ein Webframework in Python, das unsere Anforderungen über das Web verarbeitet. Da alles in Python geschrieben ist, können wir alles Nötige innerhalb eines einzigen Skripts bewerkstelligen.

Haben Sie eine aktuelle Fassung von Raspbian installiert, sollte Flask schon darin enthalten sein. Falls nicht, können Sie es über `pip`, das Paketverwaltungsprogramm von Python, folgendermaßen installieren:

```
sudo apt-get install python-pip
```

Anschließend können Sie *Flask* mit `pip` installieren:

```
sudo pip install flask
```

Außerdem kommen die Bibliotheken `requests` (bereits installiert) und `pigpio` (Installation in Kap. 7 erklärt) zum Einsatz. Zuerst müssen Sie das folgende Python-Skript als Datei auf Ihrem Raspberry Pi speichern. Dieses Skript wird dann einen Webserver starten und auf Anfragen (Requests) warten, um die Website *…/licht_an* aufzurufen. Kommt die Anfrage, wird eine LED ein- oder ausgeschaltet. Gleichzeitig wartet das Skript auf das Drücken eines Knopfes (bzw. Tasters), um eine ausgehende Anfrage an IFTTT zu senden, das wiederum die E-Mail auslöst, die wir vorher eingerichtet haben:

```
#!/usr/bin/python

## Importieren der Bibliotheken
from flask import Flask
import requests
import pigpio
import time

## Variablendefinition
app = Flask(__name__)
event = "knopfdruck" #Eventname bei IFTTT
key = "Hier steht Ihr Schlüssel" #Key von IFTTT Webhooks
# Belegung der GPIO-Pins fuer LED und Schaltknopf
led = 18
button = 24
pi = pigpio.pi()
# Ausgangszustand der LED als aus
pi.set_mode(led, pigpio.OUTPUT)
pi.write(led, 0)
# Einrichtung des Schaltknopfes
pi.set_mode(button, pigpio.INPUT)
pi.set_pull_up_down(button, pigpio.PUD_UP)
#Schaltknopf entprellen
pi.set_glitch_filter(button, 100000)
```

```
#Diese Funktion wird bei Knopfdruck aufgerufen
def button_callback(gpio, level, tick):
   url = "https://maker.ifttt.com/trigger/%s/with/key/%s"
   r = requests.post(url % (event,key))
   print str(r.status_code) + ":" + r.text
   print "Event %s wurde ausgeloest." % event

# Feststellung, ob der Knopf gedrueckt wurde und
# Aufruf der Funktion button_callback()
b_detect = pi.callback(button, pigpio.FALLING_EDGE, button_callback)

# Bestimmung der URL zum Ausloesen von hallo()
@app.route("/")
def hallo():
   return "Hallo Welt! Warte auf Eingabe."

# Bestimmung der URL zum Ausloesen von licht_an()
@app.route("/licht_an")
def licht_an():
   if pi.read(led) == 0:
      pi.write(led, 1)
   else:
      pi.write(led, 0)
   return "Licht ist ein- bzw. ausgeschaltet."

# Starten des WebserverWebservers
if __name__=="__main__":
   app.run(host='0.0.0.0', port=80)
```

Abweichend vom hier angegebenen Skript müssen Sie zwei Dinge austauschen: Bei der Variablen `event` muss der derjenige Name stehen, mit dem Sie auch Ihr Event bei IFTTT bezeichnet haben (hier steht `knopfdruck`). Des Weiteren müssen Sie bei der Variablen `key` Ihren eigenen IFTTT-Schlüssel eintragen. Ist dies geschehen, geben Sie Ihrer Datei noch Ausführungsrechte (`chmod 755 Dateiname`), damit Sie sie als Skript laufen lassen können (`python Dateiname`). Läuft `pigpio` gerade nicht im Hintergrund, weil Sie z. B. zwischendurch Ihren Raspberry Pi neu gestartet haben, starten Sie es vorher noch (`sudo pigpiod &`).

Jetzt verbinden Sie die Anode (+) einer LED mit dem GPIO-Pin 18 und die Kathode (-) mit der Masse (GND) Ihres Raspberry Pi. Verbinden Sie dann noch einen Taster (Druckknopf) mit GPIO-Pin 24 und der Masse (GND) wie in Abbildung 9–11 zu sehen.

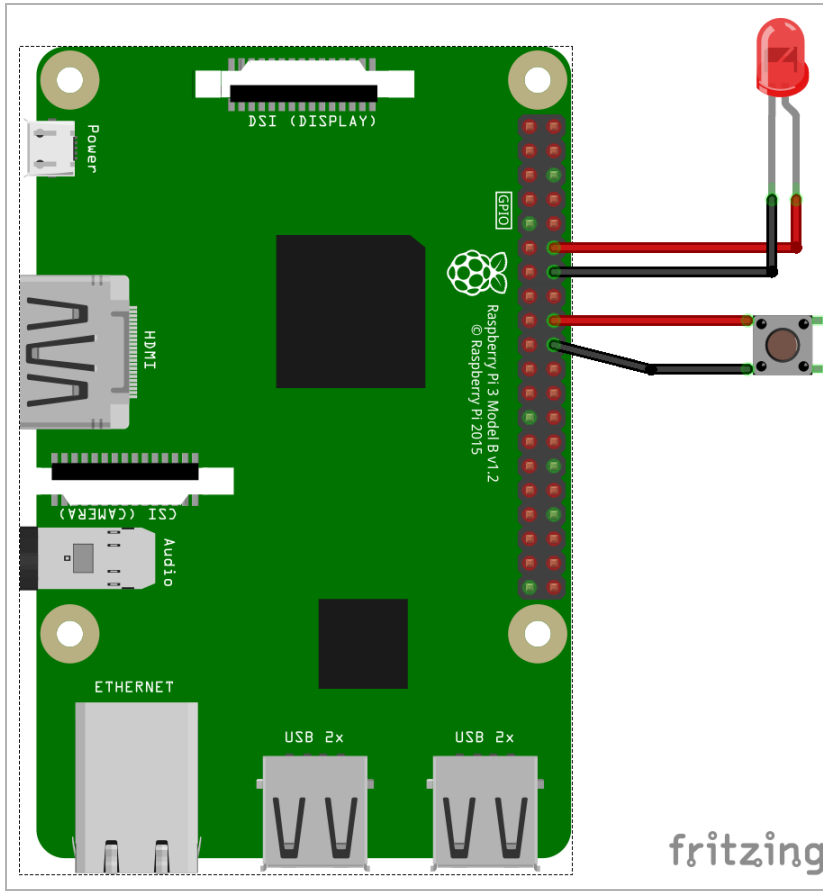


Abb. 9–11 Anschluss einer LED und eines Tasters an einen Raspberry Pi

Jetzt lassen Sie Ihr Skript als `sudo` laufen. Innerhalb Ihres Netzwerks sollten Sie jetzt von einem anderen Computer aus die IP-Adresse Ihres Raspberry Pi, gefolgt von */licht_an*, im Browser eingeben können (z.B. *http://xxx.xxx.xxx/licht_an*), auf der Seite */licht_an* landen und Ihre LED sollte dabei entweder an- oder ausgehen. Sobald Sie den Knopf am Taster drücken, sollten Sie noch eine E-Mail erhalten.

9.3 Probieren Sie es selbst

Versuchen Sie nun, ein IFTTT-Applet einzurichten, das die Seite */licht_an* auf Ihrem Raspberry Pi aufruft und die LED ein- oder ausschaltet. Vielleicht können Sie den Dienst Twitter dazu verwenden, die Seite jedes Mal dann aufzurufen, sobald jemand Sie in einem Tweet erwähnt. Über Facebook könnten Sie die Seite immer dann aufrufen lassen, wenn Sie in einer Bildunterschrift vorkommen.

Für den Teil »this« eines Applets in IFTTT können Sie jeden nur erdenklichen Dienst einspannen und dann mit dem Dienst Webhooks den Teil »that« einrichten.

Den Raspberry Pi von IFTTT aus erreichen

Stellen Sie für diese Art von Vorhaben sicher, dass IFTTT Ihren Raspberry Pi auch aus dem Internet erreichen kann. Damit der Datenverkehr von draußen auf Port 80 ankommt, kann es notwendig sein, auf Ihrem Router eine Portweiterleitung (port forwarding) auf die IP-Adresse Ihres Raspberry Pi einzurichten. Falls Sie bereits einen anderen Webserver darauf betreiben, müssen Sie eventuell sogar auf eine andere Portnummer ausweichen. Suchen Sie am besten im Internet nach Anleitungen für Portweiterleitungen mit Ihrem Routermodell.

9.4 Einen dedizierten Webserver einrichten

Es ist zwar sehr praktisch, wenn man mit Flask einen einfachen Webserver einrichten kann, doch manchmal braucht man eben doch einen größeren Funktionsumfang. Kleine dedizierte Webserver ermöglichen eine bessere Integration mit anderer Software und sind zudem zuverlässiger und sicherer als ein einfaches Python-Skript. Oftmals erleichtert ein Webserver das Einbinden von Cloud-Diensten wie IFTTT, wie wir gerade gesehen haben. Größere Webserver wie Apache können zwar auf dem Raspberry Pi betrieben werden, erfordern aber sehr viel mehr Ressourcen, sodass alles eventuell etwas langsamer läuft. Außerdem ist Apache mitunter ziemlich schwierig zu konfigurieren und zu verwalten. Ich empfehle stattdessen *Lighttpd*, das einfach zu installieren und für die meisten Projekte, die auf dem Raspberry Pi laufen werden, auch einfach einzurichten ist.

Installation

Sie können Lighttpd ganz einfach mit `apt-get` installieren:

```
sudo apt-get install lighttpd
```

Konfiguration für Python

Möchten Sie Python-CGI-Skripte (Common Gateway Interface) auf Ihrem Webserver laufen lassen, müssen Sie ein paar Änderungen vornehmen. Als Erstes öffnen Sie die Konfigurationsdatei in `nano`:

```
sudo nano /etc/lighttpd/lighttpd.conf
```

Jetzt fügen Sie am Ende dieser Datei folgende Zeilen ein:

```
$HTTP["url"] =~ "^/" {
   cgi.assign = (".py" => "/usr/bin/python")
}
```

Danach speichern und schließen Sie die Datei, indem Sie Ctrl-X, dann J und Enter drücken. Jetzt müssen Sie noch das Modul CGI für Lighttpd aktivieren, indem Sie folgendes Kommando eingeben:

```
sudo lighttpd-enable-mod cgi
```

Zum Abschluss starten Sie den Lighttpd-Server neu:

```
sudo service lighttpd restart
```

Testen Sie Lighttpd

Befinden Sie sich in einer Desktop-Umgebung, können Sie einen Browser öffnen und ihn zu *http://localhost* leiten. Allerdings finde ich es besser, einen Webserver von einem anderen Computer im Netzwerk aus zu testen, um wirklich sicherzugehen, dass alles funktioniert. Auf diesem geben Sie anstelle von *localhost* die IP-Adresse Ihres Raspberry Pi ein (siehe Abb. 9–12).

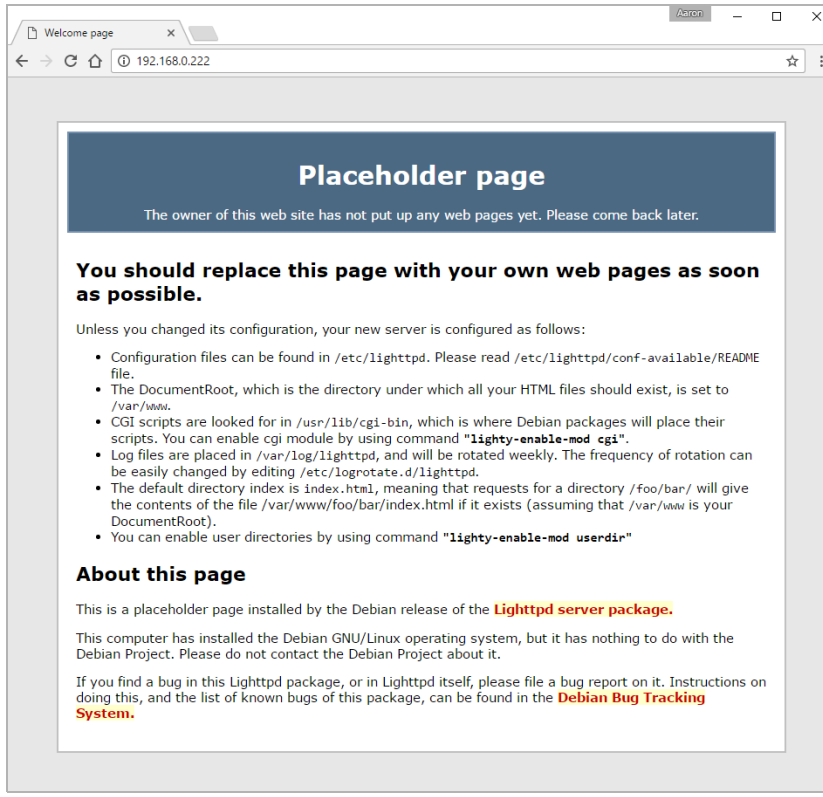


Abb. 9–12 Die unveränderte Website von Lighttpd

Ab hier können Sie Ihre eigenen Websites oder Python-Skripte entwickeln, mit denen Sie Ihre Projekte über das Internet zum Leben erwecken. Sie können die Konfigurationsdatei *lighttpd.conf* dahingehend ändern, dass sie als Startseite eine Pyhthon-Datei (z.B. *index.py*) akzeptiert, oder *index.html* so modifizieren, dass sie auf Ihr Python-Skript verweist, wie in diesem Fall:

```
<html>
<head>
   <meta http-equiv="refresh"
   content="0; url=/cgi-bin/index.py" />
<head>
<html>
```

Bei `url=` tragen Sie den relativen Pfad zu Ihrem Skript ein. Um mehr über das Programmieren von Python-Skripten für die Arbeit mit Webserver zu erfahren, schauen Sie sich das Modul CGI genauer an (*https://docs.python.org/2/library/cgi.html*). Beachten Sie dabei, dass */var/www/html* das voreingestellte Verzeichnis ist, in dem der Webserver nach Dateien sucht. Für das eben skizzierte Beispiel wäre der Ort für *index.py* also */var/www/html/cgi-bin/index.py*. Außerdem läuft der Webserver unter dem Benutzer »www-data«. Deshalb empfiehlt es sich, das Besitzrecht jeder neuen Datei entsprechend zuzuweisen. Wie das geht, haben Sie in Kapitel 6 gelernt.

9.5 Setzen Sie Ihren eigenen Cloud-Dienst auf

Statt auf Cloud-Dienste im Internet zu setzen, können Sie Ihren eigenen auf Ihrem Raspberry Pi betreiben. Dadurch sind Sie in der Lage, Dateien innerhalb Ihres eigenen Netzwerkes zu speichern und zu teilen und das völlig ohne Datenfluss über das Internet. Es gibt viele Cloud-Lösungen, die auf dem Raspberry Pi laufen. Dazu gehören die populären Vertreter OwnCloud und NextCloud, die jedoch für ungeübte Anwender schwierig zu installieren sind. Stattdessen werde ich Ihnen zwei andere Cloud-Dienste für Ihren Raspberry Pi vorstellen, die viel einfacher einzurichten sind und weniger Ressourcen in Anspruch nehmen.

Nimbus

Bei Nimbus (*https://cloudnimbus.org*) handelt es sich um eine Cloud-Software, die speziell für den Raspberry Pi entwickelt wurde. Zurzeit befindet sich diese Software noch in der Betaversion, ist aber recht leistungsfähig und für den Heimgebrauch einsatzbereit. Für Windows gibt es einen Client, der Ihnen Ihre Dateien synchronisiert. Als Erstes müssen Sie ein Verzeichnis für die Nimbus-Dateien anlegen und danach dort hinein wechseln:

```
mkdir /home/pi/nimbus
cd /home/pi/nimbus
```

Jetzt können Sie die Software auf Ihr System mit `wget` herunterladen:

```
wget http://cloudnimbus.org/dist/0.6.2-BETA/nimbus-0.6.2-BETA.tar.gz
```

Diese Datei müssen Sie vor Gebrauch entpacken, was Sie in diesem Fall mit dem Befehl `tar` machen:

```
tar -zxvf nimbus-0.6.2-BETA.tar.gz
```

Nimbus enthält ein Skript zur Installation sämtlicher benötigter Software, die eventuell noch nicht auf Ihrem Raspberry Pi vorhanden ist. Lassen Sie dieses Skript über die Kommandozeile als `sudo` laufen:

```
sudo ./install_helper_programs.sh
```

Jetzt starten Sie Nimbus mit dem Skript *nimbus.sh*:

```
./nimbus.sh start
```

Im Grundzustand verwendet Nimbus den Port 8080, sodass Sie ihn im Browser einer Desktop-Umgebung mit *http://localhost:8080* erreichen. Von einem anderen Computer aus kommen Sie mit dem Browser über die IP-Adresse Ihres Raspberry Pi, gefolgt von :8080, auf Ihre Nimbus-Cloud (siehe Abb. 9–13).

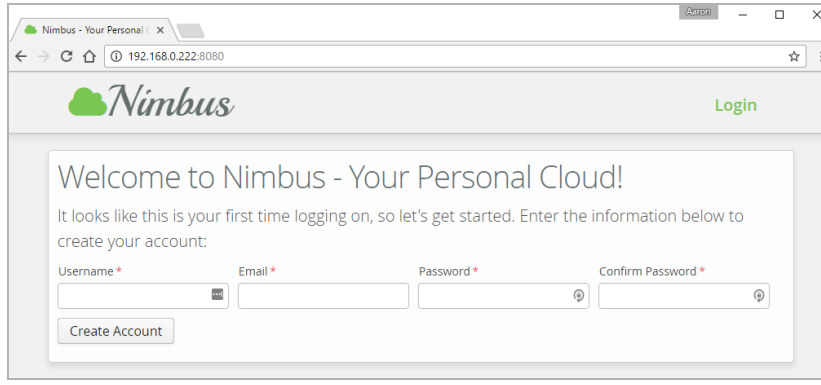


Abb. 9–13 Die Willkommensseite beim ersten Start von Nimbus

Beim ersten Zugriff auf Ihre Nimbus-Homepage sollen Sie ein Benutzerkonto anlegen. Sobald dies geschehen ist, können Sie sich in den Dienst einloggen und mit dem Laden und Weitergeben von Dateien loslegen (siehe Abb. 9–14).

Läuft alles nach Wunsch und möchten Sie Nimbus bei jedem Hochfahren Ihres Raspberry Pi starten, können Sie das mitgelieferte Skript als Dienst auf Ihrem System einrichten:

```
sudo ./add_to_startup_programs.sh
```

Sie können sogar eine externe USB-Festplatte anschließen, um den verfügbaren Speicherplatz zu erweitern. Wie das genau funktioniert, erfahren Sie auf der Website von Nimbus (*http://cloudnimbus.org*).

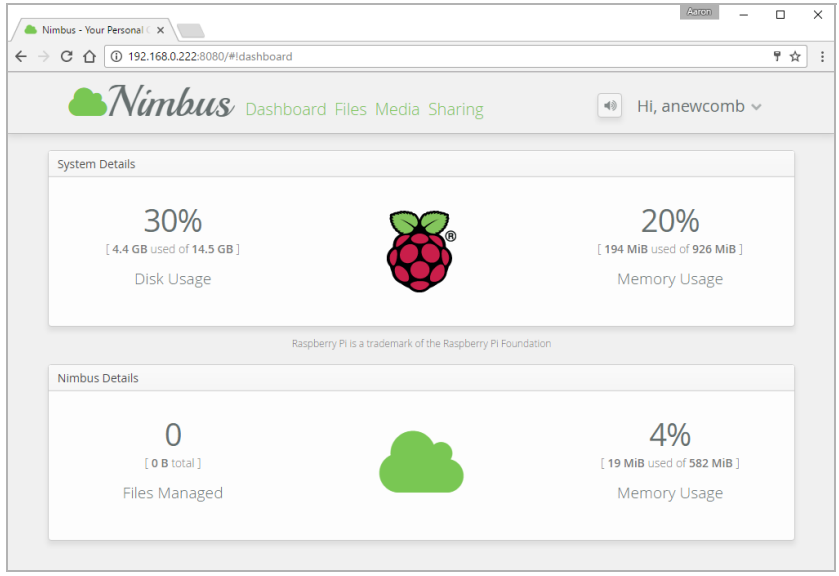


Abb. 9–14 Das Dashboard von Nimbus

Tonido

Bei Tonido handelt es sich um einen weiteren Cloud-Speicherservice, den Sie auf Ihrem Raspberry Pi laufen lassen können. Auch er wurde für SBCs entwickelt und existiert bereits eine ganze Weile, wodurch die Software inzwischen ausgereift ist. Mit Tonido können Sie Dateien speichern, teilen und streamen. Passende Clients gibt es für Windows, macOS, Linux, Android und iOS. Auch wenn die öffentliche Tonido-Seite weder Ihr Passwort noch irgendwelche Dateien von Ihnen speichert, stellt sie doch eine Verbindung zu Ihrem privaten Server her, auch wenn Sie sich gerade nicht in Ihrem Netzwerk befinden. Tonido ist für Einzelbenutzer ausgelegt, ermöglicht aber die Einrichtung von Gastzugängen, um anderen Zugriff auf bestimmte Dateien zu erlauben. Die Installation ähnelt sehr der von Nimbus. Auch hier müssen Sie zuerst ein Verzeichnis für die Tonido-Dateien anlegen, in das Sie anschließend wechseln:

```
mkdir /home/pi/tonido
cd /home/pi/tonido
```

Jetzt können Sie die Software mit `wget` herunterladen:

```
wget http://patch.codelathe.com/tonido/live/installer/armv6l-rpi/
tonido.tar.gz
```

Vor Gebrauch müssen Sie diese Datei mit dem Befehl `tar` entpacken:

```
tar -zxvf tonido.tar.gz
```

Starten Sie Tonido jetzt durch das Skript *tonido.sh*:

```
./tonido.sh start
```

In der Standardeinstellung verwendet Tonido Port 10001, sodass Sie in einer Desktop-Umgebung mit einem Browser über *http://localhost:10001* auf Tonido kommen. Von einem anderen Computer aus, erreichen Sie Tonido mit dem Browser über die IP-Adresse Ihres Raspberry Pi, gefolgt von :10001 (siehe Abb. 9–15).

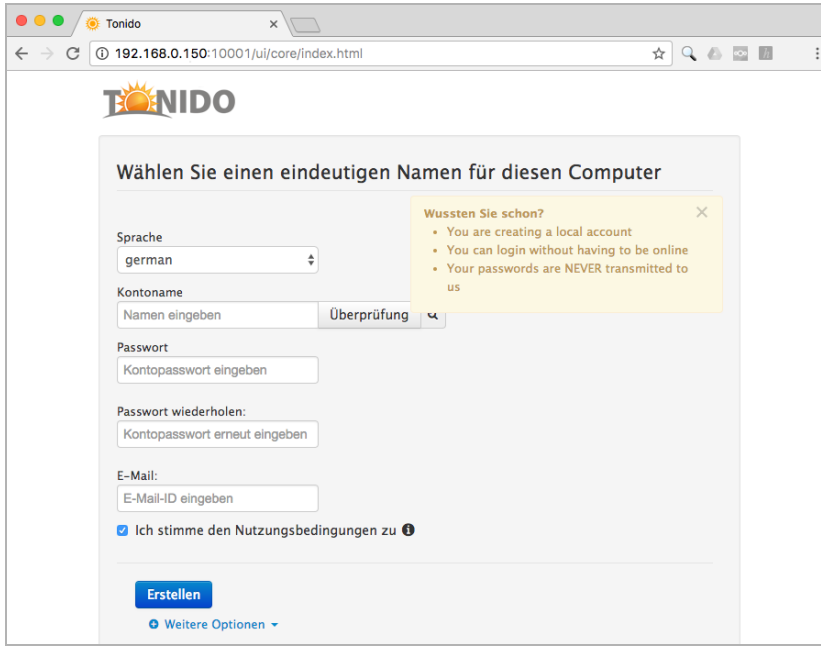


Abb. 9–15 Die Willkommensseite von Tonido beim ersten Start

Nehmen Sie die geforderten Eintragungen vor, um Ihr Benutzerkonto anzulegen. Dadurch wird gleichzeitig eine öffentlich zugängliche Webadresse erzeugt, mit der Sie leichter auf Ihren Tonido-Server gelangen. Sobald Sie angemeldet sind, können Sie Dateien hochladen und weitergeben (siehe Abb. 9–16).

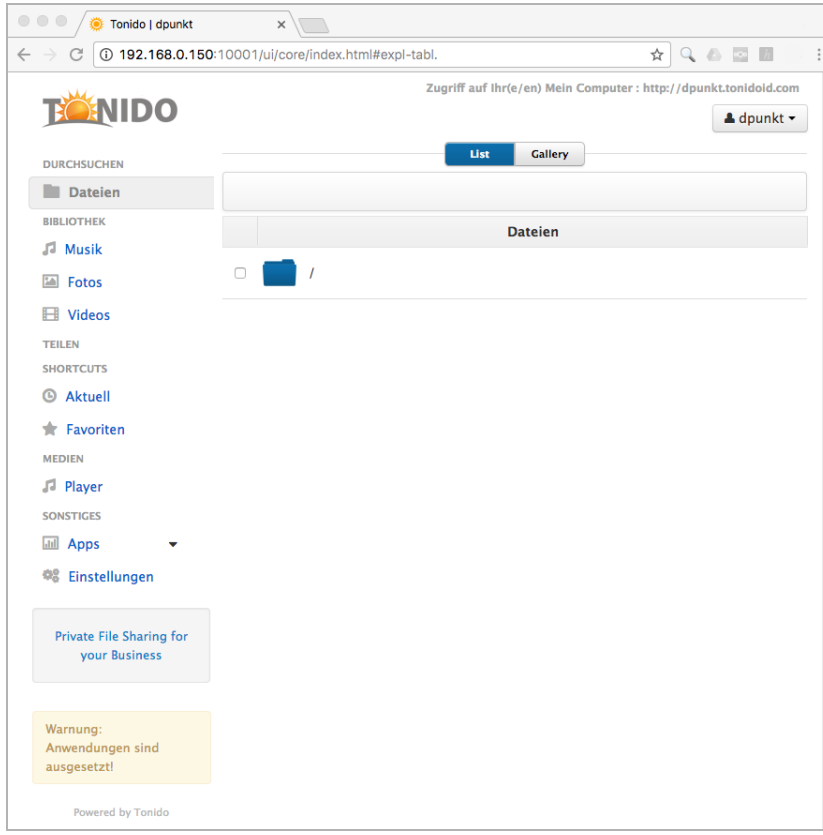


Abb. 9–16 Benutzer-Homepage bei Tonido

Beachten Sie, dass einige der mitgelieferten Anwendungen, mit denen man suchen oder von mobilen Apps zugreifen kann, sich zunächst im »gesperrten« Zustand befinden können. Um zu den entsprechenden Einstellungen zu gelangen, klicken Sie auf den Warnhinweis auf der Benutzer-Homepage, der in der Abbildung 9–16 blau umrandet ist. Klicken Sie dann auf jede einzelne Anwendung und wählen Sie im Popup-Fenster *Fortsetzen*. Der Button *Alle Anwendungen wieder aufnehmen* funktionierte bei mir nicht.

Um Tonido beim Hochfahren Ihres Raspberry Pi automatisch zu starten, können Sie den Startbefehl in Ihre Datei *rc.local* eintragen, wie in Kapitel 6 demonstriert.

9.6 Warum dies für Maker wichtig ist

Das Wissen um die Einrichtung und Verwendung von Cloud-Diensten kann einem Maker eine ganz neue Welt voller Möglichkeiten eröffnen. Sie können dadurch nicht nur Dateien übertragen und speichern, sondern Ihr Projekt auch über das Internet steuern. Mit zunehmender Verbreitung von Cloud-basierten Systemen werden auch Sie immer mehr eigene IoT-Geräte mit Ihrem Raspberry Pi bauen können.

10 Virtueller Raspberry Pi

Maker haben immer viel um die Ohren und nicht immer einen Raspberry Pi vor sich, mit dem sie experimentieren können. Vielleicht sind sie sich aber auch noch nicht sicher, ob sie die Hürde Linux nehmen wollen und die Anschaffung eines Raspberry Pi oder eines anderen SBC wagen sollen. Für solche Fälle habe ich einen virtuellen Raspberry Pi erzeugt, der unter VirtualBox von Oracle läuft und in dem Sie Linux erkunden und Dinge ausprobieren können. Viele der Bilder und Übungen in diesem Buch sind sogar mit meinem virtuellen Raspberry Pi entstanden. In diesem Kapitel erkläre ich, wie man eine vollständige Raspberry-Pi-Umgebung über Windows, macOS oder Linux anlegen kann, wenn man das echte System nicht zur Hand hat.

Bevor Sie loslegen, sollten Sie noch über ein paar Dinge der virtuellen Umgebung Bescheid wissen und diese berücksichtigen. Erstens haben Sie in dieser Umgebung keine realen GPIO-Pins, die Sie ansteuern könnten. Daher gibt es auch keinerlei Möglichkeiten, echte Bauteile anzuschließen. Sollten Sie etwa daran gedacht haben, Code für I^2C- oder SPI-Bauteile testen zu wollen, geht das einfach nicht. Zweitens ist der Arbeitsspeicher bei dieser Emulationsumgebung auf 256 MB begrenzt. Drittens wird das Raspberry-Pi-Image gewissermaßen doppelt virtualisiert: Zum einen gibt es das zugrunde liegende Gastbetriebssystem Linux in Form einer Debian-Distribution. Innerhalb dieser läuft aber zusätzlich QEMU, um die ARM-Prozessorumgebung des Raspberry Pi zu emulieren. Dieses Konstrukt scheint keine wesentlichen Nachteile zu haben, mit Ausnahme des etwas umständlichen Zugangs ins Internet, doch dieses Problem habe ich für Sie gelöst.

10.1 Systemanforderungen

- Ein Computer mit einer aktuellen Version von Windows, macOS oder Linux
- 15 GB freier Festplattenspeicherplatz

10.2 Installation

Laden Sie sich die aktuelle Version der VirtualBox von Oracle herunter (*virtualbox.org*) und installieren Sie sie.

Laden Sie sich meine Image-Datei für den virtuellen Raspberry Pi über Google Drive (*http://bit.ly/2o98Dg1*) herunter und speichern Sie sie an einem Ort, den Sie leicht wiederfinden. Es handelt sich dabei um eine Datei von etwa 5 GB, sodass der Download eine Weile dauern kann.

Starten Sie nun VirtualBox und gehen Sie auf *Datei/Appliance importieren…* (siehe Abb. 10–1).

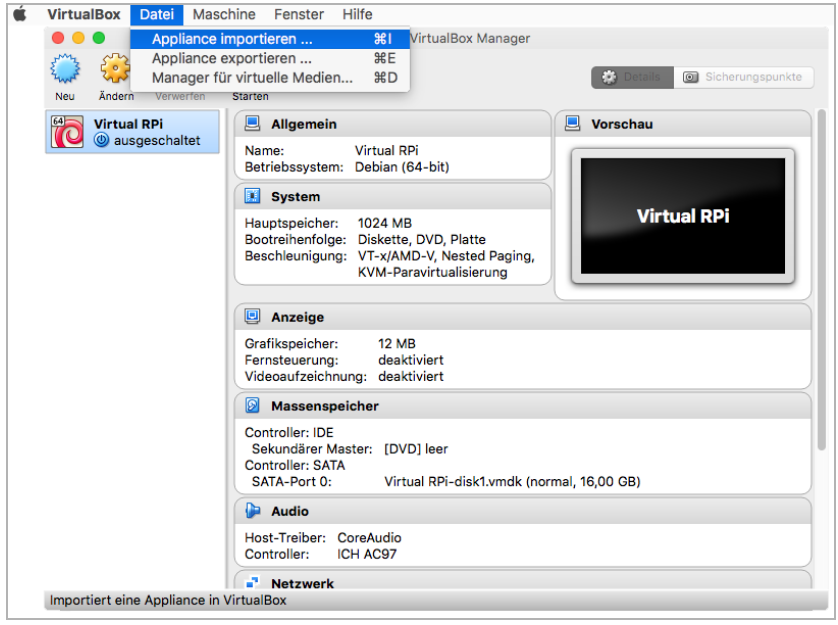


Abb. 10–1 Import einer vorkonfigurierten Image-Datei für VirtualBox

Im Datei-Dialogfenster wählen Sie die von Google Drive heruntergeladene Image-Datei aus. Dadurch wird das System Virtual RPi für Sie in VirtualBox importiert und konfiguriert.

10.3 Einsatz

Um die virtuelle Maschine Virtual RPi zu starten, wählen Sie sie in der linken Liste aus und klicken auf den Button *Starten*. Das System läuft dann in einem separaten Fenster. Ihnen wird auffallen, dass zweimal hintereinander Linux-Bootmeldungen erscheinen, was daran liegt, dass der Raspberry Pi, wie bereits erwähnt, innerhalb einer virtuellen Debian-Installation emuliert wird. Sobald der Boot-Vorgang beendet ist, sehen Sie den Prompt in der Kommandozeile des virtuellen Raspberry Pi (siehe Abb. 10–2).

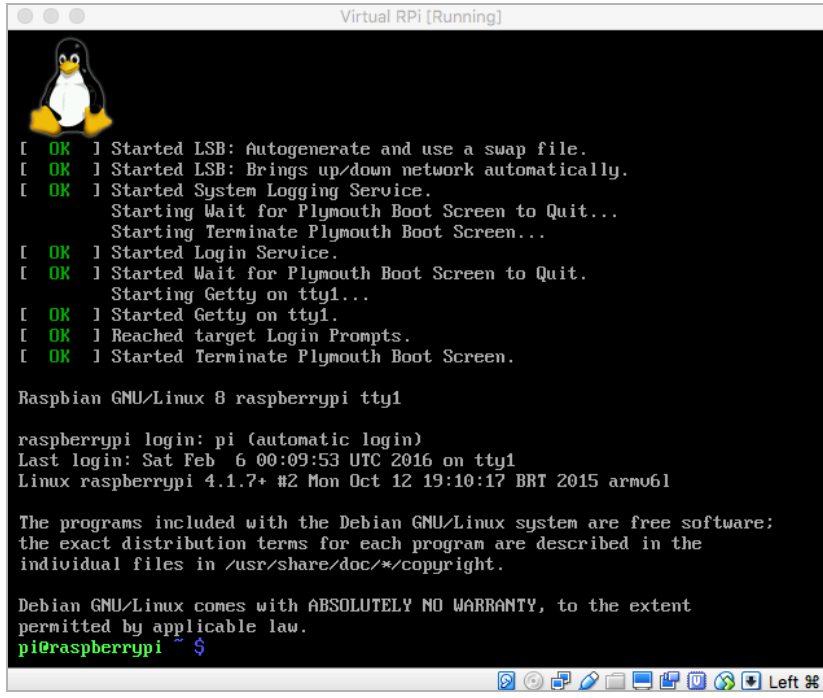


Abb. 10–2 Das Terminal des Raspberry-Pi-Image nach dem Hochfahren

Die Desktop-Umgebung startet nicht automatisch, lässt sich aber über den Befehl `startx` aufrufen.

Um die virtuelle Maschine Virtual RPi herunterzufahren, müssen Sie zunächst den Befehl für einen Neustart von Virtual RPi eingeben. Erst dann fahren Sie die Debian-Umgebung herunter. Um den virtuellen Raspberry Pi herunterzufahren, geben Sie also Folgendes ein:

```
sudo shutdown -r now
```

Dies ist zwar normalerweise der Befehl zum Neustart, der allerdings nicht stattfindet, weshalb man sich anschließend auf dem Desktop von Debian befindet. Um nun auch Debian herunterzufahren, klicken Sie auf das Menü-Icon unten links, auf *Logout* und anschließend auf *Shutdown* (siehe Abb. 10–3).

Nicht herunterfahren

Bei diesem virtuellen System sollte man nicht shutdown -h verwenden, da man auf diese Weise nicht zum Debian-Desktop zurückkehrt. Falls man es doch gemacht hat, kann man Ctrl-Alt-F drücken, um aus dem Vollbildmodus herauszukommen und das Emulatorfenster zu schließen.

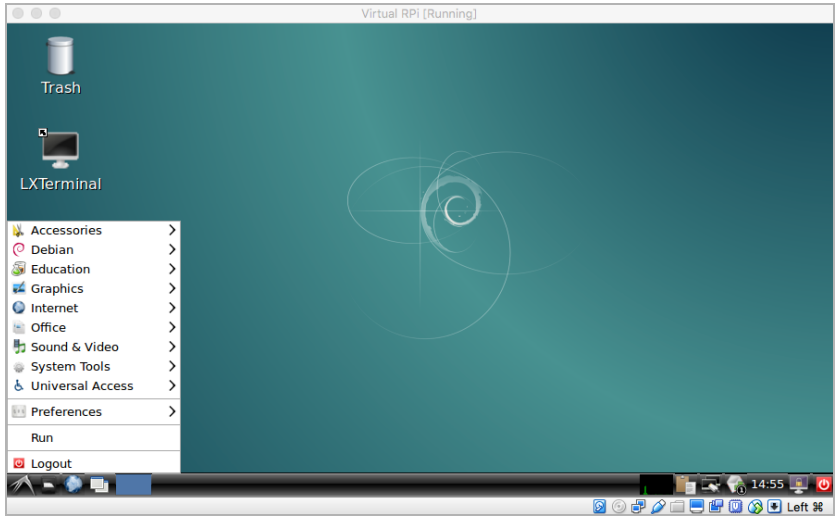


Abb. 10–3 Der Debian-Desktop nach dem Rebooten des Raspberry-Pi-Image

Wenn Sie im Raspberry-Pi-Image feststecken und zum Debian-Desktop zurückkehren möchten, können Sie Ctrl-Alt-F drücken, um aus dem Vollbildmodus herauszukommen. Um in den Mausmodus zu wechseln, drücken Sie Ctrl-Alt, wenn Sie sich nicht im Vollbildmodus befinden.

10.4 Warum dies für Maker wichtig ist

Einfach weil es geht! Manchmal macht es Spaß, neue Dinge auszuprobieren, um zu sehen, wie sie funktionieren. Auch wenn durch den Mangel an Arbeitsspeicher und realer GPIO-Pins diese Methode nicht gerade die bevorzugte Art und Weise darstellt, den Raspberry Pi kennenzulernen, so kann es doch Spaß machen, etwas herumzuprobieren. Sie können mit dieser virtuellen Installation einer Raspbian-Distribution auch die Syntax eines Programms überprüfen, Softwarequellen bestätigen und auch, wenn Sie gerade kein echtes System zur Hand haben, Material für Ihren Blog-Post mit Anleitungen zum Thema generieren. Darüber hinaus lernen Sie dabei etwas über Virtualisierungen oder sogar, wie man damit ein Buch schreibt!

A Wissenswertes über Linux

In diesem Anhang geht es um die Geschichte des Betriebssystems Linux sowie um einige grundsätzliche Fragen, die sich Maker häufig stellen, wenn sie überlegen, Linux für ihre Projekte zu verwenden. Linux wird auf der ganzen Welt auf den unterschiedlichsten Plattformen eingesetzt. In der Geschäftswelt ist es das bevorzugte Betriebssystem für Rechenzentren. Es liegt zudem allen Betriebssystemen von Android-Smartphones zugrunde, wodurch es zahlenmäßig das am häufigste verwendete Betriebssystem der Welt geworden ist. Darüber hinaus tut Linux in Netzwerkroutern, Kartenzahlsystemen, medizinischen Geräten, Set-Top-Boxen und digitalen Fernsehern seinen Dienst. Hinzu kommen natürlich noch die Maker, die das Betriebssystem auf SBCs wie Raspberry Pi, BeagleBone Black, C.H.I.P., Pine 64 und Onion-Plattformen einsetzen. Da es so viele verschiedene Linux-Versionen gibt, kann man mit ihnen auch ganz einfach alte Hardware wieder zum Leben erwecken.

A.1 Kurze Geschichte des Originalbetriebssystems der Maker

Es ist sicherlich nicht übertrieben, zu behaupten, dass Linux selbst als das ultimative Maker-Projekt betrachtet werden kann. Bevor es Linux gab, bestand der einfachste und gangbarste Weg, ein UNIX-ähnliches Betriebssystem zu nutzen, auf kommerzielle Produkte zurückzugreifen. Unix wurde von Ken Thompson und Dennis Ritchie während deren Zeit in den Bell Labs (heute AT&T) entwickelt und kam 1970 auf den Markt. Damals wurden Betriebssysteme noch für die jeweiligen Hardwareplattformen entwickelt. Der heute übliche Computereinsatz steckte noch in den Kinderschuhen, sodass man Software in der Regel nicht von einer Plattform auf eine andere übertragen konnte. Im Zuge der steigenden Beliebtheit von Unix entstanden ähnliche Betriebssysteme, darunter BSD (Berkeley Software Distribution) und SunOS.

Während der folgenden 20 Jahre mussten die Firmen, um Unix in ihren Rechenzentren betreiben zu können, Lizenzen von AT&T, HP, IBM, Sun Microsystems oder einem anderen Anbieter kaufen. In den meisten Fällen haben diese Firmen ihre Betriebssysteme gewissermaßen im Vakuum entwickelt. Wie Sie sich vorstellen können, waren sie darauf bedacht, die Früchte ihrer Entwicklungsarbeit vor dem Zugriff der Konkurrenz zu schützen. Doch selbst wenn der Programmcode stetig verbessert und Fehler bereinigt wurden, so blieb der Kunde dem Anbieter ausgeliefert, was die Priorität der Fehlerbereinigungen und die Erweiterung des Funktionsumfangs anging.

Wo blieben da die Maker? Im Grunde ziemlich außen vor, denn obwohl das Betriebssystem Unix sehr leistungsfähig und mit der Zeit auch immer vielseitiger wurde, so blieben die Softwarelizenzen für die meisten Einzelnutzer unerschwinglich. Die Hardware war zudem sehr kostspielig, sodass jemand, der Unix für ein Projekt nutzen wollte, Serverzeit von einer Firma mieten musste, die das System vorhielt. Stellen Sie sich nur einmal vor, ein Raspberry Pi, der heute 40 € kostet, würde 40.000 € kosten und Sie müssten für das Raspbian noch 10.000 € extra für die Ehre zahlen, es überhaupt nutzen zu dürfen. Oder wie wäre es, wenn Ihnen die Raspberry Pi Foundation 40 € pro Stunde Nutzungszeit berechnen würde?

In dieser Zeit entstand das Konzept der Mehrbenutzersysteme und des *Timesharings*, sodass teure Hardwareressourcen einerseits besser genutzt werden konnten und sich andererseits die hohen Anschaffungskosten der Firmen besser amortisierten. Hatte man in jener Zeit das Glück, als Student an einer Universität eingeschrieben zu sein, die Timesharing-Systeme implementiert hatte, konnte man diese kostenlos nutzen. Dies war die Ausgangslage für neue Ideen.

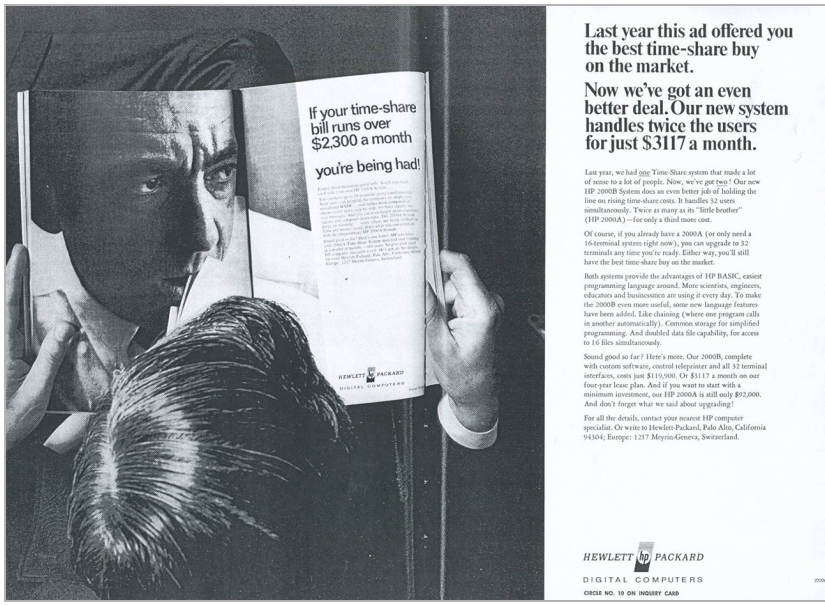


Abb. A–1 Anzeige für Timesharing im Jahr 1970 (Bild aus dem HP Computer Museum)

Als Richard Stallman 1983 das Projekt GNU (GNU's Not Unix) begann, war er wissenschaftlicher Mitarbeiter im Labor für künstliche Intelligenz am Massachusetts Institute of Technology (MIT). Der beschränkte Computerzugang im Labor und der Trend zu proprietärer Software, die sich weder verändern noch weitergeben ließ, frustrierten ihn. Er hatte beispielsweise die Software eines Laserdruckers so modifiziert, dass sie dem Benutzer mitteilte, dass sein Druckauftrag erledigt war. Als dann ein neuer Drucker mit proprietärer Software angeschafft wurde, konnte er seine Änderungen nicht mehr umsetzen, sodass die Mitarbeiter zum Teil mehrfach die Treppe auf- und abgehen mussten, um nachzuschauen, ob deren Druckauftrag schon durch war.

Das Ziel dieses GNU-Projekts besteht bis heute darin, den Menschen Zugang zu Software zu verschaffen, die frei verwendet, weitergegeben und verändert werden darf. Möchte man also einen Fehler beheben, kann man dies einfach tun, ohne jemand vorher fragen oder den Code hacken zu müssen. Man kann seine Änderungen außerdem an andere weitergeben, damit diese daraus lernen und sie in ihre eigene Software integrieren können. Das GNU-Projekt entwickelte dann eigene Versionen der Programme, die man bei Unix für die Softwareentwicklung braucht, wie zum

Beispiel einen Editor (Emacs), einen Compiler (GCC) und einen Debugger (GNU Debugger) sowie die bekannten Tools ls, grep und make. In vielerlei Hinsicht waren Stallman und die anderen Mitstreiter im GNU-Projekt selbst Maker. Genau wie die Maker von heute waren auch sie der Ansicht, dass man seine Projekte selbst machen und mit denen teilen sollte, die ähnliche Interessen haben.

Um diesen Ansichten eine Form zu geben und die Software, die er und andere entwickelt hatten, zugänglich zu machen, kam Stallman auf die Idee der GNU Public Licence (GPL). Vielleicht ist ihnen dieses Lizenzmodell schon auf Ihrem Computer oder Smartphone begegnet. Große Anteile von Android, Wordpress, GIMP, VLC Media Player und auch der Linux-Kernel, auf den wir gleich noch zu sprechen kommen, unterliegen der GPL. Durch GPL ist gesichert, dass Sie die Software verändern, kopieren und weitergeben dürfen. Das heißt aber auch, dass die Software danach weiterhin der GPL unterliegt. Dadurch wird gewährleistet, dass auch allen nachfolgenden Benutzern die gleichen Rechte gewährt werden, ganz gleich, wie oft die Software zwischenzeitlich verändert wurde.

Diese und andere zu dieser Zeit geschaffenen Softwarelizenzmodelle sind mittlerweile unter dem Begriff *Open-Source-Software* (quelloffene Software) bekannt. Während einer proprietären, also geschlossenen Software Beschränkungen auferlegt sind, die die Entwickler oder die Firmen schützen sollen, hat Open-Source-Software das Ziel, sowohl den Entwickler zu schützen als auch dem Benutzer Rechte einzuräumen. Auch wenn es noch viele andere quelloffene Lizenzen gibt, so ist GPL immer noch die beliebteste. Die Einstellung, dass Software frei zu verändern und zu verbessern sein sollte, hat sich zum Teil auch auf die Hardwareentwicklung ausgewirkt. Einige der populärsten Plattformen, die Maker für ihre Projekte einsetzen, wie den Arduino, aber auch das RepRap Project, Lulzbot, BeagleBone Back sowie große Teile des Raspberry Pi, werden auch als *Open-Source-Hardware* bezeichnet. Praktisch bedeutet dies, dass deren Spezifikationen veröffentlicht sind und die Leute sie völlig frei verändern können.

A.2 Probieren Sie es selbst

Auch wenn es nicht zwingend gefordert wird, dass die Software kostenlos ist, haben doch viele kostenlose Programme eine Open-Source-Lizenz. Öffnen Sie doch einfach einmal ein paar Ihrer Lieblingsprogramme und schauen Sie im Fenster Über…/About… (in der Menüleiste unter dem Programmnamen oder unter *Hilfe*) nach, welche Lizenz Ihre Software verwendet. Alternativ können Sie diese Information auch im Internet bekommen. Sie werden überrascht sein, wie hoch der Anteil an Software ist, der auf Open-Source-Lizenzen beruht. Verglichen mit den frühen Tagen der Computernutzung hat sich hier viel getan.

A.3 Linus Torvalds

Auch wenn die Arbeit an den Programmen des GNU-Projekts gut voranging, fehlte damals immer noch ein vernünftiger Unix-Kernel. 1991 hatte sich ein Informatikstudent der Universität Helsinki namens Linus Torvalds einen neuen Computer und einen Unix-Clone, Minix, gekauft. Der Quellcode von Minix war zwar einsehbar, jedoch durfte man ihn nicht verändern, geschweige denn weitergeben. Wie alle guten Maker, fasste Torvalds dies als Herausforderung auf. Seiner Meinung nach sollte es ein frei verfügbares, Unix-ähnliches Betriebssystem geben, das auf den damals noch aktuellen x86-Computerplattformen laufen sollte. Hier ist sein Beitrag im Minix-Forum, in dem er sein neues Betriebssystem ankündigte:

> *Hallo an alle, die Minix benutzen –*
>
> *Ich arbeite gerade an einem (kostenlosen) Betriebssystem (nur ein Hobby, nicht so professionell) für 386(486)-AT-Clones. Ich sitze seit April dran und werde so langsam fertig. Ich hätte gerne etwas Feedback über die Dinge, die die Leute an Minix mögen/nicht mögen, da mein Betriebssystem ihm gewissermaßen gleichkommt (gleiche Anordnung des Dateisystems (aus praktischen Gründen) und andere Dinge mehr).*

Gerade habe ich bash(1.08) und gcc(1.40) portiert und es scheint alles zu funktionieren. Daraus folgt, dass ich in ein paar Monaten etwas Brauchbares habe, und ich möchte gerne wissen, welche Features am meisten gefragt sind. Alle Vorschläge sind willkommen, aber ich kann natürlich nicht versprechen, dass ich sie implementieren werde :-)

Linus (torvalds@kruuna.helsinki)

PS. Ja – es enthält keinen Minix-Code und hat einen multi-threaded fs. Es ist NICHT portierbar (verwendet 386 task switching etc.) und wird vermutlich niemals etwas anderes als AT-Festplatten unterstützen, da ich nichts anderes habe :-(.

– Linus Torvalds[1]

Dieses neue Betriebssystem wurde später als Linux bekannt, einer Wortschöpfung aus »Linus« und »Unix.« Es ist ganz interessant, dass es damals nur ein Projekt zum Spaß war. Torvalds dachte noch nicht daran, dass es eines Tages »professionell,« auf anderen Rechnern lauffähig sei oder ein großes Spektrum an Peripherie bedienen könnte. Meine Güte, wie sich die Dinge entwickelt haben. Heutzutage haben Firmen wie Red Hat große Business-Softwarepakete auf der Basis von Linux entwickelt. Darüber hinaus ist Linux heute auf den unterschiedlichsten Plattformen lauffähig, ganz zu schweigen, dass es auf Zehntausenden unterschiedlichsten alten wie neuen Geräten läuft. Oftmals finden sich sogar Open-Source-Treiber bereits im System, sodass sie nicht erst installiert werden müssen.

Linux wurde deswegen so populär, weil es nichts kostet, auf einfach zu pflegen, zu modifizieren und zu verbessern ist. Man kann es in dieser Hinsicht mit dem Arduino vergleichen, der in der Maker-Szene weit verbreitet ist. Arduinos sind sehr preisgünstig, einfach über das Internet zu beschaffen und wenn einem das Layout des Arduino Uno nicht gefällt, kann man seinen eigenen von Grund auf selbst bauen. Im Zuge der wachsenden Popularität entstanden dank der offenen Kultur der Entwicklung eine Vielzahl von Communities. Torvalds erhielt also nicht nur Anregungen zur Änderung des Codes, sondern auch fertigen Code von Programmierern, die das Betriebssystem ausprobierten. Da der Quellcode nun für

1) Linus Torvalds. »What Would You Like to See Most in Minix?« Usenet group *comp.os.minix*, August 25, 1991.

jeden einsehbar war, waren die Entwicklungszyklen für neue Funktionen und Fehlerbereinigungen viel kürzer als das bei dem proprietären Unix-System der Fall war. Aufgrund der vielen eingereichten Änderungsvorschlägen wurde ein Gremium geschaffen, das diese begutachtete, bevor sie in die neuen Linux-Versionen übernommen wurden.

Linux wird heute im Prinzip noch genau so weiterentwickelt. Torvalds und eine Handvoll anderer sogenannter »Kernel-Maintainer« wachen über diesen Prozess und auch heute steht er im Prinzip jedem offen, der den Code verbessern oder neue Funktionen hinzufügen will. Sie haben keinen Treiber für Ihr neues WLAN-Modul? Dann dürfen Sie selbst einen schreiben. Sie haben einen Fehler beim Hochfahren des Systems entdeckt? Sie können einen entsprechenden Patch dazu einreichen. Sie merken: Linux ist *das* Betriebssystem für Maker.

A.4 Der Linux-Kernel

Das Stichwort Linux-Kernel ist schon ein paarmal gefallen und je mehr Sie über Linux für Ihre Projekte erfahren, wird Ihnen klar, warum in Anleitungen und Forenbeiträgen immer wieder darauf Bezug genommen wird. Wie der Kern einer Nuss oder einer Frucht ist der Kernel das zentrale Programm, das die Funktionen eines Betriebssystems verwaltet. Es ist normalerweise das erste Programm, das beim Hochfahren eines Systems läuft. Der Kernel sitzt zwischen den Anwendungen und den Hardwarekomponenten und regelt, wie und wann auf diese Komponenten zugegriffen wird. Ohne Kernel könnten diese Anwendungen nicht laufen, da sie nicht wüssten, wie sie Zugang zu CPU, Arbeitsspeicher, Festplattenspeicher und anderer Hardware, aus der der Computer besteht, bekämen. Zudem fungiert der Kernel als eine Art Verkehrspolizist, der verhindert, dass die Anwendungen »zusammenstoßen«, da viele von ihnen dieselben Ressourcen beanspruchen. In modernen Betriebssystemen laufen mehrere Tausend Programme gleichzeitig. Gäbe es keinen Kernel, könnte man allenfalls ein einziges Programm laufen lassen, ohne dass es zu Problemen käme. Wie Sie sich daher denken können, ist der Kernel der wichtigste Teil des Betriebssystems. Weil er so wichtig ist, wird er in einem abgesicherten Teil des Systemspeichers geladen, sodass er vor unerwünschten Änderungen geschützt ist.

Vom Linux-Kernel gibt es viele Versionen. Er wird regelmäßig von den Kernel-Maintainern aktualisiert und kann entsprechend den Bedürfnissen des Systems so angepasst werden, dass alle oder nur bestimmte Bestandteile genutzt werden. Muss Ihr System beispielsweise mit wenig Speicherplatz auskommen, kann ein Entwickler alle nicht dringend benötigten Teile entfernen, um den Kernel zu verkleinern. Ist Ihr System nicht besonders schnell, kann ein Entwickler die Teile herausnehmen, die nicht die ganze Zeit laufen müssen, damit der Kernel effizienter arbeitet. Einzelne Benutzer gehen sogar so weit, dass sie sich eigene Versionen des Kernels schaffen, indem sie enthaltene Komponenten verändern oder bestimmte Parameter modifizieren, damit ihr System genau so funktioniert, wie sie es brauchen. Für den Maker ist das Kompilieren eines eigenen Linux-Kernels unnötig, da diese Arbeit bereits von denen geleistet wurde, die Rechner wie den Raspberry Pi auf den Markt gebracht haben.

Da sich der Linux-Kernel zu weiten Teilen anpassen lässt, enthält er oft die nötige Software, die die Hardwarekomponenten und die Peripherie für das Betriebssystem erkennbar machen. Diese Software sind die sogenannten *Treiber*, die, je nach Hersteller, proprietär oder quelloffen sind. Sind sie quelloffen, können sie in den Kernel übernommen werden, wodurch die Verwendung der entsprechenden Komponenten in Linux wesentlich erleichtert wird. Fast jede Maus oder Tastatur wird daher von Ihrem Linux-basierten System erkannt. Bei anderen Betriebssystemen wie etwa Microsoft Windows muss die Treibersoftware entweder aus dem Internet oder von der Festplatte geladen werden, bevor das jeweilige Gerät verwendet werden kann. Oftmals sind die Treiber für ältere Geräte schwer aufzutreiben oder es gibt für die aktuelle Version des Betriebssystems gar keine mehr. Dass dieser Fall eintritt, ist bei quelloffenen Treibern im Linux-Kernel sehr viel unwahrscheinlicher, da auf sie, wenn sie einmal geschrieben wurden, bei Bedarf immer wieder zugegriffen werden kann. Dies ist ein weiterer Grund, warum Open-Source-Software für Maker wichtig sein sollte, und ich komme später noch einmal darauf zurück.

A.5 Distributionen

Im Vorwort erwähnte ich, dass ich mich in erster Linie auf die Linux-Distribution Raspbian konzentrieren würde. Doch was ist eigentlich eine Distribution? Das Wort *Distribution* klingt danach, als ob etwas an viele Leute verteilt würde, was den Sachverhalt auch ziemlich gut trifft. Durch Hinzufügen und Verändern der diversen Programme kann man sich auf der Grundlage von Linux im Prinzip sein eigenes Betriebssystem zusammenstellen. Dieses lässt sich dann in ein einfach zu installierendes Format bringen, das sicherstellt, dass jede Installation exakt gleich ausfällt. Diese Zusammenstellung vorkonfigurierter Software nennt man eine Distribution. In der Praxis ist die Existenz einer Distribution vor allem dann wichtig, wenn man mehr als ein System zur gleichen Zeit installieren muss. Möchte man beispielsweise Linux auf 100 Servern installieren, müsste man ohne eine Distribution für jeden einzelnen Server jeweils den Kernel kompilieren, die Desktop-Umgebung auswählen, sämtliche Software installieren und jeden Dienst einzeln konfigurieren. Durch die Verwendung einer Distribution lässt sich dies drastisch vereinfachen, weil man auf jedem Server eine Sammlung vorkonfigurierter Software installieren kann und somit sichergestellt ist, dass auf jedem Server die gleiche Installation läuft.

Darüber hinaus ermöglichen Distributionen, dass Firmen und Einzelpersonen gewissermaßen ihre eigenen Geschmacksrichtungen von Linux in die Welt setzen. Sobald die Software steht, kommt noch ein Installationsprogramm obendrauf, damit die Installation leichter und schneller vonstattengeht. Anschließend werden sämtliche Dateien, aus denen das Betriebssystem besteht, noch in eine einzige Datei gepackt, damit man es leichter herunterladen kann. Wie beim Linux-Kernel auch werden die Distributionen von mehreren Leuten zusammen regelmäßig gewartet und aktualisiert, damit Fehler ausgemerzt werden und neue Funktionen hinzukommen.

Die Möglichkeit, seine eigene Distribution eines Betriebssystems zu erschaffen, gibt es eigentlich nur bei Linux. Um eine angepasste Version von Windows oder macOS zu bekommen, müsste man Microsoft bzw. Apple dazu bringen, dies für einen zu erledigen. Da Linux auf quelloffener Software beruht, kann man selbst entscheiden, welche Software man für welche Aufgabe in seinem Betriebssystem braucht. So wie man in seiner Werkstatt seinen Bestand an Werkzeugen selbst bestimmt, kann man die Linux-Distribution auswählen, die am besten den Anforderungen seines Projekts gerecht wird.

Zu den berühmten Vertretern unter den Distributionen gehören Linux Mint, Debian, Ubuntu, OpenSUSE und Arch. Einige Firmen wie Red Hat haben sowohl eine kommerzielle als auch eine freie Distribution (Fedora) herausgebracht. Einige Distributionen beruhen auf anderen. So wurden Ubuntu und Raspbian beispielsweise von Debian abgeleitet, wohingegen Linux Mint wiederum auf Ubuntu basiert. Es wurde also jeweils eine Distribution als Ausgangsbasis verwendet, dann Änderungen an Software und Aussehen vorgenommen und anschließend als selbstständiges System verteilt. Es gibt auch Distributionen, die für ganz bestimmte Aufgaben oder Hardware entwickelt wurden. So richtet sich Ubuntu Studio speziell an Medienschaffende im Bereich Audio, Video und Grafikdesign. Bei Tiny Core Linux handelt es sich um ein vollständiges Desktop-Betriebssystem, das nur 16 MB groß ist, komplett im Arbeitsspeicher läuft und sich von einem USB-Stick oder einer CD booten lässt. Auch bei GParted Live handelt es sich um eine Distribution, die von einem externen Datenträger gestartet wird, um zum Beispiel Probleme mit der Speicherhardware zu erkennen und zu beheben. Es gibt viele Distributionen, die auch auf dem Raspberry Pi laufen. Raspbian wurde speziell für den Raspberry Pi entwickelt und ist deshalb dort die meistverwendete Distribution. Außerdem wird sie durch die Raspberry Pi Foundation offiziell unterstützt. Zu den weiteren Distributionen für den Raspberry Pi gehören:

Ubuntu Mate
: Eine spezielle Version von Ubuntu Mate für den Raspberry Pi 2 und 3. Sie eignet sich vor allem für die Verwendung des Raspberry Pi als Desktop-Computer oder wenn man bereits mit Ubuntu vertraut ist.

OpenElec
: Ein um Kodi entwickeltes eingebundenes Betriebssystem für die alleinige Verwendung des Raspberry Pi als Media-Center.

Open Source Media Center
: Eine weitere Media-Center-Software-Distribution die nur auf Open-Source-Software beruht.

PiNet
: Eine Distribution, die in einer Netzwerkumgebung läuft, um es Lehrenden leichter zu machen, den Raspberry Pi im Unterricht zu verwenden.

A.6 Probieren Sie es selbst

Mittlerweile stehen Tausende Distributionen von Linux zum Download zur Verfügung. Die bekanntesten von ihnen können Sie auf der Website DistroWatch (*http://distrowatch.com*) begutachten. Auf dieser Seite gibt es eine Rangliste der Downloads der zahlreichen Linux-Distributionen und Hinweise auf Updates bei Distributionen und beliebten Softwarepaketen. Schauen Sie sich dort um und lassen sich durch Anklicken einiger Distributionen zeigen, wie diese aussehen und worin sie sich voneinander unterscheiden.

A.7 Wie Open-Source-Software funktioniert

Quelloffene Software unterscheidet sich in vielerlei Hinsicht von proprietärer Software. Zunächst einmal ist der Quellcode, wie das Wort *offen* bereits impliziert, für jedermann einsehbar. Möglicherweise fragen Sie sich beispielsweise, warum ein Programm x macht, wenn Sie y tun, nicht aber wenn Sie z tun. Vielleicht wollen Sie das wissen, weil die Software nicht richtig arbeitet oder weil Sie ein Programmierer sind, der einen ähnlichen Algorithmus, eine ähnliche Funktion oder Programmiertechnik in der eigenen Software verwenden möchte. Bei proprietärer Software kann man das nicht herausfinden. Sie müssten die Firma bitten, die die Software entwickelt hat, den Fehler zu beheben, was dann eventuell geschieht oder eben auch nicht. Bei quelloffener Software hingegen kann man mit ein paar Programmierkenntnissen den Code tatsächlich einsehen und schauen, was los ist. Ob Sie nun in der Lage sind, den Code zu verstehen bzw. das Problem zu beheben oder nicht, so agieren Sie jedenfalls nicht hinter einer Wand voller Geheimniskrämerei und Unsicherheit.

Bloß weil der Code öffentlich zugänglich ist, gehen einige Leute fälschlicherweise davon aus, dass damit große Sicherheitsrisiken verbunden wären. Sie glauben, dass dies bösen Mächten Tür und Tor öffnen würde, um Schadsoftware und Viren einzuschmuggeln. Es stimmt zwar, dass diese Offenheit im Prinzip zu solchen Aktionen einlädt, doch eben diese Offenheit ist ein Grund, der verhindert, dass sie zu einem tatsächlichen Problem werden. Durch die Quelloffenheit steht der Code unter ständiger Beobachtung. Experten auf der ganzen Welt prüfen neuen Code auf Herz und Nieren und decken solche Fehler und Sicherheitslücken häufig viel früher auf, als es bei kommerzieller Software der Fall ist.

Zweitens ist es so, dass durch die Verfügbarkeit des Quellcodes jeder aus der Gemeinschaft der Benutzer etwas zu ihm beitragen kann. Tritt ein Problem mit der Software auf, kann man es selbst beheben, einen Fehlerbericht einreichen oder gleich einen Patch zur Verfügung stellen, der es behebt. Wenn das Programmieren nicht Ihre Welt ist, gibt es dennoch Möglichkeiten, etwas beizutragen. Open-Source-Projekte suchen immer nach Leuten, die bei anderen Aufgaben, wie Dokumentationen, Übersetzungen, Marketing, Webdesign oder interner Kommunikation behilflich sind. Sich an einem Open-Source-Projekt zu beteiligen ist eine wunderbare Möglichkeit für diejenigen, die über eine Laufbahn in der Softwareentwicklung oder dem Marketing nachdenken oder die einfach Kontakt zu Entwicklern suchen, um Erfahrungen zu sammeln.

Drittens kann quelloffene Software an Dritte weitergegeben werden, ohne gegen das Gesetz oder Lizenzvereinbarungen zu verstoßen. Ganz im Gegenteil, die Weitergabe wird sogar gefördert. Dies steht im starken Kontrast zu dem, was man sonst in Sachen Weitergeben von Inhalten gewohnt ist. Interessengemeinschaften der Medienindustrie wie RIAA oder MPAA (GEMA in Deutschland) verwenden große Anstrengungen darauf, dass man die gesetzlich geschützten Werke nicht einfach kopiert. Quelloffene Projekte unterliegen derlei Restriktionen nicht, sondern veröffentlichen deren Inhalte auf Webseiten, die die Zusammenarbeit und das Weitergeben zum Zweck haben, darunter GitHub und SourceForge. Mithilfe von Software zur Weitergabe wie BitTorrent kann man den Benutzern den Download erleichtern, da er für diesen Zweck völlig legal ist.

Open-Source-Software leistet einen wertvollen Beitrag zur Maker-Community, da die Ideen und Projekte einfacher implementiert, weitergegeben und verbessert werden können, sobald Software verändert und zur Verfügung gestellt wurde. Es gibt allerdings bei der Verwendung quelloffener Software in Projekten ein paar Dinge zu beachten. Wie bei jeder anderen Software auch kann Open-Source-Software in unterschiedlichen Entwicklungsstadien vorliegen. Dabei kann es vorkommen, dass die Softwareentwicklung zum Erliegen kommt. In solchen Fällen besteht die Gefahr, dass Funktionen, die Sie für Ihr Projekt eigentlich benötigen, niemals implementiert werden. Zudem geschieht es häufig, dass quelloffene Software bereits früh in der Entwicklungsphase freigegeben wird, damit sie getestet und daraufhin verbessert wird. Hier geht man insofern ein Risiko ein, dass diese Versionen mehr Fehler enthalten und sich die Funk-

tionalität schneller ändern kann, als dies bei den stabilen Versionen der Fall ist. So kann es passieren, dass Sie aufgrund einer bestimmten Funktion, die Sie brauchen, die neue Version in der Frühphase der Entwicklung verwenden und alles wie gewünscht funktioniert. Sobald Sie dann aber ein Update machen, besteht die Möglichkeit, dass dann gar nichts mehr geht. Wie sich später herausstellt, wurde der Code so verändert, dass Ihre Verwendungsart nicht mehr möglich ist und Sie nun gezwungen sind, alles so zu verändern, dass es wieder funktioniert. Solche Probleme mit quelloffener Software sind glücklicherweise leicht zu beheben, doch es lohnt sich, ein Auge auf den Entwicklungsstand einer Software zu haben, um negative Auswirkungen auf sein Projekt zu vermeiden.

Viertens ist es gut zu wissen, mit welcher Open-Source-Lizenz man es bei seiner Software genau zu tun hat. Die sehr verbreitete GPL habe ich bereits erwähnt, doch neben ihr gibt es noch Hunderte weiterer Open-Source-Lizenzmodelle. Während einige von ihnen ziemlich viel zulassen, gibt es bei anderen starke Beschränkungen. Die meisten von ihnen ermöglichen den kommerziellen und den privaten Gebrauch sowie das Recht auf Veränderung und Weitergabe. Fast immer enthalten sie Haftungsausschlüsse, falls die Software missbraucht wird oder nicht korrekt funktioniert. Wie bei der GPL wird bei vielen gefordert, dass Veränderungen am Code in Form von Open-Source-Software zur Verfügung gestellt werden, wohingegen dies bei anderen (wie etwa BSD) nicht der Fall ist. Wenn Sie jemals das Terminal unter macOS benutzt haben, ist Ihnen wahrscheinlich aufgefallen, wie ähnlich es dem Linux ist. Das liegt daran, dass macOS zum Teil auf einem Unix-Abkömmling, dem FreeBSD, beruht, das, wie Sie bereits vermuten, die BSD-Lizenz verwendet. Da bei dieser nicht zwingend alle an der Software vorgenommenen Änderungen offengelegt werden müssen, kann Apple den Code nach Belieben verändern und verbessern, ohne es den Nutzern sagen zu müssen. Das ist natürlich deren gutes Recht, aber wäre es nicht schön, wenn diese Veränderungen der Community mitgeteilt würden, die den Code anfangs mitgestaltet hat? Es mag irgendwann so weit kommen, dass Sie selbst aus einem Softwareprojekt etwas Eigenes machen oder an einem Code arbeiten, der unter quelloffener Lizenz steht. Dabei müssen Sie sich stets darüber im Klaren sein, dass es erforderlich sein kann, das Lizenzmodell beizubehalten und den Code wieder an die Community zu übergeben. Mehr über die diversen Open-Source-Lizenzmodelle erfahren Sie bei *opensource.org*.

A.8 Einplatinencomputer versus Mikrocontroller

Auch wenn dieses Thema streng genommen nicht zur Geschichte von Linux gehört, ist dies dennoch eine gute Gelegenheit, über die Gemeinsamkeiten und Unterschiede zwischen Einplatinencomputern (SBCs) und Mikrocontrollern zu sprechen. Diese beiden Begriffe werden in der Maker-Community immer wieder genannt, doch manchmal ist nicht ganz klar, welchen Einsatzzweck die beiden jeweils haben.

Mikrocontroller sind Platinen, auf denen sich ein Prozessor, etwas Arbeitsspeicher und vor allem Schnittstellen mit diversen Eingangs- und Ausgangsverbindungen befinden. Diese liegen häufig in Form von Pins vor, sodass man sie leicht an andere Bauteile und Module anschließen kann. Einige Mikrocontroller haben einen USB-Port und den dazugehörigen Controller, um sie direkt an einen Computer anzuschließen und die entsprechende Firmware hochzuladen. Typische Vertreter dieser Mikrocontroller sind Arduino, Teensy, ESP8266 und das Ti MSP LaunchPad.

Einplatinencomputer (SBCs) sind, wie der Name schon sagt, vollständige Computersysteme, die auf eine einzige kleine Platine passen. In der Regel haben sie neben der CPU einen Arbeitsspeicher, einen Controller für den Speicher, USB-Anschlüsse, Video- und Audiochips sowie Netzwerkfähigkeiten. Außerdem läuft auf ihnen ein vollständiges Betriebssystem wie Linux. Beispiele für Einplatinencomputer sind der Raspberry Pi, BeagleBone, Odroid und C.H.I.P. In Abbildung A–2 sind stellvertretend ein Raspberry Pi und ein Arduino zu sehen.

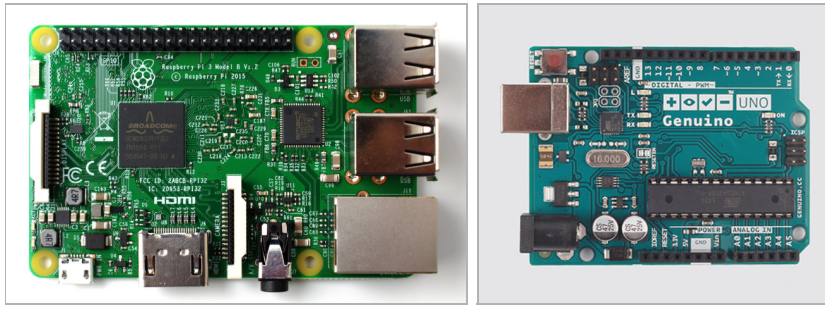


Abb. A–2 Der Raspberry Pi 3 (links) und der Arduino Uno (rechts)

Sowohl Einplatinencomputer als auch Mikrocontroller stellen leistungsfähige Plattformen für Bastelprojekte dar, da sie beide mit anderen Bauteilen verbunden werden und mit diesen kommunizieren können. In letzter Zeit konnten beide sehr klein gebaut werden, sodass sie sich räumlich sehr gut in Projekte integrieren lassen. Eine weitere Gemeinsamkeit besteht darin, dass sie modular aufgebaut sind, also ihr Funktionsumfang durch Anschluss von Bauteilen oder Platinen erweitert werden kann. Beim Raspberry Pi spricht man bei den Erweiterungsplatinen von *Hats* und beim Arduino von *Shields*.

Mikrocontroller eignen sich vor allem für einfache Aufgaben, die zuverlässig und stetig wiederholt werden müssen. Im Gegensatz zu Einplatinencomputern haben Mikrocontroller wenig Arbeitsspeicher, sodass die Anzahl von Befehlen und somit die Programmgröße limitiert ist. Da Mikrocontroller allerdings kaum Peripherie haben und keine anderen Programme um Ressourcen ringen, erledigen sie ihre Aufgaben sehr schnell. Da die Firmware eines Mikrocontrollers direkt in dessen Speicher abgelegt ist und sofort nach dem Hochfahren läuft, verhalten sich Mikrocontroller nach jedem Einschalten stets gleich. Sie eignen sich zum Beispiel für Aufgaben wie die Ansteuerung von einer LED-Matrix, dem Einholen von Sensordaten oder dem sich stets wiederholenden Schicken von Befehlen an Schrittmotoren einer CNC-Maschine oder einen 3D-Drucker.

Einplatinencomputer hingegen lassen sich wie ein normaler Computer verwenden. Da sie mehr Arbeits- und Datenspeicher sowie ein vollständiges Betriebssystem besitzen, können sie mehrere Aufgaben gleichzeitig erledigen. Dies bringt allerdings mit sich, dass die Ressourcen knapp werden können. Da sich das Betriebssystem nicht immer vollständig im Arbeitsspeicher befindet, darf man dem System nicht einfach die Stromzufuhr entziehen, da es beim nächsten Mal eventuell nicht mehr hochfährt oder instabil läuft. Statt die Firmware wie bei einem Mikrocontroller nach jeder Änderung hochladen zu müssen, kann man seine Änderungen an den Programmen und Funktionen direkt am laufenden System über den Desktop oder das Terminal vornehmen. Ein weiterer Vorteil der Einplatinencomputer mit Linux besteht darin, dass man sehr einfach mit den unterschiedlichsten Programmier- und Skriptsprachen arbeiten kann. Auch wenn vornehmlich Python zum Bau von Projekten verwendet wird, kann man ebenso gut mit Perl, Jana, Go oder C arbeiten.

A.9 Warum dies für Maker wichtig ist

Maker profitieren von leistungsfähigen und vielseitigen Entwicklungsumgebungen für ihre Projekte. Die Möglichkeit, aus vielen verschiedenen Arten von Software und Programmiersprachen wählen zu können, führt dazu, dass sie das für sie passende aussuchen können. Maker müssen im Verlauf der Entwicklung ihrer Projekte direkten Zugriff auf ihre Software haben, wobei das Budget häufig sehr limitiert ist. Linux, das für sich genommen auch ein Maker-Projekt darstellt, gibt Makern die Freiheit, ohne zusätzliche Kosten und mit sehr anpassungsfähigen Softwarepaketen die neueste Technologie für ihre Projekte einzusetzen. Viele Open-Source-Projekte sind aus persönlichen Bedürfnissen heraus entstanden. Auch Maker-Projekte starten oft klein durch Einzelpersonen und können zu tragfähigen Geschäftsmodellen werden, ebenso wie Open-Source-Projekte, die aus einem Hobby heraus zu etwas wurden, das Tausende täglich nutzen, manchmal sogar, ohne dass es dem Entwickler auffiel. Open-Source-Software ist das Herzstück von Linux und so liegt es nahe, dass Maker sie genauso selbstverständlich einsetzen wie einen Schraubenzieher, Hammer oder 3D-Drucker. Linux in seinem Projekt zu verwenden mag zunächst sehr mühsam erscheinen, doch mit ein paar Hilfestellungen und Tipps, so glaube ich, werden auch Sie die Freiheit und die endlosen Möglichkeiten von Linux schätzen lernen.

Index

Rezensieren

Sie dieses Buch

Senden

Sie uns Ihre Rezension
unter **www.dpunkt.de/rez**

Erhalten

Sie Ihr Wunschbuch aus
unserem Verlagsangebot