

Topthemen dieser Ausgabe

Administration von Debian & Co im Textmodus – Teil II

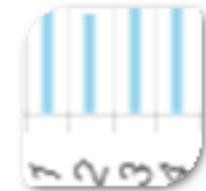
Seite 5

Im zweiten Teil des insgesamt dreiteiligen Workshops für alle, die sich grundlegende Kenntnisse in der Administration von Debian-basierten Systemen aneignen müssen oder wollen, geht es um wichtige Orte, Dämonen, das Netzwerk, Fernzugriff und um die Paketverwaltung. ([weiterlesen](#))

R-Webapplikationen mit Shiny und OpenCPU

Seite 19

Moderne Webapplikationen müssen nicht auf ein statistisches Backend verzichten. Mit Shiny von RStudio und dem OpenCPU-Projekt von Jeroen Ooms werden zwei populäre Frameworks vorgestellt, um das Statistikprogramm R in Webseiten einzubinden. Passend dazu gibt es im Magazin auch noch eine Rezension zu „Datendesign mit R“. ([weiterlesen](#))



Browser: alle doof. – Eine Odyssee durch die Welt der Webbetrachter

Seite 26

Während der normale Linuxnutzer bei der Auswahl seiner Distribution, Desktopumgebung und Programme meist nach persönlichen Vorlieben entscheidet, haben Sehbehinderte diesen Luxus nicht. Hier kommt es darauf an, wie gut sich eine Oberfläche anpassen lässt oder überhaupt zugänglich ist. Es gibt zwar spezielle Distributionen für Blinde, doch auch hier ist die Auswahl für „nur“ Sehbehinderte im Prinzip nicht viel anders als bei anderen Linuxen. Das gilt auch für eines der wichtigsten Programme, den Webbrowser. ([weiterlesen](#))



Editorial

Kurzgeschichte

Im letzten Monat erreichte uns ein Leserbrief mit zahlreichen Artikelvorschlägen. Ein Vorschlag darin bezog sich auf eine technisch basierte Kurzgeschichte. Bisher haben wir keine Kurzgeschichten oder ähnliches veröffentlicht, da der Schwerpunkt von **freiesMagazin** auf Anleitungen und Berichten über Neuigkeiten aus den Gebieten Freier Software, Open Source und Linux liegt.

Wir haben uns aber entschieden das Experiment zu wagen und veröffentlichen in der aktuellen Ausgabe unter dem Titel *Recht und Unrecht* auf **Seite 35** die Kurzgeschichte von Stefan Wichmann. Für die Auswertung des Experiments sind wir auf Sie als Leser von **freiesMagazin** angewiesen und haben dazu eine kurze Umfrage [1] auf unserer Homepage erstellt. Dort können Sie mit zwei Klicks Ihre Meinung zum Thema Kurzgeschichten in **freiesMagazin** abgeben.

Wir freuen uns natürlich auch über Ihre Kommentare und Leserbriefe zu dem Thema.

Der Schwerpunkt von **freiesMagazin** wird mit Sicherheit auf Neuigkeiten und Wissenswerten aus dem Bereich Freier Software bleiben, aber wenn Sie als Leser solche Geschichten häufiger lesen möchten und wir entsprechende Einsendungen erhalten, könnten wir uns eine Veröffentlichung in loser Folge gut vorstellen.

Browserbetrachtung

Eine Begutachtung einer ganzen Reihe von verfügbaren Webbrowsern unter einem besonderen Gesichtspunkt liefert der Artikel *Browser Odyssee* auf **Seite 26**. Die Bewertung der Autorin Jennifer Rößler erfolgt nach Kriterien, die für viele Nutzer wohl kaum eine Rolle spielen, für Sehbehinderte aber essentiell sind, um sich vernünftig im Internet zu bewegen. Wie der Artikel zeigt, stehen diese Kriterien für die Entwickler der Browser häufig nicht im Vordergrund oder werden gar nicht beachtet, was das Nutzen des Internets für diese Nutzergruppe unnötig schwierig macht.

Haben auch Sie Erfahrung mit dem Thema Barrierefreiheit im Internet und können einen Artikel zu diesem Thema schreiben? Wir würden uns sehr über weitere Einsendungen zu diesem Thema freuen.

Und nun wünschen wir Ihnen viel Spaß mit der neuen Ausgabe!

Ihre **freiesMagazin**-Redaktion

LINKS

[1] <http://www.freiesmagazin.de/20140907-moechten-sie-weitere-kurzgeschichten-in-freiesmagazin-lesen>

Inhalt

Linux allgemein

Der August im Kernelrückblick S. 3

Anleitungen

Administration von Debian & Co im Textmodus – Teil II S. 5

Spacewalk – Teil 2: Registrierung und Verwaltung von Systemen S. 13

Software

R-Webapplikationen mit Shiny und Open-CPU S. 19

Browser: alle doof. – Eine Odyssee durch die Welt der Webbetrachter S. 26

Äquivalente Windows-Programme unter Linux – Systemprogramme und Zubehör S. 30

Community

Recht und Unrecht S. 35

Rezension: Datendesign mit R S. 39

Rezension: Design Patterns mit Java S. 41

Rezension: Android – Einstieg in die Programmierung S. 44

Magazin

Editorial S. 2

Leserbriefe S. 46

Veranstaltungen S. 47

Vorschau S. 47

Impressum S. 48



Teilen



Kommentieren

Der August im Kernelrückblick von Mathias Menzer

Basis aller Distributionen ist der Linux-Kernel, der fortwährend weiterentwickelt wird. Welche Geräte in einem halben Jahr unterstützt werden und welche Funktionen neu hinzukommen, erfährt man, wenn man den aktuellen Entwickler-Kernel im Auge behält.

Linux 3.16

Den August leitete Torvalds mit der Freigabe von Linux 3.16 ein [1]. Der Entwicklungszyklus war ausgesprochen kurz: Zwischen den Versionen 3.15 und 3.16 liegen gerade einmal 56 Tage. Auch wenn man die Extrawoche hinzuzählt, die Torvalds das Merge Window früher eingeläutet hat [2], ist der Zeitraum beeindruckend kurz geraten. Dennoch siedelt sich 3.16 mit fast 14.000 Änderungen oberhalb des Durchschnitts der Reihe der 3er-Kernel an.

Auch der neueste Stable-Kernel folgt wieder der Strategie der vielen kleinen Schritte und hat keine große Umwälzungen oder Neuerungen, dafür jedoch viele kleine Nettigkeiten an Bord. So wird nun beispielsweise Suspend und Resume für den Hypervisor Xen auf ARM-Systemen unterstützt, wodurch das besonders schnelle Beenden und wieder Starten virtueller Maschinen ermöglicht wird. Außerdem kam bei Xen die Unterstützung mehrerer Warteschlangen für virtuelle Netzwerkschnittstellen hinzu, wodurch Verbesserungen der Leistung erwartet werden. KVM (Kernel based Vir-

tual Machine) konnte ebenfalls einige Verbesserungen erfahren, allerdings hauptsächlich auf den Architekturen s390, Power8 und MIPS. Weiterhin lassen sich nun virtualisierte Gastsysteme innerhalb eines Xen-Gastsystems betreiben solange der Host auf einer x86-Hardware betreiben wird. Der eigentliche Mieter kann nun sozusagen Platz an einen oder mehrere Untermieter weitervermieten.

Mit TCP Fast Open (TFO) soll der Aufbau von Netzwerkverbindungen zwischen Client und Server beschleunigt werden. Diese Funktion ist seit Linux 3.13 standardmäßig aktiviert (siehe „Der Januar im Kernelrückblick“, [freiesMagazin 02/2014 \[3\]](#)), allerdings war sie bislang nur für das alte Internetprotokoll IPv4 verfügbar. Die Unterstützung für IPv6 wurde nun nachgereicht.

Einiges an Arbeit floss in die Control Groups (cgroups). Sie dienen der Verwaltung der System-Ressourcen indem diese auf Gruppen von Prozessen je nach Priorität verteilt werden können. Diese Gruppen lassen sich hierarchisch zusammenfassen, was bislang auch mit mehreren Hierarchien funktionierte. Eine wichtige Neuerung ist, dass dies nun auf eine Hierarchie beschränkt werden wird, vergleichbar dem Linux-Dateisystem. So kann das neue Control Group Virtual File System auch mit dem `mount`-Befehl eingehängt werden, auch wenn dies im derzeitigen Stadium eher für Entwickler von Interesse sein dürfte.

Ebenfalls hinter den Kulissen fand die Arbeit am Block Layer statt. Die Steuerung von Ein-/Ausgabe-Vorgängen kann nun in mehrere Warteschlangen aufgeteilt werden. Bislang wurden die Zugriffe auf blockorientierte Geräte, insbesondere Datenträger wie Festplatten, SSDs und Speichersticks, von jeweils nur einer Warteschlange abgewickelt. Anfragen wurden an deren Ende angehängt und nacheinander abgearbeitet oder gegebenenfalls umsortiert, um Zugriffe zu optimieren. Doch diese Warteschlange kann nur von jeweils einer Instanz gesperrt und bearbeitet werden, was in Zeiten leistungsfähiger Datenträger durchaus zu Leistungseinbußen führen kann. Die neue Methode stellt nun eine Warteschlange für jede CPU ab, die Anfragen der auf diesem Prozessor laufenden Prozesse entgegennehmen kann, wodurch das gegenseitige Sperren von Zugriffen auf die Warteschlange entfällt. Ein erster Treiber PCIe-Flash-Speicher von Micron (mtip32xx) wurde bereits für die Verwendung von „Block Layer Multiqueue“ (blk-mq) angepasst.

Eine ausführliche Auflistung der Neuerungen findet sich in einer dreiteiligen Artikelserie des englischsprachigen Portals LWN.net [4] [5] [6].

Linux 3.17

Das Experiment mit dem um einige Tage vorgezogenen Merge Window wurde bei Linux 3.17 nicht wiederholt. Und so sammelte Torvalds fast zwei Wochen lang Pull Requests der Maintainer

und legte Mitte August die erste Entwicklerversion vor [7]. Eher ungewöhnlich ist, dass diesmal mehr Quellcode entfernt als hinzugenommen wurde. Der Grund dafür kann nur in Aufräumarbeiten zu finden sein, die in der Vergangenheit schon öfter in einzelnen Bereichen durchgeführt wurden und von denen diesmal sogar mehrere anstanden. Diese „Cleanups“ betreffen die Architekturen ARM und x86. Ein weiterer nicht ganz kleiner Teil entfällt auf Vereinfachungen der Interprozesskommunikation [8] für alle Architekturen. Den bei weitem größten Wurf leistet aber der Staging-Bereich, in dem die Entwickler einen verspäteten Frühjahrsputz durchgeführt haben. So wurden 14 Treiber herausgeworfen, für die sich niemand mehr zuständig fühlte und weitere Treiber konnten überarbeitet und ihr Quelltext aufgeräumt werden. Konkret entfallen alleine auf den Staging-Zweig über 250.000 der insgesamt 650.000 Removals, dagegen kamen hier nur 33.000 Zeilen an Quellcode hinzu, die sich dann im Vergleich zu den knapp 630.000 hinzugefügten Textzeilen des ganzen -rc1 eher bescheiden ausnehmen.

Auch im Hinblick darauf, dass sich ein guter Teil der Entwicklungsgemeinde beim diesjährigen Kernel Summit befand, fiel Linux 3.17-rc2 [9] recht kompakt aus. Das bemerkenswerteste an dieser Version war wohl die Tatsache, dass sie exakt 23 Jahre nach Torvalds Ankündigung seines freien Betriebssystems [10] veröffentlicht wurde. Dies war allerdings kein Zufall, Torvalds nannte seine Sentimentalität als Grund hierfür.

Die Größe der dritten Entwicklerversion [11] kann durchaus als Zeichen gewertet werden, dass das Gros der Entwickler wieder sicher zuhause angekommen ist. Die Zahl der Änderungen ist ein wenig gestiegen, doch finden sich hier fast ausschließlich Korrekturen. Auch hier macht sich der Staging-Zweig wieder bemerkbar. 11.000 Codezeilen wurden hier wieder entfernt, sie tauchen allerdings auch auf der Haben-Seite wieder auf; Der usbip-Treiber [12], der die Bereitstellung von USB-Geräten an andere Rechner im Netzwerk ermöglicht, hat den Staging-Zweig verlassen und die einzelnen Komponenten wurden nun auf den Treiber- und Userspace-Bereich verteilt.

Zwei-Faktor für Kernel-Git

Die Linux Foundation hat eine Initiative gestartet, um die Git-Repositoryen des Linux-Kernel besser abzusichern [13]. Künftig sollen Zugriffe auf die wichtigsten Kernel-Repositories mittel Zwei-Faktor-Authentifizierung [14] erfolgen. Zu diesem Zweck wurden auf dem Kernel Summit Hardware-Token der Firma Yubico an Kernel-Entwickler verteilt, die damit die bisherigen SSH-Schlüssel ersetzen sollen. Diese Hardware-Token verhalten sich wie eine USB-Tastatur, die auf Knopfdruck ein Einmal-Kennwort eintippt. Dieses wird dann von einer neu entwickelten Erweiterung für gitolite [15] verifiziert und der Zugang daraufhin gestattet oder abgelehnt. Es soll damit verhindert werden, dass Unbefugte sich als Kernel-Entwickler ausgeben und Schadcode in den Kernel einbringen können.

- [1] <https://lkml.org/lkml/2014/8/3/82> 
- [2] <https://lkml.org/lkml/2014/6/1/264> 
- [3] <http://www.freiesmagazin.de/freiesMagazin-2014-02>
- [4] <https://lkml.org/lkml/2014/8/16/79> 
- [5] <http://lwn.net/Articles/601152/> 
- [6] <http://lwn.net/Articles/601726/> 
- [7] <http://lwn.net/Articles/602212/> 
- [8] <https://de.wikipedia.org/wiki/Interprozesskommunikation>
- [9] <https://lkml.org/lkml/2014/8/25/558> 
- [10] <https://groups.google.com/forum/#!msg/comp.os.minix/dlNtH7RRrGA/SwRavCzVE7gJ> 
- [11] <https://lkml.org/lkml/2014/8/31/173> 
- [12] <http://usbip.sourceforge.net/> 
- [13] <http://www.pro-linux.de/-0h2153a5>
- [14] <https://de.wikipedia.org/wiki/Zwei-Faktor-Authentifizierung>
- [15] <http://gitolite.com/> 

Autoreninformation

Mathias Menzer ([Webseite](#)) behält die Entwicklung des Linux-Kernels im Blick, um über kommende Funktionen von Linux auf dem Laufenden zu bleiben.



Administration von Debian & Co im Textmodus – Teil II von Hauke Goos-Habermann und Maren Hachmann

Im zweiten Teil des insgesamt dreiteiligen Workshops für alle, die sich grundlegende Kenntnisse in der Administration von Debian-basierten Systemen aneignen müssen oder wollen, geht es nun um wichtige Orte, Dämonen, das Netzwerk, Fernzugriff und um die Paketverwaltung.

Um richtig mitmachen und alles selbst ausprobieren zu können, benötigt man zwei virtuelle Maschinen. Der Link zum Herunterladen und eine Anleitung zum Einrichten sind im vorangegangenen Artikelteil in [freiesMagazin 08/2014 \[1\]](#) zu finden. Bisher wurden noch keine wesentlichen Änderungen am System vorgenommen, sodass auch „Quereinsteiger“, die sich nur für die Themen dieses Workshops interessieren, nach der Einrichtung gleich mit dem zweiten Teil loslegen können.

Auch weiterhin ist es für jeden, der möglichst viel von den Inhalten mitnehmen möchte, empfehlenswert, die hier aufgeführten Kommandos selbst auszuprobieren, auch wenn nicht ausdrücklich dazu aufgefordert wird. Probleme, die beim Ausprobieren auftreten, und die entsprechenden Problemlösungsstrategien lassen sich so am einfachsten finden, erproben und im Gedächtnis verankern.

Auf dem virtuellen Workshop-Computer kann ja zum Glück – im Gegensatz zum Server „im echten

Leben“ – nichts kaputt gehen, was durch ein erneutes Importieren der virtuellen Maschine nicht wiederherzustellen wäre.

Was ist wo?

Auf Linux-Systemen legen die meisten Anwendungen ihre Dateien in nach Art der Information sortierten Verzeichnissen ab, sodass man z. B. die ausführbare Datei gemeinsam mit anderen ausführbaren Dateien in einem Verzeichnis findet und die Einstellungen, die ein bestimmter Benutzer an diesem Programm vorgenommen hat, an einer festgelegten Stelle in seinem Benutzerverzeichnis.

Diese durch den FHS (Filesystem Hierarchy Standard [2]) vereinheitlichte Verzeichnisstruktur vereinfacht es, sich auf fremden Linuxsystemen zu rechtzufinden.

Dies ist ein kleiner Wegweiser durch den Linux-Verzeichnisbaum, den es sich zu merken lohnen kann:

- Ausführbare Dateien für alle Benutzer finden sich in **/bin** und **/usr/bin**.
- Ausführbare Dateien für root beziehungsweise das System liegen unter **/sbin** und **/usr/sbin**.
- Gerätedateien sind im Ordner **/dev** zu finden.
- Kernel und initrd [3] befinden sich in **/boot**.
- Grub [4] liegt im Verzeichnis **/boot/grub**.

- Globale Einstellungen werden unter **/etc** gespeichert.
- Globale Programmeinstellungen kann man unter **/etc/Programmname** oder **etc/default/Programmname** finden.
- Benutzerverzeichnisse sind normalerweise **/home/Benutzername**.
- Start-/Stop-Skripte gibt es in **/etc/init.d**.
- Protokolldateien werden in **/var/log** geschrieben.

Dienste und Dämonen

Die Dämonen sind die dienstbaren Geister des Systems oder, einfach ausgedrückt, Programme, die im Hintergrund unauffällig ihren Dienst versehen (sollten). Der Apache-Webserver z. B. läuft als Dämon, um Webseitenanfragen entgegenzunehmen und zu beantworten. Doch auch auf reinen Desktopsystemen laufen eine Reihe von Dämonen wie z. B. ein Dämon zum automatischen Anpassen der CPU-Taktfrequenz. Viele Dämonen werden direkt beim Systemstart mitgestartet.

SysVinit

Das SysVinit [5] ist zwar schon etwas in die Jahre gekommen, aber in der einen oder anderen Form noch auf vielen Systemen vorhanden. Es ist der erste Prozess, der beim Hochfahren vom Kernel gestartet wird. Er sorgt dafür, dass die anderen Dienste in einer festgelegten Reihenfolge gestartet werden. Daher kann es nicht schaden, sich damit auseinanderzusetzen:

- Liste aller Dämonen ausgeben:

```
# service --status-all
```

In der Ausgabe zeigen die Zeichen +, - und ? an, ob der jeweilige Dienst gestartet oder gestoppt ist beziehungsweise, ob sein Status unbekannt ist.

- Einen (gestoppten) Dämon starten:

```
# /etc/init.d/<Dämon> start
```

oder

```
# service <Dämon> start
```

- Einen (gestarteten) Dämon stoppen:

```
# /etc/init.d/<Dämon> stop
```

oder

```
# service <Dämon> stop
```

- Einen (laufenden) Dämon beenden und neu starten:

```
# /etc/init.d/<Dämon> restart
```

oder

```
# service <Dämon> restart
```

Neben den Kommandos **start**, **stop** und **restart** können je nach Dämon weitere hinzukommen. Normalerweise verrät

```
# /etc/init.d/<Dämon>
```

die möglichen Kommandos.

Upstart

Upstart [6] ist eine Alternative zum klassischen SysVinit und wird unter anderem bei Ubuntu verwendet. Teilweise sind auf ein und demselben System auch beide – SysVinit und upstart – anzutreffen, was auch gelegentlich zu Problemen durch eine von der Planung abweichende Startreihenfolge führen kann.

- Liste aller Dämonen ausgeben:

```
# initctl list
```

- Einen (gestoppten) Dämon starten:

```
# initctl start <Dämon>
```

- Einen (gestarteten) Dämon stoppen:

```
# initctl stop <Dämon>
```

Systemd

systemd [7] ist ein weiteres init-System, das ab Debian 8 (Jessie) und in den kommenden Ubuntu-Veröffentlichungen zum Standard werden soll. In diesem Fall lauten die entsprechenden Befehle:

- Liste aller Dämonen ausgeben:

```
$ systemctl --all
```

- Einen (gestoppten) Dämon starten:

```
# systemctl start <Dämon>
```

- Einen (gestarteten) Dämon stoppen:

```
# systemctl stop <Dämon>
```

- Status- und Loginformationen über Dämon abfragen:

```
$ systemctl status <Dämon>
```

- Loginformationen über Dämon abfragen:

```
$ journalctl -u cron
```



Debian Logo. © Software in the Public Interest, Inc. (CC-BY-SA-3.0) 🔍

Netzwerk

Was wäre das Leben ohne Netzwerk? Ziemlich öde – oder vielleicht auch deutlich produktiver (ohne die ganzen Ablenkungen)... Jedenfalls kann es sehr nützlich sein, wenn man weiß, wie man das Netzwerk selbst einrichtet. Bei der Workshop-VM ist es nämlich zunächst ziemlich kaputt...

Aktuelle Netzwerkeinstellungen

Zunächst gibt das Kommando

```
# ifconfig
```

einen Überblick über die aktuellen Einstellungen aller aktiven Netzwerkkarten (siehe Ausgabe auf nächster Seite).

```

1 eth0      Link encap:Ethernet Hardware Adresse 00:f3:ab:77:3d:28
2          inet Adresse:192.168.1.123 Bcast:192.168.1.255 Maske:255.255.255.0
3          inet6-Adresse: fe80::200:f0fe:fe83:83f6/64 Gueltigkeitsbereich:Verbindung
4          UP BROADCAST RUNNING MULTICAST MTU:1500 Metrik:1
5          RX packets:850053 errors:0 dropped:0 overruns:0 frame:0
6          TX packets:533491 errors:0 dropped:0 overruns:0 carrier:0
7          Kollisionen:0 Sendewarteschlangenlaenge:1000
8          RX bytes:1021113415 (973.8 MiB) TX bytes:63432614 (60.4 MiB)
9          Interrupt:19
10
11 lo       Link encap:Lokale Schleife
12          inet Adresse:127.0.0.1 Maske:255.0.0.0
13          inet6-Adresse: ::1/128 Gueltigkeitsbereich:Maschine
14          UP LOOPBACK RUNNING MTU:16436 Metrik:1
15          RX packets:917067 errors:0 dropped:0 overruns:0 frame:0
16          TX packets:917067 errors:0 dropped:0 overruns:0 carrier:0
17          Kollisionen:0 Sendewarteschlangenlaenge:0
18          RX bytes:725063143 (691.4 MiB) TX bytes:725063143 (691.4 MiB)
19
20 wlan0    Link encap:Ethernet Hardware Adresse 00:2e:2d:56:34:cc
21          UP BROADCAST MULTICAST MTU:1500 Metrik:1
22          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
23          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
24          Kollisionen:0 Sendewarteschlangenlaenge:1000
25          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

```

In diesem Beispiel kennt das System drei Netzwerkanalysen:

- die „echte“ Netzwerkkarte für kabelgebundenes Netzwerk (ab Zeile 1)
- die „lokale Schleife“ oder das „Loopback“ (ab Zeile 11); eine Art Netzwerk, das die Kommunikation von lokal installierten Servern und Clients untereinander erlaubt
- die WLAN-Karte (ab Zeile 20). Diese hat aktuell keine inet-Adresse bzw. inet6-Adresse, also keine IP – sie ist damit nicht aktiv.

Den Anschluss finden

Wenn das Netzwerk streikt, kann folgende Testreihe zum Erkennen des Fehlers beitragen:

1. Hat die Netzwerkkarte(n) eine „richtige“ IP?

```
# ifconfig
```

2. Funktioniert die lokale Schleife?

```
# ping 127.0.1.1
```

3. Kann der Router/Gateway „gepingt“ werden?

```
# ping <Router-IP>
```

4. Kann ein Server im Internet erreicht werden?

```
# ping 8.8.8.8
```

5. Funktioniert die Namensauflösung?

```
# ping freiesMagazin.de
```

Das Programm ping versucht in regelmäßigen Abständen, die ihm genannte Adresse „anzurufen“. Eine ping-Ausgabe kann dann z. B. so aussehen:

```

PING freiesMagazin.de (46.4.183.98) ~
56(84) bytes of data.
64 bytes from fmserver1.xen-host.de ~
(46.4.183.98): icmp_req=1 ttl=49 time~
=34.9 ms
64 bytes from fmserver1.xen-host.de ~
(46.4.183.98): icmp_req=2 ttl=49 time~
=35.4 ms
...

```

Nach Programmabbruch mit **Strg** + **C** fasst ping das Ergebnis zusammen:

```

11 packets transmitted, 11 received, 0% ~
packet loss, time 10013ms
rtt min/avg/max/mdev = ~
25.809/32.952/49.173/6.683 ms

```

wobei 0% verlorene Pakete eine gute Nachricht sind.

Sollten alle Tests bestanden sein, so liegt anscheinend kein Fehler vor. Ansonsten kann auch schon einmal das Netzkabel herausgerutscht sein (leidige abgebrochene Plastiknasen bei den Steckern...), der Router ausgefallen sein oder, falls der Befehl **ifconfig** die Netzwerkkarte gar nicht auflistet, der Netzwerkkartentreiber fehlen.

Hat die Netzwerkkarte die falsche Bezeichnung (z. B. eth1 statt eth0), so kann dies daran liegen, dass durch udev [8] ein neuer Name vergeben wurde. In diesem Fall sollte man zuerst die udev-Regeldatei für Netzwerkkarten als Backup in ein anderes Verzeichnis verschieben. Die Datei findet sich unter **/etc/udev/rules.d/*-persistent-net.rules**, das Sternchen steht hier für eine Zahl, die die Auswertungsreihenfolge der Regeln im Verzeichnis beschreibt und auf verschiedenen Systemen unterschiedlich sein kann:

```
# mkdir /etc/udevbackup
# mv /etc/udev/rules.d/*-persistent-net.~
rules /etc/udevbackup/*-persistent-net.~
rules
```

Anschließend startet man den Rechner neu:

```
# reboot
```

und beginnt die Testreihe von vorne.

Netzwerkeinstellungen per Hand

Wenn gar nichts mehr geht, kann man einen Rechner relativ einfach wieder ins Netz bringen. Dieses sollte man auf der Workshop-VM unbedingt tun, damit weitere Programme (mc, gpm, ...) installiert werden können und auch das Updaten des Debian-Systems ausprobiert werden kann.

Statische IP

Mit drei Zeilen werden alle nötigen Einstellungen vorgenommen. Die erste Netzwerkkarte (eth0) der Workshop-VM soll die IP **192.168.1.123** zugewiesen bekommen. Die IP des Routers/Gateways (IPCop-VM) ist **192.168.1.4** und der öffentliche, nicht zensierende DNS-Server besitzt die IP **85.88.19.10**. Diese Einstellungen sind jedoch temporär und gehen nach dem nächsten Booten wieder verloren.

Zunächst setzt man für die Netzwerkkarte (eth0) die IP auf **192.168.1.123**:

```
# ifconfig eth0 192.168.1.123
```

Jetzt muss man dem System noch mitteilen, über welchen Computer es das Internet erreichen kann. In diesem Fall ist das die IPCop-VM. Dazu fügt man das Gateway mit folgendem Kommando hinzu:

```
# route add -net default gw 192.168.1.4
```

Nun benötigt das System noch eine Angabe dazu, welcher DNS-Server zur Namensauflösung

genutzt werden soll. Dieser wird mit echo in die Datei **/etc/resolv.conf** eingetragen:

```
# echo 'nameserver 85.88.19.10' > /etc/~
resolv.conf
```

Anschließend sollte man die Netzwerkverbindung wie oben beschrieben testen.

Dynamische IP

Werden Netzwerkeinstellungen dynamisch durch einen DHCP-Server vergeben, so holt

```
# dhclient
```

die Informationen direkt vom DHCP-Server und trägt sie in die Konfiguration des Systems ein. Sollte das nicht funktionieren, so hilft ggf. ein

```
# dhclient -r eth0
```

wodurch die Bindung der Netzwerkkarte an den DHCP-Server gelöst wird, und dann der Befehl

```
# dhclient eth0
```

zum erneuten Konfigurieren der Netzwerkkarte mit den Angaben des DHCP-Servers.

Bei der Workshop-VM lässt sich ein Anschluss dynamisch und statisch herstellen.

Ist einmal nicht klar, welche Maschine als DHCP-Server fungiert, kann man dies in der Datei **/var/**

`lib/dhcp/dhclient.leases` nachlesen. Die IP des DHCP-Servers findet sich in der Zeile, die mit `option dhcp-server-identifizier` beginnt.

Netzwerkeinstellungen über Dateien

Damit die Einstellungen nicht bei jedem Neustart verlorengehen, werden diese in der Datei `/etc/network/interfaces` definiert.

Als Orientierung können diese beiden Beispiele für `/etc/network/interfaces` dienen:

```
# Statische IP
allow-hotplug eth0
iface eth0 inet static
address 192.168.1.123
netmask 255.255.255.0
network 192.168.1.0
broadcast 192.168.1.255
gateway 192.168.1.4

auto lo
iface lo inet loopback
```

oder

```
# Dynamische IP (DHCP):
allow-hotplug eth0
iface eth0 inet dhcp

auto lo
iface lo inet loopback
```

Nun muss das System nur noch die IP des DNS-Servers dauerhaft abspeichern. Diese schreibt

man mit folgender Notation in die Datei `/etc/resolv.conf` (z. B. mit dem Editor nano):

```
nameserver 85.88.19.10
```

Dass die Änderungen wirksam werden, bewirkt der Befehl

```
# /etc/init.d/networking restart
```

worauf eine Meldung folgt, dass diese Methode veraltet ist. Eine mögliche Alternative wäre das manuelle Neustarten der einzelnen Netzwerk-Interfaces mittels `ifup/ifdown`. Manchmal kann (falls das Netzwerk jetzt noch nicht funktioniert) auch ein Neustart des Systems erforderlich sein.

Fernzugriff

Anmerkung: Da die Workshop-VM sich in einem eigenen, vom Netzwerk des Gastgeber-Rechners getrennten, virtuellen Netzwerk befindet, ist eine direkte SSH-Verbindung vom Host-Rechner, auf dem die Virtualisierungssoftware läuft, zur Workshop-VM ohne weitere Einstellungen an der IPCop-Maschine bzw. Änderungen an der Netzwerkkonfiguration der VM nicht möglich. Zum Ausprobieren empfiehlt sich hier das Herstellen einer Verbindung von der Workshop-VM zur IPCop-VM, auf der ein SSH-Server (auf Port 8022) läuft.

Auf die Shell von Linux-Systemen kann (bei entsprechender Einstellung) auch über das Netzwerk zugegriffen werden. Dieser Zugriff erfolgt in der Regel mittels Secure Shell (`ssh` [9]). Das Installationspaket, das hierfür benötigt wird, heißt unter Debian `openssh-server` (auf dem Zielrechner zu installieren) bzw. `openssh-client` (auf dem Rechner, an dem man arbeitet, zu installieren). Ein „Anruf“ mit SSH sieht dann so aus:

Die Angabe `-p <Port>` wird nur dann benötigt, wenn der SSH-Server auf einem anderen als dem Standard-Port 22 lauscht. Mit `<Benutzer>` ist der Benutzer auf dem entfernten Rechner gemeint. Sind der Benutzername auf lokalem und entfernten System identisch, kann die Angabe von `<Benutzer@>` entfallen. `<Rechner>` steht für die IP des Rechners oder dessen auflösbaren Namen. Wird als zusätzlicher Parameter `-X` angegeben, so können sogar grafische Programme auf dem entfernten Rechner ausgeführt und die Ausgabe auf dem lokalen System benutzt werden, falls auf diesem ein X-Client installiert ist.

```
$ ssh -p <Port> <Benutzer>@<Rechner>
```

Zum Einloggen auf der IPCop-VM führt man folgenden Befehl aus:

```
$ ssh -p 8022 root@192.168.1.4
```

Die dabei erscheinende Frage, ob man mit dem Verbindungsaufbau fortfahren möchte, ist mit **yes** zu bejahen. Das Passwort für den root-Benutzer ist **testtest**.

Für das Kopieren von Dateien und Verzeichnissen zwischen entferntem und lokalem System gibt es

die Programme SCP [10] und rsync [11], wobei rsync zwar mächtiger (Fortsetzen von abgebrochenen Kopieraktionen, „intelligentes“ Kopieren, ...), aber auch nicht überall installiert ist.

Beispiele: Um die Datei `/etc/issue` von der IPCop-VM in das Verzeichnis `/tmp` auf die Debian-VM zu kopieren (Achtung: großes **P** bei der Portangabe), benutzt man folgenden Befehl:

```
$ scp -P 8022 root@192.168.1.4:/etc/issue /tmp
```

Ein lokales Verzeichnis **Adminzeug** mit Unterverzeichnissen (`-r`) kopiert man wie folgt in das Heimatverzeichnis von **admin**:

```
$ scp -r Adminzeug admin@entfernteKiste:~
```

Dasselbe mit rsync geschieht wie folgt:

```
$ rsync -Pazy Adminzeug admin@entfernteKiste:~
```

Die Parameterkombination `-Pazy` steht dabei für Fortschrittsanzeige (`-P`), Archivmodus (mit Unterverzeichnissen, symbolische Links als symbolische Links, Berechtigungen erhalten, Änderungszeiten erhalten, Gruppenzugehörigkeit erhalten, Besitzer erhalten, Gerätedateien und Spezialdateien erhalten (`-a`)), Dateien komprimiert übertragen (`-z`), sowie Optimierung der übertragenen Datenmenge (`-y`).

Paketverwaltung

Die meisten Linux-Distributionen verwendet eine Paketverwaltung, um Software zu installieren, deinstallieren oder zu aktualisieren. Bei Debian und abgeleiteten Distributionen wird APT [12] verwendet.

Quellen einrichten

Damit APT weiß, von wo die Pakete heruntergeladen/kopiert werden können, werden in der Datei `/etc/apt/sources.list` und/oder in `*.list`-Dateien unter `/etc/apt/sources.list.d/` die Quellen festgelegt.

```
# für Binär-Pakete benötigt man für Debian Wheezy diese Zeilen,
# Zeile 4 mit Multimedia-Codecs nur bei Bedarf:
deb http://ftp.de.debian.org/debian/ wheezy main non-free contrib
deb http://security.debian.org/ wheezy/updates main contrib non-free
deb http://www.deb-multimedia.org wheezy main non-free

# Quellcode-Pakete (werden selten benötigt):
deb-src http://ftp.de.debian.org/debian/ wheezy main non-free contrib

# Eine lokale Paketquelle könnte so aussehen:
deb file:/meinmirror/ wheezy main non-free contrib

# Wenn man Pakete von einer CD/DVD installieren möchte:
deb cdrom:[Debian DVD]/ wheezy main non-free contrib
```

Hinter der Angabe der jeweiligen Art der Quelle (`deb` für fertige Pakete, `deb-src` für Quellcode) folgen die Angabe der Quelladresse, die Angabe der Version (hier **wheezy**) und die Angabe der Archivbereiche, aus denen Software geladen werden soll.

main beschreibt hierbei den Teil der Quelle, der ausschließlich Software enthält, die fest zur Distribution gehört. **non-free** enthält Software, die nach den Debian Free Software Guidelines [13] als unfrei gilt, und **contrib** schließlich enthält Inhalte, die zwar selbst nach dieser Richtlinie als frei gelten, aber von Software aus **non-free** abhängig sind.

Auf der Workshop-VM finden sich zusätzlich noch Paketquellen für Knoppix, m23 und den Trinity-Desktop. Diese können ohne Verlust entfernt werden.

Software verwalten

Für die Installation und Aktualisierung von Software benutzt man das Programm `apt-get`:

- Informationen über installierbare Pakete aktualisieren:

```
# apt-get update
```

- Installierbare Pakete suchen:

```
# apt-cache search <Suchbegriff>
```

- Paket(e) installieren:

```
# apt-get install <Paket1> <Paket2>
```

- Installierte Paket(e) entfernen:

```
# apt-get remove <Paket1> <Paket2>
```

- Installierte Pakete aktualisieren:

```
# apt-get upgrade
```

- Installierte Pakete aktualisieren (falls notwendig werden hierbei zusätzliche Pakete installiert):

```
# apt-get dist-upgrade
```

Mit dem Debian Package Manager (dpkg [14]) kann man auch Paketdateien installieren, die sich im aktuellen Verzeichnis befinden:

```
# dpkg -i <Paket1.deb> <Paket2.deb>
```

Zur Erinnerung: mc (Midnight Commander) und gpm (Maus im Textmodus) könnte man jetzt endlich installieren, um den Komfort zu erhöhen.

Informationen über Pakete fragt man ebenfalls mit dpkg ab:

- Liste (de)installierter Pakete:

```
# dpkg --get-selections
```

- Liste der installierten Pakete, aufbereitet für die Benutzung mit **apt-get install**, z. B. für den Fall, dass man auf einem anderen System dieselben Pakete wieder installieren möchte:

```
# dpkg --get-selections | grep -v ^\sdeinstall$ | tr -d '[:blank:]' | sed \s/s/install$/g' | awk -v ORS=' ' '{\sprint("$0" " ")}'
```

- Alle Dateien in einem installierten Paket auflisten:

```
# dpkg -L <Paketname>
```

- Aus welchem installierten Paket kommt <Datei/Verzeichnis>?:

```
# dpkg -S <Datei/Verzeichnis>
```

- Informationen zu einem installierten Paket abfragen:

```
# dpkg -s <Paketname>
```

- Paketinhalte von verfügbaren Paketen aktualisieren (auf der Workshop-VM muss zuvor das Paket **apt-file** installiert werden):

```
# apt-file update
```

- Welches verfügbare Paket enthält <Datei/Verzeichnis>?:

```
# apt-file search <Datei/Verzeichnis>
```

Wurde die Installation von Paketen unterbrochen (z. B. wegen nicht erfüllter Abhängigkeiten nach einer manuellen Paketinstallation mittels **dpkg -i Paket.deb**), so kann diese mit

```
# apt-get install -f
```

wieder aufgenommen werden. In seltenen Fällen (meist, wenn in der Paketquellenliste Quellen kombiniert werden, die nicht zusammenpassen) kann der „Holzhammer“ nötig sein, um ein Paket dennoch zu installieren:

```
# dpkg -i --force-all Paketname.deb
```

Dies sollte aber nur der letzte Lösungsansatz sein, denn es kann dabei im schlimmsten Fall auch das System ernsthaft beschädigt werden.

Wurde das Paket mit APT heruntergeladen, so liegt es im Verzeichnis **/var/cache/apt/archives/**.

debconf

Einige Pakete (z. B. **gpm**) bringen Konfigurationsdialoge mit. Die dort eingegebenen Daten werden in der debconf abgelegt und dann von einem Skript im jeweiligen Paket verarbeitet.

Möchte man die Konfiguration eines Paketes nachträglich ändern, so geschieht dies über den Befehl

```
# dpkg-reconfigure <Paketname>
```

Die debconf kann mittels

```
# debconf-get-selections
```

(aus dem Paket **debconf-utils**) ausgegeben bzw. in eine Datei umgeleitet werden:

```
# debconf-get-selections > debconf.bak
```

Einzelne Werte oder auch die komplette deb-conf können mit dem Befehl

```
# debconf-set-selections <Debconf-Backup~  
-Datei>
```

wiederhergestellt werden, wobei sich die Datei an dieselbe Syntax wie die Ausgabe von **debconf-get-selections** halten muss.

Installation automatisieren

Betreut man viele Rechner oder Webserver, wird es lästig, da (Sicherheits-)Aktualisierungen regelmäßig eingespielt werden wollen. Für diese Zwecke gibt es Spezialisten wie z. B. das freie Softwareverteilungssystem m23 [15].

Möchte man die Systeme aber nur automatisch aktualisieren, reicht auch das Paket **cron-apt**. Nach der Installation muss es aber noch konfiguriert werden, da es ansonsten nur neue Pakete herunterlädt, aber nicht installiert. Im Verzeichnis **/etc/cron-apt/action.d** legt man dazu die folgenden vier Dateien mit dem folgenden Inhalt ab bzw. ändert deren Inhalt:

```
# Datei "0-update":  
update -o quiet=2
```

```
# Datei "3-download":  
autoclean -y  
upgrade -d -y -o APT::Get::Show-Upgraded~  
=true
```

```
# Datei "5-upgrade":  
upgrade -y
```

```
# Datei "9-notify":  
-q -q --no-act upgrade
```

Nun läuft cron-apt einmal täglich (bei Debian z. B. jeden Morgen um 4 Uhr), was aber auch in der Datei **/etc/cron.d/cron-apt** anders eingestellt werden kann.

In der Datei **/etc/cron-apt/config** stellt man bei den Parametern **MAILTO="<Mail>"** die E-Mail-Adresse desjenigen ein, der Informationen über den Abschluss der cron-apt-Aktionen zugeschickt bekommen soll und bei **MAILON="<Typ>"** die Art der Benachrichtigung. Hierbei kann **<Typ>** **always** sein, wenn man über jede Aktion informiert werden möchte oder **upgrade**, um nur dann informiert zu werden, wenn Pakete aktualisiert wurden.

Damit die Benachrichtigung funktioniert, muss allerdings ein funktionsfähiger Mailserver bzw. Mail Transport Agent (kurz MTA) auf dem System vorhanden sein.

LINKS

- [1] <http://www.freiesMagazin.de/freiesMagazin-2014-08>
- [2] http://de.wikipedia.org/wiki/filesystem_Hierarchy_Standard
- [3] <http://de.wikipedia.org/wiki/Initrd>
- [4] http://de.wikipedia.org/wiki/Grand_Unified_Bootloader
- [5] <http://de.wikipedia.org/wiki/SysVinit>
- [6] <http://de.wikipedia.org/wiki/Upstart>
- [7] <http://de.wikipedia.org/wiki/Systemd>
- [8] <http://de.wikipedia.org/wiki/Udev>
- [9] <http://de.wikipedia.org/wiki/Ssh>
- [10] http://de.wikipedia.org/wiki/Secure_Copy
- [11] <http://de.wikipedia.org/wiki/Rsync>
- [12] http://de.wikipedia.org/wiki/Advanced_Packaging_Tool
- [13] http://de.wikipedia.org/wiki/Debian_Free_Software_Guidelines
- [14] http://de.wikipedia.org/wiki/Debian_Package_Manager
- [15] <http://m23.sf.net>

Autoreninformation

Hauke Goos-Habermann und Marren Hachmann laden alle Leser ein, am 19. und 20. September 2014 die kostenfreien Vorträge und Workshops der Kieler Open Source und Linux Tage zu besuchen.



Teilen



Kommentieren

Spacewalk – Teil 2: Registrierung und Verwaltung von Systemen von Christian Stankowic

Im ersten Teil dieser Artikel-Serie in freiesMagazin 08/2014 [1] wurde die grundlegende Installation und Konfiguration von Spacewalk thematisiert. In diesem Artikel liegt der Fokus auf der Registrierung und Verwaltung von Linux-Systemen.

Registrierung von Client-Systemen

Damit Systeme mit Spacewalk verwaltet werden können, müssen entsprechende Client-Programme installiert und mit dem Spacewalk-Server registriert werden. Jedes registrierte System wird mit einem Systemprofil verknüpft, über welches dem Server jederzeit detaillierte System-Informationen, wie beispielsweise installierte Software-Pakete, zur Verfügung stehen. Weitere Client-Programme übernehmen die Kommunikation mit dem Spacewalk-Server und veranlassen die Fernwartung.

Unterstützung verschiedener Distributionen

Der Fokus von Spacewalk liegt auf RPM-basierenden Distributionen. Offiziell werden Fedora, openSUSE, CentOS, Scientific Linux und Debian/Ubuntu unterstützt. Letzteres wird derzeit jedoch nur lückenhaft unterstützt. So ist es z. B. nicht möglich, Systeme in Echtzeit zu verwalten. Normalerweise erfragen registrierte Linux-Systeme in periodischen Abständen ausstehende Aufgaben – mithilfe der Zusatzsoftware osad kann dieser Zeitraum drastisch verkürzt werden, da ausstehende Aktionen direkt über das Jabber-

Protokoll auf registrierten Systemen ausgeführt werden. Unter Debian sind die benötigten Client-Programme lediglich vollständig im "unstable" Distributionszweig vorhanden, vereinzelt Programme sind auch im stabilen Zweig zu finden. Ubuntu 14.04 LTS liefert die Pakete hingegen im stabilen Software-Zweig aus. Configuration Management ist derzeit unter Debian/Ubuntu nicht möglich.

Spacewalk Client-Pakete

Während die entsprechenden Client-Pakete unter Debian und Ubuntu fester Bestandteil der Distributor-Repositorys sind, müssen unter Fedora, Enterprise Linux und openSUSE zusätzliche Repositorys eingebunden werden.

■ Fedora 19:

```
# yum localinstall http://yum.spacewalkproject.org/2.2-client/Fedora/19/x86_64/spacewalk-client-repo-2.2-1.fc19.noarch.rpm
```

■ Fedora 20:

```
# yum localinstall http://yum.spacewalkproject.org/2.2-client/Fedora/20/x86_64/spacewalk-client-repo-2.2-1.fc20.noarch.rpm
```

■ Enterprise Linux 5:

```
# yum localinstall http://yum.spacewalkproject.org/2.2-client/RHEL
```

```
/5/x86_64/spacewalk-client-repo-2.2-1.el5.noarch.rpm
```

■ Enterprise Linux 6:

```
# yum localinstall http://yum.spacewalkproject.org/2.2-client/RHEL/6/x86_64/spacewalk-client-repo-2.2-1.el6.noarch.rpm
```

■ Enterprise Linux 7:

```
# yum localinstall http://yum.spacewalkproject.org/2.2-client/RHEL/7/x86_64/spacewalk-client-repo-2.2-1.el7.noarch.rpm
```

■ openSUSE 12.3:

```
# zypper ar -f http://download.opensuse.org/repositories/systemsmanagement:/spacewalk:/2.2/openSUSE_12.3/spacewalk-tools
```

■ openSUSE 13.1:

```
# zypper ar -f http://download.opensuse.org/repositories/systemsmanagement:/spacewalk:/2.2/openSUSE_13.1/
```

Unter Enterprise Linux wird zusätzlich das EPEL-Repository [2] benötigt. Damit können nun die folgenden Software-Komponenten installiert werden:

- RHN-Plugin für Zypper/YUM/APT – nahtlose Integration in den Paket-Manager der Distribution
- **rhnsd** – Red Hat Network Dämon, prüft in periodischen Abständen auf anstehende Aufgaben
- **rhn-check** – erfragt beim Spacewalk-Server ausstehende Aufgaben und führt diese umgehend aus
- **rhn-client-tools** – beinhaltet Handbuchseiten, Konfigurationen und Programme zur Kommunikation mit dem Spacewalk-Server
- **rhn-setup** – beinhaltet Programme zur Registrierung des Systems am Spacewalk-Server
- **rhncfg-actions** – beinhaltet Programme und Konfigurationen für das Spacewalk Configuration Management
- **osad** – Dienst zur Systemverwaltung in Echtzeit, Aufgaben werden über das Jabber-Protokoll gestartet (steht nicht unter Debian/Ubuntu zur Verfügung)

Je nach Distribution variieren hier die Befehle zur Installation der benötigten Pakete:

- Fedora:

```
# yum install -y rhn-{check,setup,client-tools} rhnsd yum-rhn-plugin
```

- Enterprise Linux und openSUSE:

```
# zypper install rhn-{check,setup,client-tools} zypp-plugin-spacewalk yum
```

- Debian Unstable und Ubuntu 14.04 LTS:

```
# apt-get install apt-transport-https spacewalk python-{rhn,ethtool} rhn{sd,-client-tools}
```

Aktivierungsschlüssel oder manuelle Registrierung

Ein System kann über zwei Wege mit dem Spacewalk-Server registriert werden. Eine Möglichkeit ist die Verwendung des Programms **rhn_register**, bei welchem gültige Login-Daten eines Administrators verwendet werden müssen. Im Laufe der Registrierung kann ausgewählt werden, welche System-Informationen hochgeladen werden sollen. Der Nachteil dieser Methode ist, dass sie interaktiv erfolgen muss. Sollen Systeme automatisiert installiert werden (z. B. mit Bare-Metal-Kickstart) empfiehlt sich die Verwendung eines Aktivierungsschlüssels.

Bei dieser Möglichkeit wird ein System ohne manuelle Eingaben mithilfe des Kommandos **rhnreg_ks** und eines Aktivierungsschlüssels registriert. Ein weiterer Vorteil gegenüber der manuellen Registrierung ist, dass bei der Verwendung eines Aktivierungsschlüssels zeitgleich zusätzliche Software-Kanäle und -Pakete zugewiesen werden können. Somit kann direkt mit der Registrierung eines Systems sichergestellt werden, dass grundlegende Software-Pakete installiert sind. Insbesondere bei der automatisierten Installation von Hosts ist diese Funktion hilfreich. Aktivierungsschlüssel müssen im Spacewalk-Backend generiert werden, bevor sie verwendet werden können.

Damit ein System erfolgreich registriert werden kann, müssen auf diesem Informationen zum Spacewalk-Server vorliegen. Diese Informationen werden in der Konfigurationsdatei **/etc/sysconfig/rhn/up2date** hinterlegt. Neben einigen vordefinierten Parametern sind vor allem die folgenden Einstellungen essenziell:

- **enableProxy**: Definiert, ob die Kommunikation zum Spacewalk-Server über einen Proxy-Server erfolgt.
- **serverURL**: Legt die URL zum XMLRPC-Interpreter des Spacewalk-Servers fest. Hier muss ein vollqualifizierter DNS-Name (FQDN) verwendet werden: <http://mein-spacewalk.meinedomain.loc/XMLRPC>.
- **proxyPassword**: Passwort, das zur Verwendung des Proxy-Servers benötigt wird.
- **proxyUser**: Benutzername, der zur Authentifizierung am Proxy-Server verwendet wird.
- **sslCACert**: Pfad zum SSL-Zertifikat des Spacewalk-Servers. Es muss eine Kopie der Datei auf dem zu registrierenden System vorliegen. Das SSL-Zertifikat kann vom Spacewalk-Server heruntergeladen werden: <http://mein-spacewalk.meinedomain.loc/pub/RHN-ORG-TRUSTED-SSL-CERT>. Standardmäßig wird die lokale Kopie des Zertifikats unter **/usr/share/rhn/RHN-ORG-TRUSTED-SSL-CERT** gespeichert.
- **httpProxy**: URL des zu verwendeten Proxy-Servers.
- **enableProxyAuth**: Definiert, ob eine Authentifizierung am Proxy-Server notwendig ist.

Die wichtigste Einstellung ist die URL des XMLRPC-Interpreters, hier muss in aller Regel lediglich der Hostname angepasst werden. Bei einer Registrierung mithilfe eines Aktivierungsschlüssels muss dieser als Parameter angegeben werden:

```
# rhnreg_ks -activation-key=...
```

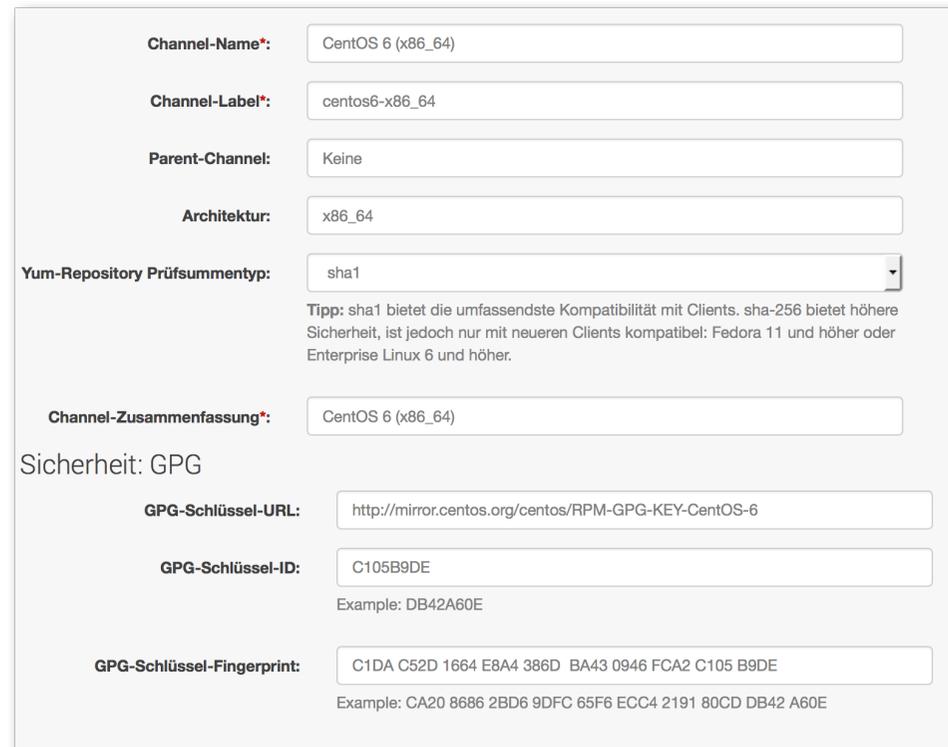
Nachdem ein System registriert wurde ist die eindeutige System-ID in der Datei `/etc/sysconfig/rhn/systemid` zu finden.

Software-Kanäle

Anschließend kann das System zwar schon grundlegend verwaltet werden, es fehlt jedoch noch die Möglichkeit Software zu verteilen. Damit Software-Pakete über Spacewalk verteilt werden können, müssen diese heruntergeladen und in einen Software-Kanal importiert werden. Spacewalk kann komplette APT- und YUM-Repositoryys über Netzwerk-Spiegel synchronisieren. Prinzipiell unterscheidet Spacewalk zwischen Parent- und Sub-Channels. Einem Parent-Channel können mehrere Sub-Channels untergeordnet sein, ein Sub-Channel stellt hier eine Teilmenge zusätzlicher Software-Pakete dar. Ein praktisches Beispiel hierfür ist die Aufteilung der CentOS 6 Software-Pakete:

- CentOS 6-Base: Basis-Kanal
- CentOS 6-Update: Untergeordneter Kanal, beinhaltet Updates
- CentOS 6-Extras: Untergeordneter Kanal, beinhaltet zusätzliche Pakete

Ein System kann immer nur einem Parent-Channel zugewiesen werden, man spricht hier auch von einem Basis-Software-Channel. Der Zugriff auf untergeordnete Software-Kanäle lässt sich pro System limitieren.



Channel-Name*:

Channel-Label*:

Parent-Channel:

Architektur:

Yum-Repository Prüfsummentyp:

Tipp: sha1 bietet die umfassendste Kompatibilität mit Clients. sha-256 bietet höhere Sicherheit, ist jedoch nur mit neueren Clients kompatibel: Fedora 11 und höher oder Enterprise Linux 6 und höher.

Channel-Zusammenfassung*:

Sicherheit: GPG

GPG-Schlüssel-URL:

GPG-Schlüssel-ID:
Example: DB42A60E

GPG-Schlüssel-Fingerprint:
Example: CA20 8686 2BD6 9DFC 65F6 ECC4 2191 80CD DB42 A60E

Beispiel eines Software-Kanals für CentOS 6 64-bit. 

Software-Kanäle lassen sich komfortabel über die Web-Oberfläche unterhalb „Channels → Software-Channels verwalten“ erstellen. Bei der Erstellung eines Software-Kanals müssen unter anderem folgende Informationen angegeben werden:

- Channel-Name: „lesbarer“ Name des Kanals (z. B. CentOS 6 Base)
- Channel-Label: eindeutiger Bezeichner des Kanals (z. B. centos-el6-base-x86_64)
 - Parent-Channel: Name des übergeordneten Kanals (sofern ein Sub-Channel)
- Architektur: Prozessor-Architektur der Software-Pakete (z. B. IA-32/i386, x86_64) – für Debian und Solaris gibt es gesonderte Einträge
- Channel-Zusammenfassung: kurze Zusammenfassung des Kanals (z. B. „CentOS 6 x86_64 Basis-Repository“)

Falls das zu synchronisierende Repository signiert wird, empfiehlt es sich zusätzlich noch Informationen zum GPG-Schlüssel zu hinterlegen. Diese Informationen finden sich entweder auf der Webseite des entsprechenden Repositorys oder unterhalb des Ordners `/etc/yum.repos.d`, sobald das Repository manuell auf einem System eingebunden wurde.

Oftmals fehlen jedoch Informationen zur GPG-Schlüssel-ID und dem dazugehörigen Fingerprint. Diese Informationen werden zur Verifizierung der Authentizität der Software-Pakete benötigt – mithilfe des Kommandos **gpg** lassen sich diese Informationen leicht aus einem bestehenden GPG-Key auslesen:

```
# gpg --with-fingerprint RPM-GPG-KEY-
CentOS-6
pub 4096R/C105B9DE 2011-07-03 CentOS-6
Key (CentOS 6 Official Signing Key) <
centos-6-key@centos.org>
Schl.-Fingerabdruck = C1DA C52D 1664
E8A4 386D BA43 0946 FCA2 C105 B9DE
```

C105B9DE entspricht der GPG-Schlüssel-ID, der längere Schlüssel am Ende stellt den Fingerabdruck dar.

Sobald ein Software-Kanal angelegt wurde, muss noch ein entsprechendes Repository erstellt und mit dem Kanal verknüpft werden. Repositories werden komfortabel über die Web-Oberfläche unterhalb des Menüs „Channels → Software-Channels verwalten → Repositorien verwalten“ erstellt. Bei der Erstellung eines Repositorys werden lediglich ein eindeutiger Bezeichner und eine gültige URL benötigt. Anschließend muss der kürzlich angelegte Software-Kanal editiert werden, unterhalb „Repositorien“ muss das soeben angelegte Repository ausgewählt werden.

Anschließend lassen sich die Software-Pakete mit **spacewalk-repo-sync** synchronisieren:

```
# spacewalk-repo-sync --channel centos-
el6-base-x86_64
```

Es existiert ein Programm namens **spacewalk-common-channels**, um die Erstellung von Software-Kanälen zu vereinfachen. Zur Installation wird das gleichnamige Paket über den Paket-Manager der verwendeten Distribution ausgewählt. Das Skript dient zur automatischen Erstellung zahlreicher bekannter Software-Kanäle. Unter anderem werden derzeit unterstützt:

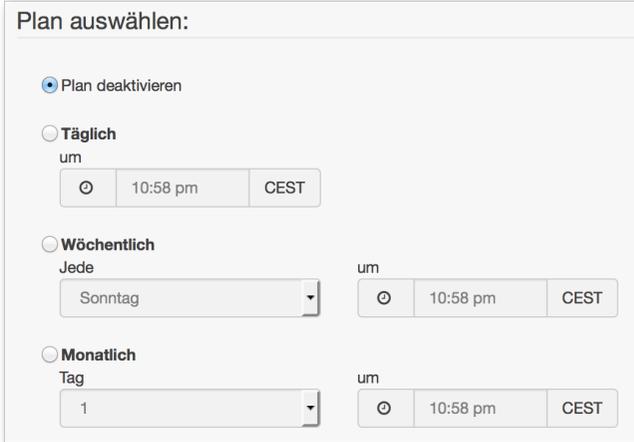
- CentOS und EPEL 5 - 7
- Fedora 19, 20
- openSUSE 12.3, 13.1
- Oracle Linux 5, 6

Möchte man beispielsweise die beiden CentOS 6 Software-Kanäle (Basis- und Update-Pakete) für 64-bit-Systeme konfigurieren, kann man sich mit den folgenden Befehlen manuelle Arbeit ersparen:

```
$ spacewalk-common-channels -u admin -a
x86_64 'centos6'
$ spacewalk-common-channels -u admin -a
x86_64 'centos6-updates'
```

Anschließend sind sowohl Software-Kanäle als auch Repositories konfiguriert und können synchronisiert werden.

Es empfiehlt sich, diese Synchronisation regelmäßig ausführen zu lassen, beispielsweise jede



Auswählen eines Zeitpunkts der Synchronisation. 🔍

Nacht. Eine Möglichkeit ist es, den Aufruf **spacewalk-repo-sync** in einem Cronjob ausführen zu lassen. Spacewalk verfügt jedoch ebenfalls über eine eigene Job-Steuerung, die auch verwendet werden kann. Um eine automatische Kanal-Synchronisierung zu konfigurieren muss der entsprechende Software-Kanal editiert werden. Unterhalb des Menüs „Repositorien → Sync“ kann eingestellt werden, wie häufig ein Netzwerk-Spiegel synchronisiert werden soll (z. B. täglich, wöchentlich, monatlich)

Systemverwaltung

Über die Web-Oberfläche lassen sich registrierte Systeme komfortabel konfigurieren, ohne auf Protokolle, wie z. B. SSH, zurückgreifen zu müssen.

Zu den ausführbaren Verwaltungsaufgaben zählen unter anderem:

- Ausführen von Shell-Befehlen
- Migrieren eines Hosts in eine Organisation (z. B. untergeordnete Tochter-Gesellschaft)
- Installieren, Aktualisieren oder Entfernen von Software-Paketen
- Überprüfen und Übernehmen von zentral verwalteten Konfigurationsdateien
- Neuinstallation des Systems über das Netzwerk (Kickstart)

Der volle Umfang der Verwaltungsaufgaben steht nur zur Verfügung, wenn dem Host die „Provisioning“-Berechtigung (Entitlement) erteilt wurde – diese lässt sich in den Systemeigenschaften des entsprechenden Systems in der Spacewalk-Weboberfläche einstellen. Es ist auch erforderlich, dass auf dem Client-System die Berechtigungen für das Überprüfen und Ausrollen von Konfigurationsdateien erteilt wurden. Dies wird mit dem folgenden Aufruf sichergestellt:

```
# rhn-actions-control --enable-all
```

System-Set-Manager

Der System-Set-Manager dient dazu, logisch gruppierte Hosts wie einen einzelnen Host zu verwalten. So könnte man beispielsweise alle vorhandenen Web-Server in einer Gruppe zusammengefasst zeitgleich aktualisieren und konfigurieren. Insbesondere bei großen Systemlandschaften erspart diese Funktion viel Zeit. Gruppen lassen sich im Menü „Systeme → Systemgruppen“ erstellen, anschließend können einzelne Hosts ausgewählt und der angelegten Gruppe hinzugefügt



System	Updates	Errata	Pakete	Konfigurationen	Abstürze	Zuletzt angemeldet	Basis-Channel	Berechtigung
stankowic.loc	1	3	5	2	(keine)	vor 3 Minuten	CentOS 6 Base - x86_64	Management, Überwachung, Provisioning
stankowic.loc	1	3	12	1	(keine)	vor 3 Minuten	CentOS 6 Base - x86_64	Management, Überwachung, Provisioning

Beispiel des System-Set-Managers mit zwei ausgewählten Systemen. 

werden. Es ist möglich Hosts mehreren Gruppen hinzuzufügen. Für mich hat es sich als nützlich erwiesen, Hosts nach Applikationen (z. B. Datenbank, Web, ERP) und Gewichtung (z. B. Test, Entwicklung, Produktion) zu gruppieren. So kann man anstehende Patches beispielsweise erst auf Test-Servern und anschließend auf Entwicklungssystemen ausrollen, bevor kritische Systeme aktualisiert werden.

Möchte man nun mehrere Systeme verwalten, kann man entweder im Menü „Systemgruppen“ die entsprechende Gruppe oder in der Systemübersicht einzelne Hosts auswählen und zum System-Set-Manager hinzufügen. Nun lassen sich sämtliche Verwaltungsaufgaben auf den ausgewählten Hosts ausführen.

Bei der Auswahl mehrerer Gruppen lässt sich entweder mit einer Schnittmenge (Intersection, lediglich Hosts, die in allen ausgewählten Gruppen

vorhanden sind) oder einer Vereinigung (Union, Summe aller Hosts aus allen ausgewählten Gruppen) arbeiten.

Aktionen

Sämtliche über Spacewalk ausgeführte Aufgaben werden unterhalb des Menüpunkts „Plan“ als Aktionen zusammengefasst. In vier Kategorien

können geplante und ausgeführte Aktionen überwacht werden:

- Ausstehende Aktionen
- Fehlgeschlagene Aktionen
- Abgeschlossene Aktionen
- Archivierte Aktionen

Ausstehende, fehlgeschlagene oder abgeschlossene Aufgaben können zu Dokumentationszwecken archiviert werden – diese Aufgaben werden dann in der vierten Kategorie angezeigt. Es empfiehlt sich gelegentlich ältere Aufgaben zu entfernen, um die Größe der Spacewalk-Datenbank (und somit auch die Performance) zu regulieren.

Configuration Management

Spacewalk kann zentral symbolische Verknüpfungen, Konfigurations- und Binärdateien verwalten und auf Systemen platzieren. Zentral verwaltete Dateien werden einem Konfigurationskanal

Filtern nach Channel-Name: 

<input type="checkbox"/>	Channel-Name	Channel-Label	Dateien insgesamt	Einsetzbare Dateien*	Position
<input type="checkbox"/>	 stankowic-lan	stankowic-lan	1 Datei	1 Datei	1
<input type="checkbox"/>	 stankowic-generic	stankowic-generic	3 Dateien	3 Dateien	2

1 - 2 von 2(0 ausgewählt)

Beispiel für ein System mit mehreren zugewiesenen Konfigurationskanälen. 

zugeordnet. Ein System kann einem oder mehreren Konfigurationskanälen zugewiesen werden. So können beispielsweise je nach Applikation entsprechende Konfigurationskanäle erstellt werden. Wenn ein System mehreren Konfigurationskanälen zugewiesen wird, muss eine Gewichtung der einzelnen Kanäle festgelegt werden. Liegt eine Datei in mehreren Konfigurationskanälen vor, wird die Konfigurationsdatei aus dem am höchsten priorisierten Konfigurationskanal platziert.

Konfigurationsdateien können über die Web-Oberfläche von Spacewalk erstellt werden (Menü „Konfiguration“). Für jede Konfigurationsdatei werden die folgenden Informationen definiert:

- Dateiname/-Pfad
- Besitzer und Gruppenzugehörigkeit

- Dateiberechtigungs-Modus
- SELinux-Kontext (sofern relevant)
- Benutzerdefinierte Makro-Trennzeichen (ggf. je nach Anwendung notwendig, um die Syntax der Konfigurationsdatei zu erhalten)

Über Makros lassen sich Informationen des Systems, auf dem die Konfigurationsdatei platziert werden soll, integrieren. Hier lohnt sich ein Blick in die Red Hat Satellite-Dokumentation [3], die die einzelnen Makros beispielhaft erläutert.

Alternativ lassen sich bereits vorhandene Konfigurationsdateien hochladen, anstatt sie über den integrierten ASCII-Editor einzugeben.

Zentral verwaltete Konfigurationsdateien werden mit einer Prüfsumme und einer Revisionsnum-

mer versehen, sodass Änderungen nachvollzogen werden können. Ältere Revisionen können eingesehen und verglichen werden. Es erfolgt keine automatische Platzierung aktualisierter Konfigurationsdateien. Werden Konfigurationsdateien aktualisiert, müssen diese manuell über Spacewalk auf den Systemen platziert werden. Über die Web-Oberfläche können die Inhalte von Konfigurationskanälen komfortabel auf einzelnen oder allen subscribierenden Systemen ausgerollt werden.

Systeme lassen sich nun zentral konfigurieren und mit Software-Paketen versehen – der nächste Teil dieser Artikel-Serie wird sich ganz der Automatisierung von Installationen (Kickstart) widmen.

LINKS

- [1] <http://www.freiesMagazin.de/freiesMagazin-2014-08>
- [2] <https://fedoraproject.org/wiki/EPEL> 
- [3] <http://red.ht/1qrMYbB> 

Autoreninformation

Christian Stankowic ([Webseite](#))

beschäftigt sich seit 2006 mit Linux. Nach seinen privaten Erfahrungen mit Debian, CRUX und ArchLinux widmet er sich seit seiner Fachinformatikerausbildung insbesondere RHEL, CentOS und Spacewalk.

 [Teilen](#)

 [Kommentieren](#)

R-Webapplikationen mit Shiny und OpenCPU von Markus Herrmann

Moderne Webapplikationen müssen nicht auf ein statistisches Backend verzichten. Mit Shiny von RStudio und dem OpenCPU-Projekt von Jeroen Ooms werden zwei populäre Frameworks vorgestellt, um das Statistikprogramm R in Webseiten einzubinden.

Dass mathematisch-statistische Anwendungen nicht nur auf schwarz-weiße Konsolenapplikationen beschränkt sind, wird in nahezu jedem Studien- oder Berufsfeld deutlich, welches auf diese Werkzeuge zurückgreifen muss. Als bekannte Vertreter mathematisch-statistischer Software mit graphischer Benutzeroberfläche wären hier beispielsweise die kommerziellen Programme Matlab, SAS, SPSS oder Stata zu nennen. Die freien Vertreter Julia, Python oder R sind aber nicht minder populär.

R-Applikationen

Insbesondere die Programmiersprache R entwickelt sich zunehmend zur Lingua franca der Statistik. Die Software ist kostenlos, läuft auf allen gängigen Systemen und ist leicht zu installieren bzw. in bestehende Systeme zu integrieren [1].

R kann auf Konsolenebene oder in Verbindung mit graphischen Benutzeroberflächen oder Erweiterungen für gängige Texteditoren genutzt werden. Eine der bekanntesten Benutzeroberflächen ist RStudio IDE vom Hersteller RStudio [2].

Weiterhin sind inzwischen über 6700 zusätzliche R-Pakete („R-packages“) mit weit über 130000 Funktionen dokumentiert [3], die über das „Comprehensive R Archive Network“ (CRAN) [4] bezogen werden können.

Eine Vielzahl dieser Pakete enthält auch APIs für die verschiedensten Webdienste (Twitter, Google, Amazon Web Services etc.). Diese Programmierschnittstellen ermöglichen beispielsweise direkte Abfragen von Tweets, Google-Suchergebnissen oder Google-Geocoding-Daten, aber auch die Steuerung ganzer Cluster in der Amazon Cloud [5].

RStudio Server

Der Remote-Zugriff auf R selbst ist ebenfalls komfortabel realisierbar. Die Anwendung RStudio Server, welche eine browserbasierte Schnittstelle für R in Form der RStudio IDE bereitstellt, erfreut sich dabei zunehmender Beliebtheit. Mit dieser Software ist es dem R-Anwender möglich, über den Browser R-Code oder -Skripte auf einem entfernten System – dem RStudio Server – auszuführen.

Der Hersteller stellt RStudio Server als quelloffene Version als Code [6] oder Installationspakete [7] bereit. Eine kommerzielle Version mit erweitertem Umfang und Support ist ebenfalls erhältlich.

RStudio Server lauscht nach der Installation auf HTTP-Port 8787 und eine bereits implementierte

PAM-Authentifizierung ermöglicht die Anmeldung mit dem Benutzernamen und Passwort des Systems. Für den Benutzer macht es dann keinen Unterschied, ob er vor einem lokalen oder entfernten System sitzt. Dieser Umstand erleichtert zwar auf den ersten Blick eine Absicherung der Applikation, öffnet aber auch – wahlweise mit Benutzer- oder Administratorrechten – Tür und Tor zum System (z. B. mit dem Aufruf `system()`). Es ist aber alternativ möglich, die PAM-Authentifizierung abzuschalten und den RStudio Server hinter einem Reverse-Proxy (z. B. Apache oder nginx) durch einen (anderweitig authentifizierten) Benutzer mit eingeschränkten Rechten auszuführen.

Im Allgemeinen beschränkt aber der codebasierte Ansatz der Programmnutzung die Anwendergruppe von RStudio Server weiterhin auf Benutzer mit Kenntnissen der Programmiersprache R, sowie auf PC- oder Laptopnutzer, da die sinnvolle Nutzung mit dem Smartphone oder Tablet aufgrund des statischen Layouts der Anwendung schwerlich vorstellbar ist. RStudio Server ist aber auch nicht als Webapplikation konzipiert, sondern als Web-IDE für R.

Shiny

Eine Alternative zur codebasierten R-Nutzung bieten moderne HTTP- bzw. Web-APIs zu R, die es ermöglichen, R in Webapplikationen zu integrieren und die Inhalte geräteunabhängig (responsive) darzustellen.

Mit Shiny stellt RStudio ein Produkt zur Verfügung, das die Anwendung von R-Applikationen auch für Endnutzer interessant macht, welche die Programmiersprache R nicht beherrschen [8]. Es war zwar schon vor Shiny möglich, mit verschiedenen R-Paketen eine grafische Benutzeroberfläche für R-Applikationen zu entwickeln, doch der Nachteil dieser Lösungen besteht in zahlreichen R-Paket- und Softwareabhängigkeiten, die auf dem ausführenden Zielsystem vorhanden sein müssen.

Das R-Package `shiny` hingegen erlaubt die komfortable Erstellung einer lokalen, browserbasierten und interaktiven Applikation auf Basis von R-Code. Eine Vielzahl an Beispielanwendungen kann auf der RStudio Webseite innerhalb der „Shiny gallery“ ausprobiert werden [9].

Die jeweils neueste freie Version von Shiny ist über das Github Repository von RStudio zu bekommen [10]. Das R-Package `shiny` kann aber auch innerhalb der R-Konsole mittels `install.packages(„shiny“)` installiert werden.

RStudio wirbt zwar damit, dass HTML-, CSS- oder JavaScript-Kenntnisse zur Erstellung der Applikation nicht notwendig sind, allerdings werden sie durchaus nützlich, wenn auch die serverseitige Variante „Shiny Server“ zur Generierung von R-Webapplikationen genutzt werden soll.

Shiny Server

Shiny Server ist im Kern eine `node.js`-Webapplikation [11], d. h. eine javascriptbasierte

Anwendung mit integriertem Webserver. Shiny Server setzt hierbei entweder auf eine vorhandene `node.js`-Umgebung auf oder bringt sie bei seiner eigenen Installation mit.

Shiny Server ist als daher nur folgerichtig auch als „node packaged module“ (`npm`-Modul) erhältlich und kann bei einer bereits vorhandenen `node.js`-Umgebung auf der Konsole mit dem `npm`-Paketmanager installiert werden [12]. Über die RStudio-Webseite sind weiterhin distributionspezifische Installationspakete, sowie eine kommerzielle Serverversion erhältlich [13].

Freie Version = verminderter Umfang

Die freie Version des Shiny Server ist (ähnlich zur RStudio Server-Distribution) gegenüber der „Pro-Version“ im Umfang deutlich eingeschränkt. So sind in der freien Version keinerlei Möglichkeiten vorhanden, die Applikation mittels einer Benutzerverwaltung oder HTTPS abzusichern.

In der Standardkonfiguration lauscht Shiny Server ungesichert auf `HTTP`-Port 3838 und sollte daher nicht ohne weitere Modifikation über das Internet erreichbar sein. Dieser Umstand soll aber nicht vor einem Einsatz des Servers abschrecken, denn einerseits kann die Verbindung mit dem Shiny Server über einen `ssh`-Tunnel aufgebaut werden, andererseits sollte es dem geübten JavaScript-Entwickler mit einigen Handgriffen gelingen, den `node.js`-Webserver so anzupassen, dass die Socketverbindungen auch hinter einem Reverse-Proxy mit `SSL` funktionieren.

Die Benutzerverwaltung und Authentifizierung ließe sich dann zum Beispiel durch die Einbindung einer Mongo-Datenbank über die `npm`-Module `mongoose` und `passport` [14] oder über eine Vielzahl anderer Datenbanken und Authentifizierungsmethoden realisieren. Zur Basisauthentifizierung kann natürlich auch `.htaccess` verwendet werden.

„Hallo Shiny!“

Shiny-Inhalte werden im Browser dargestellt, wobei die Programmierung dieser Inhalte bei einer generischen Shiny-Applikation über zwei R-Skripte (`ui.R` und `server.R`) erfolgt, die im vordefinierten Webroot-Verzeichnis des Shiny Servers abgelegt werden müssen.

Das folgende „Hallo Welt“-Beispiel zeigt den Inhalt der Datei `server.R` und die dazugehörige Datei `ui.R`. Im R-Skript `server.R` erfolgt der eigentliche R-Aufruf. Hierbei wird ein Histogramm über normalverteilte Zufallszahlen generiert, wobei die Anzahl der Züge vom Objekt `input$obs` abhängt, welches in der Layout-bestimmenden Datei `ui.R` als interaktiver „Slider“-Input mit einem Wertebereich zwischen 1 und 1000 und einem Standardwert von 500 definiert wurde.

```
library(shiny)
shinyServer(function(input, output) {
  output$distPlot <- renderPlot({
    dist <- rnorm(input$obs)
    hist(dist)
  })
})
```

Listing 1: `server.R`

```
library(shiny)
shinyUI(pageWithSidebar(
  headerPanel("Hallo Welt Histogramm!"),
  sidebarPanel(
    sliderInput("obs",
      "obs:",
      min = 1,
      max = 1000,
      value = 500)
  ),
  mainPanel(
    plotOutput("distPlot")
  )
))
```

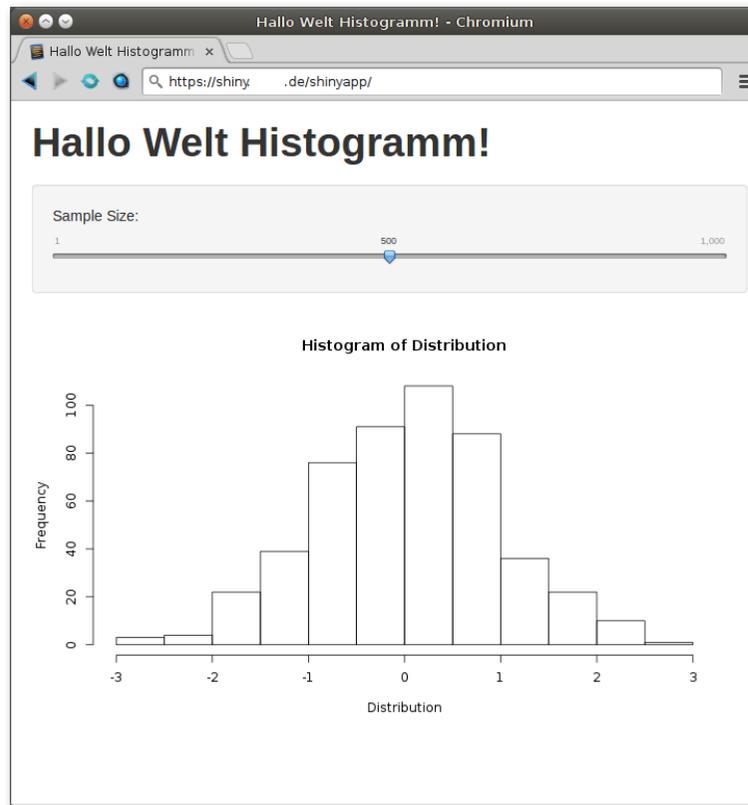
Listing 2: ui.R

Mit der Einstellung dieser beiden Skripte in das Shiny-Webroot-Verzeichnis ist das Deployment für das „Hallo-Welt“-Beispiel fertig, das Ergebnis ist unter <http://localhost:3838> abrufbar.

Responsive & Reactive

Zur responsiven Darstellung der Inhalte auf unterschiedlichen Endgeräten wird hierbei unter anderem das Bootstrap-Framework (in diesem Zusammenhang nicht zu verwechseln mit der statistischen Methode Bootstrapping) verwendet [15]. Es können aber auch eigene CSS-Stile implementiert werden [16].

Die interaktiven Komponenten – von RStudio „reactive functions“ genannt – werden über Websocketverbindungen realisiert. Eine Integration von Shiny-Applikationen in eigene JavaScript-Applikationen ist zwar auch möglich, allerdings



Shiny: Histogramm mit Input-Slider. 

gibt es hierfür bislang lediglich eine experimentelle Demo [17] und keinen offiziellen JavaScript-Client.

OpenCPU Local Single-User-Server

Das OpenCPU-Framework von Jeroen Ooms [18] ist als universelle HTTP-API zu R zu sehen. OpenCPU bietet dazu eine „Mapping-Funktionalität“, die Parameter, die mit den HTTP-Methoden GET und POST übergeben wurden, auf R-Funktionen,

Skripte oder Objekte anwendet. Während GET zum Anfordern von Ressourcen verwendet wird, wird POST zum Aufruf von R-Funktionalitäten benutzt.

OpenCPU wurde quelloffen unter der Apache2-Lizenz veröffentlicht und ist als lokaler „Single User Server“ in Form des R-Package **opencpu** und als „Cloud Server Version“ erhältlich.

Die lokale Variante von OpenCPU kann innerhalb von R über CRAN installiert werden **install.packages("opencpu")**. Nach dem Laden des Pakets mittels **library(opencpu)** wird ein Single-User-Server gestartet und die Server-Adresse in der R-Konsole ausgegeben.

OpenCPU Cloud Server

Der OpenCPU Cloud Server ist im Gegensatz zur lokalen Variante öffentlich erreichbar und für eine Mehrfachnutzung geeignet. Die Kommunikation mit dem Benutzer kann hierbei über ein Frontend oder geeignete HTTP-Methoden erfolgen. Eine lokale R-Umgebung ist nicht erforderlich.

Installation

Als Unterbau für den OpenCPU Cloud Server wird vom Projekt ausdrücklich Ubuntu Server 14.04 empfohlen. Nach der Registrierung der OpenCPU-Paketquellen durch die Eingabe von

```
# add-apt-repository ppa:opencpu/opencpu~
-1.4
```

und nach einem Update der Paketquellen kann der OpenCPU Cloud Server mit dem APT-Paketmanager installiert werden

```
# apt-get install opencpu
```

Der Quellcode zum Bau eines eigenen Installationspakets, distributionsspezifische Installationspakete, Dokumente mit detaillierten Installations- und Bedienungsanleitungen, sowie zahlreiche Aufruf- und Codebeispiele sind über das Github-Repository von Jeroen Ooms erhältlich [19].

Funktionalität

OpenCPU folgt im Frontend einem ähnlichen Ansatz – den Zugriff auf R und die Darstellung von R-Inhalten im Browser zu ermöglichen – wie Shiny, erhebliche Unterschiede finden sich aber im Backend. Hier nutzt OpenCPU den Webserver Apache und als Cacheserver nginx, um mit der Außenwelt zu kommunizieren. In der Standardeinstellung lauscht OpenCPU sowohl auf HTTP-Port 80 als auch auf HTTPS-Port 443 im Webroot-Verzeichnis `/opencpu`. Die Einstellungen können Apache-typisch aber angepasst und gesichert werden.

Generell kann jegliche Software, die HTTP-Aufrufe absenden und deren Rückgaben empfangen kann, über OpenCPU auch R-Funktionen und R-Skripte aufrufen, ohne den R-Code generieren

oder interpretieren zu müssen. Ein Webentwickler kann also, ohne eine Zeile R-Code verstehen zu müssen, R-Funktionen oder -Skripte aufrufen und deren Ergebnisse im Browser darstellen.

In eingeschränkter Form sind derartige R-Funktionsaufrufe u. a. auch über low-level APIs (z. B. mit den R-Packages **Rserve** oder **R-DCOM**) oder über einen (für den reinen R-Programmierer im Normalfall relativ umständlichen) Aufruf von Skriptlösungen mit PHP oder CGI (z. B. R-Package **rApache**) möglich. OpenCPU bietet aber neben der universellen HTTP-Schnittstelle, auch einen offiziellen JavaScript-Client [20].

Die OpenCPU-JavaScript-Bibliotheken müssen hierfür zunächst in den HTML-Code eingebunden werden, der R-Aufruf erfolgt dann entweder Inline oder mit einer separaten JavaScript-Datei. Zusätzlich gibt es mit dem npm-Modul `node-opencpu` die Möglichkeit, OpenCPU innerhalb einer `node.js`-Applikation anzusprechen [21].

Eine Authentifizierung oder Benutzerverwaltung ist für das genuine OpenCPU-Framework zunächst nicht nötig, da der Zugriff auf die OpenCPU API-Endpunkte auch von einem weiteren gesicherten Frontend mittels „cross-domain opencpu requests“ (CORS) über Domaingrenzen hinweg erfolgen kann. Zu beachten ist hierbei, dass in der Standardeinstellung die „Same Origin Policy“ für jeden Server umgangen wird, d. h. es wird ein „*“ im „Access-Control-Allow-Origin“-Header gesetzt [22].

Hallo OpenCPU!

Das folgende Beispiel zeigt die Systematik eines OpenCPU-Calls in JavaScript. Zunächst werden mit der Funktion `call1` 50 gleichverteilte Zufallszahlen erzeugt und mit Hilfe der Chart-Bibliothek **Chart.js** geplottet [23]. Anschließend werden die erzeugten Zufallszahlen nochmals an R übergeben, um einen generischen R-Plot zu erhalten.

```
function call1(){
  ocpu.seturl("//ocpu-server/ocpu/~
  library/stats/R");
  ocpu.call("runif", {n: 50, min: 0, max~
  : 10},
  function(session){
    session.getObject(function(data){
      create(data);
      getPlot(data);
    });
  });

  ocpu.seturl("//ocpu-server/ocpu/library/~
  graphics/R")
  function getPlot(source){
    var req = $("#plotdiv").rplot("~
    boxplot", {
      x : source
    });
  };
};
```

Listing 3: `opencpu.js`

Mit `ocpu.seturl` wird der OpenCPU Cloud Server (der in diesem Fall auf einer anderen Domain zu finden ist) und die benötigte R-Library **stats**

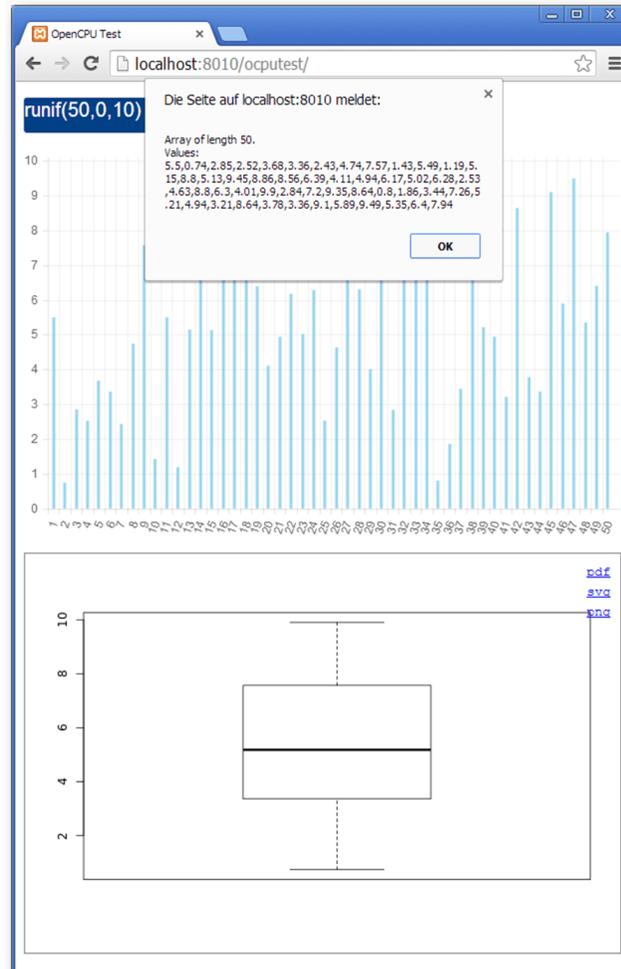
referenziert. Der anschließende `ocpu.call` ruft dann die R-Funktion `runif` auf und erhält ein Array mit 50 gleichverteilten Zufallszahlen im Wertebereich zwischen 0 und 10 zurück. Die Funktion `getPlot` übergibt dann dieses Array als Vektor an die neu referenzierte R-Library `graphics`, um einen Boxplot zu erhalten. Im Anschluss muss das Array (`data`) nur noch an die Chartengine übergeben und gerendert werden. Der Boxplot (`#plotdiv`) kann direkt als DIV-Element in die Webseite eingebunden werden.

Speicher- und Kommunikationssteuerung

Statt einer erneuten Übergabe des Ergebnisvektors an R besteht auch die Möglichkeit, das Zwischenergebnis direkt und ohne erneuten Schreibvorgang als Input für die Boxplot-Funktion zu nutzen. OpenCPU ermöglicht dies durch einen „functional state“ in der Speicher- und Kommunikations-Ablaufsteuerung. Dieser Zustand ist am besten durch das Aufzeigen einiger Unterschiede von zustandsloser (`stateless`) und zustandsbehafteter (`stateful`) Kommunikation zu erklären.

Stateless Communication

Als `stateless` wird eine sequenzielle Request-Response-Abfolge mit Unabhängigkeit zwischen Request- und Responseanforderungen bezeichnet. Zu dieser Art von Anwendungen gehören bspw. Skriptlösungen mittels PHP oder CGI mit `rApache`, die Ergebnisse in eine Datenbank schreiben und anschließend wieder auslesen. `Stateless`-Skriptlösungen sind meist auch für eine Mehrfachnutzung zur gleichen Zeit ausgelegt.



OpenCPU: Histogramm mit externer Chartengine und generischer R-Boxplot. 

Stateful Communication

`Stateful`-Lösungen wie Shiny hingegen steuern den Datenfluss über persistente, bidirektionale Websocketverbindungen. Diese Verbindungen erlauben dadurch, dass sich Client und Server je-

derzeit Daten senden können, ohne eine Verbindung erneut aufbauen zu müssen, eine nahezu echtzeitfähige, interaktive Benutzung. Allerdings funktioniert dies nicht mit mehreren Benutzern gleichzeitig innerhalb derselben Session. Für jede neue Shiny-Session muss daher ein separater R-Prozess – mit der damit verbundenen Speicherallokation – gestartet und so lange aufrecht erhalten werden, bis die Shiny-Session beendet wird.

Dabei besteht durch die Abhängigkeit zwischen Request- und Responseanforderung außerdem die Gefahr, dass ein eingefrorener R-Prozess die gesamte Webapplikation zum Absturz bringt, da jeder neue eingehende Request erst auf die Abarbeitung vorheriger Abfragen warten muss, bevor er beantwortet wird.

Functional State in OpenCPU

OpenCPU nutzt nun eine Mischform zwischen zustandsbehafteter und zustandsloser Kommunikation, den `functional state`. Hierbei bleiben angelegte Objekte persistent, nicht aber deren Prozesse. Dieser Zustand wird dadurch erreicht, dass OpenCPU als Mapper zwischen den im HTTP-Aufruf übergebenen Parametern und der R-Funktion agiert und die Ergebnisse eines jeden Aufrufs zusammen mit einem privaten temporären Session-Key zwischenspeichert und zurückgibt.

Dieser temporäre Schlüssel ist im weitesten Sinne mit einer Variable im R-Workspace vergleichbar. Die Ergebnisse eines vorherigen Rechenschrittes können so zurückgegeben und auch in der nächs-

ten Prozedur verwendet werden. Eine interaktive Applikation ließe sich somit als sequentielle Abfolge dieser „remote procedure calls“ (RPC) realisieren, bei der die Schlüssel konsekutiv aufeinander referenzieren. Insbesondere den „asynchron sprechenden“ Clients (JavaScript, node.js) kommt dieser – eigentlich zustandslose – Umstand bei Callbacks, die ausgeführt werden, sobald das angeforderte Ergebnis vorliegt, entgegen.

Weiter können durch die eindeutige Zuordnung des temporären Schlüssels zu Benutzer und RPCs auch mehrere Benutzer gleichzeitig und unabhängig voneinander, oder auch mehrere Benutzer in einer gemeinsamen Session, den Server nutzen, ohne dass eine gesonderte Benutzerverwaltung oder Authentifizierung notwendig ist.

Ein Nachteil davon ist allerdings, dass der temporäre Schlüssel nicht verändert oder überschrieben werden kann. Wird ein Objekt verändert, so wird eine Kopie des Objekts verändert und ein neuer Schlüssel erstellt. Objekte können natürlich zu jeder Zeit auch hoch- oder heruntergeladen werden, wobei man hochgeladene Daten direkt als Input an Funktionen oder Skripte übergeben kann [24].

Einbindung externer JavaScript-Bibliotheken
Zudem besteht bei einer Nutzung von OpenCPU auch immer die Möglichkeit, externe JavaScript-Statistik-Bibliotheken wie jStat [25], numbers [26], gauss [27] oder science [28] unkompliziert einzubinden. Mit diesen Werkzeugen wäre eine Client-Serverseitige-Arbeitsteilung (z. B. zur Ausführung

eines ungeschützten bzw. schützenswerten Codes/Skripts) durchaus praktikabel, insbesondere wenn es sich um eine umfangreiche statistische Webapplikation handelt.

Nutzbarkeit

Die Frage, welche Applikation bzw. welches Framework für ein bestimmtes Projekt am besten geeignet ist, bleibt in diesem Artikel unbeantwortet, da sich die vorgestellten Lösungen im Kern zu stark voneinander unterscheiden. Ein kurzer Überblick, unter Beachtung von Umfang, Anwenderfreundlichkeit und Sicherheit, soll nun helfen, eine Antwort für das eigene Projekt zu finden.

RStudio Server

Sollen ausschließlich R-Code oder -Skripte in entfernter Umgebung ausgeführt und editiert werden, erhält der Anwender bei einer Nutzung des RStudio Server generell die umfangreichsten Möglichkeiten. Hierfür sind allerdings R-Kenntnisse notwendig, es sei denn, man beschränkt sich auf die Ausführung fertiger Funktionen oder Skripte.

Zur Nutzung von Auto-Reporting-Tools oder Anwendungen, die Ergebnisse in Datenbanken oder festgelegte Verzeichnisse speichern, mag dies vollkommen ausreichend sein, allerdings könnte eine derartige Funktionalität auch mit serverseitigen Skriptlösungen erreicht werden. Hierfür ist allerdings ein Web-Entwickler bzw. Serveradministrator gefragt, der die Skriptlösung (z. B. PHP oder CGI) programmiert und implementiert. Dies ist für die Inbetriebnahme der RStudio Serverlö-

sung aufgrund der ausführlichen Dokumentation zunächst nicht notwendig, für einen sicheren Betrieb des Servers sind Grundkenntnisse in der Serveradministration aber mehr als vorteilhaft.

Betont werden muss, dass RStudio Server auf Grund seiner Konzeption als Web-IDE für R nicht als Webapplikation im ursprünglichen Sinne bezeichnet werden kann und sich daher auch nur sehr schwer mit Shiny oder OpenCPU vergleichen lässt. Die Anwendung kann aber gerade bei der Entwicklung von R-Webapplikationen hilfreiche Unterstützung bei der serverseitigen Anpassung von R-Code oder beim Debugging leisten.

Shiny

Shiny- bzw. Shiny Server-Lösungen erscheinen für die Erstellung einfacher Browser- und Webapplikationen als erste Wahl. Die Bezeichnung „einfach“ bezieht sich hierbei allerdings nicht auf die R-Applikation an sich, sondern auf die Darstellung im Browser. Denn eine unkomplizierte Integration von Shiny-Applikationen in bestehende komplexe Webapplikationen ist von RStudio momentan nicht vorgesehen und nur mit erheblichem Programmieraufwand möglich.

Wie aber das angeführte „Hallo Shiny“-Beispiel gezeigt hat, sind zur Erstellung eigenständiger Shiny-Applikationen lediglich R-Kenntnisse notwendig. Persistente Websocketverbindungen, in Verbindung mit reactive functions und dem Bootstrap-Framework, erlauben mit geeigneten Browsern eine interaktive Nutzung auf nahezu

jedem Endgerät. Wenn dabei auf den Schutz geistigen Eigentums völlig verzichtet und der R-Code somit weitergegeben werden kann, ist nur das R-Package **shiny** notwendig, um Shiny-Applikationen lokal auszuführen.

Um Shiny-Applikationen in geschützten Netzwerken via Browser verfügbar zu machen, ist Shiny Server bereits in der freien Version bestens geeignet. Ein sicherer Betrieb im Internet erfordert aber, wie im Artikel bereits erwähnt, mindestens einen Reverse-Proxy und erhebliche Anpassungen im Quellcode oder eben die Lizenzierung der kommerziellen Shiny Server-Version, die eine Absicherung beinhaltet.

OpenCPU

Wesentlich komfortabler als Shiny Server kann OpenCPU Cloud Server den eigenen Bedürfnissen angepasst werden. Im Gegensatz zur Codeentwicklung bei Shiny-Applikationen, muss für Webapplikationen mit einer OpenCPU-Anbindung der R-Code nicht angepasst werden. Für die Einbindung in ein Frontend sind jedoch Kenntnisse in der Webentwicklung notwendig (v. a. wenn Inhalte responsive dargestellt werden sollen). Dann aber sind den Möglichkeiten und der Kreativität, R-Funktionalitäten in bestehende Webseiten zu integrieren, nahezu keine Grenzen gesetzt.

Die HTTP-API erlaubt es dem Webentwickler geschützte R-Funktionen oder -Skripte aufzurufen und die Ergebnisse im Browser darzustellen, als Download verfügbar zu machen oder in Form von

weiteren Funktionsaufrufen oder Speichervorgängen weiter zu verarbeiten. Komponenten von Webapplikationen, wie z. B. Authentifizierungsmodule und Benutzerverwaltungen oder Datenbankverbindungen, bleiben hiervon unberührt und können uneingeschränkt eingebunden werden. Ein durchdachtes Frontend verlangt dann vom Endbenutzer letztlich nur noch inhaltliche Kenntnisse der Applikation, aber keine R-Kenntnisse.

Fazit

Wenn neben R-Kenntnissen auch Kenntnisse in der Webentwicklung vorhanden sind, bietet OpenCPU flexiblere Möglichkeiten als Shiny, um eine umfangreiche R-Webapplikation zu erstellen und im Internet erreichbar zu machen. Die Weiterentwicklung von Shiny sollte aber dennoch unbedingt verfolgt werden. Wären in der freien Version nicht die Hürden der unzureichenden Absicherung und der komplizierten Integration in bestehende Projekte zu nehmen, wären Shiny-Applikationen sicherlich häufiger im Web anzutreffen.

LINKS

- [1] <http://cran.r-project.org/manuals.html> 
- [2] <http://www.rstudio.com> 
- [3] <http://www.rdocumentation.org> 
- [4] <http://cran.r-project.org> 
- [5] <http://journal.r-project.org/archive/2014-1/mair-chamberlain.pdf> 
- [6] <https://github.com/rstudio/rstudio> 
- [7] <http://www.rstudio.com/products/rstudio/download-server> 
- [8] <http://shiny.rstudio.com> 

- [9] <http://shiny.rstudio.com/gallery> 
- [10] <https://github.com/rstudio> 
- [11] <http://nodejs.org> 
- [12] <https://www.npmjs.org/package/shiny-server> 
- [13] <http://www.rstudio.com/products/shiny/shiny-server> 
- [14] <http://scotch.io/tutorials/javascript/easy-node-authentication-setup-and-local> 
- [15] <http://getbootstrap.com> 
- [16] <http://shiny.rstudio.com/articles/css.html> 
- [17] <https://github.com/wch/shiny-jsdemo> 
- [18] <https://www.opencpu.org> 
- [19] <https://github.com/jeroenooms/> 
- [20] <https://www.opencpu.org/jslib.html> 
- [21] <https://npmjs.org/package/opencpu> 
- [22] https://www.w3.org/Security/wiki/Same-Origin_Policy 
- [23] <http://www.chartjs.org/> 
- [24] <http://arxiv.org/abs/1406.4806> 
- [25] <https://github.com/jstat/jstat> 
- [26] <https://github.com/sjkaliski/numbers.js> 
- [27] <https://github.com/wayoutmind/gauss> 
- [28] <https://github.com/jasondavies/science.js> 

Autoreninformation

Markus Herrmann beschäftigt sich im Bereich Marketing & Data Sciences oft und gerne mit R und seinen Schnittstellen.

 [Teilen](#)

 [Kommentieren](#)

Browser: alle doof. – Eine Odyssee durch die Welt der Webbetrachter von Jennifer Rößler

Während der normale Linuxnutzer bei der Auswahl seiner Distribution, Desktopumgebung und Programme meist nach persönlichen Vorlieben entscheidet, haben Sehbehinderte diesen Luxus nicht. Hier kommt es darauf an, wie gut sich eine Oberfläche anpassen lässt oder überhaupt zugänglich ist. Es gibt zwar spezielle Distributionen für Blinde [1], doch auch hier ist die Auswahl für „nur“ Sehbehinderte im Prinzip nicht viel anders als bei anderen Linuxen. Das gilt auch für eines der wichtigsten Programme, den Webbrowser.

Ob eine Internetseite für Sehbehinderte ohne Probleme bedienbar ist, entscheidet oft nicht nur die sogenannte Barrierefreiheit, sondern auch welche Möglichkeiten ein Browser bietet. Für Blinde ist nur entscheidend, ob ein Browser mit Screenreader problemlos bedienbar ist oder nicht [2], für Sehbehinderte spielen völlig andere Kriterien eine große Rolle.

Der Browser ist das Fenster ins Internet – oder nicht?

Während ein Normalsehender einen Browser installiert und loslegen kann, müssen Sehbehinderte, je nach Umfang der Sehbehinderung, erst diverse Einstellungen vornehmen, um vernünftig mit einem Browser arbeiten zu können. Für einen Großteil der Sehbehinderten genügt es sicherlich, lediglich die Schriftgröße oder, wenn möglich, viel-

leicht noch die Schriftart zu ändern. Ob es dann mit der Vergrößerung klappt, hängt zum großen Teil auch von den einzelnen Internetseiten ab, denn gelegentlich kann es vorkommen, dass bei der Vergrößerung nicht mehr der ganze Text sichtbar ist. Auf anderen Seiten wiederum hat man dann die Wahl, entweder nach links oder rechts zu scrollen oder CSS ganz auszuschalten, um den kompletten Text auf einen Bildschirm zu bekommen. Wer ein eingeschränktes Gesichtsfeld hat oder nur mit einem Auge liest, hat oft nicht die ganze Seite im Blick und je mehr anderes, wie z. B. Werbung, auf einer Seite ist, umso länger dauert es das eigentlich Wichtige, den Text, zu finden. Abgesehen von der Gefahr durch Werbeeinblendungen mit Schadcode eingedeckt zu werden,

sind Werbeblocker also schon allein deshalb eine Notwendigkeit.

Es gibt aber auch eine Gruppe von Sehbehinderten, die mit einem hellen Hintergrund nicht zurechtkommt, weil dies das Zurechtfinden erschwert und bei längerfristigem Lesen anstrengend wird und Kopfschmerzen verursacht. Bevorzugt wird von dieser Gruppe ein dunkler, am besten schwarzer oder dunkelgrauer Hintergrund und weiße oder gelbe Schrift. Außerdem muss die Schriftgröße beliebig zu ändern sein und es ist von Vorteil, wenn im Browser nicht erst eine Stunde nach den Einstellungen gesucht werden muss. Des Weiteren ist eine Statusleiste der besseren Übersicht halber praktisch und deren Anzeige oft

auch besser den Systemfarben angepasst, als dies bei Erweiterungen manchmal der Fall ist.

Google Chrome

Schon einmal stand man vor der Entscheidung, Firefox, der bereits einige Jahre ein treuer Begleiter war, durch einen anderen Browser zu ersetzen, da er immer länger zum Starten benötigte und die FeedReader-Erweiterung Brief nach einem Firefox-Update plötzlich nicht mehr richtig

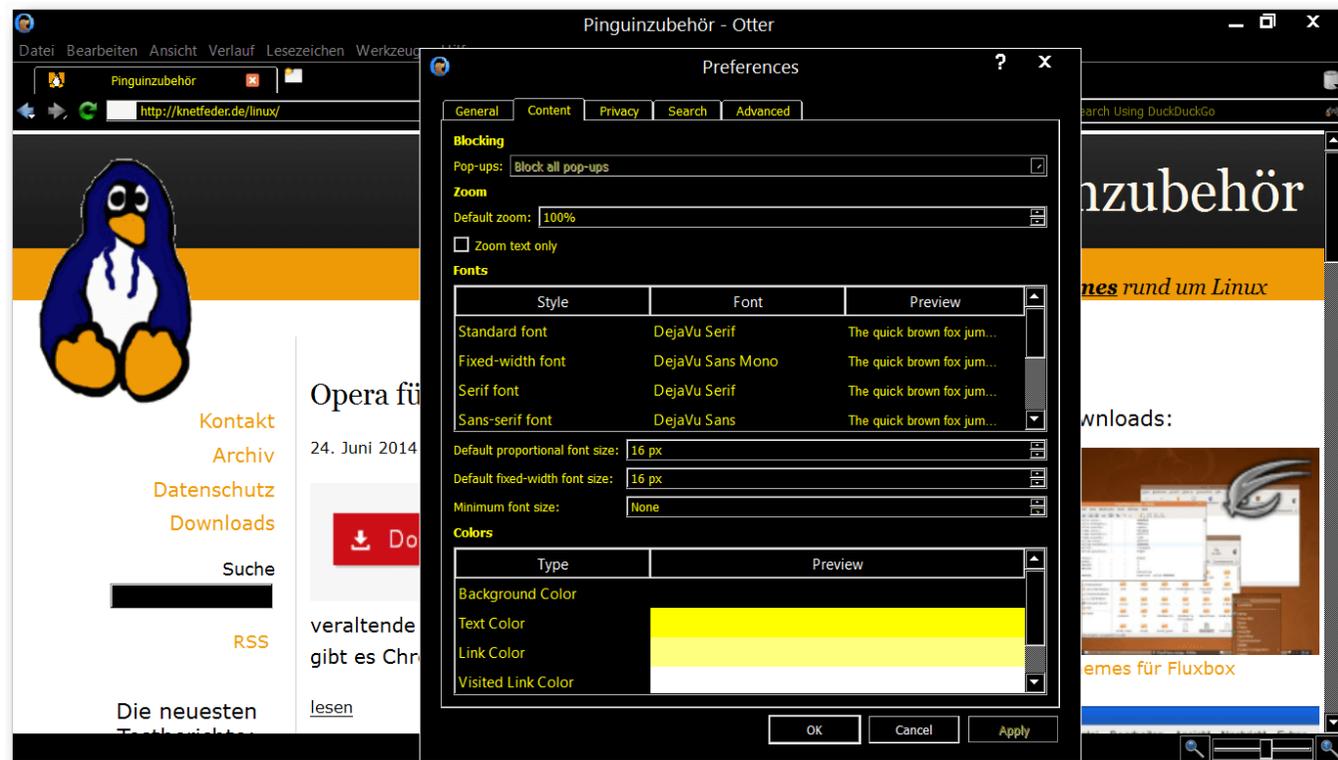


Rekonq-Browser in sehbehindertenfreundlichen Farben.

funktionierte. Im Zuge dessen wurden Google Chrome [3] und K-Meleon [4] bereits vor längerer Zeit einmal auf die oben genannten Kriterien getestet und es stellte sich heraus, dass sie generell nur schwer zu bedienen sind [5]. Zwar lässt sich die Schriftgröße einer Internetseite anpassen, aber nach Einstellungen für Seitenhintergrund und -farbe sucht man vergebens. Bei K-Meleon war es zum damaligen Zeitpunkt sogar so, dass diese Einstellungen für einen Laien nicht zugänglich waren. Der Versuch, die kürzlich neu erschienene Version zu installieren, scheiterte.

Opera und others

Opera [6] war zu dem Zeitpunkt eine akzeptable Alternative. Zwar war es auf Anhieb nicht ganz leicht, die Farbeinstellungen zu finden, da es hier nur feste Kontrastdarstellungen zur Auswahl gab, aber immerhin waren schwarzer Hintergrund und weiße Schrift möglich und die Vergrößerung auch kein Problem. Ein großes Plus waren außerdem die „seitenspezifischen Einstellungen“, die es einem ohne zusätzliche Erweiterung ermöglichten, JavaScript, Flash oder Cookies beliebig an- und auszuschalten. Außerdem bekam man einen wirklich guten FeedReader, wenn man hier auch bei den Farben Abstriche machen musste. Der Hintergrund der Feedeinträge war zwar nicht weiß, sondern hellgrau, und die Schrift schwarz, aber da kein völlig weißer Hintergrund, akzeptabel. Doch als Opera dann den FeedReader abschaffte und mit Version 20 keine Farbeinstellungen mehr möglich waren, kehrte man komplett zu Firefox zurück.



Otter. 

Otter [7] könnte vielleicht eine akzeptable Alternative werden. Die kürzlich erschienene Beta-Version lässt hoffen. Die Windows-Version lässt sich problemlos installieren und passt sich nahtlos dem System an. Zwar ist das Menü noch stellenweise zweisprachig und einige Funktionen sind noch nicht verfügbar, aber die Aussicht besteht, dass sich das in folgenden Versionen noch ändert. Selbst Hintergrund- und Textfarbeinstellungen bis hin zur Linkfarbe sind vorhanden, haben aber im Moment noch keine Auswirkung.

Die Möglichkeit, einfach den Kontrast umzustellen, fehlt jedoch (noch).

Firefox und Faksimiles

Die Aussichten, die Firefox 29 mit Australis bietet, machen es nun erforderlich, sich wieder nach einem brauchbaren Ersatz umzusehen. Hierbei geht es inzwischen aber nicht mehr vorrangig um windowsbasierte Browser, sondern solche, die auch unter Linux laufen. Die Tatsache, dass ein Browser jedoch auch von einem Laien installiert

werden können muss, schränkt die Suche weiter ein.

Firefox ESR [8] und Iceweasel [9] sind vorläufig nette Alternativen und bieten, wie auch Firefox selbst, in den Einstellungen die Möglichkeit, Hintergrund einer Seite und Schriftfarbe und -größe beliebig anzupassen. Den Farbeinstellungen des Systems passt sich deren Menü problemlos an. Doch leider werden sie aufgrund ihrer Entwicklung irgendwann auch bei Australis angelangt sein und damit wäre die Oberfläche weniger komfortabel.

Seamonkey [10] bietet, was Schrift- und Farbeinstellungen angeht, dieselben Möglichkeiten wie Firefox, wenn man aber bereits ein anderes

E-Mailprogramm und seit dem Wegfall des FeedReaders bei Opera Akregator als FeedReader verwendet, ist das zu viel des Guten.

Pale Moon [11] ist unter Windows die optimale Alternative. Er passt sich dem System problemlos an, die Einstellungen für Schriftgröße, Hintergrund- und Schriftfarbe finden sich in etwa an derselben Stelle, wo sie auch beim Firefox zu finden sind, was die Einrichtung sehr simpel macht und eine Statusleiste ist auch vorhanden.

Die benötigten Erweiterungen, die man bereits von Firefox kennt, gibt es auch für Pale Moon. Zwar ist das Umstellen auf Deutsch etwas umständlich, aber damit kann man leben. Leider lässt

sich Pale Moon für Linux für einen Laien nicht installieren, sonst wäre die Suche an dieser Stelle beendet gewesen.

Midori

Midori [12] wäre eine durchaus brauchbare Alternative, der Browser selbst passt sich den Systemfarben problemlos an, ein Adblocker ist sogar schon integriert, die Statusleiste ist auch noch da und auch die Schrift-

größe lässt sich problemlos ändern. Doch nach Farbeinstellungen für Hintergrund und Schrift der Internetseiten sucht man vergeblich.

QupZilla

QupZilla [13] ist einfach zu installieren und gefällt wirklich gut. Er kommt in den Systemfarben daher, hat von Haus aus eine Statusleiste und bringt noch einen Adblocker mit. Die Schriftgröße und Schriftart können problemlos in den Einstellungen geändert werden. Doch leider gibt es keine Möglichkeit, die Farbeinstellungen des Systems auch auf die Internetseiten auszudehnen wie beim Firefox.

Konqueror und Konsorten

Konqueror [14], den KDE von Haus aus mitbringt, hat einen eigenen Adblocker integriert und bietet die Möglichkeit, mit einem Klick JavaScript oder Cookies abzuschalten, auch eine Statusleiste ist vorhanden. Doch weitere Einstellungen, die Schriftgröße oder Farben betreffend, sind nicht vorhanden. Dabei ist sogar die Möglichkeit ins Menü integriert, sich eine Internetseite mit der KDE-eigenen Sprachausgabe Jovie vorlesen zu lassen.

Rekonq [15] soll Konqueror ersetzen, er ist tatsächlich aber kein vollwertiger Ersatz. Auch er kommt mit einem eigenen Adblocker und bietet umfangreiche Einstellungen bezüglich JavaScript, Webkit und HTML5 bis hin zur Wahl von Schriftgröße und Schriftart. Farbeinstellungen fehlen leider völlig. Eine Statusleiste gibt es aber nicht



Midori. 



Epiphany bzw. Web. 

mehr, schnelles Ein- und Ausschalten diverser Funktionen direkt aus dem Menü ist nicht möglich. Das Menü ist zwar umfangreich, jedoch ähnlich wie bei Firefox an den rechten Rand gerückt, und es lässt sich kein klassisches Menü einrichten. Die Möglichkeit, sich Internetseiten vorlesen zu lassen, fehlt hier.

GNOME

Epiphany (neuerdings Web genannt [16]) passt sich, da GNOME-basiert, nicht an Systemfarben an. Das Menü ist extrem rudimentär, Farb- oder Schriftgrößeneinstellungen gibt es nicht. Nur die Möglichkeit, mit `[Strg] + []` die Seite zu vergrößern, funktioniert. Der Browser taugt nicht mal für einen eigenen Artikel. Da werden Alpträume bezüglich Google Chrome wach.

Dillo

Dillo [17] ist kein Browser, sondern ein Zustand. Nämlich der Zustand der Verzweiflung, der die

absoluten Laien beim Aufruf einer Internetseite befallen würde, wenn sie den Browser ohne Vorwissen ausprobieren. Er beherrscht kein JavaScript, CSS ist zwar inzwischen implementiert, Objekte werden aber nicht richtig positioniert und teilweise verformt dargestellt. Einstellungen für Vergrößerung und Farben gibt es keine und auch ans System ist keine Anpassung vorhanden. Selbst mit `[Strg] + []` ist keine Vergrößerung möglich. Es ist ein rudimentäres Menü vorhanden, dessen Schrift allerdings so klein ist, dass sie je nach Grad der Sehbehinderung kaum lesbar ist.

Fazit

Tatsächlich ist die Zahl der Browser, die Farbeinstellungen ermöglichen, sehr überschaubar. Während Seamonkey aus anderen Gründen als Browser der Wahl ausscheidet, Iceweasel und Firefox ESR auch irgendwann den Entwicklungsstand des jetzigen Firefox eingeholt haben werden und Otter sich noch etwas weiterentwickeln muss, ehe er als Ersatz in Frage kommen kann, bleibt, neben Pale Moon für Windows, für Linux vorläufig tatsächlich nur Firefox (einschließlich ESR). So wird man sich wohl erst einmal damit arrangieren müssen, nicht mehr alles im Blick zu haben, und darauf hoffen, dass diese Einstellungen nicht auch aus Firefox irgendwann ganz verschwinden.

LINKS

[1] <http://www.knetfeder.de/linux/index.php?id=65>

- [2] <http://www.knetfeder.de/magazin/2009/internet/blindes-internet/>
- [3] <https://www.google.de/intl/de/chrome/browser/>
- [4] <http://kmeleon.sourceforge.net/>
- [5] <http://jenny-box.de/blog/2010/11/04/google-chrome-%E2%80%93-ein-erster-eindruck/>
- [6] <http://www.opera.com/de>
- [7] <http://otter-browser.org/> 
- [8] <https://www.mozilla.org/en-US/firefox/organizations/all/> 
- [9] <https://wiki.debian.org/Iceweasel> 
- [10] <http://www.seamonkey-project.org/> 
- [11] <http://www.palemoon.org/> 
- [12] <http://midori-browser.org/> 
- [13] <http://www.qupzilla.com/>
- [14] <http://konqueror.org/> 
- [15] <http://rekonq.kde.org/> 
- [16] <https://wiki.gnome.org/Apps/Web> 
- [17] <http://www.dillo.org/> 

Autoreninformation

Jennifer Röbler ([Webseite](#)) ist begeisterte Linuxnutzerin und sehbehindert und benötigt für ihre Arbeit am PC möglichst schwarzen Hintergrund und gelbe Schrift. Barrierefreiheit ist für sie aber ein abstrakter Begriff, da man Zugänglichkeit nicht immer pauschalisieren kann.

 [Teilen](#)

 [Kommentieren](#)

Äquivalente Windows-Programme unter Linux – Systemprogramme und Zubehör von Maria Seliger

Dieser Artikel beschäftigt sich mit Programmen, die man unter Windows in den Bereichen „Systemprogramme“ oder „Zubehör“ findet.

Archivierungsprogramme

Archivierungs- oder Packprogramme fassen Dateien in einem Container zusammen, sodass sie leichter zu handhaben sind und kleiner werden, z. B. zum Versenden. Umgekehrt entpacken sie auch entsprechende Container wieder zu einzelnen Dateien.

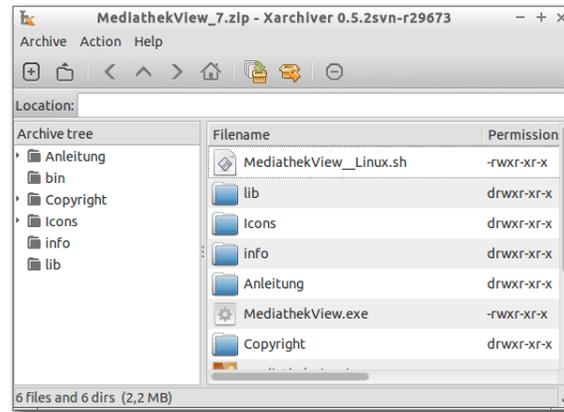
Archive Manager (File-Roller)

Der Archive Manager (der auch unter dem Namen File-Roller [1] bekannt ist) stellt eine grafische Oberfläche für eine Vielzahl von Packformaten bereit und ist Teil der GNOME-Umgebung. Zu den unterstützten Formaten zählen u. a. 7z, tar, RAR, zip, gzip, bzip, Arj, LHA, JAR, Debian- und RPM-Pakete sowie ISO-Abbilder, sofern die entsprechenden Programme der Kommandozeile für das Archivformat installiert wurden.

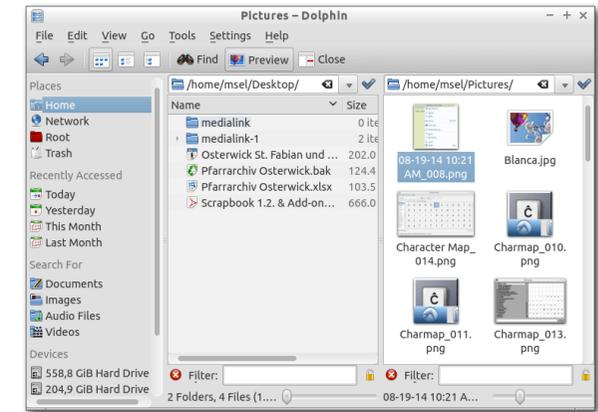
- Homepage: <http://fileroller.sourceforge.net/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

Xarchiver

Xarchiver [2] besitzt eine grafische Oberfläche basierend auf GTK+ und unterstützt eine Vielzahl von Datenkompressions- und Packprogrammen.



Xarchiver mit geöffnetem Archiv. 🔍



Dolphin mit geteilter Anzeige sowie Bildervorschau. 🔍

Vom Umfang ist es ähnlich wie der Archive Manager – es soll allerdings besonders ressourcenschonend sein.

- Homepage: <http://xarchiver.sourceforge.net/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

Dateimanager

Dateimanager unterstützen die Organisation und Verwaltung von Dateien auf dem Computer.

Dolphin

Dolphin [3] ist der Standard-Dateimanager für KDE. Das Programm bietet eine Dokumentenhistorie, eine Suchfunktion sowie die Möglichkeit, ein Terminal zu öffnen. Außerdem kann der Dateimanager eine Vorschau von bestimmten Dateien, z. B. Bildern, anzeigen.

- Homepage: <http://dolphin.kde.org/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

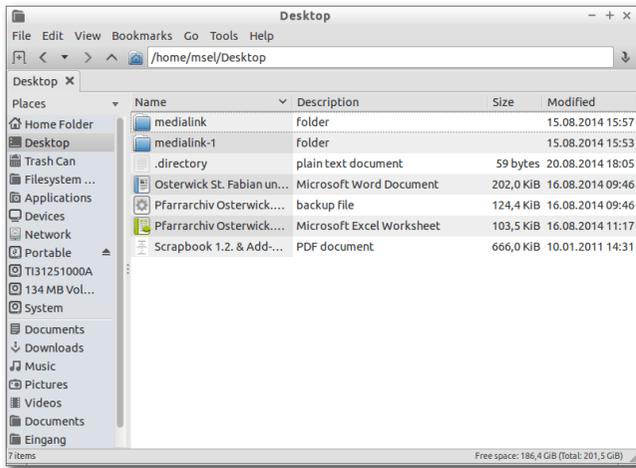
Nautilus

Das Gegenstück zu Dolphin ist Nautilus [4], welcher der Standard-Dateimanager für die GNOME-Umgebung ist. Vom Funktionsumfang gleicht das Programm ungefähr Dolphin, es unterstützt aber u. a. auch einen Remote-Access zu einem Server.

- Homepage: <https://wiki.gnome.org/Nautilus>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

PCMan File Manager

PCMan File Manager [5] ist ein leichtgewichtiger Dateimanager, der unter LXDE und Ubuntu zum



PCMan File Manager mit Detailansicht. 

Einsatz kommt. Vom Funktionsumfang gleicht er Nautilus – so unterstützt er auch den Remote-Access zu einem Server.

- Homepage: <http://wiki.lxde.org/en/PCManFM>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

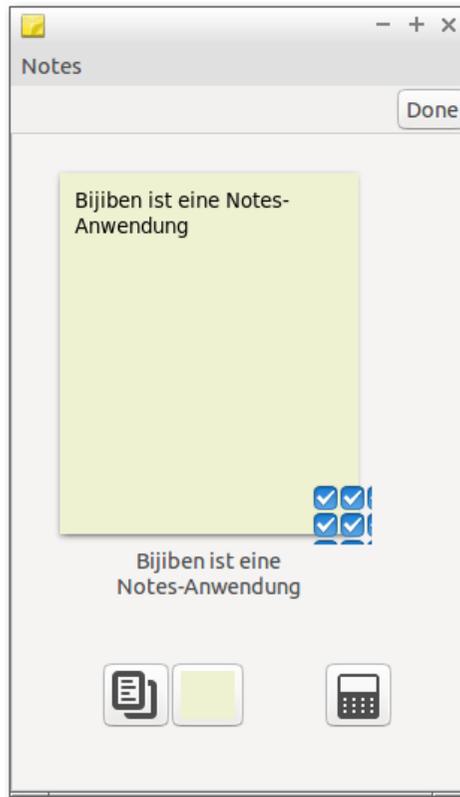
Das Gegenstück zum PCMan File Manager unter XFCE ist Thunar [6].

Kurznotizen

Kurznotizen sind kleine Programme zum Notieren von Aufgaben u.ä., die meist sichtbar am Desktop erscheinen.

Notes (Bijiben)

Notes (früher Bijiben [7] genannt) ist eine Notizzettelverwaltung, die in GNOME 3 integriert ist.



Bijiben Notes Übersicht. 

Im Gegensatz zu Xpad und KNotes (siehe weiter unten) werden hier die Notizzettel in einer eigenen Umgebung angeheftet und nicht auf dem Desktop angezeigt. Das Programm unterstützt auch den Import von Notizen aus anderen Programmen (z. B. Tomboy)

- Homepage: <https://wiki.gnome.org/Apps/Bijiben>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

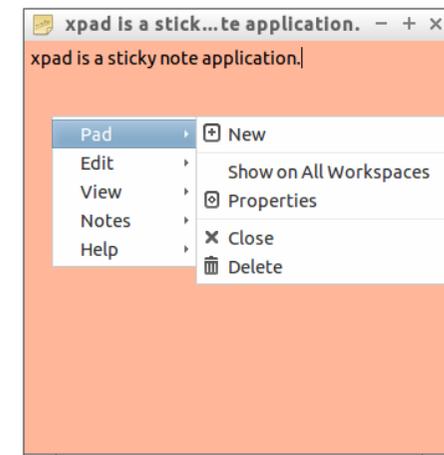
KNotes

KNotes [8] ist eine KDE-Anwendung, mit der Notizzettel erstellt werden und z. B. auf den Desktop angeheftet werden. Die Notizzettel können verschiedenen Sammlungen zugeordnet, einzeln gespeichert und mit einem Alarm versehen werden. Außerdem unterstützt das Programm das Drucken und das Versenden der Notizen. Zusätzlich besitzt das Programm eine Rechtschreibprüfung.

- Homepage: <http://www.kde.org/applications/utilities/knotes/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

Xpad

Eine typische Kurznotiz-Anwendung ist Xpad [9]. Mit diesem Programm lassen sich schnell einfache Aufgaben und Notizen notieren und wiederfinden.



Xpad-Notiz mit Kontextmenü. 

- Homepage: <https://wiki.gnome.org/Apps/xpad>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

Rechner

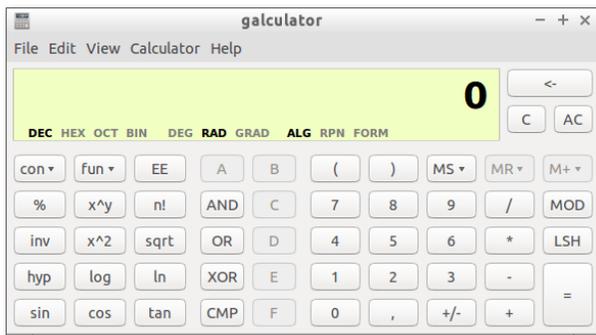
Calculator

Der Rechner Calculator [10] ist ein Taschenrechner, der drei verschiedene Modi anbietet:

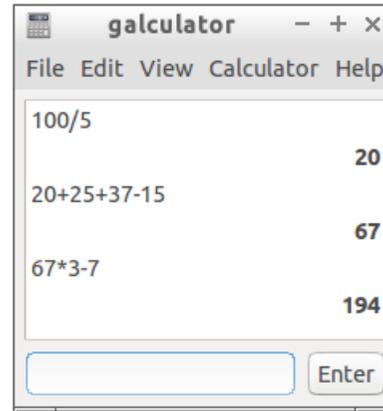
- Basic Mode (einfacher Modus): einfacher Taschenrechner
- Scientific Mode (wissenschaftliche Modus): wissenschaftlicher Taschenrechner für Schule und Studium
- Paper Mode (Papiermodus): Taschenrechner mit Druckausgabe

Das Programm unterstützt auch die umgekehrte polnische Notation (UPN).

- Homepage: <http://calculator.sourceforge.net/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux



Calculator Wissenschaftliche Ansicht. 

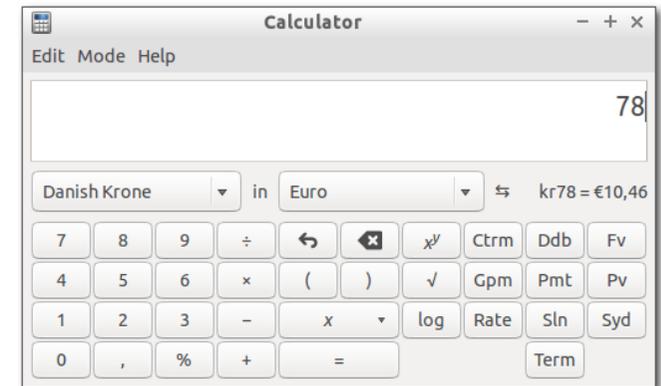


Calculator Papiermodus. 

GNOME Calculator

Calculator [11] ist ein Taschenrechner für die GNOME-Umgebung, der vier verschiedene Modi anbietet:

- Basic (einfacher Modus): einfacher Taschenrechner
- Advanced (fortgeschritten): wissenschaftlicher Taschenrechner
- Financial (Finanzrechner): ein Finanzrechner, der u. a. auch Währungsumrechnungen durchführt
- Programming (Programmierrechner): Logikrechner speziell für Binär-, Oktal- und Hexadezimalzahlen, der logische Verknüpfung wie AND oder OR unterstützt
- Homepage: <https://wiki.gnome.org/action/show/Apps/Calculator>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

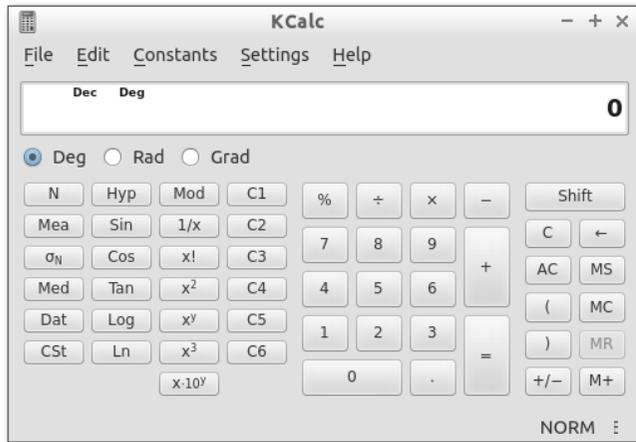


GNOME Calculator Finanzrechner. 

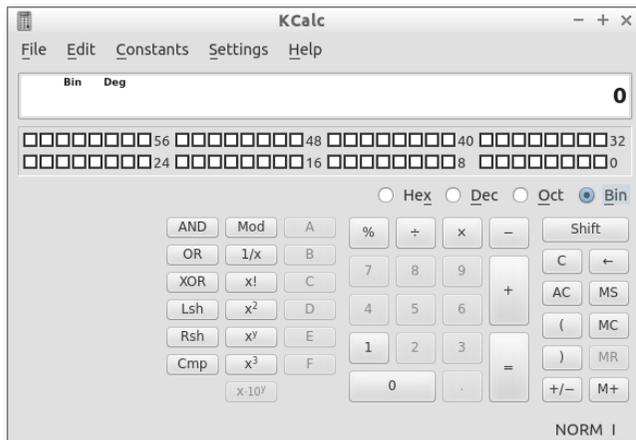
KCalc

KCalc [12] ist ein Taschenrechner für die KDE-Oberfläche. Dieser besitzt vier verschiedene Modi:

- Simple Mode (einfacher Modus): einfacher Taschenrechner
- Scientific Mode (wissenschaftlicher Modus): wissenschaftlicher Taschenrechner für Schule und Studium
- Statistic Mode (statistischer Modus): Statistikrechner für spezielle statistische Aufgaben in Schule und Studium
- Numeral System Mode (numerischer Modus): Logik-Rechner speziell für Binär-, Oktal- und Hexadezimalzahlen, der logische Verknüpfung wie AND oder OR unterstützt
- Homepage: <http://utils.kde.org/projects/kcalc/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux



KCalc Statistikrechner.

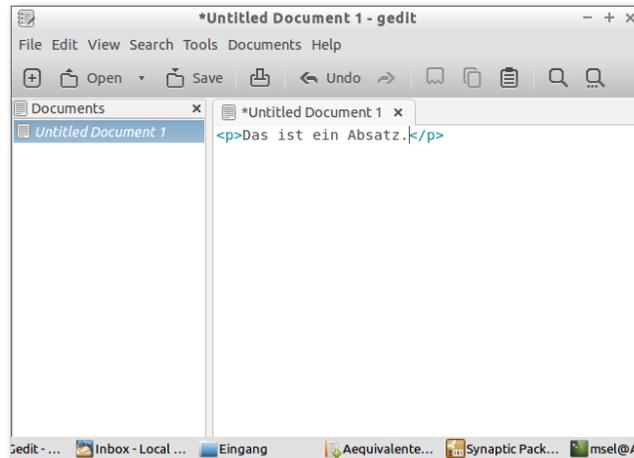


KCalc Binärrechner.

Texteditoren

gedit

gedit [13] ist ein UTF-8-Texteditor für die GNOME-Umgebung. Der Editor kann über Plug-ins mit weiteren Funktionen ausgestattet werden. So besitzt er auch eine Rechtschreibprüfung und kann



gedit mit HTML-Code, der hervorgehoben wird.

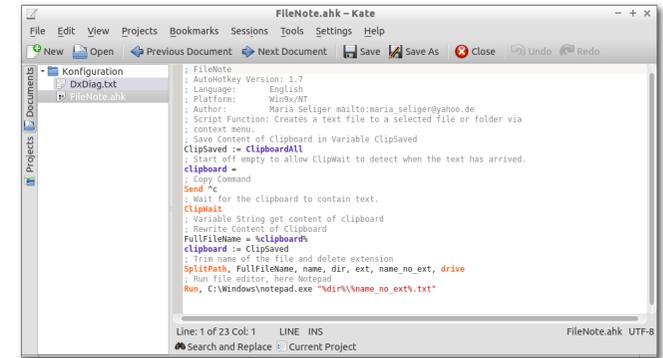
die Elemente von verschiedenen Programmiersprachen hervorheben.

- Homepage: <https://wiki.gnome.org/Apps/Gedit>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux, Mac OS X, Windows

Kate

Kate [14] ist ein UTF-8-Texteditor für KDE. Durch Syntaxhervorhebung und Codefaltung für etliche Programmiersprachen eignet sich der Editor besonders für fortgeschrittene Nutzer wie Programmierer oder Administratoren. Das Programm kann gleichzeitig mehrere Dateien öffnen.

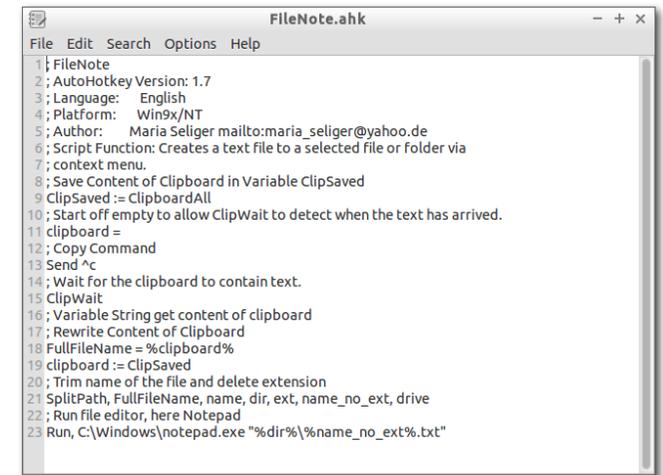
- Homepage: <http://kate-editor.org/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux, Windows



Kate mit zwei geöffneten Dateien.

Leafpad

Leafpad [15] ist ein einfacher UTF-8-Texteditor basierend auf GTK+ für die LXDE-Umgebung und Ubuntu, der sehr ressourcenschonend ist und schnell startet. Im Funktionsumfang entspricht er in etwa dem Editor Notepad bei Windows.



Leafpad mit geöffneter Datei.

- Homepage: <http://tarot.freeshell.org/leafpad/>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

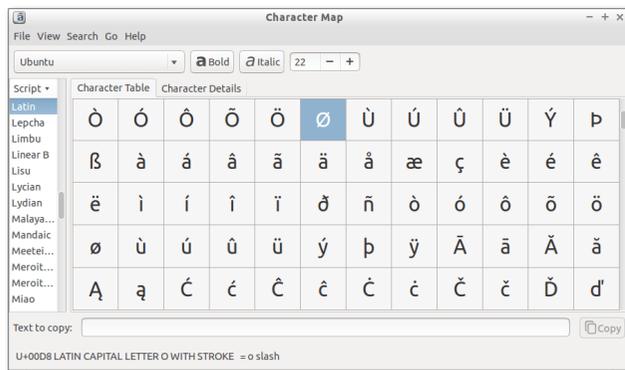
Weitere Alternativen

Vim [16] und Emacs [17] zählen sicher zu den bekannten Texteditoren unter Linux. Beide sind jedoch für Windows-Anwender sehr gewöhnungsbedürftig und vertreten sehr unterschiedliche Philosophien bzgl. der Bedienung. Allerdings handelt es sich um sehr mächtige Texteditoren, mit denen umfangreiche Aufgaben gelöst werden können.

Zeichentabelle

Character Map (gucharmap)

Character Map (früher gucharmap [18]) ist eine Zeichentabelle für die Auswahl von Unicode-Zeichen für GNOME. Nachdem man die Schriftart gewählt hat, kann man die Zeichen auswählen und in die Zwischenablage kopieren. Das Programm hat auch eine Suchfunktion.



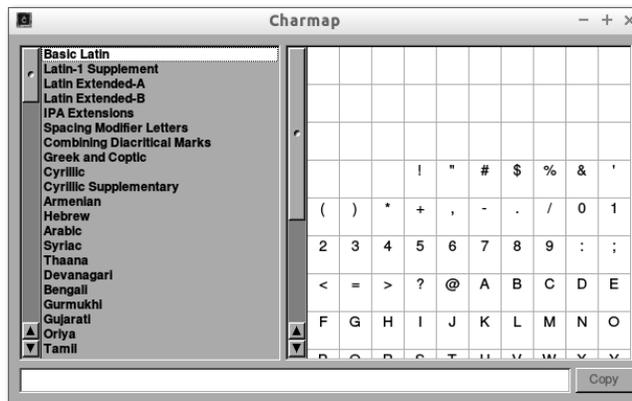
Character Map für die Auswahl von Unicode-Zeichen. 

- Homepage: <https://wiki.gnome.org/Apps/Gucharmap>
- Lizenz: GPL
- Unterstützte Betriebssysteme: Linux

Charmap

Charmap [19] ist eine Zeichentabelle für GNUstep [20]. Nachdem man eine Schriftart ausgewählt hat, zeigt die Zeichentabelle geordnet die Zeichen dieser Schriftart an.

- Homepage: <http://wiki.gnustep.org/index.php/Charmap.app>
- Lizenz: GPL, LGPL
- Unterstützte Betriebssysteme: Linux, Mac OS X, Windows (MinGW)



Charmap-Zeichentabelle. 

- [4] <https://wiki.gnome.org/Nautilus> 
- [5] <http://wiki.lxde.org/en/PCManFM> 
- [6] <http://docs.xfce.org/xfce/thunar/> 
- [7] <https://wiki.gnome.org/Apps/Bijiben> 
- [8] <http://www.kde.org/applications/utilities/knotes/> 
- [9] <https://wiki.gnome.org/Apps/xpad> 
- [10] <http://calculator.sourceforge.net/> 
- [11] <https://wiki.gnome.org/action/show/Apps/Calculator> 
- [12] <http://utils.kde.org/projects/kcalc/> 
- [13] <https://wiki.gnome.org/Apps/Gedit> 
- [14] <http://kate-editor.org/> 
- [15] <http://tarot.freeshell.org/leafpad/> 
- [16] <http://www.vim.org/> 
- [17] <https://www.gnu.org/software/emacs/> 
- [18] <https://wiki.gnome.org/Apps/Gucharmap> 
- [19] <http://wiki.gnustep.org/index.php/Charmap.app> 
- [20] <http://www.gnustep.org/> 

Autoreninformation

Maria Seliger (Webseite) ist vor über einem Jahr von Windows 7 auf Ubuntu umgestiegen, was wider Erwarten schnell und problemlos ging, da sich für die meisten Programme unter Windows eine gute Alternative unter Linux fand.

 [Teilen](#)

 [Kommentieren](#)

LINKS

- [1] <http://fileroller.sourceforge.net/> 
- [2] <http://xarchiver.sourceforge.net/> 
- [3] <http://dolphin.kde.org/> 

Recht und Unrecht von Stefan Wichmann

Mitten ins Auge! Ich fluchte. Hastig wischte ich mir die Druckerpaste aus dem Gesicht, entdeckte die flatternde Leitung, aus der immer noch die klebrige Masse in die Gegend spritzte und hielt sie mit meinem Daumen zu. Der Druck war zu hoch und die Zuleitung zu spitz. Meine eigene Konstruktion! Hätte es funktioniert, wäre ich stolz gewesen, so aber bohrte sich der dünne Kunststoffschlauch in meine Haut und spritzte mir mühelos meine selbst entwickelte Nanoschalungspaste in den Daumen. Ich stöhnte, während sich unter der Epidermis eine Beule bildete ...

„Wissen Sie, dass in der Nähe des Daumens die Vena cephalica beginnt? Ich fürchtete, dass das Zeug direkt in meine Blutbahn gelangt!“

Fürchterliche Gedanken schossen mir durch den Kopf, viele tausend Vorstellungen. Plötzlich ging auch noch die blöde Härtelampe an und härtete die Anschwellung aus! Ich war mit meinem Konstrukt verbunden! Meine linke Hand tastete nach dem Ausschalter. „Mist!“, rief ich. Der Schalter ist auf der rechten Seite des 3-D-Druckers angebracht! „Alles in Ordnung, Schatz?“, rief meine Frau. „Ja, Süße, alles gut“, log ich. Ängstlich zog ich an meiner Hand. Ein heftiger Schmerz durchzog meinen Körper, als die bereits ausgehärtete Paste sich in meiner Haut wie ein Widerhaken verankerte. Ich stöhnte. Die Härtelampe erlosch.

Zu spät. Mir schlug das Herz bis zum Hals. Unruhe breitete sich zuerst in meinem Magen aus und stieg alsdann wie ein Sodbrennen auf. Es war kein Sodbrennen ...

„Es war Panik. Reinste Panik.“

Zu diesem Zeitpunkt dachte ich nicht daran, dass dieser mein erster Prototyp einer unbenannten Spielzeugdrohne mich vor den Kadi zerren könnte. Aber bereits wenige Tage später erzählte ich eben diese Geschichte einem Richter. Selbstverständlich hatte ich mich vorher über ihn erkundigt. Er war gerade eingereist und noch unerfahren.

„Sie waren bereits bei der Panik“, unterbrach er meinen Redeschwall. „Sie sagten, ich zitiere: reinste Panik.“

Ich nickte.

„Weiter“, drängte er. Ich sah mich zu der Kommission um. Dies war kein ordentliches Gericht, bei dem der Fall in den Zentralcomputer gehackt und mit einer üblichen Strafe geahndet wurde.

„Weiter, weiter! Was haben Sie da denn überhaupt versucht zu erfinden?“

„Wie bereits angegeben arbeite ich aus der Not heraus als Erfinder. In meinem gelernten Beruf als Krankenpfleger kriege ich ja keinen Job und

die Wartezeit zum Medizinstudium wird mir durch andere ...“

„Weiter!“, bellte er.

Ich sammelte mich kurz. „Meine Idee, sofort gebrauchsfertige Waren auszudrucken, schien für mich umsetzbar. Analog zu den früher gebräuchlichen Tintenstrahldruckern mit 4 Farbpatronen installierte ich verschiedene Kammern im 3-D-Drucker. Jede enthielt eine andere Druckpaste. Die mit Dielektrikum bezeichnete fügt Füllmaterial an allen Stellen ein, die später beweglich sein müssen. So konnte ich beispielsweise ein Spielzeugauto komplett ausdrucken und es fuhr sofort.“

„Ja, und?“

„Mit einer Pinzette platzierte ich die dritte Düse im 3-D-Drucker und kam dabei auf den Anschlagknopf. Er ratterte sofort los, die Zuführleitung riss ab und sprühte durch die Gegend.“

„Das hatten wir schon. Ihr Daumen, die Lampe, das Ding war an Sie angebunden und ausgehärtet. Worum handelte es sich denn nun?“

„Nun ja, dazu muss ich weiter ausführen.“ Ich blickte in ein ungeduldiges Gesicht.

„Wissen Sie, was eine Drohne ist?“ Er lehnte sich schnaufend in seinem Sessel zurück. „Ich meine

keine große Kampfdrohne, für die eine Erlaubnis benötigt wird, wenn diese über bewohntes Gebiet fliegt. Ich rede von einem persönlichen kleinen Helferlein, wie er aus dem Comic bekannt ist und von Daniel Düsentrieb genutzt wird.“ Der Richter schnaufte erneut ungeduldig.

„Stellen Sie sich vor. Jeder ist in der Lage sich aus seinem 3D-Drucker sofort ein kleines Fluggerät auszudrucken! Es ist sofort einsatzfähig und per Nanoprogrammierung mit Standardaufgaben zu betrauen!“

Der Richter beugte sich vor. „Ich könnte diese loschicken und Ihnen ins Ohr brüllen lassen, dass Sie endlich zum Punkt kommen sollen?“

Ich stutzte und nickte. „Wenn Sie so wollen . . . Die Drohne kann losgeschickt werden, um Brötchen zu holen, Vertreter zu vertreiben, egal!“

„Tolle Idee. Warum haben Sie das Ding nicht gedruckt, zum Patent angemeldet und damit Geld verdient?“

Die Tür des Gerichtssaales ging auf und jemand trat ein. Vermutlich ein weiterer Zuschauer? Ich wand mich kurz um und sah, dass er sich zu den Mitklägern setzte. Er sah im Gesicht übel zugerichtet aus. Überhaupt sahen sie alle übel zugerichtet aus. Diese Raubkopierer, diese Ideenschleher, die sich einer nach dem anderen in mein System gehackt und meine Unterlagen gestohlen hatten. Als ich mich neuerlich nach vorn wandte,

striefte mein Blick kurz eine Fensterscheibe. Sie spiegelte meinen zerschundenen Körper.

„Also die Drohne war gedruckt und in meinem Finger steckte eine ausgehärtete Schicht des gleichen selbstentwickelten Komplexes, dass ich auch in die Drohne eingefügt hatte. Es handelt sich um das hardwarecodierte Programm, dass der Drohne Leben einhaucht.“ Ich konnte meinen Stolz nicht unterdrücken: „So kann das Komplettsystem ausgedruckt werden, ohne es nachträglich programmieren zu müssen!“

„Toll!“, unterbrach mich einer der Nebenkläger, „Wirklich ganz toll“. Ich zuckte die Schultern.

„In meinem Fall ging mein Drucker genau in dem Moment kaputt, als diese Codierung gedruckt werden sollte. Mein Prototyp hatte also lediglich sehr eingeschränkte Funktionen. Ich stellte ihn auf den Boden, schaltete ihn ein und das Ding stieg tatsächlich auf in die Luft!“, jubelte ich.

Der Kopf des Richters deutete auf meinen demolierten Körper. „Und das da?“

„Nun, wie gesagt, Prototypen sind eh anfällig für Fehlschaltungen und wenn die Logik fehlt und das Ding nicht weiß, was es tun soll, na dann muss man mit allem rechnen. Da es ja noch mit meiner Hand verbunden war, hatte ich Glück.“

„Das war nicht die Frage“, murmelte der Richter entnervt.

Ich senkte den Kopf und sagte leise: „Es griff mich an.“

Der Richter beugte sich vor. „Es tat was?“

„Es stieg in die Luft und flog einfach los. Es merkte, dass es wenig Energie hatte und durch die Fotosensoren suchte es wie eine Motte das Licht den hellsten Punkt im Raum. Leider war ich das. Ich trug eine helle Jacke mit Leuchtstreifen!“

„Aha und durch die Helligkeit gewinnt Ihr Ding da Strom, um den internen Akku aufzuladen.“

Ich nickte gottergeben. Er hatte es verstanden.

Hinter mir erhob sich einer der Nebenkläger. „Meine Drohne hatte die komplette Schaltung intus. Das Ding griff nicht nur an, sondern zerstückelte auch mein ganzes Mobiliar, bevor es gegen meine Stirn donnerte und mich zeichnete.“

Tatsächlich prangte eine Vertiefung auf seiner Stirn, die das Logo meiner Drohne dauerhaft in seine Haut gedrückt hatte: ‚D‘. Gleich darunter prangte mein Nachname ‚Rake‘. Wer alle Buchstaben zusammen las übersetzte ‚DRAKE‘ in Entenvogelmännchen [1]. Allein mein Nachname hatte mir zeitlebens viele Lacher eingebracht: Rake steht für Wüstling.

Ich sah die anderen Nebenkläger an. Je nach Breite der Stirn hatten nicht alle Buchstaben draufgepasst. Bei einem prangte lediglich ‚rak‘, was ich

als polnisches Wort für Geschwür kannte. Auf einem Gesicht war lediglich ‚Ra‘ zu erkennen, wie RA, der Sonnengott. Ich schmunzelte unwillkürlich. Wahrscheinlich brachte mir das die hohe Strafe am Ende der Verhandlung ein.

Der Gutachter des Staatsanwaltes führte aus, dass sich bei der anschließenden Explosion des „Luftfahrzeugs“, wie er es nannte, das Logo förmlich in die Stirn einbrannte. Durch die Wucht des „Fluggerätes“ sei in allen Stirnen eine leichte Delle entstanden, die ein Entfernen, selbst durch einen Laser, allenfalls „verschlimmbessern“ würde.

Ich drehte mich um. „Wie gesagt, bei mir hing es an der Hand. Als ich versuchte es abzuwehren, zerschellte es an der Wand.“

Mit dem Finger deutete ich anklagend auf den Nebenkläger: „Sie haben einen Prototypen geklaut, der noch in der allerersten Erprobung stand! Es war noch nicht, wie meine anderen Programme und Erfindungen, als Open Source freigegeben. Selbst schuld!“, donnerte ich.

Der Hammer des Richters hingegen donnerte auf seinen Tisch. „Sie hätten den Zugriff auf dieses druckbare Teufeldrohngedöhns verhindern müssen, solange die Auswirkungen so gefährlich sein können. Ich verurteile Sie dazu, den durch Ihr ‚Spielzeug‘ entstandenen Schaden zu begleichen und den Betroffenen eine lebenslange Rente zu zahlen. Mit so einem Abdruck kann ja keiner mehr arbeiten gehen!“

Er führte er aus: „Nach Paragraph StGB Art. 143bis gilt, ich zitiere auszugsweise [2]: Wer ohne Bereicherungsabsicht [...] unbefugterweise in ein fremdes, gegen seinen Zugriff besonders gesichertes Datenverarbeitungssystem eindringt, wird, [...] bestraft. Zu beachten ist dabei, dass dieser Tatbestand nicht das Eindringen in fremde Datenbestände [...] betrifft. [...] Der unbefugte Zugang zu individuellen Daten [...] durch einen zur Benutzung einer Großanlage Befugten fällt nicht unter diesen Tatbestand.“

Der Richter sah mich an. „Das Internet ist ja wohl als Großanlage zu sehen, oder?“

Wütend trommelte ich mit den Fäusten auf den Tisch und erhielt gleich noch eine Strafe wegen Missachtung des Gerichts. In der Urteilsbegründung zog er einen Vergleich zu einem Hund, der einen Einbrecher auch nicht schädigen dürfe [3].

Ich verstand die Welt nicht mehr. Mutlos ging ich nach Hause. Als ich den Stecker meines Routers ziehen wollte, bemerkte ich, dass bis dato Daten über meine Leitung liefen. Tausende von Leuten zogen meine druckbare Drohne von meinem Rechner! Musste ich jetzt jedem, der das Ding tatsächlich ausdrückte eine lebenslange Rente zahlen? Mir wurde heiß und kalt. Klopfenden Herzens zog ich endlich den Stecker.

In den Nachrichten wurde mein Urteil breitgetreten. Weltweit, so der Nachrichtenandroid, hatten sich tausende Arme, Bedürftige oder Arbeitslose

meine Drohne ausgedruckt, offensichtlich in der Absicht so an Geld durch meine Versicherung zu kommen. Als ich aus dem Fenster schaute, beobachtete ich einen Nachbarn, der sich einen Reflektorstreifen auf die Stirn geklebt hatte und meine Drohne vor sich hielt. Die donnerte ihm voll ins Gesicht und explodierte. „Scheiß Arbeitslosigkeit“, murmelte ich. Jeder, der Geld brauchte, würde jetzt probieren, von meiner Berufshaftpflichtversicherung Geld zu erhalten. Ich hatte die Schmach und alle anderen die Gelddruckmaschine ihres Lebens, die sie nur ein einziges Mal schmerzhaft nutzen mussten.

Ich machte mich auf zum Richter. Vielleicht konnte ich ja mit ihm reden und ihm den Irrsinn klar machen. Bald lungerte ich vor seinem Haus herum. „Es sind alles Schmarotzer!“, würde ich sagen. „Suchen sich keine Arbeit. Sehen nur zu, vom Staat oder egal woher Geld zu ergattern. Da sind wohl alle Menschen, Länder und Staaten gleich. Jeder denkt nur an sich selbst und nimmt die Chance wahr, sich auf Kosten anderer versorgen zu lassen!“

Endlich fasste ich Mut und klingelte. Als er die Tür öffnete, stand ich ihm gegenüber. Auf seiner Stirn prangte das Zeichen meiner Drohne.

„Aber Sie haben doch einen Job?“, fragte ich.

„Ja, aber die Arbeitslast ist zu hoch, ich kann nicht mehr. Durch die Computer wurde uns die Arbeit nicht erleichtert, sondern nur beschleunigt.“

Er tat mir fast leid, wie er da stand. Freudlos, kläglich, verloren in der Hektik der Zeit.

Die Tür schloss sich. Ich bummelte nach Hause. Per Smartphone rief ich meine Mails ab und sah, dass die Versicherung mir gekündigt hatte. Nachträglich. Es gäbe da einen Paragraphen in den AGBs, der dies ermöglichte. Was ich jetzt brauchte, war ein guter Anwalt. Am besten ein kostenfreier. Aber taugte einer dieser Open-Source-Anwälte etwas?

Bereits am nächsten Morgen saß ich ihm gegenüber – oder ihr, ich weiß nicht. Es handelte sich um ein Modell Android, das in Big Data nach einem ähnlich gelagerten Fall suchte und tatsächlich einen Ausweg fand!

Ich hatte ja eher an so etwas wie Befangenheit des Richters oder Einbruch in meine Privatsphäre gedacht. Meinem Anwaltsandroiden zeigte ich auf, dass ich meiner Meinung nach, um dem Richterspruch entgegenzuwirken, nur nachweisen bräuchte, dass ich alle mir möglichen Vorkehrungen getroffen hatte, einen Diebstahl zu verhindern. Das hieß, ich belegte ein Schloss vor der Tür zu haben und ein Schloss vor meiner Datenleitung, sprich eine Anti-VirusTrojanerWurmFirewall. Beides hatte ich natürlich installiert!

Mein herzloser Anwalt schüttelte sein Blechhaupt. „Der Paragraph 143bis gilt in der Schweiz, der Tatbestand liegt jedoch in Deutschland. Der Fall ist neu zu verhandeln. In Deutschland stütze ich

mich auf Paragraph 202a in dem ganz klar geregelt ist, dass derjenige eine Geldstrafe zahlt, der sich unbefugt Daten unter Überwindung der Zugangssicherung verschafft. Das weist ein Forensiker durch kriminaltechnische Untersuchungen nach.“

Ich hätte diesen alten Schrotthaufen in die Arme schließen können.

„Sie sollten schnellstmöglich Ihre Konten sperren.“

Ich tat es sofort über mein Smartphone.

Quietschend lehnte sich der Android zurück. „Dachten Sie schon einmal darüber nach,“, blecherte er hochtrabend, „dass Sie ja jetzt tausende Kunden haben, die das von Ihnen entwickelte Gerät nutzen? Sie erkennen jeden einzelnen von Ihnen am Zeichen auf der Stirn. Setzen wir einen einmaligen Preis pro Gerät von 100 Euro an, dann dürften Sie bereits jetzt Millionär sein.“

Er reichte mir seine kalte Hand. „Meinen Glückwunsch.“

LINKS

- [1] <http://www.vice.com/de/read/drake-tattoo-dummheit>
- [2] <http://www.admin.ch/opc/de/classified-compilation/19370083/index.html#a143bis>
- [3] <http://www.123recht.net/Sowas-gibt-es-glaube-ich-nur-bei-uns-in-Deutschland-f58550.html>

Autoreninformation

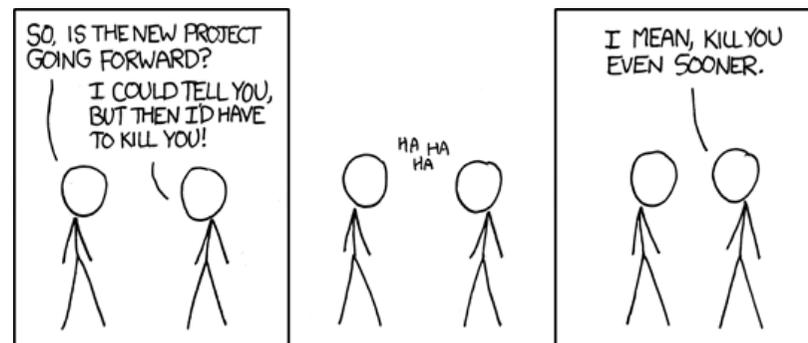
Stefan Wichmann ist kein Anwalt, die Geschichte ist ausgedacht, unter künstlerischer Freiheit erzählt und der Erfinder weiß, dass die Menschen nicht so schlecht sind, wie er es in seiner Bedrängnis dargestellt hat.



Teilen



Kommentieren



“Joshing” © by Randall Munroe (CC-BY-NC-2.5), <http://xkcd.com/707/>

Rezension: Datendesign mit R von Christian Schnell

Der Schwerpunkt des Buches „Datendesign mit R“ von Thomas Rahlf liegt auf ausführlichen Beispielen inklusive der benötigten Skripte für die Visualisierung umfangreicher und komplexer Daten.

Redaktioneller Hinweis: *Wir danken Open Source Press für die Bereitstellung eines Rezensionsexemplares.*

Das Buch teilt sich im Wesentlichen in zwei Teile auf: Den kleineren Teil mit ca. 100 Seiten bilden Grundlagen und Technik. Dagegen nimmt der Teil Beispiele mit ca. 280 Seiten den Hauptteil des Buches ein. Zusätzlich gibt es noch ein Vorwort und einen längeren Anhang.

Nach dem kurzen Vorwort mit Erläuterungen zum Inhalt und Ansprechen der Zielgruppe beginnt der eigentliche Inhalt mit einer Einführung. In dieser wird zunächst auf die Bedeutung und Vielfalt von Datenvisualisierung und statistischen Grafiken in den verschiedensten Bereichen von Alltag, Wirtschaft, Wissenschaft und Journalismus hingewiesen. Nachdem die Vorteile der Nutzung von R in diesem Bereich dargestellt werden, geht es mit dem ersten Teil des Buches zu Grundlagen und Technik los.

Inhalt

Dieser erste Teil besteht aus insgesamt vier Kapiteln. Das erste Kapitel „Aufbau und technische

Voraussetzungen“ startet mit einer Reihe von Definitionen und Begriffen, um die verschiedenen Teile, aus denen sich eine Abbildung zusammensetzen kann, zu erläutern. Der Abschnitt Perzeption, in welchem Anmerkungen zur Wahrnehmung von Daten zu finden sind, behandelt in zwei Beispielen, wie man die dargestellten Informationen besser (nicht) visualisieren sollte. Der Autor bleibt dabei nah an den beiden Beispielen, tiefergehende Erklärungen finden sich hier nicht.

Dagegen nimmt der nächste Abschnitt „Schriften“ einen breiteren Raum ein. Hier findet der Leser Informationen zur historischen Entwicklung der heute benutzten Schriftarten auf den verschiedenen Betriebssystemen und grobe Erläuterungen zu der Klassifikation der verschiedenen Schriftarten. Auch die Quellen für die im Buch benutzten Schriften finden sich hier. Anschließend folgt ein ähnlicher Abschnitt zum Thema Symbole und Piktogramme. Außerdem finden sich in diesem Kapitel noch Abschnitte über Farbmodelle.

Das nächste Kapitel geht nun erstmals detailliert auf R selbst ein. Zunächst wird kurz die Installation erläutert, bevor die Grundkonzepte von R vorgestellt werden. Dazu gehören Datentypen, der Import von Daten aus verschiedenen Quellen wie z. B. Excel-Dateien oder RDF- und Karten-Daten. Wie in einem Buch mit diesem Titel nicht anders zu erwarten, nimmt der Abschnitt „Grafikkonzepte in R“ einen großen Teil ein. Ausführlich wer-

den verschiedene Grafikkonzepte erläutert und die verschiedenen Möglichkeiten zur Einstellung der möglichen Parameter an zahlreichen Beispielen klar gemacht. Am Ende werden diese auch übersichtlich mit einer Kurzbeschreibung aufgelistet. Abgeschlossen wird dieses Kapitel mit einer Auflistung und kurzen Beschreibung der im Buch verwendeten Pakete und Funktionen.

Kapitel vier beschäftigt sich mit der Zusammenarbeit von R mit \LaTeX und Inkscape, zumindest soweit diese für die Inhalte dieses Buches wichtig sind. Eine umfangreiche Einführung sollte man weder für Latex noch für Inkscape erwarten, auch wenn die grundlegenden Schritte und Funktionen erwähnt werden. Der Leser erfährt aber, welche Möglichkeiten sich aus der Zusammenarbeit der drei Programme ergeben.

Der erste Teil des Buches endet mit einer systematischen Einordnung der Möglichkeiten, statistische Daten zu visualisieren. Zudem werden die Voraussetzungen erläutert, damit der Leser die Visualisierungsbeispiele mit den gleichen Daten wie der Autor nachvollziehen kann (Details zur Datenquelle und der Aufbereitung für die Beispiele im Buch finden sich im Anhang).

Entsprechend der oben erwähnten Einordnung werden die Beispiele auf die nun folgenden Kapitel aufgeteilt. Ob kategoriale Daten mit Säulen- oder Kreisdiagrammen, Verteilungen mit

Buchinformationen

Titel	Datendesign mit R [1]
Autor	Thomas Rahlf
Verlag	Open Source Press, 1. Auflage 2014
Umfang	426 Seiten
ISBN	978-3-95539-094-5
Preis	44,90 € (Buch inkl. E-Book), 39,99 € (PDF, E-PUB, MOBI)

Boxplots und Lorenzkurven, Zeitreihen, Streudiagrammen oder Karten – jedes dieser Kapitel enthält zahlreiche Beispiele zur Darstellung der Daten. Die Kapitel starten mit einer kurzen Einführung zum jeweiligen Diagrammtyp, woher die Daten stammen und wie sie aufgebaut sind. Außerdem ist das jeweilige R-Skript abgedruckt, mit dessen Hilfe das Diagramm erstellt wird. In der folgenden ausführlichen Erklärung wird das Skript durchgearbeitet und teilweise Zeile für Zeile erläutert. Der Leser kann die Diagramme also gut mit den Beispieldaten oder auch eigenen Daten nachmachen und die gleiche Darstellung wie der Autor erreichen. Und das lohnt sich durchaus. Die gezeigten Diagramme und Abbildungen sind wirklich gut gelungen und machen Lust aufs Nachvollziehen.

Kritik

Das Buch ist in einem nüchternen Stil geschrieben und liest sich sehr gut. Die Kapitel zu Grundlagen und Technik könnten wohl etwas ausführlicher sein, auch wenn dieses Buch natürlich keine Einleitung in R oder Statistik sein soll. Absolu-

te R-Anfänger gehören allerdings auch nur dann zur Zielgruppe, wenn sie die Beispiele genauso wie abgedruckt nachbauen möchten. Auch wenn die Skripte gut erläutert werden, benötigt man für größere Änderungen doch tiefergehende R-Kenntnisse.

Die Diagramme selbst sind sehr anschaulich und gut gestaltet, sodass sie sich durchaus als Vorlage für die Visualisierung eigener Daten eignen können. Der Leser profitiert hier eindeutig von den Erfahrungen des Autors.

Positiv hervorzuheben sind außerdem die guten Quellenangaben im Text und im Anhang, sodass der interessierte Leser schnell weiterführende Informationen und Originalquellen finden kann, wenn er sich für ein angesprochenes Thema weiter interessiert.

Fazit

Wenn man keine Einführung in R und Statistik an sich erwartet, sondern auf der Suche nach anschaulichen Beispielen inklusive Skripten für die Visualisierung eigener Daten ist, sollte man sich dieses Buch auf jeden Fall anschauen.

LINKS

- [1] <http://www.opensourcepress.de/de/produkte/Datendesign-mit-R/13587/978-3-95539-094-5>



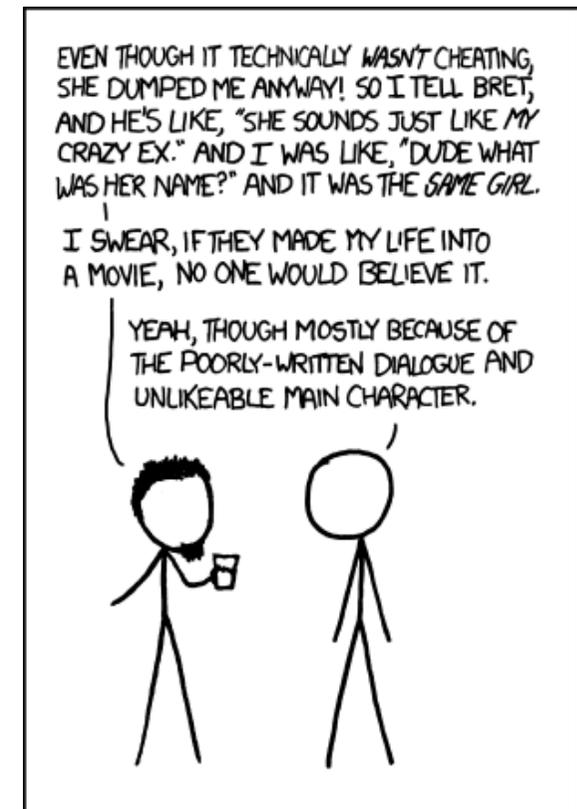
Teilen



Kommentieren

Autoreninformation

Christian Schnell nutzt R im beruflichen Umfeld zur Darstellung von Ergebnissen aus neurowissenschaftlichen Experimenten und hat aus diesem Buch einige Punkte zur Verbesserung mitnehmen können.



“Sloppier Than Fiction” © by Randall Munroe (CC-BY-NC-2.5), <http://xkcd.com/1006/>

Rezension: Design Patterns mit Java von Dominik Wagenführ

Wer bei der „Gang of Four“ zuerst an die Daltons [1] denkt, hat sicherlich wenig mit Entwurfsmustern (Englisch: Design Patterns) zu tun. Wenn man aber zur programmierenden Zunft in einer objektorientierten Sprache gehört, sind Entwurfsmuster oft das A und O und sollten von jedem zumindest einmal gehört worden sein. Die Rezension soll zeigen, ob das Buch „Design Patterns mit Java“ dabei helfen kann.

Redaktioneller Hinweis: Wir danken dem Carl Hanser Verlag für die Bereitstellung eines Rezensionsexemplares.

Geschichtliches

Entwurfsmuster reichen (in der Zeitrechnung der Programmierung) schon ziemlich lange zurück. Es handelt sich dabei um Muster, die man zur Lösung immer wiederkehrender Probleme und Fragestellungen nutzen kann. Auch wenn die Gang of Four (kurz GoF), dazu gehören Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides, die Entwurfsmuster nicht erfunden haben, haben sie 1995 mit ihrem Buch „Design Patterns. Elements of Reusable Object-Oriented Software“ [2] zumindest sehr zu deren Verbreitung beigetragen.

In dem GoF-Buch sind 23 Design-Patterns beschrieben mit Beispielanwendungen in der Programmiersprache C++, die der Autor Florian Siebler in seinem Buch „Design Patterns mit Java“

ebenfalls aufgreift, erklärt und beispielhaft mit Java umgesetzt.

Buchaufbau

Das Buch ist, in Anlehnung an die Gang of Four, in 24 Kapitel untergliedert: 23 für Entwurfsmuster und eines für Designprinzipien, darunter auch ein Teil der SOLID-Prinzipien [3]. Die Kapitel bauen dabei so gut wie gar nicht aufeinander auf, sodass man nicht gezwungen ist, das Buch von vorne nach hinten durchzulesen. Man kann sich auch einfach nur mit einem Kapitel beschäftigen, wenn einen ein bestimmtes Entwurfsmuster interessiert.

In jedem Kapitel wird das jeweilige Entwurfsmuster vorgestellt und mindestens ein kleines Beispiel dazu gezeigt. In manchen Kapiteln wird dabei auch erst eine naive Implementierung in Java gezeigt und auf deren Vor- und Nachteile hingewiesen. Danach folgt meist eine Verbesserung der Umsetzung oder eine alternative Lösung. Dies ist vor allem in den ersten zwei Kapiteln zum Singleton-Pattern und Observer-Pattern in Bezug auf nebenläufige Verwendung sehr hilfreich.

Zusätzlich zeigt Florian Siebler auch immer eine Verwendung des Entwurfsmusters in einer der Java-Bibliotheken auf, sodass Leser sofort sehen, dass die Muster keine theoretischen Überlegungen sind, sondern auch in der realen Welt zum Einsatz kommen.

Die Listings und Beispiele sind dabei oft nur gekürzt enthalten. Der Autor verweist dazu immer wieder auf die Webseite zum Buch patterns-Buch.de [4].

Etwas überraschend ist der Abschluss des Buches, denn es gibt keinen. Nach den 23 Entwurfsmustern kommt der Index und es ist Schluss.

Zielgruppe

Das Buch heißt nicht umsonst „Design Patterns mit Java“, denn Java wird zum Verständnis der Muster bzw. deren Umsetzung vorausgesetzt. Vor allem die Hinweise auf die Verwendung der Entwurfsmuster in den Java-Bibliotheken verstehen oft nur die Java-Entwickler, welche die jeweilige Bibliothek auch schon im Einsatz hatten. Das macht ein Teil des Buches für Nutzer anderer Programmiersprachen leider hinfällig, da man nicht immer folgen kann und diese Abschnitte dann eher überspringt.

Dennoch kann man auch ohne (große) Java-Kenntnisse den Umsetzungen folgen und diese in einer Sprache seiner Wahl nachprogrammieren. Von den online bereitgestellten Beispielen hat man in dem Fall natürlich sehr wenig.

Die online bereitgestellten Beispiele zeigen aber auch, dass die Zielgruppe am besten über einen Internetanschluss verfügt, wenn man das Buch durcharbeiten möchte. Denn ansonsten fällt es

manchmal etwas schwer, nur anhand der Code-Auszüge der Umsetzung zu folgen.

Design Patterns en detail

Florian Siebler geht sehr gut auf die einzelnen Entwurfsmuster ein. Wie oben schon erwähnt zeigt er nicht nur verschiedene Realisierungen mit deren Vor- und Nachteilen auf, sondern geht auch auf Besonderheiten wie Nebenläufigkeit ein – etwas, was bei anderen Design-Patterns-Büchern gerne unterschlagen wird. Die vorgestellten Lösungen sind dabei natürlich Java-abhängig und in anderen Sprachen muss man sich bei Nebenläufigkeit andere Lösungen ausdenken bzw. die vorgestellte Idee zumindest anders umsetzen. Sehr schön ist auch, dass er auf sogenannte Anti-Patterns eingeht, zu denen sehr viele Entwickler beispielsweise das Singleton-Pattern zählen.

Für die Erklärung der Patterns benutzt Siebler mehr Text und wenig Klassen-Diagramme. Als Grund hierfür führt der Autor an, dass die Diagramme der unterschiedlichen Entwurfsmuster oft sehr ähnlich bzw. sogar identisch aussehen. Einem Einsteiger in Entwurfsmuster bringen diese Diagramme daher wenig. So werden sie auch nur gezielt dort eingesetzt, wo es der Übersicht hilft. Dennoch wäre eine Gesamtübersicht der Entwurfsmuster mit deren Klassenbeziehungen am Ende des Buches schön gewesen.

Ein Problem bei der Umsetzung der Entwurfsmuster im Buch ist aber, dass nicht jede Realisierung sofort klar ist. Beispielsweise wird beim

State-Pattern eine Lösung eingesetzt, in der ein Zustand das Hauptobjekt und seine Folgezustände kennen muss. Da man normalerweise eine lose Koppelung anstrebt, ist das verwunderlich, wird aber später erklärt. Dies passiert dem Autor an vielen Stellen im Buch, dass die Motivation für eine Umsetzung oder einen Exkurs (wie z. B. die Tabellenmodelle auf Seite 45) erst ganz am Ende des Abschnitts erklärt werden. Als Leser fragt man sich aber viel eher, wieso etwas auf eine bestimmte Art und Weise gemacht wird oder was dieser Exkurs genau bringen soll.

Ob die Auswahl der Patterns sinnvoll ist, darüber lässt sich sicherlich streiten. Florian Siebler orientiert sich an den Entwurfsmustern der Gang of Four, was so schlecht nicht sein kann. Dabei wird aber beispielsweise auch das Interpreter-Pattern erklärt, was so speziell und komplex ist, dass es vor allem für Einsteiger schwer zu verstehen ist.

Fazit

Mit dem Buch „Design Patterns mit Java“ will Florian Siebler laut Untertitel eine „Einführung in Entwurfsmuster“ geben. Vergleicht man direkt mit dem GoF-Buch, was ja als Vorlage diente, so ist ihm das gelungen. Wo das GoF-Buch doch eher trocken herüberkommt, gibt es bei Siebler meist einfach zu verstehende Beispiele, die teilweise mit Pop-Culture-Referenzen gespickt sind. So liest sich das Buch recht flüssig und unterhaltsam.

Ohne Java-Kenntnisse fällt es aber oft schwer zu folgen. Vor allem die Nachweise in den Java-

Bibliotheken sind nicht immer verständlich, wenn man die Bibliothek noch nie genutzt hat. Glücklicherweise sind die Nachweise aber nur ein Zusatz. Die Entwurfsmuster werden immer auch an einem eigenen Beispiel erprobt.

Diese Beispiele wiederum werden leider nur zum Teil abgedruckt. Es ist so nicht möglich, ohne Internetzugang allen Ausführungen zu folgen bzw. diese nachzuimplementieren. Teilweise empfindet man die zahlreichen Verweise auf die Buchwebseite patternsbuch.de sogar als störend, weil sie den eigentlichen Gedanken- bzw. Arbeitsfluss unterbrechen.

Und wenn es um Entwurfsmuster und Java geht, drängt sich ein Vergleich mit dem Buch „Entwurfsmuster von Kopf bis Fuß“ („Head First Design Patterns“, [freiesMagazin](http://www.freiesMagazin.de) 03/2012 [5]) natürlich auf. Hier hat das Head-First-Buch nicht nur den Kopf, sondern auch die Nase vorn. Didaktisch ist es so wieso nicht zu schlagen, aber alle Beispiele sind so einfach, verständlich und vor allem unterhaltsam gehalten, dass das Lesen und Durcharbeiten einfach nur Spaß macht. Vor allem können auch Nicht-Java-Programmierer das Head-First-Buch fast ohne Probleme durcharbeiten (wenn auch nicht ganz).

Wer also eine etwas nüchterne, aber dennoch unterhaltsame Erklärung der GoF-Entwurfsmuster sucht und diese auch noch mit Java umsetzen will, kann ohne Bedenken zum Buch von Florian Siebler greifen. Wer dagegen keine Java-Kenntnisse

Buchinformationen

Titel	Design Patterns mit Java– Eine Einführung in Entwurfsmuster [6]
Autor	Florian Siebler
Verlag	Carl Hanser Verlag, 2014
Umfang	311 Seiten
ISBN	978-3-446-43616-9
Preis	29,99 Euro (Print), 23,99 Euro (PDF)

hat und das eher trockene GoF-Buch mit seinen C++-Beispielen nicht durcharbeiten möchte, dem sei „Entwurfsmuster von Kopf bis Fuß“ empfohlen (am besten im englischen Original, was weniger Fehler aufweist).

Redaktioneller Hinweis: Da es schade wäre, wenn das Buch bei Dominik Wagenführ im Regal verstaubt, wird es verlost. Die Gewinnfrage lautet:

„Wie viele Entwurfsmuster sind im GoF-Buch beschrieben?“

Die Antwort kann bis zum **14. September 2014, 23:59 Uhr** über die Kommentarfunktion oder per E-Mail an redaktion@freiesMagazin.de geschickt werden. Die Kommentare werden bis zum Ende der Verlosung nicht freigeschaltet. Das Buch wird unter allen Einsendern, die die Frage richtig beantworten konnten, verlost.

LINKS

- [1] [https://de.wikipedia.org/wiki/Die_Daltons_\(Comicfiguren\)](https://de.wikipedia.org/wiki/Die_Daltons_(Comicfiguren))
- [2] [https://de.wikipedia.org/wiki/Entwurfsmuster_\(Buch\)](https://de.wikipedia.org/wiki/Entwurfsmuster_(Buch))
- [3] https://de.wikipedia.org/wiki/Prinzipien_objektorientierten_Designs#SOLID-Prinzipien

- [4] <http://patternsbuch.de/>
- [5] <http://www.freiesmagazin.de/freiesMagazin-2012-03>
- [6] <http://www.hanser-fachbuch.de/buch/Design+Patterns+mit+Java/9783446436169>

Autoreninformation

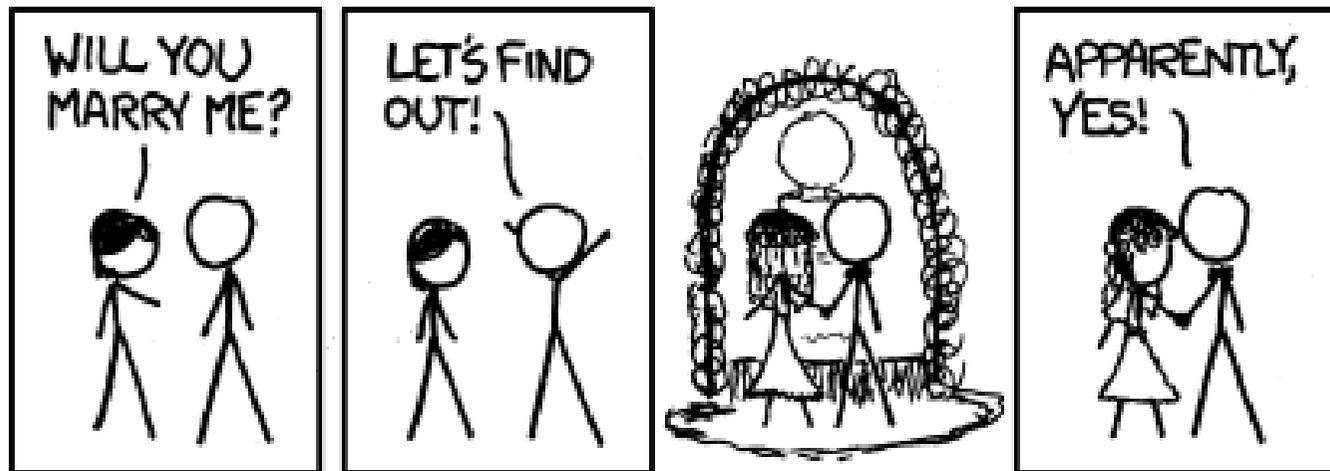
Dominik Wagenführ ([Webseite](#)) ist Software-Entwickler und hat fast täglich mit Design Patterns zu tun. Einige Beschreibung im Buch haben ihm davon auch ein besseres Verständnis gegeben.



Teilen



Kommentieren



“Empirical” © by Randall Munroe (CC-BY-NC-2.5), <http://xkcd.com/943/>

Rezension: Android – Einstieg in die Programmierung von Sujeewan Vijayakumaran

Das Buch „Android – Der schnelle und einfache Einstieg in die Programmierung und Entwicklungsumgebung“ von Dirk Louis und Peter Müller erschien in der ersten Auflage im Carl Hanser Verlag.

Redaktioneller Hinweis: *Wir danken dem Carl Hanser Verlag für die Bereitstellung eines Rezensionsexemplares.*

Was steht drin?

Das Buch umfasst insgesamt 20 Kapitel, abgedruckt auf 474 Seiten. Alle Kapitel sind übergeordnet in die Teile „Einführung“, „Grundlagen“ und „weiterführende Themen“ unterteilt.

Neben den 20 Kapiteln gibt es einen Anhang mit zusätzlichem Material sowie eine DVD. Letztere beinhaltet neben der benötigten Software weitere Anleitungen, wie etwa ein Java-Tutorium.

Der erste Teil führt in die Grundlagen der Android-Entwicklung ein. Das erste Kapitel behandelt die Einrichtung von Eclipse, die Installation von JRE sowie die Nutzung des Android-SDK-Managers. Die Installationsanleitung thematisiert lediglich die Windows-Installation. Wie man genau die Einrichtung unter Linux und Mac OS durchführt, wird daher in dem Buch leider nicht beschrieben.

Bereits im zweiten Kapitel wird die erste kleine „Hallo Welt!“-App entwickelt. Es wird dabei Schritt

für Schritt sowohl der Umgang mit Eclipse als auch das Ausführen der App in einem Emulator erklärt. Das dritte und letzte Kapitel der Einführung erläutert ausführlich den Aufbau eines Android-Projekts.

Mit dem vierten Kapitel starten die Grundlagenkapitel. Zunächst wird der allgemeine Umgang mit Eclipse erklärt, aber nicht nur die grundlegende Bedienung dargestellt, sondern es werden auch zahlreiche Tipps und Tricks gegeben, welche das Arbeiten mit Eclipse deutlich erleichtern.

Im fünften Kapitel folgt die Einweisung in das Design von Android-Apps. Es wird ausführlich erklärt, welche Design-Prinzipien man beachten sollte, und wie man diese auch umsetzen kann. Das anschließende sechste Kapitel behandelt die Nutzung von Ressourcen, also etwa die Nutzung von Grafiken, Größen-Definitionen oder auch Strings.

Das siebte Kapitel enthält das Thema der Anwender-Interaktion. Fragen, wie zum Beispiel was Views sind und wie sie arbeiten, werden genauso beantwortet wie die Nutzung von normalen OnClick-Listenern oder auch die Implementierung von Wisch-Gesten.

Ein sehr wichtiges und teilweise besonders für Anfänger kompliziertes Thema ist der Lebenszyklus einer Android-App. Dies ist das Thema des ach-

ten Kapitels, welches zudem noch einige Grundlagen erläutert, was sich unter der Haube einer Android-App befindet.

Der dritte Teil des Buches behandelt weiterführende und vertiefende Themen. Kapitel 9 erläutert zuerst die Nutzung von Canvas innerhalb von Apps, während anschließend das zehnte Kapitel die Nutzung von Menüs und Dialogen beinhaltet.

Das elfte Kapitel gehört zwar auch zu den weiterführenden Themen, behandelt aber ein wichtiges Thema: die Intents.

Kapitel 12 beinhaltet wiederum ein klassisches Java-Thema, nämlich das Arbeiten mit Dateien. Dabei wird auch auf die Android-spezifischen Sonderfälle eingegangen, wie das Schreiben auf externen Speicher.

Im 13. Kapitel wird erstmals eine vollständige App entwickelt. Anhand einer Quiz-App werden die vorher erlernten Dinge gesammelt nochmals erläutert. Die darauf folgenden Kapitel behandeln weiterhin die Themen Multimedia, die Einbindung von Sensoren, den Einsatz von SQLite-Datenbanken sowie die Geolokalisation.

Das 18. Kapitel fasst nochmals viele erlernte Dinge aus den vorherigen Kapiteln in einer Beispiel-App zusammen, dieses Mal in Form eines Tic-Tac-Toe Spiels. Interessanterweise landete die

Thematik rund um Fragments, die in Apps genutzt werden können, fast zum Schluss im 19. Kapitel, obwohl diese mittlerweile eines der wichtigsten Elemente jeder App sind. Im letzten Kapitel werden schlussendlich noch einige Tipps und Tricks genannt.

Anschließend wartet noch der Anhang, der ebenfalls in drei Teile unterteilt ist. Dort werden weiterführende Themen behandelt, die nach der Fertigstellung einer App warten, etwa dem Veröffentlichen der App im Google Play Store.

Wie liest es sich?

Das Buch setzt keine Kenntnisse mit der Android-Entwicklung voraus. Nichtsdestoweniger werden gute Kenntnisse in Java benötigt, um mit dem Buch sinnvoll arbeiten zu können. Der Umgang mit der Entwicklungsumgebung Eclipse wird hingegen nicht vorausgesetzt. Die Nutzung wird an vielen Stellen ausführlich genug erläutert und mit zahlreichen Screenshots verbildlicht. Java-Neulinge sollten vor dem Anschaffen dieses Buches zunächst auf anderem Wege Java erlernen. Viele Dinge bezüglich der Java-Programmierung werden trotzdem häufig genug am Rande erläutert oder es wird auf die Buch-DVD verwiesen. Dort ist nämlich auch ein Java-Tutorium enthalten.

Die Erklärungen in dem Buch sind grundsätzlich gut beschrieben und werden ausführlich erläutert, sodass erfahrene Java-Entwickler keine Probleme in den Einstieg in die Android-Programmierung

haben sollten. Die wichtigsten Themen rund um die Android-Entwicklung werden grundsätzlich abgedeckt. Das Buch umfasst auch zahlreiche Bilder, welche häufig wichtig sind, um etwa die richtigen Schaltflächen schnell zu finden.

Kritik

An diesem Buch gibt es wenig auszusetzen. Für knapp 30 € findet man mit diesem Buch einen guten Einstieg in die Entwicklung von Android-Apps. Die Buchseiten sind komplett schwarz-weiß, farbige Seiten werden in dem Buch allerdings sowieso nicht gebraucht.

Weiterhin bietet das Buch eine gute Abwechslung zwischen Theorie und Praxis. Besonders die beiden Praxis-Beispiele anhand der beiden Apps bieten dadurch nicht nur eine gute Abwechslung, sondern fassen den vorher gelernten Inhalt sinnvoll zusammen. Allwissend wird man durch dieses Buch allerdings nicht. Wer eine App vollständig entwickeln will, kommt um weitergehende Lektüre nicht herum, was auch nicht Ziel des Buches ist. Das Ziel eines schnellen und einfachen Einstiegs in die Android-Entwicklung wird das Buch durchaus gerecht.

Redaktioneller Hinweis: Da es schade wäre, wenn das Buch bei Sujeevan Vijayakumaran im Regal verstaubt, wird es verlost. Die Gewinnfrage lautet:

„Nach welchem Produkt ist der Releasename der aktuellen Android-Version benannt?“

Die Antwort kann bis zum **14. September 2014, 23:59 Uhr** über die Kommentarfunktion oder per E-Mail an redaktion@freiesMagazin.de geschickt werden. Die Kommentare werden bis zum Ende der Verlosung nicht freigeschaltet. Das Buch wird unter allen Einsendern, die die Frage richtig beantworten konnten, verlost.

Buchinformationen

Titel	Android – Der schnelle und einfache Einstieg in die Programmierung und Entwicklungsumgebung [1]
Autor	Dirk Louis, Peter Müller
Verlag	Carl Hanser Verlag, 2014
Umfang	474 Seiten
ISBN	978-3-446-43823-1
Preis	Preis: 29,99 € (broschiert), 23,99 € (E-Book)

LINKS

[1] <http://www.hanser-fachbuch.de/buch/Android/9783446438231>

Autoreninformation

Sujeevan Vijayakumaran interessiert sich sehr für das Mobile-Computing und programmiert hin und wieder auch an seiner eigenen Android-App.



Teilen



Kommentieren

Leserbriefe

Für Leserbriefe steht unsere E-Mailadresse redaktion@freiesMagazin.de zur Verfügung – wir freuen uns über Lob, Kritik und Anregungen zum Magazin.

An dieser Stelle möchten wir alle Leser ausdrücklich ermuntern, uns auch zu schreiben, was nicht so gut gefällt. Wir bekommen sehr viel Lob (was uns natürlich freut), aber vor allem durch Kritik und neue Ideen können wir uns verbessern.

Leserbriefe und Anmerkungen

Administration von Debian & Co – Teil I

❖ Das Tutorial „Administration von Debian & Co im Textmodus“ ist wirklich ausgezeichnet und selbst für Anfänger nachvollziehbar. Eine Sache finde ich aber doch schade, das ich bis zum zweiten Teil einen Monat warten muss. Danke, dass Ihr Euch immer solch eine Mühe gebt.

Matthias

❖ Meine Bemerkungen:

1. Wozu dient die IPCop-VM?
2. Statt **sudo bash** wäre ein **sudo -i** besser.
3. Debian hat zwei Standard-Shells. Für Systembenutzer wird dash genutzt und für normale Benutzer bash.
4. Der Teil über die Zugriffsrechte ist sehr schön erklärt und sehr verständlich.

Ansonsten finde ich, dass dieser erste Teil viele oft benutzte Kommandos/Programme enthält

und recht gut erklärt. Teilweise könnt ihr die Erklärungen noch ausführlicher gestalten. Auf jeden Fall bin ich auf die nächsten beiden Teilartikel gespannt.

txt.file ([Kommentar](#))

❖ *Vielen Dank für das Lob. Hier die Antwort zu den Anmerkungen:*

1. *Die IPCop-VM sorgt für eine definierte Netzwerkkumgebung, so dass Dinge wie DHCP und Netzwerkeinrichtung (Teil II) im Workshop für alle, die mitmachen, gleich sind. [. . .]*
2. *Stimmt. Wir dachten hier vor allem an die Shell, die man als Benutzer zu sehen bekommt, also die „Standard-Interaktive-Shell“ (default interactive shell).*

Maren Hachmann

Lob

❖ Dank der EPUB-Ausgabe [von] freiesMagazin kann ich es auf meinem Kobo-E-Book-Reader jederzeit lesen. Auch auf der Terrasse in der Sonne. Macht bitte weiter so.

Franz-Josef

❖ [Ich möchte Euch] einfach mal ein Lob dalassen, ohne Hintergedanken. Ich finde das Magazin sowohl von der Aufmachung her, als auch was die Inhalte betrifft einfach super. Inzwischen lade ich am liebsten die EPUB-Variante herunter, weil die sich so schön am Tablet oder Smartphone lesen lässt. Manche Artikel würde ich wahrschein-

lich nicht lesen, wenn sie nicht im Magazin wären aber so blättert man sich durch und erweitert seinen Horizont an der einen oder anderen Stelle.

Ihr könnt Euch von diesem Lob nichts kaufen, aber als jemand, der in einer IT-Abteilung arbeitet, weiß ich, dass es nicht so oft Lob gibt, wenn alle Systeme einfach funktionieren. Wenn dann mal was brennt wird aber sofort geschrien. Deswegen hier einfach mal die Ermunterung, die gute Arbeit weiterzumachen.

Viktor

Artikelwünsche

❖ Gerne würde ich einen Artikel zu Ubuntu (14.04) und Grafikkartentreiber installieren (AMD oder Nvidia) und chroot-Umgebung in einer Live-CD-Session vorschlagen.

Herr K.

❖ *Wir nehmen das Thema gerne auf unsere Artikelwunschliste [1] mit auf. Vielleicht findet sich ein Autor hierfür.*

Dominik Wagenführ

LINKS

[1] <http://www.freiesmagazin.de/artikelwuensche>

Die Redaktion behält sich vor, Leserbriefe gegebenenfalls zu kürzen. Redaktionelle Ergänzungen finden sich in eckigen Klammern.

 [Teilen](#)

 [Kommentieren](#)

Veranstungskalender

Messen

Veranstaltung	Ort	Datum	Eintritt	Link
MRMCD14	Darmstadt	05.09.–07.09.2014	42 EUR	https://mrmcd.net/
Datenspuren	Dresden	13.09.–14.09.2014	frei	http://www.datenspuren.de/
play14	Hamburg	16.09.–20.09.2014	10 EUR	http://www.play14.de/
Kieler Linuxtage	Kiel	19.09.–20.09.2014	frei	http://www.kieler-linuxtage.de/
Software Freedom Day	Weltweit	20.09.2014	frei	http://softwarefreedomday.org/
GUUG FFG2014	Bochum	23.09.–29.09.2014	–	https://www.guug.de/veranstaltungen/ffg2014/
VCFB	Berlin	03.10.–05.10.2014	–	http://www.vcfb.de/
8. LIT-OL	Oldenburg	17.10.–18.10.2014	frei	http://www.lit-ol.de
Ubucon	Katlenburg	17.10.–19.10.2014	15 EUR	http://ubucon.de/
OpenRheinRuhr	Oberhausen	08.11.–09.11.2014	–	http://www.openrheinruhr.de/
LinuxDay	Dornbirn	22.11.2014	frei	http://linuxday.at/
data2day	Karlsruhe	26.11.–28.11.2014	–	http://www.data2day.de/

(Alle Angaben ohne Gewähr!)

Sie kennen eine Linux-Messe, welche noch nicht auf der Liste zu finden ist? Dann schreiben Sie eine E-Mail mit den Informationen zu Datum und Ort an redaktion@freiesMagazin.de.

Vorschau

freiesMagazin erscheint am ersten Sonntag eines Monats. Die Oktober-Ausgabe wird voraussichtlich am 5. Oktober u. a. mit folgenden Themen veröffentlicht:

- Siebter **freiesMagazin**-Programmierwettbewerb: Tron
- Administration von Debian & Co im Textmodus – Teil III
- Reprepro – Debian-Systeme mit einem selbst aufgesetzten Paket-Repo versorgen
- Rezension: Seven Web Frameworks in Seven Weeks
- Rezension: Spieleprogrammierung mit Android Studio

Impressum

freiesMagazin erscheint als PDF, EPUB und HTML einmal monatlich.

Kontakt

E-Mail redaktion@freiesMagazin.de
 Postanschrift **freiesMagazin**
 c/o Dominik Wagenführ
 Beethovenstr. 9/1
 71277 Rutesheim
 Webpräsenz <http://www.freiesmagazin.de/>

Autoren dieser Ausgabe

Hauke Goos-Habermann und Ma- **S. 5**
 ren Hachmann
 Markus Herrmann **S. 19**
 Mathias Menzer **S. 3**
 Jennifer Rößler **S. 26**
 Christian Schnell **S. 39**
 Maria Seliger **S. 30**
 Christian Stankowic **S. 13**
 Sujeevan Vijayakumaran **S. 44**
 Dominik Wagenführ **S. 41**
 Stefan Wichmann **S. 35**

ISSN 1867-7991

Erscheinungsdatum: 7. September 2014

Redaktion

Christian Schnell Matthias Sitte
 Dominik Wagenführ (Verantwortlicher Redakteur)

Satz und Layout

Jonas Dabelow Moritz Kiefer
 Christoph Lehmann Kai Welke

Korrektur

Daniel Braun Frank Brungräber
 Vicki Ebeling Stefan Fangmeier
 Mathias Menzer Christian Schnell
 Karsten Schuldt

Veranstaltungen

Ronny Fischer

Logo-Design

Arne Weinberg (CC-BY-SA 4.0 Unported)

Dieses Magazin wurde mit \LaTeX erstellt. Mit vollem Namen gekennzeichnete Beiträge geben nicht notwendigerweise die Meinung der Redaktion wieder. Wenn Sie freiesMagazin ausdrucken möchten, dann denken Sie bitte an die Umwelt und drucken Sie nur im Notfall. Die Bäume werden es Ihnen danken. :-)

Soweit nicht anders angegeben, stehen alle Artikel, Beiträge und Bilder in freiesMagazin unter der Creative-Commons-Lizenz CC-BY-SA 4.0 International. Das Copyright liegt beim jeweiligen Autor. Die Kommentar- und Empfehlen-Icons wurden von Maren Hachmann erstellt und unterliegen ebenfalls der Creative-Commons-Lizenz CC-BY-SA 4.0 International. freiesMagazin unterliegt als Gesamtwerk der Creative-Commons-Lizenz CC-BY-SA 4.0 Unported mit Ausnahme der Inhalte, die unter einer anderen Lizenz hierin veröffentlicht werden. Das Copyright liegt bei Dominik Wagenführ. Es wird erlaubt, das Werk/die Werke unter den Bestimmungen der Creative-Commons-Lizenz zu kopieren, zu verteilen und/oder zu modifizieren. Die xkcd-Comics stehen separat unter der Creative-Commons-Lizenz CC-BY-NC 2.5 Generic. Das Copyright liegt bei Randall Munroe.