



Die Androiden-Fibel

Kleiner Androiden-Führer
für
Einsteiger und Fortgeschrittene
von
Andreas Itzchak Rehberg

[Version 63](#)¹

Dieses eBook unterliegt einer [Creative Commons Lizenz](#)². Es darf also auf jeden Fall in unveränderter Form weitergegeben – und bei Quellen-Nennung auch zitiert werden. Weitere Details finden sich im Lizenz-Text, der mit dem folgenden Bild verlinkt ist:



Die jeweils aktuellste Version dieses eBooks findet sich [hier](#)³.

Cover-Design: Izzy
(Foto: Kolumbus-Denkmal am Hafen von Barcelona, © Izzy 2014)

Unter dem Namen **Das inoffizielle Android-Handbuch** (ISBN: [978-3-645-60311-5](#)⁴) gibt es eine inhaltlich leicht abweichende Print-Ausgabe dieses Buches vom Franzis-Verlag (4. Auflage, 2014). Eine Neuauflage erscheint voraussichtlich im Frühjahr 2016 bei Wiley-VCH im Vierfarb-Druck.

1. <https://ebooks.qumran.org/books/androidenfibel/history.html#v63>

2. <https://creativecommons.org/licenses/by-nc-sa/3.0/deed.de>

3. <https://ebooks.qumran.org/androidenfibel>

4. https://www.amazon.de/gp/product/3645603115/ref=as_li_tf_t1?ie=UTF8&tag=izzyrehbergsi-21



Andreas Itzchak Rehberg

Die

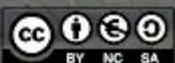
Androiden-Fibel

Für Einsteiger und Fortgeschrittene

Grundlagen - Sicherheit - Privatsphäre - Apps - Tuning -
Permissions - Alltagsprobleme - Begriffserklärungen



IzzyOnAndroid



INHALT

0. Vorwort	3
1. Grundlagen für den Einsteiger	6
1. Grundlegendes zur Bedienung des Androiden	7
2. Google Konto	18
3. Schaltzentrale: Home-Screen, Widgets & „Home Replacements“	21
2. Mit Android arbeiten	25
1. Steuerzentrale: Einstellungen und „Switches“ (Konfiguration)	25
2. Anwendungen verwalten (Installieren, Aktualisieren, Bereinigen)	32
3. Apps organisieren	46
4. Datensicherung	49
5. Zurücksetzen	60
6. Von Taskkillern und anderen bösen Buben	64
7. Datenaustausch mit dem PC	65
8. Datenaustausch zwischen Android-Geräten	69
9. Den Funktionsumfang mit Apps erweitern	74
10. Benutzer und Profile	97
3. Sicherheit	99
1. Was brauche ich wirklich?	99
2. GMV	100
3. Firewall und Anti-Virus: Worum handelt es sich da eigentlich?	102
4. Anti-Virus und Anti-Malware	104
5. Bei Diebstahl und Verlust	106
6. Worauf Apps Zugriff haben	109
7. Apps vor unbefugtem Zugriff schützen	111
8. Kinderschutz	113
9. In fremden Netzen	115
4. Privatsphäre	118
1. Privacy First?	119
2. kontakte und Kalender	121
3. Ortsdaten	123
4. Welche Daten sammelt Google eigentlich?	125
5. Digitales Testament	130
6. Welche Apps und Unternehmen sind sonst noch fleißig am Sammeln?	131
7. Die Cloud	136
8. Google Now	138
9. Zwischenbilanz	143

10.	Weitere Aspekte	144
11.	Werbefinanzierte Apps	145
12.	Was bringen sichere Apps, wenn die Schnüffler ohnehin schon im System sitzen?	151
13.	Gibt es noch mehr zu beachten?	153
14.	Es sind doch nur Verbindungsdaten!	155
5.	Fragen aus Alltag und Praxis	157
1.	Google Account	157
2.	Google Play Store	159
3.	Apps	170
4.	Backup	176
5.	Medien	179
6.	Umgang mit der SD-Karte	181
7.	Netzwerk	185
8.	Telefonie	189
9.	Sicherheit & Privatsphäre	193
10.	Weiteres	200
6.	Tiefergehendes für Fortgeschrittene	205
1.	Der Super-User „root“	206
2.	Apps am automatischen Starten hindern	211
3.	Vorinstallierte Apps entfernen	214
4.	Tuning – Das Android-System auf Trab bringen	215
5.	Durststrecke – mehr aus dem Akku herausholen	222
6.	ROMs: Stock, Vendor, und Custom	231
7.	Zugriffe sperren: Firewalls & Permission-Blocker	234
8.	ADB: Die Android Debug Bridge	239
7.	Anhang	252
1.	Android Permissions – und was sie bedeuten	252
2.	Secret Codes oder Magische Nummern	262
3.	Leistungsaufnahme verschiedener Komponenten	265
4.	Begriffserklärungen	268

VORWORT



Izzy

Eine Fibel? Ist das nicht etwas für Erstklässler? Auch das. Doch die [Begriffserklärung bei Wikipedia](#)⁵ versteht darunter ebenfalls ganz *allgemein ein bebildertes Handbuch oder Nachschlagewerk zu einem bestimmten Thema*. In diesem Fall natürlich zum Thema "Android". Ursprünglich (2011) als eine thematisch sortierte Sammlung von Apps begonnen, hat sich der Charakter des Buches mittlerweile stark gewandelt, und es ist ein ausgewachsenes Android-Handbuch daraus geworden. Die Übersichten finden sich nun bei [IzzyOnDroid](#)⁶, und werden nach wie vor aus diesem Werk referenziert.

Es handelt sich bei diesem eBook um eine Übersicht, die den Einstieg erleichtern soll – doch fortgeschrittene Anwender kommen ebenfalls durchaus auf ihre Kosten: Auch vor tiefer gehenden Themen wird hier nicht „Halt gemacht“. Hoffentlich immer auf eine Art, die beiden Gruppen gerecht wird.

Eine Haupt-Quelle meiner Recherchen ist [Stack Exchange](#)⁷. Diese große, internationale Community tauscht sich in vielen „Stacks“ (was man hier mit „Fachbereiche“ wiedergeben könnte) aus. Bei den Begriffs-Erklärungen habe ich [Stack Exchange](#) detaillierter beschrieben. Dort bin ich u. a. im Android-Bereich aktiv unterwegs, und fühle mich richtig zu Hause – auch wenn die englische Sprache Mittel zur Verständigung ist. Auf Fragen erhält man nirgends so schnell und so fachkundig Antworten wie bei [Stack Exchange™](#)!



Stack Exchange™ Logo

Noch eins muss ich loswerden: Viele der hier kurz vorgestellten (oder auch nur genannten) Apps habe ich selbst nie getestet – etwa, weil ich nicht die Voraussetzungen dazu habe (ich nutze kein Facebook, und mein Smartphone auch nicht zum Spielen, um nur zwei Dinge zu nennen). Ein Grund mehr, auf die Erfahrungen der Community zurückzugreifen.

Zu guter Letzt noch ein kleiner technischer Hinweis: Sofern der verwendete eBook-Reader die StyleSheets korrekt unterstützt, lassen sich verschiedene Arten von Links an ihrer Textfarbe erkennen: Rote gehen „[nach draußen](#)“ (öffnen also wahrscheinlich einen Web-Browser), während grüne auf [Begriffserklärungen](#) im



IzzyOnDroid

5. <https://de.wikipedia.org/wiki/Fibel>

6. <https://android.izzysoft.de/applists>

7. <https://android.stackexchange.com/>

Anhang verweisen, und auch blaue „drinnen bleiben“ (also Querverweise innerhalb dieses eBooks sind). Des Weiteren sind Ausführungen, die root voraussetzen, entsprechend farblich hinterlegt.

Vorwort zur Version 61

Ganze sechzig Versionen lang ist dieses eBook nun ständig gewachsen, stets kam Neues hinzu. Nach vier Jahren wird es daher Zeit für einen „Hausputz“: Kapitel wurden gestrafft, einige Überflüssiges entfernt, und auch ein wenig reorganisiert sowie neu strukturiert. Viele werden dies begrüßen, anderen mag es weniger zusprechen. Wer die umfangreicheren, ursprünglichen Kapitel vermisst, findet sie jetzt im „[Bonus-Material auf meiner Website](#)⁸.

Bedanken möchte ich mich bei all denen, die dieses Buch in den letzten vier Jahren begleitet und unterstützt haben: Durch Korrekturlesen (Hallo Sabine!), Kauf [meiner Bücher bei Amazon](#) (oder auch anderer Dinge nach Nutzung dieses Amazon-Partner-Links), hilfreiche Kritiken und Hinweise, sowie durch Bekanntmachen und Verlinken auf den verschiedensten Webseiten. Euer Engagement macht mir Mut und zeigt mir, dass meine Zeit hier sinnvoll investiert ist – bitte weiter so, und mehr!

Doch nun: Viel Spaß bei der Lektüre!



IzyOnDroid: Das
inoffizielle Android-
Handbuch

8. <https://android.izzysoft.de/books.php?topic=anwender>

GRUNDLAGEN FÜR DEN EINSTEIGER

Im ersten Teil dieses kleinen Handbuchs geht es um die Grundlagen. Fortgeschrittenere Anwender können diesen also getrost überspringen – und gleich zum zweiten oder gar dritten Teil schreiten.

Die erste Inbetriebnahme und Grund-Einrichtung des Androiden erlaube ich mir an dieser Stelle zu überspringen: Zum Einen unterscheiden sie sich je nach Hersteller ein wenig, zum Anderen liegt dem Gerät zumindest dafür in der Regel eine Kurzanleitung bei. Wer dennoch Starthilfe benötigt, findet sie z. B. in einem [Workshop bei Chip.DE](#)⁹.

Wie soll Steve Jobs am Ende seiner Vorstellung des ersten iPhone gesagt haben: „Ach ja, telefonieren kann man damit auch.“ Natürlich sind wir mit Android im „ganz anderen Lager“ (laut Stevie in der Schmuddel-Ecke – aber wir wissen es natürlich besser). Dennoch gehe ich hier ähnlich vor, und klammere das Telefonieren zunächst aus. Zuerst einmal geht es um Grundsätzliches: Wie bedient man ein Android-Gerät? Und was sind eigentlich die einzelnen „Bedien-Elemente“?



Workshop bei
Chip.DE

9. http://www.chip.de/artikel/Android-Guide-Erste-Schritte-mit-dem-Google-Handy_43791328.html

Grundlegendes zur Bedienung des Androiden

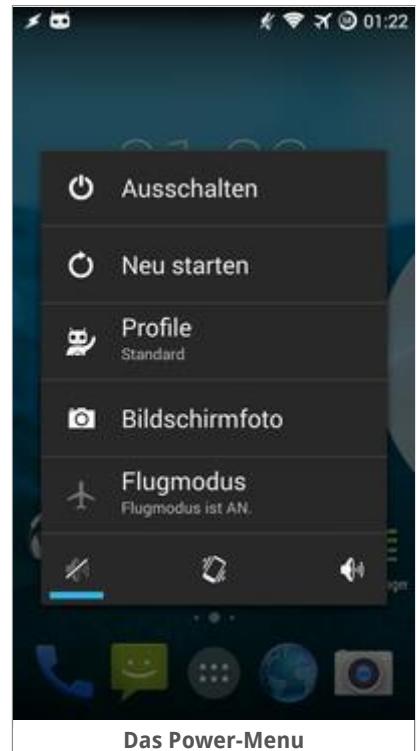
Knöpfe

Auch wenn ein Androide überwiegend über den [TouchScreen](#) bedient wird, gibt es da doch noch ein paar Knöpfe, die sich drücken lassen. Was so ein richtiger Power-Riegel ist, der verfügt auch über einen gleichnamigen Knopf. Kein Gerät kommt ohne diesen. Und was lässt sich damit nun so besonderes Anstellen, dass er an dieser Stelle extra erwähnt werden muss?

Zunächst das triviale: Das Gerät lässt sich damit anschalten. War es zuvor komplett ausgeschaltet, muss der Power-Knopf dafür ein wenig länger gedrückt werden. Anders sieht es aus, wenn nur das Display ausgegangen ist (das tut es, um Strom zu sparen) – dann genügt ein kurzes Antippen. Das Gleiche noch einmal, und der Bildschirm geht wieder aus. Noch immer trivial. Allerdings wird der Bildschirm dabei auch gleich gesperrt – sodass man ihn bei erneutem Anschalten zunächst auch wieder entsperren muss. Das verhindert zum einen unbeabsichtigte Bedienung in der Hosentasche – kann aber, sofern die Sperre mit einem PIN, Muster oder Kennwort-Schutz versehen wurde, auch vor unbefugtem Zugriff schützen.

Interessanter wird es, drückt man diesen Knopf bei aktivem Display ein wenig länger – denn dann kommt plötzlich ein Menü zum Vorschein. Je nach Android-Version lassen sich hier verschiedene Dinge auswählen: Gerät Herunterfahren / Neustarten sind fast generell dabei. Spannende Dinge gibt es jedoch gelegentlich auch: Spätestens ab Android 4.0 lässt sich bei den meisten Geräten über dieses Menü auch ein Bildschirm-Foto auslösen. Und manche Geräte bieten an dieser Stelle auch einen schnellen „Profil-Wechsel“ an – etwa eben einmal auf lautlos stellen, oder in den Flugzeug-Modus wechseln ...

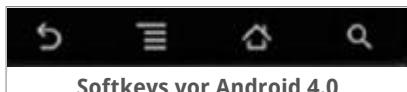
Doch ein Android-Gerät bietet noch weitere „Knöpfe“. Die zur Lautstärke-Regelung seien nur kurz erwähnt, und auch zum Auslösen der eingebauten Kamera ist gelegentlich ein Knopf reserviert. Und dann sind da noch drei bis vier weitere, die meist nicht ganz so offensichtlich sind: Auf neueren Androiden handelt es sich hier nämlich nicht um „Hardware-Knöpfe“, sondern um so genannte „Soft Keys“, die meist bei eingeschaltetem Display auch beleuchtet (und bei ausgeschaltetem Display ohne Funktion) sind.



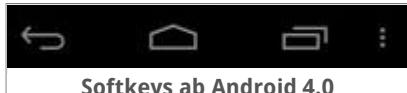
Das Power-Menu

Symbole sollen diese Knöpfe intuitiv bedienbar machen. In den meisten Fällen ganz rechts außen findet sich eine Lupe – zwar ohne Hut, aber der Detektiv steht anbei: Hiermit steht vielerorts eine Suchfunktion zur Verfügung. Dann ist da ein Haus: Dies ist der so genannte „Home-Key“, der von überall sofort auf den „Home-Screen“ führt. Die gerade genutzte Anwendung wird dabei nicht beendet, sondern wartet im Hintergrund. Und damit muss auch die zweite Belegung dieser Taste sofort erwähnt werden: Ein langes Drücken öffnet eine Liste der zuletzt gestarteten Apps, sodass man wieder zur wartenden App zurückgelangen kann.

Weiterhin wäre da noch der „gebogene Pfeil“, der fast schon „Bitte wenden!“ zu rufen scheint. In Menüstrukturen hat er die Funktion „Zurück“, was auch bei vielen Apps gilt: So letztere nicht explizit einen Knopf zum Beenden bieten, soll diese Taste das erledigen. Gelegentlich hilft ein langer Druck hier, eine App auch wirklich zu beenden – doch in der Regel ist so etwas speziellen [Custom-ROMs](#) vorbehalten.



Einen haben wir noch – einen Knopf, meine ich. Mal eine Liste, mal mit vier Quadranten, von denen eines ausgemalt ist. Nein, das ist nicht der Knopf, um schnell Yatzee (oder ein anderes Würfel-Spiel) zu starten – sondern der Menü-Knopf (so vorhanden – denn ab Android 4.0, und mit Einführung des Holo-Designs, verliert er langsam seine Bedeutung). Bei vielen Apps (und auch auf dem HomeScreen) lassen sich damit Zusatz-Funktionen aufrufen.



Ab Android 4.0 haben sich die Softkeys ein wenig verändert. In aktuellen Geräten sind sie nun nicht mehr fest integriert, sondern werden dynamisch vom System behandelt: Steht

beispielsweise keine Menü-Funktion zur Verfügung, wird die „Menü-Taste“ auch gar nicht angezeigt. Auch die Funktionalität hat sich im Vergleich zu früheren Versionen leicht geändert:

- Die „Zurück-Taste“ ist geblieben, und funktioniert wie gehabt.
- Auch die Taste mit dem Haus führt nach wie vor zum Homescreen. Bei langem Drücken öffnet sich jedoch nicht mehr die Liste zuletzt geöffneter Apps – stattdessen poppt ein „Google“ Kreis auf, über den man zu „Google Now“ gelangt.
- Neu ist die Taste mit den zwei Rechtecken, die man „Multi-Tasking-Taste“ benennen könnte: Hierüber öffnet man nun die Liste der zuletzt genutzten Apps. Unerwünschte Kandidaten lassen sich mit einer Wisch-Bewegung aus selbiger entfernen (wobei das System sie höflich bittet, sich doch gleich ganz zu beenden).
- Die „Menü-Taste“ ist nun ein „senkrechter Strich“, und (wie beschrieben) nur sichtbar, wenn auch Menü-Funktionen zur Verfügung stehen.



Unter Android 5 hat sich das Aussehen der Softkeys nochmals verändert: Jetzt steht ein einfaches Dreieck für die „Zurück-Aktion“, ein Kreis kennzeichnet die „Home-Taste“, und ein einfaches Quadrat führt zur Liste der zuletzt genutzten Apps.

Der TouchScreen

Android-Geräte werden i. d. R. Über einen Touchscreen bedient – nur wenige bieten zusätzlich eine Hardware-Tastatur. Während es noch offensichtlich ist, dass sich eine App durch einfaches Antippen des zugehörigen Icons starten lässt,

sind viele Interaktionen für den Anfänger ein wenig „versteckt“. Da wären zum einen die Menüs, die sich – sofern vorhanden – über die Menütaste aktivieren lassen. Und oftmals fördert ein „langes Drücken“ ein Kontext-Menü zutage.

In vielen Apps finden zusätzlich Wischgesten Verwendung: So gelangt man etwa durch waagerechtes Wischen zu weiteren Bildschirmen (bei einer eBook-Lese-App etwa zur vorigen bzw. nächsten Seite), oder kann durch senkrechtes Wischen entlang der linken Bildschirmkante die Helligkeit des Displays regeln. Beliebt sind auch Zwei-Finger-Gesten, wie etwa das sogenannte „Pinch-to-Zoom“: Hierbei berührt man das Display mit zwei Fingern, und zieht diese auseinander – um etwa in ein Bild hinein zu zoomen. Umgekehrt verkleinert man das Ganze wieder, indem man die Finger aufeinander zu bewegt. Das klappt nicht nur beim Betrachten von Bildern in der Galerie, sondern beispielsweise auch in den meisten Webbrowsern.

Der Sperrbildschirm

Wie bereits erwähnt, schaltet man mit dem Power-Knopf den Bildschirm an. Um ein versehentliches Bedienen in der Hosentasche zu vermeiden, wird an dieser Stelle ein Sperrbildschirm (auch als „Lock-Screen“ bezeichnet) aktiv. Je nach Android-Version sieht dieser unterschiedlich aus; gemein ist jedoch allen Versionen, dass er sich mit einer Wisch-Bewegung entriegeln lässt. Oftmals verbergen sich hier auch Zusatzfunktionen – so lassen sich gleichzeitig mit dem Entriegeln etwa auch noch Aktionen ausführen. Die rechte Abbildung zeigt einen Lock-Screen unter Android 4.4 (Kitkat): Ausgehend vom Zentrum entsperrt man das Gerät, indem man den Finger auf eines der Symbole zieht. Bewegt man ihn zum Schloss, wird das Gerät lediglich entsperrt; bewegt man ihn auf eines der anderen Symbole, wird gleichzeitig die zugehörige App gestartet. Eine sinnvolle Vorbelegung der einzelnen Icons ist bereits konfiguriert – kann vom Anwender jedoch unter *Einstellungen > Bildschirmsperre > Sperrbildschirm-Anwendungen* angepasst werden.





Sicherheit gegen unbefugte Bedienung bietet das jedoch noch nicht: Lässt man das Gerät etwa auf dem Kneipentisch liegen, während man auf die Toilette geht, haben die Freunde (oder auch andere Kneipen-Besucher) mit dieser Art von Sperrbildschirm leichtes Spiel – und könnten nicht nur problemlos auf die Inhalte zugreifen, sondern auch teure Anrufe tätigen oder gar Schadsoftware installieren. Doch dagegen lässt sich etwas unternehmen, indem man einen Sperr-Code einrichtet. Dies erledigt man in den System-Einstellungen unter *Bildschirmsperre* → *Bildschirmsicherheit* → *Display-Sperre*. Standardmäßig ist keine Passcode-geschützte Sperre aktiviert – das wäre ja auch fatal, denn woher sollte der neue Anwender den Sperr-Code kennen?

Seit der ersten Android-Version mit dabei, freut sich das so genannte „Sperr-Muster“ (auch als „Pattern-Lock“ bekannt) großer Beliebtheit. Es ist vom Prinzip her auch wesentlich sicherer als der altbekannte PIN-Code (bei dem viele Anwender entweder nur „1234“ oder das Geburtsdatum verwendeten – was sich mit ein wenig „Social Engineering“ schnell erraten lässt). Hier muss ein Muster gezeichnet werden, welches mindestens vier Punkte verbindet (siehe linke Abbildung). Da hilft das beste Social-Engineering nicht weiter, da ein Bezug zur Person höchst unwahrscheinlich ist. Dennoch ließe sich das Muster u. U. [auf dem Bildschirm erkennen](#).

Eine andere (und noch sicherere) Möglichkeit ist die Vergabe eines Passwortes – sofern hier nicht wieder obiges „1234“ verwendet wird. Ein sicheres Passwort besteht aus einer Kombination von Buchstaben und Ziffern (sowie ggf. Sonderzeichen), die sich nicht in einem Wörterbuch finden lässt. Wie man sich so etwas merken soll? Ganz einfach, beispielsweise mit einem Merksatz. Nehmen wir als Beispiel den Satz „Ich habe ein sicheres Passwort“. Und nun von jedem Wort den ersten Buchstaben: „IhesP“ - schaut doch schon recht kryptisch aus! Noch eine Ziffer eingebaut: „ein = 1“ ergibt sodann: „Ih1sP“. Steht in keinem Wörterbuch – und lässt sich (Dank des Satzes) dennoch einfach merken. Das rechte Bild zeigt, wie das dann aussehen könnte.

Eine kleine Unbequemlichkeit ergibt sich damit natürlich: Es dauert ein paar Sekunden mehr, bis man die nunmehr zwei Sperrbildschirme überwunden hat, und wieder mit dem Gerät arbeiten kann.



Die Benachrichtigungsleiste

Auch als *Notification Bar*, *Status Bar*, oder *Statusleiste* bekannt, ist diese auf Tablets und Smartphones leicht unterschiedlich umgesetzt: Findet sie sich bei ersteren i. d. R. in der unteren rechten Ecke, ist sie bei Letzteren am oberen Bildschirmrand untergebracht. Den Aufbau erkläre ich hier für Smartphones:



Grob lässt sich die Leiste in zwei Bereiche unterteilen. Auf der rechten Seite finden sich Status-Informationen wie Uhrzeit, Ladezustand des Akkus, oder die Empfangsstärke des Mobilfunk- bzw. WLAN-Signals. Dieser Bereich wird vom System verwaltet; Apps können hier normalerweise keine Informationen unterbringen. Für sie ist die



linke „Hälfte“ reserviert, in der sich etwa anstehende Termine oder Wecker, aber auch Hinweise auf neue Nachrichten oder entgangene Anrufe finden.

Während sich die meisten Status- und Benachrichtigungs-Icons relativ leicht zuordnen lassen, sind andere nicht ganz so intuitiv interpretierbar. Eine vollständige Liste gibt es leider nicht einmal für die vom System verwalteten Status-Icons – da abgesehen von unterschiedlichen Android-Versionen auch hier wieder viele Hersteller ihr eigenes Süppchen kochen. Einige umfangreichere Auflistungen, die sich zumindest größtenteils auf andere Geräte übertragen lassen sollten, finden sich beispielsweise:

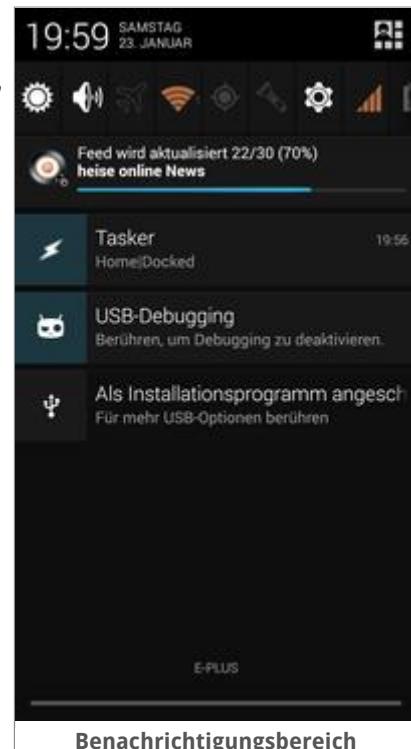
- Bei *BeginAndroid* [am Beispiel von Gingerbread](#)
- Bei *CellphoneForums* [am Beispiel des LG G2](#) (Android 4.x)
- Im *GalaxyS5Manual* [für das Samsung Galaxy S5](#) (Android 4.x)

Der Benachrichtigungsbereich

Zumindest für die Benachrichtigungen gibt es zusätzliche Details, wenn man den Benachrichtigungsbereich (auch *Notification Area* genannt) öffnet. Auf dem Tablet tippt man die „Leiste“ dafür an, während man sie auf dem Smartphone mit einem Finger „herunterzieht“. Wie im nebenstehenden Screenshot zu sehen, gibt es zu den Icons der Benachrichtigungsleiste hier nähere Informationen – und wir erfahren, dass ...

- Der „Blitz“ für die App „Tasker“ steht, bei dem gerade die Profile „Home“ und „Docked“ aktiv sind
- Der „Alien-Kopf“ für aktives „USB-Debugging“ steht, und das Gerät per USB angeschlossen ist
- Die RSS-Feeds gerade aktualisiert werden (und wie es um den Fortschritt dieser Aktion bestellt ist)

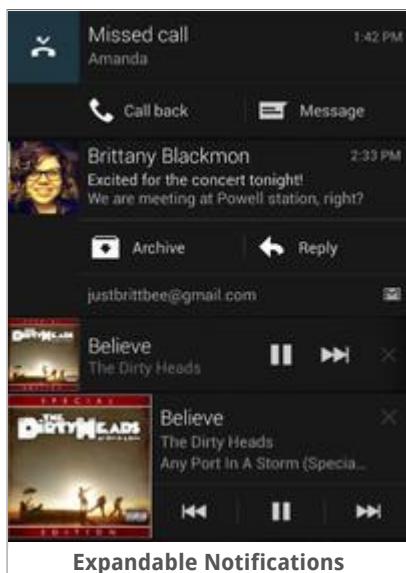
Andere Details können etwa die Anzahl verpasster Anrufe, wartender SMS, oder neu eingegangener Mails kundtun. Ebenso möglich sind Hinweise auf fehlgeschlagene Hintergrund-Aktivitäten (beispielsweise der automatischen Synchronisation von Dateien), oder auf anstehende Termine. Je nachdem, welche Apps man nutzt, kommen weitere Möglichkeiten hinzu. Hat



man die Benachrichtigung zur Kenntnis genommen, kann man sie entweder durch „Herauswischen“ entfernen – oder öffnet durch „Antippen“ die zugehörige App, was im Regelfall dort auch gleich zum passenden Bildschirm führt. Ist einmal nicht klar, wer eine Benachrichtigung zu verantworten hat, hilft bei aktuellen Android-Versionen ein langes Drücken auf selbige: Das öffnet die Einstellungen der zugehörigen App, in denen man u. a. auch [der Anwendung weitere Benachrichtigungen verbieten kann](#).

Des Weiteren finden sich, zumindest in aktuellen Android-Versionen, noch nützliche Schnellzugriffe („Shortcuts“) im Benachrichtigungsbereich, mit deren Hilfe sich schnell das WLAN oder GPS (de-)aktivieren, die Bildschirm-Helligkeit regulieren, oder in den Flugzeug-Modus schalten lässt. Je nach Gerät und Android-Version unterscheiden sich die an dieser Stelle verfügbaren Schalter.

Expandable Notifications



Wer glaubt, das wäre schon alles – der irrt: Mit [Jelly Bean](#) legte Android noch einmal „eins drauf“, und führte die so genannten „expandable notifications“ (etwa: „erweiterbare Benachrichtigungen“ – auch als „rich notifications“, also „reichhaltige Benachrichtigungen“ bekannt) ein. Um zu verdeutlichen, worum es sich dabei handelt, habe ich statt eines einfachen Screenshots einmal eine „Collage“ eingefügt.

Kurz gesagt, lassen sich einige Benachrichtigungen erweitern – indem man sie auf ähnliche Weise öffnet, wie den Benachrichtigungsbereich selbst: Durch „Aufziehen“. Je nach Gerät und Android-Version geschieht dies entweder mit einem oder mit zwei Fingern, jeweils von oben nach unten

gezogen. Streicht man dabei über den gesamten Benachrichtigungsbereich, werden alle „erweiterbaren Benachrichtigungen“ erweitert – und wie in der Collage zu sehen, stehen damit zusätzliche Möglichkeiten der Interaktion zur Verfügung. So können etwa Nachrichten direkt beantwortet (bzw. archiviert) – oder der Medioplayer bedient werden, ohne dass man die eigentliche App öffnet.

Erweiterte Benachrichtigungen lassen sich auch wieder schließen. Beim „Zwei-Finger-System“ streicht man dazu einfach mit beiden Fingern nach oben. Das

„Ein-Finger-System“ scheint dies zunächst nicht zu unterstützen, doch hier hilft ein kleiner Trick: Ohne den Finger abzusetzen, zunächst kurz nach unten und dann gleich nach oben streichen, schließt auch hier die „erweiterte Ansicht“.

Leider werden „expandable notifications“ auch zwei Jahre nach ihrer Einführung von viel zu wenigen Apps wirklich genutzt. Wer sich zusätzlich darüber ärgert, dass die vorhandenen Exemplare nicht automatisch im „erweiterten Zustand“ angezeigt werden – der hat, ein gerootetes Gerät vorausgesetzt, mit dem *Xposed* Modul [All Notifications Expanded](#)¹⁰ die Möglichkeit, dem Abhilfe zu schaffen.

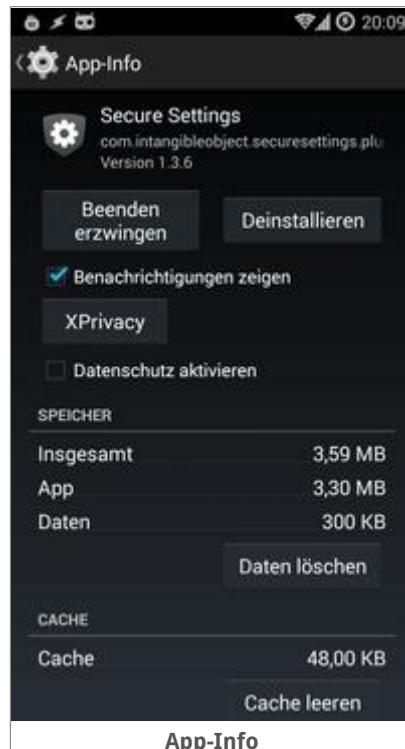


XDA: View Expanded Notifications by Default with Xposed

Benachrichtigungen für einzelne Apps verbieten

Eine weitere mit *Jelly Bean* eingeführte Neuerung ist, dass man nun für jede App einzeln festlegen kann, ob man von dieser Benachrichtigungen empfangen möchte. Nicht immer ist klar, wer einem da dieses „Ei ins Nest gelegt“ hat – daher ist einer der Zugangspunkte zu dieser Funktion auch direkt im Benachrichtigungsbereich zu finden: Drückt man länger auf eine Benachrichtigung, lassen sich die Einstellungen der zugehörigen App öffnen. Entfernt man hier das Häkchen bei „Benachrichtigungen anzeigen“, sollte sich diese App im Benachrichtigungsbereich nicht mehr blicken lassen.

Auf diese Weise kann man sich auch von „unhöflichen Apps“ befreien, welche Benachrichtigungen für Werbung missbrauchen: Gerade in solchen Fällen ist der Übeltäter auf andere Weise häufig kaum zu identifizieren. Neben der Möglichkeit, ihm hier derartige Anzeigen zu untersagen, lässt sich selbiger durch Betätigen des mit „Deinstallieren“ beschrifteten Buttons auch gleich ganz aus dem System entfernen. Schließlich kann man schwer sagen, welche weiteren „unerwünschten Tätigkeiten“ er sonst noch auf Lager hat.



10. <http://www.xda-developers.com/android/view-expanded-notifications-by-default-with-xposed/>

Die „App-Infos“ sind übrigens auch über das System-Menü [Anwendungen verwalten](#) zugänglich, sollte gerade einmal keine Benachrichtigung zur Hand sein.

Benachrichtigungsverlauf

Unter [Benachrichtigungsbereich](#) habe ich ja bereits darauf hingewiesen, dass man Benachrichtigungen durch Wischen aus selbiger entfernen kann. Was wenige wissen: Es gibt auch eine Möglichkeit, sich den Verlauf später noch einmal anzuschauen. Wie das ab Android 4.3 geht, beschreibt [ein Artikel bei Stack Exchange](#)¹¹:

1. Den [App Drawer](#) öffnen, und auf den Reiter *Widgets* wechseln
2. Die Liste nach dem Widget für *Einstellungsverknüpfung* durchsuchen. Dieses dann (wie unter *App Drawer* beschrieben) auf den Homescreen befördern
3. Es öffnet sich nun eine lange Liste mit Einstellungen, aus der man *Benachrichtigungen* auswählt.

Nun befindet sich ein [Shortcut](#) (Schnellzugriff) zur passenden Seite auf dem Homescreen. Tippt man diesen an, werden die letzten Benachrichtigungen aufgelistet.

Android Benachrichtigungen

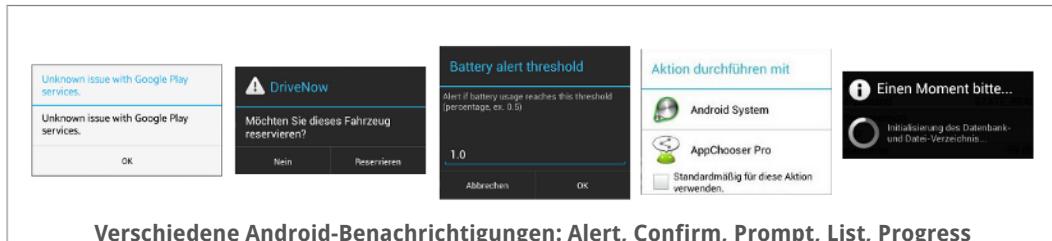
Die im vorigen Kapitel behandelten „Notifications“ sind sicher die bekannteste Art von System-Benachrichtigungen unter Android – jedoch bei Weitem nicht die einzigen. Da wären beispielsweise noch die „Toasts“: Hinweise, die kurz eingeblendet werden, um sogleich wieder von selbst zu verschwinden. Im Regelfall sind diese dafür gedacht, kurzes Feedback zu geben. Daher tauchen „Toasts“ für gewöhnlich auf, wenn der Benutzer manuell eine Aktion angestoßen hat: Einen Kalendereintrag erstellt, einen Wecker aktiviert, oder einen Download gestartet. Natürlich beschränkt sich das nicht auf diese drei Beispiele.



Als Drittes wären noch die verschiedenen Dialogboxen zu nennen. Da gäbe es Warnungen („Alerts“), die einfach so lange stehen bleiben, bis man den einzigen verfügbaren Button betätigt hat. Anders als ein „Toast“ schließt sich so ein Dialog

11. <http://android.stackexchange.com/q/50177/16575>

nicht automatisch von selbst – um sicher zu gehen, dass der Anwender die Nachricht auch zur Kenntnis genommen hat. Bei Bestätigungen („Confirm“) versteht sich das von selbst, schließlich soll hier eine Auswahl getroffen werden. Anders als in der Abbildung, kann ein solches „Confirm“ durchaus über mehr als zwei Buttons verfügen (etwa „Ja“, „Nein, nie“, und „Jetzt nicht – frag später nochmal“).



Verschiedene Android-Benachrichtigungen: Alert, Confirm, Prompt, List, Progress

Auch ein „Prompt“ gehört in diese Kategorie. Ein solcher fordert den Anwender zu einer Text-Eingabe auf: Ein Wert, Name, Email-Adresse, o. ä. Schließlich gibt es auch noch die „List“-Boxen, bei denen der Anwender eine Auswahl treffen soll. Und natürlich diverse „Sand-Uhren“ („Progress“) zur Fortschrittsanzeige bei einer länger währenden Aktion – entweder mit einem Kreisel (bei unbekanntem Fortschritt), oder aber auch mit Fortschritts-Balken.

Warum ich dies hier alles auseinandernehme? Bei Problemen ist es hilfreich, dem Helfer/Entwickler möglichst genaue Details nennen zu können. Beschreibt man ein Problem mit „dann erscheint kurz ein Toast ...“, ist der Sachverhalt wesentlich klarer als bei „dann kommt da so ein Text ...“

Google Konto

Bei der ersten Inbetriebnahme weist ein Android-Gerät seinen Nutzer i. d. R. darauf hin, dass er doch bitte ein Google-Konto einrichten möge (siehe auch [Google Account](#) unter *Fragen aus Alltag und Praxis*). Für die „echte Google-Experience“ sei so etwas schließlich unabdingbar. Zwar funktioniert ein Android-Gerät auch ohne einen solchen Account – allerdings lässt er dann einige Bequemlichkeiten vermissen, welche Google in Android integriert hat:

- Das [Google Cloud Backup](#)
- Die Nutzung des [Google Playstore](#), u. a. zur Installation von Apps
- Die [Synchronisation von Kontakten und Kalendern](#)
- Die volumnfängliche Nutzung von weiteren Google-Diensten, wie etwa [Google Now](#), GMail, u. a. m.

Während dies für den Einen genau das ist, was Android ausmacht, läuten bei Anderen an selbiger Stelle die Alarmglocken: Wie ist es dabei um meine [Privatsphäre](#) bestellt? (Anmerkung: Auch oder gerade wer meint, ja „[nichts zu verbergen](#)“ zu haben, sollte vor einer endgültigen Entscheidung das entsprechende Kapitel zumindest kurz „überfliegen“) Bedenken sind u. a. auch deshalb nicht ganz unberechtigt, da sich hier eine hohe Konzentration privater Daten an einer Stelle bildet. Gelingt es einem „Bösewicht“, sich Zugang zu verschaffen, besteht auch die Gefahr eines Identitäts-Diebstahls. Einige zentrale Punkte sollen daher im Folgenden kurz angeschnitten werden.

Einstellungen mit Auswirkung auf die Privatsphäre

Ganz auf einen Google-Account verzichten möchten sicher die Wenigsten, schränkt dies doch insbesondere die Auswahl verfügbarer Apps enorm ein – der [Google Playstore](#) ist nun einmal die umfangreichste App-Quelle. Doch auch mit eingerichtetem Account lassen sich Einstellungen zum Schutz der Privatsphäre treffen:

- **Datensicherung:** Gleich die erste Frage bei der Account-Aktivierung lautet sinngemäß: „Möchten Sie Ihre Daten bei Google sichern?“ Was sich dahinter verbirgt, ist genauer im Kapitel [Google Cloud Backup](#) beschrieben. Kurzgefasst: Nur wenige Apps unterstützen dies. Dafür gelangen aber auch sensible Informationen wie etwa konfigurierte WLAN-Netze einschließlich ihrer Klartext-Passworte auf die Google-

Server. Da überdies die Wiederherstellung der so gesicherten Daten nicht unbedingt immer funktioniert, stellt sich zumindest die Frage, ob der Nutzen das Risiko wert ist.

Mit Android 6.0 soll sich das ändern (Apps müssen dann nicht mehr speziell angepasst sein, um „mitzuspielen“) – doch das muss sich zunächst einmal in der Praxis zeigen und bewähren.

- **Synchronisation:** Selbst wenn die vorige Frage mit „Nein“ beantwortet wurde, werden standardmäßig zahlreiche Daten mit den Google-Servern synchronisiert – ohne den Nutzer davon in Kenntnis zu setzen. Dazu gehören etwa Kalendereinträge und Kontakte. Wer dies nicht wünscht, sollte zum einen die entsprechenden Optionen deaktivieren (zu finden unter *Einstellungen > Konten & Synchronisation*, siehe auch [Kontakte und Kalender](#)) – zum anderen aber auch in Erwägung ziehen, in der Kontakte- sowie der Kalender-App den (lokalen) Geräte-Speicher statt des Google-Accounts als Vorgabe für neue Einträge festzulegen.
- **Standort-Dienste:** Natürlich ist es praktisch, wenn das Gerät schnell weiß, wo man ist. Ortsbezogene Daten erweisen sich an vielen Stellen als nützlich („Geschäfte in der Nähe“ sind da nur ein Beispiel). Allerdings lassen sich so u. a. auch prima Bewegungs-Profiles erstellen. Wer das lieber vermeiden möchte, findet Details zum Abstellen im Kapitel [Ortsdaten](#).

Apps und Privatsphäre

Immer mehr Dienste werden in Google-Apps integriert. Google ist nun einmal in erster Linie eine Firma, die ihr Geld mit Werbung verdient – und daher daran interessiert, möglichst viel über uns in Erfahrung zu bringen. Denn so lässt sich Werbung am Besten verkaufen. Werfen wir also einmal einen Blick auf diverse „Integrationen“ sowie deren Alternativen:

- **SMS/MMS:** Ab Android 4.4 wurden diese in *Google Hangouts* integriert. Wem damit nicht wohl ist, der installiert sich besser eine alternative SMS-App (siehe [Nachrichten verschicken und empfangen](#)), die sich dann als Standard festlegen lässt. Wer Hangouts anderweitig nutzt, kann dessen SMS-Handling zur Sicherheit auch unter *Einstellungen > SMS* in der App selbst deaktivieren.

- **Galerie:** Beginnend mit Android 4.4, wird auch diese allmählich in Google+ integriert. Wer hier rechtzeitig vorbereitet sein will, hält schon einmal Ausschau nach einer alternativen Galerie-App¹².
- **Google+:** Diese App wird in letzter Zeit mehr und mehr zum Zwang. So lassen sich ohne einen „G+ Account“ im *Playstore* keine Apps mehr bewerten, die Galerie wird nach und nach ebenfalls hier integriert (siehe voriger Punkt), und einiges mehr. Wer sich diesem Zwang nicht entziehen kann (oder will), sollte jedoch zumindest erwägen, von gewissen Rechten Gebrauch zu machen – und der Nutzung seiner persönlichen Daten zu Werbezwecken zu widersprechen¹³.
- **Google Now:** Hier kann man sich einmal so richtig vor Augen führen, wie sich die von Google gesammelten Daten nutzen lassen – wie Sabine mit Erschrecken feststellen musste. Zugegeben, die Funktionalität ist sehr verlockend.



Übersicht:
Bildbetrachter und
Fotogalerien



TechCrunch: How To
Opt Out Of Google's
Weird New Ads That
Use Your Face And
Name

12. http://android.izzysoft.de/applists/category/named/foto_galleries

13. <http://techcrunch.com/2013/10/12/opt-out-google-ads/>

Schaltzentrale: Home-Screen, Widgets & „Home Replacements“

Wenn es bei Android so etwas wie eine „Schaltzentrale“ gibt, ist dies sicher am ehesten der **Homescreen**: Hier starten alle Aktivitäten. Das ist es, was der Anwender nach dem Start seines Androiden zu sehen bekommt – von hier startet er seine Apps – hier platziert er (so er dies tut) seine Übersichten wie aktuelle Kalender-Ereignisse, News-Feeds, und so weiter. Daher macht es durchaus Sinn, dass sich das erste Kapitel dieses Abschnittes zunächst diesem widmet.



Holo Launcher und Apex Launcher als Alternativen zum vorinstallierten Homescreen

Eigentlich sollte ich besser schreiben: „diesen“. Klar gibt es einen „Standard-Launcher“ bzw. „Stock-Launcher“ („Launcher“ ist ein anderes Wort für den Homescreen, welches obigen Sachverhalt betont: Dass man von hier alle Aktivitäten „lässt“, also startet). Auf fast allen Geräten ist jedoch bereits eine Alternative installiert: Da wäre HTC mit ihrem Sense Launcher, Motorola mit der



Holo Launcher



Apex Launcher



Übersicht: Home-Replacements

MotoBlur Oberfläche, etc. pp.. Und zahlreiche Alternativen sind im Play Store verfügbar – wie etwa der [Holo Launcher¹⁴](#) (linkes Bild), oder [Apex Launcher¹⁵](#) (rechtes Bild). Jeder hat so seine Besonderheiten und Vorteile gegenüber den anderen. Da wären auf's Ressourcen-Schonen getrimmte Launcher, minimalistische Launcher (sowie deren Gegenstücke) – und, und, und. Ein genauerer Überblick findet sich natürlich wieder in der passenden [Übersicht¹⁶](#).

Docking Bar

Das ist i. d. R. der Bereich „unten“, in dem besonders häufig genutzte Funktionen verankert sind (auf obigen Screenshots auch gut zu erkennen). Bei einigen *Launchern* sind diese Aktionen „fest verdrahtet“, und lassen sich nicht ändern/anpassen. Die Auswahl der Aktionen ist dabei für die Masse durchaus tauglich: Telefon ist immer dabei (das Gerät heißt ja auch „SmartPhone“, und nicht „MiniComputer“ – auch wenn die Grenzen da schwer zu definieren sind), dazu kommen meist Anrufliste und Kurznachrichten, sowie der *App-Drawer*.

Die meisten (mir bekannten) *Launcher* erlauben es jedoch zumindest, die Aktionen selbst auszuwählen. So lassen sich entsprechende „Icons“ z. B. bei o. g. *Apex Launcher* per Drag-and-Drop platzieren (und entfernen), auch die Reihenfolge lässt sich nachträglich ändern. Einige gehen sogar noch weiter, und lassen den Benutzer an die grafische Ausgestaltung direkt heran. Wer also alles individuell gestalten möchte, kann dies durchaus tun!

App-Icons

Diese lassen sich in der Regel auf dem Launcher (s. o.), und generell auf den HomeScreens platzieren. Letzteres gilt auch für die *Shortcuts* und *Widgets* (siehe unten). Für alle drei ist das Standard-Vorgehen zur Platzierung, eine freie Stelle auf dem „Desktop“ „lange zu drücken“. Daraufhin öffnet sich ein Kontext-Menü und fragt nach, was es denn sein darf – wobei unsere drei Kandidaten, und ggf. (je nach Launcher) auch noch weitere Dinge zur Auswahl stehen können. Spätestens ab [Ice Cream Sandwich](#) gibt es auch die Möglichkeit, App-Icons und Widgets direkt aus dem App-Drawer heraus auf den gewünschten Homescreen zu ziehen. Wieder entfernen lassen sie sich ebenfalls durch „langes Drücken“ (diesmal auf das Icon selbst) und anschließendes „Ziehen“ auf die sich öffnende (meist rote) Mülltonne.

14. <https://play.google.com/store/apps/details?id=com.mobint.hololauncher>

15. <https://play.google.com/store/apps/details?id=com.anddoes.launcher>

16. http://android.izysoft.de/applists/category/named/tools_launcher

Unsere *App-Icons* haben nun keine weitere Funktion, als die zugehörige App zu öffnen. Nicht viel, aber mehr braucht es ja oft auch nicht: Von zentraler Stelle die wichtigsten Dinge schnell starten, ohne sich erst durch den „Drawer“ (die komplette Applikationsliste) wühlen zu müssen. Benötigt man doch einmal etwas Spezielleres, kommen unsere anderen beiden Kandidaten zum Einsatz:

Shortcuts

Nomen est Omen, wie der Latiner sagt: Hier geht es um „Abkürzungen“, die einige Apps anbieten. Was auf dem HomeScreen wie ein gewöhnliches (gerade eben beschriebenes) *App-Icon* aussieht, ist es auch – nur mit ein wenig Zusatz-Funktionalität. Es springt bei der zugehörigen App gleich zu einem bestimmten Bildschirm, oder löst eine bestimmte Aktion aus. Ein „klassisches Beispiel“ wäre bei [Note Everything¹⁷](#) zu finden: Die Startseite (mit den Übersichten) überspringen, und direkt eine neue Notiz öffnen. Oder bei den weiter unten unter [Apps Organisieren](#) genannten „Organizern“ das Öffnen eines bestimmten Ordners (was die beiden o. g. Launcher auch selbst anbieten). Bei diesen Dingen handelt es sich um *Shortcuts*.



Note Everything

Dazu muss gesagt werden, dass Shortcuts von den Apps selbst bereitgestellt werden müssen: Was die App nicht anbietet, steht da auch nicht zur Verfügung.

Widgets

Gleiches gilt auch für die Widgets: Grafische Elemente, die erweiterte Informationen zur Verfügung stellen – und optional auch noch als Shortcuts dienen können. Einige Beispiele dafür finden sich im obigen Screenshot des *Apex Launcher*.



Widgets von [DroidStats¹⁸](#), die Informationen zu aktuellen Statistiken (hier: Telefonminuten und SMS) geben – und bei „Antippen“ die App gleich auf der zugehörigen Detail-Seite öffnen.



DroidStats



Widgets von [Mini-Info¹⁹](#), die über diverse System-Informationen auf dem Laufenden halten. Tippt man sie an, wird die App (ganz normal) gestartet.



Mini-Info

17. <https://play.google.com/store/apps/details?id=de.softxperience.android.noteeverything>
 18. <https://play.google.com/store/apps/details?id=nitro.phonestats>

Ein TaskManager-Widget, welches über freien Speicher sowie die Anzahl gerade laufender Prozesse informiert. Die bei Antippen ausgeführte Aktion ist konfigurierbar – etwa das Starten der App, oder Killen aller „black-listed“ Apps. Übrigens: Auch die Uhr im Screenshot am Anfang dieses Kapitels ist ein Widget.



App-Drawer

Auch zu diesem zu guter Letzt noch ein paar Worte. Manch einem mag er wie eine „unübersichtliche Lagerhalle von Icons installierter Apps“ vorkommen. Dem Hörensagen nach muss das nicht generell so sein. Es soll Launcher geben, die hier alternativen Implementierungen folgen, und Dinge wie „Reiter“, „Unter-Ordner“, „Kategorien“, und Ähnliches anbieten. Wer hier also gern ein wenig aufräumen würde, und einem „alternativen Launcher“ nicht abgeneigt ist, sollte bei der Auswahl auch darauf achten. Womit er sich ggf. auch den unter [Apps Organisieren](#) genannten separaten „Organizer“ erspart.

Wo wir gerade vom App-Drawer sprechen: Ab Android 4.0 („Ice Cream Sandwich“) findet sich in diesem ein zusätzlicher Reiter, der verfügbare Widgets auflistet. Somit hat man endlich eine Übersicht darüber, welche Widgets verfügbar sind. Auf den Home-Screen kann man selbige dann befördern, indem man sie ganz dolle drückt: Der App-Drawer blendet sich dann aus, und man lässt das Widget schließlich an der gewünschten Stelle einfach „fallen“. Ab Android 4.4 fällt dieser Reiter jedoch wieder weg; das Hinzufügen von Widgets zum Homescreen erfolgt dann erneut, wie unter [App-Icons](#) beschrieben.

19. <https://play.google.com/store/apps/details?id=com.dynotes.miniinfo>

MIT ANDROID ARBEITEN

Steuerzentrale: Einstellungen und „Switches“

Haben wir den *Home-Screen* als „Schaltzentrale“ bezeichnet – so ist der Ort, an dem die ganzen Systemeinstellungen getätigt werden, ja wohl die „Steuerzentrale“. Und es gibt so einiges einzustellen bei Android, die Liste ist also nicht unbedingt kurz. Hinzu kommt, dass vieles „historisch gewachsen“ ist – und somit manche Dinge an den verschiedensten Orten zu suchen sind, obwohl sie aus subjektiver Sicht eigentlich zusammen gehören ...

Es handelt sich bei aktuellen Android-Versionen schon um recht komplexe Systeme, wo man an vielen Schräubchen drehen müssen muss. Und insbesondere für Neueinsteiger sind das meist zu viele (wobei genau die, die man gerne hätte, natürlich fehlen). Doch gibt es hier auch einige Apps, die für Erleichterung sorgen: Entweder, weil sie die Auswahl auf wesentliche (häufig benutzte) Punkte zusammenstauchen – oder, weil sie in spezifischen Bereichen zusätzliche Einstellungsmöglichkeiten schaffen. Zu beiden Themen finden sich in meiner [Übersicht zu Einstellungen](#)²⁰ zahlreiche Kandidaten.

Konfiguration

Bei Android lässt sich so einiges konfigurieren. Und mit jeder neuen Version kommen neue Dinge hinzu. Ich möchte jetzt nicht auf alles eingehen – doch einige zentrale Einstellungen sollten hier erläutert werden.

Die folgenden Dinge sind alle in den „Einstellungen“ von Android untergebracht. Wie man dorthin gelangt? Vom „[Home-Screen](#)“ ausgegangen, geht es zunächst über die „Menü“-Taste ins Menü, und von dort in den Punkt „Einstellungen“ (alternativ ruft man die App „Einstellungen“ aus dem [App-Drawer](#) auf). Dann geht es entsprechend weiter, wie in den folgenden Abschnitten beschrieben.



Übersicht: System-Einstellungen

20. http://android.izzysoft.de/applists/category/named/tools_settings

WLAN

Klar, mit so einem Smartphone möchte man gern ins Netz. Und wenn man noch keinen vernünftigen Daten-Tarif gebucht hat: Was liegt da näher, als das heimische WLAN zu nutzen? Oder das bei Freunden und Verwandten? Zumal es in der Regel ja auch schneller ist als die [mobile Datenverbindung](#). Was also ist zu tun?

In den Einstellungen wählen wir den Punkt „Drahtlos & Netzwerke“. Hier lässt sich WLAN schon erst einmal generell aktivieren (indem man das passende Häkchen setzt, bzw. den entsprechenden Schalter „umlegt“). Sodann tauchen wir in den Punkt „WLAN-Einstellungen“ ab – und gelangen zu einem Bildschirm, der dem rechts dargestellten ähnelt.

Der erste Punkt entspricht dem generellen Aktivieren der WLAN-Funktionalität (wie auf der vorigen Seite). Ist WLAN aktiv, und mit einem Netzwerk verbunden, wird das an dieser Stelle auch angezeigt. Darunter nun werden alle aktuell verfügbaren WLAN-Netze aufgelistet. Es wird ebenfalls dargestellt, ob (und wenn ja wie) diese verschlüsselt sind. Hier müsste also auch das „eigene“ WLAN (bzw. das, in welches sich eingebucht werden soll) nun stehen. Einfach antippen, ggf. den Schlüssel (das „Verbindungs-Passwort“) eingeben, und auf „Verbinden“ tippen – und wenige Sekunden später sollte die Verbindung stehen. Bei Bedarf lässt sich ab Android 4.0 nach Aktivieren der Checkbox *Erweiterte Optionen einblenden* auch ein für das jeweilige Netzwerk zuständiger Proxy-Server angeben.

Einmal eingegeben, merkt sich übrigens Android die Verbindungsdaten: Kommt man das nächste Mal bei aktiviertem WLAN in die Nähe dieses Netzes, erfolgt die Verbindung automatisch.

Am Ende des Bildschirms sind noch drei Symbole zu sehen. Mit diesen lässt sich etwa (sofern vom Router unterstützt) per WPS schnell und einfach eine verschlüsselte Verbindung aufbauen, ohne dass man ein kilometerlanges Passwort eingeben muss, oder (mit dem „+“ Symbol) ein Netzwerk manuell hinzufügen. Die drei übereinander gestapelten Punkte öffnen ein Menü für weitere Einstellungen, wie der Screenshot zeigt. So kann man unter „Erweitert“ beispielsweise festlegen, dass man „unterwegs“ über verfügbare offene WLAN-



Netzwerke informiert werden möchte. Wer jedoch vertrauliche Daten auf seinem Androiden hat, sollte mit solchen Netzen vorsichtig sein: Man weiß ja nie.

Mobiles Datennetz



Der „moderne Mensch“ ist ja heutzutage permanent online. Unser WLAN können wir aber nicht überall hin mitnehmen. Was tun?

Die Antwort heißt: Einen passenden Daten-Tarif (Volumentarif oder Flatrate) buchen, und das „mobile Datennetz“ konfigurieren! Ersteres gibt es beim Provider – und letzteres findet sich wieder unter „Drahtlos & Netzwerke“ (ab Android 4.0 versteckt es sich hinter *Mehr*). Der Punkt *Mobilfunknetze* (bzw. *Mobile Netzwerke*) führt dann zu einem Bildschirm ähnlich dem dem links dargestellten.

Ein kurzer Blick auf das Bild dürfte auch bereits helfen, eine der größten Sorgen auszuräumen: Was ist mit meinen Daten-

Kosten, wenn ich im Ausland bin? Ja was? Das hängt ganz davon ab, was auf dieser Seite konfiguriert wurde. Standardmäßig sind die Häkchen bei „Roaming“ *nicht* gesetzt (also wie auf dem Bild zu sehen). Im Ausland bzw. generell im Netz eines „Fremdanbieters“ wird daher die Datenverbindung gar nicht erst aufgebaut. Daher sollten hier auch keine diesbezüglichen Kosten entstehen.

Weiter unten lässt sich noch ein Häkchen setzen, welches die Datenverbindung auf „2G“ beschränkt (*Bevorzugter Netzwerktyp*). Das ist zwar nicht so schnell wie 3G oder gar 4G – spart aber u. U. einiges an Energie (siehe [2G versus 3G](#)), sodass man mit einer Akku-Ladung länger auskommt. Für ein wenig E-Mail und Web ist das auch völlig ausreichend. Sollte man tatsächlich einmal mehr Durchsatz benötigen, kann man das jederzeit umschalten.

Woher weiß Android denn nun, wie es ins Internet kommt? Diese Einstellungen verbergen sich hinter den „Zugangspunkten“. So manch [Custom-ROM](#) (wie z. B. [CyanogenMod](#)²¹) ermittelt diese Konfiguration automatisch: Anhand der SIM-Karte erkennt es den Anbieter, und ordnet die entsprechenden Zugangsdaten aus seiner Datenbank zu. Wer dieses Glück nicht auf seiner Seite hat, findet die



CyanogenMod
Homepage

21. <http://www.cyanogenmod.com/>



IzzyOnDroid: APN-Einstellungen ausgewählter Netzbetreiber

passenden Zugangsdaten jedoch hoffentlich [bei IzzyOnDroid](#)²² – andernfalls lassen sie sich beim Provider erfragen.



An dieser Stelle ist es später sicher auch interessant zu wissen, wie es eigentlich um den Datenverbrauch bestimmt ist: Wie viel Datenvolumen ist noch übrig? Welche App hat wie viel verbraucht – und wie lassen sich „Datenfresser“ im Zaum halten?

Seit Version 4.0 bringt Android die dafür notwendigen Werkzeuge selbst mit. Abgesehen vielleicht von einem Widget oder der Prognose des Datenverbrauchs bis zum Ende des Abrechnungszeitraumes, decken diese auch alles ab, was zuvor Tools wie [3G Watchdog](#)²³ erledigen mussten: Den Anwender warnen, bevor sein Datenvolumen aufgebraucht ist, bei Erreichen des Limits optional die mobilen Daten deaktivieren, und die Datennutzung nach Verbrauchern aufschlüsseln. Mehr noch, lässt sich sogar einzelnen Apps [die Nutzung von Hintergrunddaten unterbinden](#).

Zu finden ist all dies unter *Einstellungen > Datennutzung*, wo man auch die „mobilen Daten“ selbst (de-)aktivieren kann. Sind diese eingeschaltet, und die Checkbox „Limit für mobile Daten“ wurde aktiviert, lässt sich in der Klappbox darunter der Abrechnungszeitraum des Mobilfunkanbieters einstellen. Der rote und der orangefarbene Balken in der Grafik darunter steht jeweils für das Limit sowie die Warnschwelle, und kann entsprechend der eigenen Bedürfnisse verschoben werden.

Welche App nun wie viel Daten verbraucht hat, findet sich unterhalb der Grafik aufgeführt – die größten Verbraucher zuvorderst. Wer genau hinschaut, erkennt einen durch zwei senkrechte Balken eingegrenzten Bereich im Graphen: Auf diesen bezieht sich die genannte Auflistung. Auch diese Balken lassen sich manuell verschieben (der ausgewählte Zeitraum wird unmittelbar darunter angezeigt) – sodass sich der Datenverbrauch gut analysieren, und ein „Amoklaufender Datenfresser“ auch leicht isolieren lässt.

22. http://android.izzysoft.de/books.php?topic=apn_list

23. <https://play.google.com/store/apps/details?id=net.rgruet.android.g3watchdog>

Tethering



An dieser Stelle folgt oft die Frage: „Ich habe da noch ein Tablet/Notebook/... Kann ich jetzt irgendwie die mobile Datenverbindung mit nutzen?“ Vor Android 2.2 (aka „Froyo“ oder „Frozen Yoghurt“) hieß die Antwort eindeutig: Jein. Mit [root](#) und einer App wie [Wireless Tether](#)²⁴ ließ sich dies erreichen. Ansonsten galt der übliche Spruch: „Ohne root sich nix tut“.

Zum Glück hat sich das eindeutig geändert: Seit [Froyo](#) gehört Tethering zur „Standard-Ausrüstung“, und ist natürlich auch noch bei den aktuellen Versionen mit an Bord (siehe Screenshot). Wer sicher gehen möchte, dass kein Dritter „mitsurft“, kann das Netzwerk über USB weiterreichen. Einfacher geht es jedoch, wandelt man seinen Androiden in einen „mobilen Hotspot“ um. Hierzu wird der zweite Punkt aktiviert, und die Details werden unter

„WLAN-Hotspot-Einstellungen“ eingetragen: Eine SSID (Name für den Zugangspunkt) kann nach Gusto vergeben, eine Verschlüsselung gewählt, und natürlich auch der zugehörige Schlüssel/Passwort hinterlegt werden. Schon steht dem Surf-Vergnügen nichts mehr im Wege.

Je nach Hersteller, Gerät, Android-Version und [ROM](#) sieht dieses Menü allerdings unterschiedlich aus: Einmal fehlt *USB-Tethering* (zum Beispiel seinerzeit bei meinem *LG Optimus 4X* mit Stock Android 4.0.3), oder es sind zusätzlich Dinge verfügbar wie *WiFi Direct* oder *NFC*. Also bitte nicht wundern.



24. <https://play.google.com/store/apps/details?id=android.tether>

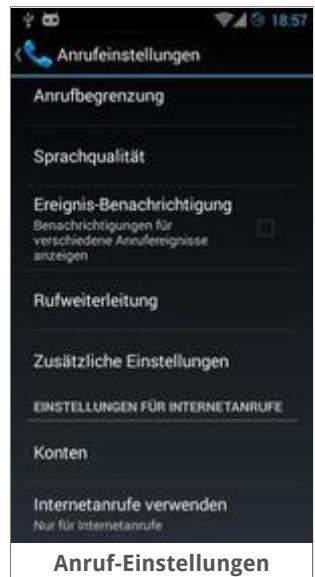
Internet-Telefonie

Oh nein, den Netzanbieter hat das sicher nicht sonderlich gefallen. Aber dennoch: Seit [Gingerbread](#) gehören die [SIP](#)-Einstellungen zu den Bordmitteln. Wer noch kein Gingerbread (oder eine aktuellere Version) auf seinem Androiden hat, muss dafür zu Drittanbieter-Software wie [SIPGate](#)²⁵ greifen, kommt aber auch zum Ziel.

Versteckt sind diese Konfigurationsdaten unter den „Anrufeinstellungen“ der Telefon-App (also *nicht* unter „Drahtlos & Netzwerke“ in den Systemeinstellungen), und zwar ganz am Ende des Bildschirms (siehe Abbildung). Von hier aus geht es über „Konten“ in die Übersicht eingerichteter SIP-Konten – beim ersten Aufruf dürfte diese leer sein. Ein Button, beschriftet mit „Konto hinzufügen“, wartet jedoch schon auf Betätigung. Die passenden Zugangsdaten gibt es beim VoIP-Anbieter. Essentiell sind Nutzernname, Passwort und Server – etliche optionale weitere Einstellungen wie Proxy u. a. stehen ebenfalls zur Verfügung.

Mittels einer Checkbox lässt sich ein SIP-Konto als „primär“ festlegen – was aber nur bei mehreren Konten interessant ist. Über dieses Konto werden dann ausgehende Anrufe geführt.

Auf einigen Geräten (mein *Optimus 4X* ist wieder einmal ein solches) sind die SIP-Funktionen allerdings einfach nicht auffindbar – obwohl Android 4.0 oder höher zum Einsatz kommt. Da wollen sich einige Hersteller scheinbar bei den Netzanbietern „lieb Kind machen“. Mit ein paar Tricks lassen sich die Konfigurations-Bildschirme oftmals dennoch aufspüren: So bietet beispielsweise der [Apex-Launcher](#)²⁶ die Möglichkeit, dem Homescreen außer Widgets und Shortcuts auch „Aktionen“ hinzuzufügen. Was sich dahinter verbirgt, ist eine Liste von allen Shortcuts, über welche diverse Apps zwar verfügen – sie aber nicht unbedingt offiziell anbieten. Hier sucht man nun nach der „Settings“ App (aka „Einstellungen“), und schaut, was sie interessantes anzubieten hat (sehr viel!). Einen Shortcut generiert, und schon kann man auch auf einem *LG Optimus 4X* auf die SIP-Einstellungen zugreifen.



SIPGate

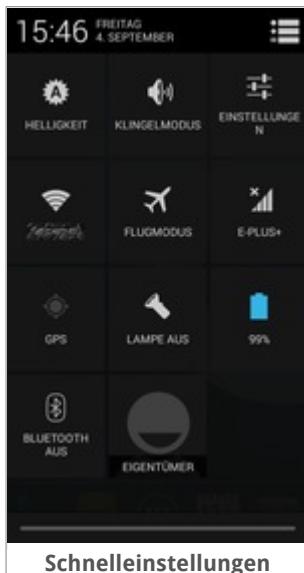


Apex Launcher

25. <https://play.google.com/store/apps/details?id=com.sipgate>

26. <https://play.google.com/store/apps/details?id=com.anddoes.launcher>

Schnelleinstellungen



Sich für jede kleine Änderung durch den Dschungel der System-Einstellungen kämpfen zu müssen, ist besonders für Android-Einsteiger alles andere als komfortabel. Musste man früher für Alternativen auf Drittanbieter-Apps zurückgreifen, hat sich in dieser Hinsicht in der Zwischenzeit zum Glück etwas getan. Nachdem zunächst einige Hersteller (z. B. Samsung) und Custom-ROMs (wie CyanogenMod) entsprechende Schalterchen in den Benachrichtigungs-Bereich integrierten, zog Android mit Version 4.2 schließlich selbst nach: Die „Quick Settings“ sind nun fester Bestandteil des Android-Systems. Je nach Android-Version kann man über diese die wichtigsten Einstellungen durch einfaches Antippen des zugehörigen Icons umschalten, und gelangt durch langes Drücken zum entsprechenden Konfigurations-Bildschirm – oder umgekehrt. Auch lassen sich Reihenfolge sowie Auswahl meist konfigurieren.

Wer jedoch noch eine ältere Android-Version einsetzt, oder mit dem Vorhandenen nicht zufrieden ist, findet in den App-Märkten nach wie vor zahlreiche Alternativen²⁷.



Übersicht: System-Einstellungen

Zusätzliche Einstellungen

Während die einen es lieber kompakter hätten, gibt es da auch noch die Gruppe derer, denen die vorhandenen Konfigurationsmöglichkeiten nicht ausreichen. Auch dieser kann (in einem gewissen Rahmen) geholfen werden: Unter Zusätzliche / versteckte Einstellungen²⁸ finden sich bei *IzzyOnDroid* zahlreiche Helferlein für spezielle Einstellungen des Displays, der Lautstärke-Regler, für Netzwerk, Benachrichtigungen, Synchronisation, und vieles mehr. Man möge mir vergeben, dass ich sie hier nicht alle vorstelle.



Übersicht: Zusätzliche / versteckte Einstellungen

27. http://android.izzysoft.de/applists/category/named/tools_settings

28. https://android.izzysoft.de/applists/category/named/tools_settings#group_537

Anwendungen verwalten

Um folgende Themen geht es in diesem Kapitel:

- Installieren von Apps
- Aktualisieren (neudeutsch: „Updaten“) von Apps
- Apps Bereinigen (Cache löschen etc.)
- De-Installieren (Löschen) von Apps

Also im Prinzip der ganz normale „Lebens-Zyklus“ einer App auf dem Androiden: Erst wird sie installiert (und benutzt), hin und wieder gibt es neuere Versionen, und schließlich – ist man ihrer überdrüssig geworden, oder hat etwas besseres gefunden – soll sie wieder vom Gerät verschwinden. Zwischendurch gilt es u. U. den Cache zu bereinigen: Entweder, um Platz freizuschaffen, oder um diverse kleine Probleme zu lösen. Für all diese Aufgaben gibt es verschiedene Ansätze.

Apps? APK-Datei?

Zuallererst gilt es noch ein paar Begriffe zu klären:

App nennt man die Anwendungen unter Android (und übrigens auch bei Apples iOS). Dieses Kürzel leitet sich nicht etwa von *Apple* ab, wie einige meinen, sondern vom Wort *Application* – jaja, der „Aküfi“ (Abkürzungs-Fimmel) ist nicht ausschließlich © Germany.

Eine APK-Datei enthält in der Regel eine solche App, und ist so etwas wie ein „Installations-Archiv“. Erhältlich im Play Store²⁹, und oftmals auch auf den Webseiten der jeweiligen Entwickler – diese beiden Quellen können als relativ sicher gelten (sowohl was „Systemsicherheit“ als auch was „Legalität“ anbetrifft; was in letzterem Fall natürlich beides insbesondere von der Vertrauenswürdigkeit des Entwicklers abhängt), sind sie auch in der „freien Wildbahn“ anzutreffen (etwa als Suchergebnis einer Google-Suche nach „<AppName>.apk“). Im letzten Falle ist allerdings Vorsicht geboten: Zum einen sind derartige Angebote nicht selten illegal, zum anderen die hier auffindbaren APK-Dateien oftmals auch manipuliert (Stichwort: DroidDream).

Ergo: Das Futter für den Androiden sollte besser ausschließlich aus verlässlichen Quellen besorgt werden. Und die schauen wir uns jetzt an.



Google Play Store

29. <https://play.google.com/store/>

Bordmittel

Natürlich bringt Android passende Hausmittel für die genannten Aufgaben mit – irgendwie müssen ja die ersten „Früchtchen“ in den Korb gelangen können. Zu diesem Zweck ist auf den meisten Androiden (einige Hersteller kochen hier wieder ihr eigenes Süppchen, so dass dies nicht pauschal für alle Android-Geräte gilt) eine App namens *Play Store* (ehemals *Google Market*) vorinstalliert. Dieser möchte ich zunächst ein wenig Grundwissen vorausschicken, bevor ich auf die App selbst eingehe:

Zunächst einmal ist der Playstore die wohl umfangreichste (und darüber hinaus auch die „offizielle“) Quelle für Android-Apps. Daher sind Entwickler natürlich bestrebt, ihre Produkte auf jeden Fall an dieser Stelle unterzubringen. Von wenigen Ausnahmen einmal abgesehen, kann man daher fast sagen: Eine App, die es hier nicht gibt, gibt es einfach nicht (oder sie ist von Google unerwünscht). Darüber hinaus ist der überwiegende Teil der Apps (etwa 75%) gratis verfügbar – was allerdings häufig mit Werbe-Einblendungen in den jeweiligen Apps verbunden ist, denn irgendwie möchte der Entwickler ja auch für seine Mühe entlohnt werden. Eine Tatsache, die übrigens auch für die meisten anderen App-Märkte gilt.

Die Nutzung des *Google Playstore* setzt einen gültigen [Google-Account](#) voraus. Diesen kann man sich direkt auf der Webseite gratis anlegen. Bei der Ersteinrichtung eines mit der genannten App ausgestatteten Androiden wird dieser auch abgefragt und kann, sollte man noch nicht über einen Account verfügen, direkt vom Einrichtungs-Assistenten aus erstellt werden (wer dies zunächst übersprungen hat, findet den entsprechenden Punkt in den „Einstellungen“ unter „Konten & Synchronisation“). Will man auch kostenpflichtige Apps nutzen, musste bislang zudem eine Kreditkarte mit dem Google-Account verknüpft werden – andere Bezahlmöglichkeiten waren leider, trotz zahlreicher Nutzerproteste, lange nicht verfügbar. Doch mittlerweile hat sich hier etwas getan: Mobilfunk-Anbieter können auch die Bezahlung über die Telefonrechnung ermöglichen. In Deutschland macht davon T-Mobile bei von ihnen selbst verkauften (also entsprechend „gebrandeten“, siehe [Branding](#)) Geräten Gebrauch, auch O2 hat mittlerweile nachgezogen (Vodafone ist [wieder abgesprungen](#)³⁰). Außerdem gibt es seit Juli 2013 endlich auch Gutschein-Karten im Handel zu erwerben, mit denen man sein Google-Konto aufladen kann. Nachdem Anwender diesen Wunsch lange Zeit ins Leere sprachen, hat überdies im Sommer 2014 auch Paypal als Zahlungsmittel hier Eingang gefunden.

Ist nun also ein Zahlungsmittel mit dem Google-Account verknüpft (oder das Konto mittels Guthaben-Karte aufgeladen), lassen sich kostenpflichtige Apps



AndroidPIT:
Vodafone – Bezahl
im Play Store via
Handyrechnung
nicht mehr möglich

30. <http://www.androidpit.de/vodafone-bezahlen-play-store-handyrechnung-nicht-mehr-moeglich>

über die Playstore-App erwerben. Also darauf aufpassen, dass der Sprössling nicht auf Einkaufstour geht – oder Unbefugte am Gerät spielen und Schabernack treiben! (Um dies zu verhindern, lässt sich der Bezahlvorgang mit einem Passwort schützen, was in aktuellen Playstore-Versionen auch automatisch unter Verwendung des Google-Passwortes geschieht). Außerdem ist die Rückgabefrist bei Fehlkäufen mit derzeit zwei Stunden etwas kurz geraten – hier gilt es also gegebenenfalls, schnell zu handeln, und den Fehlkauf sogleich wieder rückgängig zu machen.

Übrigens: Da jede im Playstore erworbene App mit dem zugehörigen Google-Account verknüpft ist, lassen sich einmal gekaufte Apps auf allen mit diesem Account versehenen Androiden nutzen. Also keine Sorge, dass man bei Gerätewechsel alles neu erwerben muss!

Play Store

Über den *Play Store* (ehemals *Google Market*, siehe Screenshot) soll der Android-Jünger sich also seine Apps besorgen. Die Fülle an Apps kann hier grob nach Rubriken durchblättert oder, so der Name der gesuchten App bekannt ist, auch gezielt durchsucht werden. Letzteres ist natürlich auch nach Stichworten möglich, die in Namen oder der Beschreibung der Apps vorkommen. Aufgrund der großen Anzahl an im Play Store verfügbaren Apps ist das Ergebnis aber nicht unbedingt immer befriedigend – zumal die Ergebnisliste mittlerweile auf 48 Einträge begrenzt wurde. Filtermöglichkeiten (etwa das Ausblenden unerwünschter Entwickler oder das Ausschließen bestimmter Begriffe) gibt es in der App leider nicht.

Etwas komfortabler wird das Ganze, wenn man [die Website des Play Store](https://play.google.com/store)³¹ mit dem Browser am PC benutzt: Hier lassen sich viele der aus der „erweiterten Google-Suche“ bekannten Tricks verwenden – etwa um mit dem Begriff vorangestellten „-“ Zeichen Begriffe auszuschließen (sogar einige erweiterte Attribute des „großen Bruders“, wie etwa „intitle:Begriff“ wenn der App-Name „Begriff“ enthalten soll, sind möglich). So findet man z. B. durch eine

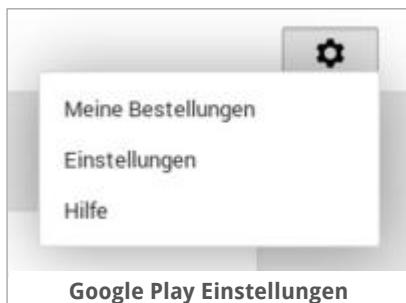


Google Playstore

31. <https://play.google.com/store/>

Suche nach „+scuba -log“ (oder „+dive -log“ - jeweils ohne die Anführungszeichen) Apps zum Thema Scuba-Diving (Tauchen), schließt jedoch Logbücher aus. Die Informationen lassen sich am größeren Bildschirm auch weit bequemer sichten. Ist die gesuchte App gefunden, kann sie überdies, sofern man mit seinem Google-Account angemeldet ist, mit einem einfachen Klick auf den Button „Installieren“ auf den Androiden befördert werden: Schon wenige Sekunden später sieht man dort in der Regel den Download und schließlich auch den Installationsprozess starten. Sind mehrere Geräte mit dem selben Google-Account verknüpft, lässt sich das gewünschte Zielgerät natürlich auswählen. Auch filtert der Play Store automatisch die Apps aus, die mit dem Zielgerät nicht kompatibel sind (siehe Anhang: „[Warum finde ich die App im Play Store nicht?](#)“).

Am linken Rand der Seite findet sich übrigens auch ein Navigationsmenü, dessen erste vier Punkte jedoch lediglich wieder in die jeweiligen Sektionen des Playstore zurückführen. Auch der fünfte, mit „Geräte“ beschriftete Button listet nicht etwa die eigenen Geräte auf, sondern führt wiederum in den entsprechenden Shop-Bereich. Ähnliches gilt auch für die Drop-Down-Box direkt rechts daneben: Die hier aufgeführten Punkte beziehen sich lediglich auf die eigenen Einkäufe, und filtern entsprechend die Liste. Interessant sind somit nur noch die letzten beiden Punkte, über die sich die eigene Wunschliste verwalten, bzw. Google-Play-Gutscheine einlösen lassen.



Die eigenen Geräte findet man im zweiten Punkt des „Rädchen“ (direkt unter dem Account-Namen oben rechts), der mit „Einstellungen“ beschriftet ist. Hier kann man seinen Geräten Namen geben und festlegen, ob sie in den Menüs auf App-Seiten als Installationsziel auftauchen sollen (Altgeräte entfernen kann man leider nicht). Auch Wünsche zu Email-Benachrichtigungen (Angebote vom Play Store u. a. m.) lassen sich hier festlegen. Sortieren lässt sich allerdings auch diese Liste nicht, was jedoch hier nicht so stark ins Gewicht fällt: Die Anzahl mit einem Account verwendeteter Geräte hält sich normalerweise in Grenzen. Wie sich dort ermitteln lässt, welche Apps man auf welchem Gerät installiert hat(te), findet sich unter [Fragen aus Alltag und Praxis](#).

Um nun aber die Nutzung des *Play Store* auf dem Androiden zu verbessern bieten sich, abgesehen von „[alternativen Market-Apps](#)“, auch diverse „[Market-Ergänzer](#)“ an.

Anwendungen verwalten

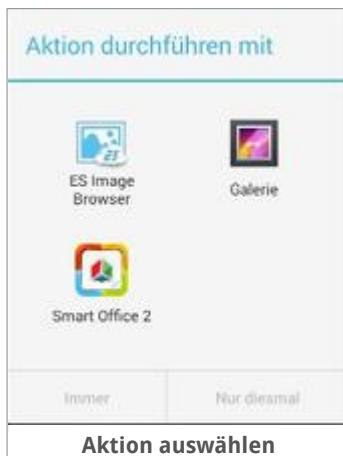


Der Punkt „*Anwendungen verwalten*“ (Screenshot links) findet sich unter *Einstellungen* > *Anwendungen* im Android-Menü, und ist in Tabs eingeteilt: Heruntergeladene (also selbst installierte; dieser Tab wird bei Aufruf von „Anwendungen verwalten“ geladen), Alle, Auf SD-Karte, Ausgeführte. Die ersten drei Tabs sind ruckzuck geladen, der vierte braucht ein paar Sekunden und ist ein vollwertiger Task-Manager.

Ist die Liste aufgebaut, lassen sich Details zu den Anwendungen einsehen. Dies geschieht, indem die gewünschte App in der Liste kurz angetippt wird. Sodann offenbart sich: Wie viel Platz belegt die App selbst, wie viel ihre Daten. Wie viel Cache benutzt sie. Und so weiter. Von hier aus lassen sich dann z. B. auch der **Cache leeren**, die **Daten löschen** – oder die Anwendung deinstallieren (löschen).

Alternativen zur Cache-Bereinigung finden sich im Kapitel [Tuning](#).

Default-Apps



Verschiedene Aktionen sind bei Android mit bestimmten Apps verknüpft. Betätigt man beispielsweise die Taste „Home“ (bzw. den entsprechenden „Soft-Key“), so gelangt man zu seinem Home-Screen. Wie im entsprechenden Kapitel beschrieben, gibt es für diesen Zweck eine größere Auswahl an Apps. Was nun, wenn mehrere davon gleichzeitig auf dem Gerät installiert sind – welche App soll dann gestartet werden? Für eine sinnvolle automatische Auswahl fehlt dabei einfach der Kontext.



Vernünftigerweise führt Android in einer derartigen Situation keine Zufalls-Auswahl durch, sondern fragt den Anwender nach dessen Präferenzen – wie im Screenshot zu sehen. Möchte man dies von Fall zu Fall erneut entscheiden, tippt man nach Auswahl der App auf den Button „Nur diesmal“. Damit erfolgt die Abfrage beim nächsten Mal erneut. Das ergibt etwa Sinn, wenn man neben der „normalen“ Telefon-App noch eine weitere App installiert hat, die über Telefonie-Funktionen verfügt – beispielsweise *Skype* oder *Linphone*. Telefonate im Funk- und Festnetz führt man dann etwa bevorzugt mit der „normalen“ Telefon-App durch, wählt aber die günstigere Alternative für Auslandstelefonate. Möchte man jedoch nicht jedes Mal neu entscheiden müssen, tippt man auf „Immer“ – und wird in Zukunft nicht mehr behelligt, was insbesondere beim Home-Screen sinnvoll erscheint.

Speziell modebewusste Anwender (hust, hust) ändern jedoch hin und wieder ihre Vorlieben. Was also tun, wenn man den „für Immer“ eingestellten Default einmal ändern möchte? Dafür gibt es verschiedene Möglichkeiten:

- Im zuvor beschriebenen Menü Anwendungen verwalten die derzeitige Default-App aufsuchen, deren Details aufrufen, und etwas nach unten

scrollen. Dort findet sich ein Button, mit dem sich die „Standardeinstellung zurücksetzen“ lässt (siehe Screenshot). Beim „nächsten Versuch“ erfolgt dann wieder die Abfrage.

- Eine weitere App installieren, welche für diese Aktion zuständig ist. Das ändert den „Status Quo“, und die Abfrage erfolgt ebenfalls aufs Neue.
- Eine [App zur Verwaltung von Default-Apps³²](#) installieren. Diese bieten den Vorteil, dass man nicht groß überlegen muss, wie die aktuelle Default-App heißt.



IzzyOnDroid:
Default-Apps
festlegen



Playstore-Ergänzungen

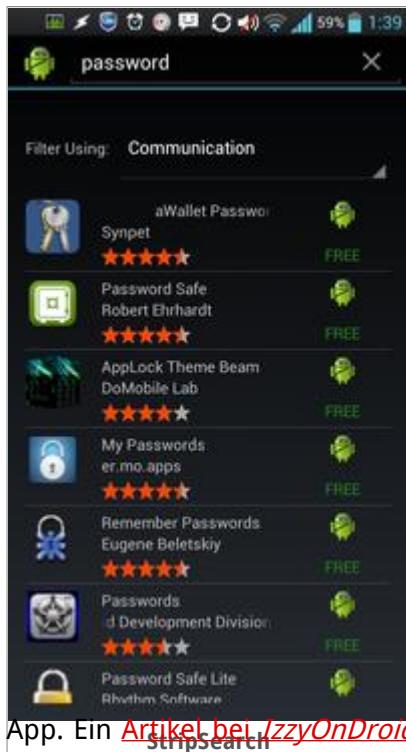
Im Google Playstore findet sich eigentlich alles – die Frage ist nur, wie lange man für die Suche benötigt. Besonders Besitzer eines Tablets ärgern sich oft darüber, dass die Apps das größere Display nicht vernünftig ausnutzen. Oder dass es so schwierig ist, Apps zu finden, die dies tun. Zum Glück gibt es da Abhilfe: Apps wie [Tablet Market³³](#) und [Tablified Market HD³⁴](#) haben sich nämlich darauf spezialisiert, nur Tablet-geeignete Apps aufzuspüren.

Tablet Market



Tablified Market HD

32. http://android.izzysoft.de/applists/category/named/apps_organize#group_5
33. <https://play.google.com/store/apps/details?id=com.andromo.dev86.app119>
34. <https://play.google.com/store/apps/details?id=tablifiedapps.tablifiedmarket>



App. Ein [Artikel bei IzzyOnDroid](#)

Ist jemand hingegen – nicht zu Unrecht – besonders besorgt bezüglich der Berechtigungen, die einige Apps verlangen, greift er für die App-Suche im Playstore am besten auf [StripSearch](#)³⁵ zurück. Diese App stellt ein alternatives Front-End für die Suche im *Google Playstore* dar – welches wesentlich aufgeräumter wirkt: Man wird nicht überall mit zahlreichen Grafiken bombardiert, sondern die gewünschten Informationen stehen im Vordergrund. Direkt von der Startseite aus lässt sich eine Suche starten, nachdem man optional einen passenden Filter gesetzt hat. Einige sinnvolle und nützliche Filter sind bereits vorkonfiguriert – lassen sich aber sowohl den eigenen Bedürfnissen anpassen, als auch um weitere persönliche Filter ergänzen. Eine Obergrenze für den Preis kann man ebenso vorgeben, wie die Sortierreihenfolge der Ergebnisliste. Für Details zur und Installation der App geht es dann wieder in die Playstore-

„App-Suche nach Permissions“ lieber mit dem

Web-Browser vornimmt, findet dort auch [eine Möglichkeit dazu](#)³⁶ – auch wenn selbige lediglich einen Bruchteil der in den Märkten verfügbaren Apps abdeckt.

Außerdem gibt es [Apps für App-Schnäppchen-Jäger](#)³⁷, die bei ihrer App-Auswahl weniger „auf ihre Kosten“ kommen wollen.

Genervt von der aktuellen PlayStore-App? Verärgert, dass die Liste aller jemals gekauften (und u. U. nicht mehr installierten) Apps verschwunden ist? Ist das Android-Gerät gerootet (siehe [Super-User „root“](#)), kann Paul O'Brians [Legacy Play Store](#)³⁸ Abhilfe schaffen. Paul hat eine ältere Version (als der Play Store noch Market hieß) gepatcht, sodass sie sich parallel zur ständig zwangswise aktualisierten PlayStore-App installieren lässt (natürlich greift sie dabei auf die gleiche Quelle zu, wie auch Googles eigene Playstore-App). In diesem Fall lassen



StripSearch



IzzyOnDroid: Mit StripSearch die Appsuche nach Permissions filtern



App-Suche nach Permissions bei IzzyOnDroid



Übersicht: Topp Apps, Apps Sonderangebote, und ähnliches



Legacy Play Store

- 35. <https://play.google.com/store/apps/details?id=com.hasslefixes.stripsearch>
- 36. <http://android.izzysoft.de/applists/search>
- 37. http://android.izzysoft.de/applists/category/named/apps_markets#group_9
- 38. <http://www.modaco.com/topic/356009-legacy-play-store-with-all-purchased-apps-visible/>



MarketEnabler

sich auch mit [MarketEnabler](#)³⁹ die regionalen Beschränkungen aushebeln: Diese App gaukelt dem *Play Store* vor, man wäre mit einem ganz anderen Provider in einem ganz anderen Land unterwegs. Und mit AT&T in den Staaten dürfen „US only“ Apps natürlich installiert werden 😊

Stack Exchange:
alternative-markets
tag-wiki

Playstore-Alternativen

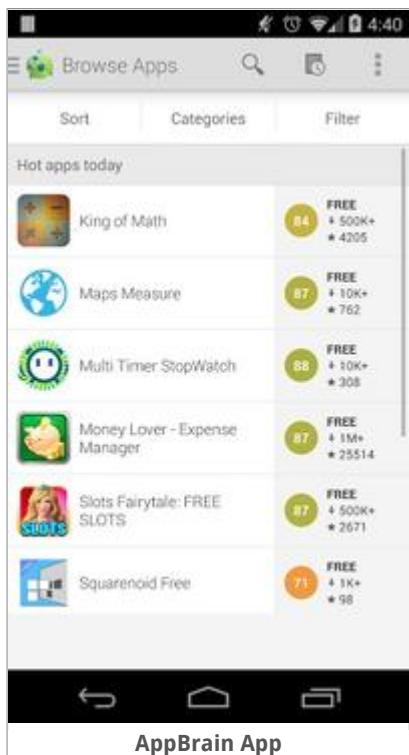
Derer gibt es viele: Amazon, AppBrain, PDassi ... Da fällt es schon bald schwer, über alle auf dem Laufenden zu bleiben. Diese Übersicht soll auch keinesfalls vollständig sein – vielmehr beschränke ich mich auf ein paar Beispiele, die mir besonders sinnvoll erscheinen. Für weitere Quellen möchte ich auf [Stack Exchange](#)⁴⁰ verweisen.

Eines ganz zu Anfang: Auch wenn es durchaus Sinn machen kann, mit mehreren/verschiedenen dieser Alternativen parallel zu arbeiten empfiehlt es sich, Einkäufe immer an der gleichen Stelle zu machen. Sonst verliert man recht leicht den Überblick – und weiß etwa nach einer Neuinstallation oder dem Wechsel auf ein neues Gerät nicht mehr, aus welchem Market man nun die gekaufte App wieder bekommt, ohne sie nochmals bezahlen zu müssen.

39. <https://code.google.com/p/market-enabler/>

40. <http://android.stackexchange.com/tags/alternative-markets/info>

AppBrain



[AppBrain](#)⁴¹ ist weniger ein alternativer Store – als vielmehr ein alternatives, aufgeräumteres Front-End zum *Google Play Store*. Die Kombination aus Website und App kann sich durchaus sehen lassen: Da alles weit weniger überfrachtet ist, laden die Seiten auch wesentlich schneller.

Über einen Filter lässt sich die Auswahl optional einschränken: gratis- oder Kauf-Apps, neue Apps, aktualisierte, sowie im Preis gesenkte Apps sind hier die Kriterien. Man kann den Katalog nach Kategorien durchstöbern, und die Ergebnisliste sortieren. Ergänzt wird dies von einer Liste mit Empfehlungen, die anhand der bereits installierten Apps ermittelt werden. Hier kann man „ungewünschte Artikel“ auch entfernen, und erhält dann wieder neue Vorschläge.

Gut gelöst ist auch das Update: Egal, aus welcher Quelle eine App installiert wurde – sofern sie im *Play Store* enthalten ist, wird sie auch von *AppBrain* gefunden. Nach der

Synchronisation der Liste von auf dem Androiden installierten Apps mit der im eigenen AppBrain-Konto (der Login dort erfolgt mit dem eigenen Google-Konto) einmal auf den Button „Perform Installs“ gedrückt, und ab die Lutze! wird alles in einem Rutsch gemacht. Naja, fast – eine kleine Mogelpackung ist es naturgemäß, schließlich muss der Telefon-Besitzer ja noch die „Permissions“ abnicken. Und das erfolgt dann lustigerweise wieder in der originalen Play Store-App.

Ein weiteres Plus dieser App: Einzelne Apps lassen sich vom Update ausschließen. Bei diesen erfolgt dann auch keine Benachrichtigung über verfügbare Updates mehr, ebenso werden sie beim gerade beschriebenen Sammel-Update nicht mehr angefasst. Auch können einzelne Updates einer App übersprungen werden – dann erfolgt eine neue Benachrichtigung für diese erst wieder beim nächsten Update. Beides Dinge, die man in Googles *Play Store* vergeblich sucht.

Wer sich nach dem letzten Fliesenleger-Update auf der *Google Play* Website nicht mehr zurecht findet, oder sich von der Beschränkung auf gerade einmal 48



AppBrain

41. <http://www.appbrain.com/>



AppBrain Website

Einträge in den Suchergebnissen eingeschränkt fühlt, sollte auch einmal auf der [AppBrain Website](#)⁴² vorbeischauen. Leider ist die Übersichtlichkeit der App-Detail-Seiten für den End-Anwender einem Relaunch im vierten Quartal 2015 zum Opfer gefallen, was hoffentlich im Nachgang noch wieder bereinigt wird; Entwickler dürften sich jedoch sicherlich über die zahlreichen Statistiken freuen. Für die App-Installation gibt es extra einen „Google Play“ Button, sodass man seinen „Fund“ nicht erst wieder neu dort suchen muss. Dass (potentielle) „Spam-Apps“ in der Ergebnisliste „ausgefiltert“ werden, ist ein weiteres großes Plus.



Amazon App-Shop

Alternative „geschlossene“ Märkte

Als derzeit größter und bekanntester Gegenspieler zu Google's *Play Store* muss natürlich der [Amazon App-Shop](#)⁴³, ein von Amazon eigens eingerichteter Appstore für Android, genannt werden. Wie man es als Amazon-Kunde ohnehin bereits gewohnt ist, finden sich auch hier „personalisierte Empfehlungen“ – und wie von anderen Stores gewohnt, Bewertungen und Rezensionen anderer Nutzer. Für Kauf-Apps gilt: Sie verifizieren ihre Lizenz über die entsprechende Amazon-App. Entfernt man also letztere von seinem Gerät, stellen die im Shop erworbenen Kauf-Apps früher oder später (bei der nächsten Lizenz-Prüfung) ihren Dienst ein.



PDassi

Der Vollständigkeit halber sei auch die App von [PDassi](#)⁴⁴ an dieser Stelle kurz erwähnt. Diese bietet u. a. zusätzliche Bezahlmethoden wie z. B. Paypal, Bankeinzug oder Überweisung – für all jene, die entweder keine Kreditkarte haben oder diese nicht mit ihrem Google-Account verknüpfen möchten.

Öffentliche Märkte

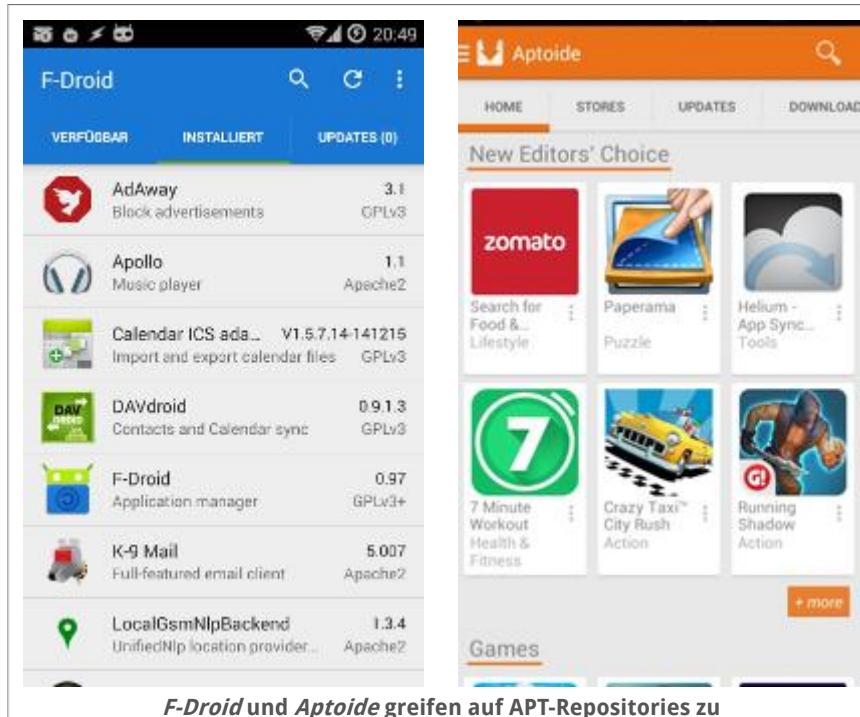
Linux-Anwender kennen „Software [Repositories](#)“: Diese halten Software-Pakete bereit (und pflegen Updates für selbige), welche die Entwickler selbst einstellen. Je nach Betreiber des jeweiligen Repositories sind Restriktionen für das Einstellen hier wenig bis gar nicht vorhanden. Der lesende Zugriff seitens der Anwender (für die Suche nach Software und deren Installation auf dem eigenen Rechner) ist entweder allen möglich („public“ bzw. „öffentliches“ Repository), oder nur einem ausgewählten Personenkreis (z. B. firmenintern, oder für Produkt-bezogene Entwicklung).

42. <http://www.appbrain.com/>

43. <http://www.amazon.de/mobile-apps/b?node=1661648031>

44. <http://android.pdassi.de/>

Einer der bekanntesten Repository-Typen ist [APT](#)⁴⁵, das Advanced Packaging Tool – hauptsächlich bei [Debian](#)⁴⁶ und dessen [Derivaten](#) im Einsatz. Gibt es aber auch für Androide-Soft:



F-Droid und Aptoide greifen auf APT-Repositories zu

Mit Tools wie [F-Droid](#)⁴⁷ (linkes Bild) oder [Aptoide](#)⁴⁸ (rechtes Bild – der Name *Aptoide* ist ganz offensichtlich eine Kreuzung aus „APT“ und „Androide“) lässt sich auf derartige Repositories zugreifen. Da Tools zur Pflege solcher Repositories ebenfalls existieren (und zwar als OpenSource Anwendungen), steht auch dem eigenen Repository (etwa für Entwickler, oder auch Firmen) nichts im Wege. An diese Zielgruppe richtet sich *Aptoide* (Entwickler/OEMs können hier ihren eigenen Store realisieren), während sich *F-Droid* auf OpenSource Apps konzentriert. Details zu diesen beiden Marktplätzen finden sich u. a. in meinem Artikel [Android Markets: Wie sicher sind alternative Quellen?](#)⁴⁹



Wikipedia: APT



Wikipedia: Debian



F-Droid



Android Markets:

-
- 45. http://de.wikipedia.org/wiki/Advanced_Packaging_Tool
 - 46. <http://de.wikipedia.org/wiki/Debian>
 - 47. <http://f-droid.org/>
 - 48. <http://apps.store.aptoide.com/app/market/cm.aptoide.pt/406/3338044/Aptoide>

Weitere

Ständig tauchen weitere Alternativen auf, die mehr oder weniger kurzlebig sind. Benötigt man einmal eine ältere Version einer App, hilft ein Blick in den [Android Drawer](#)⁵⁰: Hier finden sich .apk Dateien freier Apps sowohl in der aktuellen, als auch in historischen Versionen (wie übrigens auch bei *F-Droid*). Auch [SlideMe](#)⁵¹ ist kein Unbekannter in diesem Bereich.

Eine Übersicht alternativer Marktplätze findet sich, wie bereits erwähnt, bei [Stack Exchange](#)⁵².



Android Drawer

Stack Exchange:
alternative-markets
tag-wiki

Alternative Verwaltung

Jetzt haben wir alles Mögliche installiert und in Gang gebracht. An dieser Stelle fragte schon Goethes Zauberlehrling:

*Ach, da kommt der Meister!
Herr, die Not ist groß!
Die ich rief, die Geister,
Werd ich nun nicht los.*

Wie bzw. wo also nun de-installieren? Die „Hausadresse“ findet sich, wie bereits unter „Bordmittel“ festgestellt, unter *Einstellungen* › *Anwendungen* › *Anwendungen verwalten* – ist aber keinesfalls die einzige Möglichkeit. So gut sie auch mittlerweile gelöst ist: Da gibt es einiges, was fehlt, oder sich besser machen ließe.

Höchste Zeit also für die Ghost Busters, Krümel- oder besser: [AppMonster](#)⁵³:



AppMonster

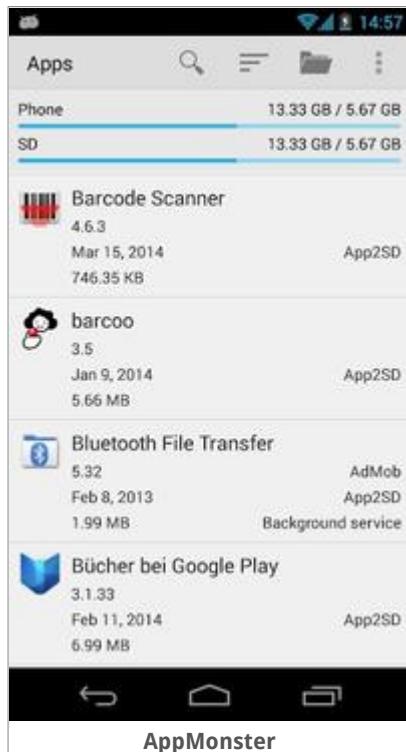
49. http://android.izzysoft.de/articles/named/android_markets_safe_to_use

50. <http://www.androiddrawer.com/about/>

51. <http://www.slideme.org/sam>

52. <http://android.stackexchange.com/tags/alternative-markets/info>

53. https://play.google.com/store/apps/details?id=de.android_telefonie.appmanager



Wie am linken Screenshots unschwer zu erkennen, handelt es sich hier um einen vollwertigen Software-Manager. Für ca. drei Euro gibt es die Vollversion – nachdem man zunächst natürlich die Gratis-Version ausgiebig testen kann. Einmal installiert, läuft die MonsterApp im Hintergrund (äh, nicht wirklich – genau genommen wird sie bei Bedarf vom Event-Manager aufgerufen; nämlich immer dann, wenn etwas Neues installiert wurde). Wurde etwas Neues installiert, schlägt das Monster zu – und macht sogleich ein Backup auf die SD-Karte. Kommt ein Update – schwupps, das Gleiche. Wie, die neue Version tut nicht? AppMonster, hol mal die vorige raus! Kein Thema.

Wipe, Flash, neues ROM – na und? AppMonster installiert, „Batch Install“ des aktuellen Backups – und schon sind alle Apps wieder da. Nagut, für die Daten braucht's dann allerdings root.

Was wollten wir eigentlich? Achso, loswerden wollten wir eine App – wie zu erwarten, findet sich dieser Punkt im jeweiligen Kontext-Menü (also lange auf den entsprechenden Eintrag „drücken“). Dazu muss die App natürlich erstmal aufgerufen werden – was jedoch zügig vonstatten geht, da sich AppMonster ausschließlich für die vom Benutzer installierten Apps interessiert – und die anderen brav in Ruhe sanften lässt (oder so).

Ist das Android-Gerät gerootet, mausert sich das Monster zur vollwertigen Backup-App, die sich auch um die zugehörigen Daten kümmert.

Neben Apps zur „alternativen App-Verwaltung“ gibt es dann auch solche, die sich auf das Apps entfernen⁵⁴ spezialisiert haben, oder auf das Verschieben von Apps auf die SD-Karte. Und zahlreiche weitere.



Übersicht: Apps entfernen und mehr

54. https://android.izzysoft.de/applists/category/named/apps_organize#group_4

Apps aus „alternativen Quellen“

Die Voreinstellung eines Android-Smartphones besagt: „Du sollst keine anderen Quellen haben neben mir“. Und „mir“ meint natürlich den *Play Store*. Dahinter steht der Sicherheits-Gedanke: Apps sollten nur aus vertrauenswürdigen Quellen installiert werden. Und die einzige derartige, die Google kennt, ist nun einmal Google.

Hin und wieder will/muss man aber mal eine App aus „alternativen Quellen“ installieren: Sei es, dass einem der Entwickler was zum „Testen“ zugeschickt hat („Schau mal, ob das Dein Problem löst!“), oder eine App mit dem Browser heruntergeladen wurde, da das Android-Gerät es im *Play Store* nicht findet – oder, oder, oder. Okay, die *.apk-Datei haben wir nun – aber wie die App installieren?

Klar kann man das *.apk einfach in den passenden Ordner von *AppMonster* (siehe voriges Kapitel) packen, und es dann damit installieren. Einfacher machen es zahlreiche Datei-Manager, die dann beim Antippen einer solchen Datei den Installer aufrufen. Zu den beliebtesten Kandidaten hier zählen [Astro Dateimanager⁵⁵](#) und [ES Dateimanager⁵⁶](#) – beide gut erweiterbar, und z. B. auch für den Zugriff auf lokale Netzwerke via FTP oder Windows-Freigaben (aka „SMB“) geeignet.



Astro Dateimanager



ES Dateimanager

Welche dieser Möglichkeiten man aber auch verwenden will: Immer kommt der Hinweis „Du darfst hier nicht rein!“ – denn zuerst muss die Installation aus „Fremdquellen“ generell einmal erlaubt werden. Mit einem kleinen Häkchen an der richtigen Stelle. Dieses findet sich je nach Android-Version unter *Einstellungen > Anwendungen* oder *Einstellungen > Sicherheit*, und ist dort mit „Unbekannte Quellen“ beschriftet.

Apps organisieren

Jetzt sind jede Menge Apps installiert und die Frage drängt sich auf: Wie soll man da den Durchblick behalten? Öffnet man den „Drawer“ (also die Liste der auf dem Gerät verfügbaren Apps), ist die Liste recht lang. Und nicht unbedingt übersichtlich. Alle 87 Apps (oder wie viel auch immer) teilweise ohne jede erkennbare Ordnung (oder im besten Falle alphabetisch sortiert) in einem Ordner.

Zwanzig mal hin-und-her scrollen auf der Suche nach der zu startenden App ist nicht jedermannss Sache. Wie leicht passiert es, dass man ein wenig „zu kräftig

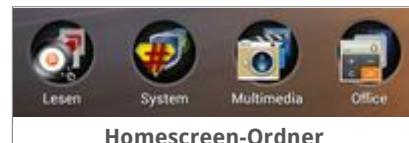
55. <https://play.google.com/store/apps/details?id=com.metago.astro>

56. <https://play.google.com/store/apps/details?id=com.estrong.s.android.pop>

schubst“ – und schon scrollt die Liste in einem Wahnsinns-Speed vorbei. Oder man schubst „zu langsam“ – und das dumme Teil meint, die gerade unter dem Daumen befindliche App starten zu müssen ... Und das „Vollklatschen“ aller Desktops mit Icons für jede App ist auch nicht unbedingt die wahre Lösung. Was also tun?

Bordmittel

Bei aktuellen Android-Versionen (spätestens ab Version 4.0 aka Ice Cream Sandwich) bietet bereits der Standard-Launcher die Möglichkeit, auf dem Homescreen Ordner anzulegen – und auf diese Weise häufig genutzte Apps für den schnellen Zugriff zu gruppieren. Zur Erstellung eines solchen Ordners zieht man einfach das Icon einer App auf das einer anderen: Damit wird der Ordner angelegt, und die beiden Apps wandern in selbigen. Den Namen des Ordners kann man anschließend bei geöffnetem „Folder“ bearbeiten.



Homescreen-Ordner

Wie das Ergebnis aussieht, erkennt man in den bereits zum Thema Schaltzentrale eingebundenen Screenshots – zur Rechten noch einmal der relevante Teil: Es sind quasi „Widgets“ oder „Shortcuts“, die ihren Inhalt bei Antippen als „Overlay“ preisgeben. Wem dies nicht genügt, der greift zu einem alternativen Launcher: Einige dieser Kandidaten verfügen über erweiterte Möglichkeiten, und lassen etwa Ordner auf der Schnellstart-Leiste ablegen, oder erlauben sie auch im App-Drawer.

Drittanbieter

Die wohl bekanntesten „Organisierer“ sind Apps Organizer⁵⁷ (der leider seit 2011 nicht mehr aktualisiert wurde) und Folder Organizer⁵⁸, der sich selbst als „the evolution of Apps Organizer“ bezeichnet. Die App kann alles das, was Apps Organizer auch kann – und mehr. So lassen sich Apps, Lesezeichen, Kontakte und „Shortcuts“ (z. B. zu Systemeinstellungen) mit „Labels“ versehen – und dann nach selbigen auflisten. Jedes Label kann auch als „Folder“ auf dem Homescreen platziert werden. Je nach Bildschirmgröße (und -auflösung) sowie Anzahl der Apps mit dem zugehörigen Label ist nun oftmals gar kein Scrollen mehr nötig: Der Start einer App klappt jetzt also mit nur zwei Taps! Die mit diesen



Apps Organizer



Folder Organizer

57. <https://play.google.com/store/apps/details?id=com.google.code.appsorganizer>

58. <https://play.google.com/store/apps/details?id=com.abcOrganizer>

„Organizern“ erstellten Labels lassen sich zudem auch in [Titanium Backup](#) zur [Auswahl von Apps](#)⁵⁹ verwenden.



Titanium Backup
Usability Tips

Folder Organizer kümmert sich nicht nur um Apps, sondern auch gleich mit um Kontakte, Lesezeichen, und mehr

Natürlich gibt es noch eine ganze Reihe weiterer Kandidaten – die sich jedoch im Großen und Ganzen letztendlich weitgehend mit einer der beiden gerade vorgestellten Apps vergleichen lassen. Zu finden sind diese in der zugehörigen Übersicht ([Apps Organisieren](#)⁶⁰) auf meiner Android-Website. Außerdem bieten einige „alternative Launcher“ (siehe [Home Replacements](#)) von Haus aus die Möglichkeit, Ordner direkt im „App-Drawer“ oder auf dem Homescreen anzulegen, wie bereits erwähnt.



Übersicht: Apps Organisieren

59. http://www.titaniumtrack.com/kb/titanium-backup-kb/titanium-backup-tips-suggestions.html#Usability_tips

60. http://android.izzysoft.de/applists/category/named/apps_organize

Datensicherung

Jetzt ist klar, wie die Apps auf den Androiden (und auch wieder von selbigem herunter) kommen, wie man sie verwaltet und organisiert. Natürlich würde es völlig den Rahmen sprengen, die Funktionsweise aller möglichen Apps hier zu erklären – aber einen wichtigen Punkt gibt es noch: Backups. Vielleicht muss das Android-Gerät ja irgendwann [auf Werkseinstellungen zurückgesetzt](#) werden, ein neues Android-Gerät kommt ins Haus, oder ein Update „zerschießt“ etwas! Die Apps lassen sich zur Not von Hand wieder zusammensuchen. Aber warum umständlich, wenn es auch einfacher geht? Darüber hinaus ist es gut zu wissen, dass die Daten dann wieder zur Hand sind – denn die lassen sich nicht so leicht wieder „irgendwo auftreiben“.

AppMonster habe ich ja bereits unter [Anwendungen verwalten](#) kurz vorgestellt: Es sichert bei jeder Installation (und jedem Update) die jeweilige App. So lässt sich nicht nur zu einer beliebigen, bereits zuvor einmal installierten, Version einer App zurückkehren – sondern auch auf einen Rutsch die jeweils aktuelle Version jeder zuvor installierten App aufspielen. Dies macht zum Beispiel bei einem Gerätewechsel, aber ebenso nach einem Werksreset viel Sinn – vor allem, wenn nicht alle Apps direkt über den *Google Play Store* installiert wurden. In diesem Falle ginge das nämlich auch über die Play Store-App – allerdings nur, solange die App auch noch im Play Store verfügbar ist.

Was aber ist mit Anwendungs-Daten? Was ist mit den Kurznachrichten, Telefonbüchern, und so weiter? Die Kontakte lassen sich noch aus der gleichnamigen App (*Menü > Importieren/Exportieren*) sichern bzw. wieder herstellen. Oder sie werden mit dem Google-Konto bzw. einem anderen Dienst synchronisiert, was auch mit den Kalender-Einträgen geht. Für alles andere steht gar kein Bordmittel bereit (warum eigentlich nicht?). Aber auch hier gibt es Abhilfe. Doch bevor wir uns die Backup-Helperlein ansehen, gilt es noch eine wichtige Frage zu klären:

Wie Android die Daten verwaltet

Was wollen wir eigentlich sichern? Wie speichert Android was, und wer kann worauf zugreifen? Mit diesen Grundlagen ist es einfacher, die verschiedenen Backup-Helfer zu verstehen.

Datentyp	Beschreibung
Apps	Diese werden im Allgemeinen in Form von .apk Dateien in einem Bereich abgelegt, auf den jede App lesend zugreifen kann.
„private“ App-Daten	Jede App erhält automatisch einen nur für sie reservierten Bereich zum Speichern von Daten zugewiesen. Außer der App selbst hat normalerweise nur das System (bzw. root) darauf Zugriff. Eine andere „normale App“ kann diese Daten daher nicht sichern.
„zentrale Daten“	Einige Daten werden als „zentral“ betrachtet, und vom System selbst verwaltet. Diese stehen allen Apps über so genannte „Provider“ zur Verfügung, sofern sie über die entsprechende Berechtigung (siehe Android Permissions) verfügen. Hierzu gehören beispielsweise SMS, Kontakte, und Anruflisten.
„shared data“	Wörtlich etwa „geteilte Daten“. Dies bezieht sich auf einen Speicherbereich, auf den alle Apps gleichermaßen Zugriff haben. In der Regel ist dies die SD-Karte .
Systemdaten	Auf einige Daten hat nur das System selbst (und natürlich root) Zugriff. Teilweise überschneidet sich dieser Bereich mit den „zentralen Daten“, wenn entsprechende Schnittstellen existieren.

Betrachtet man obige Tabelle, so wird schnell klar: Da gibt es zumindest zwei Bereiche, die eine „normale Backup-App“ nicht sichern kann, da sie keine Zugriffsmöglichkeit hat. Dies betrifft zum einen die „privaten App-Daten“, und zum anderen die Systemdaten. Verspricht eine App also ein „vollständiges Backup“, ohne dass sie root benötigt, kann (und sollte) man mit Recht skeptisch sein. Erst ab Android 4.0 weicht diese Grenze ein wenig auf. Mit dieser Version hat Android eine Möglichkeit geschaffen, das System um die Erstellung eines Backups zu bitten (siehe [vollständiges Backup ohne root](#)).

„Shared Storage“ – oder „Die“ SD-Karte

Einige fragen jetzt vielleicht: „Welche?“ Entweder, weil sie ein aktuelles Nexus-Gerät haben (welches über keinen microSD-Einschub verfügt) – oder weil ihr Gerät gleich zwei SD-Karten ausweist: Eine interne und eine externe SD-Karte.

Gedacht war sie ursprünglich als Erweiterung für größere Datenmengen. Doch sowohl bezüglich ihres Einsatzes als auch ihrer Umsetzung und Verwendung hat sich im Verlauf der Android-Evolution einiges geändert. Ein kurzer geschichtlicher Abriss soll dies belegen und veranschaulichen:

Als erstes Android-Smartphone für den „Massenmarkt“ erschien im Juni 2009 das HTC Dream⁶¹, ausgestattet mit „üppigen“ 256 MB internem Speicher und einem Einschub für microSD-Karten. Die „externe SD-Karte“ war also von Anfang an dabei. Sogar das allererste Nexus-Gerät⁶², erschienen im Januar 2010 mit installiertem Android 2.1, war mit einer solchen bestückt – und verfügte offensichtlich noch nicht über eine „interne SD-Karte“. Den ersten Beleg für letztere fand ich erst in einem Forensics-Blog von 9/2010⁶³, welcher das im Juni 2010 (ebenfalls mit Android 2.1) erschienene Samsung Galaxy S i9000⁶⁴ verwendete – und explizit die „interne SD-Karte“ erwähnte.

Von etwa diesem Zeitpunkt an gab es Android-Smartphones, die ausschließlich mit einer internen (überwiegend Nexus-Geräte), ausschließlich einer externen (nur noch wenige aktuelle Modelle), oder aber beiden SD-Karten ausgestattet waren (letztere dürften am häufigsten anzutreffen sein). Eine „dumme Sache“ mit der internen SD-Karte war jedoch, da sie sich auf einer eigenen Partition befand: Auch wenn dort noch so viel freier Speicher verfügbar war, lief der „interne Speicher“ des Gerätes gern einmal voll – und es kam zu der berüchtigten Fehlermeldung: „nicht genügend Speicher vorhanden um ...“. Zum Beispiel bei der Installation von Apps. Da dies insbesondere für den Anwender recht verwirrend ist („Aber ich habe doch noch so viel Platz frei?“), sollte sich dies mit Android 3.0 ändern. So berichtet The CommonsBlog⁶⁵, frei von mir übersetzt:

Android 3.0 hat dies geändert, indem es ermöglicht, internen und externen Speicher auf derselben Partition zu vereinen – in verschiedenen Verzeichnissen. Dies gab dem Anwender wesentlich mehr Flexibilität, da nun keine künstliche Unterscheidung mehr zwischen internem und externem Speicher mehr gemacht wurde. Gerätehersteller haben noch immer die Möglichkeit, getrennte Partitionen oder sogar Wechseldatenträger zu verwenden – tun dies jedoch üblicherweise nicht.

(Anmerkung: der Begriff „externer Speicher“ ist hier als Gegensatz zum „Gerätespeicher“ zu verstehen – bezeichnet also sowohl die externe, als auch die interne SD-Karte. Und der Umgang mit der neuen Möglichkeit ist seitens der Hersteller, anders als der Blog-Artikel darstellt, eher durchwachsen.)

Die nächste Änderung folgte auf dem Fuße: Der beliebte USB Massenspeicher wurde von MTP abgelöst. Das hat zwar den Vorteil, dass die SD-Karte bei Einbindung am Computer auf dem Androiden verfügbar bleibt (also keine Apps mehr abstürzen, weil sie plötzlich hier nicht mehr zugreifen können) – aber auch den Nachteil, dass auf nahezu allen Plattformen plötzlich zusätzliche Treiber



HTC Dream
Gerätespezifikation
bei GSMArena.COM



Google Nexus One
Gerätespezifikationen
bei GSMArena



Forensics Ferret
Blog: Android
Browser Forensics



The CommonsBlog:
The storage situation

61. http://www.gsmarena.com/htc_dream-2665.php
62. http://www.gsmarena.com/htc_google_nexus_one-3069.php
63. <https://forensicsferret.wordpress.com/2010/09/30/android-browser-forensics/>
64. http://www.gsmarena.com/samsung_i9000_galaxy_s-3115.php
65. <http://commonsware.com/blog/2014/04/08/storage-situation-external-storage.html>

nötig wurden. Außerdem kann die meiste Software zur Datenrettung auf diese Weise nicht von außen arbeiten, da sie keinen „direkten und ausschließlichen Zugriff“ mehr erhält. Leider steht der USB Massenspeicher i. d. R. auch nicht mehr optional zur Verfügung.



AndroidPIT:
microSD-Karten und
Android 4.4 KitKat

Dann kam Kitkat, und damit der „große Hammer“: Plötzlich war den „normalen Apps“ der Schreibzugriff auf die externe SD-Karte nicht mehr möglich⁶⁶ (mit Ausnahme eines festgelegten, App-spezifischen Verzeichnisses). Kein übergreifendes Bearbeiten von Dokumenten, kein Aktualisieren von ID3-Tags in MP3-Dateien mehr. Dateimanager, Music-Player, Office-Apps, und mehr büßten zumindest einen Großteil ihrer Funktionalität ein – wenn sie nicht gar insgesamt nahezu ihren Sinn verloren. Obwohl einige Hersteller gegensteuerten (und diese Änderung in ihren angepassten ROMs korrigierten), änderte sich von offizieller Seite hieran nichts: Kein Update zur Behebung erfolgte.



AndroidPIT: So löst
Android 5.0 Lollipop
das SD-Karten-
Problem

Ganz ungehört schienen die Proteste der Community jedoch nicht zu verhallen, denn mit Android 5.0 wurde das Problem schließlich auf andere Art gelöst⁶⁷, die allerdings Änderungen im Code der Apps erfordert. Statt den Schreibzugriff quasi „per se“ zu verbieten, kann der Anwender jetzt genau entscheiden, welche App überhaupt und auf welche Verzeichnisse der externen SD-Karte zugreifen darf. Android 6.0 soll überdies die Möglichkeit bieten, die externe Karte in ihr internes Pendant zu integrieren (allerdings verschlüsselt, sodass sie am PC nicht mehr ohne Weiteres lesbar ist). Wie sich diese in der Praxis bewährt, muss sich jedoch erst noch erweisen.

Google Cloud Backup

Sobald man sein Android-Gerät zum ersten Mal in Betrieb nimmt, und das Google-Konto initialisiert, erfolgt auch bereits die Frage: „Möchten Sie Ihre Daten bei Google sichern?“ Gemeint ist damit das *Google Cloud Backup*. Klingt zunächst wirklich praktisch: Das Backup läuft immer im Hintergrund – und nach einem Werksreset wird alles automatisch wieder hergestellt. Soweit die Theorie.



Google Ticket
#17354: Automatic
Restore rarely

Und wie sieht es in der Praxis aus? Da häufen sich nicht nur Beschwerden, dass das Restore oftmals nur teilweise oder gleich überhaupt nicht funktioniert (siehe Google Ticket #17354⁶⁸, eröffnet im Juni 2011 und noch immer nicht bearbeitet; im Januar 2014 jedoch schließlich von einem Projekt-Mitglied auf „falsches Forum“ gesetzt: Man fühlt sich also nicht zuständig). Es wird bei Weitem auch nicht alles gesichert. Denn damit die Daten einer App gesichert werden können, muss die App selbst das aktiv unterstützen – eine Sache, um die sich nicht unbedingt jeder Entwickler kümmert.

66. <http://www.androidpit.de/microsd-karten-android-4-4-kitkat>

67. <http://www.androidpit.de/so-loest-android-5-0-lollipop-das-sd-karten-problem>

Was genau gesichert wird, damit befasst sich u. a. ein [Artikel bei Stack Exchange](#)⁶⁹. Dabei wird aus der offiziellen Dokumentation zu Android Honeycomb (3.x) zitiert:

- Android Einstellungen, wie beispielsweise gespeicherte WLAN Netzwerke und Passworte (dummerweise [im Klartext](#)⁷⁰), Benutzerwörterbuch, etc.
- Die Einstellungen vieler Google-Apps, wie beispielsweise Browser-Lesezeichen
- Aus dem Playstore installierte Apps

Nicht unbedingt präzise Angaben: „etc“, „beispielsweise“. Was genau in der Google Cloud gespeichert ist, soll man im [Google Dashboard](#)⁷¹ sehen können. Ein kurzer Check mit meinem Kollegen Dan von Stack Exchange (der das *Google Cloud Backup* aktiviert hat) ergab jedoch, dass die Angaben dort alles andere als vollständig sind:



Stack Exchange:
What info does
Google backup?



Google Dashboard

68. <http://code.google.com/p/android/issues/detail?id=17354>
69. <http://android.stackexchange.com/q/15434/16575>
70. <http://www.heise.de/-1917386.html>
71. <https://www.google.com/settings/dashboard?hl=de>

The screenshot shows the Google Dashboard interface. On the left, there's a sidebar with 'Accounts' and 'Dashboard' sections. Under 'Accounts', 'Dashboard' is highlighted. The main area shows a list of devices connected to the Google Cloud:

Device	Last activity seen on	Registered date
asus Nexus 7	18 Jul 2013	1 Nov 2012
samsung Nexus 10	18 Jul 2013	17 Jul 2013
asus Transformer TF101	18 Jul 2013	28 Jul 2011

Below the devices, there's a section for 'Applications with backup on servers' which lists three items:

- Android Wallpaper: Backup date: 18 Jul 2013 21:58, Backup size: 115 B
- Android System Settings: Backup date: 17 Jul 2013 20:20, Backup size: 3.46 KB
- Android Market: Backup date: 18 Jul 2013 00:32, Backup size: 18 B

Im Google Dashboard soll man alles sehen können, was man in der Google Cloud gespeichert hat

Es darf sicher mit Recht bezweifelt werden, dass Dan auf seinem Nexus 10 keine einzige App installiert hat (die drei aufgeführten Apps gehören zum Android-System selbst).

Ein weiterer Punkt ist die Datensicherheit. Wie [Heise berichtet](#)⁷², werden die gespeicherten Daten auch dann nicht gelöscht, wenn der Nutzer dies explizit anweist: Tage nach der Löschung richtete man ein neues Gerät mit dem „alten“ Google-Account ein, worauf sich dieses sofort mit dem hausinternen WLAN verbinden konnte – ohne, dass die Zugangsdaten vom Anwender konfiguriert wurden. Sie kamen schlicht aus dem „gelöschten“ Backup. Und apropos WLAN Zugangspunkte: Schon gewusst, dass diese inklusive der zugehörigen Passwörter

Heise: Android und
die WLAN-
Passwörter: Google
löscht nicht

72. <http://www.heise.de/-1922971.html>

unverschlüsselt gespeichert werden⁷³? Nicht etwa nur auf dem Android-Gerät selbst, sondern auch im *Google Cloud Backup*. Darauf angesprochen, reagiert Google mit flauen Antworten. Heise fasst dies⁷⁴ etwa so zusammen:

Das entspricht also gerade mal dem Schutz herkömmlicher Mails. Google geht in keiner Weise darauf ein, dass und warum es keine Option gibt, diese äußerst sensiblen Geheimnisse mit einem speziellen Passwort zu sichern, das die Firma nicht kennt. [...]

Google erachtet eine besondere Sicherung der Passwörter anscheinend nicht als notwendig. Somit muss man davon ausgehen, dass im Rahmen der Zugriffe auf Google-Daten durch die NSA auch die Passwörter direkt in die Hände des US-Geheimdienstes gelangen. Dank der Ortsbestimmung über Netzwerkdienste (per Default aktiviert) ist der Standort des zugehörigen WLAN-Routers dort natürlich ebenfalls bekannt.

Doch es gibt nicht nur Negatives darüber zu berichten. Wenn es denn funktioniert (und mit neueren Android-Version scheinen auch die Chancen dafür zu steigen), werden verschiedene Daten (wie etwa die WLAN-Zugangspunkte) auch über verschiedene Geräte hinweg synchronisiert, sofern auf ihnen der gleiche Google-Account zum Einsatz kommt, was manchmal ganz praktisch sein kann (wenn es auch nicht unbedingt immer erwünscht ist; konfigurieren lässt es sich leider nicht). Wie die mit dem gleichen Account erstellten Backups über mehrere Geräte hinweg organisiert sind, damit befasst sich wiederum ein Artikel bei Stack Exchange⁷⁵.

Mit „Android M“ (also Version 6) soll sich dies jedoch grundlegend bessern. So berichtet z. B. Go2Android⁷⁶, dass Google hier „ein Auto-Backup für Apps und deren Daten“ implementiert, welches App-Entwickler *nicht* erst mit Extra-Code in ihren Apps aktivieren müssen. Die Sicherung soll dabei auf *Google Drive* erfolgen, wo hierfür expliziter Speicherplatz bereitgestellt wird – und zwar verschlüsselt. Wie gut das letztendlich funktioniert, und welche Daten dabei (nicht) gesichert werden, bleibt abzuwarten. Wer sich für weitere Details interessiert, dem sei dieser Artikel bei Phandroid⁷⁷ (in englischer Sprache, dafür mit Video) nahegelegt.

Fazit ist und bleibt jedoch: Auf dieses Backup allein sollte man sich nicht verlassen. Und sei es nur aus dem Grund, dass es nicht alle Daten erfasst.



Inside-Handy.DE:
WLAN-Passwörter
werden
unverschlüsselt
gespeichert



Heise: Google
reagiert auf Kritik an
Androids Passwort-
Speicherung



Stack Exchange:
Multiple devices



Go2Android:
Android M bringt
Auto-Backup für
Apps



Phandroid: Google
further details
Android

73. <http://www.inside-handy.de/news/28753>

74. <http://www.heise.de/-1920836>

75. <http://android.stackexchange.com/q/42245/16575>

76. <http://www.go2android.de/android-m-bringt-auto-backup-fuer-apps-152533/>

Vollständiges Backup ohne root und Cloud

Mit Android 4.0 wurde diese Möglichkeit still und heimlich in ADB-Daemon integriert, ohne dass man groß darüber sprach: Aktiviert man in den Entwickler-Einstellungen das USB-Debugging, wird auf dem Android-Gerät der ADB Daemon gestartet, sodass er u. a. auch von einem auf dem PC installierten ADB Client angesprochen werden kann. Das war schon in früheren Android-Versionen der Fall – ab Android 4.0 jedoch erhielt der Daemon erweiterte Rechte. So lässt sich nun über den Befehl `adb backup` die Erstellung eines Backups anfordern, und zwar sowohl für einzelne Apps einschließlich ihrer Daten (sofern der jeweilige App-Entwickler das nicht explizit unterbunden hat), oder auch für das gesamte System. Das Backup-Archiv wird dann an den Client übergeben, und kann so auf dem PC gespeichert werden. Auf letzterem sind dafür ein paar Voraussetzungen zu erfüllen, auf die ich im Kapitel ADB installieren näher eingehen – und das Ganze steht dann auch nur an der Kommandozeile zur Verfügung (beschrieben unter [ADB Backup & Restore](#)).



Da dies für Einsteiger etwas umständlich sein dürfte, haben sich glücklicherweise einige Entwickler gefunden, und für einfachere Möglichkeiten gesorgt. Um auf diese Funktionalitäten ohne viel Umstand zugreifen zu können, stellen sie grafische FrontEnds bereit – die auch in einem Artikel bei Stack Exchange⁷⁸ erwähnt werden. Eine davon möchte ich an dieser Stelle herausgreifen:

XDA-Developer omegavesko⁷⁹ hat ein einfaches Programm erstellt, das den Backup-Vorgang (und natürlich auch die Wiederherstellung) gerade für

77. <http://phandroid.com/2015/07/29/best-android-marshmallow-feature-auto-backup-for-apps-video/>

78. <http://android.stackexchange.com/q/28296/16575/>

79. <http://forum.xda-developers.com/member.php?u=4682694>

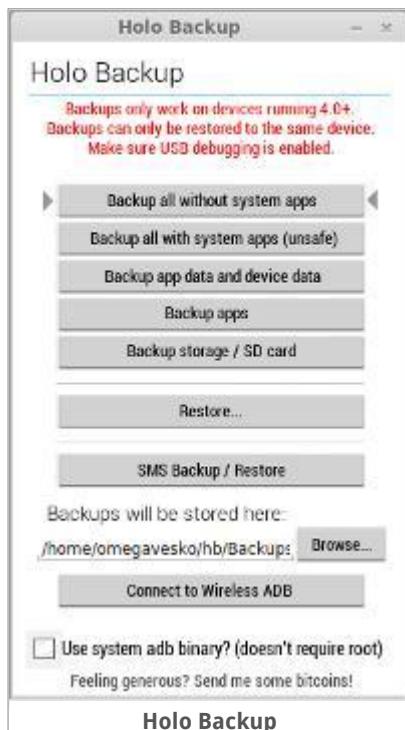


ASE: Full Backup for
non-rooted devices



XDA-Developer
omegavesko

Unerfahrene ermöglichen sollte. *Holo Backup* wurde dort [im Forum vorgestellt](#)⁸⁰, für Linux und Windows lässt es sich gratis [bei Github herunterladen](#)⁸¹.



Die Bedienung sollte eigentlich selbsterklärend sein – es ist ja alles beschriftet. Was beim Backup allerdings zu beachten ist: Man kann sich aus dem erstellten Backup nicht ohne Weiteres einzelne Dinge zur Wiederherstellung heraussuchen, es ist (mit Bordmitteln) immer ein Alles-oder-Nichts. Also ggf. besser zusätzlich zum „vollständigen Backup“ auch noch das eine oder andere kleinere Päckchen schnüren, etwa die wichtigsten Apps inklusive ihrer Daten jeweils separat.

Nachdem man in *Holo Backup* (oder auch von der Kommandozeile) ein Backup oder eine Wiederherstellung angestoßen hat, muss man diesen Vorgang noch auf dem Gerät selbst bestätigen. Dies dient als Sicherheits-Maßnahme, damit nicht etwa ein Unbefugter eben schnell ein Kabel anschließt, um sich die Daten herunter zu laden. Vor dem Bestätigen des Vorgangs lässt sich auch ein Passwort für die Verschlüsselung festlegen. Dieses Passwort

sollte man sich gut merken: Ein verschlüsseltes Backup lässt sich nur mit dem vergebenen Passwort wiederherstellen.



XDA Developers:
Holo Backup



Github: Holo Backup

80. <http://forum.xda-developers.com/showthread.php?p=36499906&nocache=1>

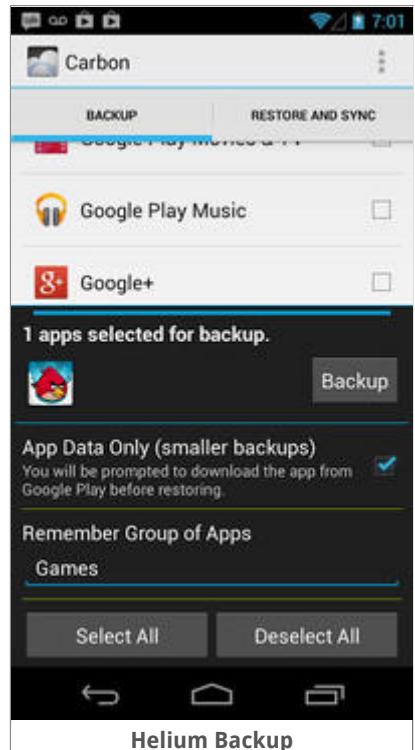
81. <https://github.com/omegavesko/HoloBackup>

Obwohl die Möglichkeit zur vollständigen Sicherung über ADB bereits ein großer Fortschritt gegenüber, ähm, gar keinem Backup ist, hat sie immer noch einen Haken: Sie benötigt einen PC. Unterwegs im Urlaub, oder auf einer Geschäftsreise, könnte sich das schwierig gestalten. Da dürfte es einige Herzen höher schlagen lassen zu lesen, dass [Koushik Dutta](#)⁸² aka Koush aka Mr. [ClockworkMod](#)⁸³ eine App erstellt hat, die ein vollständiges Backup ohne root direkt auf dem Androiden ermöglichen soll (siehe Bild!).

Vergleichbar mit dem beliebten *Titanium Backup* (welches root voraussetzt), sichert seine [Helium Backup](#)⁸⁴ (ehemals *Carbon Backup*) genannte App sowohl Apps als auch deren Daten – unabhängig von einem etwa angeschlossenen PC. Die Backups werden dabei auf der SD-Karte abgelegt.

Erwirbt man für knapp vier Euro die Lizenz zur Pro-Version, erhält man noch einige Extras: Keine Werbung mehr in der App, sowie eine Synchronisation von Apps zwischen mehreren Geräten ist damit möglich. Ein Scheduler für zeitgesteuerte Backups ist ebenfalls mit dabei, und Backups lassen sich auch bei diversen Cloud-Diensten (Dropbox, Box, Google Drive) ablegen.

Wer zum Start gern eine Kurzanleitung hätte, findet diese übrigens in einem [Blog-Beitrag bei AndroidPIT](#)⁸⁵.



Koushik Dutta



ClockworkMod Homepage



Helium Backup

AndroidPIT Blog:
Carbon Backup
Anleitung

Lösungen von Drittanbietern

Wer wirklich *alles* vollständig sichern will, kommt auch in Zeiten von Kitkat, Lollipop und Marshmallow an „root“ nicht vorbei – und nein, das wird sich auch mit Nougat, Ozelotohren, und Persipan kaum ändern. Die Killer-App hierfür

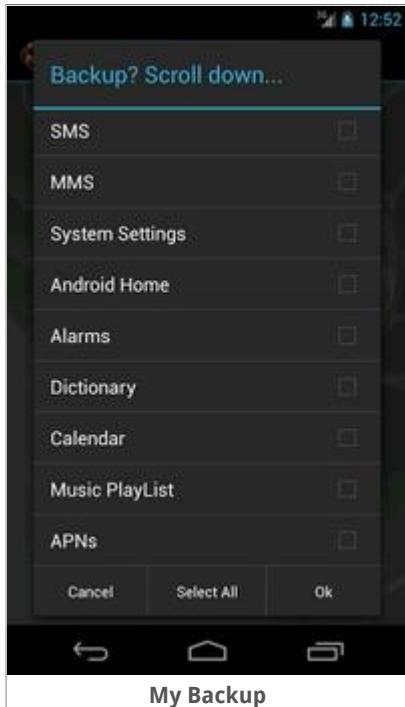
82. <http://www.koushikdutta.com/>

83. <http://www.clockworkmod.com/>

84. <https://play.google.com/store/apps/details?id=com.koushikdutta.backup>

85. <http://www.androidpit.de/carbon-backup-anleitung>

heißt [Titanium Backup](#)⁸⁶, und ich werde im Abschnitt [Fortgeschrittenes](#) (Kapitel [Vorinstallierte Apps entfernen](#)) näher auf sie eingehen – denn diese App setzt „root“ voraus.



Für nicht gerootete Androiden gibt es daher allenfalls Teil-Lösungen, die in der entsprechenden [App-Übersicht zum Thema](#)⁸⁷ aufgeführt sind. Wie bereits angemerkt: Verspricht eine App ein „vollständiges Backup“, ohne dass sie root benötigt, kann (und sollte) man mit Recht skeptisch sein, denn das ist (zumindest derzeit) technisch einfach nicht möglich: Android lässt eine „gewöhnliche App“ einfach nicht überall zugreifen.

Gute Kandidaten für nicht-gerootete Geräte wären beispielsweise [My Backup](#)⁸⁸ (siehe Screenshot; Kosten der Vollversion: Etwa vier Euro), sowie [Backup Your Mobile](#)⁸⁹. Beide bieten einen recht großen Funktionsumfang, erlauben lokale Sicherungen sowie Sicherungen in die Cloud, bringen einen Scheduler für automatische Backups mit, und bieten auf gerooteten Geräten zusätzliche Möglichkeiten.

Diese beiden Apps bieten somit eine grundlegende Lösung, die die meisten Sachen abdecken sollten. Ein wirklich vollständiges Backup leisten jedoch auch sie nur auf gerooteten Geräten.

In der eingangs genannten Übersicht finden sich jedoch auch Backup-Lösungen, die sich auf „Einzel-Probleme“ spezialisiert haben. Hilfreich, wenn es z. B. nur darum geht, eben einmal schnell eine Sicherung aller Kurznachrichten (etwa mit [SMS Backup & Restore](#)⁹⁰) oder der Anruflisten ([Call Logs Backup & Restore](#)⁹¹) zu erstellen.



Titanium Backup



Übersicht „Backup und Datensicherungen“



My Backup



Backup Your Mobile



SMS Backup & Restore

Call Logs Backup & Restore

-
- 86. <https://play.google.com/store/apps/details?id=com.keramidas.TitaniumBackup>
 - 87. http://android.izzysoft.de/applists/category/named/file_backup#group_141
 - 88. <https://play.google.com/store/apps/details?id=com.rerware.android.MyBackup>
 - 89. <https://play.google.com/store/apps/details?id=com.backupyourmobile>
 - 90. <https://play.google.com/store/apps/details?id=com.riteshsahu.SMSBackupRestore>
 - 91. <https://play.google.com/store/apps/details?id=com.riteshsahu.CallLogBackupRestore>

Zurücksetzen

Zurück auf LOS! Was ist los? Wo ist LOS? Und was, bitte, wohin zurücksetzen?

Den „älteren Semestern“ unter uns ist sicher die „Reset“-Taste am PC noch ein Begriff. So als Reißleine, Notbremse, letzte Ausflucht, wenn nichts mehr geht. Auch das ist eine Form von „Zurücksetzen“. Unter Android gibt es da mehrere Rücksetz-Möglichkeiten, mit zum Teil recht unterschiedlichen Auswirkungen. Und daher auch recht unterschiedlichen Verwendungszwecken.

Softreset

Dieses „weiche Zurücksetzen“ lässt sich am ehesten mit dem „Affengriff“ unter Windows (Strg-Alt-Delete) vergleichen. Nur dass die Tastenkombination, je nach Gerät, noch wesentlich abenteuerlicher ist. Bei HTC-Geräten z. B. üblicherweise das gleichzeitige Drücken der *Leiser*-Taste, der *Action*-Taste (Trackball), und des Einschaltknopfes. Möglichst ohne das Gerät dabei fallenzulassen ...

Bewirken soll das Ganze dann ein „sanftes“ Herunterfahren des Systems – üblicherweise wenn gar nichts anderes mehr funktioniert (sonst könnte man ja auch normal über das Menü abschalten).

Hardreset

In seltenen Fällen kann es vorkommen, dass das Android-Gerät komplett einfriert, und sich überhaupt nicht mehr bedienen lässt: Nichts reagiert mehr. Auch zu einem Softreset lässt es sich nicht mehr bewegen. Da hilft dann nur noch die harte Methode: Akku entfernen, bzw. bei Geräten mit fest verbautem Akku mit einem spitzen Gegenstand das Reset-Löchlein anpieksen. Gibt es weder einen herausnehmbaren Akku, noch ein Reset-Löchlein (was beispielsweise auf das *Huawei P6* zutrifft), hilft oftmals etwas Ausdauer beim Gedrückthalten des Power-Knopfes: Hält man dies für 15 bis 30 Sekunden durch, wird auch hier die Energiezufuhr unterbrochen. Was meist auch bei anderen Geräten (etwa meinem *LG Optimus 4X*) funktioniert, selbst wenn sie über einen herausnehmbaren Akku verfügen.

Factory-Reset

Hierbei handelt es sich um das „Zurücksetzen auf Werkseinstellungen“, was sich zum Beispiel über den genauso benannten Punkt unter *Einstellungen* > *Datenschutzeinstellungen* erreichen lässt. Dabei werden alle vom Anwender installierten Apps sowie sämtliche Einstellungen gelöscht – das Gerät ist somit wieder in einem „jungfräulichen“ Zustand (abgesehen von der internen/externen SD-Karte, die hier i. d. R. nicht angefasst wird).

Wird es anschließend wieder angeschaltet, muss mit der Einrichtung ganz am Anfang begonnen werden – als hätte man das Gerät gerade zum ersten Mal aus der Originalverpackung geholt. Das ist auch einer der Gründe, für den diese Funktionalität benötigt wird: Wenn das Gerät verkauft/verschenkt/weitergegeben werden soll. Natürlich möglichst, ohne private Datenspuren darauf zu hinterlassen.

Apropos Datenspuren: Die verbleiben oftmals trotz eines „Factory-Reset“. Zumindest etwas versiertere Anwender könnten gelöschte Daten wieder herstellen. Wer also ganz auf „Nummer sicher“ gehen möchte, nutzt eine App, die gründlich putzt. Die passenden Kandidaten finden sich natürlich wieder einmal in einer [Übersicht bei IzzyOnDroid](#)⁹². Wie das Ganze funktioniert? Statt die Daten einfach nur zu löschen, wird alles mit Zufalls-Daten überschrieben. Ein etwaiger Schnüffler findet dann nur noch Kauderwelsch.

Außerdem ist der „Factory-Reset“ auch noch ein „Last Resort“, wenn der Androide komplett verrückt spielt. Der Hersteller verlangt dies meist, um Probleme mit der Hardware ausschließen zu können: Löst ein *Werksreset* das Problem, ist die Hardware nämlich ganz offensichtlich unschuldig – es hat sich nur die Konfiguration verdreht.

Ein *Factory-Reset* besteht prinzipiell aus drei Schritten: Löschen der Cache-Partition, des Dalvik-Caches, sowie der (User-) Datenpartition. Ist der Anlass also ein „verrückt-spielender Androide“, kann man, bevor man zu diesem Schritt greift, auch eine "datenerhaltene Lösung" versuchen.

Die Cache-Partition lässt sich aus dem *Recovery-Menü* heraus löschen – unabhängig davon, ob das Gerät gerootet, oder ein Custom-Recovery installiert wurde. Für den Dalvik-Cache sieht das etwas anders aus:



Übersicht:
Datenbereinigung

92. http://android.izzysoft.de/applists/category/named/security_datawipe

Wipe des Dalvik-Cache

Dieser erzwingt die Neu-Übersetzung des Programmcodes aller installierten Apps (siehe [Dalvik](#) bei den [Begriffs-Erklärungen](#)). Das wäre ein Schritt, den man bei nicht behebbarem „ungewöhnlichen Verhalten“ des Systems noch durchführen kann, ohne das ganze Gerät komplett auf Werkseinstellungen zurückzusetzen. Voraussetzung dafür ist allerdings, dass das Gerät [gerootet](#) ist – die Hersteller haben diese Möglichkeit von Haus aus leider nicht vorgesehen.

Android-Apps sind in Java geschrieben, und Java ist bekanntlich Plattform-unabhängig. Vereinfacht ausgedrückt, ist das ein Zwischending zwischen einem Skript wie einer Batch-Datei oder einem PHP-Skript, und einem kompilierten Programm. Vor der eigentlichen Ausführung muss da also noch eine Übersetzung in Maschinensprache stattfinden, die möglichst nah am verwendeten System ist. Bei Java nennt man dies „Byte-Code“. Um die schmalen Ressourcen von mobilen Android-Geräten noch schonender zu nutzen, geht man bei Dalvik-VMs (so nennt sich die spezielle „Java-Variante“ unter Android) noch einen Schritt weiter, und nutzt zusätzliche Optimierungs-Möglichkeiten.

Damit dies nun nicht bei jedem Aufruf einer App geschehen muss (das wäre unerträglich langsam), macht Android das unmittelbar nach der Installation einer App – und legt den optimierten „Byte-Code“ im sogenannten [Dalvik-Cache](#) ab. Wird dieser gelöscht, erzwingt dies lediglich eine Neu-Übersetzung (wie bereits beschrieben) – die Anwendungsdaten und Einstellungen bleiben jedoch vollständig erhalten.

Durchführen lässt sich der Wipe aus einer Custom-Recovery (das Standard-Recovery-Menü sieht einen separaten Punkt hierfür seltsamerweise nicht vor) – oder alternativ mit einer passenden „root-App“ wie beispielsweise [ROM Toolbox](#)⁹³.

Dieser Schritt ist definitiv zu empfehlen, wenn ein (neues/anderes) [Custom-ROM](#) eingespielt werden soll. Bei „offiziellen Updates“ sollte sich der Hersteller darum kümmern, sofern dies nötig ist.

Ab Android 5.0 wurde Dalvik endgültig durch [ART](#) abgelöst. Das Verzeichnis für den Dalvik-Cache ist dabei geblieben, ein neues für den ART-Cache allerdings hinzugekommen. Dies gilt es bei Obigem natürlich zu berücksichtigen (aktuelle Custom Recoveries tun dies).



ROM Toolbox

93. <https://play.google.com/store/apps/details?id=com.jrummy.liberty.toolbox>

Safe-Mode

Etwas Ähnliches ist vielen aus der Windows-Welt als „Abgesicherter Modus“ bekannt: Bootet man in diesen, werden beim Systemstart alle Anwender-Apps ignoriert. Bei Android-Geräten ist dies beispielsweise hilfreich, wenn das Gerät bei einem normalen Start [in einer Force-Close-Schleife festhängt](#): Das System fährt hoch, eine „verkorkste App“ wird geladen, bringt das System zum Absturz – und dann geht das Ganze von vorn los.

Im „Safe-Mode“ hingegen wird die „verkorkste App“ (sofern es sich nicht um eine vorinstallierte App handelt) vom System ignoriert, sodass der Bootvorgang erfolgreich abgeschlossen werden kann. Der Anwender kann nun zu *Einstellungen > Apps* navigieren, und dem Übeltäter den Garaus machen: Entweder gleich deinstallieren, oder es zunächst mit dem Löschen von Cache sowie ggf. Daten versuchen.

Wie man in den Safe-Mode gelangt (und auch wie man wieder herauskommt), ist [bei den Fragen aus Alltag und Praxis](#) erklärt. Weitere Details sowie Links zu zusätzlichen Informationen rund um den Safe-Mode finden sich u. a. [bei Stack Exchange](#)⁹⁴.



Android.SE safe-mode tag-wiki

94. <http://android.stackexchange.com/tags/safe-mode/info>

Von Taskkillern und anderen bösen Buben

Oh ja, ich höre schon die Schreie: „Taskkiller gehören verboten! Android kann das selbst!“ Und gleich aus der Gegenrichtung: „Taskkiller muss man haben, mein System läuft jetzt viel flüssiger!“

Wer hat nun Recht? Beide. Keiner. Denn hier gibt es kein einfaches Schwarz und Weiß. Sicher ist jedoch: Wer nicht weiß, wie man eine Spritze setzt, sollte sich nicht als Arzt ausgeben – das kann sonst gehörig in die Hose gehen.

Man sollte also schon genau wissen, was man tut – und Taskkiller, Autostart-Helfer, & Co. können sehr nützlich sein. Wer dies nicht weiß, lässt besser die Finger weg!

Kurz zusammengefasst (ausführliche Erläuterungen finden sich [in einem Forums-Thread⁹⁵](#)): Hier handelt es sich um ein sehr kontrovers diskutiertes Thema. Worin mir allerdings (fast) jeder zustimmen dürfte: Eingriffe ins System setzen eine gute Kenntnis desselben voraus.

Es ist korrekt, dass Android sich um die Speicherverwaltung selber kümmert. Dennoch haben Task-Manager / Task-Killer durchaus ihre Berechtigung – solange man weiß, was man da tut:

- Falsch: „ich will den Speicher freiräumen“. Dafür ist der „OOM Killer“ (direkt im Android-System integriert, näheres dazu im [Tuning-Kapitel](#)) zuständig.
- Richtig: „eine App hat sich aufgehängt, und blockiert [irgendwas]“. Hier ist der Task-Killer angesagt – weil bis der „OOM-Killer“ hier zuschlagen würde ... Und ein Reboot ist nicht gerade die wünschenswerte Alternative.
- Richtig: „eine App läuft Amok“ (Panik-Mode: Man erwischt gerade eine App dabei, wie sie alle persönlichen Daten inkl. Nackt-Fotos auf eine berüchtigte Website hoch lädt ...). Oh ja: Abschießen! Oder gleich abschalten. Weil: Bis der OOM-Killer ... genau, da ist es dann eh zu spät.

95. <http://www.androidpit.de/de/android/forum/thread/409249/Wird-das-RAM-knapp-Von-Taskkillern-und-anderen-boesen-Buben>

Datenaustausch mit dem PC

Bis zu Android 4.0 war dies recht einfach möglich: USB-Kabel angeschlossen, und der Androide wurde vom Computer automatisch als USB Massenspeicher erkannt. Mit Android 4.0/4.1 änderte sich dies allerdings, da auf Android-Seite der USB Massenspeicher-Modus durch MTP abgelöst wurde. Dieser Modus bietet zwar einige Vorteile⁹⁶, jedoch auch einen gravierenden Nachteil: *Unter Linux, Windows und OS X gibt es oft Probleme beim Zugriff auf MTP-Geräte.* Und das Ganze muss oftmals erst händisch eingerichtet werden (für Linux am Beispiel Ubuntu hier erklärt⁹⁷). Unter Windows braucht es häufig wieder einen „gerätespezifischen Treiber“, den man sich zunächst auf der Herstellerseite besorgen muss; manchmal klappt es aber auch automatisch. Bei Problemen hilft dieser Artikel⁹⁸ hoffentlich weiter. Und auch Mac-User brauchen dafür spezielle Software, und seien dafür auf diesen Artikel⁹⁹ verwiesen. Ist diese Hürde genommen, sollte der Androide nach Anschließen des USB-Kabels in den jeweiligen Datei-Managern sichtbar sein.

Alternativ und ebenfalls über das USB-Kabel funktioniert der Datenaustausch via ADB, was im Kapitel Dateien kopieren mit ADB beschrieben ist. Unter Linux lässt sich mittels ADB sogar das Android-Dateisystem am Rechner einbinden.

Doch zum einen ist diese Art Kabel-Verbindung zu umständlich, zum zweiten ist laut Murphy genau dann kein Kabel zur Hand, wenn man es bräuchte, und zum dritten ist das ja sowas von uncool und unzeitgemäß ... Kurzum: Oftmals geht es kabellos bequemer vonstatten – wobei man auch das natürlich zunächst einmal einrichten muss. Die Möglichkeiten reichen hier von diversen Datei-Servern¹⁰⁰ wie FTP und WebDAV, bis hin zu hübschen grafischen WiFi-Datei-Managern à la WebSharing¹⁰¹. Auch „echte“ Windows-Freigaben sind, z. B. mit Samba



Wikipedia: Vorteile von MTP



Ubuntu Wiki: MTP



MacLife:
Datenabgleich
zwischen Android
und Mac – So
funktioniert es



Übersicht: Server



WebSharing

-
96. https://de.wikipedia.org/wiki/Media_Transfer_Protocol#Vorteile
 97. <http://wiki.ubuntuusers.de/mtp>
 98. <http://www.media-web.de/de/knowledgebase/44-iphone/195-android-usb-pc-verbindung-funktioniert-nicht-mtp-treiber.html>
 99. <http://www.maclife.de/tipps-tricks/mac-os-x/os-x-109-mavericks/datenabgleich-zwischen-android-und-mac-so-funktioniert-es>
 100. http://android.izzysoft.de/applists/category/named/network_server



Samba Server

Übersicht:
Dateimanager

MyPhoneExplorer

Filesharing¹⁰² oder **Samba Server**¹⁰³ möglich. Diverse **Datei-Manager**¹⁰⁴ bieten darüber hinaus die Möglichkeit, vom Androiden aus über das Netzwerk auf Freigaben des Rechners zuzugreifen.

Wem das nicht weit genug geht: Man kann auch **das Android-Gerät vom PC aus verwalten**, und zwar nahezu vollständig: Anruflisten, SMS, Kontakte etc. einsehen, Anrufe initialisieren, SMS schreiben, Dateien verwalten (und mit dem PC austauschen), Apps organisieren, Fotos anschauen, Musik und Videos wiedergeben, und mehr ist auf diese Weise möglich – oftmals weitaus bequemer, als beispielsweise auf dem Bildschirm des Smartphones.

Die beliebteste Lösung hierfür heißt **MyPhoneExplorer**¹⁰⁵ – benötigt aber auf PC-Seite ein Windows-Programm, und ist somit nur für Windows verfügbar. Dafür lässt sie sich aber auch nutzen, um Kontakte sowie Kalender mit Outlook abzugleichen, sowie verschiedene andere Daten vom Androiden auf den PC zu sichern. Zusätzlich zum genannten Windows-Programm ist für die Nutzung von **MyPhoneExplorer** die Installation einer App auf dem Androiden nötig.

-
101. <https://play.google.com/store/apps/details?id=nextapp.websharing.r1>
 102. <https://play.google.com/store/apps/details?id=com.funkyfresh.samba>
 103. <https://play.google.com/store/apps/details?id=com.icecoldapps.sambaserver>
 104. http://android.izzysoft.de/applists/category/named/file_fileman
 105. <https://play.google.com/store/apps/details?id=com.fjsoft.myphoneexplorer.client>



Mit *MyPhoneExplorer* und auch mit *AirDroid* kann man seinen Androiden vom PC aus verwalten

Darüber hinaus gibt es jedoch auch reine Browser-basierte Lösungen, die lediglich eine App auf dem Androiden voraussetzen – und somit unabhängig vom auf dem PC installierten Betriebssystem verwendbar sind. Für den „klassischen Anwender“ empfiehlt sich beispielsweise [AirDroid](#)¹⁰⁶ (siehe Screenshot). Ist jemand technisch versierter, und möchte die verfügbare Lösung selbst auf die eigenen Bedürfnisse hin erweitern, lohnt sich ein Blick auf den [PAW Server](#)¹⁰⁷. Beide Apps hatte ich in älteren Versionen dieses Buches kurz vorgestellt; das



AirDroid



PAW Server

106. <https://play.google.com/store/apps/details?id=com.sand.airdroid>

107. <https://play.google.com/store/apps/details?id=de.fun2code.android.pawserver>

entsprechende Kapitel findet sich jetzt [auf meiner Website](#)¹⁰⁸. Ebenso natürlich eine passende [Übersicht mit weiteren Kandidaten](#)¹⁰⁹.



IzzyOnDroid: Das
Android-Gerät vom
PC aus verwalten



Übersicht: Den
Androiden vom PC
aus verwalten

108. http://android.izzysoft.de/books.php?topic=apps_remotemaint

109. http://android.izzysoft.de/applists/category/named/various_remotemanagement

Datenaustausch zwischen Android-Geräten

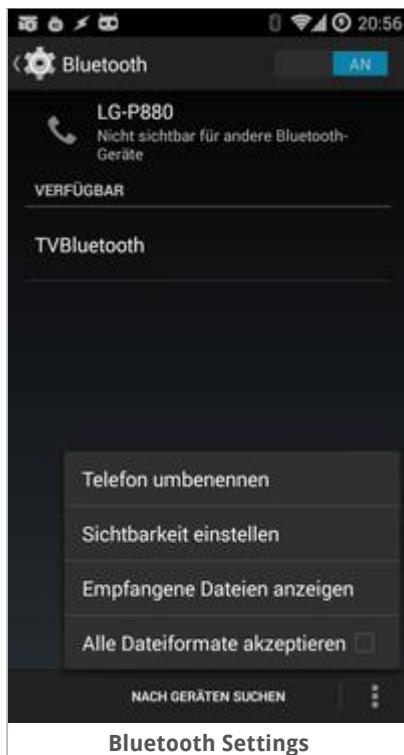
Sicher greifen hier einige der bereits vorgestellten Möglichkeiten. Und natürlich kann man Daten auch per Anhang an Mails/MMS oder über entsprechende Speicherplätze in der Cloud (wie z. B. Dropbox) austauschen. Aber das kann ja wohl nicht alles sein! Irgendwie muss sich doch auch ein direkter Datenaustausch zwischen zwei Android-Geräten erreichen lassen? Einige der dafür vorhandenen Varianten wollen wir im Folgenden betrachten:

Bluetooth

Den Bluetooth-Standard gibt es bereits seit den 1990er Jahren. Verwendet wird er u. a. für schnurlose Tastaturen, Kopfhörer, Headsets, GPS-Mäuse, und mehr – aber auch zur Übertragung von Daten über kurze Distanzen. Und so konnten schon recht früh Visitenkarten und Dateien von einem Gerät zu einem anderen gesendet werden. Besonders schnell geht das Ganze nicht vonstatten: Anfangs waren es rund 100 „theoretische“ Kilobit (also in der Praxis etwa 10 Kilobyte pro Sekunde), mit Bluetooth 2.0 ging es dann bereits mit 2.1 Megabit (also rund 250 Kilobyte pro Sekunde) vonstatten. Für eine 100 Megabyte Video-Datei darf man damit noch immer eine Übertragungsdauer von ca. 400 Sekunden (also knapp 7 Minuten), für ein 5 Megabyte großes Foto noch 20 Sekunden veranschlagen. Alles andere also als ein Full-Speed-Highway. Dennoch für kleinere Datenmengen (Visitenkarten, URLs) völlig ausreichend – und für ein „Schnell Mal eben“ durchaus in Kauf zu nehmen.



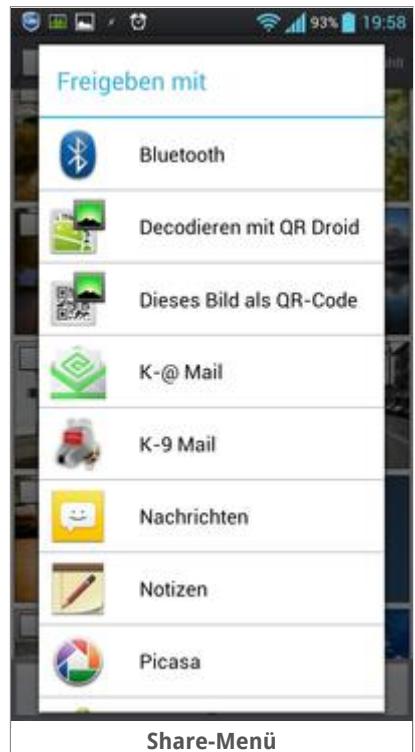
Und wie geht das „Schnell Mal eben“? Über das so genannte „Share-Menü“. Das findet sich an verschiedenen Stellen, und ist meist über ein Icon zu erreichen, das wie ein durch Verbinden dreier Punkte entstandenes, auf der Seite liegendes V aussieht. Tippt man dieses an, gelangt man zu einem Bildschirm, wie er rechts dargestellt ist (eventuell, nachdem man zuvor noch die zu verteilenden Objekte ausgewählt hat). Schwer zu übersehen, gleich an aller oberster Position: Bluetooth.



Das muss nun natürlich auch auf beiden beteiligten Geräten aktiviert sein, sonst kann kein Kontakt zustande kommen. Per Default

„versteckt“ sich ein Gerät mit aktiviertem Bluetooth dennoch, um sich gegen Angreifer zu schützen. Also gilt es zumindest für den Empfänger, sein Gerät sichtbar zu machen, und ihm einen Namen zu geben. Dies geschieht unter *Einstellungen > Bluetooth*, wie im linken Bild ersichtlich. In den meisten Fällen bleibt das Gerät so für eine begrenzte Zeit „sichtbar“, um anschließend automatisch zu verschwinden – nur von der Anzeige anderer Geräte, versteht sich, nicht vom Tisch.

Das muss aber auch einfacher gehen, dachte man sich bei Google. Und spendierte Android 4.0 ein neues Feature. Selbiges hört auf den Namen:



Android Beam

„Beam me up, Scotty!“ ist sicher das Erste, was Star Trek-Fans dabei in den Sinn kommt. Allerdings werden bei *Android Beam* nicht etwa materielle Dinge, sondern nur Inhalte transportiert. Falsch eingestellt ist der „Transporter“ dabei sicherheitshalber ebenfalls, damit das Original erhalten bleibt. So lassen sich Kontakte, Websites, Apps, Maps, Routenplanungen und Youtube-Videos auf ganz einfache Weise von einem Smartphone zum anderen schicken. Dazu holt man sich den zu verschickenden Inhalt auf den Bildschirm, und hält Sender- und Empfängergerät mit dem Rücken aneinander. Ein Signalton sowie kurzes Vibrieren kündigen sodann von der Bereitschaft: „Ready to beam up!“ Kurzes Antippen des nun verkleinert dargestellten Inhalts auf dem Sender-Gerät vollzieht schließlich den Transfer.

Welche Magie steckt dahinter? Auch wenn ich diesen Text an einem 1. April schreibe, handelt es sich definitiv nicht um einen Scherz. Trotzdem dürften sich einige beim Lesen dieser Zeilen zu früh gefreut haben. Denn *Android Beam* setzt auf die so genannte Near Field Communication ([NFC](#)) auf, die leider nicht von jedem Gerät unterstützt wird. Vorausgesetzt wird nämlich ein kleines Stück Hardware, der NFC-Chip. Dieser ist zumeist im Akku bzw. der Gehäuse-Rückwand des Android-Gerätes verbaut – was auch erklärt, warum die Geräte mit dem Rücken aneinander gehalten werden müssen: Die Reichweite dieser Chips beträgt gerade einmal 4 Zentimeter.

Ein großes Rätsel dürfte hingegen bleiben, warum Google für die Datenübertragung dabei ausschließlich auf NFC setzt – ist doch damit die Übertragungsrate auf 424 Kilobit beschränkt (und somit langsamer als selbst die Bluetooth-Implementierung, siehe voriges Kapitel). So werden „größere Inhalte“ wie Youtube-Videos oder vollständige Webseiten dann auch nicht direkt übertragen, sondern lediglich deren Link (wer sich für weitere Details interessiert, kann einen Blick auf einen [passenden Artikel bei Golem](#)¹¹⁰ werfen). Samsung hat hier einen Schritt weiter gedacht: Bei deren „S-Beam“ erfolgt der Verbindungsauflauf zwar ebenfalls über NFC. Für die eigentliche Daten-Übertragung verwendet man jedoch ...



Golem:
Datenaustausch per
NFC

110. <http://www.golem.de/1112/88621.html>

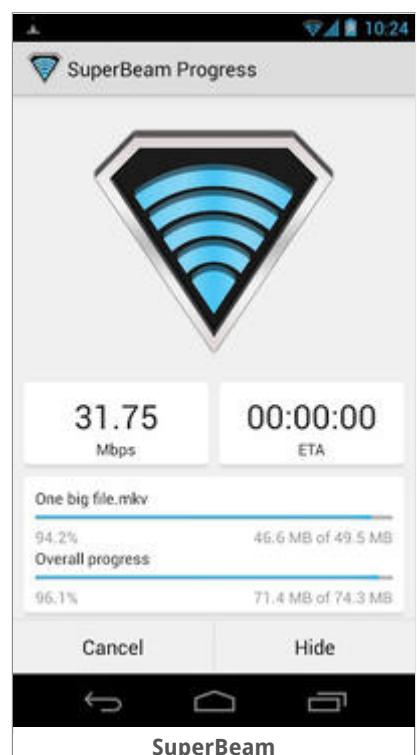
Wi-Fi Direct

Hierbei handelt es sich um einen WLAN-Standard, der ganz ohne Router auskommt. Da Wi-Fi Direct fähige Geräte von der [Wi-Fi Alliance](#)¹¹¹ zertifiziert werden, sollte man eigentlich davon ausgehen, dass das Ganze herstellerübergreifend funktioniert. Leider ist das in der Praxis nicht immer so, wie auch ein [Artikel bei Go2Android](#)¹¹² bescheinigt. Neben einem Sich-nicht-Finden oder Daten nicht übertragen können (was zwar ärgerlich, aber immer noch recht harmlos ist), weiß man dort von Folgendem zu berichten:

Doch den Vogel schießt wohl der Test zwischen dem LG Optimus G und meinem Galaxy Note ab. Auch hier ein erfolgreiches finden und verbinden, doch sobald wir ein Bild an das Samsung Gerät schicken wollten, machte das Galaxy Note einen reproduzierbaren Softreset. Welchen rein koreanischen Disput man hier auf Softwareebene aus trägt, bleibt wohl ein gut gehütetes Geheimnis der Asiaten.

Der gleiche Artikel weist aber auch auf mögliche Abhilfe hin: Wenn die Hardware etwas unterstützt, und die Hersteller lediglich bei der Implementierung ihrer Software aneinander vorbei gearbeitet haben, sollte eine passende App das Problem doch umgehen können?

Eine solche App wird auch gleich benannt: [SuperBeam](#)¹¹³. Der Name lässt richtig vermuten, dass damit *Android Beam* imitiert wird: Zur Initiierung der Übertragung wird auch hier auf den NFC-Chip gesetzt. Doch selbst wer keinen solchen in seinem Gerät hat, muss auf diese App nicht verzichten. Alternativ erzeugt sie auf dem Sender-Gerät einen Barcode, der von der gleichen App auf dem Empfänger-Gerät einfach abgescannt wird. Eine Ausweich-Möglichkeit wäre etwa die App [WiFiShare](#)¹¹⁴, die sich, wie bereits bei Bluetooth gezeigt, in das so genannte „Share Menü“ einklinkt.



111. http://de.wikipedia.org/wiki/Wi-Fi_Alliance

112. <http://www.go2android.de/was-ist-eigentlich-aus-wi-fi-direct-geworden/>

113. <https://play.google.com/store/apps/details?id=com.majedev.superbeam>

114. <https://play.google.com/store/apps/details?id=com.iiitd.muc.wifishare>

Den Funktionsumfang mit Apps erweitern

Bereits auf dem frisch erworbenen, nagelneuen Android-Gerät sind zahlreiche Apps vorinstalliert. Oftmals sogar zu viele, die lediglich Ressourcen belegen – aber letztendlich vom Anwender gar nicht genutzt werden (siehe [Bloatware](#)). Es fehlen jedoch dafür häufig Dinge, die wirklich gebraucht würden (oder die für einen Zweck vorgesehenen Apps sind in ihrem Funktionsumfang unzureichend). Zum Glück wissen wir Abhilfe: Bereits im Kapitel [Anwendungen verwalten](#) haben wir schließlich gelernt, dass wir zusätzliche Apps installieren können.

Wie viele gibt es da gerade? Ich schreibe hier besser gar keine Zahl – die würde ohnehin bereits nicht mehr stimmen, kaum dass ich auf „Speichern“ drücke. Einigen wir uns auf „echt viele“. So viele, dass man den Durchblick schnell verliert.

Wer nun meint, ich würde jetzt hier jede Menge Apps vorstellen: Weit gefehlt. Dann müsste ich wahrscheinlich im Wochen- oder doch zumindest Monatsrhythmus eine neue Version dieses eBooks veröffentlichen. Das würde zum einen das Buch zu sehr „aufblähen“ – und ist mir zum anderen zu umständlich. Dem Leser wahrscheinlich auch, weil der dann immer schauen müsste, ob er auch die aktuellste Version hat – und wenn nicht, an welcher Stelle sich denn nun was geändert hat. Daher habe ich eine bessere Lösung:

Ja, genau – es ist wieder einmal *IzzyOnDroid*. Denn hier pflege ich u. a. eine meine thematisch sortierten [App-Übersichten](#)¹¹⁵. Jede dieser Übersichten widmet sich einem bestimmten Thema, und listet eine Auswahl dazu verfügbarer Apps auf – ergänzt mit Links zu den Markets, bei denen sie erhältlich sind (Google Play, F-Droid, und Aptoide werden dabei berücksichtigt), sowie zu Reviews, Testberichten, Video-Demos, und was sonst noch an relevanten Informationen zur jeweiligen App verfügbar ist. Hinzu kommen häufig auch weitere Links zu themen-relevanten Artikeln in Foren und News. Damit sollte ein Vergleich ohne größere eigene Recherchen möglich sein. Hin und wieder stelle ich auch eine oder mehrere Apps in einem [Artikel](#)¹¹⁶ etwas ausführlicher vor. Um jedoch zumindest einen Eindruck zu vermitteln, was so alles möglich ist, möchte ich einige Themen exemplarisch herausgreifen und hier benennen. Die



IzzyOnDroid: App-Übersichten



Artikel bei IzzyOnDroid

115. <http://android.izzysoft.de/applists>

116. <http://android.izzysoft.de/articles>

umfangreicheren Vorstellungen aus früheren Ausgaben dieses Buches finden sich wiederum [auf meiner Website](#)¹¹⁷.



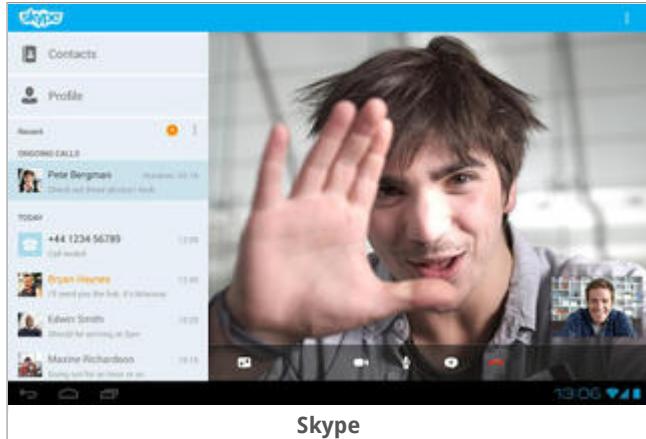
IzzyOnAndroid: Das inoffizielle Android-Handbuch



Übersicht: Kontakt-Verwaltung und Dialer

Telefonieren

Wie? Ja, auch das geht auf unseren Smartphones per App. Für „normale“ Telefonate ist die vorinstallierte Telefon-App dafür auch genügend komfortabel (wer das anders sieht, werfe bitte einen Blick in meine Übersicht [Kontakt-Verwaltung und Dialer](#)¹¹⁸). Aber das ist in Sachen Telefonie ja noch lange nicht das Ende der Fahnenstange.



Neben der „herkömmlichen Telefonie“ gibt es heutzutage natürlich auch VoIP (Voice-over-IP, also Internet-Telefonie). Die bekannteste Lösung in diesem Feld ist sicherlich [Skype](#)¹¹⁹, das für nahezu jedes System verfügbar ist. So auch für Android. Wer über eine Front-Kamera verfügt, kann auch Video-Telefonate führen (siehe Bild). Vorausgesetzt, das

Gegenüber hat auch diese Möglichkeit, sehen sich beide Gesprächsteilnehmer: Das große Bild zeigt das Gegenüber, man selbst wird als kleines Bild eingeblendet.

Anrufe von Skype-Teilnehmern untereinander (ebenso wie ihre Chats) sind kostenlos, auch Dateien können zwischen Clients gratis übertragen werden (im Mobilfunknetz können jedoch Kosten für die übertragenen Daten anfallen). Für Anrufe in „andere Netze“ benötigt man allerdings ein Guthaben. Die Preise bewegen sich hierbei in üblichen VoIP-Regionen.

Setzt der VoIP-Dienstleister hingegen auf das offene [SIP](#)-Protokoll, können dessen Android-Kunden auf den seit [Gingerbread](#) im System integrierten SIP-Client zurückgreifen, dessen Konfiguration im Abschnitt [Internet-Telefonie](#) beschrieben wurde. Dies bietet den Vorteil, dass es komplett ins System integriert ist. Vor jedem Anruf kann man sich also fragen lassen, ob dieser über VoIP



Skype

117. <http://android.izzysoft.de/books.php?topic=anwender>

118. http://android.izzysoft.de/applists/category/named/office_contacts_dialer

119. <https://play.google.com/store/apps/details?id=com.skype.raider>

geführt – und auch festlegen, dass bei verfügbarer WLAN-Verbindung auf eingehende SIP-Anrufe geachtet werden soll.

Alternativen finden sich natürlich im *Play Store*, oder in der bereits genannten [Übersicht](#)¹²⁰ – ähnliche, aber auch ganz andere...

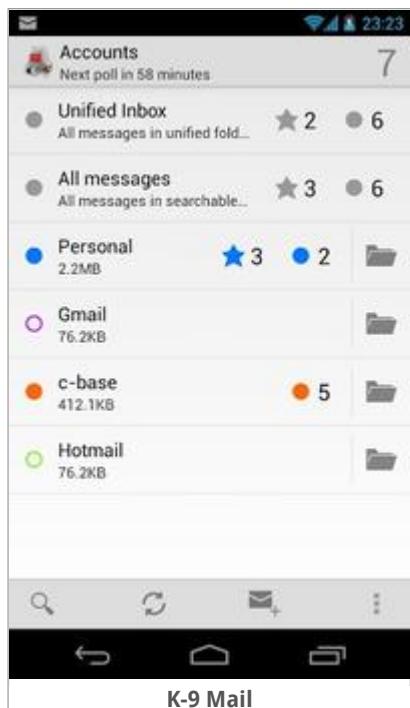
Telefon-Widgets

Über *Shortcuts* und *Widgets* haben wir ja bereits im Zusammenhang mit dem [Home-Screen](#) gesprochen. Was aber sollen jetzt bitte „Telefon-Widgets“ sein?

Klar, da hat Izzy wieder mal einen Begriff konstruiert. Da gibt es also Icons auf dem Home-Screen, die telefonieren können? Ja, so ungefähr. Unter [Apps organisieren](#) hatte ich ja bereits den *Folder Organizer* genannt, der die Permission zum Anrufen verlangt – genau hierfür: Es lassen sich damit nämlich auch *Shortcuts* zu Kontakten anlegen. Gibt es also Leute, die man öfter anruft, muss man deren „Kontakte“ nicht erst lange im Adressbuch suchen – sondern legt gleich eine passende Verknüpfung auf dem Home-Screen ab. Manche Launcher bieten ebenfalls derartige Widgets an – was dann auch bei diesen die geforderte Anruf-Berechtigung erklären dürfte.

120. http://android.izzysoft.de/applists/category/named/office_contacts_dialer

Mail



Bei den Kurznachrichten (SMS/MMS) gilt für die meisten Anwender das Gleiche wie bei der Telefon-App: Die vorinstallierte Anwendung genügt. Andere nutzen gleich gar keine SMS oder MMS mehr, und greifen stattdessen nach Messengern à la *WhatsApp* oder *Threema*. Wer dennoch Alternativen sucht, wird (wie immer) in der [passenden Übersicht](#)¹²¹ fündig.

Doch auch die EMail ist aus unserem Alltag nicht mehr wegzudenken. Das hat man bei Android ebenfalls erkannt, und liefert eine passende App gleich mit. Allerdings stößt man bei dieser hinsichtlich Komfort, Funktionsumfang, und leider auch hin und wieder der Sicherheit oft schnell an Grenzen. Und außerdem lässt sich die Frage, um welche App es sich hier handelt, allenfalls herstellerspezifisch beantworten. Damit keiner meiner Leser „außen vor bleibt“, halten wir also besser nach einer guten Alternative Ausschau.

Eine der bekanntesten und beliebtesten Apps hierfür ist [K-9 Mail](#)¹²². Unterstützt mehrere Accounts, auf Wunsch auch mit „gemeinsamer Inbox“ (quasi als „Zusammenfassung“ der einzelnen Eingangs-Ordner; das einer Mail zugehörige Postfach erkennt man dort an einer farblichen Markierung – wobei sich die Farben dafür natürlich wählen lassen). Als Protokolle werden POP3, IMAP4, und Exchange unterstützt; auch das Google Mail Konto lässt sich auf diese Weise einbinden. „IMAP-Push“ steht ebenfalls zur Verfügung (d. h. der Mail-Server gibt der App Bescheid, wenn neue Mail da ist).

Natürlich ist dies nur ein Auszug aus dem Funktionsumfang – da könnte man noch weit mehr aufzählen. Zum Beispiel die Unterstützung für PGP (signieren/verschlüsseln von Mails), konfigurierbare Benachrichtigungen in der „[Notification Area](#)“ sowie per Audio, Shortcuts für den Home-Screen, Signatur-Unterstützung ...

Für weitere Alternativen empfehle ich einen Blick in die [passende Übersicht](#)¹²³.



Übersicht:
Kurznachrichten



K-9 Mail



Übersicht EMail-
Apps

121. http://android.izzysoft.de/applists/category/named/office_contacts_shortmessage

122. <https://play.google.com/store/apps/details?id=com.fsck.k9>

123. http://android.izzysoft.de/applists/category/named/network_emailapps



Übersicht eBook-Reader



Übersicht RSS-Reader



Übersicht:
Nachschlagen und
Moon+ Reader



Izzys eBook-Bibliothek



ColorDict

Lektüre

eBook-Reader wie Amazons *Kindle* sind den meisten bekannt. Aber warum dafür noch ein separates Gerät mitnehmen, wenn unser Androide dieses Feld ebenso beherrscht? Allenfalls die deutlich längere Akku-Laufzeit bei Geräten mit eInk-Display könnte da ein Grund sein. Android bietet zum Thema Lektüre eine Vielzahl an Apps, die von [eBook-Readern](#)¹²⁴ über [RSS-Reader](#)¹²⁵ bis hin zu [Nachschlagewerken](#)¹²⁶ reicht.

Bei den eBook-Readern ist [Moon+ Reader](#)¹²⁷ (in der Pro-Version für ca. 4 Euro – es gibt auch eine [werbe-finanzierte Gratis-Version](#)¹²⁸) mein klarer Favorit. Sofern man auf DRM-behaftetes Material verzichten kann, kann ihm keiner das Wasser reichen:

- Zugriff auf zahlreiche Online-Bibliotheken direkt aus der App (vorkonfigurierte wie z. B. [Izzys Bibliothek](#)¹²⁹ mit über 7.000 gratis verfügbaren Büchern in deutscher Sprache, und eigene)
- Formate: txt, html, epub, mobi, pdf, cbr, umd, fb2, zip
- verschiedene Themes (u. a. „Tag“ und „Nachtmodus“)
- Unterstützung für [Online und Offline Wörterbücher](#) (z. B. für [ColorDict](#)¹³⁰, siehe Screenshot)
- Highlighting, Annotations, Bookmarks, Share
- Scrolling, Vorlesen (Pro-Version)

Und damit sind nur die wichtigsten Funktionen kurz angerissen. Auch der großartige Support muss besonders hervorgehoben werden: Der Entwickler steht

A Traditional Form of Russian Wood Painting

Most visitors to Russia enjoy buying handicrafts of some kind, such as matryoshka dolls. In early times most of such items were crafted from wood by skilled village craftsmen. A traditional form of wood painting is called shoddchina.

For centuries Russians ate from intricately carved and painted bowls, and they used spoons, cups, and other vessels made of wood. The designs appearing on these were usually of plant or animal life. Entire villages might be devoted to making wooden objects of one kind or another.

Villagers would work on these projects during long, cold winters. In those days there was little to do, so people had time to paint.

project X 🔍

A project in business and science is typically defined as a collaborative enterprise, frequently involving research or design, that is carefully planned to achieve a particular aim.

Project Gutenberg - Wikipedia

Project Gutenberg (PG) is a volunteer effort to digitize and archive cultural works, to "encourage the creation and distribution of eBooks".

Project Vote Smart - Wikipedia

Project Vote Smart (PVS) is a non-profit, non-partisan research organization that collects and distributes information on candidates and political offices in the United States.

Project management - Wikipedia

Project management is the discipline of planning, organizing, directing, and controlling resources to meet specific requirements.

Moon+ Reader mit ColorDict

124. http://android.izzisoft.de/applists/category/named/reading_ebookreaders

125. http://android.izzisoft.de/applists/category/named/reading_rss

126. http://android.izzisoft.de/applists/category/named/reading_translate

127. <https://play.google.com/store/apps/details?id=com.flyersoft.moonreaderp>

128. <https://play.google.com/store/apps/details?id=com.flyersoft.moonreader>

129. <http://ebooks.qumran.org/>

130. <https://play.google.com/store/apps/details?id=com.socialnmobile.colordict>

hier definitiv hinter seinem Produkt! Mehr Details gibt es in den Reviews und Testberichten – in der genannten [Übersicht bei IzzyOnDroid](#)¹³¹ verlinkt, die auch weitere Kandidaten benennt.

Alternativen? Gibt es nicht wirklich. Der [FBReader](#)¹³² ist noch recht verbreitet – unterstützt aber weniger Formate, und möchte außerdem noch auf Konto-Informationen zugreifen (sicher für's Einkaufen von Büchern, welches er wohl unterstützt). [txtr](#)¹³³ unterstützt neben ePub auch PDF – benötigt aber doppelt soviel Platz für die Installation. Wiederum recht verbreitet und gut bedienbar ist [Aldiko](#)¹³⁴, der auf manchen Geräten bereits vorinstalliert ist (und zusätzlich DRM-geschütztes Lesematerial unterstützt). Wie bereits geschrieben: Ein Blick in die Übersicht gibt mehr Informationen: Dort ist u. a. eine Matrix enthalten, die einen allgemeinen Vergleich verfügbarer Lese-Apps ermöglicht.

Der Eine oder die Andere wird sich an dieser Stelle nicht ganz zu Unrecht fragen: „Und was ist mit bekannten und verbreiteten eBook-Reader-Apps wie [Kindle](#)¹³⁵ oder [Google Play Books](#)¹³⁶? Sind die etwa nicht erwähnenswert?“ Sicher sind sie das. Doch ich mache mich ungern von bestimmten Diensten abhängig, was bei diesen beiden Apps definitiv der Fall ist: Die Oberhoheit über das Lesematerial hat hier nämlich der jeweilige Anbieter. Und so kann es durchaus passieren, dass man, im Urlaubsland angekommen, plötzlich feststellt: Alle Bücher sind weg! Kann nicht sein? [Jim O'Donnell](#)¹³⁷ kann ein Lied davon singen. Oder dass der Zugang zu allen über den Service erworbenen Büchern plötzlich nicht mehr funktioniert, weil man die 30-Tage-Rückgabemöglichkeit bei ganz anderen Produkten einmal zu oft in Anspruch genommen hat. Nicht möglich? Wer es nicht glaubt, kann es [bei Heise nachlesen](#)¹³⁸. Ich setze lieber auf Apps, wo ich selbst die Kontrolle habe. Insbesondere über meine Inhalte.



FBReader



Aldiko



Gizmodo: Travelers
Beware: Google Play
Might Delete All Your
Books



Heise: Amazon geht
mit
Kontensperrungen
gegen hohe
Retourenquoten vor

Schule & Studium

Auch Schülern und Studenten steht „Andy“ hilfreich zur Seite: So passt z. B. der mit Formel- und Nachschlagewerken gefüllte Schulranzen früherer Tage heute

-
- 131. http://android.izzysoft.de/applists/category/named/reading_ebookreaders
 - 132. <https://play.google.com/store/apps/details?id=org.geometerplus.zlibrary.ui.android>
 - 133. <https://play.google.com/store/apps/details?id=com.txtr.android>
 - 134. <https://play.google.com/store/apps/details?id=com.aldiko.android>
 - 135. <https://play.google.com/store/apps/details?id=com.amazon.kindle>
 - 136. <https://play.google.com/store/apps/details?id=com.google.android.apps.books>
 - 137. <http://gizmodo.com/a-115983224>
 - 138. <http://www.heise.de/-1927899.html>



bequem in die Jacken- oder Hosentasche. Da wären Formelsammlungen und Referenzen¹³⁹, Apps zum Nachschlagen und Übersetzen¹⁴⁰, Vokabeltrainer & Flashcards¹⁴¹, und vieles mehr. Mit Mensa-Plänen¹⁴² ist dabei auch an das leibliche Wohl gedacht. Um zumindest einen kleinen Eindruck zu geben, habe ich eine App aus diesem Bereich ausgewählt:

Übersicht:
Nachschlagewerke
für Schule und
Studium



Übersicht:
Vokabeltrainer und
Flashcards

The screenshot shows the Merck PSE app interface. On the left is a sidebar menu with the following items: Search, Elements, Classification, Atomic properties, State of aggregation, Property ranking, Discovery, Molar Mass Calculator, Glossary, Settings, Come2Merck, Recommend, and Rate us. To the right is a large periodic table where elements are color-coded according to their group: Alkaline-earth metals (light green), Alkali metals (yellow-green), Transition metals (blue), Lanthanides (light blue), Actinides (purple), Other metals (dark purple), Semimetals (pink), Nonmetals (red), and Halogens (orange). Below the table is a legend with the same color-coded categories. At the bottom of the screen, the text "Merck PSE macht die Erkundung des Periodensystems zum Erlebnis" is displayed.



Merck PSE

Merck PSE¹⁴³ ist mit 4,6 von maximal möglichen 5 Punkten im *Play Store* absolut topp bewertet, und kann daher mit Fug und Recht hier als Vorzeige-App herhalten.

Merck's Periodensystem gibt Schülern, Chemiestudenten und Lehrenden die Möglichkeit, sich umfassend und interaktiv über die Elemente des Periodensystems zu informieren. Damit steht Interessierten ein mehrsprachiges Nachschlagewerk zur Verfügung, das komplexe Inhalte intuitiv erfahrbar macht. So wird die App im *Play Store* beschrieben.

Jede Menge Informationen stehen zu den einzelnen Elementen zur Verfügung. In verschiedenen Ansichten. So lässt sich über einen „Zeitregler“ recht einfach feststellen, welche Elemente zum Zeitpunkt X bereits bekannt waren. Oder anzeigen, wer sie entdeckt hat. Oder, oder, oder – der Möglichkeiten sind hier viele. Für weitere Details empfiehlt sich ein Blick in den Testbericht¹⁴⁴ sowie auf

-
139. http://android.izzysoft.de/applists/category/named/school_references
 140. http://android.izzysoft.de/applists/category/named/reading_translate
 141. http://android.izzysoft.de/applists/category/named/school_flashcards
 142. http://android.izzysoft.de/applists/category/named/school_mensa
 143. <https://play.google.com/store/apps/details?id=de.merck.ptc>
 144. <http://www.androidpit.de/de/android/tests/test/392280/Merck-PSE-HD-Periodensystem-nicht-nur-fuer-Chemiker>



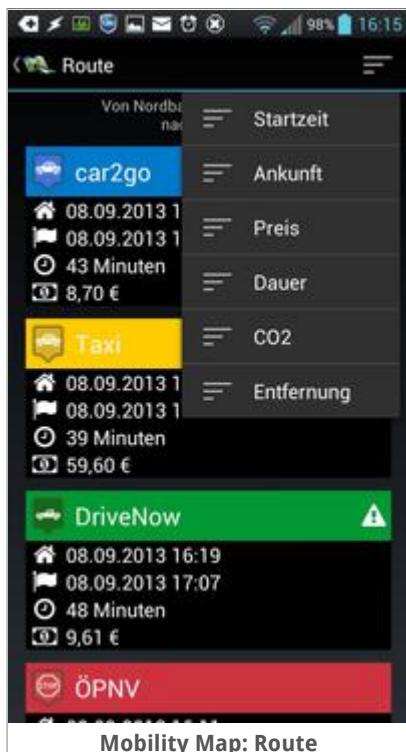
Testbericht: Merck
PSE HD

die [Projektseite](#)¹⁴⁵ – wobei man sich bei Letzterer nicht davon irritieren lassen sollte, dass laufend von irgendeinem iPhone die Rede ist ...

Und natürlich gibt es auch entsprechende Referenzen und Nachschlagewerke für andere Fächer, etwa Physik oder Mathematik, wie z. B. das sowohl für Smartphones als auch Tablets optimierte [Math Ref](#)¹⁴⁶, die für wenig Geld (etwa 0,75 Euro) eine große Menge Wissen in kompakter Form anbietet.



Unterwegs



Da es hier um *Mobi*telefone geht, sind wir natürlich auch mobil. Die Warte- und Reisezeit in Bahn, Bus und Flieger haben wir uns bereits mit [Lektüre](#) verkürzt – aber wohin soll es eigentlich gehen? Und wie kommen wir dahin? In diesem Bereich gibt es Apps für [Fahrpläne](#) (mit [Öffi](#)¹⁴⁷ als dem wahrscheinlich prominentesten Vertreter, sowie dem ebenfalls recht beliebten [DB-Navigator](#)¹⁴⁸ – und auch dem [TripAdvisor](#)¹⁴⁹, wenn es einmal etwas weiter weg gehen soll). Ein guter Allrounder und mein persönlicher Favorit wäre hier [Mobility Map](#)¹⁵⁰, welches nahezu alles unter einem Dach vereint (Öffentliche Verkehrsmittel, Taxi, Car- und Bike-Sharing) – und dabei auch noch erlaubt, Angebote nach persönlichen Präferenzen zu priorisieren: Muss es schnell gehen? Ist der Preis wichtiger? Oder sind es eher Kriterien wie Umweltfreundlichkeit bzw. Streckenlänge? Hat man sich entschieden, navigiert *Mobility Map* auch zum Standort des entsprechenden Verkehrsmittels. Selbst Buchungen sind direkt aus der App heraus möglich. Natürlich kann man auch im Vorfeld festlegen, welche Verkehrsmittel



145. <http://www.merck-chemicals.com/pteapp>

146. <https://play.google.com/store/apps/details?id=com.happymaau.MathRef>

147. <https://play.google.com/store/apps/details?id=de.schildbach.oeffi>

148. <https://play.google.com/store/apps/details?id=de.hafas.android.db>

149. <https://play.google.com/store/apps/details?id=com.tripadvisor.tripadvisor>

150. <https://play.google.com/store/apps/details?id=de.mymobilitymap.android>

überhaupt berücksichtigt werden sollen: Was bringt einem beispielsweise der *Flinkster* um die Ecke, wenn man ihn ohnehin nicht nutzen kann?

Übrigens funktioniert *Mobility Map* nicht nur in deutschen Großstädten. Auch in der „Provinz“ zeigte es mir zuverlässig die nächsten verfügbaren Busse. Im Ausland soll es ebenso funktionieren, doch das konnte ich bislang noch nicht testen.

Ein weiterer Bereich wäre die **Navigation**, zu dem nahezu jedem als Erstes **Google Maps**¹⁵¹ einfallen dürfte – welches auf den meisten Android-Geräten bereits vorinstalliert ist. Das kleine Monster bietet eigentlich grundlegend alles, was zur Navigation benötigt wird: Kostenlose GPS-Navigation mit Sprachführung, Orte finden, Bewertungen, Empfehlungen – und bindet auch soziale Komponenten ein (Freunde auf der Karte sehen und bei Orten einchecken). Die Routenplanung eignet sich sowohl für die motorisierte als auch die unmotorisierte Fortbewegung – benötigt dafür jedoch normalerweise eine ständige Netzverbindung, was sich in eingeschränktem Rahmen umgehen lässt, indem man den Karten-Cache vorher entsprechend befüllt.

Natürlich gibt es auch hier wieder Alternativen, für die ich jedoch auf die genannte Übersicht verweise. Mein persönlicher Favorit wäre da **Locus Map**¹⁵² – welches neben der Navigation auch das Erstellen eigener Tracks (also Routen-Erfassung mit anschließendem Nachschauen, wie man gelaufen/gefahren ist), das Verwenden eigener **KML**- bzw. KMZ-Dateien als Overlays (vollständig mit Bildern etc auch im Offline-Betrieb), und vieles mehr bietet.

Bei der Suche nach einem guten Restaurant, lokalen Veranstaltungen und mehr wollen uns Apps wie **meinestadt.de**¹⁵³ und **Qype**¹⁵⁴ helfen, beim **Shopping**¹⁵⁵ unterstützen uns **Barcoo**¹⁵⁶ & Co. mit **Schnäppchen und Preisvergleichen**¹⁵⁷ ... Kurzum: Auch und gerade unterwegs stehen dem Android-Nutzer zahlreiche Apps zur Verfügung. Wer sich jetzt mehr Details zu den genannten Apps wünscht, findet sie u. a. **bei IzzyOnDroid**.



Übersicht: GPS und Navigation



Google Maps



Locus Map



Übersicht: Kaufen & Verkaufen



IzzyOnDroid: Apps für unterwegs

-
151. <https://play.google.com/store/apps/details?id=com.google.android.apps.maps>
 152. <https://play.google.com/store/apps/details?id=menion.android.locus>
 153. <https://play.google.com/store/apps/details?id=com.allesklar.meinestadt>
 154. <https://play.google.com/store/apps/details?id=com.qype.radar>
 155. <http://android.izysoft.de/applists/category/named/shopping>
 156. <https://play.google.com/store/apps/details?id=de.barcoo.android>
 157. http://android.izysoft.de/applists/category/named/shopping_deals

Gesundheit

Ein weiteres Thema, welches ich bei [IzzyOnDroid](#)¹⁵⁸ ausführlicher behandle. Es gibt Apps, die beim Einkauf von **Nahrungsmitteln** hilfreich zur Hand gehen – wie etwa das bereits zum Thema [Shopping](#) genannte *Barcoo* – oder die uns, wie der [das ist drin Scanner](#)¹⁵⁹ mehr über die jeweiligen Inhaltsstoffe verraten (einschließlich eines integrierten „Lexikons“ für die „äh, äh“ – äh, E-Nummern, ohne die ja heutzutage gar nichts mehr geht: Dank sei der chemischen Industrie: Im Zeitalter von Rinderwahn, Schweinepest, Vogelgrippe, Atomfisch und EHEC-Gemüse wüssten wir ja ohne sie gar nicht mehr, was wir überhaupt noch essen könnten). Dazu sackweise Rezepte-Apps: Für die Verwaltung der eigenen Rezepte empfehle ich [Mein Kochbuch](#)¹⁶⁰ (ermöglicht auch Importe von verschiedenen Websites); wer eher ständig auf der Suche nach „anderen Rezepten“ ist, greift besser zu einem Lesezeichen, beispielsweise auf <http://m.esSEN-und-trinkEN.de/>¹⁶¹.

The screenshot shows a list of dentists (Zahnärzte) in Munich. Each entry includes a small profile picture, the dentist's name, address, distance from the user, a green rating box (e.g., 1.2), and a button labeled "Online buchbar". At the bottom right of the list, there is a blue button labeled "Filtern & Sortieren".

Arzt	Adresse	Entfernung	Rating	Aktion
Dr. Achim W. Schmidt	Hellene-Weber-Allee 19 80637 München	2,3 km entfernt	1,2	Online buchbar
Christian H.W. Markstorfer	Orleansstr. 43 81667 München	3,6 km entfernt	1,2	Online buchbar
Dr. Jan Matthias Hajtó	Briener Str. 7 80333 München	1,4 km entfernt	1,0	Online buchbar
Dr. Jürgen Pink	Maximilianstr. 34 80539 München	2,0 km entfernt	1,2	Online buchbar
Dr. Sebastian von Mohr	Rosenkavalierplatz 9		1,0	Online buchbar

Jameda Arztsuche

Hat man es mit „Essen und Trinken“ ein wenig übertrieben, bedarf es evtl. eines Helferleins zum [Abnehmen](#)¹⁶²: Die Auswahl reicht von Diät-Apps über Fitness-Programme bis hin zu kleinen Tools rund um [BMI](#)¹⁶³ und [BMR](#)¹⁶⁴. Auch Apps zur [Rauchentwöhnung](#)¹⁶⁵ gibt es zur Genüge.

Für den Ernstfall stehen schließlich Apps wie die [jameda Arztsuche](#) bereit, damit man nicht an „irgend einen Quacksalber“ gerät oder stundenlang in Warteschleifen bzw. Warteräumen zubringt: Damit lässt sich bereits im Vorfeld abklären, welcher Arzt für den konkreten Fall besonders geeignet ist. Analoges mit Apps für [Apotheken](#): Welche hat gerade Notdienst, hat sie das benötigte Medikament auch vorrätig, und wie komme ich am besten dorthin? Ist es mit dem Medikament nicht ganz so dringend, existieren auch auf Medikamente spezialisierte Apps für den Preisvergleich in Online-Apotheken – was sich aber wiederum



IzzyOnDroid:
Gesundheit!



Mein Kochbuch



Übersicht:
Schlankheitskur



Wikipedia: BMI



Übersicht:
Raucherentwöhnung



jameda Arztsuche

- 158. http://android.izzysoft.de/books.php?topic=apps_health
- 159. <https://play.google.com/store/apps/details?id=com.snoopmedia.did>
- 160. <https://play.google.com/store/apps/details?id=fr.cookbook>
- 161. <http://m.esSEN-und-trinkEN.de/>
- 162. http://android.izzysoft.de/applists/category/named/health_weight

ebenso gut mit einem kleinen Lesezeichen auf eine mobile Seite wie z. B. handy.medipreis.de¹⁶⁶ erledigen lässt.

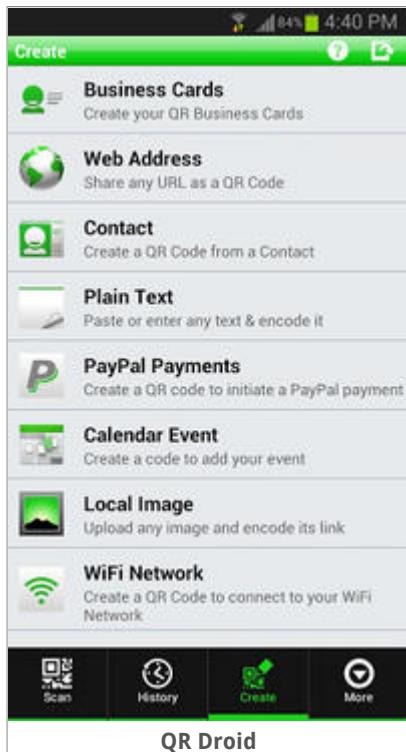
Lieben noch diverse **Notfall**-Apps – und auch diesbezüglich bleibt keiner im Regen stehen: So souffliert die [Mobile Notruf-App für Notfälle](#)¹⁶⁷ bei Anrufen auf 110/112 mit den richtigen Stichworten wie etwa den vier berüchtigten „W-Fragen“: *Wo* ist *Was* geschehen, *Wie* viele Verletzte/brennende Häuser, *Welche* Verletzungen/Gefahren, sowie dem fünften „W“ (nicht gleich wieder auflegen, sondern *Warten* auf eventuelle Rückfragen) – und gibt auch Hinweise auf etwaige weitere Notruf-Nummern (Gift-Notruf, Frauenhaus, Telefon-Seelsorge). Andere Apps erlauben, die wichtigsten medizinischen Eckdaten (Allergien, regelmäßig eingenommene Medikamente) per Widget auf dem Sperrbildschirm zu hinterlegen, damit im Falle der eigenen Unansprechbarkeit ein guter Freund oder aufmerksamer Anwesender den Notarzt darauf aufmerksam machen kann (der Arzt selbst darf nicht auf das Smartphone zugreifen, und sucht so etwas daher dort auch nicht). Ebenso kann sich die Suche nach der nächsten Toilette zu einem Notfall entwickeln, für die es passende Apps gibt (wofür gibt es eigentlich keine?).

Büro, Office & Verwaltung

Auch hier ist die ursprüngliche, wesentlich längere Textfassung [auf die Website ausgelagert worden](#)¹⁶⁸, um Platz für Neues zu schaffen.

„Früher“ sprach man von Büro-Gebäuden. Heute ist das Büro da, wo man gerade selbst ist. Dummerweise auch nach Feierabend. Schauen wir uns also mal die Ausstattungs-Möglichkeiten an.

-
- 163. <http://de.wikipedia.org/wiki/Body-Mass-Index>
 - 164. <http://de.wikipedia.org/wiki/Grundumsatz>
 - 165. http://android.izzysoft.de/applists/category/named/health_quitsmoking
 - 166. <http://handy.medipreis.de/>
 - 167. <https://play.google.com/store/apps/details?id=de.huwig.rhok.notfall>
 - 168. http://android.izzysoft.de/books.php?topic=apps_office



Barcode-Reader sind aus dem Android-Alltag kaum wegzudenken: Sei es für die quadratischen QR-Codes oder die „strichförmigen“ Barcodes, es lassen sich damit einfach Informationen erfassen und austauschen, bzw. Zusatzinformationen einholen. Etwa zu einem Produkt im Supermarkt, oder der Veranstaltung auf einem Plakat. Man kann Adress- und Kalenderdaten einfach austauschen. Sogar zur Automatisierung von Aufgaben lassen sie sich einsetzen. Mein Favorit ist hier [QR Droid](#)¹⁶⁹ (siehe Abbildung), der das meiste davon beherrscht – nicht nur lesend, sondern auch schreibend. So kann ich beispielsweise meine Kontaktdaten als QR-Code bereithalten, damit mein Gegenüber sie mit einem einfachen „Piep“ auf sein Gerät übernehmen kann. Auch habe ich einen solchen Code parat, damit sich Gäste einfach zu meinem WLAN verbinden können – ohne ein ellenlanges Passwort abtippen zu müssen. Die Notiz an der Kühlenschranktür wurde allerdings noch nicht

durch ein solches „Quadrat“ ersetzt 😊

Einen Überblick über sein **Budget** kann man mit Hilfe seines Androiden ebenfalls behalten: Sei es mit einfachen bzw. auch umfangreicheren Budget-Planern à la [Financisto](#)¹⁷⁰, mit vollwertigen Banking-Apps (zu empfehlen in Sachen Komfort und Sicherheit wären die aus dem Hause [StarMoney](#)¹⁷¹) – oder gar mit Apps zur Beobachtung der (und zum Handeln an den) Börsen, alles vorhanden. Man kann also – sollte man aber auch?

In den vorangegangenen Kapiteln klang ja bereits mehrfach an, dass ich eher konservativ bin, was persönliche Daten auf dem Smartphone allgemein und speziell auf fremden Servern und in der Cloud betrifft. Das gilt natürlich insbesondere für so sensible Dinge wie Finanzdaten. Daher sei an dieser Stelle nochmals auf eines deutlich hingewiesen: Das Risiko eines Missbrauchs ist in diesem Falle größer, als wenn sich selbige Daten lediglich auf dem stationären Rechner daheim (oder gar nur in Papierform im verschlossenen Schrank) befinden. Der Anwender muss also für sich selbst entscheiden, ob er derart



QR Droid



Financisto



StarMoney

169. <https://play.google.com/store/apps/details?id=la.droid.qr>

170. <https://play.google.com/store/apps/details?id=ru.orangesoftware.financisto>

171. <http://www.starmoney.de/>

sensible Daten überhaupt auf seinem mobilen Gerät haben, oder gar der Cloud (und somit fremden Rechnern – einige Apps ermöglichen oder erfordern gar den Upload der entsprechenden Dokumente) anvertrauen möchte.

Eine **Kalender**-App ist zwar auf jedem Androiden bereits vorinstalliert – in der Regel ist diese jedoch meist weder sehr benutzerfreundlich, noch weist sie einen besonderen Funktionsumfang auf. Zum Glück gibt es auch dafür zahlreiche Alternativen, wie etwa den abgebildeten [Business Calendar](#)¹⁷². Der lässt sich nicht nur sehr angenehm bedienen (inklusive „Copy-and-Paste“ von Terminen und flexibler Steuerung der Anzeige per „Pinch-to-Zoom“ Zwei-Finger-Geste), sondern bringt sogar ein eigenes Alarm-System mit – was ihn für mich zum Favoriten macht. Hier lässt sich mehr als nur der Alarmton auswählen: Wie oft und in welchen Intervallen soll der Alarmton wiederholt werden, wenn der Anwender nicht reagiert (ihn also offensichtlich überhört hat)? Neben dem „OK, hab's vernommen“ gibt es zudem die Möglichkeit, den gerade tönenden Alarm zu verschieben. Dafür lässt sich aus zuvor konfigurierten Intervallen auswählen, oder auch spontan ein Zeitraum definieren. Und damit wäre allenfalls an der Oberfläche gekratzt. Außerdem stammt diese App aus „deutschem Hause“, sodass auch „nicht-Fremdsprachler“ mit dem aktiven und schnell reagierenden Support keine Probleme haben sollten.



Als Alternative, ebenfalls mit deutschem Support, wäre auch [aCalendar](#)¹⁷³ zu nennen, der sich wiederum voll auf das in Android integrierte Kalendersystem stützt – was die Auswahl oft auf den *Google Calendar* sowie den „lokalen“ Kalender beschränkt. Dafür punktet die App allerdings an anderer Stelle: Sie ist relativ leichtgewichtig, und kommt auch in der kostenlosen Version komplett ohne Werbung daher. Geburtstage werden in der Tagesansicht gleich mit Kontaktbild eingeblendet, Kalendertermine lassen sich über QR-Codes und NFC austauschen, und vieles mehr.

Zusätzliche Features schaltet man sich mit der Kaufversion frei: Erweiterte Einstellungen, Aufgabenlisten, Business-Features wie Einladungen oder die

172. <https://play.google.com/store/apps/details?id=netgenius.bizcal>

173. <https://play.google.com/store/apps/details?id=org.withouthat.acalendar>



Business Calendar



aCalendar

Verwaltung von „frei/belegt“ Zeiten wären nur einige davon. Außerdem tut man dabei gleich noch etwas Gutes: Zehn Prozent seiner Einnahmen reicht *TapirApps* an den *World Land Trust* zum Schutz des bedrohten Wald-Tapirs weiter. Mit jedem Kauf von *aCalendar* erhält dieses Tier ca. 4 m² Lebensraum als Eigentum.

Synchronisieren lassen sich Kalender (und auch Adressbücher) übrigens nicht nur über Google oder Samsung, sondern durchaus auch direkt mit eigenen Ressourcen. Details dazu wieder einmal [in der zugehörigen Übersicht](#)¹⁷⁴.

Passwörter sollen möglichst sicher gespeichert werden. Apps dazu gibt es ja scheinbar wie Sand am Meer – sicher sind diese aber nicht unbedingt. Daher sollte bei der Auswahl unbedingt ein Blick in die [entsprechende Übersicht](#)¹⁷⁵ geworfen werden! Dort werden zwar nicht alle verfügbaren Apps ausführlich vorgestellt, doch die Liste hilft schon einmal, die „potentiell unsicheren Kandidaten“ zu eliminieren. Open-Source Produkten empfehle ich hier den Vorzug zu geben, z. B. [KeePassDroid](#)¹⁷⁶ bzw. [KeePass2Android](#)¹⁷⁷. Beide bieten die Möglichkeit, die eigene Passwort-Datenbank Plattform übergreifend synchron zu halten – also nicht nur auf dem Androiden, sondern auch auf dem PC nutzen zu können.

Generell vorsichtig sein sollte man in diesem Zusammenhang mit dem Austausch von Passwörtern per Copy-Paste: Auf das Clipboard des Androiden haben auch andere Apps Zugriff. Daher vorher schauen, welche Schutzmaßnahmen der jeweilige Passwort-Safe dagegen bietet.

Office-Pakete mögen sich auf dem Smartphone allenfalls zur Betrachtung von Dokumenten oder vielleicht noch kleineren Korrekturen eignen – doch auf Tablets (ggf. gar noch mit externer Tastatur) können sie durchaus für ein „mobiles Büro“ herhalten. Die meisten Apps in diesem Bereich haben sich jedoch leider voll und ganz (und ausschließlich) auf Microsoft-Formate festgelegt, kaum jemand unterstützt mit ganzem Herzen offene Dokumentenformate wie [ODF](#)¹⁷⁸. Eine rühmliche Ausnahme bilden da die Produkte aus dem Hause *SoftMaker*: Je eine App für Text-, Tabellen- und Präsentations-Dokumente bieten vollständige Unterstützung für MS und ODF Formate, sodass man zu beiden Seiten kompatibel ist. Jede einzelne dieser Apps schlägt allerdings mit knapp 7 Euro zu Buche, wobei es in allen drei Fällen auch eine kostenlose Testversion mit 30 Tagen Laufzeit gibt.

PDF-Dateien lassen sich unter Android sowohl anzeigen als auch erstellen. Vorinstalliert sind passende Apps in den seltensten Fällen – sie lassen sich jedoch



Übersicht:
Synchronisation von
Kontakten und
Kalendern



Übersicht: Passwort-
Safes



KeePassDroid



KeePass2Android



Wikipedia:
OpenDocument

174. http://android.izzysoft.de/applists/category/named/office_calendar_sync

175. http://android.izzysoft.de/applists/category/named/office_password

176. <https://play.google.com/store/apps/details?id=com.android.keepass>

177. <https://play.google.com/store/apps/details?id=keepass2android.keepass2android>

178. <https://de.wikipedia.org/wiki/OpenDocument>



in der [passenden Übersicht bei IzzyOnAndroid](#) problemlos finden. Wer sich für die [Lektüre](#) ohnehin bereits den dort genannten *Moon+ Reader Pro* zugelegt hat, bekam damit gleichzeitig einen guten und schnellen PDF-Reader – der es auch erlaubt, in den PDF Dateien Markierungen und Notizen zu hinterlegen. Für die Übersicht: PDF-Stuff Erstellung von PDF Dokumenten buhlen zahlreiche Apps um die Gunst des Anwenders; hier muss man jedoch beachten, dass die Verarbeitung fast immer auf einem Server in der Cloud stattfindet – Vorsicht daher mit „brisantem Material“.



Eine dritte Kategorie sind die „Hosentaschen-Kopierer“, wie beispielsweise der [CamScanner Phone PDF Creator](#)¹⁷⁹. Diese fügen i. d. R. lediglich mit der Kamera aufgenommene Bilddateien zu einem PDF-Dokument zusammen. Ob dafür „die Cloud“ in Anspruch genommen wird, muss in jedem Fall einzeln geprüft werden – die „Internet-Berechtigung“ mag zwar ein Indiz sein, ist aber nicht unbedingt ein Beleg dafür.

Sicher nicht nur für Freiberufler interessant ist das Thema **Zeiterfassung**: Wie viel Zeit habe ich an welchem Projekt verbracht? Diese und ähnliche Fragen beantworten Apps aus der entsprechenden Kategorie. Mein Favorit: [Xpert Timer](#)¹⁸⁰. Wiederum steht eine deutsche Firma mit guten Support hinter dem Produkt, welches sich denkbar einfach bedienen lässt – wie eine Stempeluhr: Bei Beginn der Tätigkeit auf Start, bei eventuellen Pausen auf Pause, und bei Feierabend auf Stop gedrückt. Natürlich müssen vorher einmal Kunde und Projekt erfasst sein – aber dann bekommt man neben zahlreichen Statistiken und Übersichten auch eine Stundenübersicht, die man sogar direkt aus der App heraus verschicken kann. Stundensatz eingetragen? Dann zeigt sich auch gleich, wie es ums Finanzielle bestellt ist. Ein integrierter Barcode-Scanner lässt sich darüber hinaus zur Automatisierung nutzen, indem man ihn z. B. mit bestimmten Projekten verknüpft: Der erste Pieps startet dann die Arbeit an selbigem, ohne dass man es zuvor manuell heraussuchen muss. Projekte lassen sich optional noch in Tasks unterteilen, Reports kann man direkt



179. <https://play.google.com/store/apps/details?id=com.intsig.camscanner>

180. <https://play.google.com/store/apps/details?id=de.xpertdesign.xtdroid>

aus der App heraus verschicken, oder auch im HTML bzw. CSV Format exportieren.

Augmented Reality



Anders als bei der *Virtual Reality* braucht es hier keine zusätzliche Hardware à la [Google Cardboard](#)¹⁸¹. Bei *Augmented Reality* wird die Realität *erweitert*, nicht ersetzt. Dazu werden mehrere Dinge gemischt: Das Kamera-Bild wird mit weiteren Informationen versehen. Meist mit Daten eines oder mehrerer Sensoren. Oder mit Karten-Informationen. Oder weiteren Informationen zu auf dem Bild ersichtlichen Objekten. Oder einer Mischung mehrerer Komponenten ...



Google Cardboard
bei Amazon

Ein gutes Beispiel ist [Google Goggles](#)¹⁸². Hier ist *Augmented Reality* eigentlich nur ein Teilespekt der App, wie im Screenshot zu sehen: Wo bin ich eigentlich, und was schaue ich da gerade an? Auf Wunsch kann *Goggles* entsprechende Informationen einblenden. Auch ohne aktiviertes GPS (wie hier im Bild); mit GPS sind die Informationen natürlich etwas genauer. Und manche scheint die App auch nur preiszugeben, wenn GPS aktiviert ist.



Google Goggles

Wenn ich schon *Goggles* hier erwähne, dann möchte ich auch noch kurz einige weitere Features der App nennen – im übertragenen Sinne lassen sie sich ja alle in dieser Kategorie unterbringen: Man macht Fotos von realen Dingen – und *Goggles* sagt einem, was man da fotografiert hat: DVDs und Bücher (*Goggles* nennt Titel, Preis, und Erwerbsquellen), Logos, Kunstwerke (geniale Sache zum Angeben: Kurzes Foto machen und dann wissend tun, dass van Gogh dieses Gemälde namens ... im Jahre ...), Barcodes (Produkt-Infos und Kaufangebote), Visitenkarten (Übernahme der Daten in die Kontaktliste), und mehr.



Übersicht:
Augmented Reality

Weitere Möglichkeiten – von GeoTags über Navigation bis hin zu Spielen – finden sich in der [Übersicht Augmented Reality](#) bei *IzzyOnDroid*.

181. http://www.amazon.de/dp/B00TF9CBUO/ref=as_li_tf_tl?ie=UTF8&tag=qumranbook-21

182. <https://play.google.com/store/apps/details?id=com.google.android.apps.unveil>

Fernbedienen und Überwachen



Kommen wir uns nicht alle hin und wieder etwas fremdgesteuert vor? Und was fällt uns dazu bei unserem Androiden ein? Das logischste und naheliegendste ist, ihn als Fernsteuerung zu benutzen – was das [entsprechende Kapitel auf meiner Website](#)¹⁸³ ausführlicher beschreibt.

So gibt es verschiedene Apps, die die **Fernsteuerung des PCs** erlauben – für einzelne Programme wie Media-Player, als Ersatz für Tastatur und Maus, aber auch um den kompletten Desktop auf den Androiden zu holen. Die Gegenrichtung ist ebenso möglich. Oder man benutzt den Androiden als **Fernbedienung für Multimedia-Geräte** – seien es Fernseher, Verstärker, oder Set-Top-Boxen; das geht sogar soweit, dass man Medien von selbigen auf sein Android-Gerät streamen lässt. Speziell Fernbedienungen fürs TV-Programm kommen dabei oftmals sogar mit einer EPG-Vorschau, sodass man vor dem Umschalten auch sieht ob sich selbiges überhaupt lohnt. Eine weitere Möglichkeit ist natürlich die Verwendung von [DLNA](#): Das Android-Gerät kann dabei als Client, Server, und auch Kontroll-Instanz genutzt werden.

IzzyOnDroid:
Fernbedienen und
Überwachen



AndRovio

Beim Stichwort „Überwachung“ denken heutzutage sicher viele an unsere Geheimdienste. Deren Apps kenne ich zwar nicht; jedoch für die **Hausüberwachung und -automatisierung** gibt es zahlreiche Lösungen, welches System auch immer im Einsatz ist: Sei es HomeMatic, ZWave, oder KNX, für (fast) alles gibt es eine App. Für Video-Kameras ebenso. Wer Lust hat, kann „ungebetene Gäste“ (etwa einen Einbrecher) sogar „automatisch verfolgen lassen“: Mit [AndRovio](#)¹⁸⁴ und der [dazugehörigen Hardware](#) ist auch das kein Problem.

Zu guter Letzt kümmern wir uns noch darum, wie wir unsere(n) **Server überwachen**. Vom einfachen Website-Monitor über das Monitoring mehrerer Dienste auf einem Server, bis hin zu Clients für Nagios & Co gibt es Android-Apps, die uns dabei unterstützen. Ebenso mit dabei: Spezielle Überwachungs-Apps für den *Raspberry Pi*.

Lunte gerochen? Feuer gefangen? Mehr Details finden sich bei *IzzyOnDroid* im eingangs genannten „Kapitel“ – sowie in den [Remote-Droide Übersichten](#)¹⁸⁵.



IzzyOnDroid:
Remote-Droide
Übersichten

183. http://android.izysoft.de/books.php?topic=apps_remotecontrol

184. <https://play.google.com/store/apps/details?id=com.poignantprojects.androvio>

185. <http://android.izysoft.de/applists/category/named/remotedroid>

Multimedia: Alles, was Krach macht

Zur Vielseitigkeit unserer kleinen Dauer-Begleiter gehört auch die Wiedergabe multimedialer Inhalte. Im allgemeinen Sprachgebrauch meint das: Audio und Video. Im übertragenen Sinne halt: Alles, was Krach macht. Ausführlicher vorgestellt wiederum [bei IzzyOnDroid](#)¹⁸⁶ – und hier der „Kurzabriss“:



Die passenden App-Übersichten [finden sich hier](#)¹⁸⁸.

Mucke alleine reicht nicht – es soll auch auf dem Screen zucken? Kein Thema, auch **Video-Player** für Android sind nicht gerade dünn gesät. Ein richtiger Allesköninger in diesem Bereich ist der [MobePlayer](#)¹⁸⁹: Große Format-Vielfalt, Unterstützung für Untertitel und multiple Audio-Streams, Playlists, Streaming aus dem Netz, sortieren, Thumbnails ... Auf den ersten Blick scheint nichts zu fehlen. Auch eine spezielle Vorbereitung der Videos (Konvertierung) soll nicht nötig sein: Zum Einsatz kommt hier die [FFMpeg-Engine](#)¹⁹⁰; alles, was die versteht, kann also

Die wohl gefragteste Gruppe sind sicher die **Apps zur Musik-Wiedergabe**: Warum noch einen MP3-Player zusätzlich mitschleppen? Allenfalls aufgrund der Akku-Laufzeit (sonst läuft am Ende nur noch der Träger und schnauft, während das restliche Equipment keinen Ton mehr von sich gibt). Entsprechend groß ist hier die Auswahl: Die „üblichen Kandidaten“ spielen die Musik nach Genres (Tags) bzw. Alben ab. Doch nicht nur Nostalgiker greifen gern zu „Folder Playern“, welche sich nach guter alter Manier die Dateien verzeichnisweise vornehmen – für Hörbücher ist dieses Feature schließlich ein „Muss“. Andere Apps wiederum bieten auch das Bearbeiten der [ID3-Tags](#)¹⁸⁷ einschließlich Cover-Bildern an, oder zeigen den Songtext des gerade abgespielten Titels passend zur Musik. Für das Streaming von Musik aus dem Netz gibt es ebenso Apps, wie zur Erkennung des gerade irgendwo laufenden Musikstücks. Die



IzzyOnDroid:
Multimedia Apps



Wikipedia: ID3-Tag



Übersichten: Musik



MobePlayer

186. http://android.izzysoft.de/books.php?topic=apps_multimedia

187. <https://de.wikipedia.org/wiki/ID3-Tag>

188. http://android.izzysoft.de/applists/category/named/multimedia_music

189. <http://m.aptoide.com/app/5972688/MobePlayer>

190. <http://de.wikipedia.org/wiki/FFmpeg>



Übersicht: Wecker abgespielt werden. Und das ist nicht gerade wenig. Leider ist die App aus dem Playstore verschwunden; der App-Link führt daher zu Aptoide.



Zum Bereich Multimedia gehören irgendwie auch die **Wecker**¹⁹¹. Natürlich ist auf jedem Androiden ein solcher vorinstalliert, der für „das Nötigste“ reicht. Aber einmal ehrlich: Vom Hocker haut der uns nicht. Und so gibt es neben den „schnöden“ allgemeinen Weckern eine Reihe von teils recht ausgefallenen Spezialitäten: Etwa Wecker, die wie z. B. Sleep as an Droid¹⁹² den Schlaf überwachen¹⁹³. Nicht nur kann man sich am nächsten Morgen die Aufzeichnungen ansehen – die App achtet auch darauf, dass sie einen nicht gerade aus der Tiefschlafphase reißt und man Gefahr läuft, „mit dem falschen Bein aufzustehen“. Andere Wecker erlauben das Stummschalten erst, wenn man eine Rechen- oder Rätselaufgabe gelöst – oder der Kamera den zu einem genannten Produkt passenden Barcode serviert hat. Und schließlich wären da noch die ortsbasierten Wecker¹⁹⁴, die nicht „wann“, sondern „wo“ losgehen – zum Beispiel, egal ob Verspätung oder nicht, zehn Kilometer vor dem Zielbahnhof.

Sleep as an Droid



Wikipedia:
Schlafphasenwecker



Übersicht:
Ortsbasierte Wecker
& Erinnerer

Tools

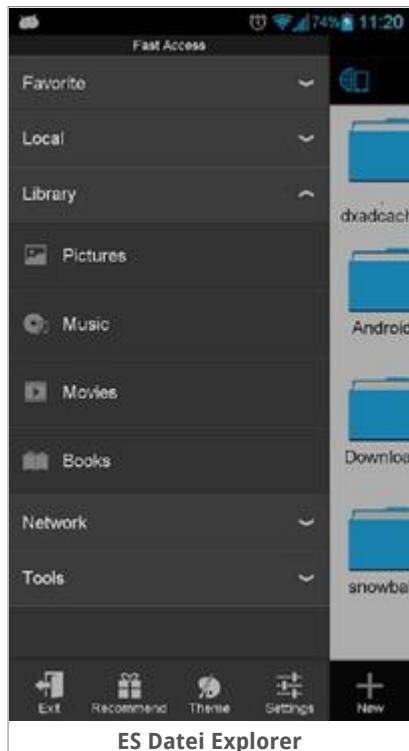
„Tools“ ist ein recht weit gefasster Begriff. An dieser Stelle soll er „technische Helferlein für System-Aufgaben“ bedeuten. Und diesbezüglich muss man des Öfteren zu einer Drittanbieter-App greifen:

191. http://android.izzysoft.de/applists/category/named/multimedia_alarmclocks

192. <https://play.google.com/store/apps/details?id=com.urbandroid.sleep>

193. <https://de.wikipedia.org/wiki/Schlafphasenwecker>

194. http://android.izzysoft.de/applists/category/named/travel_positioning_locationalert



So bringen beispielsweise nicht alle Geräte einen **Datei-Manager**¹⁹⁵ mit, den man jedoch häufiger einmal benötigt. Und selbst wenn ein solcher bereits vorinstalliert ist, fehlen dem Anwender oft einige Funktionalitäten. Die App-Märkte halten jedoch eine Vielzahl an Alternativen bereit: Von einfachen Apps, die sich lediglich um die SD-Karte kümmern, über Datei-Manager für alle lokalen Dateisysteme – bis hin zu solchen, die auch den Zugriff auf den PC oder auf Cloud-Storage ermöglichen. Manche bringen sogar gleich weitere Helferlein mit: Etwa einfache Text-Editoren, Bildbetrachter, Video-Abspieler, oder die Unterstützung von (ZIP) Archiven. Der **ES Datei Explorer**¹⁹⁶ wäre ein solcher Allrounder.



Übersicht: Datei-Manager



ES Dateimanager



Übersicht:
Keyboards und
alternative Eingabe-
Methoden



Graffiti

Spracheingabe, Schrifterkennung (kennt noch jemand **Graffiti**¹⁹⁸ vom Palm? Gibt es auch für Android), bessere Unterstützung für mehrsprachliches arbeiten, Emotikons – oder gar Unterstützung für externe Tastaturen. Bei der Vielzahl an Apps in diesem Bereich bin ich sogar sicher, den einen oder anderen Kandidaten übersehen zu haben.

195. http://android.izzysoft.de/applists/category/named/file_fileman

196. <https://play.google.com/store/apps/details?id=com.estrong.s.android.pop>

197. http://android.izzysoft.de/applists/category/named/tools_keyboards

198. https://play.google.com/store/apps/details?id=com.access_company.graffiti



Übersicht: System-Info Tools



SystemPanel



Übersicht:
Verschlüsseln von
Dateien und
Verzeichnissen

IzzyOnDroid: Tools

Richtig „haarig“ wird es, wenn im System einmal etwas „Seltsames“ vor sich geht, auf das man sich „keinen Reim machen“ kann. Auf diesen Fall sind die wenigsten Android-Geräte vorbereitet. Zum Glück habe ich natürlich eine Übersicht¹⁹⁹ parat, in der sich das Passende finden lassen sollte. Die Palette reicht hier von einfachen **System-Info-Tools** bis hin zu Paketen wie SystemPanel²⁰⁰, die sich für umfangreiches **Monitoring** eignen. Mit letzterem lässt sich beispielsweise auch im Nachhinein feststellen, welche App zu welchem Zeitpunkt aktiv war, sowie welche Ressourcen sie für welchen Zeitraum belegte (siehe Akkusauger).

Ebenfalls in den Bereich der System-Tools gehört das Thema **Verschlüsselung**. Zwar unterstützt Android seit Version 4 das Verschlüsseln ganzer „Laufwerke“ (Geräte-Verschlüsselung) von Haus aus – doch ist dies nicht unbedingt auch das, was der Anwender wünscht. Oft möchte man lediglich einzelne Dateien oder Verzeichnisse verschlüsseln²⁰¹.

Auf diese Weise kann man sensible Daten besonders schützen – während weniger Sensibles im einfacheren Zugriff bleibt. Ein Vorteil gegenüber der Geräteverschlüsselung: Verschlüsselte *Dateien* lassen sich auch relativ „gefährlos“ im Netzwerk speichern – verschlüsselte *Geräte* nicht.

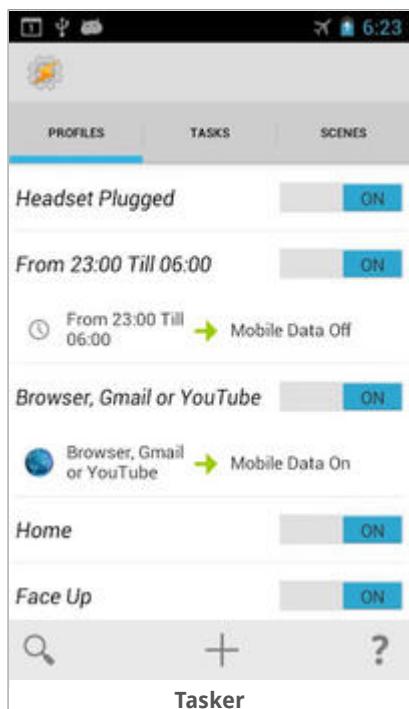
Wer dieses Kapitelchen gern etwas ausführlicher hätte, findet es hier²⁰².



System Panel

-
199. http://android.izysoft.de/applists/category/named/tools_systeminfo
 200. <https://play.google.com/store/apps/details?id=nextapp.systempanel.r1>
 201. http://android.izysoft.de/applists/category/named/file_encrypt
 202. http://android.izysoft.de/books.php?topic=apps_tools

Automatisieren von Aufgaben



Wozu hat man eigentlich einen Hosentaschen-Computer, wenn man dann doch jede Kleinigkeit selber machen muss? Und Mensch ist ja so vergesslich: Wieder einmal das Telefon auf dem Schreibtisch liegen lassen, zum Mittagessen gegangen, und die Kollegen hat das dauernde Klingeln „erfreut“? Oder vergessen, vor dem Starten des Navis GPS anzuschalten? Oder...

Was kann man also tun? Meine [Übersicht zum Thema](#)²⁰³ zeigt da etliche Möglichkeiten auf. Da gibt es einfache Apps für einfache Möglichkeiten (und einfache Leute) – und auch richtig komplexe Dinge, die schon ein wenig Einarbeitungszeit benötigen. Egal, was die Wünsche hier sind – dort sollte sich eine passende App finden.

Hat man einen sehr geregelten Tagesablauf, und möchte lediglich zeitgesteuert ein paar „kleine Dinge“ erledigt haben – wie nachts in den Flugzeugmodus, morgens wieder an, und

von 9-17 Uhr leise? Dann reicht eine zeitgesteuerte App wie [Timeriffic](#)²⁰⁴ völlig aus. Geht jemand oft ins Kino, aber zu unterschiedlichen Zeiten – und da soll der „kleine Quälgeist“ gefälligst still sein? Dann greift dieser eher zu einer „ortsgesteuerten“ App wie [Llama](#)²⁰⁵ – die ebenso Zeit- und Ereignisgesteuertes (Ladegerät angeschlossen, Batteriestand niedrig, etc.) abdeckt.

Wer das Ganze aber richtig ausreizen will, greift zu Apps wie [Locale](#)²⁰⁶ oder besser noch [Tasker](#)²⁰⁷ (Bild links). Gerade bei letzterem kann man sich so richtig austoben – den Möglichkeiten sind hier (fast) keine Grenzen gesetzt: Bei Ankunft zu Hause das WLAN aktivieren. Um Mitternacht in den Flugmodus schalten, morgens um 7 wieder zurück, und dann auch gleich den Audio-Stream der Lieblings-Internet-Radio-Station (oder ein Random-MP3 von der Karte) auf die Ohren. Wenn der Kopfhörer angeschlossen wird, gleich den Musikplayer starten – und wenn die Navi-App gestartet wird, GPS anmachen. Bei beiden Apps lässt



Übersicht:
Automatisierung –
Profile-Switcher & Co.



Timeriffic



Llama



Tasker

203. http://android.izzysoft.de/applists/category/named/tools_automation

204. <https://play.google.com/store/apps/details?id=com.alfray.timeriffic>

205. <https://play.google.com/store/apps/details?id=com.kebab.Llama>

206. <https://play.google.com/store/apps/details?id=com.twofortyfouram.locale>

207. <https://play.google.com/store/apps/details?id=net.dinglisch.android.taskerm>



IzzyOnDroid: Tasker
Ressourcen

sich die Funktionalität überdies noch mit zahlreichen Plugins erweitern. Meine [Tasker Material-Sammlung](#)²⁰⁸ führt etliche davon auf, und verlinkt überdies zu entsprechenden Tutorials, Howtos und Beschreibungen.

208. http://android.izzysoft.de/applists/category/named/resources_tasker

Benutzer und Profile

Android ist mehrbenutzerfähig: für Tablets gilt dies seit Version 4.2, Smartphones wurde diese Funktionalität mit Version 5.0 spendiert. Es können sich also mehrere Anwender ein Gerät teilen – einem „Familien-Tablet“ steht somit nichts im Weg. Für unterschiedliche Szenarien sind sogar unterschiedliche Möglichkeiten vorgesehen: Es gibt Benutzer, Profile, und sogar einen „Gastzugang“.

Detailliert beschrieben ist das Multi-User Konzept auf Englisch [bei Android.com²⁰⁹](#); auf Deutsch findet sich u. a. ein [Artikel im DroidWiki²¹⁰](#).



Supporting Multiple Users



DroidWiki: MultiUser

Benutzer

Ein **Benutzer** erhält eine eigene Daten-Partition sowie einen eigenen Bereich auf der SD-Karte – kann also seine eigenen Apps und Daten verwalten und hat seine eigenen Einstellungen. Der erste eingerichtete Benutzer ist der „Eigentümer“ des Gerätes: Nur dieser kann weitere Benutzer anlegen. Die Apps eines Benutzers können im Hintergrund weiterlaufen, während sich ein zweiter Benutzer anmeldet. Jeder Benutzer muss, nachdem er angelegt wurde, zuerst den ganz normalen Einrichtungs-Wizard durchlaufen – als würde ein neues Android-Gerät erstmals in Benutzung genommen. Er kann sein eigenes Google-Konto einrichten, Apps verwalten, hat seine eigene Bildschirmsperre mit PIN, Password, oder Pattern, etc.



Haben mehrere Benutzer die gleiche App installiert, werden allerdings nur deren Daten getrennt verwaltet – die App selbst wird physisch nur einmal installiert. Gleiches gilt auch für deren Aktualisierungen: Akzeptiert Benutzer A ein Update, erhält Benutzer B selbiges automatisch auch – ob er das will oder nicht. Und zwar einschließlich etwaiger neuer Berechtigungen, welche die App sich eventuell genehmigt.

209. <https://source.android.com/devices/tech/admin/multi-user.html>

210. <https://www.droidwiki.de/Multiuser>

Es gibt unterschiedliche Benutzer-Typen:

- **Primärer Benutzer:** Der „Eigentümer“ des Gerätes. Nur dieser kann (unter *Einstellungen > Nutzer*) weitere Benutzer anlegen. Ein primärer Benutzer lässt sich nur durch einen Factory-Reset wieder vom Gerät entfernen.
- **Sekundärer Benutzer:** Das wäre jeder weitere Benutzer, der vom Eigentümer angelegt wurde. Ein sekundärer Benutzer kann sich selbst wieder vom Gerät löschen, oder auch vom primären Benutzer entfernt werden.
- **Gast:** Ein Gast ist ein „temporärer sekundärer Benutzer“. Es kann nur jeweils einen geben, und der lässt sich schnell wieder „bereinigen“, indem sich der Gast einfach wieder abmeldet – vergleichbar also mit dem „Inkognito-Modus“ eines Web-Browsers, oder einer „Kiosk-Installation“ in einem Internet-Café. Während der Gast angemeldet ist, kann er das Android-Smartphone fast wie ein „normaler Nutzer“ verwenden – einschließlich der Installation von Apps oder der Einrichtung eines Sperrbildschirms. Keine dieser Änderungen überlebt allerdings seinen „Logout“.

Profile

Ein **Profil** ist an einen existierenden Benutzer gebunden, und teilt dessen Ressourcen (z. B. WLAN und Bluetooth); ein Benutzer kann mehrere Profile haben. Jedes Profil hat einen eigenen, abgesonderten Speicherbereich für seine Daten.

Für Eltern mit jüngerem Nachwuchs besonders interessant sind „eingeschränkte Profile“ (siehe [Kinderschutz](#)). Diese sind an den primären Benutzer gebunden – der u. a. entscheiden kann, welche Apps dort zur Verfügung stehen.

SICHERHEIT

Was brauche ich wirklich?

Anti-Virus, Anti-Malware, Diebstahlschutz ... Was braucht es eigentlich wirklich auf dem Androiden? Klar gibt es auch hier wieder für alles eine App – und natürlich auch eine passende [Übersicht bei IzzyOnDroid](#)²¹¹. Das Wichtigste sollte man jedoch (hoffentlich) nicht all zu lange suchen müssen:



Übersicht:
Sicherheits-Apps

211. http://android.izzysoft.de/applists/category/named/security_antimalware

GMV



Wikipedia: GMV

GMV²¹² sollte bereits im biologischen Speicher vorinstalliert sein. Leider wird es oft mit Worten wie „No risk, no fun!“ deaktiviert – was dann meist unschöne Folgen hat. In der Regel taucht der/die Betroffene kurz darauf im Forum auf, und öffnet einen Thread mit dem aussagekräftigen Titel „HILFEEEE!“ (aha, GMV noch immer deaktiviert).

GMV? Was ist das denn nun wieder? Oh-oh ... Das sollte eigentlich jeder haben, zumindest ein wenig davon: **Gesunder Menschenverstand**. Hilft enorm. Auch gegen „Viren“ und „Malware“.

Seien wir doch mal ehrlich: Wie viele Viren gibt es wirklich für Android? Und wie kommen die aufs Gerät? Wie kommt Malware aufs Gerät? Indem man ohne nachzudenken auf alles klickt, was sich bewegt? Indem man eine „böse App“ installiert? Die wichtigsten Regeln beachtet, kann so etwas eigentlich kaum passieren. Vor der Installation einer App sollte man sich z. B. folgende Fragen stellen:

- Ist die Quelle vertrauenswürdig?
 - Positiv-Beispiele: Play Store, F-Droid, Website des bekannten (!) Entwicklers (siehe auch: [Android Markets: Wie sicher sind alternative Quellen?](#)²¹³ bei *IzzyOnDroid*)
 - Negativ-Beispiele: Bei Rapidshare „gefunden“, in einer Tauschbörse aufgetrieben, per eDonkey aus unbekannter Quelle gezogen ...
- Sagen die Permissions vernünftig aus?
 - Positiv-Beispiele: Ein Webbrower muss ins Web, eine SMS-App kann natürlich SMS lesen/schreiben/schicken und braucht ggf. auch (lesend) Zugriff aufs Adressbuch
 - Negativ-Beispiele: Eine Wallpaper-App braucht in der Regel keine Telefonnummern anrufen, ein Ballerspiel muss keine SMS senden, ein Taschenrechner nicht auf Konten zugreifen.
 - Besondere Vorsicht: Apps, die auf persönliche Daten (Kontakte, Kalender, Nachrichten) zugreifen und gleichzeitig ins Internet wollen. Leider lässt sich bei letzterem (Internet) die Frage der Notwendigkeit nicht so einfach beantworten – es könnte auch einfach nur für Werbung-Laden gebraucht werden.

212. http://de.wikipedia.org/wiki/Gesunder_Menschenverstand

213. http://android.izzysoft.de/articles/named/android_markets_safe_to_use

- Hilfestellung hier: Die [Permissions-Liste mit Erklärungen](#)²¹⁴ bei [IzzyOnDroid](#). Die [App-Listen](#)²¹⁵ weisen hier auch zusätzlich potentiell riskante Kombinationen aus, die auf der Seite [Concerns](#)²¹⁶ näher erläutert werden.
- Was sagen andere Nutzer zur App/zum Entwickler (Bewertungen, Forum)?
 - Auch hier wieder GMV aktivieren. Kommentare wie „Geil!“, „Super“, etc. sagen nicht wirklich etwas aus (da hat eher jemand bei deaktiviertem GMV einen Kommentar hinterlassen)
 - Gleches gilt für manchen negativen Kommentar: Nicht gerade selten passiert es, dass jemand einfach zu blöd war. Oder die Anforderungen der App gar nicht verstanden hat.
 - Nicht alle Bewertungen beziehen sich wirklich auf die App. Die kann schließlich nix dafür, wenn der Play Store mal wieder klemmt, und daher der Download nicht funktioniert. Oder die HD-Video-App, die mindestens WVGA benötigt, mit dem Motorola Flipout (mini-Display) im Play Store nicht gefunden wird ...
 - Ganz neue App? Noch keine Bewertungen? Im Zweifelsfall im Forum nachfragen, ob schon jemand die App kennt und etwas dazu sagen kann.

Natürlich können andere Apps aus der „Sicherheits-Abteilung“ eine gute Ergänzung zu GMV sein. Insbesondere bei [Verlust des Gerätes](#) – denn dagegen macht auch GMV nicht immun.

214. <http://android.izzysoft.de/applists/perms>

215. <http://android.izzysoft.de/applists>

216. <http://android.izzysoft.de/applists/concerns>

Firewall und Anti-Virus: Worum handelt es sich da eigentlich?

Bevor man versucht, mit Kanonen auf Spatzen zu schießen, sollten vielleicht ein paar oftmals fehlinterpretierte Begriffe geklärt werden. In den Foren taucht häufig die Frage auf: „Welche Firewall/Antivirus-App sollte ich verwenden?“ In sehr vielen Fällen ist dem Fragesteller nicht einmal bekannt, was diese Apps eigentlich bezwecken. Daher eine kurze Erläuterung:

Eine **Firewall** hat nicht etwa zum Ziel, den Datentraffic einzuschränken, damit die Flatrate des Nutzers weniger strapaziert wird – auch wenn dies eine nicht unbedingt unerwünschte Nebenwirkung darstellt. [Wikipedia](#)²¹⁷ beschreibt eine Firewall kurz und bündig als *ein Sicherungssystem, das ein Rechnernetz oder einen einzelnen Computer vor unerwünschten Netzwerkzugriffen schützt*. Hier geht es in erster Linie darum, unautorisierte Zugriffe von außen zu unterbinden. Die sind im Mobilfunknetz aber eher zu vernachlässigen, da i. d. R. bereits vom Provider unterbunden. Außerdem würde ein Zugriff von Außen voraussetzen, dass auf dem Androiden auch „eine Tür offen“ ist (das ist jedoch normalerweise nicht der Fall – es sei denn, der User hat sie explizit aufgemacht; also etwa eine FTP-Server-App o. ä. installiert, gestartet, und entsprechend konfiguriert). Erst in zweiter Linie geht es um die Gegenrichtung, wobei wiederum der Sicherheits-Aspekt (Datenschutz) im Vordergrund steht. Dieser Part ist es auch, auf den sich Android-Firewall-Lösungen beschränken.

Kurz gesagt: Eine Firewall hat unter Android den Zweck, Apps den Netzwerkzugriff generell zu unterbinden.

Die Notwendigkeit einer **Antivirus** App setzt voraus, dass es auch Viren gibt. Hier möchte ich wieder auf [Wikipedia](#)²¹⁸ verweisen, wo man sich zumindest die ersten paar Absätze durchlesen sollte. Dort heißt es u. a.:

Ein Computervirus [...] ist ein sich selbst verbreitendes Computerprogramm, welches sich in andere Computerprogramme einschleust und sich damit reproduziert. Die Klassifizierung als Virus bezieht sich hierbei auf die Verbreitungs- und Infektionsfunktion.

(Die Hervorhebung im Zitat stammt von mir). So etwas scheitert am Aufbau von Android. Dass sich Apps im Playstore als „Anti-Virus“ bezeichnen, ist daher ein Widerspruch in sich selbst: „Anti-Malware“ wäre korrekter. Doch „Anti-Virus“

217. <http://de.wikipedia.org/wiki/Firewall>

218. http://de.wikipedia.org/wiki/Virus_%28Computer%29

klingt einfach bedrohlicher – und mit der Angst von Menschen lassen sich nun einmal gut Geschäfte machen. Anders als bei Anti-Virus Lösungen auf dem PC, findet bei den Android-Pendants nicht etwa ein Signatur- oder gar heuristischer Scan statt; es werden i. d. R. lediglich die Paketnamen installierter Apps mit (in einer Datenbank gespeicherten) Paketnamen bekannter Malware abgeglichen (siehe auch entsprechende Artikel bei [N-Droid](#)²¹⁹ sowie [Slashdot](#)²²⁰).

Aus meiner Sicht ist der Einsatz solcher Apps eher von fragwürdigem Nutzen. Der Anwender wiegt sich in falscher Sicherheit („Mir kann ja nichts passieren, ich habe schließlich ein Anti-Virus!“), und schaltet somit schneller einmal den [GMV](#) ab. Da die vermeintliche „Schutz-App“ jedoch in vielen Fällen nicht greift, erfolgt genau aus diesem Grunde oftmals eine „Infektion“. Wer Anti-Malware-Lösungen jedoch bewusst als zusätzlichen Schutz verwendet, kann durchaus seinen Nutzen daraus ziehen.



N-Droid.DE:
Virensanner für
Android bieten
kaum Sicherheit



Slashdot: Popular
Android Anti-Virus
Software Fooled By
Trivial Techniques

219. <http://www.n-droid.de/forschung-virensanner-fur-android-bieten-kaum-sicherheit.html>

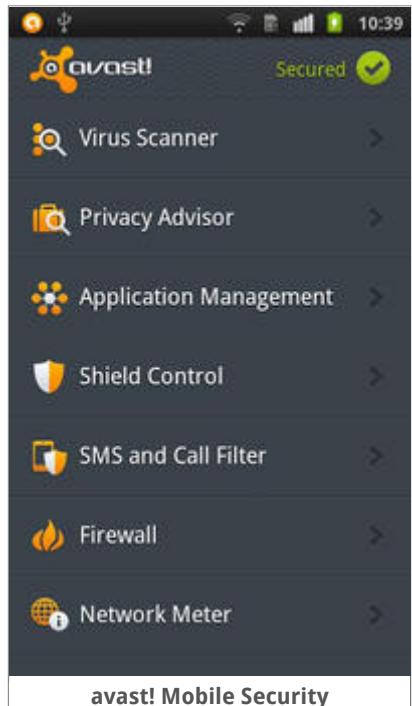
220. <http://it.slashdot.org/story/13/05/07/0226229/>

Anti-Virus und Anti-Malware

Viren und Malware (nein, hier sind jetzt keine Apps zum Malen gemeint – sondern bösartige, hinterhältige Apps wie Spartaner, äh, Trojaner) lassen sich schwer trennen. Und da es von ersteren für Android nicht viele gibt, kümmert sich auch eine „reine Antivirus-App“ wie selbstverständlich mit um letztere. Manche dieser Schutzapps wollen auch gleich den gesamten Sicherheitsbereich abdecken, wie etwa das recht beliebte avast! Mobile Security²²¹. Sie will vor Viren und Malware schützen (mit On-Demand sowie Echtzeit-Scans), bietet einen „Privacy Advisor“ zum Aufspüren von Apps mit verdächtigen Berechtigungen, einen Filter gegen unerwünschte Anrufe und SMS, eine Firewall, und mehr. Sogar ein Datenmonitor und eine App-Verwaltung sind mit an Bord.



avast! Mobile Security



Geht das Gerät einmal verloren (d. h. es wurde entweder verlegt, oder ein Langfinger hat es „abgegriffen“), kann man z. B. einen lauten Alarm auslösen („DIEBE! ICH BIN EIN GEKLAUTES HANDY!“ – äh, ich weiß nicht, wie es mit der Auswahl des Sounds aussieht). Oder aber in wilder Panik gleich alle Daten löschen und das Gerät sperren lassen. Sowas geht einfach per SMS mit dem entsprechenden „Codewort“. Natürlich kann man auch erstmal seinen GMV aktivieren, und sich auf der Karte (Google Maps) zeigen lassen, wo sich der Androide gerade herumtreibt. Damit das alles in vollem Umfang funktioniert (insbesondere der Diebstahlschutz), bedarf es allerdings eines Accounts beim Anbieter.

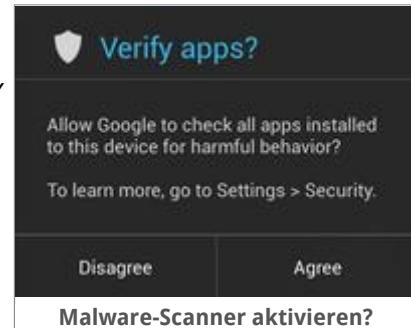
Bei so vielen Features ist jedoch auch die Anzahl der geforderten Permissions entsprechend umfangreich; man sollte also immer überlegen, was man wirklich braucht.

Aufgrund meiner Skepsis gegenüber dieser Software-Gruppe möchte ich an dieser Stelle nicht groß auf die zugehörige App-Auswahl eingehen. Auf meiner

221. <https://play.google.com/store/apps/details?id=com.avast.android.mobilesecurity>

Website findet sich [der Text, mit dem ich das zuvor hier getan habe](#)²²² – und ebenso die [zugehörige Übersicht](#)²²³.

Mit Erscheinen von Android 4.2 aka [Jelly Bean](#) wurde übrigens ein Malware-Scanner bereits im System – genauer gesagt, in die *Google Play Services* – integriert: Installiert man erstmals eine App aus „fremder Quelle“ wird man gefragt, ob man dieses Feature aktivieren möchte. Auch bereits installierte Apps lassen sich damit überprüfen – was automatisch im Hintergrund geschieht, sodass der Anwender davon im Regelfall gar nichts bemerkt. Dabei wird anhand der App-Signaturen (jede [APK-Datei](#) ist mit einer solchen Signatur versehen) mit einer Liste auf den Google-Servern abgeglichen, ob die betroffene App potentiell gefährlich ist. Stellt Googles Cloud-VirensScanner dabei fest, dass es sich bei der zu installierenden App um Malware handelt, wird die Installation abgebrochen. Erscheint die App lediglich verdächtig, erfolgt ein Warn-Hinweis – und der Anwender kann selbst entscheiden, ob mit der Installation fortgefahrene werden soll. Details dazu finden sich beispielsweise in einem [Artikel bei ZDNet](#)²²⁴.



IzzyOnDroid: Schutz vor Schädlingen



Übersicht: Anti-Virus, Anti-Malware, Anti-Wechisses



Google untersucht Android-Geräte jetzt durchgehend auf schädliche Apps

222. http://android.izzysoft.de/books.php?topic=apps_malwareschutz

223. http://android.izzysoft.de/applists/category/named/security_antimalware

224. <http://www.zdnet.de/88190470/google-untersucht-android-geraete-jetzt-durchgehend-auf-schaedliche-apps/>

Bei Diebstahl und Verlust

Eine App, die wirklich gegen Diebstahl und Verlust schützt, muss sicher erst noch erfunden werden. Apps in dieser Kategorie werden also i. d. R. erst dann aktiv, wenn das Kind bereits in den Brunnen gefallen ist. Nur ist es dann natürlich für eine Installation meist zu spät – darum sollte man sich also bereits im Vorfeld kümmern!

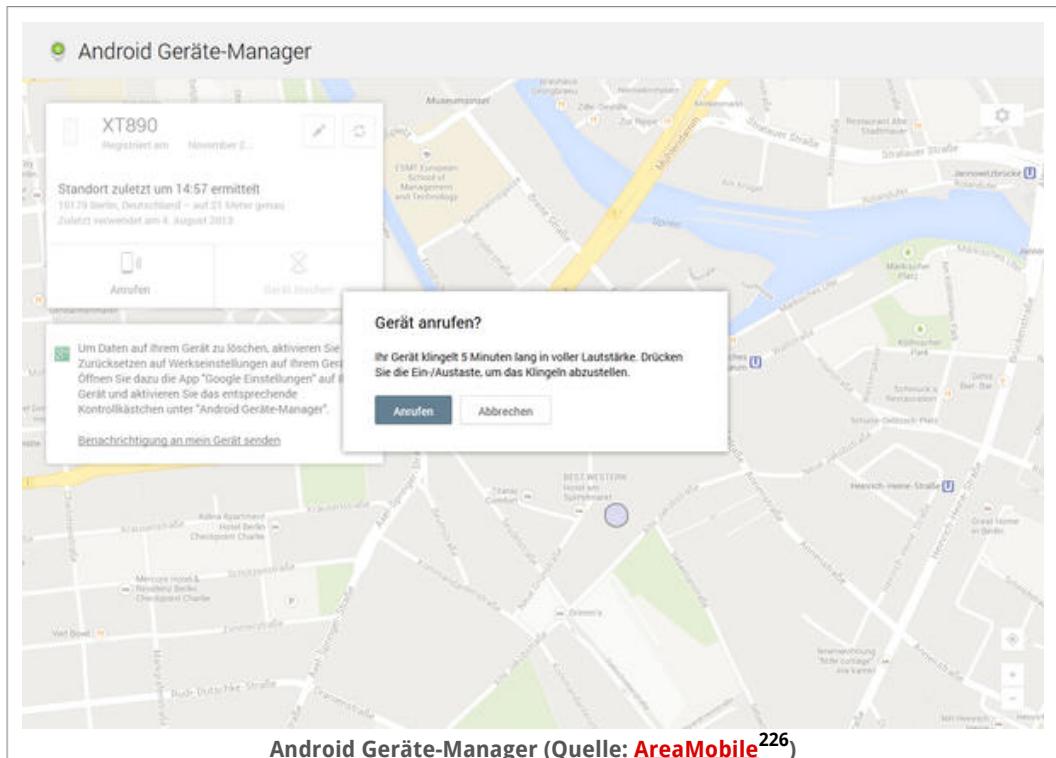
Wer hierfür keine Apps von Drittanbietern einsetzen möchte, kann oftmals auch auf Services des jeweiligen Geräteherstellers zurückgreifen. Nach einer Registrierung auf der entsprechenden Webseite, soll sich auch hier das Gerät bei Verlust wieder aufspüren lassen: Bei aktuellen HTC-Geräten etwa über HTC Sense, bei Motorola via MotoBlur, und auch Samsung bietet entsprechendes an.

Ganz unabhängig vom Hersteller, gibt es seit August 2013 auch eine Möglichkeit, dies direkt über den eigenen Google-Account zu realisieren – nämlich über den [Android Geräte-Manager](#)²²⁵. Die passenden Einstellungen vorausgesetzt, lassen sich die eigenen Geräte über diesen orten. Für den Fall, dass ein Gerät lediglich in der eigenen Wohnung verlegt wurde, kann man es auch zum Klingeln bringen. Eine Löschung sämtlicher Daten auf dem Gerät (etwa bei Diebstahl) ist ebenfalls möglich – sofern man dies zuvor auf selbigem aktiviert hat.



Android Geräte-
Manager

225. <http://www.google.com/android/devicemanager>



Android Gerät-Manager (Quelle: [AreaMobile](#)²²⁶)

Zu finden sind die zugehörigen Einstellungen auf Android-Geräten ab Version 2.2 (Froyo) in der App *Google Einstellungen* unter *Android-Gerätemanager*.



Welche Variante auch genutzt wird: Man sollte im Vorfeld prüfen, wie sie funktioniert (und ob sie dies überhaupt tut), damit man im Ernstfall gewappnet ist. Sonst passiert u. U. das Gleiche, wie in meinem Fall: Mit drei Geräten habe ich den *Android Gerät-Manager* getestet: Ein *Milestone 2* mit Android 2.3.6, ein *LG Optimus 4X* mit Android 4.0.3, und mein Sieben-Zöller-Tablet *Cat Stargate 2* mit Android 4.1.1 traten dazu an. Auf allen drei Geräten waren die Standortdaten

226. <http://www.areamobile.de/bilder/109299>



aktiviert, alle drei waren im WLAN eingebucht, das *Optimus* hatte sogar GPS aktiviert. Alles, was mir der Geräte-Manager am PC mitteilen konnte, war: „Standort nicht verfügbar“ (auch die [Google-Hilfe](#)²²⁷ konnte dieses Problem nicht lösen: Scheinbar klappt es einfach nicht [mit @googlemail.com Adressen](#)²²⁸). Was nebenbei erklärt, warum ich mir den obigen Screenshot bei *AreaMobile* „borgen“ musste. Unnötig zu sagen, dass auch das „Klingeln lassen“ bei keinem der Geräte funktionierte (was zumindest beim Tablet noch verständlich ist). Wie eine kurze Google-Suche bestätigt, bin ich damit in guter Gesellschaft: Auch so manches Nexus-Gerät kämpft mit diesem Phänomen ...

Was aber, wenn das Kind bereits in den Brunnen gefallen, und das Gerät verschwunden ist, bevor man eine der genannten „sichernden Maßnahmen“ treffen konnte? In diesem Fall helfen Apps wie [Android Lost](#)²²⁹ – vorausgesetzt, der Akku ist noch ausreichend gefüllt, und das Gerät mit dem Daten-Netz des Anbieters verbunden. Dank Remote-Installations-Feature lässt sich die App dann nämlich auch noch aus der Ferne installieren, indem man die App-Seite im Playstore besucht, und dort den „Install“ Button drückt. Das löst einen so genannten „Push Install“ aus – der Playstore „schiebt“ die App auf das Gerät.

Nach erfolgreicher Installation sollte sich der Androide nun von der [Android-Lost Website](#)²³⁰ aus per SMS steuern lassen, wofür eine ganze Reihe von Befehlen verfügbar sind. Darunter befinden sich solche, die still und unbemerkt im Hintergrund ablaufen (z. B. einen Status-Bericht anfordern, GPS anschalten und die aktuelle Position mitteilen, eine Tonaufnahme der Umgebung starten – oder die SD-Karte löschen bzw. gleich einen Wipe durchführen) – aber auch andere, die gar nicht unbemerkt bleiben sollen (Alarmsound abspielen, das Gerät einen Text sprechen („Komm nach Hause, Kleiner!“) oder auf dem Display anzeigen lassen. Über die [Website](#)²³¹ kann man ebenfalls auf die Inhalte seines Gerätes zugreifen, und so noch wichtige Daten in Sicherheit bringen bzw. löschen. Das Gerät lässt sich von hier aus sogar steuern – was besonders für Tablets ohne SMS-Funktionalität interessant sein dürfte.

Das alles setzt natürlich voraus, dass der Langfinger den auf dem Gerät eingerichteten Google-Account dort noch nicht gelöscht hat.

-
227. <https://support.google.com/accounts/answer/3265955?hl=de>
 228. <http://www.android-hilfe.de/android-sicherheit-antivirus-firewalls/456013-android-device-manager-offiziell-von-google-14.html>
 229. <https://play.google.com/store/apps/details?id=com.androidlost>
 230. <http://www.androidlost.com/>
 231. <http://www.androidlost.com/>

Worauf Apps Zugriff haben



Wer hat sich nicht schon einmal gefragt, was eigentlich bei der Installation einer neuen App aus dem *Play Store* der seltsame Hinweis sagen möchte: „Diese App darf auf folgendes zugreifen:“ – gefolgt von einer teilweise recht langen Liste komischer Dinge? Nun: Der so Fragende ist hier genau richtig. Zu viele Benutzer ignorieren das nämlich einfach, ohne darüber nachzudenken. Und am Monatsende ist die Überraschung dann gelungen, wenn beim Blick auf die Mobilfunkrechnung die Frage aufkommt: „Moment – ist das jetzt der Betrag, oder die Kontonummer für die Überweisung? Wer hat denn da so viele Premium-SMS ... und all die Anrufe bei 0900-*???"

Was also darf eine App? Oder, anders herum gefragt: Welche App darf denn ...? Auf beide Fragen gibt z. B. [aSpotCat](#)²³² gute und aussagekräftige Antworten. Ein kleiner Punkt neben dem entsprechenden Eintrag zeigt nämlich jeweils an, wie schwerwiegend der *potentielle* Schaden ist, der mit der jeweiligen Berechtigung angerichtet werden könnte. Was natürlich nicht heißt, dass die jeweilige App das auch tut – denn natürlich muss eine SMS-App SMS verschicken können, sonst ergibt sie ja wirklich wenig Sinn. Eine Wallpaper-App hingegen muss das nicht unbedingt.

Außerdem erklärt die App, wofür die entsprechende Permission eigentlich gedacht ist. So hat man diese Information gleich im passenden Kontext.

Eine kurze Übersicht mit ausgewählten Permissions sowie einer kurzen Beschreibung derselbigen findet sich übrigens auch [im Anhang](#); umfangreicher



aSpotCat

232. <https://play.google.com/store/apps/details?id=com.a0soft.gphone.aSpotCat>



und ständig aktualisiert steht diese [bei IzzyOnDroid](#)²³³ zur Verfügung. Und eine Liste alternativer Apps zum Thema, wie gewohnt, ebenfalls [bei IzzyOnDroid](#)²³⁴.

Wer sich bereits vor der Installation absichern möchte, keine Apps mit unerwünschten Berechtigungen auf sein System zu lassen, wirft am Besten einen Blick auf die unter [Playstore Ergänzungen](#) bereits beschriebene App *StripSearch*, mit der sich schon im Playstore das Suchergebnis entsprechend filtern lässt. Wie man Apps im Nachhinein ausgewählte Berechtigungen wieder entzieht, wird unter [Zugriffe Sperren: Firewalls & Permission-Blocker](#) beschrieben.

Permissions erklärt
bei IzzyOnDroid



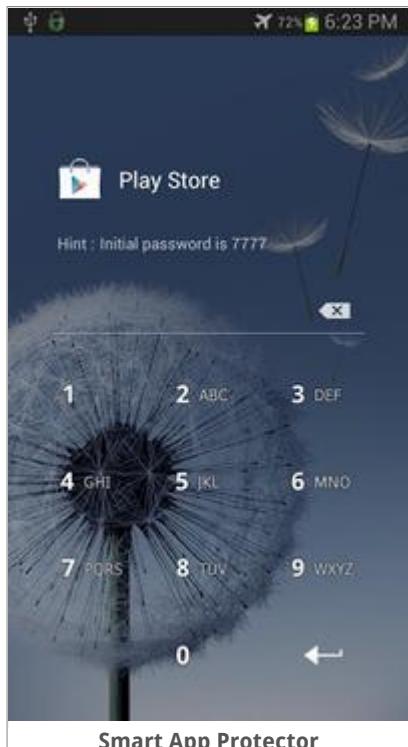
Übersicht:
Permissions

233. <http://android.izzysoft.de/applists/perms>

234. http://android.izzysoft.de/applists/category/named/security_permissions

Apps vor unbefugtem Zugriff schützen

Der Mensch ist von Natur aus neugierig. Was per se nicht schlecht ist – denn dieser Neugier haben wir so manche Entdeckung zu verdanken. Unschön wird es erst, wenn dabei die Privatsphäre verletzt wird. Wie kann man sich also vor „Schnüfflern“ schützen?



Zunächst einmal bringt Android von Haus aus entsprechende Schutzmechanismen mit. So lässt sich konfigurieren, dass bei jedem Einschalten des Displays zunächst ein Pin-Code bzw. Passwort eingegeben, oder ein Entsperrmuster gezeichnet werden muss. Ab Android 4.0 ist auch eine Gesichtserkennung zur Entsperrung möglich. Die entsprechenden Einstellungen finden sich im Menü *Standort & Sicherheit > Display-Sperre einrichten*.

Nicht immer möchte man jedoch das komplette Gerät sperren. Gibt man das Gerät zeitweilig aus der Hand, sollen bestimmte Apps aber unangetastet bleiben: Der Gast soll beispielsweise keine neuen Apps aus dem Playstore installieren, und auch die Finger von der einen oder anderen App lassen. Während sich ab Android 4.2 für solche Fälle ein Gast-Zugang einrichten lässt, können auch in früheren Versionen [App-Locker](#)²³⁵ gute Dienste leisten.

So kann beispielsweise [Smart App Protector](#)²³⁶ jede App mit einem „Zugangscode“ versehen – ohne den sich selbige nicht mehr starten lässt. Das kann ein Pin-Code, ein Muster, ein Passwort, oder auch eine „Geste“ (einfache Fingerzeichnung auf dem Display) sein. Geht der Bildschirm aus, während eine entsperrte App geöffnet ist, muss der Zugangscode erneut eingegeben werden. Natürlich startet sich diese App automatisch nach dem Booten, damit der Schutz auch nicht einfach umgangen werden kann.

Im Playstore ist *Smart App Protector* überdurchschnittlich gut bewertet. Bedenklich stimmen lediglich die verlangten Berechtigungen zum Tätigen von Anrufen, Auffangen ausgehender Anrufe sowie dem Empfangen von SMS, die



Übersicht: App-Locker

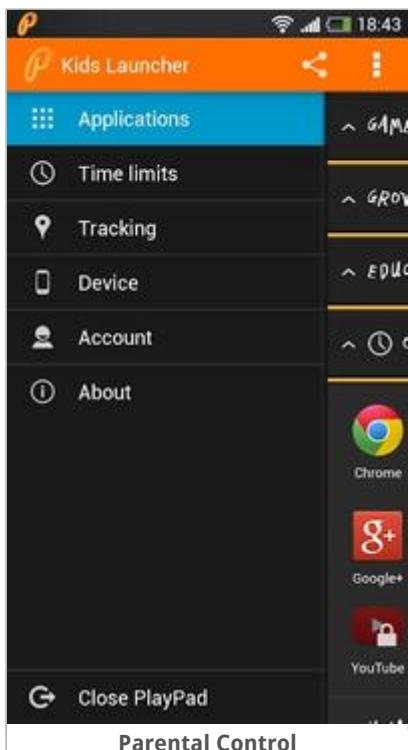


235. http://android.izzysoft.de/applists/category/named/security_applocker

236. <https://play.google.com/store/apps/details?id=com.sp.protector.free>

laut App-Beschreibung für das Sperren ein- und ausgehender Anrufe, bzw. zum Sperren aus der Ferne ("Remote Lock by SMS keyword") gedacht zu sein scheinen. Wem dies Bauchschmerzen bereitet, der findet in der oben verlinkten Übersicht eine ganze Reihe von Alternativen, die ähnliches leisten.

Kinderschutz



Kinderschutz ist ein ganz eigenes Thema, das ich mit diesem Kapitel allenfalls anreißen kann. Den „politischen Teil“ (ab wann sollte ein Kind überhaupt ein Smartphone bekommen, etc.) möchte ich dabei bewusst außen vor lassen – diese Entscheidung lässt sich weder pauschal, noch gar „von oben herab“ treffen, sondern ist eine „private Angelegenheit“ zwischen den jeweiligen Eltern und Kindern. Je früher man den Nachwuchs an diese Technik heranführen möchte, desto größer sind jedoch auch die Ansprüche an das Thema Sicherheit.

Zum Glück lassen uns die Entwickler mit diesem Thema nicht im Dunkeln sitzen – und die zugehörige Übersicht bei IzzyOnDroid²³⁷ gibt uns eine ganze Reihe Apps an die Hand. Herausgreifen möchte ich hier Parental Control²³⁸. Diese App erlaubt neben dem Sperren von Apps auch das Einschränken der Nutzungsdauer nicht gesperrter Apps (um beispielsweise nächtliches Spielen unter der Bettdecke zu unterbinden).

Je nach Altersgruppe können jedoch auch andere Kriterien entscheidend sein. So gibt es etwa auch Kinderschutz-Apps die festlegen lassen, zu welchen Zeiten telefoniert/gesimst werden darf, welche Kontakte überhaupt dafür in Frage kommen, und anderes.

Kommt auf dem Gerät bereits Android 4.3 oder neuer zum Einsatz, bedarf es i. d. R. keiner Drittanbieter-Apps mehr. Eine mit dieser Version eingeführte Neuigkeit nennt sich „eingeschränkte Profile“. Gab es ab Android 4.2 bereits die Möglichkeit, mehrere Benutzer auf einem Tablet zu nutzen (für Telefone war dies ursprünglich nicht gedacht, wurde jedoch mit Android 5.0 nachgeliefert), lassen sich deren Rechte nun auch vom „Besitzer“ anpassen. So kann man einzelne Apps sperren oder auch In-App-Käufe unterbinden. Um diese Funktionalität zu nutzen, muss lediglich unter *Einstellungen > Benutzer > Hinzufügen* statt eines neuen Benutzers ein neues eingeschränktes Profil erstellt werden.



Übersicht:
Kinderschutz



Parental Control

237. <http://android.izzysoft.de/applists/category/named/childprotect>

238. <https://play.google.com/store/apps/details?id=com.appgranula.kidslauncher>

Einen kleinen Haken hat die Sache noch: Die feiner granulierten Berechtigungen (etwa welche Inhalte in Apps nicht angezeigt werden sollen) funktionieren nur, wenn die jeweiligen App-Entwickler auch die entsprechende Unterstützung in den Apps implementiert haben. Hier sollte man sich daher nicht blind auf das Funktionieren verlassen, sondern ggf. detailliert prüfen.

In fremden Netzen

Neben den Apps gibt es auch andere Dinge, die sich von Außen ihren Weg auf unsere Androiden bahnen wollen. Um beispielsweise Daten auszuschnüffeln, oder andere böse Dinge anzustellen. Die Rede ist von „Angriffen aus dem Netz“. Während man im „mobilen Datennetz“, dem heimischen WLAN, oder dem WLAN der Firma noch relativ sicher vor selbigem ist, lässt sich das für sogenannte „offene WLANs“, wie sie häufig in Hotspots anzutreffen sind, nicht unbedingt behaupten: Hier kann sich schließlich jeder anmelden, eine Prüfung findet kaum statt.

Sind zwei Geräte im gleichen Netz (hier: WLAN) unterwegs, können mit passender Software alle Pakete des einen Gerätes mit dem anderen inspiziert werden. Im Klartext übertragene Daten (etwa bei Benutzung des unverschlüsselten HTTP-Protokolls) lassen sich, im Gegensatz zu verschlüsselt übertragenen (etwa HTTPS) auch im Klartext auslesen. Das können lapidare Dinge wie besuchte URLs sein – aber auch auf Webseiten eingegebene Passwörter, und andere Sachen. Auf diese Weise kann sich der Schnüffler also Zugang zu fremden Benutzerkonten verschaffen! Und da so mancher aus Bequemlichkeit die gleiche Benutzername/Passwort Kombination für verschiedene Konten nutzt, kann das recht böse enden.

Worauf sollte man also besonders achten, und wie kann man sich schützen?

- Insbesondere in offenen WLANs die Übermittlung vertraulicher Daten möglichst vermeiden
- Darauf achten, dass Daten (besonders Passwörter) verschlüsselt übertragen werden. Beim Browsen im Web beispielsweise weist ein „<https://>“ (anstelle eines einfachen „<http://>“) am Anfang einer URL auf eine verschlüsselte Verbindung hin (das „s“ steht für „secured“, also „abgesichert“)
- Hintergrunddaten besser abschalten – besonders diejenigen, bei denen vertrauliche/private Daten übermittelt werden, und bei denen man nicht sicher ist, ob die Übertragung verschlüsselt geschieht. Im Zweifelsfall einfach die Hintergrunddaten komplett deaktivieren, während man sich in fremden Netzen herumtreibt.

Zu finden ist das passende Schalterchen je nach Android-Version entweder unter *Einstellungen > Konten & Synchronisierung* oder *Einstellungen > Datenverbrauch*, und ist treffenderweise meist auch mit „Hintergrunddaten“ beschriftet.

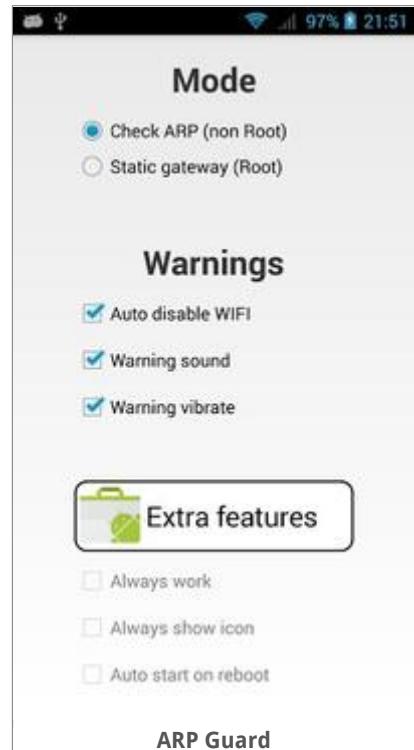
- Wer des Öfteren auf die Nutzung fremder Netze angewiesen ist, sollte über die Einrichtung eines [VPN](#) nachdenken. Android unterstützt das von Haus aus.



ARP Guard

Eine weitere Schutzmöglichkeit bietet z. B. die App [ARP Guard](#)²³⁹. Diese überwacht die eigene Netzwerk-Schnittstelle auf Angriffsmuster bekannter Übeltäter wie [DroidSheep](#), [FaceNiff](#) und Co. Wird ein solcher entdeckt, erfolgt sofort eine Benachrichtigung, und der Vorgang wird protokolliert. Wer auf „Nummer Sicher“ gehen will, kann in einem derartigen Fall auch gleich die Verbindung kappen lassen, und so dem Angreifer den Spaß verderben.

Einige der zum Thema [Anti-Malware](#) erwähnten „Rundum-sorglos-Pakete“ versprechen auch, generell beim Surfen zusätzliche Sicherheit zu bieten – indem sie etwa vor potentiell gefährlichen Seiten (die in einer Datenbank gespeichert sind) warnen.



239. <https://play.google.com/store/apps/details?id=com.myprog.arpguard>

Einen ausführlichen Artikel zu diesem Thema bietet u. a. [Spiegel Online](#)²⁴⁰. Auch wenn sich dieser hauptsächlich auf Windows-Laptops bezieht, lassen sich die meisten Tipps direkt auf Android übertragen. Wer des Englischen mächtig ist, mag darüber hinaus bei [Lifehacker](#)²⁴¹ vorbeischauen, wo sich gleich eine ganze Reihe relevanter Artikel findet.

Dass Gefahr nicht nur in fremden Netzen droht, zeigt ein [Artikel bei Heise](#)²⁴²: Immer weiter geht unsere Vernetzung. Haben wir unsere eigenen Komponenten nicht genügend gesichert, können „Außenstehende“ ohne großen Aufwand unseren Whirlpool an und das Licht ausschalten. Unter anderem.



Spiegel: Datenschutz
– Öffentliches WLAN
sicher nutzen



Lifehacker: How to
protect yourself
from WiFi hacking
apps?



Heise: Smart Home
Hacking

240. <http://www.spiegel.de/netzwelt/web/a-913947.html>
241. <http://lifehacker.com/-955023487>
242. <http://www.heise.de/-1927124>

PRIVATSPHÄRE

Da sitzt man nach einem arbeitsreichen Tag beim Abendbrot (oder, als Schichtler, nach arbeitsreicher Nacht beim Frühstück), im Hintergrund läuft die Lieblingsmusik. Man beginnt, sich langsam zu entspannen. Und plötzlich klingelt das Telefon. Wer mag das sein? Sicher Sascha (oder Mascha), ein angenehmer Plausch war ohnehin längst fällig. Also freudig zum Hörer gegriffen, und ... Nix Sascha, nix Mascha. Am anderen Ende meldet sich eine Firma, von der man zuvor nie gehört hat, und möchte einem etwas verkaufen. Woher haben die schon wieder meine Nummer? Oder die Absender all der an mich persönlich adressierten Werbepost, mit denen ich zuvor ebenfalls nie das „Vergnügen“ hatte ...

Auch in den „News“ hört und liest man immer wieder von Skandalen mit Adresshandel. Oder von Hackern, die wieder einmal die Datenbank eines größeren Unternehmens kopiert haben, und somit nun in Besitz sämtlicher Kundenstammdaten sind. Wofür sie die erbeuteten Daten nun verwenden werden, darüber wollen wir besser nicht nachdenken: Verkaufen? In unserem Namen auf „große Einkaufstour“ gehen? Mit den gestohlenen Identitäten gar Straftaten begehen?

Sollten die Firmen unsere ihnen anvertrauten Daten nicht besser schützen? Diese und ähnliche Fragen werden immer dann laut, wenn ein solcher Skandal an die Öffentlichkeit gelangt. Doch spätestens, wenn es zu Smartphones (und Tablets) kommt, sei eine weitere Frage erlaubt: Was tun wir eigentlich selbst für den Schutz unserer Privatsphäre? Wie viele Daten geben wir freiwillig (und oftmals ohne nachzudenken) Preis?

Privacy First?

Schon bei der Ersteinrichtung eines Google-Accounts auf einem Android-Gerät wird uns eine passende Frage serviert: „Möchten Sie Ihre Daten auf Google Servern sichern?“ Ja, welche Daten sind das denn? Sehr viel wird dazu nicht mitgeteilt. Die Rede ist hier vom „[Google Cloud Backup](#)“ – und gesichert werden neben den Anwendungs-Daten von Apps (die dies explizit unterstützen müssen, was bei Weitem nicht alle tun) und der Liste installierter Apps auch diverse System-Einstellungen, Anruflisten, Browser-History, WLAN-Passworte, und einiges mehr. Es sind also durchaus einige Daten darunter, die als „sensibel“ betrachtet werden können. Und mit Fug und Recht darf man davon ausgehen, dass Google uns diese gratis Dienstleistung nicht aus rein altruistischen Gründen zur Verfügung stellt:

„Wenn man ein Produkt gratis bekommt, ist man nicht der Kunde. Man ist das Produkt. Der Bauer betreibt seine Farm nicht für das Vieh.“ ([Eric Ries](#)²⁴³)

Natürlich lassen sich diese Daten für gezielte Werbung verwenden: Welche Web-Seiten hat man besucht? Mit wem steht man in Kontakt? Auch die bevorzugten Einstellungen des Systems verraten einiges über unsere Vorlieben. Das alles ist mit unserem Google-Account verknüpft – über den übrigens auch unser E-Mail Verkehr läuft, sofern dafür GMail zum Einsatz kommt.

Damit stehen wir zwischen zwei Fronten. Die Einen sagen: „Meine privaten Daten gehen niemanden etwas an! Ich möchte davon nichts in der Cloud sehen!“ Während den Anderen das völlig egal ist: „Ich habe schließlich [nichts zu verbergen](#).“ Wie pflegt mein Vater immer zu sagen: „Ein weites Feld, Luise.“ Und so ist es auch: Es gilt immer abzuwägen, wie viel Privatsphäre wir für mehr Bequemlichkeit zu opfern bereit sind.

Zum Glück werden wir ja explizit gefragt: „Möchten Sie Ihre Daten auf Google Servern sichern?“ Da respektiert also offensichtlich jemand unsere Privatsphäre.



Eric Ries: Trading your Privacy

243. <http://seekingalpha.com/article/1167171>

Wir können also hier einfach „Nein“ sagen, und kümmern uns um unsere Datensicherung selbst (siehe [Datensicherung](#); wer diese Einstellungen im Nachhinein anpassen möchte, findet sie unter *Einstellungen > Sicherung & Zurücksetzen*).



Phandroid: Google says Gmail users shouldn't expect privacy, challenges your definition of the word

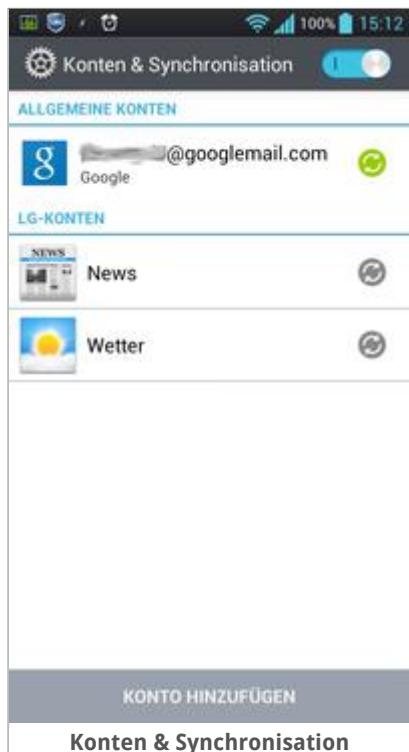
Denken wir zumindest. Doch was Google in diesem Zusammenhang von Privatsphäre hält, stellt u. a. ein [Artikel bei Phandroid](#)²⁴⁴ klar (übrigens mit einem erstklassigen Microsoft-Werbe-Video garniert). Grob übersetzt heißt es dort:

In einer gegen Google vor dem United States District Court für Nord-Kalifornien verhandelten Sammelklage gab Google zu Protokoll: Seine Nutzer sollten davon ausgehen, dass alles, was elektronisch über Google-Server ausgetauscht wird, Freiwild zur Nutzung für Werbung oder auch andere Zwecke ist.

244. <http://phandroid.com/2013/08/13/gmail-privacy-concerns/>

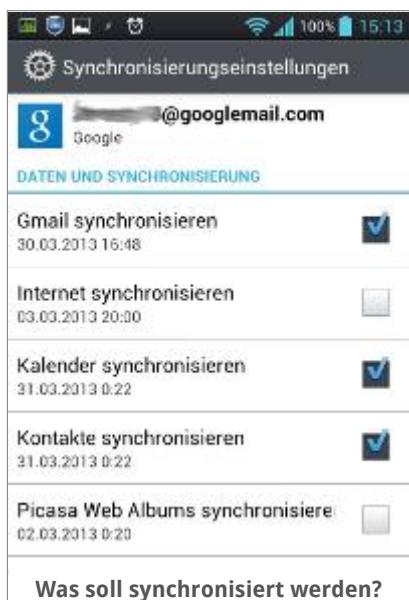
Kontakte und Kalender

Und so erfassen wir freudig unsere Kontakte und Termine mit dem neuen Android-Gerät. Macht sich richtig gut, so haben wir überall Zugriff darauf – schließlich ist das Smartphone ja immer dabei. Und das Tablet zumindest einfacher mitgenommen als der heimische Computer oder Laptop. Doch spätestens, wenn ein zweites Android-Gerät mit dem gleichen Google-Account eingerichtet (oder die Web-Variante von GMail besucht) wird, staunt man nicht schlecht: Hoppla – wie kommen denn die ganzen Daten hierher? Adressen und Termine erscheinen wie von Geisterhand auf dem neuen Gerät – obwohl man doch die Frage „Möchten Sie Ihre Daten auf Google Servern sichern?“ mit einem definitiven „Nein!“ beantwortet hat?



Das sind also entweder keine „Daten“ – oder die „Synchronisation“ von Kalendern und Adressbüchern ist etwas grundlegend anderes als „sichern“. Denn hier wird man nicht ausdrücklich um Genehmigung gebeten. Der „Wunsch“ wird einfach vorausgesetzt. Natürlich nur zu unserem Besten, damit uns die Daten nicht verloren gehen. Doch nicht nur Paranoiker wissen, dass sich Kontakte und Kalenderdaten natürlich u. a. vorzüglich zur Personifizierung von Werbung eignen können. Wenn etwa jemand zwei Mal wöchentlich einen Termin im Sportverein hat, interessiert er sich für Sport – oder benötigt Sportkleidung, Fitness-Artikel, u. s. w.

Die zugehörige Einstellung findet sich auf dem Android-Gerät unter *Einstellungen* > *Konten & Synchronisation*. Und zwar für jeden eingerichteten Account separat. Ein grünes Symbol (das verdächtig an Recycling erinnert – uns also die Wiederverwendbarkeit unserer Daten vor Augen halten sollte) zeigt an, dass zumindest Teile dieses Kontos synchronisiert werden. Tippt man den Eintrag (nicht das Symbol) an, offenbaren sich die synchronisierten Details: Kalender, Kontakte, GMail, und mehr. Jeder Eintrag wieder mit dem bereits erwähnten „grünen Punkt“ oder, wie im rechten Screenshot, einem einfachen Häkchen. Wer also seine Termine oder Kontaktdaten nicht auf fremden Servern sehen möchte, muss das hier explizit deaktivieren. Also nichts mit „Privacy First“.



Wer nun denkt, jetzt hätte man sie alle erwischt: Offensichtlich nicht. Einstellungen zur Privatsphäre scheinen bunt über das Gerät verstreut zu sein. So fand ich gerade in der App *Google Einstellungen* einen interessanten Punkt namens *Smart Lock für Passwörter*. Ein neugieriger Blick hinein verriet mir, was dort aktiviert war: „.... ermöglichen Sie es, dass die Passwörter Ihrer Apps und Websites in Ihrem Google-Konto gespeichert werden.“ Wie gut, dass ich (mit Ausnahme des Playstore-Accounts) auf dem Gerät keines meiner Passwörter benutzt habe! Anderen bleibt u. U. nur die Hoffnung, dass zumindest dies an die globale Google-Backup Einstellung gebunden ist. Weiteres findet sich z. B. [bei Stack Exchange](#)²⁴⁵.

Damit kein falscher Eindruck entsteht: Natürlich hat es seine Vorteile, wenn die betroffenen Daten zum einen gesichert, und zum anderen von verschiedenen Geräten aus gleichermaßen zur Verfügung stehen. Die Entscheidung, ob man dies möchte (und wenn, ob dies über den Google-Service, oder aber über Lösungen von Drittanbietern umgesetzt wird), sollte jedoch dem Anwender überlassen werden. Und zwar nicht über ein „Opt-Out“, sondern per „Opt-In“: Wer es gern hätte, aktiviert es. Wie im vorigen Kapitel: „Möchten Sie, dass Ihre Kontakte, Termine, ... über Google-Server synchronisiert werden?“



Android.SE: Google Account activation:
What should I consider concerning privacy?

245. <http://android.stackexchange.com/q/56994/16575>

Ortsdaten



Google weiß immer, wo wir sind. Oder wo wir wann waren. Denn diese Daten werden über die so genannten „Location Services“ (zu Deutsch: Ortsdienste oder Standortdienste) von uns bereitgestellt. Unter *Einstellungen > Standortdienste* kann man konfigurieren, was genutzt werden soll. Klar, GPS kommt ganz ohne die Google Cloud aus, wenn es separat genutzt wird. Doch bereits bei AGPS (Assisted GPS) kommen externe Datenbanken ins Spiel: Anhand der IDs gerade genutzter Mobilfunkzellen wird dabei der ungefähre aktuelle Standort ermittelt, um so den Aufbau der Verbindung zum Satellitenetzwerk zu

beschleunigen.

Ähnlich sieht es beim „Google Standort Dienst“ aus. Hinter diesem steht eine Google-Datenbank, in der die Positionen von Mobilfunk-Masten und WLAN Access Points gespeichert sind. Auf selbige muss das Gerät also zugreifen – wobei durchaus die Geräte-ID oder das verwendete Google-Konto preisgegeben werden können. Und schließlich fragt man sich sicher nicht zu Unrecht, was es wohl mit den „anderen Services“ auf sich haben mag, für welche der Standort Verwendung finden soll.

Achtung bei Geräten mit Android 4.4 und neuer: Diese haben noch einen weiteren, versteckten Schalter für die Ortsdienste. Zu finden ist er unter *Einstellungen > WLAN > Erweitert*, und möchte „Auch bei deaktiviertem WLAN nach Netzwerken scannen“.

Ebenfalls interessant ist in diesem Zusammenhang der Ortsdaten-Cache, der zur Beschleunigung der Standort-Ermittlung Verwendung findet. Wer gerade noch an der Weltzeit-Uhr auf dem Berliner Alexanderplatz stand, kann sich schließlich fünf Minuten später kaum bereits am Londoner Trafalger Square befinden. Bis zur genauen Ortsbestimmung darf also getrost zunächst davon ausgegangen werden, dass der Anwender „in der Nähe“ des Alex ist. Doch Dank dieses Caches kann sich ebenso jemand, der unbefugt Zugriff auf selbigen „erhalten“ hat, ein gutes Bild von unserem Tagesablauf machen. Mit Bordmitteln scheint es jedenfalls nach wie vor unmöglich, diesen Cache bei Bedarf manuell zu leeren. Zumdest nicht auf eine Art, die für den normalen Anwender ersichtlich wäre.

Welche Daten sammelt Google eigentlich?

Dieser Frage geht auch ein [Artikel bei Stack Exchange](#)²⁴⁶ nach. Eine umfassende Antwort sollte sich in [Google's Datenschutzerklärung](#)²⁴⁷ finden.



Stack Exchange:
What information
does Android send
to Google?

Was Google sammelt

Da sind natürlich die Angaben, die ein Nutzer von sich aus und bewusst macht. Etwa bei der Erstellung eines Google-Accounts. Welche Angaben das umfasst, steht natürlich jedem selbst anheim (ebenso, ob diese den Tatsachen entsprechen – denn dies wird zumindest derzeit nicht geprüft).

Weitere Daten werden bei der Nutzung von Google Diensten erfasst. Welche dies sind, ist nicht unbedingt jedem sofort klar – schließlich findet diese Erfassung im Hintergrund statt. In diese Kategorie fallen u. a.:

- **Gerätebezogene Informationen:** Modell, Betriebssystem, eindeutige Gerätekennungen (z. B. IMEI/IMSI), Telefonnummern. Einige dieser Informationen werden u. U. auch automatisch mit dem verwendeten Google-Account verknüpft.
- **Protokolldaten:** Suchanfragen, IP-Adresse, Cookies, Geräte-Ereignisse (Abstürze, Hardware-Einstellungen, Browsertyp).
Was vielen nicht bewusst ist: Aus dem Anruf-Protokoll des Android-Gerätes wird auch gespeichert, wen man wann, wie oft, etc. kontaktiert hat! Details dazu finden sich im Dashboard (s. u.).
- **Standort-Informationen:** Sensor-Daten, WLAN-Netzwerke, Sendemasten



Google's
Datenschutz-
Erklärung

Dies ist nur ein kurzer Auszug aus o. g. Datenschutzerklärung. Hinzu kommen noch weitere, dort nicht explizit genannte Dinge, die so manchem selbstverständlich erscheinen mögen. Bereits genannt wurden Kontakte und Kalender. Darüber hinaus nutzen viele jedoch weitere Dienste von Google, wie etwa [Google Drive](#)²⁴⁸ mit seinem „Online Office“. Natürlich landen auch hier alle Daten auf Google's Servern.



Google Drive

246. <http://android.stackexchange.com/q/43361/16575>

247. <http://www.google.com/policies/privacy/>

248. <https://drive.google.com/>

Was Google mit den gesammelten Daten macht

Auch davon spricht die Datenschutzerklärung. Die Einleitung des entsprechenden Abschnittes möchte ich an dieser Stelle einfach einmal zitieren:

Wir nutzen die im Rahmen unserer Dienste erhobenen Informationen zur Bereitstellung, zur Instandhaltung, zum Schutz sowie zur Verbesserung dieser Dienste, zur Entwicklung neuer Dienste und zum Schutz von Google und unseren Nutzern. Wir nutzen diese Informationen außerdem, um Ihnen maßgeschneiderte Inhalte anzubieten – beispielsweise um Ihnen relevantere Suchergebnisse und Werbung zur Verfügung zu stellen.

Ferner ist die Rede davon, dass die gesammelten Informationen allen Google-Diensten zur Verfügung stehen. Datenschützer nennen so etwas ein „Super-Profil“ (bei Spiegel gibt es dazu eine ganze Artikel-Serie, die beispielsweise in einer Box links [auf dieser Webseite](#)²⁴⁹ zusammengefasst ist): Die angebotenen Dienste sind derart umfangreich, dass sich mit ihrer Hilfe ein eben so umfangreiches Profil des Nutzers erstellen lässt.

Während über weitere Nutzungsarten bislang nur spekuliert wurde, kündigen sich erste Dinge mittlerweile handfest an. So berichtete [Engadget Anfang Oktober 2013](#)²⁵⁰ von einer Änderung in Google's Nutzungsbedingungen, welche es dem Service erlaubt, sämtliche Reviews, Kommentare, +1, etc. zu Werbezwecken für das entsprechende Produkt nutzen zu können. Mit anderen Worten: Kein Google+ Nutzer sollte sich wundern, bald sein Foto neben einem Produkt im Web zu finden – zusammen mit dem Hinweis, „Martin Mustermann empfiehlt ...“ (wobei man natürlich „Martin Mustermann“ mit dem eigenen Namen ersetzen muss). Derartige Werbung dürfte sich wahrscheinlich am Häufigsten bei denjenigen zeigen, mit denen man „verzirkelt“ bzw. anderweitig verknüpft ist. Denn wenn der Martin das toll findet ...

Und weiter geht's: Die bereits genannten Standort-Daten in Verbindung mit ortsbezogener Werbung gelangen wohl bald in eine neue Dimension. So berichtet Digiday [in einem Artikel](#)²⁵¹ von einem neuen Beta-Programm Googles, welches sich kurz wie folgt beschreiben lässt: Der Anwender sucht nach einem Begriff (Beispiel: Schraubenzieher). Ein in der Nähe befindlicher Laden hat bei Google den entsprechenden Service gebucht (und das Stichwort hinterlegt), et voila, taucht auch dessen Werbe-Anzeige sogleich prominent auf. Es wird nicht darauf eingegangen, ob die Anzeige nur in der Suche eingeblendet wird; denkbar wäre auch, dass hierzu die Browser-Chronik zum Tragen kommt. Hat man also am Montag nach einem Schraubenzieher gesucht, und kommt am Mittwoch

249. <http://www.spiegel.de/netzwelt/netzpolitik/a-884100.html>

250. <http://www.engadget.com/2013/10/11/google-users-shared-endorsements-ads/>

251. <http://digiday.com/platforms/google-tracking/>



Spiegel: Google's Super-Profil



Engadget: If you use Google services, you could become an ad next month



Digiday: Google Takes Its Tracking Into The Real World

zufällig in der Nähe eines passenden Geschäfts vorbei ... Minority-Report lässt grüßen.

Damit ist jedoch noch lange nicht das Ende der berüchtigten Fahnenstange erreicht. So berichtet Slashdot von einem neuen Google-Patent²⁵², das den „Faktor Mensch“ ziemlich in den Hintergrund rutschen lässt, und auch ein wenig an den Chat-Bot ELIZA²⁵³ erinnert. Anhand eines Beispiels wird das Patent beschrieben: In Reaktion auf eine eingehende Nachricht mit einem „Hallo, wie geht's?“ sucht das System im Hintergrund automatisch nach einer passenden Antwort, die dem Anwender dann als solche vorgeschlagen wird. Hierbei wird u. a. auf Informationen aus sozialen Netzen, aber auch auf vorige Konversationen zwischen Empfänger und Sender, sowie zwischen Sender und „gemeinsamen Freunden“ (Google+ Kreise) zurückgegriffen. Eine mögliche Antwort könnte dann so aussehen: „Hi David, mir geht's gut. Du bist ja nun nach 3 Jahren von Firma ABC zu XYZ gewechselt. Wie gefällt Dir Dein neues Umfeld?“



Slashdot: Google Patents Fooling Friends With Snooping, Chatbots



Wikipedia: ELIZA

Wo man die erfassten Daten kontrollieren kann

Einen vollständigen Einblick in alle gesammelten Daten bekommt der Anwender nicht (was sicher dem Schutz seiner Gesundheit dient, da dies die Gefahr eines Herzinfarktes drastisch erhöhen könnte). Doch zumindest grob thematisch lässt sich schauen, was denn da so gesammelt wurde. Dafür bietet Google das mit dem Account verknüpfte Dashboard²⁵⁴. In diesem findet man:

- alle mit dem Account verknüpften Android-Geräte (die dazu gespeicherten Informationen lassen sich einsehen: IMEI, letzte Aktivität, wann registriert)
- Daten zu diversen Diensten, wie etwa Chrome Lesezeichen, GMail, Google Docs, Kalender. Informationen lassen sich hier teilweise verwalten, in wenigen Fällen (Chrome Lesezeichen) sogar löschen.
- Webprotokoll: Immer, wenn man mit seinem Google-Account angemeldet eine Websuche durchführt, wird dies protokolliert. Wer das nicht wünscht, meldet sich am besten immer explizit ab, wenn dies möglich ist (und man nicht beispielsweise gerade auf seine Google-Mails zugreifen muss). Das Protokoll lässt sich hier allerdings auch löschen.



Google Dashboard

Einen tieferen Einblick in das Webprotokoll erhält, wer auf einer Suchergebnis-Seite rechts oben auf das Zahnrad-Symbol klickt, und dort „Webprotokoll“ auswählt: Hier sieht man nämlich alle erfassten Suchen.

252. <http://tech.slashdot.org/story/13/11/20/161244/>

253. <http://de.wikipedia.org/wiki/ELIZA>

254. <https://www.google.com/dashboard/?hl=de>

Jetzt noch einmal auf das Zahnrad-Symbol, und „Einstellungen“ auswählen. So, hier lässt sich das Sammeln abschalten. Vorher gleich noch den Link zum Löschen aller Google-Suchaktivitäten betätigt, damit auch ältere Einträge verschwinden. Oder zumindest uns nicht mehr angezeigt werden: *Sicher ist [nur], dass nichts sicher ist. Und selbst das ist nicht sicher.* (Joachim Ringelnatz)



Android-Digital: So erfährst du was Google über dich weiß



Google Standortverlauf



Google Suchverlauf



Google Geräte und Aktivitäten

Einige zusätzliche Anlaufstellen nennt [Android-Digital.DE](#)²⁵⁵:

- **Standortdaten:** Bei aktivierten Ortungsdiensten (*Einstellungen > Standortzugriff*) zeichnet Google auf, wann man sich wie lange an welchem Ort aufhält. Deaktivieren lässt sich dies in der App *Google Einstellungen* unter *Kontoverlauf > Google Standortverlauf*. Dies muss man auch tun, will man den Standortverlauf löschen. Gesammelte Daten lassen sich [bei Google Maps einsehen](#).
- **Google Suche:** Alle über die Suchmaske eingegebenen Anfragen werden ebenfalls von Google gespeichert. Diese Sammlung lässt sich [bei Google History einsehen](#).
- **Google Geräte und Aktivitäten:** Natürlich wird auch darüber Buch geführt. Was von Google gespeichert wurde, findet sich [bei Google Security](#).
- **Google Takeout:** Ein Archiv mit Daten aller verwendeter Google-Dienste bietet [Google Takeout](#)²⁵⁶. Hier lässt sich auswählen, von welchen Diensten man die archivierten Daten herunterladen möchte. Per Gmail wird man benachrichtigt, sobald das Archiv (für einen begrenzten Zeitraum) zum Download verfügbar ist, und unter welcher URL man es abrufen kann.

Welche Daten wohin weitergegeben werden



Google Takeout

Auch dazu äußert sich die Datenschutzerklärung. Es würden keine Daten weitergegeben, außer ...

- mit expliziter Einwilligung des Nutzers
- „Domain-Administratoren“ haben Zugriff auf die Daten. Dies betrifft u. a. Anwender von Google Apps.
- für Verarbeitung durch andere Stellen. Das sind „vertrauenswürdige Unternehmen“, die im Auftrag von Google arbeiten, und an Google’s Datenschutzerklärung gebunden sind.
- aus rechtlichen Gründen (z. B. auf behördliche Anordnung hin). Hier dürften dann auch Dinge wie der [PRISM-Skandal](#)²⁵⁷ anzusiedeln sein.

255. <http://www.android-digital.de/news/erfaehrst-du-google-ueber-dich-weiss-anleitung-5456/>

256. <https://www.google.com/takeout>

Darüber hinaus werden möglicherweise „zusammengefasste, nicht-personenbezogene Daten“ an Partner wie etwa Verlage, Werbeunternehmen, etc. weitergegeben.

257. <https://de.wikipedia.org/wiki/PRISM>

Digitales Testament



Inactive Account Manager

„Was passiert mit Deinen Online-Daten, wenn der Sensenmann überraschend vorbeikommt?“, fragt Google in einem Werbe-Video. Seit einer Weile hat man nämlich in Mountain View eine Antwort darauf: Den [Inactive Account Manager](#)²⁵⁸. Dieser lässt sich über die [Account-Einstellungen](#)²⁵⁹ erreichen. Ist man längere Zeit nicht online, kann selbiger eine automatische Löschung vornehmen – oder einem Kontakt des Vertrauens den Zugang gewähren. Zur Sicherheit erfolgt jedoch zuvor noch eine Rückfrage von Google via SMS sowie Mail and das hinterlegte Zweitkonto: Das Fernbleiben könnte ja auch andere Gründe haben, wie z. B. Krankheit.



Laut [Tagesschau](#)²⁶⁰ unterstützt diese „Testamentsfunktion“ u. a. Youtube, GMail, Picasa, und Google+. [N-Droid](#)²⁶¹ benennt zusätzlich die Dienste Blogger, Drive, und Google Voice. Inwiefern sich die digitale Erbschaft auch auf gekaufte Inhalte wie etwa Apps, Musik, oder Videos bei Google Play erstreckt, gibt auch das [offizielle Statement](#)²⁶² von Google nicht her; es darf aber sicher bezweifelt werden.

Während andere Dienste teilweise ihre eigenen Lösungen anbieten, gibt es auch weitere Ansätze. So erwähnt genannter Tagesschau-Artikel u. a. „World without me“ und „Planned Departure“, wo man zu Lebzeiten Passwörter bzw. Videos hinterlegen kann, in denen der gewünschte Umgang mit dem Online-Erbe erklärt wird.



N-Droid: Google ermöglicht digitales Testament



Google Blog: Plan your digital afterlife

258. <https://www.google.com/settings/u/0/account/inactive>

259. <https://www.google.com/settings/account>

260. <http://web.archive.org/web/20130415232645/http://www.tagesschau.de/googleplus114.html>

261. <http://www.n-droid.de/neues-google-feature-ermöglicht-digitales-testament.html>

262. <http://googlepublicpolicy.blogspot.de/2013/04/plan-your-digital-afterlife-with.html>

Welche Apps und Unternehmen sind sonst noch fleißig am Sammeln?

19 Datenschutzbehörden aus aller Welt haben in ihren Ländern erstmals Mobilanwendungen und Webauftritte von Unternehmen und öffentlicher Einrichtungen untersucht. Insgesamt nahmen die Kontrolleure im Rahmen des Global Privacy Enforcement Networks (GPEN) mehr als 2200 Webseiten und Apps unter die Lupe, berichtete der damalige Bundesdatenschutzbeauftragte Peter Schaar. Vor allem Apps fielen in dem Test reihenweise durch: 90 Prozent der mobilen Anwendungen seien als mangelhaft beim Schutz der Privatsphäre der Nutzer eingestuft worden.

(aus: [Heise: Viele Apps fallen im Datenschutztest durch](#)²⁶³)



Heise: Viele Apps fallen im Datenschutztest durch

Zu diesem Thema hat Johannes Wallat einen interessanten und umfangreichen [Blog-Artikel bei AndroidPIT](#)²⁶⁴ verfasst. Zwar haben sich einzelne Details mittlerweile wieder geändert – doch die Tendenz scheint sich gehalten zu haben. Beispiele gefällig?

Da wäre etwa die offizielle Twitter-App. Sie verlangt als Berechtigungen u. a. den Vollzugriff auf die Kontakte sowie Lesezugriff auf die Anrufliste. Für die Funktionalität der App wird dies definitiv nicht benötigt (ich habe auch noch keine Twitter-App von Drittanbietern gefunden, welche diese Permissions verlangt). Ein Schelm, wer böses dabei denkt. Der Bericht, dass sich Twitter [ganze Adressbücher heruntergeladen](#)²⁶⁵ hätte, klingt vor diesem Hintergrund durchaus plausibel.

Doch nicht nur Twitter fällt in dieser Form auf. Auch „kleinere“ Dienste wie Foursquare werden in diesem Zusammenhang genannt. Nicht fehlen dürfen natürlich Größen wie Facebook (Kontakte Lesen/Ändern, Anrufliste Lesen/Bearbeiten – welche natürlich prompt gleich [beim ersten Start der App auf den Firmenserver hochgeladen](#)²⁶⁶ werden) oder WhatsApp (Zitat: „Überträgt ohne Nachfrage alle gespeicherten Telefonnummern nicht anonymisiert an den eigenen Server. Überträgt Mobilfunkanbieter.“). Über die Google-Dienste habe ich ja bereits weiter oben berichtet.



AndroidPIT: Daten-Sammel-Apps



AndroidPIT: Geklauter Daten



AreaMobile: Facebook App übermittelt unerlaubt Handynummern

263. <http://www.heise.de/-1936262.html>

264. <http://www.androidpit.de/whatsapp-facebook-skype-daten-sammel-apps>

265. <http://www.androidpit.de/Geklaute-Daten>

Ein weiterer Kandidat in dieser Reihe ist der beliebte Messenger *Skype*, der ja seit einer Weile einer gewissen Firma in Redmond gehört. Hier berichtete [Heise Security](#)²⁶⁷ im Mai 2013, dass ein Leser sie auf ungewöhnlichen Netzwerkverkehr nach einem Skype-Chat informierte – wobei der eigene Server auf eine mögliche Replay-Attacke hinwies. Und tatsächlich bestätigte ein Test: Eine IP-Adresse aus Redmond, die sich eindeutig Microsoft zuordnen ließ, griff nach der Chat-Session auf sämtliche in selbiger verschickten [https](https://) Links zu. Eine der Test-URLs enthielt dabei Anmeldeinformationen, eine andere verwies auf private Dateifreigaben eines Cloud-Dienstes. Von Heise zur Rede gestellt, beteuerte Skype, die Nachrichten würden lediglich gescannt, „um Links zu Spam- und Phishing-Seiten zu filtern“ – was alles andere als logisch ist: Spam- und Fishingseiten verwenden normalerweise unverschlüsselte [http](http://) Verbindungen. Auch solche URLs hatte Heise in seinem Test „geskypt“ – jedoch griff Microsoft auf keine einzige davon zu.

Wenn einer, angesichts des Grau'n,
Will nur Vorinstalliertem trau'n,
Schon meint, dass er nun sicher wär,
So irrt sich der.

(frei nach Wilhelm Busch, [Der fliegende Frosch](#)²⁶⁸)

Man muss sich nicht unbedingt Apps von Drittanbietern installieren, will man ausspioniert werden. Das Erschreckende ist: So etwas kann einem bereits mit vorinstallierten Apps passieren. So berichtet Ben Lincoln in seinem Artikel [Motorola is Listening](#)²⁶⁹ von einer Entdeckung, die er im Juni 2013 auf seinem *Motorola Droid X2* machte: Er benutzte dieses Gerät, um einige Dinge im Zusammenhang mit Microsoft Exchange AutoSync zu testen. Zu diesem Zweck wollte er den Traffic überwachen, und nutzte dafür einen so genannten „intercepting proxy“ (auf Deutsch als „[transparenter Proxy](#)²⁷⁰“ bekannt), der sämtliche Netzwerkpakete mitschneidet. Bei der Durchsicht des Mitschnitts fielen ihm erstaunlich viele Pakete auf, die an einen Server namens *ws-cloud112-blur.svcmot.com* adressiert waren – was ihn natürlich stutzig machte. Kurz zusammengefasst: Als er sich die Sache näher anschaut, musste er feststellen, dass sämtliche Accountdaten wie verwendete Mailserver und Email-Adressen, Facebook und Twitter Konten, Photobucket und Picasa, Youtube Account, Exchange Kontodaten – und zwar jeweils mit den zugehörigen Passwörtern, und zumeist im unverschlüsselten Klartext – zu eben diesem Server geschickt wurden! Und das war nur ein Ausschnitt, es sind noch weit mehr

266. <http://www.areamobile.de/news/24618>

267. <http://www.heise.de/-1857620>

268. <http://www.wilhelm-busch-seiten.de/werke/frosch.html>

269. http://www.beneaththewaves.net/Projects/Motorola_Is_Listening.html

270. http://de.wikipedia.org/wiki/Proxy_%28Rechnernetz%29#Transparenter_Proxy



Heise Security:
Vorsicht beim
Skypen - Microsoft
liest mit



Wilhelm Busch: Der
fliegende Frosch



Ben Lincoln:
Motorola is Listening



Wikipedia:
Transparenter Proxy

Dienste betroffen. Interessanterweise passierte dies nur, wenn die vorinstallierten Apps für diese Dienste genutzt wurden; Motorola scheint die Änderungen also nicht am Android-System selbst vorgenommen zu haben.

Um diesem „noch einen obendrauf zu setzen“, spürte Ben auch noch einen „Command and Control²⁷¹“ Server auf: Motorola verwendet hier das Jabber-Protokoll, um den jeweiligen Androiden bei Bedarf „fernzu steuern“! Zum mindesten drei Remote-Befehle konnte Ben identifizieren, die jeweils das Gerät veranlassen, bestimmte Daten an den genannten Server zu versenden.

Wer sich in der englischen Sprache nicht fit genug fühlt, Ben's Artikel zu lesen, findet eine gute Zusammenfassung in einem Blogbeitrag bei AndroidPIT²⁷². Da hinter der ganzen Sache das Motorola-spezifische *Blur* steckt (was ja auf fast jedem Androiden aus dem Hause Motorola läuft), dürfte das X2 nicht lange mit diesem Problem allein bleiben.

Sicher macht Motorola das nur, um seinen Anwendern im Servicefall besser helfen zu können. So liegt ja jederzeit ein Backup der wichtigsten Daten bereit. Wobei man leider vergessen hat, dem Anwender auch mitzuteilen, wo er selbiges bei Bedarf anfordern kann.</Sarkasmus> Die Ausrede, es hätte sich nur um ein einzelnes Testgerät gehandelt, kann Motorola übrigens nicht mehr bringen: Mittlerweile wurde ähnliches Verhalten auch von einem *Photon 4G* (Android 2.3.5) berichtet. Ein *Defy* (Android 2.*), ein *Milestone 2* (Android 2.3.5), und ein *Xoom Tablet* (Android 3.* und 4.*) haben sich ebenfalls bereits dazugesellt, auch ein *Droid Razr Maxx I* verhielt sich derartig. Was man da wohl vom neuen *Moto X* erwarten darf, bei welchem selbst das Mikrofon ständig aktiviert ist? Dass die Geheimdienste dies bereits zur Aufnahme von Gesprächen nutzen²⁷³, ist mittlerweile bekannt. Die FAZ betitelt es subtil: Das Smartphone, die freiwillige Fußfessel²⁷⁴...

Falls sich da gerade jemand schadenfroh die Hände reiben will: So etwas passiert nicht nur bei Motorola-Geräten, sondern beispielsweise auch bei den



Wikipedia:
Command-and-
Control-
Technologien



AndroidPIT:
Motorola sammelt
angeblich heimlich



Gizmodo: The FBI
Can Remotely
Activate
Microphones in
Android Phones to
Record
Conversations



FAZ: Das
Smartphone, die
freiwillige Fußfessel

271. <http://de.wikipedia.org/wiki/Botnet#Command-and-Control-Technologien>

272. <http://www.androidpit.de/motoblur-motorola-nutzerdaten-sicherheit>

273. <http://gizmodo.com/996086550>

274. <http://www.faz.net/aktuell/feuilleton/debatten/ueberwachung/a-12317519.html>

ach so sicheren Business-Schwarzbeeren. Denn selbst der [Blackberry 10 schickt fleißig Zugangsdaten nach Hause](#)²⁷⁵.

Was denn: Kein Motorola- oder Blackberry-Gerät, und jetzt enttäuscht, dass sich niemand für die eigenen Daten zu interessieren scheint? Keine Angst, es gibt ja noch immer die Telekommunikationsunternehmen, die sämtliche Verbindungsdaten für ein halbes Jahr (sicher? Nicht länger?) aufheben. Und was sich damit so alles anstellen lässt, kann man in einem [Artikel in der Zeit](#)²⁷⁶ nachlesen. Klingt zunächst banal, was da gespeichert wird: Es sind ja keine Inhalte, versucht man uns zu beruhigen, sondern „[nur die technischen Eckdaten](#)“. Also wer wann in welcher Zelle eingebucht war, mit wem zu welcher Zeit telefonierte, wann, wohin und woher SMS/MMS verschickte, wie viele Daten von welchem Standort aus übertrug ... Der Grünenpolitiker Malte Spitz wollte es wissen. Er [verklagte erfolgreich die Telekom](#)²⁷⁷ auf Herausgabe dieser Daten, und stellte sie der Zeit zur Verfügung.

Die genannten Daten stehen schließlich nicht isoliert. Hinzu kommen zahlreiche Dinge, die sich mit ihnen verknüpfen lassen. Etwa Statusmeldungen auf Facebook und Twitter: Über ihren Zeitstempel lassen sie sich wunderbar zuordnen. Was dabei herauskommen kann, findet sich im verlinkten Artikel in Form eines interaktiven Videos. Wer mag, kann Herrn Spitz also für ein halbes Jahr durch sein Leben begleiten – virtuell, versteht sich.

Schließlich darf man auch nicht die ganzen staatlichen Überwachungsprogramme vergessen. Allen voran [PRISM](#)²⁷⁸, das im Juni 2013 in aller Munde war. Laut verlinktem Wikipedia-Artikel ganz vorn mit dabei: *Microsoft (u. a. mit Skype), Google (u. a. mit YouTube), Facebook, Yahoo, Apple, AOL und Paltalk*. Mit Ausnahme der letzten beiden Teilnehmer, wurden tatsächlich alle Firmen bereits erwähnt. PRISM steht für *Planning Tool for Resource Integration, Synchronization, and Management*, und existiert als streng geheimes Programm in den USA bereits seit 2005. Dabei greifen NSA und FBI live auf alle möglichen Kommunikationsdaten zu – sowohl in Sachen Telekommunikation, als auch Internet Datenübertragungen. Und bevor jetzt jemand meint, so etwas könne ja nur bei den Amis passieren: Weit gefehlt. Nur wenig später flog (noch im gleichen Monat) das britische [Tempora](#)²⁷⁹ Projekt auf, welches ähnlich gelagert ist. Und einem [Artikel in Zeit Online](#) spioniert auch Frankreich seine Bürger bereits seit Jahren gründlich aus. Passiert bei uns nicht? Sicher? Oder wurde es nur noch nicht aufgedeckt?

275. <http://frank.geekheim.de/?p=2379>

276. <http://www.zeit.de/digital/datenschutz/2011-02/vorratsdaten-malte-spitz>

277. <http://www.zeit.de/online/2009/35/vorratsdaten-spitz-telekom>

278. http://de.wikipedia.org/wiki/PRISM_%28%C3%9Cberwachungsprogramm%29

279. <http://de.wikipedia.org/wiki/Tempora>



Blackberry 10 macht E-Mail-Passworte für NSA und GCHQ zugreifbar



Die Zeit: Was Vorratsdaten über uns verraten



Die Zeit: Grüne wollen Schweigen der Telekom brechen



Wikipedia: Tempora



Zeit Online: Frankreich spioniert

Übrigens bringt es nicht viel, nun auf Mail-Dienste zu verzichten, und ausschließlich per Briefpost zu kommunizieren. Auch schon abgedeckt, zumindest in den USA – wie wiederum Zeit Online berichtet²⁸⁰. So wird dort der gesamte Briefverkehr abgelichtet: Wer hat wem wann geschrieben. Das geschieht ebenfalls bei der Deutschen Post²⁸¹. Allerdings werden die Umschläge (in beiden Fällen) dabei nicht geöffnet. Normalerweise nicht. Wird uns jedenfalls versichert. Auch geht die Deutsche Post wesentlich sparsamer mit der Datenerfassung und -speicherung um: So wird laut einem Bericht von Spiegel Online²⁸² lediglich die Anschrift, nicht aber der Name einer Adresse erfasst, sowie dieser Datensatz lediglich für 3 Tage gespeichert.

Völlig Paranoiden bleibt somit nur die Rückkehr zur Buschtrommel: Da hört zwar jeder mit – aber keiner weiß, wer da wem etwas mitteilen will ...



Zeit Online: US-Regierung scannt offenbar gesamten inländischen Briefverkehr



Spiegel Online:
Deutsche Post
erfasst nur
Adressen, keine
Namen

280. <http://mobil.zeit.de/politik/ausland/2013-07/usa-briefe-mict-ueberwachung-usps>

281. <http://web.archive.org/web/20130709013408/http://www.tagesschau.de/inland/deutschepost114.html>

282. <http://www.spiegel.de/netzwelt/netzpolitik/a-910043.html>

Die Cloud

Noch eine? Es ging doch bereits weiter oben um die „Google Cloud“! Das ist prinzipiell richtig – nur betraf es dort die bereits vorinstallierten und teilweise automatisch aktivierten Dinge. Aber es gibt noch weit mehr. Apps und Dienste, die wir oft ohne großes Nachdenken nutzen. So landen weitere Daten auf fremden Servern, und vervollständigen etwa über uns angelegte persönliche Profile. Ja, schimpft mich einen Paranoiker! Spätestens im nächsten Kapitel jedoch werde ich zeigen, dass solche „Profile“ wirklich existieren. Und zwar nicht nur bei den jeweiligen Diensten, sondern weit darüber hinaus, wie ein [Artikel bei Heise](#)²⁸³ zeigt. Denn auch die Schlapphüte der NSA (siehe auch [weiteres](#)) bedienen sich hier mit teilweise dubiosen Methoden.

Streut man seine Daten über möglichst viele verschiedene Dienste (natürlich von möglichst verschiedenen Anbietern), erschwert dies selbstverständlich die Bildung eines „Komplett-Profil“. Nutzt man dazu noch unterschiedliche „Identitäten“, hilft dies weiterhin der Verschleierung. Dummerweise widerspricht das jedoch der Bequemlichkeit: Wie fein ist doch ein „Single Sign-In“. Man muss sich nur ein einziges Passwort merken, und kommt an alles heran. Das Google-Passwort wird bereits an so vielen Stellen akzeptiert, das es fast danach schreit: „Melden Sie sich mit Ihrem Google-Konto an!“

Neben den zahlreichen Google-Diensten (oh, [Google Drive](#)²⁸⁴ als bequeme Daten- und Dokumentablage habe ich noch gar nicht erwähnt? Oder Google's [Picasa](#)²⁸⁵ für die Bilder, die natürlich mit GeoTags versehen hochgeladen werden? Oder [Google+](#)²⁸⁶ für den „sozialen Austausch“?) gibt es ja durchaus noch weitere. Wer mich kennt, wartet sicher schon darauf, dass ich die Gruppe benenne: Ja, ich bezeichne sie als „asoziale Netzwerke“, und habe dafür schon so manche Kopfnuss bekommen. So etwas ist halt Geschmackssache.

Da wären also noch Facebook und Twitter als namhafteste Vertreter, von denen ersteres des Öfteren für Schlagzeilen aufgrund seiner sich ständig wandelnden „Datenschutz-Bestimmungen“ sorgt (und man sich so manches mal fragt, vor wem die Daten da eigentlich geschützt werden sollen). Natürlich steht es jedem selbst zu, wie viel er wo von sich Preis gibt. Und zugegeben: Sogar ich habe ein Profil bei Xing und nutze Twitter, bin also ebenfalls ein wenig „asozial“.

283. <http://www.heise.de/-2037324.html>

284. <https://drive.google.com/>

285. <http://picasa.google.com/>

286. <https://plus.google.com/>



Heise: „Muscular“
stellt Sicherheit der
Cloud in Frage

Nicht zu vergessen auch Dinge wie [Evernote](#)²⁸⁷, die beliebte Notiz-App, die längst ebenso auf Desktop-Systemen Einzug gehalten hat. Und so erfolgreich ist, dass Google mit seinem [Keep](#)²⁸⁸ die ganze Sache nachmacht. Was war da noch gleich ... Oh, [Dropbox](#)²⁸⁹ & Co. als Datenspeicher. Oder aber, Last but not Least:

287. <https://evernote.com/>
288. <https://drive.google.com/keep/>
289. <https://www.dropbox.com/>

Google Now

Der Super-Service aus dem Hause Google, der mit Jelly Bean (Android 4.1) auf Android-Geräten Einzug gehalten hat, und Apple's Siri alt aussehen lässt wie eine Bahnsteigansage. Oder besser wie einen Info-Stand im Kaufhaus. Denn Google Now²⁹⁰ beantwortet unsere Fragen bereits, bevor wir sie stellen. Was meine Korrektur-Leserin Sabine mit Erschrecken erkannte:

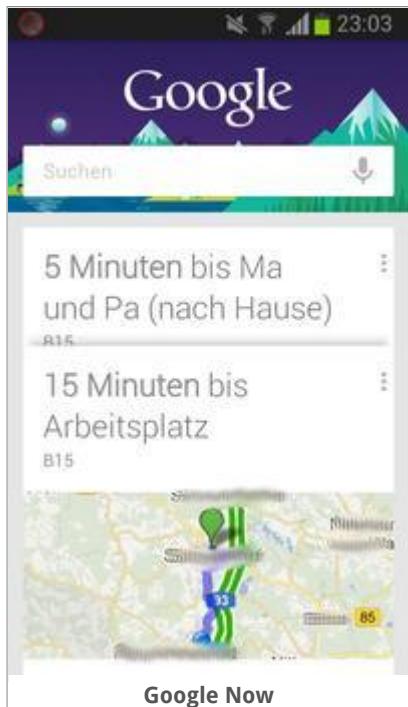
Dass meine Standorte „überwacht“ werden, war mir ja in der Theorie bekannt. Das in der Praxis ums Ohr geschlagen zu bekommen, war doch noch unheimlich. Sagt mir doch morgens Google, dass ich 15 Minuten zur Arbeit brauche (und kein besonderer Verkehr wäre)! Google hat quasi registriert, dass ich mich Tags zuvor mehrere Stunden an ein und der selben Stelle aufgehalten habe – und hat daraus geschlossen, dass ich dort arbeite.

Zudem hat er gleich meine Kontakte im Hinterkopf und mir heute gesagt, dass ich fünf Minuten zu meinen Eltern brauche, nachdem ich Tags zuvor ein paar Minuten dort war. So deutlich ist es einfach gruselig.

Gewusst hab ich's ja. Aber es so zu spüren ist doch nochmal eine andre Nummer!



290. http://en.wikipedia.org/wiki/Google_Now



Wie bereits zuvor erwähnt, hat Sabine ja Google alle dafür notwendigen Daten bereitgestellt: Kontakte und Termine wurden mit Google synchronisiert, die Ortsdaten abgefragt, und so weiter. Wer bisher davon ausging, dass dies alles separate Dienste seien, ist damit nicht mehr auf dem aktuellen Stand: Im März 2012 hat Google seine Privacy Policy entsprechend angepasst – und versteht sich nunmehr als „ein großer Dienst mit mehreren Abteilungen“, die sich den gemeinsamen Datenbestand teilen. Wogegen Europas Datenschützer aktuell wieder vorgehen, wie in einem [Spiegel-Artikel vom 3.04.2013](#)²⁹¹ zu lesen ist.

Die Nutzung dieses Dienstes erfordert jedoch (neben einer Android-Version von 4.1 oder höher) auch die aktive Einwilligung des Anwenders. Für den Zugriff auf die Daten, die er ohnehin schon hochgeladen hat? Der bereits zitierte [Artikel von SeekingAlpha](#)²⁹² beschreibt



Spiegel: EU Datenschützer leiten Untersuchung von Google-Praktiken ein

es etwa folgendermaßen:

Wenn jemand sich aktiv für den Dienst anmeldet, wird er Google beim Versuch, den Dienst an seine Bedürfnisse anzupassen, unweigerlich mehr Daten bereitstellen, als er ursprünglich beabsichtigt.

Ein [Artikel bei ReadWrite](#)²⁹³ beschreibt das Funktionieren von Google Now folgendermaßen:

Google Now aggregiert die Informationen, die Google ohnehin bereits auf täglicher Basis über den Benutzer sammelt: Zugriffe auf Mails, Kalender, Kontakte, Textnachrichten, den aktuellen Standort, Einkaufs-Gewohnheiten, Zahlungs-Gewohnheiten, ebenso wie die Vorlieben bei Musik, Filmen und Büchern. Es kann sogar die Fotos des Anwenders scannen und anhand ihres Themas (nicht nur des Dateinamens) identifizieren. Der einzige Aspekt unseres Online-Lebens, der hier noch nicht erfasst ist, sind auf Google+ zum Ausdruck gebrachte Meinungen. Aber das wird zweifellos noch folgen.



ReadWrite: Google knows more about you than your family does

291. <http://www.spiegel.de/wirtschaft/soziales/a-892168.html>

292. <http://seekingalpha.com/article/1167171>

293. <http://readwrite.com/2012/06/29/google-now-knows-more-about-you-than-your-family-does-are-you-ok-with-that>

Das klingt erschreckend – kann aber auch so erschreckend bequem sein, wie ein [Artikel bei WebProNews](#)²⁹⁴ feststellt:

Es teilt uns das Wetter mit, bevor wir in den Tag starten. Sagt uns, mit welchem Verkehr wir auf dem Weg zur Arbeit rechnen müssen. Steht man auf dem Bahnsteig, tut es kund, wann der nächste Zug kommt. Oder es verkündet den aktuellen Spielstand des gerade laufenden Fußballspiels. Das Beste daran: Das alles geschieht automatisch. Die Karten tauchen den ganzen Tag über immer genau dann auf, wenn man sie braucht.

Nicht nur Privatpersonen sehen den Dienst allerdings kritisch. Besonders Sicherheits-Abteilungen in Firmen haben starke Bedenken, wie u. a. [CSOOnline](#)²⁹⁵ berichtet:

Während Vertreter der Konsumenten sich über die Privatsphäre sorgen, denken Firmen über die Implikationen nach, Google Now auf dem gleichen Gerät installiert zu wissen, mit dem der Angestellte auch auf das firmeneigene Intranet oder den Mailserver zugreift. Zumindest sind Firmen daran interessiert, hier die Kontrolle zu bekommen, das Feature zu deaktivieren.

Fragt sich da jemand, ob das noch steigerungsfähig ist? Oh ja, aber sicher doch. Nach *Google Now* folgt *Google Glass*. Und dann wird auch noch aufgezeichnet, was man sieht. Wie lange man worauf schaut. Spätestens dann wird es wichtig, in den richtigen Augenblicken auch einmal abzuschalten.

Bei zukünftigen Android-Versionen dürften die Cloud-Dienste sogar noch tiefer ins System eingebunden werden. So [berichtet etwa AreaMobile](#)²⁹⁶ zu den Neuerungen in Android 4.4: Keine SMS-App mehr, das macht jetzt *Google Hangouts* nebenbei mit (Update: Ab *Lollipop* doch wieder separat). Gleiches gilt für die Galerie-App, die nun in *Google+* integriert wurde (Update: auch hier wird mittlerweile wieder zurückgerudert). Das Drucken via Cloud-Services wurde ebenfalls direkt ins System eingebunden. Des Weiteren sucht die Telefon-App bei Eingabe von Telefonnummern automatisch nach passenden Einträgen naheliegender Geschäfte (wo sie das wohl tut?) sowie Profilbildern etc. [bei Google+](#)²⁹⁷ für eingehende Anrufe, und das Mikrofon ist ständig für das Code-Wort „OK Google“ bzw. „OK Jarvis“ aktiv (zum Glück nur beim Nexus 5). Bleibt zu hoffen, dass sich so etwas auch abschalten lässt!

Wer sich für weitere kritische Lektüre zu *Google Now* interessiert, kann sich u. a. an folgende (größtenteils bereits zitierte) Artikel halten:

- [Google Now: Trading Your Privacy For The Future](#)²⁹⁸ (SeekingAlpha)

294. <http://web.archive.org/web/20141012230815/http://www.webpronews.com/google-now-do-you-want-google-using-your-information-in-this-way-2012-07>

295. <http://www.csoonline.com/article/709578/>

296. <http://www.areamobile.de/news/25645-android-4-4-kitkat-die-neuerungen-im-ueberblick>

297. <http://www.androidpit.de/google-verlinkt-profilfotos-mit-telefonnummer>



CSOOnline: Google Now draws caution amongst security experts



AreaMobile: Android 4.4 Neuerungen im Überblick



AndroidPIT: Google verlinkt Profilfotos von Google+ mit

- ["Google Now" Knows What You're Doing, Right Now²⁹⁹](#) (PMG.Com)
- ["Google Now" Knows More About You Than Your Family Does - Are You OK With That?³⁰⁰](#) (ReadWrite.Com)
- ['Google Now's' Terrifying, Spine-Tingling, Bone-Chilling Insights Into Its Users³⁰¹](#) (Forbes)
- [Google Now: Do You Want Google Using Your Information In This Way?³⁰²](#) (WebProNews, via Archive.ORG)
- [Google Now draws caution among security experts³⁰³](#) (CSOOnline.Com)

Wer stattdessen daran interessiert ist, was sich so alles mit Sprachbefehlen in *Google Now* anstellen lässt, der findet in einem [Artikel bei Go2Android³⁰⁴](#) weitere Informationen.

Hat da etwa jemand *Google Now* aktiviert, möchte es aber nach obiger Lektüre lieber wieder loswerden? Ein [FieldGuide bei Gizmodo³⁰⁵](#) beschreibt ausführlich die notwendigen Schritte:

1. *Google Now* aufrufen, nach unten scrollen, den Menü-Button (die drei übereinanderliegenden Punkte) betätigen, „Einstellungen“ auswählen.
2. Den Schalter ganz oben rechts umlegen (ausschalten). Eine Dialog-Box wird nun nach einer Bestätigung fragen – und auch die Möglichkeit bieten, den Standortverlauf gleich mit zu deaktivieren.

Damit hat man von *Google Now* komplett Ruhe. Wem das jedoch zu weit geht, der kann auch einzelne Karten/Features deaktivieren – und zwar genau dann, wenn diese „nervtötend“ auftauchen. Diese „Karten“ verfügen nämlich ebenfalls über einen Menü-Button (i. d. R. in der oberen rechten Ecke). Dessen Betätigung führt zu einer Dialog-Box, in der man sein „Desinteresse“ bekunden (diese Karte in Zukunft nicht mehr anzeigen) kann.

Wer allerdings nicht erst auf das Erscheinen einer unerwünschten Karte warten will (oder versehentlich etwas abgeschaltet hat, und dies wieder aktivieren möchte), scrollt im *Google Now* Bildschirm ganz nach unten. Dort sollte sich ein „Zauberstab“ finden, über den sich alle verfügbaren Karten (auch bereits deaktivierte) konfigurieren lassen.



Gizmodo Fieldguide:
Disable „Google Now“

-
298. <http://seekingalpha.com/article/1167171>
 299. <https://www.pmg.com/blog/google-now-knows-youre-right-now/>
 300. <http://readwrite.com/2012/06/29/google-now-knows-more-about-you-than-your-family-does-are-you-ok-with-that>
 301. <http://www.forbes.com/sites/kashmirhill/2012/07/03/google-nows-terrifying-spine-tingling-bone-chilling-insights-into-its-users/>
 302. <http://web.archive.org/web/20131003172435/http://www.webpronews.com/google-now-do-you-want-google-using-your-information-in-this-way-2012-07>
 303. <http://www.csoonline.com/article/709578/>
 304. <http://www.go2android.de/google-now-und-nexus-5-geht-eigentlich-deutschland/>
 305. <http://fieldguide.gizmodo.com/how-to-disable-google-now-on-your-android-device-1652218566>

Zwischenbilanz

Wer bis hier hin mitgelesen hat, hält mich nun mit Sicherheit für einen Paranoiker, der alles schwarz malt. Und hinter jedem Baum einen Spion sieht. Ich will das gar nicht von vornherein bestreiten – aber ein wenig korrigieren: Ich sehe, dass hinter jedem Baum ein Spion stehen könnte.

Keinesfalls möchte ich hier die Nutzung „der Cloud“ im Allgemeinen, oder gewisser „sozialer Netze“ im Speziellen verteufeln oder „madig machen“. Das Eine oder Andere nutze ich ja zugegebenermaßen selbst. Aber mit einem kritischen Blick die Dinge hinterfragen, das sollte man auf jeden Fall. Sich die Hintergründe bewusst machen. Wissen, wie es läuft – und was dahinter steht. Und dann wissend entscheiden, welche Dienste man nutzen möchte – oder, anders ausgedrückt: Wie viel Privatsphäre man bereit ist, für wie viel Bequemlichkeit aufzugeben.

Übrigens rät auch die *Stiftung Warentest* [von der Nutzung von Cloud-Speichern ab](#)³⁰⁶. Mehrere Dienste wurden unter die Lupe genommen (darunter auch *Dropbox* und *Google Drive*). Keiner schaffte eine bessere Note als „3,2“. Anwendern, die nicht auf diese Speichermöglichkeit verzichten können (oder wollen) wird dringend angeraten, die eigenen Daten *vor dem Hochladen* zu verschlüsseln.



Cloud-Speicher:
Vernichtendes Urteil
von Stiftung
Warentest

306. <http://t3n.de/news/a-484064/>

Weitere Aspekte

Soziale Netzwerke sind nicht die einzigen, denen wir unsere Daten überlassen. So manche App sammelt im Hintergrund ebenfalls fleißig – ohne dass wir genau wissen was, wann, und wozu. Und niemand kann sagen, er hätte ihnen das nicht erlaubt: Wir haben ja, von den vorinstallierten Apps einmal abgesehen, schließlich unsere Zustimmung gegeben, als wir bei ihrer Installation die Berechtigungen abgenickt haben – da dürfen wir uns jetzt nicht beschweren, dass sie unsere Kontakte und Kalenderdaten lesen, auf den Telefon- (IMEI/IMSI, Netzanbieter) und Netzwerkstatus (WLAN-Netze in der Nähe? Wo ist das Gerät eingebucht?), die Liste konfigurierter Konten, die Log-Dateien, Kurznachrichten, Besitzer-Informationen, und anderes zugreifen, und jederzeit mit den gesammelten Daten „ins Internet“ verschwinden können. Und wir nicht einmal wissen, auf welchen Servern wir die Daten letztendlich wiederfinden ...

Das Thema „Zugriffsrechte“ wurde ja bereits im Kapitel [Worauf Apps Zugriff haben](#) besprochen. Eine Übersicht über die gebräuchlichsten „Permissions“ und ihre Bedeutung findet sich überdies in Anhang [Android Permissions und was sie bedeuten](#) (sowie eine vollständigere und ständig aktualisierte Liste von [Permissions mit Erklärungen bei IzzyOnDroid](#)³⁰⁷). Daher möchte ich an dieser Stelle auch nicht weiter in die Tiefe gehen. Nur erwähnt werden sollte es, denn auch das betrifft die Privatsphäre.

Ob und warum wir auf unsere Privatsphäre achten sollten, damit befasst sich übrigens auch ein lesenswerter [Artikel bei Lifehacker](#)³⁰⁸ (leider auf Englisch) – und geht dabei auf interessante Hintergründe und Zusammenhänge ein.



Android Permissions
mit Beschreibung



Lifehacker: Why
should you protect
your privacy?

307. <http://android.izzysoft.de/applists/perms>

308. <http://lifehacker.com/5904966/>

Werbefinanzierte Apps

„No money, no honey“, heißt es für den Entwickler. Auch er muss von etwas leben – und nicht jeder Entwickler betrachtet die Erstellung von Apps als reines Hobby. Manch einer möchte daher seine erbrachte Leistung gern honoriert sehen. Da leider nicht jeder Anwender bereit ist, für selbige ein paar Cent zu investieren, muss eine Alternative her.

„Jeder Depp hat 'ne App“ – das ist auch den Betreibern von Werbe-Netzwerken kein Geheimnis mehr. Und so kommen die Beiden zusammen: Werbenetzwerke stellen fertige „Werde-Module“ bereit, die von Entwicklern lediglich in ihre Apps eingebunden werden müssen. Auf den ersten Blick eine typische Win-Win-Situation, wären da nicht gewisse Nebeneffekte.

Schauen wir uns beispielsweise einmal an, welche Voraussetzungen derartige Werbe-Module verlangen. Für MobFox und AdMob, zwei der größten Kandidaten, beschreibt dies ein [Artikel bei TechRepublik](#)³⁰⁹ (Stand: 12/2011). Diese beiden Werbe-Module fordern folgende Berechtigungen:

- INTERNET (uneingeschränkter Internetzugriff):
 - Guter Cop: Zum Laden des Anzeigen-Materials.
 - Böser Cop: Anzeigen sind Nebensache. Hier sollen fleißig Daten gesammelt, und auf die Server der Werbeindustrie zur Profil-Erstellung hochgeladen werden! Was für Daten das sein können, sehen wir ja gleich.
- ACCESS_NETWORK_STATE (Netzwerkstatus anzeigen):
 - Guter Cop: Nur zur Ermittlung, ob auch eine Netzverbindung möglich ist.
 - Böser Cop: Auslesen, mit welchem Netz der User verbunden ist. IP-Adressen und WLAN-Namen abgreifen!
- ACCESS_COARSE_LOCATION (ungefähre Standort):
 - Guter Cop: Für Standort-basierte Werbung. Was interessieren schließlich einen Anwender in Deutschland Sonderangebote von Walmart in den USA?
 - Böser Cop: Wissen, wo sich der Anwender wann und wie oft aufhält. So etwas ist für ein gutes Nutzerprofil unheimlich sinnvoll!
- READ_PHONE_STATE (Telefonstatus lesen und identifizieren):



Android apps and advertising: A bit too cozy

309. <http://www.techrepublic.com/blog/security/android-apps-and-advertising-a-bit-too-cozy/7003>

- Guter Cop: Damit dem Anwender die Werbung nicht bei Telefonaten in die Quere kommt. (Dafür wird sie allerdings gar nicht benötigt, siehe unter [Android Permissions](#) im Anhang)
- Böser Cop: Netzwerk-Anbieter ermitteln. Eindeutige Identifikation des Anwenders anhand von [IMEI](#) und [IMSI](#). Rufnummer des Anwenders feststellen. Herausfinden, mit wem er so alles telefoniert.

Die geforderten Zugriffs-Berechtigungen lassen sich vom „guten Cop“ durchaus alle positiv erklären. Sollte er also Recht haben, wäre dagegen gar nichts einzuwenden. Wie der „böse Cop“ allerdings aufzeigt, ist das Missbrauchs-Potential nicht gerade gering: Mit den so verfügbaren Daten lässt sich ein umfangreiches Anwender-Profil erstellen (und sicher auch gut verkaufen). Da mag der Entwickler der App noch so vertrauenswürdig sein: Er hat kaum Einfluss darauf, was die Werbemodule treiben. Oftmals ist ihm diese Problematik nicht einmal bewusst. Und nicht nur diese Problematik, denn es kommt noch schlimmer:

Da App und Werbe-Modul aus Android-Sicht eine Einheit bilden, erhält das Werbemodul auch alle Berechtigungen, die der Entwickler für die App vorgesehen hat. Darf die App also z. B. auf Kalender und Adressbuch zugreifen, stehen Termine und Kontakte auch dem Werbemodul offen.

Ist das nun lediglich ein theoretisches Risiko – oder müssen wir uns wirklich Sorgen machen? Wo solches Potential lauert, bleibt es sicher nicht lange ungenutzt. Und so [schreibt FirstPost von einer Studie](#)³¹⁰: 100.000 Apps wurden hinsichtlich der von ihnen verwendeten Werbemodule untersucht. 48% sammelten die Standort-Informationen, 18,5% die IMEI, 4% sogar die Telefonnummern. Einzelne Werbemodule wurden dabei ertappt, Anruf-Protokolle auszulesen, auf Kalender und Kamera zuzugreifen, oder dynamisch weiteren Programm-Code nachzuladen. Die Schlussfolgerung ist daher naheliegend. Aus dem Artikel übersetzt:

Diese neuen Ergebnisse zeigen eine Schwachstelle im Geschäftsmodell hinter Apps auf, so Jiang (Xuxian Jiang leitete die Untersuchung, Anm. d. Ü.). Entwickler sind auf die Einnahmen über die Werbemodule angewiesen, um ihre Apps gratis zur Verfügung stellen zu können – aber sie haben keinerlei Kontrolle darüber, was diese Module tun. „Das aktuelle Modell des Einbettens von Werbemodulen in mobilen Apps zu deren Finanzierung stellt eine Gefahr der Sicherheit und der Privatsphäre dar. Diese Werbemodule können prinzipiell auf dieselben Berechtigungen zugreifen wie die App, in der sie eingebettet sind. Und gewisse Werbemodule könnten sie zu unerwünschten Zwecken missbrauchen.“



Android Ad networks found accessing users private data

310. <http://www.firstpost.com/tech/a-250713.html>

Für den eingangs genannten Artikel bei TechRepublik wurde übrigens auch eine Befragung durchgeführt. Den Teilnehmern wurden einige Beispiele eingebblendeter Werbung gezeigt. Anschließend wurde ihnen erklärt, wie man die von einer App geforderten Berechtigungen liest. Zuletzt kam die Frage: Kann das Werbemodul (wörtlich: die Werber, also die Firmen dahinter) auf sämtliche Informationen zugreifen, die der App selbst zur Verfügung stehen? 16% der Befragten antworteten mit „Nein“, 42% wussten keine Antwort. Nur 42% der Teilnehmer sagten „Ja“. Wie wir gesehen haben, lag die letzte Gruppe – leider – richtig.

Nur beschränkt sich dieses Problem nicht auf „wenige Einzelfälle“. Häufig sind selbst die Programmierer ahnungslos, welchen Risiken sie ihre User aussetzen – wie Spiegel unter Berufung auf eine FireEye-Untersuchung³¹¹ beschreibt. Heise sagt es angesichts der gleichen Quelle mit einfachen Worten: Android-Adware soll mehr als 200 Millionen Nutzer gefährden³¹². Wer an weiteren Details und Quellen zu diesem Thema interessiert ist: In einem Artikel bei Stack Exchange³¹³ habe ich einiges zusammengetragen. Sollte hingegen jemand nur neugierig sein, was so alles in der Praxis „getrackt“ wird: Der werfe einmal einen Blick in diese Seite des Taptica Wiki³¹⁴.

Ebenfalls interessant: How many apps spy on us?³¹⁵ Für diesen Artikel wurden im August 2014 die 400 Top-Apps für Android unter die Lupe genommen. Kurz abstrahiert:

- 82% der gratis und 49% der Bezahl-Apps tracken den Standort des Users
- 30% resp. 14% greifen auf das Adressbuch zu
- 88% vs. 65% greifen auf IMEI/UDID zu
- 73% bzw. 43% teilen Daten mit sozialen Netzwerken
- 56% der kostenlosen und 29% der gekauften Apps versenden App- und User-Daten mit „Crash Reports“



ASE: risks involved with each granted application permission type



Taptica: Advertiser Integration



EntonDigital: How many apps spy on us?

311. <http://www.spiegel.de/netzwelt/apps/-a-927188.html>
312. <http://www.heise.de/-1977035.html>
313. <http://android.stackexchange.com/q/44385/16575>
314. http://wiki.taptica.com/index.php/Advertiser_integration
315. <http://www.etondigital.com/security-risks-popular-apps/>

Wie kann man sich schützen?



AppBrain Ad
Detector

Zuerst einmal gilt es, mögliche Kandidaten aufzuspüren – wofür sich mehrere Helferlein gern zur Verfügung stellen. In Sachen Werbemodule dürfte AppBrain Ad Detector³¹⁶ besonders interessant sein: Wie der Name es richtig vermuten lässt, hat sich diese App auf das Aufspüren von Werbemodulen spezialisiert. Dabei wird auch aufgezeigt, was diese im Einzelnen tun. Lookout Ad Network Detector³¹⁷ informiert zusätzlich über das Verhalten der jeweiligen Netzwerke.



Die „Treffermenge“ war in meinem Kurzvergleich identisch. Während *AppBrain* die schönere Oberfläche bietet, finden sich bei *Lookout* jedoch die detaillierteren Informationen – einschließlich der Möglichkeit eines „Opt-Out“, so denn das betroffene Werbenetzwerk diese bietet. Ein Vorteil des *AppBrain Ad Detectors*: Hier werden auch neu eintreffende (also neu installierte oder aktualisierte) Apps automatisch geprüft; bei Bedenken erfolgt ein entsprechender Hinweis im Benachrichtigungs-Bereich (aka „Notification Area“).

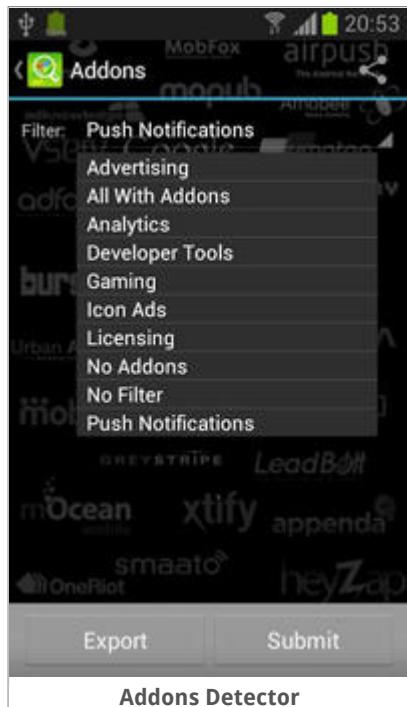
The screenshot shows two side-by-side application interfaces. On the left is the 'AppBrain Ad Detektor' app, which lists various privacy concerns (Bedenken) for the 'Business Calendar Free' app. These include:

- 7 Zugriff auf Kontakte
- 6 Zugriff auf Benutzerkonten
- 4 Zugriff auf Nachrichten
- 2 Zugriff auf Browserverlauf
- 2 Kann Kosten verursachen
- 8 Kann Ort bestimmen

Each concern is accompanied by a warning icon. On the right is the 'Business Calendar Free' app's settings screen, specifically the 'Behavior' section. It states: "Applications using the AdMob ad network are capable of displaying standard in-app ads." Below this is the 'Privacy' section, which provides a detailed explanation of AdMob's data collection practices, mentioning Android ID, build version, screen resolution, and location information. At the bottom of the 'Privacy' section is an 'Opt out' button. A large 'Opt-out' button is also visible at the bottom of the 'AppBrain' interface.

Mit *AppBrain Ad Detector* und *Lookout Ad Network Detector* fühlt man den Werbe-Modulen auf den Zahn

316. <https://play.google.com/store/apps/details?id=com.appspot.swisscodemonkeys.detector>
 317. <https://play.google.com/store/apps/details?id=com.lookout.addetector>



Wer sich nicht allein auf Werbe-Module konzentrieren will, greift vielleicht eher zu [Addons Detector](#)³¹⁸ – welcher sich auch mit Lizenz-Modulen, Analytics, und weiteren auskennt.

Hat man „furchterregende Übeltäter“ entdeckt, stellt sich die Frage, wie man mit ihnen umgeht. Natürlich kann man die betroffenen Apps einfach deinstallieren – das wäre zwar die einfachste, aber nicht unbedingt die wünschenswerteste Lösung. Ein vernünftiger erster Schritt, so man die App weiter nutzen möchte, ist ein Blick in den Google Playstore: Gibt es evtl. eine Kaufversion, die ohne das gefährliche Addon auskommt? Die paar Cent tun niemandem weh. Bei fehlender Kreditkarte hilft die Google-Play Guthabenkarte oder eine Anfrage beim Entwickler, der eventuell auch eine alternative Bezahlmöglichkeit sieht. Bei der Gelegenheit sollte man ihn auch gleich auf den Grund aufmerksam machen – er könnte durchaus zu

jenen 58% gehören, denen dieser Umstand noch gar nicht bewusst ist. In diesem Fall schaut er sich ggf. nach einem weniger gefährlichen Werbemodul um.

Greift all dies nicht: Auch andere Mütter haben schöne Töchter. Im Playstore finden sich mit Sicherheit weitere Alternativen. Und auch wenn diese etwas kosten: Ein paar Cent sollte einem die Privatsphäre schon Wert sein. Findet sich auch dort nichts, gibt es noch die alternativen Märkte.

Bei Kombinationen aus besonders aggressiver Werbung (Statusleiste, Icons auf dem Homescreen, Lesezeichen) sollte man darüber hinaus nicht vergessen, dies [Google zu melden](#)³¹⁹. Die [aktuellen Richtlinien des Google Play Store](#)³²⁰ verbieten solcherlei Aktivitäten nämlich ausdrücklich:

- *Anzeigen dürfen weder die Benutzeroberfläche einer App noch Betriebssystembenachrichtigungen oder -warnungen simulieren oder nachahmen.* (trifft auf viele Statusleisten-Werbung zu)



Addons Detector



Unangemessene Apps an Google melden



Google Play Richtlinien

318. <https://play.google.com/store/apps/details?id=com.denper.addondetector>

319. <https://support.google.com/googleplay/android-developer/contact/takedown>

320. http://play.google.com/intl/ALL_de/about/developer-content-policy.html

- *Interstitial-Anzeigen dürfen nur innerhalb der jeweiligen App geschaltet werden.* (selbstredend: „interstitial“ bedeutet soviel wie „dazwischenliegend“; die Rede ist also von Werbe-Einblendungen)
- *Mit Ihrer App verbundene Anzeigen dürfen keine anderen Apps oder deren Anzeigen beeinträchtigen.* (Der Homescreen ist eine solche „andere App“, gleiches gilt für die Browser-Lesezeichen)

Eine Sache hätte ich fast vergessen: Die Abo-Fallen, in die so mancher durch (versehentliches) Antippen eines Werbebanners bereits geraten ist. Davor schützt die sogenannte **Drittanbietersperre**³²¹, wie die Sperrung des „mobile payment“ für solche Fälle genannt wird. Nach § 45d Abs. 3 TKG sind die Mobilfunkanbieter seit dem 10.5.2012 gesetzlich verpflichtet, einer entsprechenden Forderung seitens ihrer Kunden *unentgeltlich* nachzukommen. Außerdem ist es u. U. auch hier wieder angebracht, die entsprechende App bei Google zu melden. Die genannten Richtlinien besagen nämlich auch: *Interstitial-Anzeigen müssen einfach und gut sichtbar geschlossen werden können, ohne dass dies Nachteile für den Nutzer oder einen ungewollten Klick zur Folge hat.*



Drittanbietersperre

321. <http://www.drittanbietersperre.com/>

Was bringen sichere Apps, wenn die Schnüffler ohnehin schon im System sitzen?

Eine gute und berechtigte Frage. Nur leider wird bei der Suche nach einer Antwort die Flinte zu oft am Lauf angefasst, womit die Schnüffler am Abzug bleiben. Die am meisten verbreitete Antwort auf diese Frage ist nämlich die, auf welche der Erfolg der ganzen Schnüffelei beruht: „Da kann ich es ja gleich ganz bleiben lassen.“ Das ist, als würde man die Wohnungstür sperrangelweit offen stehen lassen, nur weil ein Einbrecher das Schloss ohnehin knacken kann. Anstatt auch noch auf die „letzte Stufe“ (sichere Apps) zu verzichten, sollte man zumindest die Sicherheit in den darunterliegenden Schichten erhöhen (eine Kette an der Tür anbringen). Soweit uns das möglich ist. Aber welche Möglichkeiten haben wir überhaupt?

Nach den aktuellsten Skandalen werde ich mir zumindest in absehbarer Zeit kein Motorola-Gerät mehr kaufen (siehe [weiter vorn](#)). Ich bin mir sicher, das sehe nicht nur ich so. Ein weitreichender Boykott sollte Hersteller dreimal überlegen lassen, ob sie einen solchen für ihre Geräte riskieren wollen. Das ist ein Schritt, den wirklich jeder gehen kann.

Weitere Schritte hängen davon ab, wie technisch versiert man ist. Man kann auf die Benutzung verschiedener vorinstallierter Apps verzichten, und stattdessen auf als sicher geltende Alternativen umsteigen; ein Beispiel wäre, die vorinstallierte Mail-App durch *K-9 Mail* zu ersetzen. Open Source Apps bieten sich hier in erster Linie an, da der Quellcode von jedem einsehbar ist. Handelt es sich darüber hinaus auch noch um eine viel genutzte und weit verbreitete App, steigert das die Chancen, dass andere Entwickler von dieser Chance regen Gebrauch machen. Etwaige Sicherheitslücken werden dann i. d. R. auch recht schnell gestopft.

Wer etwas versierter ist, kann seinen Androiden auch [rooten](#). Das von Herstellern gern dagegen vorgebrachte Argument des Garantieverlusts ist Mumpitz, der [Gewährleistungsanspruch](#)³²² bleibt bestehen. Sollte daher die Hardware Defekte aufweisen, haftet dafür nach wie vor der Hersteller (es sei denn, der Anwender hat nachweislich z. B. die CPU durch Übertakten in eine Kochplatte verwandelt). Auf einem gerooteten Gerät lassen sich dann auch, je nach Firmware, entsprechende [Sicherheits-Apps](#) zum Einsatz bringen – wie beispielsweise *LBE* (closed source, und daher vielleicht ein wenig suspekt), *OpenPDroid* (nur mit bestimmten Firmwares), und *XPrivacy* – mit denen sich die Rechte der installierten Apps kontrollieren lassen. Oder man friert „fragwürdige



Wikipedia:
Gewährleistung

322. <http://de.wikipedia.org/wiki/Gew%C3%A4hrleistung>

„System-Apps“ beispielsweise mit *Titanium Backup* einfach ein, bzw. löscht sie gleich vollständig (nachdem man sich durch Einfrieren davon versichert hat, dass das keine Nebenwirkungen hat, und zur Sicherheit dennoch ein Backup davon erstellt). Alles kein Hexenwerk.

Traut man sich zu, noch einen Schritt weiter zu gehen, bietet sich die Installation eines renommierten *Custom-ROM* (wie etwa *CyanogenMod*) an. Mit einem solchen kann man relativ sicher sein, dass keine herstellerspezifischen Schnüffel-Funktionen irgendwo eingebaut sind. Auch die Nutzung anderer spezieller Sicherheitsmaßnahmen, wie etwa *VPN* sind denkbar. Verschlüsselung verwenden, wo es geht: *HTTPS* beim Surfen, analog SSL Verschlüsselung bei der Mail-Übertragung, vielleicht auch PGP Verschlüsselung von Mailinhalten (wo es sich anbietet). Je schwieriger wir es den Schnüfflern machen, desto weniger können sie schnüffeln – der Aufwand wird irgendwann einmal auch für sie zu hoch. Und wer dem noch die Krone aufsetzen will, der befreit seinen Androiden gleich ganz von den proprietären Google-Apps sowie weiteren „Zwangsbeglückern“ – und setzt ganz auf Open Source, wie in meiner Artikelreihe *Android ohne Google*³²³ beschrieben.



IzzyOnDroid:
Android ohne
Google

Weitere Ideen zivilen Ungehorsams lassen sich sicher aus anderen Bereichen übertragen: So kenne ich genügend Leute, die eine Payback- oder ähnliche Rabatt-Karte nutzen. Einige davon haben Zweit- und Drittarten, die sie mit Bekannten und Verwandten teilen. Das gibt ein locker-flockiges Kundenprofil, bei dem die Auswertung sicher Spaß macht! Analogien dazu sollten sich bestimmt auch im Umgang mit Androiden finden lassen – etwa den „mobilen Hotspot“ dazu zu nutzen, „über Kreuz“ zu surfen. Oder auch ohne mobilen Hotspot, indem man mit Personen des Vertrauens hin und wieder das Gerät tauscht. Ich weiß, das klingt jetzt wirklich ein wenig zu kompliziert, als dass man es im Alltag regelmäßig umsetzen könnte – doch hier kann jeder seiner Fantasie freien Lauf lassen, und entsprechende Ideen dann auch mit anderen Teilen.

Abschließend noch ein paar Worte an diejenigen, die „nichts zu verbergen“ haben. Auch wenn die „originalen Datenabgreifer“ ja vielleicht „gute Absichten“ hatten: Wie sicher sind die Daten auf deren Servern? „Sicher ist, dass nichts sicher ist. Selbst das nicht.“ (Joachim Ringelnatz) Schnell gelangen „die Falschen“ an die Daten. Wenn schließlich das eigene Bankkonto plötzlich leergeräumt, das Mailkonto wegen Spamversand gesperrt, und über die Konten bei Facebook & Co die „interessantesten Informationen“ verbreitet wurden – **dann** hat man wirklich nichts mehr zu verbergen.

323. <http://android.izzysoft.de/articles/named/android-without-google-1>

Gibt es noch mehr zu beachten?

Das Thema lässt sich im Rahmen dieses Buches definitiv nicht erschöpfend behandeln. Obiges hat allenfalls einige Kernpunkte angerissen – und zu viele weitere Details würden den Umfang definitiv sprengen. Wer an letzteren interessiert ist, den möchte ich stattdessen auf die [Webseite mit dem Zusatz-Material zum Buch](#)³²⁴ hinweisen. An dieser Stelle daher nur noch ein paar Andeutungen:

So mancher meint, er (oder sie) hätte ja schließlich „nichts zu verbergen“. Doch sogar Muster-Menschen kann es erwischen. So berichtet Michael Blume in seinem Artikel [Sollten sich „anständige Bürger“ wegen der Überwachung sorgen?](#)³²⁵ aus eigener Erfahrung, wie sich Daten gegen den „Besammelten“ verwenden lassen. Und man trotz späterer Aufklärung einschließlich Dementis von höchster Stelle den „Schandfleck“ nicht mehr los wird.

Judith Horchert fragt sich in ihrem Artikel [Das Gefühl der Überwachung](#)³²⁶ unter anderem: „Interessiert ein digitales Lieschen Müller die Geheimdienste?“ Dabei denkt sie darüber nach, was nicht unbedingt zusammengehörige Tweets, Mails, und Facebook-Nachrichten für einen Eindruck erwecken – wie sie ein ganz anderes Bild ergeben könnten, als es die Realität war. Schließlich kommt sie zu der Einsicht: „Ich habe etwas zu verbergen“. Denn: Selbst wenn wir alles erklären können – wollen wir das?

Zu guter Letzt kann man bereits in die „Komplett-Überwachung“ geraten, nur [weil man in der falschen Straße wohnt](#)³²⁷. Wem das nicht reicht, der findet beim Spiegel noch weitere [fünf schlechte Argumente](#)³²⁸.



IzzyOnDroid: Privacy
– Gibt es noch mehr zu beachten?



Sollten sich
anständige Bürger
wegen der
Überwachung
sorgen?



Spiegel: Das Gefühl
der Überwachung



Spiegel: Fünf
Argumente gegen
die Verharmloser

324. <http://android.izzysoft.de/books.php?topic=privacy>

325. <http://www.scilogs.de/chrono/blog/natur-des-glaubens/usa/2013-07-06/sollten-sich-anständiger-b-rger-wegen-der-berwachung-sorgen-ein-erfahrungsbericht-aus-den-schattenkriegen>

326. <http://www.spiegel.de/netzwelt/netzpolitik/a-908245.html>

327. <http://www.spiegel.de/netzwelt/netzpolitik/a-913688.html>

328. <http://www.spiegel.de/netzwelt/netzpolitik/a-911202.html>

Von einer weiteren Folge der gesammelten Skandale aus 2013+ berichtet das Kojote-Magazin. So sollen die Krankenakten aller Paranoia-Patienten überprüft werden³²⁹. Beruht die Diagnose auf der starken Überzeugung des Patienten [...], von dunklen Mächten permanent überwacht, abgehört und/oder elektromagnetisch kontrolliert zu werden, heißt das ab sofort nicht mehr „paranoid“, sondern „informiert“.

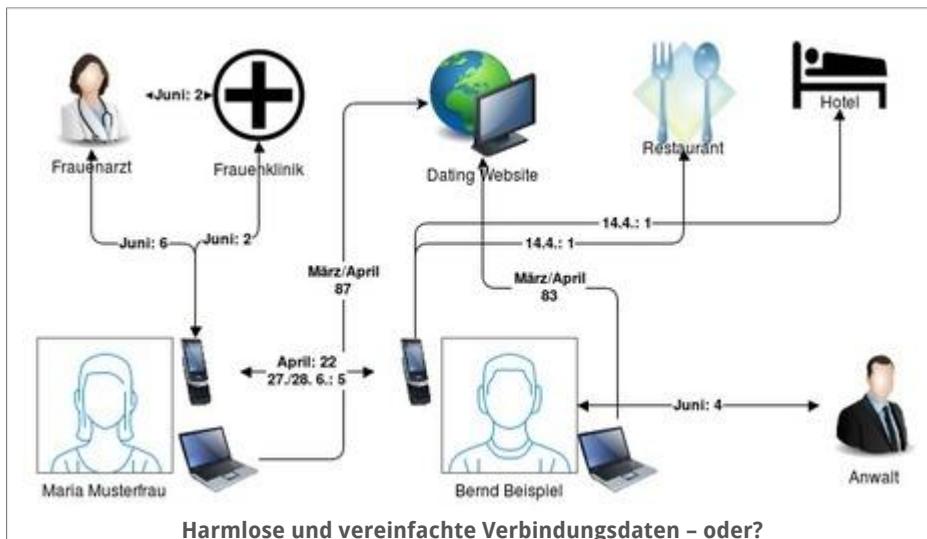


Kojote-Magazin:
Krankenakten aller
Paranoia-Patienten
werden wegen US-
Überwachung
überprüft

329. <http://www.kojote-magazin.de/2013/krankenakten-aller-paranoia-patienten-werden-ueberprueft/8624/>

Es sind doch nur Verbindungsdaten!

Abschließend noch etwas zu den „harmlosen Verbindungsdaten“, mit denen sich ja so wenig über unser Privatleben aussagen lässt. Es geht ja schließlich nicht um Gesprächsinhalte, sondern nur um Verbindungsdaten. Dann schauen wir uns doch einmal „nur Verbindungsdaten“ an, die auch noch verallgemeinert wurden: Was mag wohl hier passiert sein?



Verbindungsdaten sind oftmals nicht nur aussagekräftiger als Kommunikations-Inhalte³³⁰, sondern lassen sich auch wesentlich einfacher automatisiert aufarbeiten. Im Gegensatz zu letzteren liegen sie nämlich in strukturierten Formaten vor: Telefonnummern, E-Mail-Adressen, Zeit und Ort, und mehr lassen sich so auf einfachste Weise maschinell verknüpfen. So braucht es bei obiger Beispiel-Grafik kaum irgendwelcher Kommunikations-Inhalte um zu erraten, was dort vorgefallen sein könnte – und in welche Richtung es sich wohl weiterentwickeln wird. Dabei habe ich sogar große Teile der verfügbaren Metadaten außen vor gelassen: Wo waren Maria und Bernd bei welchem Telefonat, welches Restaurant und Hotel hat Bernd kontaktiert, wer ist die Frauenärztin von Maria (und wie oft ruft sie dort sonst an)? Alles inklusive. Und mehr. Ich zitiere einmal aus zuvor verlinktem Artikel, wiederum in vereinfachender Übersetzung:



Netzpolitik.Org:
Warum
Verbindungsdaten
noch
aussagekräftiger
sind als
Kommunikations-
Inhalte

330. <https://netzpolitik.org/2013/vorratsdatenspeicherung-warum-verbindungsdaten-noch-aussagekraefriger-sind ALS-kommunikations-inhalte/>

Kurz gesagt: Aggregierte Telefon-Metadaten erlauben es der Regierung, soziale Graphen zu erstellen, sowie deren Entwicklung und Kommunikations-Muster über Tage, Wochen Monate, oder sogar Jahre zu studieren. Die Analyse von Metadaten kann das Wachsen und Fallen intimer Beziehungen, die Diagnose einer lebensgefährlichen Erkrankung, eine sich anbahnende Fusion oder Übernahme von Unternehmen, die Identität eines voraussichtlichen Whistleblowers, die soziale Dynamik einer Gruppe, oder sogar den Namen einer anonymen Person offenbaren.

Daran sollte man übrigens auch denken, wenn der Begriff „Vorratsdatenspeicherung“ fällt. Denn auch bei diesen handelt es sich schließlich „nur um Verbindungsdaten“, und wir haben ja bekanntlich „nichts zu verbergen“.

FRAGEN AUS ALLTAG UND PRAXIS

Google Account

Wozu brauche ich einen Google-Account?

Klar kann man seinen Androiden auch ohne Google-Account benutzen. Bei einigen Tablets hat man zunächst nicht einmal eine andere Wahl. Doch manche Dinge funktionieren einfach nicht ohne. Zum Beispiel der *Google Play Store*, über den die meisten Apps kommen. Auch die Synchronisation von Kontakten und Terminen ist an einen Google-Account gebunden – zumindest, wenn man die Bordmittel benutzen möchte.

Kann ich einen Account mit mehreren Geräten nutzen?

Das ist problemlos möglich. Und das Schöne dabei: Auch die mit dem Account gekauften Apps lassen sich auf all diesen Geräten parallel nutzen, denn der Kauf ist ja an den Account gebunden.

Darüber hinaus zeigt die [Google Play Website](#)³³¹ bei jeder App an, mit welchem Gerät sie kompatibel ist – und gibt dem Anwender dann die Möglichkeit, sie dort zu installieren. Auch Kalender und Kontakte lassen sich auf allen mit diesem Account verknüpften Geräten synchronisieren (so man diese Funktion aktiviert hat). Kurzum: Die Sache scheint geradezu dafür ausgelegt.

Aufpassen sollte man allerdings mit Datensicherungen (auch diese lassen sich ja auf den Google-Servern speichern) – hier wird es eventuell etwas tricky, die Daten verschiedener Geräte auseinander zu halten (Details dazu in einem [Artikel bei Stack Exchange](#)³³²).



Google Play

Stack Exchange:
Multiple devices
using the same
account - what
happens on Restore?

Wie kann ich meinen Google-Account ändern?

Offensichtlich soll der Anwender dies nicht tun – denn die entsprechende Stelle in der Konfiguration lässt es nicht zu. Natürlich klappt das nach einer „[Werksrückstellung](#)“, da dann ohnehin alles neu eingerichtet werden muss. Das

331. <http://play.google.com>

332. <http://android.stackexchange.com/q/42245/16575>

ist aber nicht immer gewünscht: Auch alle Apps müssen dann nämlich neu installiert und konfiguriert werden.

Mit einem kleinen Trick geht es aber auch ohne „Werksrückstellung“: Vom Home-Screen aus geht man dazu in die Maske *Einstellungen* › *Anwendungen* › *Anwendungen verwalten*, und wählt nacheinander folgende Anwendungen an, um deren Daten und Caches zu löschen: GMail, Google Apps, GTalk (Google Talk), GTalk Services, IM und evtl. noch Google+ und Google Storage (auch „Speicherplatz bei Google Mail“ genannt; was nicht da ist, überspringt man halt). Jetzt hat der Androide seine „Kontoverbindung“ vergessen, und man kann diese wieder neu einrichten – mit den gewünschten „anderen Daten“, oder indem man einen neuen Account erstellt.

Zugegeben ein wenig umständlich. Und so hat offensichtlich jemand Einsehen gehabt, denn bei aktuellen Android-Versionen geht es wesentlich einfacher: Man geht zu *Einstellungen* › *Konten & Synchronisation*, wählt das zu entfernende Google-Konto aus, tippt auf die Menü-Taste, und nutzt den auftauchenden Eintrag „Konto entfernen“. Anschließend fügt man das neue Konto im erstgenannten Menü wieder hinzu.

Bei der ganzen Sache bitte bedenken: Gekaufte Apps sind, [wie bereits erwähnt](#), an den Account gebunden, mit dem sie gekauft wurden. Entfernt man diesen vom Gerät, funktionieren sie u. U. nicht mehr (für gratis Apps gilt dies natürlich nicht).

Kann ich die Youtube App mit einem alternativen Account verwenden?

Wer über mehrere Accounts verfügt (z. B. einen für GMail und Android, sowie einen anderen für Youtube), und diese gern getrennt halten möchte, kann dies auf recht einfache Weise tun:

1. Youtube-App starten
2. Aus dem Menü „Abmelden“ („Sign out“) wählen
3. Aus dem Menü „Anmelden“ („Sign in“) wählen
4. Es sollte an dieser Stelle eine Box zur Account-Auswahl erscheinen. Sofern der gewünschte Account aufgeführt ist, kann er nun gewählt werden. Alternativ lässt sich von hier aus auch ein neuer Account anlegen.

Google Play Store

Ich finde den *Play Store* auf meinem Gerät nicht!

Dies ist hin und wieder insbesondere bei manchen Tablets „normal“: Ein Android-Gerät muss bestimmte Voraussetzungen erfüllen, damit es mit den Google-Apps ausgeliefert werden darf. Das heißt jetzt aber nicht, dass „Hopfen und Malz verloren“ sind: Eine Suche in den einschlägigen Foren (viele haben gerätespezifische Unterforen), oder auch bei Google mit den Begriffen „Google Apps“ <Gerätename>, führen nicht selten zu einer Lösung. Diese heißt dann in der Regel: Die gapps.apk händisch herunterladen, und auf dem Gerät installieren. Was jedoch (zumindest für die Playstore-App) wiederum root voraussetzt. Anmerkung: Wer sich zutraut, selbst die richtige Version zu erkennen, findet die GApps auch direkt bei [TeamAndroid](#)³³³ (weitere Quellen sind im [google-apps Tag-Wiki](#)³³⁴ bei Stack Exchange genannt). Die gewählte ZIP-Datei muss dabei zur auf dem Gerät laufenden Android-Version passen.

Warum finde ich die App im *Play Store* nicht?

Hier ist die Antwort ähnlich zur vorigen – nur dass es diesmal einen direkten Bezug zur App hat. Entweder hat der Entwickler etwas vergessen anzugeben (das ist nicht gerade selten), oder aber das fragliche Android-Gerät wird tatsächlich nicht unterstützt: Zu kleines Display (auf einem Motorola Flipout HD-Videos abspielen? Macht bestimmt Spaß), zu schwacher Prozessor (CPU-intensive Spiele wie Asphalt überfordern so manches kleine Gerät), oder die installierte Android-Version ist zu niedrig. Eine Anfrage beim Entwickler schafft häufig Abhilfe, sofern letztere möglich ist.



Google Apps bei TeamAndroid

Kann ich eine solche „inkompatible App“ trotzdem irgendwie installieren?

Das ist möglich – auf verschiedene Arten und Weisen. Selbst wenn die Installation klappt, garantiert dies jedoch nicht, dass auch die App funktioniert.

Zunächst muss dafür unter *Einstellungen* › *Sicherheit* das Häkchen bei „Unbekannte Quellen“ gesetzt werden. Auf gerooteten Geräten kann nun die App

333. <http://www.teamandroid.com/gapps/>

334. <https://android.stackexchange.com/tags/google-apps/info>



[Market Helper](#)³³⁵ zum Einsatz kommen – die es allerdings nicht im *Playstore* selbst gibt. Stattdessen besorgt man sich die .apk Datei über den Link von den XDA-Developers, und installiert sie sodann händisch. Beim ersten Start muss sie nun eingerichtet werden; die Dialoge sind eigentlich selbsterklärend, finden sich aber in einem [Blogbeitrag bei AndroidPIT](#)³³⁶ ausführlich erklärt.

Nun kann es losgehen: Je nach getätigten Einstellungen glaubt der *Playstore* nun beispielsweise, man sei mit einem Nexus 4 über AT&T in den USA unterwegs, und gibt die entsprechenden Apps frei.

Oh, kein [root](#) auf dem Gerät? Eine Lösung gibt es eventuell dennoch:

- Vielleicht gibt es die App in einem anderen Market? Eine kleine Übersicht dazu gibt [ein Artikel bei IzzyOnDroid](#), aber auch die weiter unten verlinkte Liste.
 - Der Entwickler bietet sie u. U. auf seiner Seite zum Download an.
 - Wenn nicht: Einfach einmal anschreiben (eine Kontakt-Möglichkeit ist auf der Playstore-Seite der App angegeben).
- Android Markets: Wie sicher sind alternative Quellen?
- Man installiert sich [Raccoon](#)³³⁷ auf dem Computer, erstellt sich mit dessen Companion-Programm [DummyDroid](#)³³⁸ ein „kompatibles Gerät“, und lädt dann das passende APK herunter.
- Klar findet man sie eventuell auch „auf dem Schwarzmarkt“ (per Google-Suche oder über Tauschbörsen). Hier gilt jedoch: Besser die Finger davon lassen. Bei derartigen Quellen sind Apps gern einmal mit „Zusatz-Funktionalitäten“ versehen, auf die man besser verzichtet (Malware, Schnüffel-Soft, etc.).

Weitere Market-Helferlein finden sich übrigens bei *IzzyOnDroid* in der Liste [App Markets](#)³³⁹.

Wie kann ich einen Playstore-Download abbrechen?

Ärgerlich, wenn man ein Update anstößt – und plötzlich stellt man fest: Wie bitte? 257 MB Download? Im heimischen WLAN ja kein Problem. Unterwegs mit „mobilen Daten“ kann so etwas jedoch schnell das gesamte Datenvolumen verschlingen, ohne dass der Download dabei fertiggestellt wird. Und ab dann geht es nur noch gedrosselt im Schneckentempo weiter. Besser also den Download zunächst abbrechen.

335. <http://forum.xda-developers.com/showthread.php?t=2146216>

336. <http://www.androidpit.de/google-play-store-inkompatible-apps-market-helper-installieren>

337. <https://github.com/onyxbits/Raccoon>

338. <http://www.onyxbits.de/dummydroid>

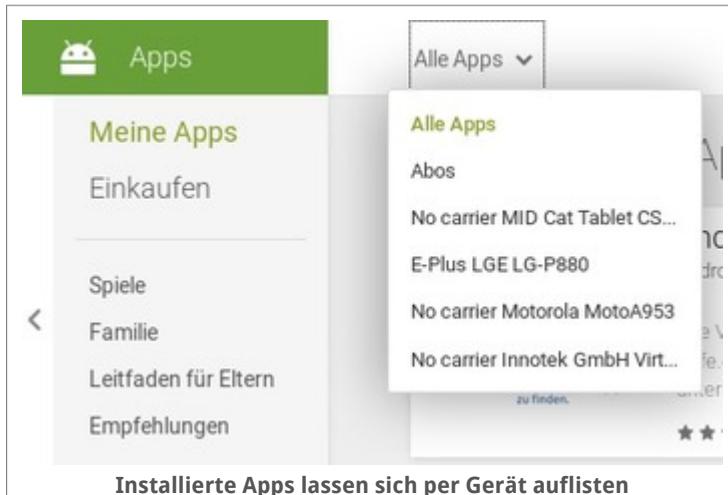
339. http://android.izzysoft.de/applists/category/named/apps_markets

Dazu öffnet man die *Playstore* App, betätigt das „Hamburger-Menü“ in der oberen linken Ecke, und wählt den Punkt „Meine Apps“ aus. Hier findet sich am Kopf des Bereiches „Herunterladen“ ein mit „Stopp“ beschrifteter Button, den es nun mit den Wurstfingern zu treffen gilt – und Ruhe ist.

Kann ich mir im Browser am PC anzeigen lassen, welche Apps ich auf welchem Gerät installiert habe?

Auch das ist möglich. Der Weg dorthin führt entweder über das [Google Dashboard](#)³⁴⁰, wo man zum Eintrag „Play Store“ scrollt, und dort auf „Apps anzeigen“ klickt – oder aber, so man bereits mit seinem Google-Account angemeldet ist, direkt über den [Link zu den eigenen Apps](#)³⁴¹.

Auf der Seite mit den eigenen Apps findet sich oben links eine Klappbox, die mit „Alle Apps“ beschriftet ist. Öffnet man diese, lässt sich das gewünschte Gerät auswählen – und die auf diesem Gerät installierten, im Playstore verfügbaren Apps werden aufgelistet.



Google Dashboard



Eigene Apps

Hilfe, der *Play Store* spinnt!

Sowas tut der in der Tat: Downloads starten nicht oder werden nicht abgeschlossen, Installationen schlagen fehl, oder die Server des *Play Store* werden erst gar nicht erreicht (Anmelde-Probleme). In der Regel liegt das in der Tat am *Play Store* bzw. den Google Servern, daher lohnt sich in solchen Fällen ein

340. <https://www.google.com/settings/dashboard?hl=de>

341. <https://play.google.com/apps>



Monitor

Blick auf deren **Monitor**³⁴², das so genannte „Dashboard“. Steht dort alles auf „grün“, zur Sicherheit noch im Forum vorbeigeschaut, ob gerade wieder eine Epidemie ausgebrochen ist (und auch andere betroffen sind).

Ist auch das nicht der Fall, ist ein lokales Problem naheliegend. Jetzt gibt es mehrere Möglichkeiten, die einfach in dieser Reihenfolge abgearbeitet werden können. Nach jedem Schritt ist natürlich zu Prüfen, ob das Problem damit schon beseitigt ist – dann kann man sich die folgenden Schritte nämlich sparen:

- In der *Play Store* App auf die „Downloads“ Seite gehen, lange (mindestens ein bis zwei Sekunden) auf die Fortschrittsanzeige „drücken“, und aus dem sich nach dem Loslassen öffnenden Menü „Download abbrechen“ auswählen. Dann erneut versuchen, die App zu installieren.
- Ein Geräteneustart (Herunterfahren, Ausschalten, evtl. zur Sicherheit noch den Akku rausnehmen, kurz warten, ggf. Akku wieder rein, neu Starten).
- Datum, Uhrzeit und Zeitzone manuell einstellen (kann später wieder auf Automatik zurückgestellt werden)
- Google Talk starten, sich dort abmelden, es wieder starten (und dann beenden)
- Unter *Einstellungen > Anwendungen > Anwendungen verwalten* der *Play Store* App den Cache leeren
- An gleicher Stelle: „Stoppen erzwingen“, dann „Daten löschen“ (damit sind aber auch die Einstellungen für Auto-Updates zurückgesetzt).
- Wieder an der gleichen Stelle: „Updates deinstallieren“. Die App aktualisiert sich allerdings selbst recht bald wieder; dennoch kann dies helfen.
- Analog zu obigem: Den Cache der „Google-Play-Dienste“ löschen.
- Prüfen, ob man etwa versehentlich den „Download-Manager“ deaktiviert hat (*Einstellungen > Anwendungen > Anwendungen verwalten*, „Download-Manager“ aufsuchen; zeigen dessen Details einen Button „Aktivieren“, diesen betätigen).
- Google-Konto entfernen und wieder hinzufügen: Zu *Einstellungen > Konten* gehen, das mit dem Playstore benutzte Konto auswählen, Menü-Taste drücken (bzw. die „Action-Bar“), „Konto entfernen“ wählen. Gerät neu starten, und Konto via *Einstellungen > Konten > Konto hinzufügen > Google* neu einrichten.
- Letzter Ausweg: Zurücksetzen auf Werkseinstellungen

Lösungen für spezielle Playstore-Probleme finden sich auch in den folgenden Punkten.

342. <http://www.google.com/appsstatus#hl=de>

Unbekannter Fehler: -18

Dieser Fehler tritt nur im Zusammenhang mit [App2SD](#) auf, wenn eine zu aktualisierende App auf der SD-Karte installiert ist (oder eine neue dort installiert werden soll). Die einfachste Lösung ist hier: App wieder in den internen Speicher zurück verschieben, das Update erneut versuchen (sollte jetzt funktionieren), und anschließend optional wieder auf die SD-Karte damit.

Schlägt das Update noch immer fehl, oder handelt es sich um eine neu zu installierende App, kann auch folgendes helfen: Androiden per USB-Kabel an den PC anschließen, und die Karte im Modus „USB Massenspeicher“ (nicht MTP/PTP) als Laufwerk freigeben. Damit steht sie dem Android-System nicht mehr zur Verfügung, was eine Installation in den internen Speicher erzwingt. Ist dies nicht möglich, weil das Gerät den [UMS](#) Modus nicht unterstützt, ggf. die SD-Karte stattdessen temporär entfernen.

Eine letzte Möglichkeit ist mit Vorsicht zu genießen: Mit einem Datei-Manager auf der SD-Karte den Ordner `.android_secure` (mit Punkt vorne!) suchen, und die darin befindliche Datei namens `smd12tmp1.asec` löschen, dann die Installation bzw. das Update noch einmal versuchen. Da dieser Ordner von Android selbst (ohne root-Zugriff) „ausgeblendet“ wird, ist hierfür ggf. ein Kartenleser am PC notwendig.

Unbekannter Fehler: -24

Dieser ist i. d. R. auf ein missglücktes Update zurückzuführen, dem „Altlasten“ im Weg lagen. Zur Behebung hilft es meist, die installierte Version der betroffenen App zu entfernen, und sodann die aktuelle Version erneut zu installieren. Tritt das Problem mit mehreren Apps auf, und die beschriebene Vorgehensweise schafft keine Abhilfe, nennen etliche Quellen einen [Factory-Reset](#) als letzten Ausweg.

Unbekannter Fehler: -25

Offensichtlich sieht der Playstore kein „Downgrade“ von Apps vor – denn dieser Fehler heißt nichts anderes als: „Sorry, aber da ist bereits eine neuere Version auf dem Gerät installiert.“ So etwas sollte normalerweise nicht vorkommen (außer man hat die App aus „anderer Quelle“ installiert). Abhilfe schafft natürlich eine De-Installation der fraglichen App – anschließend sollte die erneute Installation erfolgreich sein.



StackOverflow:
Installation error
code: -25



AndroidPIT: Error
RPC:S-5:AEC-0
verhindert App-
Installation

Hat man die `.apk` Datei der zu installierenden App-Version vorliegen, hilft auch eine [Installation mit ADB](#) mittels `adb install -r`.

(Quelle: [StackOverflow](#)³⁴³)

Error [RPC:S-5:AEC-0]

Zu diesem Fehler kommt es gern einmal, wenn der *Playstore* ein größeres Update erhalten hat. Wie man ihn behebt, beschreibt ein [Artikel bei AndroidPIT](#)³⁴⁴ im Detail: Man muss das Google-Konto entfernen und sich neu anmelden. Folgende Schritte sind dazu zu unternehmen:

1. *Einstellungen > Anwendungen verwalten* öffnen, und auf den Reiter „Alle“ wechseln
2. Jeweils die Apps *Google Play Store* sowie *Google Play Dienste* über den Button *Stoppen erzwingen* beenden, *Cache leeren*, und *Daten löschen*
3. Zu *Einstellungen > Konten* wechseln, das verwendete Google-Konto auswählen, und unten rechts den Button *Konto entfernen* betätigen
4. Das Gerät ausschalten, und neu starten
5. Zur Sicherheit ein paar Minuten warten, bis alle Dienste wieder laufen
6. Wieder zu *Einstellungen > Konten* wechseln, und das entfernte Konto neu hinzufügen

Sobald alle Daten wieder synchronisiert sind, sollte auch der Fehler behoben sein.

Achtung: Bei Geräten mit Android 2.3 und älter werden durch die Konten-Entfernung auch alle mit selbigem verknüpften Daten (also Kontakte, Kalender, etc.) gelöscht. Zur Sicherheit sollte daher zuvor ein Backup angelegt werden.

Fehler 502 beim Download

Dieser Fehler macht sich, wenn es denn zu ihm kommt, im Regelfall nur temporär bemerkbar. Die einfachste Lösung ist daher häufig, ein paar Stunden oder Tage abzuwarten. Ist dies aus irgend einem Grund nicht möglich, findet sich Abhilfe wiederum in einem [Artikel bei AndroidPIT](#)³⁴⁵:

Die (nach bereits genanntem „Abwarten“) nächst einfache Möglichkeit ist das Löschen des Caches der *Google Play Store* App. Dazu geht man zuerst zu *Einstellungen > Anwendungen verwalten*, wechselt auf den Reiter „Alle“, und scrollt sodann zu selbiger App. Nun den Eintrag öffnen, und den Button *Cache leeren* betätigen. Reicht das zur Behebung nicht aus, dann das Ganze noch

343. <http://stackoverflow.com/q/17710967/2533433>

344. <http://www.androidpit.de/google-play-store-error-rpc-s-5-aec-0-verhindert-app-installation>



AndroidPIT: Das hilft
gegen Fehler 502
beim Download

einmal – diesmal aber zusätzlich den Button *Daten löschen* auslösen. Dabei gehen allerdings auch die Einstellungen der App mit verloren.

Ist das Problem noch immer nicht gelöst, kann man wie bei [Error \[RPC:S-5:AEC-0\]](#) beschrieben vorgehen, und das Google-Konto komplett entfernen sowie wieder neu hinzufügen.

Für die Installation einer neuen App soll es gelegentlich auch helfen, diese einfach über die Webseite des *Playstore* anzustoßen: Anmelden, die App aufzusuchen, und den Button *Installieren* betätigen. Wer mehrere Geräte mit dem gleichen Konto verknüpft hat, muss nun noch das Zielgerät auswählen – und wenig später sollte die App auf selbigem erscheinen.

Fehler XXX



Es gibt noch eine ganze Reihe von Fehlern, die mehr oder weniger häufig auftreten können. Ist der gesuchte Fehler hier nicht aufgeführt, sei noch auf eine [Liste bei den XDA-Developers](#)³⁴⁶ verwiesen, in der weitere Fehlermeldungen aufgeführt und erläutert sind. Auch das [google-play-store tag-wiki bei Android.StackExchange.com](#)³⁴⁷ ist diesbezüglich eine gute Anlaufstelle.

XDA: Google Play Errors explained



Wie kann ich de-installierte Apps aus der Übersicht *Andere Apps in meiner Bibliothek* entfernen?

Android.SE: google-play-store tag-wiki

Google merkt sich alles. Das gilt insbesondere auch für die Apps: Alles, was man je installiert hatte, findet sich auf der Playstore-Seite unter *Meine Android Apps* wieder. Selbst wenn man es schon lange wieder von allen seinen Geräten entfernt hat – unter *Andere Apps in meiner Bibliothek* bleibt es stehen. Zumindest auf der Webseite ist auch nicht ersichtlich, wie man da einmal aufräumen kann.

Die Apps sind in diesem Segment als „unendliche zweispaltige Tabelle“ organisiert, für deren vollständige Anzeige man mehrfach den „mehr anzeigen“ Knopf betätigen muss. Eine Logik in der Reihenfolge war mir nicht ersichtlich; ebenso wenig lässt sich die Liste sortieren. Wer nun häufiger neue Apps testet, hat u. U. ein Problem eine (gerade deinstallierte) App wiederzufinden, wenn er sich nicht mehr an den Namen erinnert.

Hilfe findet sich hierzu in der Playstore App. Zumindest ab Version 3.10.* lässt sich mit selbiger an dieser Stelle Ordnung schaffen. Dazu sucht man zunächst

345. <http://www.androidpit.de/google-play-store-fehler-502-download-apps-hilfe>

346. <http://forum.xda-developers.com/showthread.php?t=2733038>

347. <http://android.stackexchange.com/tags/google-play-store/info>

die Seite mit den eigenen Apps auf (entweder über *Menü > Meine Apps*, oder über das Download-Symbol in der Titelseite). Hier wählt man den Tab „Alle“. Neben nicht-installierten Apps taucht in der Liste nun ein Symbol mit einem durchgestrichenen Kreis („Parkverbot“) auf. Tippt man dieses an, erfolgt eine Abfrage: „<AppName> aus meine Apps entfernen?“ Einfach auf „Ja“ tippen, und sie ist verschwunden – auch von der zugehörigen Playstore-Webseite.

Dummerweise springt die Liste anschließend automatisch wieder ganz an den Anfang zurück. Wer also richtig aufräumen (und mehrere Apps von der Liste entfernen) möchte, greift zu einem kleinen Trick: Die erste zu entfernende App einfach etwas länger drücken. Nun ist sie markiert. Am oberen Seitenrand sollte ein Balken auftauchen, der Links mit „1 App markiert“, und rechts mit „Entfernen“ beschriftet ist. Jetzt einfach die übrigen zu entfernenden Apps markieren, und dann alles in einem Rutsch aufräumen lassen – fertig.

Lässt sich irgendwo festlegen, dass nur bestimmte Apps automatisch aktualisiert werden sollen?



In der aktuellen Playstore-Version scheint sich das Auto-Update lediglich global aktivieren bzw. deaktivieren zu lassen (auch wenn verschiedene Apps wie z. B. die *Google Play Services* ihre eigenen Regeln zu haben scheinen). Was aber, wenn man nur ausgewählte Apps automatisch aktualisieren lassen möchte?

Eine Möglichkeit dazu findet sich bei [Stack Exchange](#)³⁴⁸ beschrieben. Zum Zuge kommt dabei wiederum eine App, die in diesem Buch bereits des Öfteren genannt wurde: [Titanium Backup](#)³⁴⁹. Betätigt man in dieser aus dem Hauptbildschirm die Menü-Taste, ist unter *Play Store / Market* der Punkt *Automatische Market-Updates* zu finden. An dieser Stelle lässt sich das Verhalten pro App festlegen. Die getätigten Einstellungen greifen natürlich nur, wenn in der *Google Play Store* App die automatischen Updates generell aktiviert wurden. Und leider scheinen die Google Services dies aktuell wieder zu überschreiben, sodass dieser „Fix“ nur kurzzeitig wirkt.



Kann ich auf einem Gerät mehrere Google-Accounts für den Playstore verwenden?

Sinnvoll wäre dies für verschiedene Szenarien: Geschäftliches von Privatem trennen, oder der Wechsel zu einer neuen GMail-Adresse ohne Verlust der

348. <http://android.stackexchange.com/q/50681/16575>

349. <https://play.google.com/store/apps/details?id=com.keramidas.TitaniumBackup>

gekauften Apps wären nur zwei davon. Und glücklicherweise ist dies mittlerweile auch recht einfach möglich.

Einen neuen Account kann man beispielsweise unter *Einstellungen > Konten & Synchronisation* erstellen, auch wenn dort bereits ein Google-Account besteht. Eine weitere Möglichkeit ist, dies gleich direkt in der Playstore-App zu erledigen: Unter *Menü > Konten* findet sich dafür der Punkt „Konto hinzufügen“. Beides führt zum Einrichtungs-Wizard, in dem sich entweder ein bereits bestehendes Konto eintragen, oder ein neues anlegen lässt.

An besagter Stelle in der Playstore-App lässt sich dann auch das für die jeweilige Einkaufstour zu verwendende Konto auswählen.

Wie kann ich im Playstore das Land wechseln?

Nein, es geht nicht um das Aushebeln regionaler Beschränkungen, sondern vielmehr um einen Umzug: Da gibt es Apps, die beispielsweise in Österreich verfügbar sind, in Deutschland jedoch nicht. Hat nun jemand seinen Wohnort von Deutschland nach Österreich verlegt, ist der Wunsch der entsprechenden Umstellung im Google Playstore also völlig legitim. Nur wie wird er bewerkstelligt?

Zuerst müssen die in Google Wallet hinterlegten Adressdaten aktualisiert werden. Dazu gehört auch die mit dem Google-Konto verknüpfte Kreditkarte, die in unserem Beispiel nun aus Österreich stammen sollte.

Soweit scheint alles nachvollziehbar – dennoch werden besagte österreichische Apps ggf. noch immer als „in Ihrem Land nicht verfügbar“ angezeigt: Google Play hat vom Adresswechsel offensichtlich noch nichts mitbekommen. Was hier hilft, ist der Kauf einer beliebigen App (wenn man sie nicht braucht, kann man sie ja binnen 2 Stunden wieder retournieren). Durch den Kauf muss Google Play seine Daten mit Google Wallet abgleichen, und weiß nun Bescheid: Jetzt sind besagte österreichische Apps verfügbar, der „Länderwechsel“ ist erfolgreich vollzogen.

Wie kann ich im Playstore bezahlen?

Es gibt zahlreiche Apps im Playstore, die man einfach gratis herunterladen kann. Doch oftmals bietet sich auch ein Kauf an: Sei es, dass man den Entwickler unterstützen möchte, dass die Werbung nervt, oder die Kaufversion einer App einfach mehr Features bietet. Doch welche Möglichkeiten zum Bezahlen gibt es?



Google Checkout



WireCard

Lange Zeit hieß die einzige hierzulande Verfügbare Methode „Kreditkarte“. Die Informationen zu selbiger müssen bei [Google Checkout](#)³⁵⁰ (früher *Google Wallet* genannt) hinterlegt werden; die Aufforderung dazu erfolgt automatisch beim ersten Einkauf. Es muss dabei nicht unbedingt eine „herkömmliche“ Kreditkarte zum Einsatz kommen, auch so genannte „Prepaid Kreditkarten“ wie beispielsweise [WireCard](#)³⁵¹ lassen sich verwenden. Und limitieren gleichzeitig das Risiko – denn wie viel Geld ausgegeben werden kann, lässt sich im Vorfeld beim Aufladen festlegen. Damit eignet sich dieses Zahlungsmittel auch insbesondere für den Nachwuchs, den man an die Eigen-Verantwortung heranführen möchte.

Alternativ gibt es bei einigen Mobilfunk-Anbietern die Möglichkeit, das Ganze über die Telefonrechnung abzuwickeln. Allerdings ist dies nicht bei jedem Anbieter möglich.

Seit Juni 2013 endlich gibt es sie auch für *Google Play*. Die Guthaben-Karten, die „andere“ schon lange für *iTunes* & Co. kennen. Erwerben lassen sie sich in zahlreichen Super- und Elektronik-Märkten; so bei Rewe, Saturn, MediaMarkt, und etlichen weiteren, mit einem Wert von 15, 25 oder 50 Euro bestückt. Der Guthabens-Code wird auf der Rückseite freigerubbelt, und in der *Playstore-App* unter „Einlösen“ eingetragen. MaTT von Android-TV [zeigt in einem Video](#)³⁵² detailliert, wie das funktioniert.

Von vielen sehnstüchtig erwartet: Mit Start im Mai 2014 ist schließlich auch *Paypal* ein valides Zahlungsmittel bei *Google Play* – womit auch den letzten Kaufverweigerern die Ausrede „kann nicht“ genommen wurde.

Wie kann ich den Playstore vor der Kaufwut meiner Sprößlinge schützen?

(Sorry Kids – aber die reißerische Überschrift soll die Kaufwut der Erwachsenen für mein Buch anstacheln. Ihr könnt das Folgende aber ebenso gut einsetzen, damit sich die Mama nicht kreischend per Zalando durch die Gegend bewegt.)

Gibt man den eigenen Androiden zeitweilig aus der Hand, soll der Beglückte natürlich nicht grenzenlos kostenpflichtige Apps herunterladen. Doch dagegen kann man sich mit wenigen Schritten absichern:

1. Zunächst die *Playstore* App öffnen.
2. Dort das Menü öffnen, und den Punkt *Einstellungen* auswählen.

350. <https://wallet.google.com/>

351. <http://www.wirecard.de/>

352. <http://www.go2android.de/video/video-android-für-anfänger-wie-funktioniert-die-google-play-card-folge-5/>

3. Unter „Nutzersteuerung“ findet sich der Punkt „Authentifizierung für Käufe erforderlich“. Diesen auswählen.
4. In der folgenden Liste auswählen, wie der Schutz aussehen soll. Zur Auswahl stehen i. d. R. „für alle Käufe“ (also vor jedem einzelnen Einkauf erneut), „alle 30 Minuten“ (nach einmaliger Passwort-Eingabe ist für 30 Minuten keine weitere Verifikation des Passwortes notwendig), und „Nie“ (also gar keine Passwort-Abfrage).
5. Es erfolgt nun eine Passwort-Abfrage. Hier ist das Passwort des Google-Kontos einzugeben.

Ist das erledigt, muss (bei entsprechender Auswahl) vor jedem Einkauf – sei es eine kostenpflichtige App, oder auch ein Einkauf innerhalb einer App – das Google-Passwort eingegeben werden.

Apps

Was passiert mit einer App, wenn sie aus der Liste zuletzt genutzter Anwendungen gewischt wird?

Vor Android 4.0 erreichte man die Liste zuletzt genutzter Apps durch langes Drücken auf die „Home“ Taste, und konnte so zu einer dieser Apps zurückkehren – mehr ging da nicht. Mit Android 4 wurde die „Multi-Tasking-Taste“ eingeführt, und zeigt nun Screenshots der zuletzt genutzten Apps. Beiden Varianten ist gemein, dass an dieser Stelle aufgeführte Apps nicht zwangsläufig noch im Hintergrund laufen (selbst über einen Task-Killer beendete Apps sind hier noch aufgeführt). Doch durch eine Wisch-Bewegungen lassen sie sich nun aus der Liste entfernen. Was aber passiert dabei eigentlich?

Das offensichtliche zuerst: Die betroffene App wird aus der Liste zuletzt genutzter Apps entfernt, taucht also bis zu ihrem nächsten Start hier nicht mehr auf. Doch das ist natürlich nicht alles: Sie wird gleichzeitig auch „beendet“. Und zwar in etwa so, als hätte man in der App selbst so oft die „Zurück“-Taste gedrückt, bis sie sich schließt. Mit anderen Worten: Die App wird höflich gebeten, sich doch zu beenden – sie wird jedoch nicht „gekillt“. Soll sie vollständig beendet werden, drückt man stattdessen länger auf den betreffenden Eintrag, wodurch man zu den App-Details gelangt. Hier kann man nun den „Beenden“ Button betätigen.

Wer sich für genauere Einzelheiten interessiert, findet diese u. a. in einem [Artikel bei Stack Exchange](#)³⁵³.

Verbrauchen Homescreen-Widgets Ressourcen, auch wenn der entsprechende Bildschirm nicht angezeigt wird?

Sicher werden von einem Widget auch Ressourcen belegt, wenn es gerade nicht sichtbar ist: Der verwendete Launcher muss sich ja merken, welchen Platz es belegt. Auch das generelle Layout bleibt im RAM hinterlegt, damit der Anwender nicht bei jedem Bildschirm-Wechsel erst lange auf dessen Aufbau warten muss.

353. <http://android.stackexchange.com/q/19987/16575>



ASE: What actually happens when you swipe an app out of the recent apps list?

Der Ressourcen-Verbrauch der Widgets selbst kann sicher vernachlässigt werden, so groß sind sie schließlich selten. Doch muss man sich bewusst sein, dass i. d. R. hinter jedem Widget ein Dienst steht, der dieses aktualisiert. Auch dieser benötigt Ressourcen, um die gewünschten Informationen aufzubereiten und bereitzustellen. Wie stark das ins Gewicht fällt, hängt von der Art des Widgets ab: Zeigt es nur Informationen zu einem ohnehin laufenden Dienst (etwa neue Mails) an, ist der „zusätzliche Aufwand“ eher gering. Geht es hingegen darum, welcher Song gerade auf dem Lieblings-Sender läuft, wird es definitiv mehr – diese Informationen müssen zusätzlich regelmäßig aus dem Netz abgerufen werden.

Das waren jetzt nur kurze Beispiele. Wer sich für detailliertere Hintergrund-Informationen interessiert, dem sei [Dan's Artikel bei Stack Exchange](#)³⁵⁴ empfohlen.



Stack Exchange: Do widgets run if not on current home screen?



Stack Exchange: What kind of app optimization does Ice Cream Sandwich do at the first reboot?



Beim ersten Gerät-Neustart nach einer Aktualisierung auf Android 4.x erscheint auf dem Display der Hinweis: „Android wird aktualisiert“, darunter ein Zähler: „App x von y wird optimiert“. Was aber hat es damit auf sich?

Um eine aussagekräftige Antwort zu finden, hat sich [eldarerathis bei Stack Exchange](#)³⁵⁵ den Quellcode des Package-Managers vorgenommen, und analysiert. Das Ergebnis: Der [Dalvik-Cache](#) wird aktualisiert. Dieser Schritt ist in mehreren Fällen notwendig: Bei einem neuen Gerät bzw. nach einem [Factory-Reset](#) muss der Dalvik-Cache initial erstellt werden – und nach einem System-Update, bei dem auch die Dalvik-Engine Aktualisierungen erfahren hat, muss er entsprechend angepasst werden.

Wie kann ich an ältere Versionen installierter Apps kommen?

Da ist es wieder einmal passiert: Ein Update macht eine App instabil, oder entfernt lieb gewonnene Funktionalität. Es gibt einige Gründe, warum man gelegentlich lieber auf eine ältere Version zurückgreift. Nur woher nehmen?

354. <http://android.stackexchange.com/a/45443/16575>

355. <http://android.stackexchange.com/q/21653/16575>



AppMonster



Android Drawer



F-Droid



Freeware-Lovers

Im Thema [Alternative Verwaltung](#) habe ich ja bereits erwähnt, wie man für diesen Fall Vorsorge treffen kann: Indem man beispielsweise mit [AppMonster](#)³⁵⁶ bei jeder Installation und jedem Update ein Backup erstellen lässt. Doch meist wird man erst „aus Schaden klug“, und hat daher bei erstmaligem Auftreten dieses Problems noch kein derartiges Backup zur Hand.

Im Zusammenhang mit [Playstore-Alternativen](#) wurde für diesen Fall eine passende Quelle benannt: [Android Drawer](#)³⁵⁷. Dort findet man für die meisten gratis-Apps .apk Dateien zum Download, auch für mehrere Versionen. Ähnlich sieht es beim ebenfalls dort genannten [F-Droid](#)³⁵⁸ aus. Ein weiterer Anlaufpunkt findet sich mit [Freeware-Lovers](#)³⁵⁹.

Nicht fündig geworden? Das dürfte insbesondere bei Kauf-Apps zutreffen. In jedem Fall lohnt sich dann eine Anfrage beim Entwickler. Gerade, wenn ein Bug Anlass für den Downgrade-Wunsch ist, eine gute Idee: Mit etwas Glück erhält man stattdessen sogar eine neuere, noch nicht veröffentlichte Version der App, in der das Problem schon behoben wurde.

Ich habe eine APK-Datei auf meinen PC heruntergeladen. Wie installiere ich sie jetzt?

Eine besonders nach dem vorigen Punkt berechtigte Frage, auf die es mehrere Antworten gibt:

- Bei auf dem PC installierter [Android Debug Bridge](#) einfach mittels `adb install </Pfad/dateiname.apk>` (siehe [Apps mit ADB installieren](#))
- Analog dazu: Nach Kopieren der Datei auf den Androiden, mit einer Terminal-Emulator App an der Kommandozeile `pm install <dateiname.apk>` ausführen.
- APK-Datei auf die SD-Karte des Androiden kopieren, dort mit einem [Dateimanager](#) zu selbiger navigieren, und „ausführen“. Sofern der Dateimanager auch auf das lokale Netzwerk zugreifen kann, spart man sich u. U. das Kopieren, und navigiert einfach direkt zur APK-Datei auf dem PC.
- APK-Datei an die eigene GMail-Adresse schicken, und in der GMail-App dann direkt aus dem Anhang „starten“.
- APK-Datei in den eigenen Dropbox-Account hochladen, und auf dem Androiden mit der Dropbox-App öffnen.

356. https://play.google.com/store/apps/details?id=de.android_telefonie.appmanager

357. <http://www.androiddrawer.com/about/>

358. <http://f-droid.org/>

359. <http://www.freewarelovers.com/android/apps>

- Eine App zur Verwaltung des Androiden vom PC nutzen. Die meisten Kandidaten unterstützen auch die Installation von Apps.

In allen Fällen (mit Ausnahme der Installation via ADB) gilt, dass in *Einstellungen > Apps* die Option „fremde Quellen zulassen“ aktiviert sein muss.

aLogCat zeigt unter Jelly Bean keine Einträge anderer Apps mehr an!

Das ist korrekt: Ab Android 4.1 hat Google das Sicherheits-Modell für den Zugriff auf die Systemlogs geändert. Davon betroffen sind neben *aLogCat* natürlich auch alle anderen ähnlichen Apps – sofern nicht „Herr Root“ zum Einsatz kommt.

Auch dafür gibt es verschiedene Lösungs-Möglichkeiten, von denen ein [Artikel bei Stack Exchange](#)³⁶⁰ einige aufzählt:

- Bei „angeschlossenem PC“ einfach alternativ `adb logcat` nutzen
- Mit root: Der jeweiligen Log-App die Permission `READ_LOGS` geben. Für *aLogCat* wäre der zugehörige Befehl: `pm grant org.jtb.alogcat android.permission.READ_LOGS`. Zusätzlich muss die App dann (beispielsweise mittels *Titanium Backup*) in eine System-App umgewandelt werden – denn nur diesen wird die `READ_LOGS` Berechtigung auch gewährt.
- Aus dem vorigen Punkt ließe sich ableiten: Eine Log-Lese App suchen, welche bereits über diese Berechtigung verfügt. Das reicht leider nicht: Die App scheint dennoch root zu benötigen, sofern es sich nicht um eine System-App handelt.



Stack Exchange:
Problems accessing
message logs on
Jelly Bean with
aLogcat

Ich erhalte ständig die Meldung „App xyz wurde unerwartet beendet“.

Kommt dies gelegentlich vor, ist es einfach nur ärgerlich – und man sollte sich an den Entwickler der App wenden. Tragisch wird es jedoch, wenn eine App das bereits beim Starten meldet, und zwar immer, sich also gar nicht mehr starten lässt. In den meisten Fällen sorgt hier folgendes für Abhilfe:

In den System-Einstellungen unter *Apps* die entsprechende App heraussuchen, und auf den Button *Cache leeren* drücken. Hilft dies auch nicht, an gleicher Stelle *Daten löschen* (ggf. zuvor ein Backup machen). Ab und an löst auch eine einfache Neuinstallation der App das Problem.

360. <http://android.stackexchange.com/q/27586/16575>

Handelt es sich bei der fraglichen App um den *Google Playstore* oder eine andere Google-App, könnten auch die Tipps unter [Hilfe, der Play Store spinnt](#) nützlich sein.

Google Play Dienste verbrauchen plötzlich enorm viel Akku!

Gerade gewundert, warum der Akku so schnell leer war – und unter *Einstellungen > Akkuverbrauch* festgestellt, dass *Google Play-Dienste* mit Abstand der größte Verbraucher war? Das scheint des öfteren vorzukommen. Doch selbst wenn die Ursachen nicht ganz klar sind, gibt es ein paar Möglichkeiten, dies in den Griff zu bekommen – bis ein Update das Problem hoffentlich löst:

- Häufig scheinen die Standort-Dienste dahinter zu stecken. Dann hilft es, unter *Einstellungen > Standordienste* die Option „Zugriff auf meinen Standort“ zu deaktivieren.
- Brachte das nicht den gewünschten Erfolg, hilft ein „Reset“ der App: Unter *Einstellungen > Apps* die *Google Play-Dienste* heraussuchen, deren Eintrag öffnen. Dann die Buttons „Cache leeren“ und „Daten löschen“ betätigen. Schließlich die App mit „Stoppen erzwingen“ zwangsbeenden, und den Androiden neu starten.
- Besteht das Problem weiterhin? Vorigen Schritt wiederholen. Diesmal jedoch zusätzlich vor dem Neustart noch den Button „Updates entfernen“ betätigen. Keine Angst, Updates werden automatisch wieder eingespielt – allerdings nicht unbedingt sofort.
- Da, wie eingangs erwähnt, häufig die Standort-Dienste hinter dem Problem stecken, kann u. U. auch *Google Maps* involviert sein. Dieser App ist dann ebenfalls die beschriebene Behandlung zu verpassen. Wer *Google Maps* ohnehin nicht benutzt, kann es nach der De-Installation der Updates auch „Deaktivieren“: Was nicht aktiv ist, sollte auch keine Probleme machen.

Kann ich eine App davon abhalten, ständig im Hintergrund zu laufen?

Insbesondere *Google Maps* ist dafür bekannt, des öfteren permanent im Hintergrund zu laufen – obwohl der Anwender die App gar nicht gestartet hat. Aber auch andere Apps laufen gern weiter, wenn man dies gar nicht wünscht. Zum Glück gibt es da Abhilfe:

- Wird die App überhaupt nicht benötigt: Deinstallieren – oder im Falle einer System-App deaktivieren. Beides lässt sich im Menü *Einstellungen > Apps* bewerkstelligen. System-Apps findet man hier im Reiter „Alle“.

Den Eintrag der unerwünschten App durch Antippen öffnen, und dann den entsprechenden Button betätigen.

- Benötigt man die App gelegentlich oder auch regelmäßig, hilft [Greenify](#)³⁶¹ weiter: Mit dieser App lassen sich unsere Kandidaten „temporär deaktivieren“, wann immer sie nicht im Vordergrund laufen. Welche Apps *Greenify* auf diese Weise behandeln soll, legt der Anwender selbst fest. Das Ganze passiert dann völlig transparent: Alles, was man merkt, ist dass die App nicht mehr im Hintergrund läuft – und somit auch keine Ressourcen verbraucht, wenn man sie nicht gerade aktiv nutzt. In manchen Fällen ist dabei [root](#) hilfreich, wird jedoch nicht zwingend vorausgesetzt. Genauer beschrieben ist dies im Abschnitt [Helferlein](#).
- Mit root lassen sich bei Bedarf auch gewisse „automatische Starts“ von Apps deaktivieren – beispielsweise mit Hilfe der App [Autorun Manager Pro](#)³⁶². Auf diese Weise kann man beispielsweise unterbinden, dass besagtes *Google Maps* automatisch „anspringt“, wenn sich etwa der Netzwerk-Status ändert. Näheres findet sich im Kapitel [Apps am automatischen Starten hindern](#).



Greenify



Autorun Manager Pro

361. <https://play.google.com/store/apps/details?id=com.oasisfeng.greenify>

362. <https://play.google.com/store/apps/details?id=com.rs.autorun.pro>

Backup

Welche Arten von Backups gibt es eigentlich, und was wird da jeweils gesichert?



Auf dieses Thema geht [ein Artikel bei den XDA-Developers](#)³⁶³ ein, der auch Links zu weiterführenden Informationen beinhaltet. Das Wesentliche sei hier kurz zusammengefasst:

Backup-Typ	Was wohin gesichert wird
Google Sync (aka „Google Cloud Backup“)	Kontakte, Kalender, Docs / Drive, Gmail, Google Photos, Google Reader, „Internet & Instant“ (Lesezeichen u. a.), WLAN Passwörter, und mehr werden in der Google Cloud abgelegt. Es soll auch die Daten der installierten Apps in der Cloud speichern und zwischen Geräten synchronisieren – in der Praxis hat es sich aber als extrem unzuverlässig erwiesen.
Dropbox & Co.	Vom Anwender speziell vorgegebene Dateien und Verzeichnisse. Der Umfang variiert je nach Cloud-Service. Etliche Apps verfügen zudem über Dropbox-Support. So können z. B. auch mit <i>Titanium Backup</i> erstellte Datensicherungen hier abgelegt werden.
Titanium Backup	Apps und ihre Daten, sowie Systemeinstellungen und mehr werden auf die SD-Karte gesichert. Ebenfalls möglich ist die Speicherung in der Cloud – so wird z. B. Dropbox direkt unterstützt. Benötigt <u>root</u> -Rechte.
<u>ADB Backup</u>	Apps und ihre Daten werden auf das System gesichert, welches den Backup-Prozess ausgelöst hat (i. d. R. ist dies der PC, an den das Android-Gerät per USB-Kabel angeschlossen wurde – es gibt aber auch Implementierungen dieses Backup-Typs direkt in einer App, z. B. <i>Helium</i> , ehemals als <i>Carbon Backup</i> bekannt). Erst ab Android 4.0 verfügbar.
Samsung Kies (und andere)	Nur für die Geräte des jeweiligen Herstellers verfügbar, und sowohl im Umfang als auch in Sachen Zuverlässigkeit durchaus verschieden.

363. <http://forum.xda-developers.com/showthread.php?t=1678239>

Backup-Typ	Was wohin gesichert wird
herstellerspezifischen PC-Suites)	
<u>Nandroid</u> Backup	Erstellt 1:1 Sicherungen der meisten (und aus Anwendersicht wichtigsten) Partitionen des Android-Gerätes in so genannten „Image Dateien“ bzw. <u>Tar</u> -Archiven. Auf diese lässt sich u. a. auch mit <i>Titanium Backup</i> zugreifen. Benötigt <u>root</u> und ein Custom <u>Recovery</u> .
Diverse datenspezifische Backups	Je nach Backup-App werden hier SMS, MMS, Anruflisten, Lesezeichen, Kontakte, Termine, oder andere Daten (ggf. auch Kombinationen der genannten Auswahl) auf die SD-Karte oder in die Cloud gesichert.

Wie kann ich vom Android-Gerät auf ein Nandroid-Backup zugreifen?

Am einfachsten kann dies geschehen, wenn man dafür ein kleines Helferlein aus dem Google Play Store verwendet. Dafür bietet sich beispielsweise der Nandroid Manager³⁶⁴ an, mit dem sich zusätzlich auch einzelne Komponenten wie WLAN APNs, Kurznachrichten, Anrufprotokolle, oder Apps einschließlich ihrer Daten wieder herstellen lassen – was übrigens auch von *Titanium Backup* unterstützt wird.



Nandroid Manager

Kann ich Backups auf einem anderen Gerät wieder herstellen?

Das hängt u. a. von der Art des Backups ab. Mit einem Nandroid Backup sollte man dies besser nicht versuchen (es sei denn, es handelt sich um identische Geräte). Bei „datenspezifischen Backups“ (wie sie von verschiedenen Apps im Playstore angeboten werden) dürfte es hingegen keinerlei Probleme geben. Geht es um das „Google Cloud Backup“, lässt sich ohnehin keine Auswahl treffen (es klappt, oder auch nicht, oder nur teilweise – der Anwender hat darauf keinen Einfluss).

Mit von *Titanium Backup* erstellten Sicherungen sollte es keinerlei Probleme geben, sofern man nur Benutzer-Anwendungen (sowie ihre Daten) wieder herstellt. Vorsicht ist geboten, sobald System-Anwendungen und -daten ins Spiel kommen: Zwar bietet *Titanium Backup* hierfür einen speziellen Migrations-Modus – Garantien gibt es jedoch keine.

364. <https://play.google.com/store/apps/details?id=com.h3r3t1c.bkrestore>

Ähnlich steht es um „ADB Backups“. Bei deren Wiederherstellung lässt sich bekanntermaßen keine Auswahl treffen: Es wird immer die gesamte Backup-Datei wiederhergestellt. Enthielt diese nur (eine) einzelne User-App(s), sind eigentlich keine Probleme zu erwarten – schließlich synchronisiert auch *Helium Apps* und Daten auf diese Weise geräteübergreifend. Von der Wiederherstellung einer Komplettsicherung auf einem anderen Gerät sollte man jedoch, wie bereits beim Nandroid Backup, Abstand nehmen.

Kann ich Backups automatisieren, und außerhalb meines Gerätes speichern?



Helium Backup



Titanium Backup



FolderSync

Auch das ist möglich – wobei die Auswahl der einzusetzenden Apps nicht zuletzt davon abhängt, ob das betroffene Gerät gerootet ist oder nicht. Sowohl [Helium Backup](#)³⁶⁵ (funktioniert ohne root, mit leichten Einschränkungen) als auch [Titanium Backup](#)³⁶⁶ (benötigt root) unterstützen einen „Scheduler“, über den sich Backups automatisch zu festgelegten Zeiten erstellen lassen. Beide unterstützen auch von Haus aus die Speicherung bei verschiedenen Cloud-Diensten (Dropbox, Box, Google Drive).

Wem diese Cloud-Dienste „nicht ganz geheuer“ sind, der möchte die Daten jedoch lieber auf dem eigenen Rechner (oder NAS) ablegen. Dafür greift man einfach zu einem weiteren Tool: [FolderSync](#)³⁶⁷. Neben einer Reihe weiterer Cloud-Dienste unterstützt diese App auch FTP, FTPS, SFTP, und sogar Samba/CIFS, sodass als Speichermedium problemlos der eigene Rechner verwendet werden kann. Dazu konfiguriert man „Ordner-Paare“: Einem lokalen Verzeichnis wird ein Verzeichnis auf dem „entfernten System“ zugeordnet, mit welchem es synchron gehalten werden soll. Auf diese Weise lassen sich neben den regulären Backups auch gleich weitere Dinge „in Sicherheit bringen“ – etwa aufgenommene Fotos. Erweiterte Einstellungen für die Synchronisation (Zeitpunkte, bidirektionaler Sync oder nur auf den Server kopieren, nur WLAN verwenden) sind so ebenfalls möglich – und einiges mehr.

365. <https://play.google.com/store/apps/details?id=com.koushikdutta.backup>

366. <https://play.google.com/store/apps/details?id=com.keramidas.TitaniumBackup>

367. <https://play.google.com/store/apps/details?id=dk.tacit.android.foldersync.lite>

Medien

Wie kann ich Dateien in der Mediengalerie ausblenden?

Einzelne Verzeichnisse (einschließlich deren Unterverzeichnisse) lassen sich „ausblenden“, indem man in ihnen eine Datei mit dem Namen `.nomedia` (also mit Punkt vorne) anlegt. Dies lässt sich am einfachsten mit einem Dateimanager bewerkstelligen. Damit werden diese Verzeichnisse vom Medien-Scan ausgeschlossen – und ihre Inhalte somit in der Galerie nicht mehr angezeigt.

Wie kann ich eigene Töne als Klingelzeichen, Benachrichtigung, oder für den Wecker nutzen?

Dafür gibt es spezielle Verzeichnisse auf der SD-Karte:

Verzeichnis	Verwendung
alarms	Alarm-Töne
notifications	Benachrichtigungstöne
ringtones	Klingeltöne
ui	Tastatur-Klick-sounds etc.

Wo genau diese anzulegen sind, unterscheidet sich offensichtlich bei verschiedenen Android-Geräten. Der Wahrscheinlichkeit nach geordnet, sollte dies an folgenden Stellen geschehen:

- `/sdcard/media/audio`
- `/sdcard/media`
- `/sdcard`

In den in der Tabelle genannten Unterverzeichnissen werden die gewünschten Sound-Dateien dann platziert. Damit das System sie auch findet, muss der Medien-Scanner sie aber zunächst erfasst haben – also nicht wundern, wenn sie nicht sofort auftauchen!

Eine sichere Möglichkeit, einen Medien-Scan anzustoßen, besteht im Neustart des Androiden: Nach jedem Booten wird der Scanner nämlich automatisch ausgeführt. Wer nicht so lange warten will, lädt sich das kleine Tool [SDRescan](#)³⁶⁸



SDRescan

368. <https://play.google.com/store/apps/details?id=com.bero.sdrescan>



Stack Exchange: How do I refresh/rescan the SD memory in Android 4.4 KitKat?



SD Scanner

aus dem *Play Store*, und stößt den Scan damit manuell an (was leider [ab Android 4.4 nicht mehr funktioniert](#)³⁶⁹, da Apps das dort ohne root-Rechte nicht mehr dürfen. Abhilfe schafft hier [SD Scanner](#)³⁷⁰, der einen Kitkat-kompatiblen Ansatz gefunden hat). Anschließend sollten sich die neuen „Töne“ als Klingelzeichen usw. auswählen lassen.

Wie kann ich aufgenommene Fotos und Videos automatisch auf der SD-Karte speichern lassen?

Bei den meisten Geräten ist gar nichts anderes vorgesehen, als selbst gemachte Fotos und Videos auf der Speicherkarte abzulegen – der interne Speicher ist oftmals ohnehin schon zu knapp bemessen. Ist dies jedoch einmal nicht der Fall, lässt es sich i. d. R. in der Kamera-App selbst konfigurieren. Dazu öffnet man selbige, und dort das Einstellungsmenü (sofern kein entsprechendes Icon angezeigt wird, einfach einmal die Menü-Taste betätigen). Hier sollte nun ein Menüpunkt „Speicher“ zu sehen sein – unter dem man zwischen „Telefonspeicher“ (dem internen Speicher) und „Speicherkarte“ wählen kann. Manche Kamera-Apps erlauben sogar, explizit ein bestimmtes Verzeichnis vorzugeben.

369. <http://android.stackexchange.com/q/57913/16575>

370. <https://play.google.com/store/apps/details?id=com.gmail.jerickson314.sdscanner>

Umgang mit der SD-Karte

Wo finde ich die SD-Karte im lokalen Dateisystem?

Die meisten Geräte binden sie unter `/sdcard` ein. Leider jedoch ist dies kein Standard, sodass insbesondere bei Geräten mit zusätzlicher „interner SD-Karte“ Abweichungen gelten. So findet sich die „externe“ SD-Karte beispielsweise beim Motorola Xoom unter `/mnt/external1` – andere Geräte binden die Karte unter `/sdcard/external_sd` ein, wiederum andere legen sie auf `/mnt/sdcard` oder gar `/mnt/sdcard/external_sd` ... Ein „relativer Standard“ ist `/storage/sdcard0`, aber nicht jeder Hersteller hält sich daran. Ist die Karte an keiner der genannten Stellen auffindbar, hilft ggf. ein Blick ins Handbuch – oder eine Frage im Forum.

Ein kleiner Trick an der Kommandozeile macht sich zunutze, *wie* das System SD-Karten einbindet. So lässt sich mittels [adb shell](#) oder einer Terminal-App der Befehl `mount | grep vold | awk '{print $2}'` absetzen. Die Ausgabe enthält dann alle Verzeichnisse, in denen eine SD-Karte eingebunden ist.

Wie kann ich vom PC auf die SD-Karte zugreifen?

Eine Möglichkeit wäre natürlich, sie aus dem Gerät zu entnehmen und (ggf. mittels eines Adapters) über einen Kartenleser an den PC anzuschließen. Das ist natürlich ein wenig umständlich – insbesondere dann, wenn man (wie bei einigen Geräten der Fall) nur an sie heran kommt, nachdem man den Akku entfernt hat (oder es sich um ein Nexus-Gerät handelt, das nur über eine „fest verbaute“ interne SD-Karte verfügt). Daher gibt es eine einfachere Möglichkeit: Man verbindet das Gerät mittels eines USB-Kabels mit dem Computer. Oftmals wird sie dabei automatisch auf letzterem angezeigt. Ist dies nicht der Fall, zieht man auf dem Androiden kurz die Benachrichtigungsleiste auf: Bei angeschlossenem USB-Kabel sollte sich nun hier ein Punkt finden, über den man die Karte an den PC freigeben kann. Als Symbol trägt dieser meist ein USB-Icon.

Klappt dies nicht, finden sich weiterführende Details im Kapitel „[Shared Storage](#)“ – oder „[Die“ SD-Karte](#).

Wie kann ich einen Rescan der SD-Karte veranlassen?

Das ist bereits als Teil der Frage zu [eigenen Klingeltönen](#) behandelt worden – sei aber an dieser Stelle nochmals zusammengefasst. Verschiedene Möglichkeiten kommen dazu in Frage:

- Den Androiden neu starten. Der Media-Scanner wird dann automatisch nach vollendetem Boot vom System initialisiert. Klappt immer.
- In *Einstellungen > Speicherverwaltung* die „Speicherkarte entfernen“, und anschließend wieder einbinden. Auch dadurch löst das System den Medien-Scan automatisch aus. Klappt nur, sofern auch eine externe SD-Karte vorhanden ist.
- Das gleiche per [ADB Shell-Befehl](#) emulieren: `adb shell "am broadcast -a android.intent.action.MEDIA_MOUNTED -d file:///mnt/sdcard"` (gaukelt dem System vor, die SD-Karte wäre gerade neu eingebunden worden). Wer dies per SSH versuchen möchte, muss ggf. noch [entsprechende Umgebungs-Variablen setzen](#)³⁷¹.
- Mit einer App wie [SD Scanner](#)³⁷² den Media-Scanner per Widget aktivieren.



ASE: Can I trigger a media scan via the command line



SD Scanner

Wie kann ich meine SD-Karte wechseln?

Banale Antwort: Alte Karte raus, neue rein. Aber die Frage gilt sicher in erster Linie der Tatsache, dass man die darauf enthaltenen Daten auch gern auf die neue, größere Karte mitnehmen möchte. Auch das stellt aber kein Problem dar – solange auf der Karte nur eine einzige Partition ist (wer das jetzt von seiner Karte nicht weiß, bei dem ist dies mit ziemlicher Sicherheit der Fall).

Das Vorgehen ist in etwa folgendes:

1. Über *Einstellungen > SD-Karte und Telefonspeicher* zunächst die SD-Karte trennen (alternativ: Das Telefon abschalten)
2. Die „alte“ SD-Karte entnehmen, und mit einem Kartenleser am PC anschließen
3. Sämtliche Daten in ein leeres Verzeichnis auf der Festplatte kopieren
4. Die Karte wieder sauber vom PC trennen, und die „neue“ Karte mit dem Kartenleser anschließen
5. Die Daten wieder 1:1 auf die neue Karte kopieren
6. Karte sauber vom PC trennen, dem Kartenleser entnehmen, und in den Androiden einlegen.

371. <http://android.stackexchange.com/a/51800/16575>

372. <https://play.google.com/store/apps/details?id=com.gmail.jerickson314.sdscanner>

Wer ganz auf „Nummer sicher“ gehen möchte, macht das an einem Linux-PC (z. B. mit einer Live-CD wie Knoppix). Auf jeden Fall sollte man darauf achten, dass auch „versteckte Verzeichnisse“ (z. B. `.android_secure`, siehe nächste Frage, und `.adobe-digital-editions`, sofern vorhanden) mit kopiert werden.

Fertig. Testen ob alles klappt. Das sollte es in 99% aller Fälle – und falls nicht, hat man bis zu einer Problemlösung ja noch immer die „intakte alte“ Karte.

Ich kann App xyz nicht auf die SD-Karte verschieben!

Bei einem Factory-Reset (zurücksetzen auf Werkseinstellungen) gehen bekanntlich alle selbstinstallierten Apps (und Daten) verloren. Also müssen sie neu installiert werden. War eine App zuvor auf der SD-Karte installiert, kann es nun beim Versuch, die neu installierte App wieder dorthin zu verschieben, zu folgender Fehlermeldung kommen: „App kann nicht verschoben werden“. Aber es ging doch zuvor auch?

Das Problem ist hier oftmals, dass Reste der App noch auf der SD-Karte verblieben sind – diese gilt es nun zunächst aufzuräumen. Das geht nicht am Gerät selbst, da Android (ohne root) den Zugriff auf das entsprechende Verzeichnis nicht zulässt. Daher muss die SD-Karte entnommen, und mittels eines Kartenlesers am PC eingebunden werden.

Zuerst gilt es, den Paketnamen der betroffenen App zu ermitteln. Dies geht am einfachsten, indem man sie auf der Website des Play Store (also mit dem Browser) aufsucht: Die URL nennt den Paketnamen sodann in der ID (Beispiel: Google Goggles findet sich unter <https://play.google.com/store/apps/details?id=com.google.android.apps.unveil> – der Paketname lautet hier also `com.google.android.apps.unveil`). Mit dieser Information bewaffnet, sucht man auf der SD-Karte im Verzeichnis `.android_secure` nun nach der passenden `.asec`-Datei, die bei unserem Beispiel etwa `com.google.android.apps.unveil-1.asec` heißen könnte – und löscht sie. Jetzt die Karte sauber vom PC trennen, zurück damit ins Gerät – und die App sollte sich nun wieder auf die SD-Karte verschieben lassen.

Auf meinem Android-Gerät kann ich keine Option zum Verschieben von Apps auf die SD-Karte finden!

Apps2SD wurde mit Android 2.2 eingeführt. Ist auf dem Gerät noch eine ältere Android-Version installiert, gibt es diese Möglichkeit schlicht nicht. Aber auch ab Android 4.0 haben sich einige Hersteller offenbar dazu entschlossen, ihren

Kunden diese Funktionalität nicht anbieten zu wollen. Beispiele dafür sind das *Samsung Galaxy S Duos*, oder auch das *LG Optimus 4X*: Auf beiden Geräten sucht man die passende Option vergeblich. Abhilfe schafft in diesen Fällen nur das [Rooten](#) des Gerätes, um auf alternative Möglichkeiten wie beispielsweise [Link2SD](#) auszuweichen.

Interne und externe SD-Karte vertauschen

Verfügt ein Android-Gerät sowohl über eine so genannte „interne SD-Karte“ als auch über einen Micro-SD Slot, verwenden die meisten Apps per Default die interne Karte. Dies ist besonders ärgerlich, sofern selbige nicht gerade üppig mit Speicher versehen ist. Da wünscht man sich oftmals, die Speicherplätze austauschen zu können.

Leider ist das nicht so ohne Weiteres möglich. Alle mir bekannten Möglichkeiten erfordern zumindest ein gerootetes Gerät. Die meisten Anleitungen beschreiben zu diesem Zweck die manuelle Bearbeitung der entsprechenden System-Datei, was nicht jedem gefallen dürfte (näheres dazu findet sich u. a. in einem [Stackexchange-Artikel](#)³⁷³). Eine einfachere Alternative bietet die App [External 2 Internal SD](#)³⁷⁴, die ursprünglich für das *Samsung Galaxy S3* geschrieben wurde. Sie funktioniert allerdings nicht auf jedem Gerät; es empfiehlt sich daher in jedem Fall, zunächst die Kommentare nach Erfolgsmeldungen zum eigenen Gerät zu durchforsten.



ASE: Storing data on
external SD Card in
Android 4



External 2 Internal
SD

373. <http://android.stackexchange.com/a/38053/16575>

374. <https://play.google.com/store/apps/details?id=eu.codlab.int2ext>

Netzwerk

Die Netzwerk-Icons in der Statusbar sind plötzlich weiß. Was hat das zu bedeuten?

Die Farbe der Netzwerk-Icons in der Statusbar (WLAN, Signalstärke, mobile Daten) zeigt ab Android 2.3 an, ob eine Verbindung zu den für Synchronisation etc. benötigten Google-Serven besteht. Sind die Icons Grau oder Weiß, besteht keine Verbindung. Grün (bis Gingerbread) bzw. Blau (ab Honeycomb) weisen auf eine bestehende Verbindung hin. Nachlesen lässt sich dies u. a. im [Android 2.3 Users Guide](#)³⁷⁵. Grob übersetzt heißt es dort:

Die Icons für den Netzwerkstatus werden Grün, wenn ein Google-Konto auf dem Android-Gerät konfiguriert, und das Gerät mit den Google-Services zur Synchronisation von GMail, Kalenderdaten, Kontakten, zum Backup der Einstellungen etc. verbunden ist. Ist kein Google-Konto konfiguriert, oder das Gerät beispielsweise mit einem WLAN verbunden, welches keinen Zugang zum Internet bietet, sind die Netzwerk-Icons Weiß.

Ab Android 4.4 sind die Icons jedoch standardmäßig weiß bzw. grau. Wer hier wissen möchte, ob ein Problem besteht, muss dazu auf die „Quick Settings“ zurückgreifen, in denen das entsprechende Icon dann rot bzw. orange eingefärbt ist. Auch die Anzeige, ob gerade Daten hoch- oder heruntergeladen werden (bis Android 4.3 durch kleine auf- bzw. abwärtsgerichtete Pfeile auf dem entsprechenden Icon in der Statuszeile angezeigt), wurde nun hierher verbannt.

Ich kann mit dem Browser auf Webseiten zugreifen – aber meine Apps kommen nicht ins Netz?

Die Ursache kann trivial sein. So ist dieses Verhalten normal, wenn die Zeiteinstellungen des Gerätes zu weit von der Realität abweichen (falsches Datum). In diesem Fall ist der Zugriff auf unverschlüsselte Websites („http“) zwar problemlos möglich, nicht aber der auf verschlüsselte Ressourcen („<https>“). Bei letzteren kommt es dann nämlich zu einem Fehler mit dem verwendeten Zertifikat, welches entweder noch nicht (Geräte-Datum zu weit in der



PDF: Android 2.3
Users Guide

375. <https://www.google.com/help/hc/pdfs/mobile/AndroidUsersGuide-2.3.4.pdf>

Vergangenheit) oder nicht mehr (Gerätedatum zu weit in der Zukunft) gültig ist. Abhilfe schafft in diesem Fall die Korrektur der Systemzeit des Gerätes.

Tritt das Problem jedoch ausschließlich bei mobiler Datenverbindung auf, könnte die Ursache auch mit dem verwendeten APN zusammenhängen, wie [ein Anwender bei Stack Exchange beschreibt](#)³⁷⁶. In diesem Fall überprüft man die [konfigurierten Zugangspunkte](#). Stehen für den Anbieter mehrere APNs zur Verfügung, verwendet man testweise einen anderen. Lässt sich das Problem damit nicht beheben, hilft eine Nachfrage beim Provider.

In seltenen Fällen hat sich einfach nur etwas „verhakt“, sodass ein Wechsel in den Flugzeugmodus und wieder zurück das Problem bereits behebt.

Wie bediene ich Webseiten, die ein „Mouse-Over“ Menü verwenden?

Offensichtliches Problem: Hat man keine Maus, kann auch kein Mauszeiger über einen Menüpunkt bewegt werden. Wie lässt sich ein solches Menü also auf „mauslosen Geräten“ wie Smartphones und Tablets bedienen?

Ein Ansatz ist wieder [bei Stack Exchange beschrieben](#)³⁷⁷: Langes Drücken der Stelle, über die man sonst die Maus bewegen soll, öffnet häufig das entsprechende Menü. Eventuell muss anschließend noch die „Zurück-Taste“ betätigt werden, um das Kontext-Menü der Browser-App zu schließen.

Stack Exchange:
Browser is OK, apps
cannot connect to
the internet



Stack Exchange: Web
browser with hover
mode

376. <http://android.stackexchange.com/a/15123/16575>

377. <http://android.stackexchange.com/q/25002/16575>

Wer frisst meine mobile Datenflat – und wie verhindere ich das?

Der Schreck war groß: Es ist erst der fünfte Tag des Monats, und schon war etwa ein Viertel meines Datenvolumens aufgebraucht. Dabei hat sich mein Nutzungs-Verhalten gar nicht geändert! Geht das so weiter, darf ich etwa ab dem 20. „langsamer surfen“. Was ist zu tun?

Zuerst einmal gilt es, den Übeltäter aufzuspüren. Dazu geht man auf dem Androiden zu *Einstellungen* > *Datennutzung*, und gelangt zu einem Bildschirm, wie er im Screenshot gezeigt wird. In meinem Fall war die erste aufgeführte Anwendung im unteren Bereich die *Google Play Store* App, mit gut 10 MB Datenverbrauch. Dabei hatte ich schon vor Urzeiten eingestellt, dass diese nur das WLAN benutzen darf! Irgend etwas scheint Google da also wieder einmal geändert zu haben. Legen wir diesem Missetäter daher das Handwerk:

Dazu aktivieren wir im genannten Bildschirm die Checkbox „Limit für mobile Daten einstellen“. Ist eigentlich kein „globales Limit“ gewünscht, schiebt man einfach den roten und den orangen Balken im Verbrauchsgraphen auf „utopische Werte“: So werden im dargestellten Beispiel die mobilen Daten bei Erreichen von 2,6 GB deaktiviert, nachdem bei 2,4 GB eine Warnung erfolgt. Mein monatlicher Datenverbrauch liegt normalerweise unter 100 MB; diese Werte sollte ich also nie erreichen.



Im nächsten Schritt widmen wir uns der verursachenden App. Tippt man diese in der Liste an, gelangt man zu den entsprechenden Details. Ganz am Ende dieser Seite findet sich wieder eine Checkbox, beschriftet mit „Hintergrunddaten einschränken“. Auch diese aktivieren wir nun. Als Folge darf die App nicht mehr automatisch im Hintergrund auf die mobile Datenverbindung zugreifen; Vordergrund-Aktivitäten (also explizit vom Anwender ausgelöste) funktionieren jedoch weiterhin, und auch das WLAN ist nicht betroffen.

In meinem Fall Erfolg auf ganzer Linie: Etwa zur Monatsmitte liegt die erwartete Datenmenge wieder innerhalb meines Kontingents. Auch konnte ich keine Probleme mit der *Google Play Store* App feststellen: Im heimischen WLAN

erhalte ich weiterhin meine Updates. Nur werden jetzt die *Google Play Services* nicht mehr aktualisiert, während ich „mobil unterwegs“ bin – was natürlich auch so gewünscht war.

Nebenwirkungen sind jedoch nicht ausgeschlossen, wenn die betroffene App im Hintergrund aktiv sein soll. Apps wie Facebook, WhatsApp, oder Twitter bekämen beispielsweise im mobilen Netz von eingehenden Nachrichten nichts mehr mit.

Eine schöne Zusammenfassung verschiedener möglicher Maßnahmen zur Daten-Einsparung findet sich übrigens [in einem Blogbeitrag bei AndroidPIT](#)³⁷⁸.



Datenverbrauch
einschränken: So
hält die Flatrate
länger durch

378. <http://www.androidpit.de/datenverbrauch-flatrate-einschraenken>

Telefonie

Rufumleitungen lassen sich nicht deaktivieren

Das Gepäck ist bereits aufgegeben, der Check-In absolviert, und der Aufruf zum Boarding erfolgt – ab geht es in die Sonne! Da fällt es einem siedend heiß ein: Die Rufumleitung zum Anrufbeantworter ist noch aktiviert! Das kann im Ausland teuer werden. Also schnell ins Android-Menü, und unter *Anrufe* › *Rufumleitungen* die entsprechende Rufumleitung abschalten. Und schon kann dem ahnungslosen Telefonie-Kunden ein neuer Schreck drohen, der in Form einer Fehlermeldung daher kommt: „Diese Funktion wird vom Diensteanbieter nicht unterstützt.“ Was nun?

Zeit für einen Anruf beim Service bleibt nicht mehr, das Boarding ist ja bereits eingeläutet. Auf diese Situation scheinen manche Anbieter zu hoffen, um noch ein paar zusätzliche Einnahmen generieren zu können: Nimmt man im Ausland einen Anruf nicht an, wird er nämlich häufig kostenpflichtig (zum Auslands-Tarif) auf die Mailbox weitergeleitet! Schließlich musste man das (nicht geführte) Gespräch zunächst zum Roaming-Partner, und von dort zur Mailbox zurück durchstellen.

Die schnelle Abhilfe: Ein Anruf bei der [magischen Nummer](#) ##002# deaktiviert sämtliche Rufumleitungen auf einen Schlag – also sowohl die bei Nichtannahme, als auch die bei Nichterreichbarkeit. Pech gehabt, gieriger Anbieter!

Ich fahre ins Ausland / Grenzgebiet. Wie vermeide ich Roaming-Kosten?

Die einfachste Möglichkeit ist natürlich, den Flugzeugmodus zu aktivieren – und ggf. bei Bedarf WLAN zu nutzen. Nicht immer ist dies aber das, was einem vorschwebt. Für das Datennetz gibt es einen einfachen Schalter in den Einstellungen unter „Drahtlos & Netzwerk“: Hier lässt sich das Roaming für die Datendienste deaktivieren (in der Regel ist dies auch die Voreinstellung). Darüber hinaus kann man an dieser Stelle auch den Netzbetreiber fest voreinstellen (manuell auswählen, anstatt automatisch wählen zu lassen). Somit bucht sich das Telefon auch für Gespräche nicht in ein fremdes Netz ein.

Besonders für Geschäftsleute ist es aber tragisch, wenn sie dadurch zeitweilig nicht erreichbar wären. Abhilfe verspricht für diesen Fall die App [Roaming Control](#)³⁷⁹: Mit ihr lässt sich für jedes Netz separat festlegen, ob ausgehende Anrufe erlaubt sein sollen, die Synchronisation deaktiviert bzw. die mobile Datenverbindung gekappt, oder gar in den Flugzeugmodus gewechselt wird. Auch allgemeine Regeln sind möglich, etwa für „EU Roaming“ oder unbekannte Netzwerke. Die App kostet zwar etwa dreieinhalb Euro – kann aber ein kleines Vermögen sparen helfen. Wer das zunächst prüfen möchte, findet auch eine gratis Testversion im Playstore.

Des weiteren gibt es hilfreiche Informationen im Internet, beispielsweise:



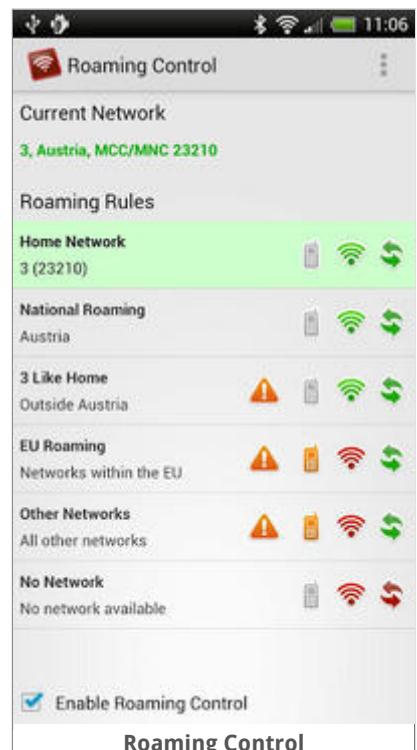
TechStage:
Telefonieren im
Ausland

- [Telefonieren im Ausland](#)³⁸⁰

Überblick bei TechStage: Welche Vorwahl hat das jeweilige Land, und wie können dort Ferngespräche geführt werden? Welche Mobilfunkprovider gibt es und welche Netze werden abgedeckt? Und: Passen meine Stecker für das Kaltstromkabel, ist die Spannung und die

379. <https://play.google.com/store/apps/details?id=com.insadco.roamingcontrol>

380. <http://www.techstage.de/news/Telefonieren-im-Ausland-Mit-TechStage-richtig-informiert-1927920.html>



Netzfrequenz in Ordnung – oder muss ich mir entsprechende Adapter besorgen?

- [So saugt Ihr Android-Smartphone weniger Daten](#)³⁸¹

Tipps von Spiegel Online zur Reduzierung des Datenvolumens – je nach Bedarf schrittweise bis auf Null



Spiegel Online: So saugt Ihr Android-Smartphone weniger Daten

Auf meinem Smartphone kann ich die SIP-Einstellungen nicht finden!

Unter [Internet-Telefonie](#) heißt es, seit *Gingerbread* gehören die SIP-Einstellungen zu den Bordmitteln. Dennoch sind sie auf einigen Geräten, trotz neuerer Android-Version, nicht auffindbar. Dies gilt beispielsweise für das *LG Optimus 4X*. Hat sich also an der Aussage etwas geändert?



Ja und nein. In [AOSP](#), also dem „originalen“ Android, finden sich diese Einstellungen nach wie vor an der entsprechenden Stelle. Nur kochen so einige Hersteller leider nicht nur diesbezüglich ihr eigenes Süppchen, wobei sie auch gern einmal das Eine oder Andere entfernen. Im konkreten Fall mag dies auch auf Wünsche der jeweiligen Netzanbieter zurückgehen, die SIP in ihren Flatrates ungern genutzt sehen. Doch zumindest beim genannten *LG Optimus 4X* wurde glücklicherweise lediglich der entsprechende Menü-Eintrag ausgeblendet – sodass man mit einem kleinen Trick wieder an die Einstellungen gelangt:

Eine Möglichkeit besteht in der Nutzung spezieller Funktionalitäten des [Apex Launcher](#)³⁸² (siehe Bild) oder auch *Nova Launcher*, Aktivitäten als Shortcut zum Homescreen hinzufügen zu können. Man drückt also einfach eine leere Stelle auf dem Homescreen, wählt „Verknüpfungen“, und sodann „Aktivitäten“. In der sich öffnenden Liste scrollt man nun zur App „Telefon“, und tippt diese an. Die sich öffnende Liste



Apex Launcher

381. <http://www.spiegel.de/netzwelt/apps/a-882238.html>

382. <https://play.google.com/store/apps/details?id=com.anddoes.launcher>

sieht nun etwa wie auf dem Screenshot aus: Zu „Sip Settings“ scrollen, antippen – und auf dem Homescreen wird der entsprechende Shortcut angelegt, mit dem man Zugriff auf die SIP-Einstellungen erhält.



Stanley

Natürlich lässt sich diese Prozedur auch für etliche andere „versteckte Dinge“ nutzen. Je nach Gerät und Android-Version kann man so einiges Interessantes zutage fördern! Wer jedoch keinen der beiden genannten Launcher installiert hat (und dies auch nicht tun möchte), dem bieten sich ebenso weitere Alternativen. Einen Blick wert sind z. B. [Stanley](#)³⁸³ zur Untersuchung installierter Apps auf verfügbare Aktivitäten (und weitere Eigenschaften), oder einfach der [Activity Launcher](#)³⁸⁴ für die direkte Erstellung der Shortcuts. Vorteil der letztgenannten App: Sie erlaubt auch die Zuordnung eines anderen Icons.



Activity Launcher

383. <https://play.google.com/store/apps/details?id=fr.xgouchet.packageexplorer>

384. <https://play.google.com/store/apps/details?id=de.szalkowski.activitylauncher>

Sicherheit & Privatsphäre

Ich habe mein Entsperr-Muster/Passwort vergessen!

Besteht gerade eine Netzwerk-Verbindung über 3G/2G (WLAN genügt eventuell nicht), gibt es noch Hoffnung: Einfach die Adresse des primären Google-Accounts („benutzername@googlemail.com“ – wobei „benutzername“ natürlich entsprechend zu ersetzen ist) und das dazugehörige Passwort eingeben. Andernfalls (oder falls das fehlschlägt), muss etwas tiefer in die Trickkiste gegriffen werden:

Die Zugangsdaten werden nicht akzeptiert, obwohl sie korrekt eingegeben wurden:

Manchmal scheint die Entsperrung fehlerhaft zu arbeiten. Ist dies der Fall, helfen folgende Schritte:

1. Den richtigen Benutzernamen, aber als Passwort „null“ (also genau diese vier Buchstaben) eingeben
2. Den Benutzernamen ohne @gmail.com (bzw. @googlemail.com) eingeben
3. Schritte 1 und 2 kombinieren
4. Per Browser das [Passwort-Recovery bei Gmail](#)³⁸⁵ verwenden, dann ggf. nochmals von 1. beginnen

(Quelle: [Einartysen](#)³⁸⁶)



GMail Password Recovery



Einartysen.se:
Unlock Android
phone after too
many pattern
attempts

Über das Web entsperren:

Dies setzt zweierlei voraus: Zum einen wieder eine bestehende Netzwerk-Verbindung auf dem gesperrten Gerät – und zum Anderen die Kenntnis des korrekten Sperrmusters (vielleicht hat ja nur der kleine Bruder zu oft versucht, sich Zugang zu verschaffen). Sind diese Voraussetzungen erfüllt, kann man folgendermaßen vorgehen:

1. Mit dem Google-Benutzernamen und zugehörigem Passwort bei Google anmelden
2. Entweder direkt den Link <https://accounts.google.com/IssuedAuthSubTokens>³⁸⁷ eingeben – oder in der rechten oberen Ecke des Browserfensters auf die eigene Mail-Adresse klicken, Konto auswählen, auf der nächsten Seite rechts auf Sicherheitseinstellungen verwalten klicken, und schließlich den Button Bearbeiten neben dem Schriftzug Autorisierung von Anwendungen und Websites betätigen.



Google Account:
Autorisierung von
Anwendungen und
Websites

385. <https://accounts.google.com/RecoverAccount>

386. <http://einartysen.se/unlock-android-phone-after-too-many-pattern-attempts/>

3. Unter *Verbundene Websites, Apps und Dienste* die Zugriffsrechte für den Android-Account widerrufen.
 3. Alternativ: Wer sein Google-Konto bereits für die Zwei-Schritt-Autorisierung eingerichtet hat und in der Lage ist, am Ende der Seite ein anwendungsspezifisches Passwort einzurichten, kann dies hier tun, und selbiges zum Entsperren des Gerätes verwenden.
 4. Jetzt auf dem Gerät mit den Zugangsdaten für Google Mail anmelden. Die Zugangsdaten sollten nun akzeptiert werden, und der Sperrbildschirm wird wieder angezeigt. Das „korrekte“ Sperrmuster zum Entsperren eingeben, und der Homescreen sollte wieder angezeigt werden.
- (Quelle: [UltraTechy](#)³⁸⁸)

Weitere Möglichkeiten:

Hat nichts von obigem Erfolg gezeitigt, bleiben nur noch technisch tiefer greifende Tricks – die für dieses Buch ein wenig zu weit gehen. Zu finden sind sie u. a. bei [Stack Exchange](#)³⁸⁹.

Was ist eine „Smudge Attack“, und wie schütze ich mich davor?

ASE: Cannot unlock device

Am Einfachsten lässt sich dieser Begriff mit „Schmutz-Attacke“, treffender jedoch als „Schmier-Attacke“ übersetzen. Sie basiert darauf, dass unsere Finger „ölige Spuren“ auf Touch-Screen-Displays hinterlassen – und nutzt diesen Fakt aus, um sich Zugang zum Gerät zu verschaffen.

Am Einfachsten ist dies, wenn der Anwender ein „Sperrmuster“ (auch als „Pattern Lock“ bekannt) verwendet. In einer Untersuchung stellte die University of Pennsylvania fest, dass sich das „Passwort“ in etwa 90% aller Fälle ermitteln ließ (siehe PDF: [Smudge Attacks on Smartphone Touch Screens](#)³⁹⁰). Jeder kann dies recht einfach für sich selbst überprüfen: Die „richtige Beleuchtung“ bringt das Muster „ans Licht“. Meist genügt dafür ein einfaches Drehen des Gerätes, um einen passenden Winkel zu bekommen. Einfacher machen es „gerichtetes Licht“ (z. B. Laser) und Foto-Equipment.

387. <https://accounts.google.com/IssuedAuthSubTokens>

388. <http://www.ultratechy.com/how-to-unlock-android-phone-after-too-many-pattern-attempts-without-factory-reset/>

389. <http://android.stackexchange.com/q/35847/16575>

390. http://static.usenix.org/events/woot10/tech/full_papers/Aviv.pdf

Wie schützt man sich nun dagegen? Bei [LifeHacker](#)³⁹¹ finden sich einige Vorschläge:

- Den Bildschirm nach jeder Nutzung gut putzen.
- Das Gegenteil: Ihn mit Zufalls-Mustern gut „verschmieren“.
- Zusätzlich zum Pattern-Lock noch ein PIN-Lock benutzen. Der PIN sollte dabei am besten mehr als 4 Ziffern „lang“ sein, ohne aus mehr als 4 Ziffern zu bestehen (also Wiederholungen enthalten); ein Angreifer würde dann nämlich i. d. R. von einer 4-Zeichen-Länge ausgehen.



LifeHacker: What's the Best Way to Prevent Touch Screen Smudge Attacks?

Wie bereite ich mein Gerät für den Verkauf vor?

Irgendwann ist ein neues Gerät fällig, und das alte soll einen neuen Besitzer finden. Letzterer sollte jedoch nicht unbedingt die Chance erhalten, auf unsere privaten Daten zuzugreifen. Daher ist vor Verkauf/Versteigerung/Verschenkung ein gründlicher „Hausputz“ angesagt:

- **Externe SD-Karte:** Da dieses Speichermedium einfach entnommen werden kann, ist „behalten“ sicher die einfachste Möglichkeit. Die Karte passt schließlich auch in das neue Gerät. Soll sie partout im Altgerät verbleiben, dann
 1. Alle darauf befindlichen Daten (insbesondere Fotos, Videos, Dokumente, sowie die Musik-Sammlung) auf den eigenen Computer bzw. das neue Gerät kopieren
 2. Die Karte löschen. Dafür findet sich unter *Einstellungen > Speicherverwaltung* der Punkt „Speicherkarte Löschen“
 3. Löschen allein hilft zwar vor vielen „neugierigen Blicken“ – doch gehört nicht viel technische Finesse dazu, die Daten wieder herzustellen. Gegebenenfalls also die Karte besser formatieren, oder „wipen“ (s. u.)
- **Interne SD-Karte:** Diese kann leider nicht einfach „einbehalten“ werden. Es gilt also „Plan B“: Wie bei der externen Karte, zunächst alle Daten sichern, dann die Karte löschen.
- **Interner Speicher:** Diesem sollte ein [Factory-Reset](#) den Garaus machen. Leider gilt jedoch auch hier: Wer technisch ein wenig versierter ist, kann Daten wiederherstellen. Nochmals daher der Hinweis auf den Wipe.

Vor dem Factory-Reset sollte also sichergestellt werden, dass kein Unbefugter private Daten wiederherstellen kann. Ein „sicheres Löschen“, auch „Wipe“ genannt, verhindert dies. Dafür stehen im Playstore verschiedene Apps bereit.

391. <http://lifehacker.com/-757563408>



Secure Wipe



Übersicht: Sicheres Löschen



Cerberus

Etwa Secure Wipe³⁹², das sich gleich um sämtliche Speichermedien (externe/interne SD-Karte und internen Speicher) kümmert. Nachdem alle Dateien gelöscht sind, überschreibt diese App den „freien Bereich“ – von den gelöschten Dateien bleiben somit keine verwertbaren oder gar wiederherstellbaren Spuren. Allerdings kümmert sie sich nur um den „leeren“ Speicherplatz: Noch existierende Dateien werden nicht „gewiped“. Da man dem neuen Besitzer jedoch ohnehin keine Dateien verkaufen möchte, löscht man am Besten zuerst alles – und setzt dann *Secure Wipe* oder vergleichbare Kandidaten³⁹³ auf den Speicher an.

War das Gerät gerootet, und vielleicht eine Custom-ROM oder eine Sicherheits-App wie Cerberus³⁹⁴ installiert, sollte man evtl. überlegen, die (Original) Firmware neu zu flashen – um auch die letzten Überreste zu beseitigen.

Last but not Least: Nicht die SIM-Karte im Gerät vergessen!

Kann die Einstellung „unbekannte Quellen“ von Apps zur Installation anderer Apps missbraucht werden?

Unter *Einstellungen > Sicherheit* findet sich die Option „unbekannte Quellen“. Will man Apps aus einer anderen Quelle als dem *Google Playstore* installieren, muss diese aktiviert werden. Dennoch erscheint bei jeder Installation ein Dialog, den der Anwender bestätigen muss.

Die Installation von Apps ist dem Package Manager vorbehalten. Die dafür notwendige Permission INSTALL_PACKAGES wird ausschließlich System-Apps gewährt – also solchen Apps, die unter /system (im ROM) installiert sind. Sollte also eine „normale“ App einen Missbrauchs-Versuch starten, erscheint dennoch die Sicherheits-Abfrage.

Was hat es mit den ACCOUNT Permissions auf sich? Ist da besondere Vorsicht geboten?

Für die Verwaltung von Benutzer-Konten für verschiedene Dienste stellt Android einen zentralen Dienst bereit: Den Account-Manager. Dieser verwaltet Zugangsdaten wie Benutzername und Passwort, sodass dies nicht jede App selbst tun (und der Anwender die Daten auch nicht für jede separat erfassen) muss. Verschiedene Permissions sichern dabei die erforderlichen Aktivitäten ab:

392. <https://play.google.com/store/apps/details?id=com.pinellascodeworks.securewipe>

393. http://android.izzysoft.de/applists/category/named/security_datawipe#group_457

394. <https://play.google.com/store/apps/details?id=com.lsdroid.cerberus>

ACCOUNT_MANAGER

Diese Permission wird nur System-Apps (i. d. R. also vorinstallierten Apps) gewährt. Ein Account-Manager arbeitet sozusagen „hinter den Kulissen“ und achtet darauf, dass alles mit rechten Dingen zugeht.

AUTHENTICATE_ACCOUNTS

Eine App mit dieser Berechtigung stellt üblicherweise eine (dem System sonst nicht bekannte) Schnittstelle für einen Dienst bereit – bringt Android also bei, wie bei diesem die Authentifizierung vonstatten geht.

GET_ACCOUNTS

Erlaubt es, eine Liste verfügbarer Accounts vom System abzufragen. So kann eine App etwa feststellen, ob ein Dropbox-Account verfügbar ist, und davon abhängig bestimmte Menüs anzeigen.

MANAGE_ACCOUNTS

Diese App möchte Accounts verwalten – also hinzufügen, ändern, oder löschen. Zwar konnte ich keine offizielle Dokumentation finden, die auch entsprechende Details preisgibt. Doch soweit sich herausfinden ließ, kann eine App mit dieser Berechtigung zwar beliebige Accounts anlegen – jedoch nur diejenigen ändern oder löschen, die sie auch selbst erstellt hat.

USE_CREDENTIALS

Hier geht es darum, im Account-Manager hinterlegte Informationen zu verwenden. Eine App mit dieser Permission darf also konfigurierte Konten benutzen. Das heißt allerdings nicht zwangsläufig, dass sie die entsprechenden Login-Daten zu sehen bekommt: I. d. R. erhält sie vom Account-Manager lediglich ein „Token“, der sie gegenüber dem Service als „angemeldet“ ausweist. Der Account-Manager sorgt ferner dafür, dass der Anwender der Verwendung eines Kontos durch die App vor der ersten Anmeldung zustimmen muss.

Wer sich für tiefergehende Details interessiert, dem seien zwei Artikel bei Stack Exchange nahegelegt:

- [What does permission „MANAGE_ACCOUNTS“ mean?](#)³⁹⁵ beschäftigt sich in erster Linie mit dem Account-Manager
- [What can an app do with the „USE ACCOUNTS ON THE DEVICE“ permission?](#)³⁹⁶ widmet sich Apps, welche die Account-Informationen nutzen wollen.



Stack Exchange:
What does
permission
„MANAGE_ACCOUNTS“
mean?



Stack Exchange:
What can an app do
with the „USE
ACCOUNTS“
permission?

395. <http://android.stackexchange.com/q/44293/16575>

Welche Android-Einstellungen betreffen meine Privatsphäre?

Davon gibt es eine ganze Reihe, und je nach verwendeten Apps kommen weitere hinzu. Eine vollständige Liste ist daher schwer zu erstellen, weshalb ich mich hier auf einige zentrale Dinge beschränken muss:

Google Cloud Backup

Eine der ersten Fragen bei der Einrichtung eines Google-Accounts ist: „Möchten Sie Ihre [Daten bei Google sichern](#)?“ Klingt zunächst gut, hat aber einige Haken: Nicht alle Apps unterstützen dies, die Wiederherstellung klappt nicht immer, und sensible Daten (wie beispielsweise WLAN-Passwörter) werden teilweise im Klartext übertragen und gespeichert.

Smart Lock für Passwörter

Dieses Feature findet sich in der App *Google Einstellungen* und soll es ermöglichen, dass die Passwörter Ihrer Apps und Websites in Ihrem Google-Konto gespeichert werden.

Synchronisation von Kalendern, Kontakten, etc.



OwnCloud



Funambol

IzzyOnDroid –
Android ohne
Google 2: ownCloud

Selbst wenn die vorige Frage mit „Nein!“ beantwortet wurde, ist diese Synchronisation standardmäßig aktiviert, sobald ein entsprechender Account eingerichtet wurde. Hier muss ggf. also unter *Einstellungen > Konten & Synchronisation* selbst Hand angelegt werden. Um einer versehentlichen Synchronisation vorzubeugen, sollte man evtl. die Voreinstellungen in den jeweiligen Apps von „Google Account“ auf entsprechende lokale Datenspeicher umstellen. Eine Synchronisation kann dann übrigens auch mit dem eigenen Server stattfinden: Software-Pakete wie [OwnCloud](#)³⁹⁷ oder [Funambol](#)³⁹⁸ schaffen die entsprechenden Voraussetzungen. Wie Android und ownCloud zusammenarbeiten, findet sich u. a. in einem [Artikel bei IzzyOnDroid](#)³⁹⁹ beschrieben.

Beginnend mit Android 4.4 wurden SMS in *Hangouts* integriert. Eine separate SMS-App ist somit nicht mehr von Haus aus vorhanden. Es kann jedoch auf Apps von Drittanbietern zurückgegriffen werden, die sich auch leicht als „Standard SMS App“ im System einrichten lassen. *Hangouts* bietet ebenso die Option, die SMS-Integration zu deaktivieren.

396. <http://android.stackexchange.com/q/38369/16575>

397. <http://owncloud.org/>

398. <http://www.funambol.com/>

399. <http://android.izzysoft.de/articles/named/android-without-google-2>

Galerie

Hier könnte die Stand-Alone App ebenfalls bald verschwinden – auch wenn sich gerade wieder ein gegenteiliger Trend⁴⁰⁰ abzeichnet. Derzeit ist das Ganze in *Google+* integriert – welches auch sogleich sämtliche Bilder in die Cloud hochladen möchte. Wie schon bei den SMS, gibt es aber auch hier genügend alternative Apps von Drittanbietern.



Bloomberg: Google Said to Plan

Google Maps

Sammelt auch gern, insbesondere Standort-Daten. Natürlich lässt sich die App deaktivieren, und durch passende Alternativen ersetzen. Zu nennen wären hier insbesondere OsmAnd⁴⁰¹ und OruxMaps⁴⁰².



OsmAnd



OruxMaps

Standort-Daten

Konnte man bis Android 2.1 noch auf die Google Standort-Dienste zugreifen, ohne den eigenen Standort permanent mitzuteilen, ist dies mittlerweile nicht mehr möglich. So hat man unter *Einstellungen > Standordienste* nun nur die Möglichkeit, beides zu erlauben oder beides zu verbieten. Da viele Apps auf diese Dienste zugreifen (machen sie doch auch bei nicht verfügbarem GPS zumindest eine grobe Standort-Bestimmung über Mobilfunkzellen und WLANs möglich), sollte man bei einer Deaktivierung prüfen, welche Apps mit „Nebenwirkungen“ zu kämpfen haben. Unter Android 4.4+ gibt es überdies noch einen zweiten Schalter zu beachten: In *Einstellungen > WLAN > Erweitert* gibt es noch den Netzwerk-Scan bei abgeschaltetem WLAN.



GamersGlobal:
Android ohne
Google

Wem das noch nicht weit genug geht: Android ohne Google ging schon 2012⁴⁰³, und funktioniert auch 2014⁴⁰⁴ mindestens genau so gut.



IzzyOnDroid:
Android ohne
Google

400. <http://www.bloomberg.com/news/2014-08-01/google-said-to-plan-separating-photo-service-from-google-.html>

401. <https://play.google.com/store/apps/details?id=net.osmand>

402. <https://play.google.com/store/apps/details?id=com.orux.oruxmaps>

403. <http://www.gamersglobal.de/news/62752/>

404. <http://android.izzysoft.de/articles/named/android-without-google-1>

Weiteres

Wie kann ich ohne root Screenshots vom Android Handy erstellen?

Dies lässt sich u. a. bewerkstelligen, indem man eine der im Abschnitt „[Das Android-Gerät vom PC aus verwalten](#)“ genannten Apps benutzt, wie etwa den *PAW Server*. Screenshots erstellt man dann einfach vom PC aus.

Ab Android 4.0 geht das jedoch auch viel einfacher: Hier gibt es häufig eine Tasten-Kombination, um einen Screenshot auszulösen. Leider unterscheiden sich diese bei Geräten unterschiedlicher Hersteller. Bei Geräten von Samsung sind meist die Power- und die Home-Taste gleichzeitig zu drücken. Bei Geräten von Motorola (und den meisten anderen Herstellern) hält man dazu die Power- und die Leiser-Taste gleichzeitig gedrückt. Auch das [Power-Menü](#) hält meist eine passende Aktion bereit.

Tipps für eine Reihe weiterer Geräte mit Besonderheiten finden sich in einem [Blogbeitrag bei AndroidPIT](#)⁴⁰⁵. Linux-Nutzer könnten sich auch für das kleine ADB-Kommandozeilen-Tool [Snap-Android](#)⁴⁰⁶ interessieren.

Einen besonderen Tipp für diejenigen, die bereits Marshmallow auf ihrem Gerät haben, weiß [Phandroid](#)⁴⁰⁷ zu benennen:

1. Den Home-Key lange drücken
2. Ganz Links das „Share“ Icon antippen
3. Eine Animation zeigt nun, dass ein Screenshot gemacht wird
4. Es öffnet sich das Share-Menü, aus dem sich eine App zur „Weiterverarbeitung“ des Screenshots auswählen lässt.

Netter Nebeneffekt: Bei den so erstellten Screenshots wurde auch gleich die Status-Leiste geleert, so dass etwaige Benachrichtigungen nicht sichtbar sind. Ideal also, um Screenshots für Blog-Artikel und ähnliches zu erstellen.

405. <http://www.androidpit.de/screenshots-mit-android-erstellen>

406. <https://github.com/snapwire-media/snap-android>

407. <http://phandroid.com/2016/01/13/android-marshmallow-screenshots/>

Das Share-Menü funktioniert plötzlich nicht mehr!

In Android 4.4 hat sich hier ein Bug eingeschlichen, der scheinbar einige, jedoch nicht alle Anwender betrifft: Es lässt sich plötzlich nicht mehr auswählen, mit welcher App ein bestimmter Inhalt geteilt werden soll. Stattdessen wird sofort zur zuletzt für diese Aktion gewählten App gesprungen – ganz so, als hätte man diese permanent als Default eingestellt.

Ja, natürlich löst ein [Factory-Reset](#) das Problem. Doch da es sich um einen „Zombie“ handelt, taucht es mit an Sicherheit grenzender Wahrscheinlichkeit anschließend recht bald wieder auf. Eine bessere (und angenehmere) Möglichkeit ist das Verwenden eines alternativen Share-Menüs. Diese Möglichkeit bieten Apps zur [Modifikation des Share-Menüs](#)⁴⁰⁸, wie z. B. [Andmade Share](#)⁴⁰⁹.

Ich bekomme plötzlich Werbung in der Benachrichtigungsleiste eingeblendet!

Airpush und Kollegen lassen Grüßen. Klar, dass sich auch Programmierer von etwas ernähren müssen – aber das geht definitiv zu weit. Es gibt jedoch mehrere Möglichkeiten, sich zu wehren. So lässt sich die verursachende App ermitteln (ab [Jelly Bean](#) geht das von Haus aus: Einfach lange auf die entsprechende Benachrichtigung drücken, und dann das sich öffnende Kontext-Menü nutzen), und der Unsinn abstellen. Tauchen die nervigen Anzeigen nicht als Benachrichtigung auf, lassen sich die Übeltäter oftmals noch immer mit passenden Helferlein identifizieren. Eine aktuelle Übersicht zum Thema findet sich bei [IzyOnDroid](#)⁴¹⁰.

Mein Gerät hängt in einer Force-Close-Schleife

Das Gerät bootet. Unmittelbar nach dem Hochfahren startet eine App, die aufgrund eines Fehlers abstürzt und sich gleich wieder neu startet – um wieder abzustürzen und neu zu starten, um wieder ... Das nennt man einen „Force Close Loop“. Das Schlimme daran: Während dieses Loops ist das Gerät auf keine Weise bedienbar, es reagiert auf keinerlei Eingaben – man kommt aus dieser Schleife also nicht heraus.

Für Fälle wie diesen gibt es bei Android den „[Safe Mode](#)“ (vielen von Windows als „Abgesicherter Modus“ bekannt). Startet man in diesen, so ignoriert das



Übersicht: Share-Menü modifizieren



Andmade Share



Übersicht:
Werbemodule und
Privacy-Checker

408. http://android.izzysoft.de/applists/category/named/tools_settings#group_547

409. <https://play.google.com/store/apps/details?id=com.andmadesoft.share>

410. http://android.izzysoft.de/applists/category/named/network_admodules

Android-Gerät alle Benutzer-Apps – so, als wären sie gar nicht installiert. Die Force-Close-Schleife sollte damit umgangen werden. Nun kann der Anwender die „missratene App“ deinstallieren, und das Gerät wieder im normalen Modus starten – das Problem sollte gebannt sein.

In den „Safe Mode“ gelangt man auf den meisten Geräten mit folgenden Schritten:

- Gerät ausschalten, Akku herausnehmen und wieder einsetzen
- Menü-Taste drücken, und bei gedrückter Menütaste das Gerät einschalten (verfügt das Gerät über keine „feste Menü-Taste“: Einschalten, und dann gleichzeitig die „Leiser“ und „Lauter“ Tasten drücken und gedrückt halten)
- Sobald der Sperrbildschirm erscheint, kann die Menü-Taste (bzw. die Lautstärke-Wippe) wieder losgelassen werden. In der unteren linken Ecke sollte nun der Schriftzug „Safe Mode“ zu sehen sein.

Hat man das Problem beseitigt, möchte man natürlich gern wieder den „Safe Mode“ verlassen:

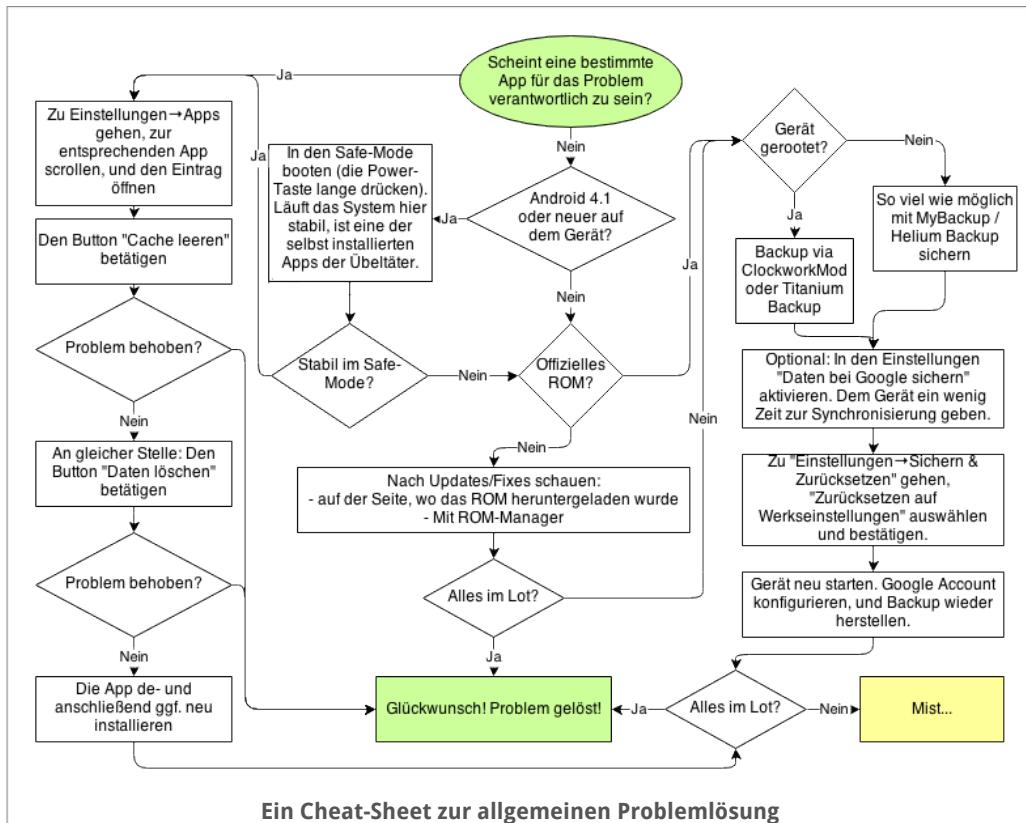
- Gerät ausschalten, Akku herausnehmen und wieder einsetzen
- Gerät wieder normal einschalten. Dabei nur die Power-Taste benutzen, und keine weiteren Tasten gleichzeitig betätigen.

In seltenen Fällen kann es passieren, dass das Gerät den „Safe Mode“ nicht wieder verlassen möchte. Hier helfen folgende Schritte:

- Gerät ausschalten, Akku und SIM-Karte entnehmen, sowie für mindestens 20 Sekunden entfernt lassen (damit keine eventuelle Restladung verbleibt). Anschließend SIM-Karte und Akku wieder einsetzen, und neu starten. Natürlich sollte während dieses Vorgangs auch kein Ladekabel angeschlossen sein.
- Ist der „Safe Mode“ noch immer aktiv, bootet man in den Recovery-Modus und bereinigt die Cache-Partition („Wipe Cache“). Wie man in den Recovery-Modus gelangt, ist von Gerät zu Gerät unterschiedlich. Meist geschieht dies, indem man bei ausgeschaltetem Gerät die „Leiser“ und „Power“ Taste gleichzeitig drückt und gedrückt hält, bis das Recovery-Menü zu sehen ist.
- Hat auch das nicht geholfen, bleibt als letzte Möglichkeit der Factory Reset (Zurücksetzen auf Werkseinstellungen).

Mein Androide spinnt!

Hm, keine sehr spezifische Angabe. Aber wir haben auch „allgemeine Antworten“:



Mein Touchscreen spinnt, wenn das Gerät am Ladekabel hängt!

Hierfür kann es verschiedene Ursachen geben, wie auch ein [Artikel bei Stack Exchange](#)⁴¹¹ ausführt. In der Regel liegt dabei aber weder ein Software-Problem, noch ein Fehler am Gerät selbst vor. Vielmehr sind entweder das Kabel oder das Netzteil die Verursacher.

So kann beispielsweise ein Billig-Netzteil schlecht abgeschirmt sein, und damit für Störungen des auf [elektromagnetische Störungen](#)⁴¹² empfindlich reagierenden Touchscreens sorgen.

Wahrscheinlicher ist hingegen, dass eine unterschiedliche Pin-Belegung für die Probleme verantwortlich ist. Eine Beschreibung der Pin-Belegung von USB-



ASE: Why does my phone have erroneous input when connected to a non-OEM power source?



Wikipedia:
Elektromagnetische
Störung

411. <http://android.stackexchange.com/q/39424/16575>

Adaptern findet sich beispielsweise bei [PinOuts](#)⁴¹³. Ins Deutsche übertragen, sieht diese etwa folgendermaßen aus:



PinOuts.Ru: Mini-USB connector of cell phones pinout

Pin	Name	Beschreibung
1	VCC	+5V DCC
2	D-	Data -
3	D+	Data +
X		Dieser Pin kann zur Kabelerkennung in einigen Fällen mit GND verbunden sein
4	GND	Masse

Zu beachten ist hier der „X“ Pin, der mit GND verbunden sein *kann* – aber nicht *muss*. Während der eine Hersteller diese Belegung also sauber auswertet, kann sie bei Geräten eines anderen Herstellers durchaus Probleme hervorrufen.

Da ich selbst einen solchen konkreten Fall erlebt habe, bin ich der Sache nachgegangen. Beteiligt waren drei Kabel verschiedener Hersteller, drei Netzteile, sowie ein HTC und ein Motorola Smartphone (das dritte Kabel sowie Netzteil waren „generisch“). Fazit: Lediglich das Motorola-Gerät „spann“, sobald das HTC-Netzteil ins Spiel kam – alle anderen Kombinationen liefen problemlos.

412. http://de.wikipedia.org/wiki/Elektromagnetische_St%C3%B6rfung

413. http://pinouts.ru/CellularPhones-A-N/cellphone_miniusb_pinout.shtml

TIEFERGEHENDES FÜR FORTGESCHRITTENE

Nachdem sich die ersten beiden Teile dieses eBooks hauptsächlich an Einsteiger gerichtet haben, sollen auch die Fortgeschrittenen unter den Lesern nicht zu kurz kommen. Die hier behandelten Themen sind mit Sicherheit nichts für Neueinsteiger: Bevor man sich an die Umsetzung der „schweren Kost“ macht, sollte man mit seinem Android-Gerät schon recht gut vertraut sein.

Dennoch heißt das nicht, dass diese jetzt das „Buch aus der Hand“ legen müssen. Ich werde versuchen, möglichst allgemeinverständlich zu schreiben (auch auf die Gefahr hin, dass sich der eine oder andere mitlesende Profi ein wenig „verscheißert“ vorkommen könnte). Dies verschafft zumindest einen Überblick über sich bietende Möglichkeiten. Und als Seiten-Effekt findet sich (insbesondere im [Tuning-Bereich](#)) sicher auch der eine oder andere hilfreiche Tipp für Neulinge.

Aber genug der Vorrede – kommen wir zum Thema. Oder besser zu den Themen:

Der Super-User „root“

Kauft man einen Windows-PC, gibt es auf diesem einen Account für den Benutzer „Administrator“ – dem man bei der Ersteinrichtung ein Passwort verpasst. Installiert man Linux, heißt das Pendant „root“ (bei einem Mac sicher ähnlich). Android basiert auf Linux – aber trotzdem gönnen uns die Hersteller den root-Zugang in der Regel nicht, sondern drohen: „Wer sich root-Zugang zu seinem Gerät verschafft, verwirkt damit den Garantieanspruch.“



EU Richtlinie: Führt ein Root / Flash zum Gewährleistungsausschluss?

Der User „root“ ermöglicht also den administrativen Zugang zum System, mit dem man alles (kaputt) machen kann. Naja, fast alles – die Hardware wohl eher nicht. Weshalb die Warnung mit der Garantie wohl letztendlich vor Gericht kaum haltbar sein dürfte, wenn man z. B. das Display wechseln lassen muss, oder der interne Speicher den Geist aufgibt (anders sieht es aus, wenn die CPU verglüht, weil man sie hoffnungslos übertaktet hat). Eine [EU-Richtlinie stellt hier sicher](#)⁴¹⁴, dass der Gewährleistungs-Anspruch trotz root nicht erlischt. Für detailliertere Beschreibungen des Unterschieds zwischen „Garantie“ und „Gewährleistung“ sei auf [Wikipedia](#)⁴¹⁵ und [Anwalts-Seiten.DE](#)⁴¹⁶ verwiesen.



Wikipedia:
Abgrenzung der
Qualitätsgarantie
gegenüber der
Gewährleistung

Braucht man den root-Zugang nun wirklich? Ja und nein. Wer mit seinem Gerät, dessen Funktionen, sowie der verwendeten Software bereits rundum zufrieden ist, alles so läuft, wie gewünscht, und „eigentlich“ nichts vermisst – der braucht auch keinen root-Zugang. Er hat ja bereits alles, was er braucht. Hat man hingegen ein Problem, was sich ohne den root-Zugang nicht lösen lässt, sieht das schon anders aus: Je nachdem, wie schwer es einen trifft, neigt sich das Zünglein an der Waage immer mehr der Anzeige zu, die mit „mach mich root!“ beschriftet ist.



Anwalts-Seiten.DE:
Abgrenzung der
Qualitätsgarantie
gegenüber der
Gewährleistung

Welche Vorteile sind es denn nun, die man mit einem root-Zugang erlangt – und welche Risiken sind damit ggf. verbunden?

Vorteile des root-Zugangs

Verschiedenste Einstellungen und Änderungen lassen sich ohne root-Zugang gar nicht vornehmen:

414. <http://www.mobilfunk-talk.de/news/92561-ratgeber-fuhrt-ein-root-zum-gewahrleistungsausschluss/>

415. https://de.wikipedia.org/wiki/Garantie#Abgrenzung_der_Qualit%C3%A4tsgarantie_gegen%C3%BCber_der_Gew%C3%A4hrleistung

416. <http://www.anwalt-seiten.de/artikel/sec1/221.html>

- [Anpassen der CPU Taktfrequenz](#) (siehe auch [Akkuleistung](#))
- [Entfernen/Deaktivieren vorinstallierter Apps](#) (Deaktivieren geht ab Android 4.0 auch ohne root)
- Bearbeiten der Start-Events (siehe [Apps am automatischen Starten hindern](#))
- Optimierung der Speicherverwaltung (siehe [Tuning](#))
- automatische Datenbereinigung (Reste de-installerter Apps; siehe [Unnütze Apps raus!](#))
- App2SD bei Android < 2.2 (siehe [Tuning](#)) und entsprechendes bei Android > 4 (siehe [Apps auslagern](#))
- Aufspielen alternativer Firmware (aka [Custom ROM](#))
- Ändern der Systemschriftart(en)
- Erstellen eines wirklich vollständigen Backups des Android-Systems (erst ab Android 4.0 [ohne root möglich](#), siehe [vollständiges Backup ohne root](#))
- Einrichten einer Firewall (siehe [Zugriffe Sperren: Firewalls & Permission-Blocker](#))

Diese Liste ist keinesfalls vollständig (natürlich auch nicht nach Relevanz sortiert – die wäre ohnehin wieder sehr subjektiv). Mit root hat man quasi überall Zugang – keine Ecke des Android-Systems bleibt verschlossen. Genau da liegt auch das Risiko – aber da liegt es auch beim root-Zugang auf dem Linux PC, oder dem Administrator-Zugang beim Windows-PC:

Risiken des root-Zugangs

Die Risiken sind schnell mit einem Satz beschrieben: Falsch angewendet, kann man sich mit root-Zugang das System unbrauchbar machen. Im schlimmsten Fall verwandelt man gar seinen Androiden in einen Ziegelstein – wenn man z. B. ohne Sinn und Verstand die CPU hoffnungslos übertaktet, und diese schließlich den Hitzetod stirbt. Mit Wissen und Verstand eingesetzt, ist der root-Zugang ein mächtiges und nützliches Werkzeug. Quasi wie ein Autoschlüssel: Setzt sich der 8-jährige Steppke damit hinters Steuer ... Was den Eindruck erweckt, dass man uns für „unmündig“ hält.

Oder sich schlicht vor unnötigen Rückgaben und Garantie-Einforderungen von sich selbst überschätzenden Anwendern schützen will. Ein nachvollziehbarer Grund – denn solche Anwender gibt es leider zu viele. Da es somit i. d. R. keinen root-Zugang ab Werk gibt, liegt für Anwender mit Sinn und Verstand das größere Risiko eher in der Erlangung eines solchen. Je nach Gerät und Verfahren ist dieses größer oder fast gar nicht vorhanden. Da die Höhe des Risikos jedoch vom verwendeten Verfahren abhängt, lässt sich hier keine allgemeingültige Aussage



treffen. Für weitere Details bietet ein [Artikel bei Stack Exchange](#)⁴¹⁷ einen guten Einstieg.

Wie bekomme ich root-Zugang?

Das jetzt so zu erklären, dass es für jeden gilt, führt ein wenig zu weit. Für diese Übersicht kurz zusammengefasst, gibt es da mehrere Möglichkeiten – und je nachdem, um welches Gerät es geht, greift davon eine, keine, oder mehrere. Was Ihnen jedoch fast ausnahmslos gemein ist: Sie alle nutzen vorhandene Sicherheitslücken im System, um root-Rechte zu erlangen.

Da ist zum einen „Software-root“: Man lädt sich die passende App auf den Androiden, startet sie, und bestätigt: „Ja, ich will root!“. Fertig. Toll: Mit so einem Gerät fühle ich mich absolut sicher. Wer sagt mir, dass eine andere App das nicht im Hintergrund tut, ohne mich zu fragen?

OK, auch die zweite Variante ist im Prinzip eine Art „Software-root“ (schließlich geht es ja um Software-seitigen Zugang). Nur geht es hier nicht um eine „einfache App“, sondern es ist schwieriger: Zunächst muss das USB-Debugging im Gerät aktiviert werden (expliziter Schritt, schwer von einer App auszuführen). Dann ist der Androide per USB-Kabel mit dem PC zu verbinden (unmöglich, dass das eine App im Hintergrund macht). Und schließlich muss man auf dem PC die „root-Software“ starten, die über das Kabel auf das Android-Gerät zugreift. Die Schritte sind noch immer einfach und nachvollziehbar – aber hier habe ich keine Bedenken, dass das ohne mein Zutun passieren könnte.

Variante Nummer drei ist das Einspielen eines speziellen update.zip über das [Recovery-Menü](#), was jedoch nur in wenigen Fällen möglich ist. Denn entweder muss hierzu besagtes update.zip vom Hersteller des Gerätes signiert, oder aber die entsprechende „Prüfung“ unterbunden worden sein.

Welche Variante jetzt für ein bestimmtes Gerät verfügbar ist, und welche Software dafür benötigt wird, recherchiert man am besten im Forum. Bei vielen Communities gibt es gerätespezifische Foren, und meist gibt es dort wiederum ein Unter-Forum für root-Fragen – wo sich die Informationen finden, die für das jeweilige Gerät zutreffend sind. Und natürlich findet man auch [bei Stack Exchange treffsichere Hilfe](#)⁴¹⁸.



ASE: How do I root my device

417. <http://android.stackexchange.com/q/35943/16575>

418. <http://android.stackexchange.com/q/1184/16575>

Für fast alle Android-Geräte bis Android 4.4.2 sollte sich [Towelroot](#)⁴¹⁹ von Geohot eignen, welches sich eine Lücke im Linux-Kernel zunutze macht (die mit Android 4.4.3 geschlossen wurde). *Towelroot* gehört in die erstgenannte Kategorie – also .apk Datei herunterladen, installieren, und auf den „Mach-mich-root-Button“ drücken.

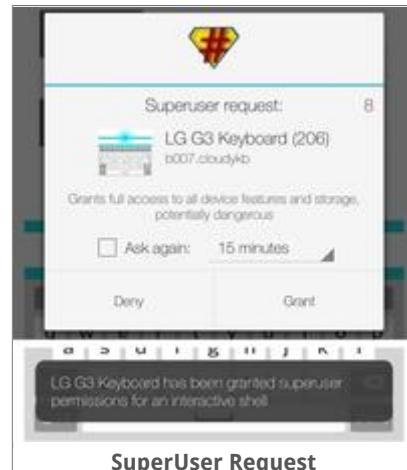


Laufen dann alle Apps mit root-Rechten?

towelroot by geohot

Eine oft aufkommende Befürchtung – zum Glück unbegründet. Also die kurze Antwort: Nein, nicht ohne ausdrücklichen Wunsch des Anwenders.

Für eine detaillierte Antwort muss ich etwas tiefer greifen. Und wir müssen uns in Erinnerung rufen: Ein Android-System läuft ja mit Linux, also gelten hier auch entsprechende Richtlinien. Und jede App läuft darüber hinaus unter einem eigenen Benutzer (das gilt auch, wenn auf einem Gerät mit Android 4.2 oder neuer mehrere Konten bzw. Profile eingerichtet sind). *root* ist ebenfalls ein Benutzer, wenn auch ein ganz spezieller. Und wenn eine „normale App“ etwas mit root-Rechten ausführen möchte, muss sie „root“ dazu auffordern. Der Befehl dazu heißt [sudo](#)⁴²⁰, was wir in unserem speziellen Kontext mit „SuperUser, DO ...“ wiedergeben können.



SuperUser Request

Wenn eine App selbst unter „root“ läuft, braucht sie auch kein „sudo“. Das betrifft aber unter Android nur System-Apps, auf die der Anwender in der Regel keinen (direkten) Zugriff hat.

Läuft sie jedoch nicht unter „root“ (und das ist bei Android die Regel: Jede App läuft, wie bereits gesagt, unter einem eigenen User), dann muss sie für Aktionen, die root-Rechte benötigen, root halt höflich bitten – und das tut sie, indem sie dem auszuführenden Befehl ein „su“ voranstellt. Also „su <Befehl>“. Derart geweckt, schaut der SuperUser in seiner Datenbank nach, ob die App denn sowas darf. Beim ersten Aufruf steht sie da noch nicht drin: Die Folge ist ein Popup der SuperUser-App „App xyz möchte etwas mit SuperUser-Rechten machen. Darf sie das?“. Dazu zwei Buttons für „Ja“ und „Nein“, sowie eine „Checkbox“, ob sich *SuperUser* diese Entscheidung für die Zukunft merken soll.



Wikipedia: Sudo

419. <https://towelroot.com/>

420. <http://de.wikipedia.org/wiki/Sudo>

Bei jedem weiteren Aufruf findet der SuperUser die App in seiner Datenbank mit dem Vermerk „die darf das immer“ (sofern der Haken beim ersten Aufruf entsprechend gesetzt war), und führt den Befehl direkt aus. Zur Sicherheit wird dieser Fakt jetzt nochmals als Hinweis eingeblendet (siehe im unteren Teil des Screenshots). Die App wird dabei nicht gebremst, es ist auch keine Interaktion nötig. Daher sollte das in diesem Falle dann sogar vom Lockscreen aus funktionieren. Etwas störend ist das natürlich im Falle einer Screenshot-App, wie das Bild zeigt – da dieser Hinweis dann auf jedem Bild verewigt ist. Deshalb lässt er sich auch in den Einstellungen der *SuperUser*-App abschalten.

Weiterführende Informationen



Ein kleines Tutorial findet sich auch in einem Artikel bei NDroid: [Rooten – Vorteile und Risiken](#)⁴²¹.

N-Droid.DE: Rooten
– Vorteile und
Risiken

421. <http://www.n-droid.de/android-tutorial-rooten-vorteile-und-risiken.html>

Apps am automatischen Starten hindern

Wer kennt das nicht: Man schaltet sein Handy ein, es fährt hoch, und ist eine gefühlte Ewigkeit später auch „betriebsbereit“. Besonders üppig mit RAM ausgestattet sind unsere Androiden ja eher selten – und trotzdem tummeln sich schon zu diesem Zeitpunkt sackweise Apps in selbigem, die ich selten oder gar nie benötige: Flickr, FM-Radio, Google Maps, Peep... Wozu? Und wie kann ich das verhindern?

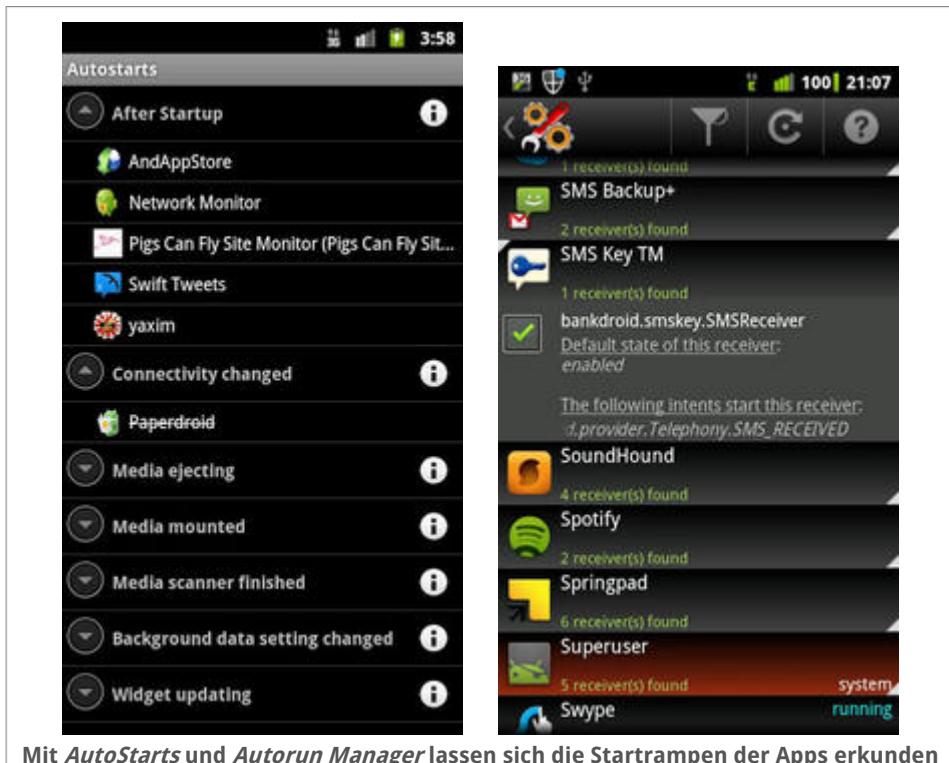
Hier soll es nun nicht um „aggressive Task-Killer“ gehen, die (ausgenommen vielleicht auf einer Ausschluss-Liste stehender Apps) wild alles abschießen, was „peep“ sagt (und nein, auch das Für-und-Wider derselben steht hier nicht zur Debatte). Stattdessen möchte ich Möglichkeiten nennen, gezielt die nicht (ständig) benötigten Apps an einem automatischen Start zu hindern (manchmal auch nachträglich, oops).

Für weitere Kandidaten gleich an dieser Stelle der [Verweis zur zugehörigen Übersicht](#)⁴²².



Übersicht: Auto-Starter unter Kontrolle halten

422. http://android.izzysoft.de/applists/category/named/tools_autostart



Mit *AutoStarts* und *Autorun Manager* lassen sich die Startrampen der Apps erkunden



Autorun Manager



AutoStarts

Ja, es gibt sie: Apps wie *AutoStarts*⁴²³ und *Autorun Manager*⁴²⁴. Sie brauchen in der Regel Root-Rechte, um ihre Tätigkeiten auszuführen. Und sie unterscheiden sich zum Teil stark – sowohl in ihrer Bedienbarkeit, Übersichtlichkeit, Wirksamkeit, als auch in der Art ihrer Vorgehensweise.

Dazu ein wenig Hintergrund-Information: Es ist nicht so, dass es da einen „Startup-Folder“ gäbe. Vielmehr können sich Apps für „Events“ registrieren, bei denen sie gern gestartet werden möchten. Der gewöhnlichste, der jedem sofort einfällt, nennt sich „boot completed“ – unmittelbar, nachdem das System komplett hochgefahren ist. Aber das ist bei weitem nicht alles! Wer einmal mit o. g. *AutoStarts* sein System durchforstet, bekommt beim ersten Mal sicher Kulleraugen, wie viele solcher „Start-Rampen“ es gibt. „USB-Kabel angesteckt“ fällt einem vielleicht noch ein. Aber wer denkt sogleich an Dinge wie „eingehende SMS“, „abgehender Anruf“, „Speicher knapp“? Klar, jetzt fällt einem sicher auch „battery low“ ein...

Je nachdem, welche dieser „Start-Rampen“ unsere App nun also kennt, findet sie mehr oder weniger Kandidaten, die vom automatischen Starten abgehalten

423. <https://play.google.com/store/apps/details?id=com.elsdoerfer.android.autostarts>

424. <https://play.google.com/store/apps/details?id=com.rs.autorun>

werden sollen. *AutoStarts* findet zum Beispiel sehr viele – *Startup Auditor* etwas weniger.

Und wie werden die Apps am Starten gehindert? Die meisten unserer „Verhinderer“ warten einfach auf deren Auto-Start, und schießen die App dann über den Haufen. Anders *AutoStarts*: Hier wird die App quasi gleich von der Rampe genommen – und *AutoStarts* merkt sich App und zugehörige Rampe, um die Aktion ggf. später wieder rückgängig machen zu können. Das ist natürlich weit effektiver (und auch Ressourcen-schonender), birgt aber eine Gefahr: Sollte man *AutoStarts* einmal deinstallieren, ohne zuvor die Änderungen rückgängig gemacht zu haben – dann kann man sie gar nicht mehr rückgängig machen (es sei denn, man hat ein gutes Backup der App-Daten von *AutoStarts* – oder installiert die betroffene App einfach neu). Hat also alles seine Vor- und Nachteile.

Achtung: Wer die Apps nicht direkt „von der Rampe nimmt“, sondern jeweils „nach dem Start abschießen lässt“ (im Falle von Unsicherheit gilt letzteres), sollte anschließend prüfen, was dabei passiert. Bei einigen Apps (z. B. *Peep* oder *Aktien*) passiert es gern, dass sie nach dem „Abschuss“ einfach wieder starten. Das kann dann in einen Kreislauf ausarten, der alles andere als Ressourcen-schonend ist! Es gibt allerdings eine App, die so etwas selbst erkennt: *AutoRun Manager* markiert eine sich so verhaltene App als „Selbst-Restarter“, sobald dieser Fall aufgetreten ist. Damit ist dann klar, dass sich diese nicht auf diese Weise am Starten hindern lässt.

Autorun Manager unterstützt übrigens beide Modi: Im „einfachen Modus“ (kein root erforderlich) schießt er die Apps nach dem Auto-Start einfach über den Haufen. Hier werden auch nur wenige Events berücksichtigt – also wahrscheinlich nicht alle Elemente erwischt. Im „Erweiterten Modus“ (erfordert root) hingegen verhält sie sich wie *AutoStarts*, und „unregistriert“ die jeweilige App vom jeweiligen Event. Hier muss man dann vor einer eventuellen De-Installation daran denken, **vorher** die ursprünglichen „Defaults“ wieder herzustellen (geht allerdings einfach: „Rescue-Mode“, und fertig).

Vorinstallierte Apps entfernen

Das kann echt nervig sein: Was hat mein Provider (bzw. der Telefon-Hersteller) da alles an Apps vorinstalliert, die „kein Mensch“ braucht? Und wie werd ich „den Schrott“ los? Jetzt kommt das böse Wort: „Ohne root? Gar nicht.“ Da wären wir also wieder. Zum Glück lassen sich ab Android 4.0 die meisten dieser Apps zumindest auch ohne root „deaktivieren“, indem man den gleichnamigen Button in den Settings der App unter *Einstellungen > Apps verwalten* betätigt.



Titanium Backup

Und mit root? Ja, da gibt es Möglichkeiten. Die bekannteste dürfte wohl Titanium Backup⁴²⁵ sein: Mit dieser App lässt sich jede ungewünschte App komplett vom System entfernen, wenn es denn sein soll. Wem das zu heikel ist, der hat auch eine Alternative: Einfrieren (was Android ab Version 4.0 auch von Haus aus anbietet). Damit taucht die App in keiner Liste (außer hier bei Titanium Backup) mehr auf, wird nicht mehr (automatisch) gestartet – und kann dennoch jederzeit wieder „aufgetaut“ werden.



Nebeneffekt der App – der Name lässt es erahnen: Man kann damit vollständige Backups machen. Von einzelnen Apps. Von deren Daten. Vom ganzen System. Und natürlich bei Bedarf Daten, Apps und System aus einem Backup zurückholen. Klasse Sache bei einem Geräte- oder ROM-Wechsel (aufpassen: Unterschiedliche Geräte/ROMs = unterschiedliche Systemdateien; hier nur die Apps und ggf. deren Daten wieder herstellen, und die System-Dinge nicht anfassen – System-Apps besser ebenso wenig).



SuperBox

Wer da noch einfrieren und auftauen kann, und noch einiges mehr, ist SuperBox⁴²⁶, das sich selbst auch als „Schweizer Taschenmesser“ bezeichnet. Definitiv auch einen Blick wert. Kann nämlich auch den Cache aufräumen, und solche Dinge – die im nächsten Kapitel zur Sprache kommen...

425. <https://play.google.com/store/apps/details?id=com.keramidas.TitaniumBackup>

426. <https://play.google.com/store/apps/details?id=net.lepeng.superboxss>

Tuning – Das Android-System auf Trab bringen

Ein altbekannter Effekt: mit der Zeit wird unser Androide immer träger. Oder kommt uns das nur so vor? Doch spätestens dann, wenn jemand seinen „guten alten“ Freund aus der ersten Generation neben einen Kameraden aus der aktuellen Mittel- oder gar Oberliga legt, liegt die Antwort auf der Hand: Ja klar, der neue ist definitiv schneller. Diesen großen Schritt werden wir mit unseren „alten kleinen Gurken“ kaum vollständig bewältigen – aber wir können die Lücke kleiner machen. Und darum soll es hier gehen, in mehreren Schritten. Nicht jeder wird alle Schritte durchführen können/wollen (ich sage nur, da kommt Herr Root ins Spiel) – dennoch ist sicher für die meisten etwas passendes dabei. Selbst wenn das Gerät bereits neuer und schneller ist: Ein wenig Finetuning kann ja nicht schaden, gelle?

Schnellwaschgang

Bevor ich ein wenig in die Details einsteige, hier ein paar Schnelltipps für Ungeduldige – sozusagen das Wichtigste auf einen Blick (eine Übersicht über Verbrauchsdaten gibt es im [Anhang](#)):

Akku-Laufzeit verlängern

- WLAN komplett abschalten, wenn es nicht gebraucht wird (ab Android 4.3 ggf. zusätzlich die „Netzwerk-suche bei abgeschaltetem WLAN“, [zu finden in den erweiterten WLAN-Einstellungen](#)⁴²⁷). Das spart enorm, und lässt sich auf Wunsch auch [automatisieren](#). Nebenwirkung: Schutz vor gewissen [schnüffelnden Mülltonnen](#)⁴²⁸.
- Wer kein mobiles High-Speed benötigt: 3G aus, das frisst auch ganz nett (siehe auch [2G versus 3G: Spart 2G wirklich so viel Akku?](#))
- Dito ggf. für GPS (spart aber nur wenig, da es im Standby so gut wie nix frisst)
- Helligkeit des Displays weitmöglichst herunterregeln. „Aufdrehen“ dann im Bedarfsfall.
- Live-Wallpaper und sonstige „animierte Dauerrenner“: Wer drauf verzichten kann, sollte das tun!
- [root und Modder: CPU nicht über-, sondern ggf. eher untertakten](#). Ich weiß, das klingt nicht besonders „cool“ – spart aber Akku.



AndroidPIT: So stoppt Ihr die permanente WLAN-Suche von Android 4.3



Slashdot: Londoners tracked by advertising firms trash cans

427. <http://www.androidpit.de/wlan-android-4-3>

428. <http://news.slashdot.org/story/13/08/11/1749238/>

- Apps, die nicht benötigt werden, deinstallieren. Was nicht da ist, macht keinen Stress.
- Apps, die man *nur gelegentlich* braucht, müssen *nicht ständig* im Hintergrund laufen. Tun sie das doch, verbietet man ihnen ggf. das Automatische Starten.
- Die Datensynchronisierung muss evtl. auch nicht ständig laufen. Bei vielen Apps, die eine solche benötigen (z. B. RSS-Reader, Mail-Apps) lassen sich die Intervalle entsprechend anpassen – oder gar den Abgleich auf Zeiträume mit WLAN-Empfang beschränken (das spart nebenbei gleich noch Datenvolumen)

Mehr Speed

- RAM Bereinigen (Android-interne Einstellungen optimieren)
- Mit der Zeit (und besonders bei starker Nutzung) wird u. U. das Dateisystem langsamer, weil der Controller länger nach freien Datenblöcken suchen muss. In diesem Fall kann Lagfix (fstrim)⁴²⁹ Abhilfe schaffen (ab Android 4.3 obsolet, da fstrim in das System integriert wurde).
- Nur für bestimmte Situationen, wo es darauf ankommt: CPU ggf. leicht übertakten. Frisst aber auch mehr Akku.
- Apps, die nicht benötigt werden, deinstallieren. Was nicht da ist, macht keinen Stress.
- Die Datensynchronisierung muss evtl. auch nicht ständig laufen. Bei vielen Apps, die eine solche benötigen (z. B. RSS-Reader, Mail-Apps) lassen sich die Intervalle entsprechend anpassen – oder gar den Abgleich auf Zeiträume mit WLAN-Empfang beschränken (das spart nebenbei gleich noch Datenvolumen)



Lagfix (fstrim)

Mehr Platz im internen Speicher schaffen

- Apps, die nicht benötigt werden, deinstallieren. Was nicht da ist, frisst kein Brot.
- Apps aus dem internen Speicher auf die SD-Karte auslagern
- Cache bereinigen

429. <https://play.google.com/store/apps/details?id=com.grilledmonkey.lagfix>

Apps auslagern

Normalerweise werden Apps im Telefonspeicher installiert – und der ist nicht immer unbedingt üppig ausgestattet. Ein paar „größere Knaller“ installiert, und voll ist er: Vor der Installation der nächsten App muss man sich also entscheiden, auf welche bereits installierte App man hier verzichten kann ...



Mit Froyo (Android 2.2) und neuer, lässt sich bereits von Haus aus App2SD nutzen – größere bzw. seltener benutzte Apps können so auf die SD-Karte ausgelagert werden. Damit hat man wieder freien internen Speicher – u. a. auch für die Apps, die kein *App2SD* unterstützen. Fertig – der Rest dieses Kapitels kann nun übersprungen werden!

Hm, noch hier? Also noch Android < 2.2 am Laufen? Oder Android ≥ 4.x, und der Hersteller hat die Unterstützung für App2SD entfernt? Oder einige Apps am Werkeln, die sich partout nicht verschieben lassen wollen? Kein root? Naja, dann hat das Folgende eher informativen Charakter. Um die hier beschriebenen Möglichkeiten zu nutzen, muss der Androide zuvor gerootet werden.

Die App Link2SD⁴³⁰ ermöglicht das Auslagern von Apps auf die SD-Karte auch unter „älteren“ Android-Versionen. Und zwar sogar „besser“ als App2SD: Es wird eine separate Partition auf der Karte angelegt (äh, falsch: *der Anwender* muss diese Partition im ersten Schritt anlegen). Auf diese verschiebt man dann die gewünschten Apps, und am ursprünglichen Speicherort werden sogenannte „symbolische Links“ auf die neue Location angelegt. Schließt man nun den Androiden per USB-Kabel an den PC an, wird nur die andere Partition dorthin exportiert – und die ausgelagerten Apps bleiben weiterhin lokal verfügbar. Das ist bei App2SD nicht der Fall.



Link2SD

430. <https://play.google.com/store/apps/details?id=com.buak.Link2SD>



Was hierfür benötigt wird, und welche Schritte nötig sind, beschreibt u. a. ein [Tutorial bei DroidWiki⁴³¹](#) ausführlich.

DroidWiki: Link2SD einrichten

Cache bereinigen

Und weiter geht's mit der Bereinigung des internen Speichers. **Dieser Schritt benötigt kein root** – kann also von jedem genutzt werden.

Cache ist schon eine feine Sache, und soll so manche Dinge ja schneller machen. Zum Beispiel, indem man Daten aus dem Internet im lokalen Dateisystem ablegt, damit man sie nicht jedes Mal neu laden muss, wenn sie wieder gebraucht werden. Der letzte Teilsatz ist aber genau der Knackpunkt: Braucht man sie überhaupt noch? Und was, wenn dummerweise mal gerade eine „kaputte Variante“ im Cache landet? Oder der Platz im internen Speicher knapp wird?

Kurz: Wird der Cache gelöscht, geht dabei nix kaputt. Die Daten müssen im schlimmsten Fall neu (z. B. aus dem Internet) geladen, oder neu generiert werden. Dafür ist aber anschließend aufgeräumt – und so manches Problem nebenbei mit behoben.

Und wie erledigt man das? Mit Bordmitteln: In den Einstellungen in die Anwendungs-Verwaltung gehen, jeden Eintrag einzeln durchforsten (ob die App überhaupt Cache nutzt), und für jede betroffene App den Cache von Hand löschen. Da geht schon mal gern ein Stündchen bei drauf – und bequem ist es auch nicht gerade. Zum Glück gibt es zahlreiche Alternativen, wie z. B. [Quick App Manager⁴³²](#), [Simple Cache Cleaner⁴³³](#), und [weitere⁴³⁴](#). Eine andere schnelle Möglichkeit, die keiner Extra-App bedarf: In das [Recovery-Menü](#) booten, und „Wipe Cache“ auswählen. Klappt bei jedem Gerät.



Quick App Manager



Simple Cache Cleaner



Übersicht: System-Bereinigung

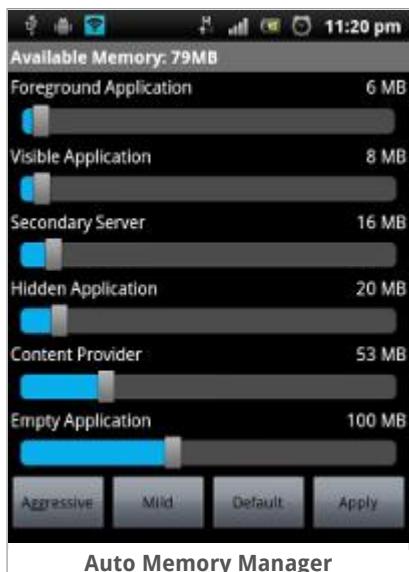
431. http://www.droidwiki.de/Link2SD_einrichten

432. <https://play.google.com/store/apps/details?id=com.cerisierbleu.qac>

433. <https://play.google.com/store/apps/details?id=jp.co.aplio.simplecacheconverter.free>

434. http://android.izysoft.de/applists/category/named/tools_systemcleaner

RAM bereinigen



Nachdem wir uns nun um den internen Speicher gekümmert haben, geht's ans Eingemachte: Das RAM ist fällig!

Wer aber nun an Task-Killer denkt – voll daneben. Kontrovers diskutiert, verteufelt, hochgejubelt ... Aus meiner Sicht ist die Aufgabe eines Task-Killers nicht, freien Speicherplatz im RAM zu schaffen (das ist eher ein Nebeneffekt) – sondern vielmehr, „hängende“ Apps zu beenden, die andernfalls z. B. Amok laufen, oder das System lahmlegen (z. B. mit hoher CPU-Last). Details finden sich weiter oben unter [Von Taskkillern und anderen bösen Buben](#).

Nächstes Argument: „Android kümmert sich doch selbst um die Speicherbereinigung“. Jetzt kommen wir der Sache näher: Ja, das stimmt – die Frage ist nur: Wann? Wie oft? Und wie? Darum geht es in erster Linie: Die zugehörigen Einstellungen für diesen sogenannten „OOM-Killer“ (**O**ut **O**f **M**emory **K**iller) anzupassen. Ihm beizubringen, wie viel freien Speicher wir benötigen – und wann er zuschlagen darf.

Auch hierfür gibt es wieder zahlreiche Helferlein. Da es sich um System-Einstellungen „unter der Haube“ handelt, wird hier i. d. R. [root](#) benötigt. Eine rühmliche Ausnahme ist da [Auto Memory Manager](#)⁴³⁵ (Bild links), der es angeblich auch ohne schafft. Selbst getestet habe ich den [AutoKiller Memory Optimizer](#)⁴³⁶, der recht gut funktioniert.



Auto Memory Manager



AutoKiller Memory Optimizer

Unnütze Apps raus!

Das sollte eigentlich sowas von klar sein: Apps, die man überhaupt nicht mehr braucht, belegen unnütz Speicher – und können auch bei Nicht-Benutzung das System (leicht) ausbremsen. Also runter damit! Die notwendigen Details hierzu finden sich bereits weiter oben unter [Apps verwalten](#). Und was man nicht

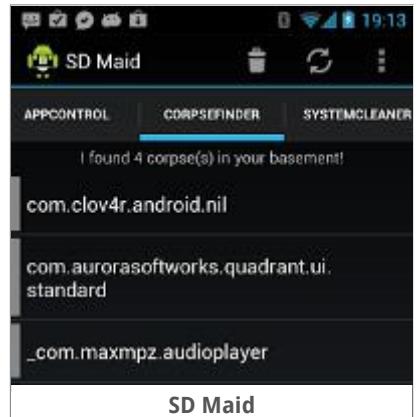
435. <https://play.google.com/store/apps/details?id=com.lim.android.automemman>

436. <https://play.google.com/store/apps/details?id=com.rs.autokiller>

deinstallieren kann/möchte, kann man ja zumindest noch [am automatischen Starten hindern](#).

Noch Leichen im Keller? Im Laufe seines Androiden-Lebens hat unser Gerät so manche App „einmal kurz gesehen“, die dann wieder entfernt wurde – etwa weil wir etwas besseres gefunden haben, oder die App aus anderen Gründen nicht mehr benötigen. Nicht jede dieser Apps wurde restlos entfernt – manch eine App ließ noch ein paar „Leichen“ zurück. Und dann wären da noch diverse „core dumps“ von „Force Closes“ („Schließen erzwingen“), System-Logs, und mehr.

Eine App verspricht, sich genau darum zu kümmern: [SD Maid](#)⁴³⁷. Was diese App tut: Sie durchsucht die typischen Verzeichnisse (/data/data und /mnt/sdcard/Android/data bzw. bei Samsung-Geräten /dbdata/databases für erstere Location) und vergleicht die dortigen Einträge mit der Liste tatsächlich installierter Apps. Auf diese Weise wird überflüssiges identifiziert, und kann entweder einzeln (Eintrag antippen) oder gleich in einem Rutsch („Clean All“) entfernt werden. Im Reiter „Clean System“ lassen sich weitere Plätze und Dinge bereinigen – etwa System-Logs, Core-Dumps oder – huch? – auch Cache. *SD Maid* kann zwar ohne root verwendet werden, bietet dann jedoch nicht den vollen Funktionsumfang.



SD Maid

CPU Taktung anpassen

Wiedermal nur für gerootete Geräte.

437. <https://play.google.com/store/apps/details?id=eu.thedarken.sdm>

Hier gibt es generell eigentlich zwei Richtungen:

- Höhere Taktung › Schnellere Reaktion & Arbeitsgeschwindigkeit – aber auch der Akku wird so schneller leer
- Niedrigere Taktung › Langsamere Reaktion & Arbeitsgeschwindigkeit – aber der Akku hält (teilweise deutlich) länger durch

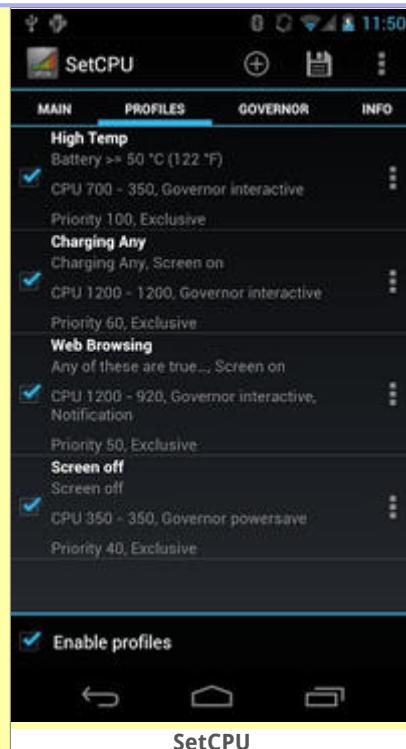
Eigentlich klar soweit – auch wenn die meisten beim Thema „CPU Taktung“ nur zuerst ans Übertakten denken, auch Untertakten kann seine Vorteile haben. Beim Übertakten sollte man überdies vorsichtig sein, dass man aus seinem Androiden nicht zuerst eine Warmhalteplatte, dann eine Kaffeemaschine, und schließlich einen Ziegelstein macht – nicht übertreiben!

Sinnvolle Szenarien beinhalten u. a.:

- Wenn das Gerät nicht benutzt wird (Bildschirm aus, Nachtschlafene Zeit, ...) › runter mit dem Takt
- Wenn das Gerät benutzt wird (Bildschirm an) › Auf „normalen“ Takt schalten (was immer das für den einzelnen ist)
- Wenn eine bestimmte (hungrige) App im Vordergrund läuft › Hoch mit dem Takt (Vorsicht – nicht zu hoch)

Viele der CPU-Taktveränderer decken die ersten beiden Punkte ab. Der dritte Punkt ließe sich z. B. mit Tasker⁴³⁸ sicher realisieren.

Wiederum gibt es etliche Apps, die beim Drehen an der CPU-Schraube behilflich sein wollen. Die bekannteste ist sicherlich SetCPU⁴³⁹ (rechtes Bild), der bekannteste Mitbewerber dazu heißt CPU tuner⁴⁴⁰. Weitere Kandidaten sind in



SetCPU



CPU tuner

438. <https://play.google.com/store/apps/details?id=net.dinglisch.android.taskerm>

439. <https://play.google.com/store/apps/details?id=com.mhuang.overclocking>

440. <https://play.google.com/store/apps/details?id=ch.amana.android.cputuner>

der passenden [Übersicht](#)⁴⁴¹ benannt, oder finden sich auf den jeweiligen Seiten der Apps als Alternativen aufgeführt.



Übersicht: CPU
Einstellungen

Durststrecke – mehr aus dem Akku herausholen

Wenn der Akku mit einer Ladung länger durchhalten soll, müssen wir den Energieverbrauch senken. Soweit klar. Also stellen sich die Fragen:

- Was verbraucht Energie?
- Was ist dafür verantwortlich?
- Wie können wir dem beikommen?

Schauen wir uns diese Fragen also einmal nacheinander an.

Was verbraucht Energie?

Gleich von vorn herein auf den Aspekt des „können wir dem beikommen“ betrachtet, lassen sich die Verbraucher schnell auf wenige Punkte reduzieren:

- CPU
- Speicherzugriffe
- Netzwerkzugriffe
- „Peripherie-Geräte“ (hier: Kamera, Display)

Dafür verantwortlich sind die Apps, welche die Ressourcen beanspruchen – und natürlich nicht zuletzt der Benutzer, der einen gewissen Einfluss auf diese Apps hat. So ergeben sich einige Lösungsansätze schon von selbst:

Wie können wir dem beikommen?

Zum größten Teil sind diese Punkte bereits in den vorigen Kapiteln behandelt worden. Die entsprechenden Kandidaten seien daher hier nur noch einmal kurz aufgezählt:

- Apps, die nicht benötigt werden: [Weg damit!](#) Was nicht da ist, kann auch nichts verbrauchen.
- Was man nur selten braucht, muss nicht ständig laufen – diese Apps sollte man also am [automatischen Starten hindern!](#) Das gilt auch für auf den Homescreens installierte Widgets: Hier handelt es sich ja ebenfalls um laufende Apps, die automatisch gestartet werden. Daher sollte man diese möglichst sparsam einsetzen.

441. http://android.izzysoft.de/applists/category/named/tools_settings#group_539

- Wenn im RAM genügend freier Platz existiert (also der „[OOM-Killer“ gut konfiguriert ist\), wird dieser als Cache genutzt – und dafür weniger auf die Speichermedien zugegriffen.](#)
- Exzessive Nutzung von [Task-Killern](#) verbraucht zusätzliche Energie – da viele Apps gleich nach dem Kill vom System wieder nachgeladen werden
- Ist der Androide [gerootet](#), lässt sich [die CPU untertakten](#) (zumindest dann, wenn man sein Gerät ohnehin nicht benutzt).
- Das Display muss auch nicht unbedingt als Scheinwerfer herhalten – die Helligkeit lässt sich entsprechend anpassen.

Die meisten Dinge haben wir bereits besprochen. Ein paar jedoch noch nicht:

Speicherzugriffe: Ja, klar: mehr Cache mit sauberem Speicher – stand doch da schon. Richtig, aber das ist noch nicht alles. Die Frage ist auch: Was für eine SD-Karte steckt in meinem Gerät? Im Allgemeinen verbrauchen „ältere Karten“ (also die langsameren Teile – erkennbar u. a. an ihrer kleineren „Klasse“) mehr Strom als neuere, da Lese- und Schreibzugriffe bei diesen länger brauchen. Eine neue Karte verschafft uns also u. U. nicht nur mehr Speicher, sondern ist auch schneller – und der Akku hält länger. Steht also ohnehin ein Neukauf an, sollte man nicht zu knauserig sein: Lieber einen Euro mehr ausgeben, und was vernünftiges holen! „Class-6“ sollte es mindestens sein.

Netzwerk-Zugriffe: Auch diese kosten Energie – manche mehr, manche weniger. 3G (UMTS) knabbert i. d. R. stärker am Akku als 2G (GPRS/EDGE). Wer also nicht unbedingt auf „HighSpeed“ an dieser Stelle angewiesen ist, schaltet 3G einfach komplett ab (schaut aber sicherheitshalber zuvor noch unter [2G versus 3G](#) nach). Die passenden Schalterchen finden sich in *Einstellungen > Drahtlos und Netzwerke > Mobile Netzwerke* (Netzwerkmodus: „Nur GSM“, bzw. „Nur 2G“). Weitere Kandidaten sind Bluetooth und WLAN, die man bei Nichtgebrauch auch deaktivieren kann (siehe dazu unter [Helferlein](#)).

In die Rubrik *Netzwerk-Zugriffe* gehört übrigens auch der **mobile Datenabgleich** – hier kann man ebenfalls einschränken: Wetterinformationen müssen nicht alle fünf Minuten aktualisiert werden (hier genügen in der Regel 3-6 Stunden als Intervall). Bei RSS-Feeds kommt es auf die Ansprüche an; normalerweise ist ein stündlicher Abgleich aber ausreichend. Bei vielen RSS-Readern lässt sich das sogar noch auf die Zeiträume beschränken, in denen eine WLAN-Verbindung verfügbar ist; das spart dann nebenbei auch noch Datenvolumen. Bei Mails empfiehlt sich ein Anbieter (und eine App), die das „IMAP Idle“ Protokoll unterstützen: Dann stößt der Server nämlich den Abgleich an, wenn neue Mail eintrifft – und man muss nicht alle paar Minuten unnötig nachschauen lassen („Pollen“). Letzteres gilt jedoch nur, wenn man wirklich

„ständig auf dem Laufenden“ sein muss; wem eine Aktualisierung alle 30 (oder mehr) Minuten genügt, für den ist „Polling“ sparsamer.

Kamera: Natürlich braucht auch diese „Saft“, und nicht unbedingt wenig. Das betrifft nicht nur das Fotografieren, sondern auch die Barcode-Scanner. Von letzteren sind einige dafür bekannt, dass sie auch nach dem Beenden noch fleißig weiter nuckeln (was an einem Bug in der betreffenden Google-API zu liegen scheint). Hier könnte also ausnahmsweise ein Task-Killer-Einsatz gerechtfertigt sein. Oder aber die Wahl eines alternativen Barcode-Scanners.

Zu letzterem Fall gibt es sicher noch Parallelen, wo eine alternative App sparsamer wäre als die gerade eingesetzte. Um solche Kandidaten aufzuspüren, kann man z. B. die Statistiken unter *Einstellungen* > *Telefoninfo* > *Akku* > *Akkerverbrauch* nutzen, die besonders hungrige Apps auflistet. Aber aufpassen: Die Statistik zählt seit dem letzten Boot – lief eine hungrige App also nur mal eben ein Minütchen, fällt sie hier (noch) nicht so ins Gewicht. Mehr Details liefert dann, je nach eingesetzter Android-Version, auch noch ein „Anruf“ bei der magischen Nummer *#*#4636#*#*...

Helperlein

Nicht verschweigen möchte ich hier einige Helperlein, die dem Nutzer beim Energiesparen „unter die Arme greifen“ können. Grob gesehen wären dies zwei Kategorien von Apps: Schnellumschalter (finden sich in dieser Übersicht⁴⁴²), und im Hintergrund laufende, auf „Events“ reagierende Apps.

Schnellumschalter sind meist Widgets auf dem Home-Screen, mit denen man „toggeln“ (also zwischen zwei Zuständen wechseln) kann. z. B. „WLAN an/aus“, „Bluetooth an/aus“, etc. Einige dieser Widgets stehen schon „von Haus aus“ zur Verfügung – aber damit kann man sich den Home-Screen schnell „zupfastern“, wenn mehrere benötigt werden. Daher gibt es auch Apps, die mehrere Schalterchen sinnvoll vereinigen, und so platzsparend mehr anbieten können. Als Beispiel sei hier Extended Controls⁴⁴³ genannt. Spätestens ab Android 4.0 (und auf einigen Geräten auch bereits früher) gibt es von Haus aus auch die „Quick-Settings“ in der Benachrichtigungsleiste.



Übersicht: Toogles, Switches, Schnellumschalter



Extended Controls

442. http://android.izzysoft.de/applists/category/named/tools_switches

443. <https://play.google.com/store/apps/details?id=com.extendedcontrols>



Hintergrund-Aufpasser beeinflussen den Energieverbrauch „ereignisgesteuert“, wobei der Anwender die Ereignisse meist selbst festlegt. Zum Beispiel: Wenn weniger als 20% Akkuleistung verbleibt, schalte Bluetooth und WLAN aus. Die bekannteste App in dieser Kategorie nennt sich [JuiceDefender](#)⁴⁴⁴, wird jedoch scheinbar nicht mehr gepflegt; ein vergleichbarer Kandidat wäre [Green Power](#)⁴⁴⁵.

Ein Hintergrund-Aufpasser der besonderen Art ist [Greenify](#)⁴⁴⁶, welches sich um „unartige Apps“ kümmert. Damit meine ich Apps, die nach ihrer Benutzung und vermeintlichen Beendigung im Hintergrund fleißig weiter Ressourcen verbrauchen (ein gutes Beispiel dafür wäre etwa *Google Maps*). *Greenify* kümmert sich um vom Anwender festgelegte Kandidaten, und befördert sie nach ihrer Nutzung automatisch in den Ruhezustand – quasi ein „Einfrieren On-the-Fly“. Wer also derartige Bösewichter auf seinem Androiden hat, der kann mit *Greenify* noch einiges an zusätzlicher Akku-Laufzeit herausholen.



JuiceDefender



Green Power



Greenify

444. <https://play.google.com/store/apps/details?id=com.latedroid.juicedefender>

445. <https://play.google.com/store/apps/details?id=org.gpo.greenpower2>

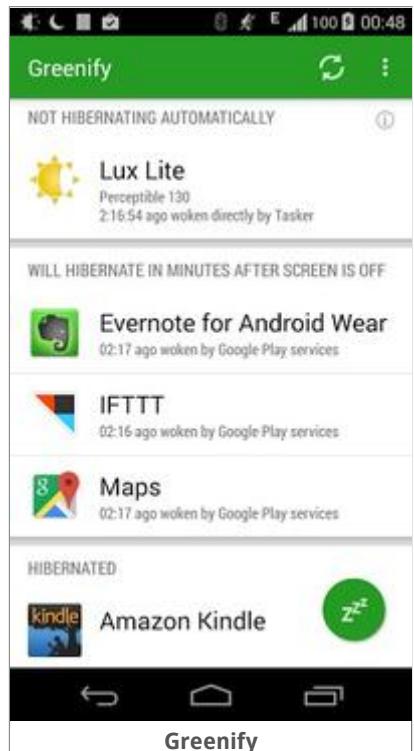
446. <https://play.google.com/store/apps/details?id=com.oasisfeng.greenify>

Wie ein einfacher Anwender passende Kandidaten ausfindig machen soll? Natürlich mit *Greenify* selbst. Die App greift ihm dabei nämlich durchaus hilfreich unter die Arme, wie der Screenshot zeigt. So werden zunächst bereits im Hintergrund laufende Apps aufgeführt, zusammen mit der von ihnen bereits verbrauchten Systemzeit. Sieht man hier auf den ersten Blick einen „Ausreißer“ mit extrem hohen Werten, wäre das sicherlich ein guter Kandidat. Diesem Segment folgen weitere: Etwa mit Apps, die das System bei bestimmten Ereignissen ausbremsen (weil sie zu diesen automatisch gestartet werden wollen). Zu guter Letzt lassen sich schließlich auch alle installierten Apps durchforsten.

Drei kleine „Handicaps“ gilt es zu beachten: Erstens Einen bedarf es zur Behandlung von System-Apps der Donation-Version. Dies betrifft z. B. mein Parade-Beispiel *Google Apps*, sofern es bereits auf dem Gerät vorinstalliert war. Zweitens sollte man mit Bedacht an die Sache herangehen: Greenified man etwa eine Nachrichten-App wie *WhatsApp*, kann diese im „Ruhezustand“ auch nicht mehr auf eingehende Nachrichten reagieren – es sei denn, man setzt das „Donation Package“ ein. Drittens lässt sich *Greenify* zwar generell auch ohne root einsetzen – ihr volles Potential spielt die App allerdings erst mit root aus.

Weiterführende Informationen finden sich beispielsweise in einem [Video bei XDA Developer TV](#)⁴⁴⁷, welches jedoch in englischer Sprache gehalten ist, sowie in einem ausführlichen [Artikel des AndroidUser](#)⁴⁴⁸ auf Deutsch.

Nicht vergessen werden sollten an dieser Stelle auch spezielle Helferlein wie [APNandroid](#)⁴⁴⁹ (mit dem man sein Telefon – temporär – daran hindern kann,



AndroidUser: Mit Greenify die Akku-Laufzeit verlängern



APNandroid

447. <http://www.xda-developers.com/android/android-app-review-save-your-battery-with-greenify-xda-developer-tv/>

448. <http://www.android-user.de/Magazin/Archiv/2013/09/Mit-Greenify-die-Akku-Laufzeit-Ihres-Androiden-verlaengern>

449. <https://play.google.com/store/apps/details?id=com.codecarpet.apnandroid.pro>

Netzwerk-Verbindungen aufzubauen) oder [3G Watchdog](#)⁴⁵⁰ (passt nicht so ganz hier, da es eher über das Datenvolumen wacht – aber es nutzt *APNandroid*, wenn man möchte, um bei einem bestimmten Volumen „abzuschalten“).

Sicher gibt es noch etliche mehr – und ich gebe auch frei heraus zu, nicht alle zu kennen. Aber ich hoffe, hier zumindest einen Einblick in die Möglichkeiten gegeben zu haben.



3G Watchdog

Wer saugt da meinen Akku leer?



Soso, irgendwer saugt da also plötzlich den Akku leer – und der Anwender hat keinen blassen Schimmer, wer oder was das sein könnte? Da gibt es zunächst ein paar Bordmittel für die Analyse. Zum Beispiel die Übersicht der „größten Verbraucher“, die sich unter *Menü > Telefoninfo > Akkuverbrauch* finden. Und auch ein paar mehr Details, wenn man mal bei *#*#4636#*#* „anruft“.

Allerdings bieten beide Stellen nur einen groben Überblick über längere Zeiträume. Ein „Dauerläufer“ sollte so zwar aufspürbar sein – doch manchmal hätten wir gern ein paar mehr Details zur Hand. Aber auch kein Thema: Einen passenden Kandidaten habe ich ja bereits unter [System-Info](#) benannt (dort auch die aktuelleren Screenshots), die [SystemPanel App](#)⁴⁵¹.

Für die detaillierte Analyse muss es allerdings in der Tat die Kaufversion sein – denn nur diese bietet das Monitoring im Hintergrund. Sobald sie für eine Weile Daten gesammelt hat, können diese dann ausgewertet werden. Wie das in etwa aussieht, zeigt der Screenshot am Beispiel eines bereits beendeten Google-Maps:

Der obere Graph lässt erkennen, dass diese App etwa von 18 Uhr bis 20 Uhr lief. Ein Ladekabel war offensichtlich nicht angeschlossen (kein grüner Balken bei „Charging“), dafür wurde der Akku aber permanent entladen (dritter Graph von



SystemPanel

450. <https://play.google.com/store/apps/details?id=net.rgruet.android.g3watchdog>

451. <https://play.google.com/store/apps/details?id=nextapp.systempanel.r1>



Übersicht: Akku-Helfer



AndroidUser:
PowerTutor zeigt,
wer den Akku
leeraugt

oben). Auch die CPU war derweil gut beschäftigt (unterster Graph), obwohl das Gerät nicht selbst durchgehend aktiv genutzt wurde (blauer Graph). Während dieser zwei Stunden leerte sich der Akku von knapp 100% auf gut 50% – oh ja, das hat gut geschluckt! Und der Schuldige am „plötzlich leeren Akku“ ist mit ziemlicher Sicherheit identifiziert.

Wie jetzt: Alternativen? Na gut, auch hier gibt es natürlich eine passende [Übersicht](#)⁴⁵². Oder einen Artikel im AndroidUser, der zeigt, wie man Akku-Fresser [mit PowerTutor aufspürt](#)⁴⁵³.

2G versus 3G: Spart 2G wirklich so viel Akku?

Eine Empfehlung, die man häufig zu hören und zu lesen bekommt: 2G geht viel sparsamer mit dem Akku um als 3G (oder gar 4G). Schaut man sich die nackten Zahlen an (siehe [Leistungsaufnahme verschiedener Komponenten](#)), scheint das auch zu stimmen. Aber ist das in der Praxis auch wirklich der Fall?

Vor dem Hintergrund dieser Frage habe ich den Selbstversuch gemacht. Als wenig telefonierender Wenigsurfer, hatte ich bislang alle meine Geräte auf 2G/GSM fixiert. Zum Test habe ich diese Sperre für ein paar Tage deaktiviert, und mein *LG Optimus 4X* per 3G (UMTS) ins Netz gelassen. Eigentlich hätte ich nun einen sich schneller leerenden Akku erwartet – aber das Ergebnis war ein anderes: Die minimalen Unterschiede, welche ich über die Tage feststellen konnte, lassen sich durchaus als „Mess-Fehler“ interpretieren. Schließlich hatte ich nicht gerade Laborbedingungen. Wie aber ist das zu erklären?

Man darf nicht ausschließlich den „Verbrauch pro Zeiteinheit der Aktivität“ betrachten. Es muss auch darauf geachtet werden, wie lange diese Aktivität ausgeführt wird! Genau das will ich an einigen praktischen Beispielen im Folgenden tun. Dabei habe ich aus den im Anhang genannten Daten Mittelwerte gebildet. Gleichzeitig habe ich 2G/GSM ein wenig „schön gerechnet“ (also recht optimale Bedingungen zugrunde gelegt), während ich 3G/UMTS etwas pessimistischer betrachtete (weniger optimale Bedingungen, geringere Bandbreite). Dies sollte m. E. der Praxis recht nahe kommen, da nicht jeder Anbieter auch die neueste Technologie voll ausreizt.

452. http://android.izzysoft.de/applists/category/named/tools_batteryhelper

453. <http://www.android-user.de/?p=8907>

Der wenig telefonierende Wenigsurfer

Hier habe ich ja bereits im Selbstversuch (siehe oben) festgestellt, dass sich nichts geändert zu haben scheint. Legen wir einmal 150–300 Minuten Telefonie und 50 MB Daten monatlich zugrunde, so kommen wir auf ca. 5–10 Minuten Telefongespräche sowie 1,5 MB Daten pro Tag. Beides Werte, die nahezu vernachlässigbar sind; es verbleibt daher der „Standby Verbrauch“. Und der beträgt bei UMTS ca. 15 mW, bei GSM ca. 10 mW. Bedenkt man, dass das Display durchschnittlich mit 700 mW zu Buche schlägt, dürfen die 5 mW Differenz getrost ignoriert werden: Für diese Anwendergruppe spielt es also keine Rolle, ob die Datenverbindung auf 2G fixiert wird oder nicht.

Der Vieltelefonierer mit minimalem Datenverbrauch

In diesem Fall schaut die Sache ganz anders aus: Für Telefonate benötigt UMTS ca. 800 mW, während GSM mit gerade einmal der Hälfte (also ca. 400 mW) auskommt. Die Länge eines Telefonats verändert sich dabei nicht (man spricht mit UMTS ja nicht schneller) – der Verbrauch ist also mit UMTS in diesem Fall tatsächlich doppelt so hoch. Diese Anwender-Gruppe fährt also in der Tat mit 2G/ GSM besser bzw. Akku-sparender.

Der wenig telefonierende Dauersurfer

Für diesen darf der Telefonie-Anteil wieder vernachlässigt werden, wie der erste Fall bereits zeigte: Im Standby sind die Unterschiede schließlich marginal. Bleibt der Datendurchsatz. Die „nackten Zahlen“ zeigen auch hier nicht viel Differenz: Etwa 1.200 mW bei UMTS stehen etwa 1.000 mW bei 2G gegenüber. Schaut also gar nicht so wesentlich aus. Aber haben wir da eventuell etwas vergessen?

Werfen wir einmal einen Blick auf die Übertragungs-Geschwindigkeiten: UMTS schafft mit HSDPA/HSUPA einen Durchsatz von ca. 7 MBit für Downloads sowie 1 MBit für Uploads. 2G kommt mit EDGE lediglich auf ca. 300 kBit beim Download sowie 100 kBit beim Upload. Der Download einer Datenmenge von 1 MB benötigt somit ca. 1 Sekunde mit UMTS – jedoch ca. 30 Sekunden mit EDGE. Gemessen an der übertragenen Datenmenge, ist der Stromverbrauch also bei EDGE fast um den Faktor 30 größer!

Ergebnis der Berechnung: Diese Anwender-Gruppe sollte besser auf 3G/UMTS setzen.

Der viel telefonierende Dauersurfer

Für diesen hängt es maßgeblich davon ab, was er unter „viel Telefonieren“ und „Dauersurfen“ versteht. Je nachdem, was von beidem überwiegt, muss er sich in eine der drei obigen Gruppen einordnen; eine generelle Aussage lässt sich hier nicht treffen.

Der „Grashüpfer“

Ein spezieller Fall, besonders in ländlichen Gegenden, in denen die Netzabdeckung ein wenig „wackelig“ ist: Muss das Gerät ständig zwischen den Netzen hin und her hüpfen, geht das ziemlich auf den Akku. GSM und UMTS verwenden verschiedene Frequenzbänder. Wird das Signal daher schwach, sucht das Gerät auch im jeweils anderen Band nach „besseren Konditionen“. Ist man von diesem Missstand betroffen, sollte man sich ggf. ebenfalls für entweder 2G oder 3G entscheiden.

Fehlt da noch etwas?

Sofern jemand in obigen Ausführungen 4G/LTE vermisst hat, dies hat gute Gründe: Zum Einen habe ich kein Gerät, mit dem ich das Testen könnte – und zum Anderen fehlen mir zu 4G auch die „nackten Daten“. Zu erwarten ist jedoch ein ähnliches Ergebnis: Es müsste für obige Ausführungen also lediglich „3G“ durch „4G“, „2G“ durch „3G“, „UMTS“ durch „LTE“, und „EDGE“ durch „UMTS“ ersetzt werden, während man die Zahlen entsprechend ignoriert.

ROMs: Stock, Vendor, und Custom

ROM – was ist das denn nun wieder? Klar, eine Abkürzung. ROM⁴⁵⁴ steht für **Read-Only Memory**. Also eigentlich Speicher, auf den nur lesend zugegriffen werden kann. Im Falle einer CD oder DVD ist das klar – im Falle unserer Androiden eigentlich glatt gelogen: Hier gehörte dann noch ein „m“ davor, für „mostly“ („meistens“). Denn natürlich kann man da auch Schreiben: Irgendwie müssen die Updates ja da rein kommen.

Wie dumm von mir: Ich sollte erst einmal sagen, was da bei Android eigentlich drin ist. Nämlich das Betriebssystem, sowie die „Core Apps“ (*Google Play Store*, Telefon & Co.). Manchmal auch noch mehr – kommt auf das ROM an. Da gibt es nämlich verschiedene: Stock, Vendor, Custom...



Wikipedia:
Festwertspeicher

Stock ROM

Als „Stock“ ROM bezeichnet man eigentlich das, was direkt aus der Google-Schmiede kommt (und für bessere Eindeutigkeit auch „Vanilla“ genannt wird). Wörtlich eigentlich aus dem „Lager“. Also das „nackte Original“, so wie es von Google eigentlich gedacht war. Naja, die meisten verstehen darunter auch (noch) das Folgende:

Vendor ROM

Das vom Hardware-Hersteller vorinstallierte ROM. Also das, was aus dem gleichen Lager kommt wie das Gerät – weshalb es meist auch als „Stock ROM“ bezeichnet wird. Obwohl das so nicht ganz korrekt ist: Es gibt kaum einen Geräte-Hersteller, der hier nicht noch seine Modifikationen einbaut. Ich denke da nur an HTCs Sense, Motorolas Moto-Blur, oder Samsungs TouchWiz Oberfläche. Oder an das ganze Gesocks an vorinstallierten Apps, die „Otto Normalbenutzer“ gar nicht und „Super User“ auch nur mit Mühe wieder los wird. Genau genommen fehlt jetzt gar noch eine Kategorie, denn auch die Provider spielen nochmals dran rum, und pappen ihren Brandy, pardon, ihr Branding noch oben drauf...

Während Updates von Google natürlich recht regelmäßig, und Updates vom Hersteller noch „relativ zügig“ kommen, bremst das Branding natürlich ein weiteres Mal: Denn auch hier muss erst das Zusammenspiel getestet werden. So

454. <http://de.wikipedia.org/wiki/Festwertspeicher>

kommt ein Hersteller-Update in der Regel frühestens im Quartal nach dem Google-Update. Und das Update fürs gebrandete Gerät kann sich durchaus noch zusätzlich um ein halbes Jahr verspätzen (ein [Artikel bei Heise](#)⁴⁵⁵ aus dem Jahr 2013 sowie [bei AreaMobile](#)⁴⁵⁶ aus 2016 beschreiben das ausführlicher). Zu diesem Zeitpunkt ist unter Garantie schon wieder mindestens ein neues Stock-ROM-Update draußen.



Heise: Der lange Weg zum Android-Update



Android-Update erklärt

Custom ROM

Kostümiert? Zugegeben, auch das gewissermaßen. Aber eigentlich steht „Custom“ für „customized“, also auf den Kunden (customer) angepasst. Im Gegensatz zum Hersteller, oder Provider, zu sehen. Hier stehen meist eingespielte Teams von Entwicklern dahinter – und ein neues Custom-ROM steht häufig schon kurz nach der „Vanilla“-Stock-Version (nicht selten auch mal vorher) zur Verfügung.



AndroidPIT Wiki: Custom ROM

Das [AndroidPIT Wiki](#)⁴⁵⁷ schreibt zu diesem Thema:

Eine auf Open Source basierende Software ist zur Entwicklung durch Dritte freigegeben.

Somit haben Entwickler die Möglichkeit, die Original-Software zu modifizieren, zu optimieren, Elemente hinzu zu fügen oder auch zu entfernen [mit dem Ergebnis], für den Verbraucher das bestmögliche Ergebnis zu bieten in den Bereichen Performance, Energie, Effizienz etc.

Womit auch bereits zahlreiche Vorteile geklärt wären. Hinzu kommt noch: Selbst Geräte, welche in Sachen „Updates“ von ihren Herstellern schon lange aufgegeben wurden, lassen sich Dank Custom ROMs noch mit der aktuellen Android-Version (oder zumindest einer aktuelleren, als vom Hersteller bereitgestellt) versorgen. Haken an der Sache: Um ein Custom ROM installieren zu können, muss der Androide i. d. R. [gerootet](#) sein.



CyanogenMod Homepage

Die bekanntesten Custom-ROMs kommen sicher vom Team um Steve Kondik, und hören auf den Namen [CyanogenMod](#)⁴⁵⁸. Aber es gibt noch zahlreiche Spezial-ROMs, die direkt auf bestimmte Geräte zugeschnitten sind (wie etwa *WildPuzzle* und *OpenFire* im Falle des HTC Wildfire). Hier informiert man sich am besten im root-Forum des betroffenen Gerätes – etwa bei [Android-Hilfe.DE](#)⁴⁵⁹, oder (auf Englisch) bei den [XDA-Developers](#)⁴⁶⁰.

455. <http://heise.de/-2072706>

456. <http://www.areamobile.de/news/36215-android-update-erklaert-von-der-entwicklung-bis-zum-roll-out>

457. <http://www.androidpit.de/de/android/wiki/view/ROMs>

Explizit erwähnt sei in diesem Zusammenhang auch [Replicant](#). Dabei handelt es sich um einen Android-Fork, der sich „pures Open Source“ zum Ziel gesetzt hat. Gerade vor dem Hintergrund der aktuellen Schnüffel-Skandale eine gute Sache: So lässt sich das Vorhandensein etwaiger vorinstallierter „Schnüffel-Software“ genauestens prüfen (bzw. ausschließen). Leider werden derzeit nur wenige Geräte unterstützt. Dennoch hat dieses Projekt sogar das Interesse der [FSF](#)⁴⁶¹ geweckt, die zur Finanzierung neuer Hardware für die Unterstützung weiterer Geräte eine [Spendenkampagne](#)⁴⁶² ins Leben gerufen hat.

Bei der generellen Suche nach einer passenden ROM ist [dieser Artikel bei Stack Exchange](#)⁴⁶³ ein guter Anlaufpunkt.



FSF startet Spendenkampagne für Replicant

Selbst installieren?



The screenshot shows the ROM Manager Premium app interface. At the top, it displays the status bar with signal strength, battery level, and time (9:53). Below the status bar is the app's header: "ROM Manager Premium v5.5...". The main menu is organized into sections: "RECOVERY" (containing "Flash ClockworkMod Recovery" and "Reboot into Recovery" options), "ROM MANAGEMENT" (containing "Install ROM from SD Card" and "Download ROM" buttons), and "BACKUP AND RESTORE" (containing "Check for ROM Updates" and "Manage and Restore Backups" buttons). At the bottom of the screen is a navigation bar with icons for back, forward, and home.

Kann man sich nun selbst ein ROM eigener Wahl installieren? Klar doch – denn außer vielleicht einem „guten Kumpel“ nimmt einem das kaum jemand ab. Am wenigsten der Hersteller. Mit den richtigen Werkzeugen (wie z. B. dem [ROM Manager](#)⁴⁶⁴) ist das auch recht problemlos bewerkstelligt – die App bietet u. a. eine Übersicht der für das jeweilige Gerät verfügbaren ROMs, lädt diese herunter, und führt durch den Installationsprozess. Einschließlich Backups, und was so dazu gehört.

Besonders vorbildlich ist das Ganze im Falle von *CyanogenMod*: Hier stellt das [Wiki](#)⁴⁶⁵ für jedes unterstützte Gerät eine Anleitung bereit. Wie dies in der Praxis aussieht, und ob es wirklich so „schwierig“ ist, wie es beim ersten Mal klingt – lässt sich



Stack Exchange: Where can I find stock or custom ROMs for my Android device?



ROM Manager



CyanogenMod Wiki

-
- 458. <http://www.cyanogenmod.com/>
 - 459. <http://www.android-hilfe.de/>
 - 460. <http://forum.xda-developers.com/>
 - 461. http://de.wikipedia.org/wiki/Free_Software_Foundation
 - 462. <http://www.go2android.de/fsf-startet-spendenkampagne-fur-android-fork-replicant/>
 - 463. <http://android.stackexchange.com/q/17152/16575>
 - 464. <https://play.google.com/store/apps/details?id=com.koushikdutta.rommanager>
 - 465. <http://wiki.cyanogenmod.com/>



AndroidPIT:
Gingerbread auf's
Wildfire

übrigens für das Beispiel „HTC Wildfire bekommt Gingerbread“ in [diesem Blog](#)⁴⁶⁶ nachlesen.

Wie bereits im vorigen Kapitel beschrieben, ist allerdings das [rooten](#) des Androiden i. d. R. eine Grundvoraussetzung, ohne die man kein Custom ROM installieren kann.

Zugriffe Sperren: Firewalls & Permission-Blocker

So mancher App möchte man kräftig auf die Finger hauen: Was will das Teil ständig im Internet? Dieser Teil der App-Funktionalität lässt sich leider auch nicht separat abschalten. Oder man weiß nicht, wann eine App ins Internet will, und wozu schon gar nicht. Nur die entsprechende Permission hat die App halt (jaja, das haben sie fast alle – in der Regel zum Laden von Werbung). Kurz und gut: Sie soll das nicht! Ist das hinzubekommen?



DroidWall



Wikipedia: Iptables

Na, wenn der Izzy das hier so dumm fragt, hat er bestimmt auch ... Richtig. Eine mögliche Antwort nennt sich [DroidWall](#)⁴⁶⁷. Dabei handelt es sich um ein FrontEnd für [iptables](#)⁴⁶⁸, welches im System integriert sein muss. Dies ist bei den vom Hersteller gelieferten Geräten von Haus aus leider nicht immer der Fall, sodass die App häufig ein [Custom ROM](#) voraussetzt. Was [root](#) natürlich gleich impliziert.

Sind diese Voraussetzungen jedoch gegeben, hat man mit *DroidWall* ein gutes Werkzeug zur Hand: Entweder, man verbietet einzelnen Apps den Zugriff auf's Internet (erstellt also eine sogenannte „Blacklist“ der „schwarzen Schafe“) – oder man verbietet den Internet-Zugriff generell für alle Apps außer denen, die auf eine sogenannte „Whitelist“ (die Apps mit der „weißen Weste“) gesetzt wurden. Noch besser: Dies lässt sich auch getrennt nach „Netzwerk Interface“ tun – was



466. <http://www.androidpit.de/de/android/blog/395396/Gingerbread-auf-s-Wildfire>

467. <https://play.google.com/store/apps/details?id=com.googlecode.droidwall.free>

468. <http://de.wikipedia.org/wiki/Iptables>

sich z. B. anbietet, wenn nur ein begrenztes Datenvolumen im Mobilfunk-Vertrag enthalten ist: So kann man einer App verbieten, die „mobile Datenleitung“ zu benutzen – ihr aber gleichzeitig den Internet-Zugriff per WLAN erlauben.

Darüber hinaus lassen sich Zugriffe auch protokollieren. So weiß man anschließend zumindest, was „abging“.

Seit einiger Zeit steht es mit [AFWall+](#)⁴⁶⁹ sogar ein verbesserter Nachfolger zur Verfügung, der noch mehr Leistungsfähigkeit verspricht. Dessen Screenshot liefert dafür auch sogleich ein Beispiel: So lässt sich zusätzlich der Netzwerk-Zugriff während des Roaming steuern.

Weitere Alternativen finden sich in einer [passenden Übersicht](#)⁴⁷⁰ – einschließlich einiger Kandidaten, die auch ohne root auskommen.



Okay, prima – aber das betrifft ja nur Internet-Zugriffe. Klar, schon eine gute Sache: Eine App, die nicht ins Internet kommt, kann auch nicht meine privaten Daten dorthin übertragen. Aber wie schaut das mit den anderen Dingen aus? Zugriff auf meinen aktuellen Standort, meine SMS, meine Adressdaten?

Das sah lange schlecht aus. Als erste Lösung gab es dafür dann [LBE Privacy Guard](#)⁴⁷¹. Der passt im Hintergrund auf – und sobald eine App auf etwas „kritisches“ zugreifen will, erhält der Benutzer eine entsprechende Warnung. Nun kann dieser entscheiden: Darf,

darf nicht? Nur diesmal – oder ist es eine dauerhafte Entscheidung? Im letzteren Falle setzt man das entsprechende Häkchen, und erhält in der gleichen Situation beim nächsten Mal keine Warnung mehr.



Übersicht: Firewalls

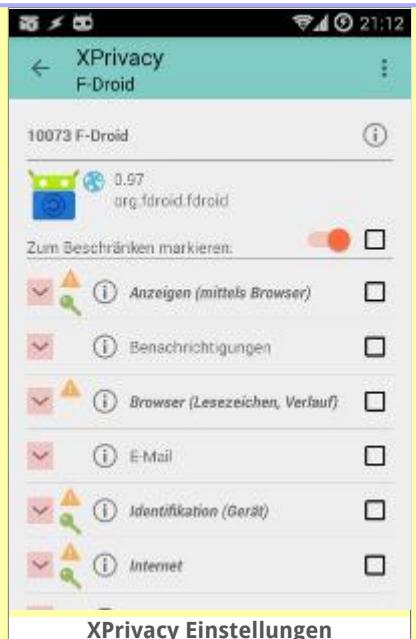


LBE Privacy Guard

469. <https://play.google.com/store/apps/details?id=dev.ukanth.ufirewall>
 470. https://android.izysoft.de/applists/category/named/network_admin_firewall
 471. <https://play.google.com/store/apps/details?id=com.lbe.security.lite>

Vorsicht ist jedoch geboten, wenn auf dem Androiden bereits Jelly Bean oder eine neuere Android-Version läuft: Keinesfalls die Version aus dem Playstore installieren, sonst hängt das Gerät anschließend in einem Boot-Loop fest! Eine Alternative wäre der *LBE Security Master*, der im Playstore allerdings lediglich in einer chinesischen Version verfügbar ist. Zum Glück haben sich unsere Freunde bei den *XDA Developers* bereits mächtig ins Zeug gelegt, und bieten lokalisierte Versionen⁴⁷² der App zum Download an.

Wem *LBE* aufgrund seiner proprietären Eigenschaften spanisch, Pardon, zu chinesisch erscheint (die App kommt aus China, und ist „Closed Source“; es kann also niemand für nichts garantieren), der mag vielleicht einen Blick auf XPrivacy⁴⁷³ werfen. Diese App ist vollständig „Open Source“ (GPL v3), und baut auf dem Xposed Framework⁴⁷⁴ auf. Das Framework legt zahlreiche Schnittstellen offen, die tief ins System reichen (daher auch der Name: „expose interfaces“). Wie der Screenshot zeigt, ist der Leistungsumfang in Sachen Zugriffsschutz durchaus mit *LBE* messbar: Auch hier lassen sich Apps einzelne Rechte entziehen. Eine Information, ob die betroffene App jemals versucht hat, von diesen Rechten Gebrauch zu machen, hat man ebenfalls: Ein grüner Schlüssel zeigt, dass die App über eine Permission verfügt – und ein rotes „Warndreieck“ weist darauf hin, dass darauf zugegriffen wurde. (Das *Xposed* Framework leistet übrigens sehr viel mehr; eine Auswahl verschiedener Möglichkeiten zeigt Nico in einem Blog-Beitrag bei AndroidPIT⁴⁷⁵, einige Ressourcen-Sammlungen⁴⁷⁶ sowie



XDA Developers: LBE Security Master



Xprivacy bei den XDA-Developers



XDA Developers:
Xposed - ROM
modding without
modifying APKs



IzzyOnDroid: Xposed
Ressourcen-
Sammlung

472. <http://forum.xda-developers.com/showthread.php?t=1422479>

473. <http://forum.xda-developers.com/showthread.php?t=2320783>

474. <http://forum.xda-developers.com/showthread.php?t=1574401>

475. <http://www.androidpit.de/xposed-framework-module-vorstellung>

476. https://android.izzysoft.de/applists/category/named/resources_xposed

eine [Vorstellung](#)⁴⁷⁷ finden sich bei *IzzyOnDroid*.) Eine Auflistung weiterer Alternativen sowie ihrer Fähigkeiten gibt ein [Post bei Stack Exchange](#)⁴⁷⁸.

Ach ja: Versteht sich von selbst, dass solche Apps natürlich [root](#) voraussetzen

...

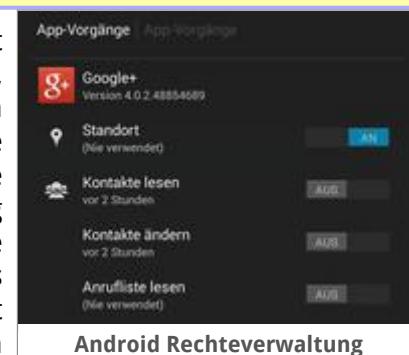
Eine gute Nachricht: Mit Android 4.3 hat Google, wenn auch zunächst noch versteckt, eine Rechteverwaltung eingeführt, mit der man Apps auch nach deren Installation einzelne Berechtigungen wieder entziehen kann (siehe Screenshot). Ein Schritt, der längst überfällig war. Zwar lassen sich hier bei weitem nicht alle Berechtigungen bearbeiten (der Zugriff auf das Netzwerk bzw. Internet etwa ist hier nicht aufgeführt), doch zumindest kann man nun endlich auch ohne root für ein wenig mehr Privatsphäre sorgen. Unterbinden lässt sich etwa das Senden, Lesen und Empfangen von SMS, der Zugriff auf Kontakte, Kalender und Anruflisten, das Erzeugen von Benachrichtigungen, sowie die Nutzung von Kamera und Mikrofon. Per App natürlich. Da hierbei allerdings tatsächlich lediglich der Zugriff unterbunden wird (anders als bei den o. g. Kandidaten, die stattdessen Fake-Informationen liefern), kann eine darauf nicht vorbereitete App beim Versuch, auf eine entsprechende Ressource zuzugreifen, dann abstürzen; was sicher einer der Gründe ist, warum diese Funktionalität zunächst noch versteckt wurde.

Wie gelangt man nun an diese Konfigurationsmöglichkeit? Das Einfachste ist sicher die Installation eines passenden Shortcuts, etwa durch die App [App Ops Starter](#)⁴⁷⁹. Wer allerdings einen Launcher wie beispielsweise *Nova Launcher* oder *Apex Launcher* verwendet, benötigt keine Extra-App dafür. Mit diesen Kandidaten lässt sich ein entsprechender Shortcut direkt erstellen, indem auf eine freie Stelle des Homescreens länger gedrückt, und dann im Menü *Verknüpfungen* → *Aktivitäten* ausgewählt wird. Nun scrollt man zu *Einstellungen*, und öffnet diesen Punkt. In der recht langen Liste ihrer Aktivitäten gilt es schließlich, den Eintrag *App-Vorgänge* (Untertitel: `Settings$AppOpsSummaryActivity`) zu finden, und durch Antippen desselben den Shortcut auf dem Homescreen zu erzeugen.

Ab Android 4.4 verlegte man sich jedoch aufs Ping-Pong spielen – und versuchte, diese Funktionalität wieder zu sperren. So funktionieren die meisten



IzzyOnDroid: Xposed Vorstellung



Stack Exchange: How to fake my personal information?



App Ops Starter

477. <https://android.izzysoft.de/articles/named/xposed-intro>

478. <http://android.stackexchange.com/q/39463/16575>

479. <https://play.google.com/store/apps/details?id=com.schurich.android.appopsstarter>



dieser [AppOps FrontEnds](#)⁴⁸⁰ lediglich bis Android 4.4.1, bzw. wurden gar nicht mehr aktualisiert. Nur eines vermerkt explizit die Kompatibilität mit Android 5. Android 6 „Marshmallow“ stellt die Einstellungen dann endlich offiziell „von Haus aus“ zur Verfügung.

Übersicht: AppOps
FrontEnds

480. http://android.izzysoft.de/applists/category/named/security_permissions#group_1027

ADB: Die Android Debug Bridge

Sie kam ja nun im Buch einige Male zur Sprache: Die Android Debug Bridge. Wie ich zeigen konnte, ist dieses Werkzeug nicht nur für Entwickler interessant; auch für den „normalen Anwender“ kann sie sich durchaus als nützlich erweisen. Mittels der ADB kann man nämlich u. a.:

- [Apps und ihre Daten sichern](#) sowie wiederherstellen
- [Apps installieren](#) (oder auch installierte Apps löschen)
- [System- und Fehlerprotokolle einsehen](#)
- [Direkt auf die Kommandozeile zugreifen](#)
- [Dateien kopieren](#) (in beide Richtungen)
- Unter Linux gar [das Android-Dateisystem einbinden](#)
- Auch [Daten retten](#)
- Und [weitere Dinge](#) tun

Und da ich weiß, dass die meisten Leser erst dann an die [Installation](#) gehen, wenn sie dafür einen Grund sehen, kommt dieselbe auch erst am Schluss.

Backup & Restore

Diesem Thema widmete sich ja bereits das Kapitel [Vollständiges Backup ohne root](#), und ging auch gleich auf grafische Helperlein ein. Nicht nur ein weitestgehend vollständiges Backup lässt sich auf diese Weise erledigen: Man kann auch gezielt einzelne Apps oder auch nur deren Daten sichern. Das bietet den Vorteil, dass sich selbige auf die gleiche Weise einzeln wieder herstellen lassen – eine Tatsache, die auch das dort vorgestellte *Helium Backup* ausnutzt.

Wer das Ganze von der Kommandozeile aus erledigen möchte, ist u. U. noch ein wenig flexibler. Allerdings muss er dazu die *Paketnamen* der zu sichernden Apps kennen. Doch diese lassen sich einfach herausfinden, indem man mit dem Webbrower die Seite der entsprechenden App im [Google Playstore](#)⁴⁸¹ aufsucht. Am Beispiel der App *Tasker* möchte ich das hier kurz veranschaulichen: Ihre URL heißt <https://play.google.com/store/apps/details?id=net.dinglisch.android.taskerm>, und wird auch in der Adresszeile des Browsers angezeigt. Den Paketnamen findet man im Parameter „id“, er lautet in diesem Beispiel also net.dinglisch.android.taskerm. Alles ab (und einschließlich) etwa folgenden „&“ oder „#“ Zeichen ist dabei zu ignorieren.



Google Playstore

481. <https://play.google.com/store/apps>

Da wir nun den Paketnamen kennen, ein kurzer Blick auf die Syntax der Befehle adb backup und adb restore:

```
adb backup [-f <file>] [-apk|-noapk] [-shared|-noshared] [-all] [-system|-nosys
```

- Mit -f <file> wird der Name der zu erstellenden Backup-Datei angegeben. Default ist: backup.ab, wobei .ab für „Android Backup“ steht.
- -apk bzw. -noapk gibt an, ob die .apk Datei der App mitgesichert werden soll. Per Default geschieht dies nicht, da man sie ja (theoretisch) jederzeit aus dem Playstore wieder beziehen kann.
- -shared und -noshared beziehen sich auf den „geteilten Speicherplatz“, also den Inhalt der SD-Karte. Standardgemäß wird dieser nicht mit gesichert.
- -all heißt genau das, was man vermuten könnte: Alle Apps sichern.
- -system / -nosystem legt fest, ob System-Apps ebenfalls mit gesichert werden sollen. Gibt man keinen der beiden Parameter an, werden sie mit erfasst.
- Mit <packages...> am Schluss der Kette listet der Anwender die Apps auf, die gesichert werden sollen. Optional, wenn entweder -all oder -shared angegeben wurde.

Unsere Beispiel-App *Tasker* einschließlich ihrer Daten, aber ohne alles andere, würde also der Befehl adb backup -f tasker.ab -apk net.dinglisch.android.taskerm in die Datei tasker.ab sichern. Und wie schaut es mit dem Restore aus?

```
adb restore <file>
```

Hier gibt es keine zahlreichen Optionen: adb restore ist immer ein Alles-oder-Nichts, und stellt jeweils den gesamten Inhalt der Backup-Datei auf dem Android-Gerät wieder her. Wer „Einzelteile“ benötigt, greift entweder zu *Titanium Backup*, oder zu einem Tool wie [Nandroid Manager](#)⁴⁸². Beide setzen allerdings ein gerootetes Gerät voraus.



Nandroid Manager

Apps installieren und löschen

Sicher, dies sollte eigentlich über den *Google Playstore* geschehen. Doch ab und an findet man sich in einer Situation, wo doch einmal eine „alternative Quelle“ herhalten muss – etwa, wenn der Entwickler auf einen eingeschickten Fehlerbericht hin darum bittet zu prüfen, ob der Fehler behoben ist – bevor er die neue Version veröffentlicht. Oder wenn man eine App auf ein Gerät übertragen

482. <https://play.google.com/store/apps/details?id=com.h3r3t1c.bkrestore>

will, welches selbst nicht ins Netz kommt. Das lässt sich dann beispielsweise mittels `adb install` erledigen:

```
adb install [-r] [-s] <file>
```

Der Befehl kennt zwar noch weitere Parameter, doch an dieser Stelle möchte ich mich auf diese zwei beschränken: `-r` für den „Re-Install“ einer bereits installierten App unter Beibehaltung der Daten (für das Beispiel der App vom Entwickler), und `-s` für die Installation auf die SD-Karte. Haben wir also eine Datei namens `Fixed.apk` erhalten, kann diese mittels `adb install -r Fixed.apk` direkt über die bereits installierte „kaputte“ App installiert werden.

Doch auch die De-Installation einer App ist auf ähnliche Weise möglich:

```
adb uninstall [-k] <package>
```

Dafür wird wieder der Paketname der App benötigt (siehe [Backup & Restore](#)). Der Parameter `-k` kann dabei verwendet werden, wenn man die Daten noch auf dem Gerät behalten möchte („keep“) – etwa weil man beabsichtigt, die App zu einem späteren Zeitpunkt wieder zu installieren.

System- und Fehlerprotokolle einsehen

Wenn einmal wieder etwas „spinnt“, und man der Sache auf den Grund gehen möchte, sind Log-Dateien eine prima Sache. Nur herankommen muss man zunächst an selbige! Mittels ADB gar kein Problem. Im Gegenteil: Gleich mehrere Helferlein stehen dafür „Gewehr bei Fuß“:

- `adb bugreport > report.txt` erzeugt einen ausführlichen Fehlerbericht, und legt diesen in der Datei `report.txt` ab. Technisch weniger versierte können diesen an den „Guru ihrer Wahl“ (oder den Entwickler der problemverursachenden App) weiterleiten.
- `adb logcat [<option>] ... [<filter-spec>] ...` ermöglicht den Zugriff auf Systemlogs. Die zahlreichen Parameter sind auf der [Hilfeseite des Befehls](#)⁴⁸³ ausführlich erklärt. Für die bessere Lesbarkeit möchte ich hier lediglich einen Parameter nennen und empfehlen: `-v time` erleichtert die Suche im Log, da es jeder Zeile zusätzlich Datum und Uhrzeit voranstellt.
- `adb shell dmesg` Listet Log-Nachrichten vom Betriebssystemkern



Logcat Help

483. <http://developer.android.com/tools/help/logcat.html>

Shell-Zugriff

Linux-Anwender haben von „der Shell“ zumindest schon einmal gehört. Die Chancen stehen für sie sogar gut, dass sie mehr oder weniger versiert im Umgang mit selbiger sind. Diese „Kommando-Zeile“ bietet Zugriff auf zahlreiche Systemfunktionen; auch Aktivitäten verschiedener Apps lassen sich derart starten. Eine kleine Anleitung zum Aufspüren verfügbarer Befehle gibt u. a. die [Android Shell Command Reference](#)⁴⁸⁴ des *Android Terminal Emulators*.

Im Umgang mit Apps sind hier insbesondere der *Package Manager* (pm) und der App Manager (am) interessant. Was man damit anstellen kann, geben beide Befehle kund, wenn man sie ohne Parameter aufruft:

```
adb shell pm
usage: pm list packages [-f] [-d] [-e] [-s] [-e] [-u] [FILTER]
    pm list permission-groups
    pm list permissions [-g] [-f] [-d] [-u] [GROUP]
    pm list instrumentation [-f] [TARGET-PACKAGE]
    pm list features
    pm list libraries
    pm path PACKAGE
    pm install [-l] [-r] [-t] [-i INSTALLER_PACKAGE_NAME] [-s] [-f] PATH
    pm uninstall [-k] PACKAGE
    pm clear PACKAGE
    pm enable PACKAGE_OR_COMPONENT
    pm disable PACKAGE_OR_COMPONENT
    pm disable-user PACKAGE_OR_COMPONENT
    pm set-install-location [0/auto] [1/internal] [2/external]
    pm get-install-location
    pm createUser USER_NAME
    pm removeUser USER_ID
[...]
```

Mit dem Package-Manager lassen sich also alle installierten Apps anzeigen, bereits auf dem Gerät befindliche .apk Dateien installieren, und generell Apps verwalten. Besonders interessant dürfte für Einige die Tatsache sein, dass sie das bevorzugte Ziel für App-Installationen angeben können: Mit `adb shell pm set-install-location 2` etwa ließe sich das System anweisen, ab sofort (sofern möglich) alle Apps auf der SD-Karte zu installieren. Was aktuell als Installationsziel eingestellt ist, kann man mit `adb shell get-install-location` in Erfahrung bringen.

Dieser kurzen Auflistung folgen in der Ausgabe noch weitere Details, welche die verfügbaren Optionen beschreiben.

```
adb shell am
usage: am [subcommand] [options]
```

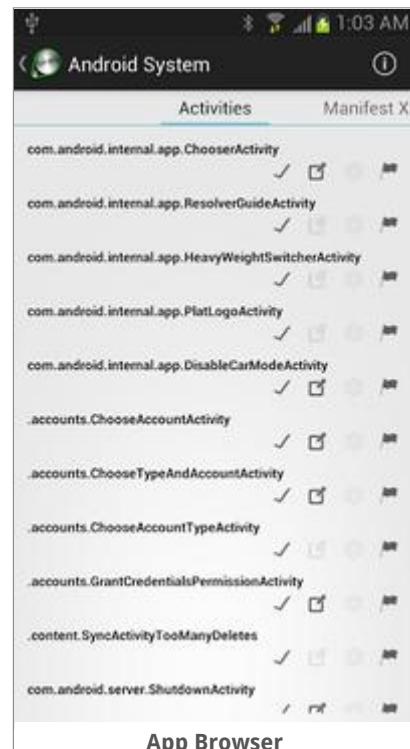
484. <https://github.com/jackpal/Android-Terminal-Emulator/wiki/Android-Shell-Command-Reference>

```
usage: am start [-D] [-W] [-P <FILE>] [--start-profiler <FILE>]
                [--R COUNT] [-S] <INTENT>
    am startservice <INTENT>
    am force-stop <PACKAGE>
    am kill <PACKAGE>
    am kill-all
    am broadcast <INTENT>
    am instrument [-r] [-e <NAME> <VALUE>] [-p <FILE>] [-w]
                  [--no-window-animation] <COMPONENT>
    am profile [looper] start <PROCESS> <FILE>
    am profile [looper] stop [<PROCESS>]
    am dumpheap [flags] <PROCESS> <FILE>
    am set-debug-app [-w] [--persistent] <PACKAGE>
    am clear-debug-app
    am monitor [--gdb <port>]
    am screen-compat [on|off] <PACKAGE>
    am display-size [reset|MxN]
    am to-uri [<INTENT>]
        am to-intent-uri [<INTENT>]
[...]
```

Ähnlich sieht es für den *App Manager* aus.
 Mit diesem lassen sich Aktivitäten von Apps direkt starten (was etwa dem Antippen von Shortcuts auf dem Homescreen entspricht). Ebenso kann man Prozesse „killen“, und mehr. Auch hier folgt der kurzen Auflistung wieder eine Erläuterung verfügbarer Optionen.

Das mag jetzt alles recht abstrakt erscheinen – daher ein paar kurze praktische Beispiele:

- adb shell "am broadcast -a



android.intent.action.MEDIA_MOUNTED -d file:///mnt/sdcard": Veranlasst den Media-Scanner, die SD-Karte auf gelöschte bzw. hinzugefügte Medien zu untersuchen (genau genommen wird dem



ASE: Can I trigger a media scan via the command line

System vorgegaukelt, die SD-Karte wäre gerade eingebunden worden). Wer dies per SSH versuchen möchte, muss ggf. noch entsprechende Umgebungs-Variablen setzen⁴⁸⁵.

- adb shell "am start -a android.intent.action.VIEW -d https://duckduckgo.com/": Öffnet die angegebene URL im Browser und bringt diesen in den Vordergrund
- am start -a android.intent.action.CALL -d tel:012345678: Startet einen Anruf bei der angegebenen Nummer. Praktisch, wenn man schon das Headset aufhat.
- adb shell "am start -n com.android.settings/com.android.settings.Settings": Öffnet das Einstellungsmenü und bringt es in den Vordergrund. Klappt auch mit anderen Apps bzw. Untermenüs (so gelangt man z. B. mit adb shell am start -a android.settings.WIRELESS_SETTINGS direkt ins Netzwerk-Menü), sofern man den Namen der zugehörigen Intents kennt. Herausfinden lassen sich selbige beispielsweise mit dem App Browser⁴⁸⁶. Wer den Apex Launcher⁴⁸⁷ einsetzt, findet die passenden Aktivitäten auch direkt aus selbigem heraus: Lange auf eine freie Stelle auf dem Homescreen drücken, Verknüpfungen, Aktivitäten.
- adb shell "am start -a android.intent.action.INSERT -t vnd.android.cursor.dir/contact -e name 'Martin Mustermann' -e phone 1234567890": Einen neuen Kontakt anlegen
- busybox sed -n 's/<package name="\\([^\"]\\+\\)".*enabled=„false”.*/\\1/p' /data/system/packages.xml | while read PKG; do pm enable "\$PKG"; done: Wenn man im Übereifer zu viele Apps „eingefroren“ (deaktiviert) hat, lassen sie sich alle auf einen Schlag wieder auftauen – auch, wenn man an das entsprechende Tool auf dem Androiden (etwa *Titanium Backup*) genau deshalb nicht mehr herankommt.

Schaut vielleicht ein wenig kryptisch aus (Wer soll sich sowas merken?) – aber es hindert uns schließlich niemand daran, es in Batch- oder Shell-Skripten zu verwenden. Ein Anruf-Skript könnte etwa folgendermaßen aussehen:

```
#!/bin/bash
if [ -z "$1" ]; then
    read -p "Welche Nummer soll angerufen werden: " NUMBER
else
    NUMBER=$1
fi
```

485. <http://android.stackexchange.com/a/51800/16575>

486. <https://play.google.com/store/apps/details?id=com.japanesecrackers.appbrowser>

487. <https://play.google.com/store/apps/details?id=com.anddoes.launcher>

```
echo "Initialisiere Anruf..."  
am start -a android.intent.action.CALL -d tel:$NUMBER
```

Das ganze unter dem Namen `anruf` gespeichert, mit `chmod +x anruf` ausführbar gemacht, sowie im `$PATH` abgelegt – und schon kann von der Kommandozeile mit `anruf 012345678` ein Anruf gestartet werden. Vergisst man, die Rufnummer mitzugeben, fragt das Skript sie sogar ab.

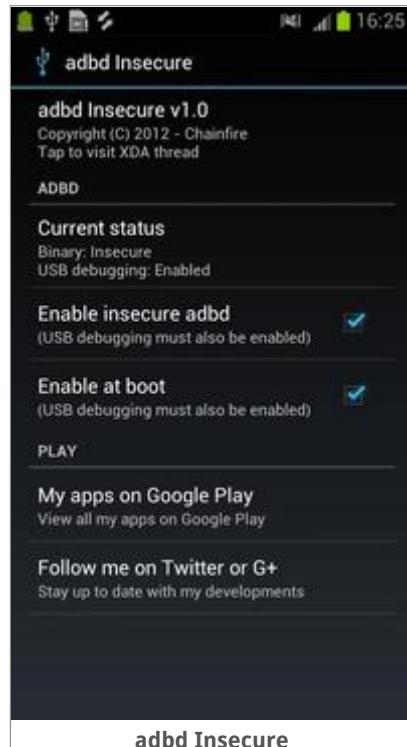
Dateien kopieren

Auch für diesen Zweck stellt die ADB zwei passende Befehle bereit: `adb push`, um etwas auf den Androiden zu befördern, sowie `adb pull`, um etwas herunterzuziehen. Dabei kann dieses „etwas“ eine Datei, aber auch ein ganzer Verzeichnisbaum sein.

Das Tückische an diesen beiden Befehlen ist: Der Typ von Quelle und Ziel muss übereinstimmen. Der Versuch, eine Datei mit z. B. `adb push MeineApp.apk /data/local` in ein Verzeichnis zu kopieren, schlägt daher fehl: Die Quelle ist eine Datei (`MeineApp.apk`), das Ziel jedoch ein Verzeichnis (`/data/local`). Damit es klappt, müsste es also heißen: `adb push MeineApp.apk /data/local/MeineApp.apk`.

Analoges gilt auch für `adb pull`. Und natürlich kann man damit nur Dinge vom Androiden holen, auf die man auch Zugriff hat. Da der ADB Daemon normalerweise nicht im root-Modus läuft (was sich auf gerooteten Geräten beispielsweise mit Chainfire's `adb Insecure`⁴⁸⁸ ändern lässt, wie im rechten Bild zu sehen), ist mit „Zugang Verboten“ Meldungen für Verzeichnisse wie `/data/data` zu rechnen.

Linux-User müssen sich jedoch nicht unbedingt mit Push und Pull herumschlagen. Sie können den Androiden On-the-Fly via ADB direkt in das lokale Dateisystem einbinden, und dann mit den „ganz normalen“ Kopierbefehlen (oder beliebigen grafischen Tools) darauf zugreifen:



adb Insecure

488. <https://play.google.com/store/apps/details?id=eu.chainfire.adbd>

Linux: Android Dateisystem am Rechner einbinden



adbfs-rootless

Linux-Anwender sollten unbedingt einen Blick auf [adbfs-rootless](#)⁴⁸⁹ werfen. Zwar muss der Code selbst kompiliert werden, doch das ist (nicht zuletzt Dank der einfachen Anleitung) ein Kinderspiel. Das erzeugte Binary namens `adbfs` kopiert man in ein Verzeichnis, welches in der `$PATH` Variable enthalten ist. Dann erstellt man sich ein Verzeichnis, in das das Android-Dateisystem eingebunden werden soll (beispielsweise `~/droid`), und schon kann man – als Anwender, ganz ohne root – jederzeit das Android-Gerät „einhängen“:

```
# Android Dateisystem einbinden:  
adbfs ~/droid  
# Dateisystem wieder aushängen:  
fusermount -u ~/droid
```

Ist das Android-Dateisystem auf diese Weise eingebunden, lässt sich darauf ganz normal zugreifen. Etwa mit dem *Midnight Commander* im Terminal-Fenster, oder auch mit grafischen Dateimanagern wie *Dolphin*, *Thunar*, *Nautilus*, oder *PCMan* – was immer man bevorzugt.

Um auf einem gerooteten Gerät auch auf Systemdateien zugreifen zu können, sei nochmals auf das bereits unter [Dateien kopieren](#) genannte *adbd Insecure* von Chainfire verwiesen.



Midnight Commander

Nutzer des [Midnight Commanders](#)⁴⁹⁰ kommen gar ganz ohne „Mount-Befehle“ aus, indem sie [mc-extfs-adb](#)⁴⁹¹ verwenden. Hierbei handelt es sich um ein „virtuelles Dateisystem“, welches sich in den *Midnight-Commander* integrieren lässt – wozu man einfach als root drei Dateien in das entsprechende Verzeichnis (`/usr/lib/mc/extfs.d` unter Debian und Derivaten, `/usr/libexec/mc/extfs.d` unter Fedora, RedHat & Co.) kopiert.



mc-extfs-adb

Anschließend lässt sich mit `cd adb://` direkt auf das Dateisystem zugreifen, ohne dass es zuvor explizit eingebunden wurde. Der erste Zugriff dauert allerdings ein wenig, da zunächst die gesamte Verzeichnisstruktur eingelesen wird. Dafür navigiert es sich anschließend um so schneller durch selbiges: Jetzt erfolgt der Zugriff auf die Strukturen nämlich ausschließlich über den Cache. Ein direkter Zugriff auf Dateien erfolgt erst dann wieder, wenn beispielsweise eine Datei kopiert oder angezeigt werden soll.

489. <https://github.com/spion/adbfs-rootless>

490. <http://www.midnight-commander.org/>

491. <http://forum.cyanogenmod.com/topic/23637-browse-phone-file-system-using-midnight-commander-linux-only/>

Da die unter zuvor genannter URL verfügbare Version bei mir unter Ubuntu 12.04 nicht sauber lief, habe ich sie ein wenig angepasst, und um eine kleine Dokumentation ergänzt. Zu finden ist diese neuere Version bei [Izzy's Android Downloads](#)⁴⁹².



Android Downloads
bei IzzyOnDroid

Daten retten

Dieses Einsatzgebiet beschreibt Joakim Roubert ausführlich [in einem Artikel auf seiner Homepage](#)⁴⁹³. Die Nutzung ist allerdings an ein paar Voraussetzungen gebunden:

- Der Androide muss gerootet sein
- Auf dem Androiden muss busybox installiert sein
- Auf dem PC benötigt man ADB (jedoch nicht, wie Jokke schreibt, das komplette SDK – eine einfache Variante tut es auch) und TestDisk⁴⁹⁴.

Eine ausführliche und illustrierte Beschreibung findet sich hinter genanntem Link. Für den Fall, dass dieser einmal nicht erreichbar sein sollte, möchte ich das Vorgehen hier kurz skizzieren.

Was getan

wird, lässt sich in einem Satz zusammenfassen: Der gesamte interne Speicher wird in eine Image-Datei auf dem Computer kopiert, wo sodann mittels *TestDisk* die Daten extrahiert werden können. Das genaue Vorgehen ist zwar ein wenig aufwändiger – sollte aber dennoch gut zu bewältigen sein.

1. Feststellen, ob sich diese Anleitung 1:1 umsetzen lässt: Gibt es die passende Block-Datei?

```

File Edit View Search Terminal Help
TestDisk 6.13, Data Recovery Utility, November 2011
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org

Disk mmcblk0.raw - 7818 MB / 7456 MiB - CHS 951 255 63
Partition Start End Size in sectors
>P MS Data 26480 26623 6144 [modems]
P MS Data 36864 53247 16384 [apps_log]
P MS Data 118784 2215935 2097152 [system]
P MS Data 2215936 2727935 512000 [/cache]
P MS Data 2727936 6922175 4194240 [/data]
P MS Data 6922240 15269887 8347648 [NO NAME]

Structure: Ok. Use Up/Down Arrow keys to select partition.
Use Left/Right Arrow keys to CHANGE partition characteristics:
  P=Primary D=Deleted
Keys A: add partition, L: load backup, T: change type, P: list files,
Enter: to continue
EXT3 Large file Sparse superblock Recover, 3145 KB / 3072 Kib

TestDisk mit erkannten Partitionen

```



Jokke: One Way to
Use a Linux
Computer to
Recover Files from
an Android Device

```
adb shell "ls /dev/block | grep mmc"
```

Taucht in der Liste mmcblk0 auf, kann es weitergehen.

2. Image erstellen und auf den Computer kopieren:

```
adb shell su -c "cat /dev/block/mmcblk0" | pv > mmcblk0.raw
```

`pv` ist der „Pipe Viewer“, der uns eine Fortschrittsanzeige bietet. Wer ihn nicht hat, lässt einfach das `| pv` weg. Dieser Schritt kann durchaus ein wenig länger dauern (wie lange, hängt von der Größe des verbauten Speichers ab). Anschließend sollte eine Datei namens `mmcblk0.raw` im lokalen Verzeichnis auf dem Computer zu finden sein.

3. Auf diese Image-Datei setzen wir nun *TestDisk* an: `testdisk mmcblk0.raw` ruft das Tool entsprechend auf. Der Startbildschirm sollte dann auch die Datei auswählen lassen.
4. Im nächsten Schritt muss die Partitionstabelle ausgewählt werden. Als „partition table type“ wählen wir dafür „EFI GPT“, und lassen dann eine „Analyse“ durchführen. Hier kommt u. U. eine Fehlermeldung a la „Bad GPT partition, invalid signature“. Ein „Quick Search“ sollte da Abhilfe schaffen.
5. Nun sollte eine Auflistung verfügbarer Partitionen zu sehen sein, wie im Screenshot dargestellt. Für uns interessant ist hauptsächlich die letzte, welche die interne SD-Karte beinhaltet. Doch zunächst bestätigen wir mit der „Eingabetaste“, dass wir die Partitionstabelle so übernehmen wollen – und wählen im nächsten Bildschirm „Write“, um sie zu schreiben. Der Hinweis auf einen etwa nötigen Reboot darf natürlich getrost ignoriert werden.
6. Jetzt wird aus dem Hauptmenü der Punkt „Advanced“, und sodann die gewünschte Partition ausgewählt – in unserem Beispiel ist das die letzte, welche die „interne SD-Karte“ beinhalten sollte. Nun gibt *TestDisk* uns die Möglichkeit, das Dateisystem zu durchforsten, Dateien auszuwählen, und sie (in das lokale Dateisystem unseres Computers) zu kopieren.

Et voilà! Schon lassen sich die verloren geglaubten Daten zurückholen. Will man jedoch auf die `/data` Partition zugreifen, sieht es etwas anders aus: Im Gegensatz zur SD-Karte verwendet diese nämlich ExtFS als Dateisystem. Hier wählt man dann „Image Creation“, um eine Datei namens `image.dd` zu erstellen. Deren Inhalt lässt sich sodan beispielsweise mit `extundelete -- restore-all image.dd` komplett auslesen.

Sollte *TestDisk* das Image nicht lesen können, hat ADB u. U. eine „Zeichensatz-Portierung“ versucht. Um das zu vermeiden, kann `stty raw` eingesetzt werden. Unser Befehl aus Schritt 2 sähe dann, entsprechend angepasst, wie folgt aus:

```
adb shell su -c "stty raw; cat /dev/block/mmcblk0" | pv >
mmcblk0.raw
```

Weitere Einsatzmöglichkeiten

Damit sind die Möglichkeiten von ADB noch lange nicht erschöpft. Auf alles im Detail einzugehen, würde jedoch den Umfang dieses Buches sprengen. Einige Dinge möchte ich hier dennoch kurz anreißen:

Eine Geräte-Dokumentation erstellen

Wie wir bereits gesehen haben, lassen sich per ADB dem Androiden so einige Angaben entlocken. Dies macht sich z. B. auch mein kleines Tool [Adebar](#)⁴⁹⁵ zu Nutze, dessen Name sich als *Android DEvice Backup And Report* lesen lässt. Neben Skripten für Backup und Restore erstellt es auch eine Geräte-Dokumentation mit Angaben zu Hard- und Software, einschließlich installierter Apps und Details zu selbigen. Dafür kommen verschiedene ADB-Befehle zum Einsatz, von denen einige bereits zuvor in diesem Buch vorgestellt wurden.



Adebar Projektseite

Den Androiden „fernbedienen“

Das erweist sich als besonders praktisch, wenn etwa das Display beschädigt ist – kann aber ebenso genutzt werden, um einen ausgedienten Androiden „neuen Aufgaben“ zuzuführen. Eine Beispiel-Anwendung dafür findet sich [bei den XDA-Developers](#)⁴⁹⁶. Das dort vorgestellte Tool wurde ursprünglich für Windows entwickelt; mittlerweile gibt es jedoch auch Portierungen für Windows-XP und Linux. Die Bedienung erfolgt aus einer grafischen Oberfläche heraus, und auch der „Bildschirm“ des Androiden wird auf dem Desktop angezeigt.



XDA: Control a device with a broken screen

Den Androiden verwalten

Tools aus diesem Bereich stehen ebenfalls zahlreich zur Verfügung. Einige nur für Windows, wie etwa Ryan Conrads [Droid Explorer](#)⁴⁹⁷ – andere aber auch Plattform-übergreifend, wie [QtADB](#)⁴⁹⁸. Weitere Tools und Apps finden sich bei [IzzyOnDroid](#) in der [ADB-Tools Übersicht](#)⁴⁹⁹.



IzzyOnDroid: ADB Tools

495. <https://github.com/IzzySoft/Adebar>

496. <http://forum.xda-developers.com/showthread.php?t=2786395>

497. <http://de.codeplex.com/>

498. <http://qtadb.wordpress.com/>

499. http://android.izzysoft.de/applists/category/named/network_admin_adb.html

ADB Installieren



Android SDK
Homepage

Hier denken wahrscheinlich die meisten Leser: Oh weh, das gesamte [Android SDK](#)⁵⁰⁰ mit seinen über 30 MB Download installieren! Und das alles nur für so ein paar kleine Dinge! Weit gefehlt: Wer nicht selbst entwickeln möchte, kommt mit weitaus weniger zurecht. Etwa mit den *Platform Tools*, deren Download lediglich ein Drittel des Umfangs hat (installiert sind es dann allerdings auch wieder über 30 MB).

Doch es geht mit noch weniger: Unter Linux und MacOS reicht eine einzige Datei, das `adb` Binary. Für Windows kommen noch ein paar `.dll` Dateien dazu. Das alles muss lediglich in ein Verzeichnis kopiert werden, welches sich im Pfad befindet (damit der Aufruf problemlos aus jedem beliebigen Verzeichnis heraus funktioniert), und fertig ist die Installation. Unter Linux (sowie wahrscheinlich auch MacOS) müssen die Binaries noch ausführbar gemacht werden (`chmod 0755 adb aapt`). Die nötigen Downloads finden sich unter bereits zuvor genannter URL auf meinem Server – und dokumentiert habe ich das Ganze u. a. in einem [Post bei Stack Exchange](#)⁵⁰¹.

Das war alles? Leider nein, denn unter allen drei genannten Betriebssystemen sind noch ein paar kleinere Anpassungen nötig. So benötigen unsere Freunde unter Windows wieder einmal einen speziellen Treiber für ihr Android-Gerät. Zum Glück muss man diesen i. d. R. nicht mehr so umständlich suchen, wie das noch vor nicht all zu langer Zeit nötig war: Der von *ClockworkMod*, *ROM Manager*, und auch *Helium Backup* bekannte Entwickler Koushik Dutta hat nämlich einen [universellen USB-Treiber für Android-Geräte](#)⁵⁰² entwickelt, der mit den meisten Androiden funktionieren sollte. Ja, auch unter Windows 8. Alternativ lohnt sich auch wieder ein Besuch [bei Stack Exchange](#)⁵⁰³.

Wie es bei unseren Freunden unter MacOS aussieht, entzieht sich meiner Kenntnis. Wahrscheinlich aber ähnlich wie bei uns unter Linux. Wir brauchen nämlich nur eine kleine Zeile in eine Konfigurations-Datei einzufügen, damit unser Androide als solcher erkannt wird. Wie genau das funktioniert, beschreibt wiederum einer meiner Posts [bei Stack Exchange](#)⁵⁰⁴:

Zunächst wird der Androide mit aktiviertem *USB Debugging* per USB-Kabel an den Rechner angeschlossen. Jetzt schauen wir nach, wie er sich denn dort zu erkennen gibt:

-
- 500. <http://developer.android.com/sdk/index.html>
 - 501. <http://android.stackexchange.com/q/42474/16575>
 - 502. <http://adbdriver.com/>
 - 503. <http://android.stackexchange.com/q/16071/16575>
 - 504. <http://android.stackexchange.com/a/39437/16575>

```
user@kiste$ sudo lsusb  
[...]  
Bus 002 Device 054: ID 18d1:4e22 Google Inc. Nexus S (debug)
```

Aha, ein Nexus im Debug-Modus. Für unsere Konfiguration sind jedoch die beiden durch einen Doppelpunkt getrennten Werte interessant: 18d1:4e22. Hierbei handelt es sich um „manufacturerID:deviceID“, also die ID des Herstellers, sowie die des Gerätes. Mit diesen Werten bewaffnet machen wir uns über eine spezielle Konfigurationsdatei her:

```
user@kiste$ sudo su -  
cd /etc/udev/rules.d  
vi 51-android.rules
```

Sollte die Datei noch nicht existiert haben, ist das nicht weiter schlimm. In diesem Fall wird sie automatisch erstellt. Und natürlich darf auch gern ein anderer Editor verwendet werden. Wichtig ist, was nun in diese Datei eingefügt wird:

```
# MyDeviceName  
SUBSYSTEMS=="usb", ATTRS{idVendor}=="18d1", ATTRS{idProduct}=="4e22", MODE="0666" GROUP=
```

Wer genau hinschaut, erkennt auch die beiden Werte wieder: Aus dem „manufacturer“ wurde ein „Vendor“, und aus dem „device“ ein „Product“. Der Eintrag bezieht sich auf das USB Subsystem, und das Gerät soll automatisch mit Lese- und Schreibrechten für alle (MODE="0666"; vorsichtige Leute verwenden besser MODE="0660", zumindest wenn sie sich den PC noch mit anderen Nutzern teilen), und der Gruppe androiddev zugeteilt werden. Ferner wird im Gerätzweig auch noch ein passender symbolischer Link erzeugt.

Ist diese Datei gespeichert, muss noch der entsprechende Dienst neu gestartet werden. Unter Ubuntu geschieht dies beispielsweise mittels `sudo service udev reload`, andere Linux Derivate verwenden ggf. noch `sudo /etc/init.d/udev reload`. Ging dieser Neustart (eigentlich wurde UDev ja nur angewiesen, die Konfiguration neu einzulesen; einen Neustart erreicht man, indem man „reload“ durch „restart“ ersetzt) problemlos über die Bühne, sollte unser Androide nach einmaliger Abtrennung vom USB-Anschluss beim nächsten Anstecken erkannt werden:

```
user@kiste$ adb devices  
List of devices attached  
xxxxxxxxxxxx device
```

ANHANG

Android Permissions – und was sie bedeuten

Normalerweise sieht man eine Kurzbeschreibung der Permission (z. B. bei der Installation einer App). Die technische Bezeichnung taucht selten im Klartext für den Anwender auf – man kann aber z. B. auch einen Blick auf das [Manifest](#)⁵⁰⁵ werfen, und da stehen sie im Klartext.

Nun werde ich aber hier nicht alle Permissions aufführen, das wäre einfach zu viel. Die findet man bei Interesse im [Entwickler-Handbuch](#)⁵⁰⁶ (allerdings auf Englisch). Ein Wiki zum Thema, an dem sich jeder beteiligen kann, existiert ebenfalls, und zwar bei [Stack Exchange](#)⁵⁰⁷ (wiederum auf Englisch), eine weitere gute (englische) Übersicht mit zusätzlichen Sicherheits-Tipps existiert bei [AndroidForums.Com](#)⁵⁰⁸, und auch bei Go2Android findet sich ein [gut erklärter Artikel von MaTT](#)⁵⁰⁹ (übrigens Teil der Serie „anDROID für Anfänger“). Eine erklärte und ständig aktualisierte Übersicht, wer hätte das gedacht, gibt es natürlich auch [bei IzzyOnDroid](#)⁵¹⁰. Ein paar ausgewählte, die doch häufiger einmal auftauchen könnten, möchte ich aber hier kurz erklären.

Permission Groups

Zunächst einmal sind die Permissions in „Berechtigungs-Gruppen“ eingeteilt. Besucht man die Playstore-Seite einer App, findet man diese „Permission Groups“ als Überschriften unter „Diese App möchte auf Folgendes zugreifen:“ wieder. Eine vollständige Auflistung gibt es in der [API Dokumentation](#)⁵¹¹ auf den Entwicklerseiten in englischer Sprache, sowie auf der gerade genannten Seite bei [IzzyOnDroid](#) auf Deutsch. Jede dieser Gruppen fasst zugehörige Berechtigungen zusammen – so etwa „ACCOUNTS“ (Konten) die Permissions für den direkten Zugriff auf vom Account Manager verwaltete Konten, „CALENDAR“ alles, was auf den Kalender zugreift, „LOCATION“ die für den Standort-Zugriff zuständigen

-
- 505. http://de.wikibooks.org/wiki/Googles_Android/_Das_Manifest
 - 506. <http://developer.android.com/reference/android/Manifest.permission.html>
 - 507. <http://android.stackexchange.com/q/38388/16575>
 - 508. <http://androidforums.com/android-applications/36936-android-permissions-explained-security-tips-avoiding-malware.html>
 - 509. <http://www.go2android.de/android-fuer-anfaenger-erklaert-teil-10-app-berechtigungen-und-was-sie-bedeuten/>
 - 510. <http://android.izysoft.de/aplists/perms>
 - 511. http://developer.android.com/reference/android/Manifest.permission_group.html

Dinge, oder „NETWORK“ mit dem Netzwerk-Zugriff zusammenhängende Permissions.

Protection Level

Eine weitere Einteilung stellen die so genannten *Protection Level* dar (siehe [Developers Reference: Permission Element](#)⁵¹²). Fünf verschiedene Ebenen legen dabei fest, wie mit den entsprechenden Permissions umgegangen wird (die Kürzel in Klammern sind für die Referenz in der eigentlichen Permissions-Tabelle):

- **normal (no):** niedriges Risiko bzw. „Standard“. Diese Permissions werden dem Anwender bei der Installation nicht extra „vorgeführt“, sondern stillschweigend akzeptiert.
- **dangerous (da):** höheres Risiko: Zugriff auf persönliche Daten, oder Kontrolle über das Gerät mit potentiell negativen Impakt für den User. Das sind die Permissions, die bei der Installation vom Anwender explizit bestätigt werden müssen.
- **signature (si):** Der Herausgeber des Zertifikats muss mit dem des ROM-Zertifikats übereinstimmen. In der Regel heißt dies: Eine solche App muss vom gleichen Hersteller stammen.
- **signatureOrSystem (sy):** Wie „Signature“; alternativ darf es aber auch eine beliebige „System-App“ sein. Da man eine solche nur mit root-Rechten installieren kann, ist dies für „normale Anwender“ eher uninteressant.
- **development (dv):** Das ROM muss mit einem Entwickler-Key signiert sein. Diese Permissions sind eigentlich nur für Entwickler/Entwicklung gedacht. I. d. R. scheinbar als „developmentOrSignatureOrSystem“ zu verstehen.



Developers
Reference:
Permission Element

Permissions

Auch hier wieder ein Auszug einiger geräublicherer Kandidaten. Nochmals kurz zur Erklärung: „PL“ bezeichnet den gerade beschriebenen [Protection Level](#). „Risk“ steht für das mit der jeweiligen Permission verbundene potentielle Risiko, sofern sich das grob abschätzen lässt:

- 0 = niedrig
- 1 = moderat
- 2 = medium

512. <http://developer.android.com/guide/topics/manifest/permission-element.html>

- 3 = hoch
- 4 = sehr hoch
- 5 = kritisch

Dieses Risiko muss jedoch auch immer im Zusammenhang mit dem Protection Level gesehen werden: Von einer „kritischen Permission“ ist der Normal-Anwender (ohne root) beispielsweise überhaupt nicht betroffen, wenn der Protection Level „Signature“ bzw. „SignatureOrSystem“ heißt (es sei denn, es handelt sich um eine vorinstallierte System-App). Gelegentlich ändert sich eine solche Zuordnung mit neueren Android-Versionen (Beispiel: READ_LOGS wanderte ab Android 4.1 nach „sy“). Quelle für die Risiko-Klassen ist hier übrigens der bereits genannte [Artikel bei AndroidForums.Com](#)⁵¹³, die meisten Protection-Level finden sich in einem [Blog-Eintrag](#)⁵¹⁴ zugeordnet. Folgende Tabelle beschreibt einige der häufiger verwendeten Permissions:

Permission	PL	Risk	Erklärung
ACCESS_COARSE_LOCATION	da	?	<i>Ungefährer (netzwerkbasierter) Standort.</i> Hier kommt kein GPS zum Einsatz, sondern die Informationen von Funkmasten (Cell-ID) sowie WLANs
ACCESS_FINE_LOCATION	da	?	<i>Genauer (GPS-) Standort.</i> Exakte, per GPS ermittelte Standortdaten.
ACCESS_NETWORK_STATE	no	?	<i>Netzwerkstatus anzeigen.</i> Informationen über Netzwerke (besteht eine Verbindung, und wenn ja zu welchem Netzwerk?)
ACCESS_WIFI_STATE	no	?	<i>WLAN-Status anzeigen.</i> Informationen über WLAN-Netzwerke (besteht eine Verbindung, und wenn ja zu welchem Netzwerk? Welche Netzwerke sind verfügbar?)
ACCOUNT_MANAGER	sy	?	<i>Als Konto-Manager fungieren.</i> App darf mit Konto-Authentifizierern interagieren. (für Systemanwendungen reserviert).
AUTHENTICATE_ACCOUNTS	da	4	<i>Als Kontoauthentifizierer fungieren.</i> Konto-Authentifizierungsfunktionen verwenden, Konten erstellen, Abrufen und Einstellen der zugehörigen Passwörter. In der Regel stellt eine solche App eine Schnittstelle zu einem neuen Dienst bereit, der nicht von Haus aus in Android integriert ist (Beispiel: Dropbox) – implementiert also die Art und Weise, wie bei diesem die Anmeldung funktioniert. Darüber hinaus kann die App u. U. auch einschränken, was eine aufrufende App mit dem Zugang anstellen darf.
BLUETOOTH	da	?	<i>Bluetooth-Verbindungen herstellen.</i> Zugriff auf bereits „autorisierte“ Bluetooth-Geräte

513. <http://androidforums.com/android-applications/36936-android-permissions-explained-security-tips-avoiding-malware.html>

514. <http://rupertrawsley.blogspot.de/2011/11/android-permissions-protection-levels.html>

Permission	PL	Risk	Erklärung
BLUETOOTH_ADMIN	da	2	<i>Bluetooth-Verwaltung.</i> Bluetooth-Geräte „autorisieren“ (also „pairen“ und so). Eine mit dieser Permission ausgestattete App darf selbstständig Bluetooth-Verbindungen aufbauen und etablieren – auch zu „wildfremden“ Geräten.
CALL_PHONE	da	3	<i>Telefonnummern direkt anrufen.</i> Anruf ohne Bestätigung durch den Anwender tätigen. Wird z. B. für Kontakt-Widgets benötigt, wenn ein „Tapp“ auf selbige direkt einen Anruf auslösen soll – macht aber bei einer Mal-App herzlich wenig Sinn. Gilt nicht für Notruf-Nummern.
CALL_PRIVILEGED	sy	?	<i>Alle Telefonnummern direkt anrufen.</i> Wie CALL_PHONE, aber inklusive Notruf-Nummern („Hallo, Polizei – Anwender ist gerade in Bank eingebrochen. Bitte Fußboden reparieren ...“)
CAMERA	da	1-3	<i>Fotos aufnehmen.</i> Vollzugriff auf die Kamera. Nebenwirkung: Diese App lässt sich nicht auf Geräten installieren, die über keine Kamera verfügen. Die API-Referenz schreibt sinngemäß: „Wenn die App auch ohne Kamera bedienbar ist, diese Permission nicht anfordern.“ Da sei die Frage erlaubt: Braucht man sie dann überhaupt, wenn es auch ohne geht?
CHANGE_NETWORK_STATE	da	?	<i>Netzwerkkonnektivität ändern.</i> Netzwerk-Status ändern (also z. B. Verbindung trennen)
CHANGE_WIFI_MULTICAST_STATE	da	?	<i>WLAN-Multicast-Empfang zulassen.</i> WLAN MultiCast aktivieren. Damit können Datenpakete an mehrere Empfänger zeitgleich verschickt werden, ohne dass dies zusätzliche Bandbreite erfordert. Macht z. B. Sinn bei einem Streaming-Server, der mehrere Clients bedient („Radio“). Gleichzeitig ermöglicht dies auch den Empfang von Netzwerk-Paketen, die nicht an das eigene Gerät gerichtet sind (Netzwerk-Sniffer).
CHANGE_WIFI_STATE	da	?	<i>WLAN-Status ändern.</i> CHANGE_NETWORK_STATE für WLAN. Kann auch Änderungen an konfigurierten WLAN-Netzen vornehmen.
CLEAR_APP_USER_DATA	sy	?	<i>Alle Cache-Daten der Anwendung löschen.</i> Benutzerdaten beliebiger/aller Apps löschen (siehe Einstellungen > Anwendungen verwalten, der Button „Daten löschen“ bei jeder Anwendung) – richtiger wäre also <i>Anwendungsdaten</i> löschen oder CLEAR_APP_DATA, das „User“ verwirrt hier ein wenig.
DISABLE_KEYGUARD	da	2-3	<i>Tastensperre deaktivieren.</i> Tastensperre (inkl. deren Passwort-Schutz) deaktivieren, sodass der Bildschirm nicht mehr automatisch gesperrt wird.

Permission	PL	Risk	Erklärung
			Sinnvoll z. B. bei Video-Apps und insbesondere bei Navis – und bei eingehenden Telefonaten.
EXPAND_STATUS_BAR	no	2-3	<i>Statusleiste ein-/ausblenden.</i> Status-Bar (Notification?) erweitern/kollabieren. Wohl die Lite-Version von STATUS_BAR
GET_ACCOUNTS	no	?	<i>Bekannte Konten suchen.</i> Liste konfigurierter Accounts abrufen (nur die Accounts, nicht die Zugangsdaten selber). Mit dieser Permission lässt sich lediglich feststellen, welche Accounts existieren. So kann beispielsweise eine App, die Dropbox verwenden möchte, feststellen, ob bereits ein passendes Zugangskonto eingerichtet ist.
GET_TASKS	da	2-3	<i>Laufende Anwendungen abrufen.</i> Informationen über laufende Anwendungen abrufen. Wird natürlich von Task-Managern und -Killern, aber auch von Akku-Statistik-Apps benötigt. „Böse Apps“ können dies nutzen um auszukundschaften, wo sich lohnende Daten zum Klauen finden lassen.
INSTALL_PACKAGES	sy	?	<i>Anwendungen direkt installieren.</i> Andere Apps installieren. Kann OK sein (App-Manager), muss aber nicht (Wallpaper etc. wollen vielleicht eher Schadsoft nachladen, wenn sie diese Permission anfordern)
INSTALL_SHORTCUT	?	1-3	<i>Verknüpfungen auf dem Homescreen erstellen.</i> Malware nutzt dies, um unerwünschte Dinge dort abzulegen. Beispielsweise ein Icon, dass wie das vom Playstore aussieht – allerdings ganz woanders hinführt.
INTERNET	da	?	<i>Uneingeschränkter Internetzugriff.</i> Öffnen von Netzwerk-Sockets. Die App kann also beliebige Internet-Verbindungen herstellen. Wird von allen Apps gebraucht, die Werbung anzeigen wollen.
KILL_BACKGROUND_PROCESSES	no	3	<i>alle Anwendungen im Hintergrund schließen.</i> Hintergrund-Prozesse „töten“, also beenden. Dabei kann es sich um die eigenen Prozesse handeln (was dem Anwender die Möglichkeit gibt, das Programm tatsächlich zu beenden, statt es nur in den Hintergrund zu schieben) – es können aber eben so gut fremde Prozesse beendet werden. i. d. R. handelt es sich dann um einen Task-Manager oder Task-Killer . Eine böswillige App könnte dies jedoch auch nutzen, um Schutzmechanismen auszuhebeln (etwa eine Anti-Malware-App abzuschießen, bevor sie mit ihrem eigentlichen Unwesen beginnt).
MANAGE_ACCOUNTS	da	?	<i>Als Konto-Manager fungieren.</i> Accounts/Zugangsdaten verwalten – also auch verändern. Die

Permission	PL	Risk	Erklärung
			Doku ist leider wieder einmal sehr vage. Laut einem Post bei Stack Exchange ⁵¹⁵ heißt dies jedoch nur, dass die betreffende App ihre eigenen Konten (nicht aber andere) mit Unterstützung des (system-eigenen) Account-Managers verwalten darf.
MODIFY_PHONE_STATE	sy	?	<i>Telefonstatus ändern.</i> Status der Telefonie anpassen: Power, MMI-Codes (z. B. Rufumleitung [de]aktivieren, Rufnummernübermittlung ein/ ausschalten) etc. – jedoch nicht Anrufe tätigen. Allerdings kann das Netzwerk (zu einem anderen Anbieter, Roaming) gewechselt oder die Mobilfunkverbindung ein- bzw. ausgeschaltet werden, ohne dass der Benutzer davon informiert wird. Auch können mit dieser Permission eingehende Anrufe abgefangen werden.
READ_ATTACHMENT	da	3	<i>Attachments lesen.</i> Bezieht sich auf Dateianhänge in E-Mails der Stock-Mail-App (trifft also weder auf die GMail-App, noch auf K-9 Mail zu). Dateianhänge können sensible Informationen enthalten, die eine böswillige App an andere Stellen weiterleiten könnte.
READ_CALENDAR	da	2	<i>Kalenderdaten lesen.</i> Sollte klar sein: Alle Termine können damit gelesen werden.
READ_CONTACTS	da	2-3	<i>Kontaktdaten lesen.</i> Damit ist das Adressbuch fällig. Was damit alles einsehbar ist, verrät die App permission.READ_CONTACTS ⁵¹⁶ .
READ_HISTORY_BOOKMARKS	da	2-3	Erlaubt lesenden Zugriff auf Lesezeichen und Browserverlauf (Chronik). Was das im Einzelnen bedeutet, lässt sich mit der READ_HISTORY_BOOKMARKS App ⁵¹⁷ ermitteln.
READ_OWNER_DATA	da	?	<i>Eigentümerdaten lesen.</i> Auslesen der auf dem Gerät gespeicherten Eigentümerdaten.
READ_PHONE_STATE	da	?	<i>Telefonstatus lesen und identifizieren.</i> Zugriff auf die Telefonfunktionen des Gerätes. Eine Anwendung erhält mit dieser Berechtigung unter anderem die Möglichkeit, die Telefon- und Seriennummer des Telefons zu ermitteln. Um festzustellen, ob ein Anruf aktiv ist, wird sie jedoch trotz gegenteiliger Behauptungen keineswegs benötigt, wie die App permission.READ_PHONE_STATE ⁵¹⁸ zeigt. Bei



Stack Exchange:
What does
permission
.MANAGE_ACCOUNTS
mean?

515. <http://android.stackexchange.com/a/44295/16575>

516. <http://www.1mobile.co.id/permission-read-contacts-164022.html>

517. <http://www.1mobile.co.id/read-history-bookmarks-app-254112.html>

Permission	PL	Risk	Erklärung
			Werbung (z. B. AdMob) wird dies häufig zum Auslesen der IMEI/IMSI genutzt um festzustellen, welche Werbung auf dem Gerät bereits angezeigt wurde (eindeutige Identifizierung, Tracking – seit August 2014 darf die IMEI laut Playstore-Richtlinien dafür nicht mehr verwendet werden, und schon stoßen auch etliche Apps diese Permission ab ...). Apps, die auch für Android 1.6 und früher kompatibel sein sollen, wird diese Permission automatisch gesetzt.
READ_PROFILE	da	2-3	<i>Das persönliche Profil des Anwenders lesen.</i> Bezieht sich auf den neuen „Me“ Kontakt für das eigene Profil. Macht beispielsweise Sinn für Messenger-Apps, die den Diskussionsverlauf darstellen.
READ_SMS	da	1-3	<i>SMS oder MMS lesen.</i> Damit lassen sich bereits gespeicherte Kurznachrichten lesen. Darunter können natürlich auch vertrauliche Informationen sein.
READ_SOCIAL_STREAM	da	3	<i>Den Social-Media-Stream lesen.</i> Diese Permission wurde mit Android 4.0 eingeführt. Eine damit ausgestattete App kann Status-Updates der sozialen Netzwerke lesen – sowohl eingehende, als auch ausgehende.
RECEIVE_BOOT_COMPLETED	no	1-3	<i>Automatisch nach dem Booten starten.</i> App möchte benachrichtigt werden, wenn der Bootvorgang abgeschlossen ist. i. d. R. heißt das: Sie möchte nach dem Booten automatisch gestartet werden.
RECEIVE_MMS, RECEIVE_SMS	da	3	<i>MMS empfangen, SMS empfangen.</i> Eingehenden MMS/SMS abfangen – da möchte wohl jemand mitlesen. Kann aber durchaus OK sein, wenn die App auf MMS/SMS reagieren soll. Auf der anderen Seite kann man damit eingehende Nachrichten auch „im Nirvana“ verschwinden lassen.
RECORD_AUDIO	da	1-3	<i>Audio aufnehmen.</i> Tonaufnahmen erstellen. Das kann sowohl für ein „Diktaphon“ genutzt werden – als auch zum Mitschneiden von Telefonaten.
SEND_SMS	da	3	<i>Kurznachrichten senden.</i> Und zwar ohne Zutun des Benutzers, auch an richtig teure Premium-Dienste (womit klar ist, wozu „böse Apps“ das gern hätten). Es gibt aber auch „gute“ Gründe für diese Permission: Natürlich die SMS-Apps, aber teilweise auch In-App-Käufe, die nicht über Google Checkout abgewickelt werden.
STATUS_BAR	sy	?	Kann die Status-Bar (Notification?) öffnen, schließen, und ausblenden. Meist will die App wohl letzteres, um einen „Vollbild-Modus“ zu ermöglichen.

Permission	PL	Risk	Erklärung
SYSTEM_ALERT_WINDOW	da	3	<i>Warnungen auf Systemebene anzeigen.</i> Fenster mit Systemwarnungen einblenden. Eine böswillige App kann so den gesamten Bildschirm blockieren. Sollte eigentlich nicht benutzt werden, da dies für System-Meldungen gedacht ist. Erlaubt das anzeigen von „Alert Windows“, d. h. Nachrichtenfenstern, die immer im Vordergrund angezeigt werden.
USE_CREDENTIALS	da	?	<i>Authentifizierungsinformationen eines Kontos verwenden.</i> Möchte die konfigurierten Zugangsdaten verwenden. Das heißt nicht unbedingt, dass es sie „zu sehen bekommt“ – aber die App kann sich quasi „im Namen des Anwenders“ anmelden. Beim ersten Mal wird der Anwender jedoch vom Account Manager gefragt, ob er dies zulassen möchte.
USE_SIP	da	2-3	App kann Internet-Telefonie nutzen (SIP ⁵¹⁹ ist das Protokoll dafür)
WAKE_LOCK	no	?	<i>Standby-Modus deaktivieren.</i> App kann das System daran hindern, einen Ruhezustand einzunehmen (also den Bildschirm zu dimmen, die CPU „schlafen“ zu lassen, etc.) Wäre doch blöd, wenn die Navi-App läuft und plötzlich der Bildschirm ausgeht.
WRITE_APN_SETTINGS	sy	?	<i>Einstellungen für Zugriffspunktname schreiben.</i> App kann die Zugangsdaten zum Internet etc. (siehe APN) verändern. Meist geht es der App nur darum, den Namen des APN zu ändern – um die Verwendung des mobilen Datenverkehrs zu steuern (Beispiel: APNAndroid ⁵²⁰).
WRITE_CALENDAR	da	2	<i>Kalenderdaten schreiben.</i> Diese Permission erlaubt lediglich den Schreib-, nicht aber den Lesezugriff auf den Kalender. Die damit versehene App kann also Termine hinzufügen, nicht aber lesen oder ändern.
WRITE_CALL_LOG	da	2-3	<i>Anruflisten schreiben.</i> Macht sicher Sinn für VoIP- und Backup-Apps. Andere Apps sollten hier jedoch nichts verloren haben.



Wikipedia: Session Initiation Protocol



APNAndroid

519. http://de.wikipedia.org/wiki/Session_Initiation_Protocol

520. <https://play.google.com/store/apps/details?id=com.codecarpet.apnandroid.pro>

Permission	PL	Risk	Erklärung
WRITE_CONTACTS	da	1-3	<i>Kontaktdaten schreiben.</i> Wie WRITE_CALENDAR, nur in Bezug auf die Kontaktdaten bzw. Browser-History (Chronik) und Lesezeichen.
WRITE_HISTORY_BOOKMARKS	da	2-3	
WRITE_EXTERNAL_STORAGE	da	2	App darf beliebige Daten auf der (externen) SD-Karte lesen, schreiben, verändern und auch löschen – prinzipiell auch die Daten anderer Apps. Diese Permission ist aber beispielsweise essentiell für diverse Backup- und Kamera-Apps, die natürlich Daten auf der Karte manipulieren müssen. Warnung: Apps, die für Android 1.5 oder älter geschrieben wurden, erhalten diese Permission implizit!
WRITE_OWNER_DATA	da	?	<i>Eigentümerdaten schreiben.</i> Schreiben/verändern der auf dem Gerät gespeicherten Eigentümerdaten.
WRITE_PROFILE	da	1-3	<i>Das persönliche Profil des Anwenders schreiben.</i> Gegenstück zu READ_PROFILE.
WRITE_SECURE_SETTINGS	dv	4	<i>Allgemeine Systemeinstellungen ändern.</i> Lesen und Schreiben von Systemeinstellungen. „Secure Settings“ können nur von Systemanwendungen (also solchen, die ins „ROM“ integriert wurden) angefordert werden.
WRITE_SETTINGS	da	2	
WRITE_SMS	da	3	<i>SMS Nachrichten schreiben</i> (jedoch nicht senden). Eine böswillige App könnte darauf hoffen, dass der Anwender die „ungesendete Nachricht“ dann doch noch auf den Weg bringt.
WRITE_SYNC_SETTINGS	da	2	<i>Synchronisierungseinstellungen schreiben.</i> Schreibzugriff auf die Einstellungen der Synchronisation. Eine mit dieser Permission ausgestattete App kann u. a. die Synchronisation von Kontakten und Kalendern aktivieren bzw. deaktivieren. Sinn macht so etwas u. U. bei einer SMS Backup App.
com.android.vending.BILLING	?	5	In-App Payment (in der App integrierte Bezahldienste, die über den <i>Play Store</i> abgewickelt werden)

Wer noch immer „Bahnhof“ versteht, dem sei ein kleines [Tutorial bei N-Droid.DE](#)⁵²¹ empfohlen. Etwas tiefgreifender wird das Thema [bei LifeHacker](#)⁵²² beschrieben. Für Anfänger gut verständlich, sofern sie keine Probleme mit der englischen Sprache haben – definitiv empfohlene Lektüre!



N-Droid.DE: Tutorial App-Berechtigungen



LifeHacker: Why Does This Android App Need So Many Permissions?

521. <http://www.n-droid.de/android-tutorial-app-berechtigungen-verstehen-und-kontrollieren-zum-teil-root-benötigt.html>

522. <http://lifehacker.com/5991099/>

Secret Codes oder Magische Nummern

Klar kann man mit einem Telefon telefonieren. Dazu gibt man eine Ziffernfolge ein, und drückt die Taste für „Abheben“. Was aber passiert, wenn man noch ein paar Sonderzeichen hinzufügt?



Wikipedia: GSM-Code

Beschränken wir uns dabei auf die Zeichen # und * erhalten wir – richtig kombiniert – so genannte [GSM-Codes](#)⁵²³. Der verlinkte Wikipedia-Artikel beschreibt ganz gut, worum es dabei geht (kurz gefasst: Steuer-Codes für diverse Netzanbieter-Funktionen, die eigentlich auf allen Geräten gleich funktionieren sollten, aber nicht bei allen Anbietern in gleichem Umfang verfügbar sind). Unterteilen lassen sich diese Codes grob in mehrere Untergruppen:

[USSD-Codes:](#)⁵²⁴

Diese folgen dem Muster *1nn# – also der * Taste, gefolgt von einer 1 und weiteren zwei Ziffern, abgeschlossen durch eine Raute (optional mit Parametern: *1nn*<Parameter>#). Dabei handelt es sich um Zugangsnummern für einfache Mobilfunkdienste, die zum Beispiel Zugang zu vorkonfigurierten Services bereitstellen, welche für den Betreiber des jeweiligen Mobilfunknetzes spezifisch sind. Gibt man einen solchen USSD-Code auf dem Mobilfunk-Gerät ein, wird die Antwort des Betreibernetzes normalerweise innerhalb weniger Sekunden auf dem Bildschirm dargestellt.

[Erweiterte Service-Codes:](#)⁵²⁵

Diese werden für erweiterte Service-Dienste wie etwa Anrufweiterleitungen oder die (De-)Aktivierung der Rufnummernübermittlung, aber auch zum Ändern bzw. Entsperren der PIN genutzt.

Geräte-, Hersteller- und Systemspezifische Codes:

Diese dienen i. d. R. dem zuständigen Service-Personal zur Abfrage/Anpassung diverser Gerät-Parameter, für Status-Tests, u. a. m.

Disclaimer: Naturgemäß kann es sein, dass einige dieser Codes nicht auf allen Geräten bzw. nicht bei allen Mobilfunkanbietern funktionieren. Sofern sie nicht im Handbuch des Gerätes aufgeführt sind, mag das durchaus seine Gründe haben: So kann der eine oder andere Punkt, unsachgemäß angewendet, u. U. Schäden verursachen. Ich übernehme keinerlei Garantien dafür, dass folgende Codes a) funktionieren, b) das tun, was da steht, oder c) „folgenfrei“ genutzt

523. <http://de.wikipedia.org/wiki/GSM-Code>

524. http://de.wikipedia.org/wiki/Unstructured_Supplementary_Service_Data

525. http://en.wikipedia.org/wiki/Supplementary_service_codes

werden können. Insbesondere übernehme ich keinerlei Verantwortung für etwaige negative Folgen! Für etwaige positive Folgen stelle ich natürlich gern mein Bankkonto zur Verfügung.

Ebenso kann es vorkommen, dass nach Eingabe des einen oder anderen Codes nichts passiert. Es kann aber auch sein, dass dies nur so scheint (bei meinen Tests fand ich anschließend – am nächsten Tag – einige der „magischen Nummern“ in der „Verbraucherliste“ wieder). Es können durchaus Hintergrund-Dienste gestartet oder aber „versteckte Klassen/Menüs“ in Apps freigeschaltet werden, was sich u. U. nur durch einen Werksreset wieder rückgängig machen lässt.

USSD Codes

Code	Bedeutung
*100#	Prepaid-Guthaben anzeigen
*135#	Eigene Rufnummer anzeigen

Erweiterte Service-Codes

Code	Bedeutung
**21*<Rufnummer># / *21# / #21# / *#21#	Anrufweiterleitung einrichten / aktivieren / deaktivieren / überprüfen
##002#	alle Rufumleitungen deaktivieren
*30# / #30# / *#30#	CLIP: Eingehende Rufnummern anzeigen / unterdrücken / Status
#31#<Rufnummer>	CLIR: Mit unterdrückter Rufnummer anrufen
*43# / #43# / *#43#	Anklopfen aktivieren / deaktivieren / Status abfragen
*76# / #76# / *#76#	COLP: Wenn der ausgehende Anruf weitergeleitet wird, Zielrufnummer anzeigen
*77# / #77# / *#77#	COLR: Bei eingehendem umgeleiteten Anruf die Ursprungsnummer anzeigen
*N# (TCom: *T#)	Wird eine SMS mit dieser Zeichenfolge begonnen, erfolgt eine SMS Empfangsbestätigung

Obige Tabelle ist keinesfalls vollständig. Weitere „GSM Codes“ finden sich u. a. bei:

- [GSMCodes-Online.DE](http://www.gsmcodes-online.de/)⁵²⁶



GSMCodes-
Online.DE

- [Inside-Handy.DE](#)⁵²⁷ (einschließlich Tipps zu versteckten Menüs)



Geräte-, Hersteller-, und Systemspezifische Codes

Die fett gedruckten Codes sollten nach Informationen bei [Stack Exchange](#)⁵²⁸ auf allen Android-Geräten (zumindest mit Android Version 4.1) gleichermaßen funktionieren. Ob das „Spielen“ damit deswegen unbedingt ratsam ist (zumal Wenn man nicht genau weiß, was sich dahinter verbirgt), steht auf einem anderen Blatt. Daher habe ich die Codes, die mir als „absolut harmlos“ bekannt sind, einmal kursiv hervorgehoben.



Die ursprünglich hier stehende längere Liste findet sich jetzt [bei IzzyOnDroid](#)⁵²⁹.

Code	Bedeutung
*#0011#	GSM-Infos
*#0228#	ausführliche Infos zum Batteriestatus
**05*<PUK Code>*<neue PIN>*<neue Bestätigung>#	Ent sperren des Telefons aus dem Notruf-Modus
*#06#	IMEI anzeigen
###1234##*	Firmware-Info
###197328640##*	Service-Menü mit verschiedenen Test-Möglichkeiten
###225##*	Kalender-Debug
###232337##	MAC-Adresse des BlueTooth-Interfaces anzeigen
###232338##*	MAC-Adresse des WLAN-Interfaces anzeigen
###2432546##*(*###CHECKIN#*#*)	Nach OTA -Updates suchen
###4636##*(*###INFO#*#*)	Öffnet ein Menü, aus dem sich wählen lässt: Telefon-Infos, Akku-Infos, Akku-History, Verbrauchsstatistiken.
###7378423##*	Ein weiteres Service-Menü: Service-Information, Service-Settings, Service-Tests...
*#7465625#	Netz- und Subnetsperre, Simlook, Service Provider und Corporate Lock (Galaxy-S?)
###759##*	GooglePartnerSetup

527. <http://www.inside-handys.de/news/28960>

528. <http://android.stackexchange.com/q/37020/16575>

529. <http://android.izzysoft.de/books.php?topic=secretcodes>



ASE: What are
Androids secret
telephony codes

IzzyOnDroid: Secret
Codes oder
Magische Nummern

Code	Bedeutung
###7594#*#*	Verhalten des Einschalt-Knopfes ändern (z. B. direktes Abschalten ohne Menü)
###7780#*#*	Zurücksetzen auf Werkseinstellungen

Leistungsaufnahme verschiedener Komponenten

Heise hat für seinen Artikel [Energiesparplan](http://www.heise.de/-1145579)⁵³⁰ ein Motorola Milestone angepasst und ausführlich getestet, was welche Komponente so verbraucht. Und wie man das Einschränken kann (das war jetzt eine klare Empfehlung, den Artikel zu lesen!). Die Daten davon werden einerseits ein „war ja klar“, aber bei einigen Daten auch ein „oh, das hätte ich jetzt nicht gedacht“ hervorrufen. Passende Angaben zum Galaxy S3 hat die c't in ihrer [Ausgabe vom August 2012](http://www.heise.de/artikel-archiv/ct/2012/17/124_kiosk)⁵³¹ gesammelt – und dabei den Test gleich noch ein wenig ausgeweitet.

Anmerkungen zum Galaxy S3 (*): Der Energieverbrauch des Displays hängt hier auch stark von den dargestellten Inhalten ab, was auf das verwendete Amoled-Display zurückzuführen ist. Bei vollständig schwarzem Display entspricht der Verbrauch auf allen Stufen nicht mehr als dem Minimum.

Weiterhin sollte man beim Vergleich der Werte auch die unterschiedliche Hardware-Ausstattung im Hinterkopf behalten: So werkelt beispielsweise im Galaxy S3 ein mit 1,4 GHz getakteter Quad-Core Prozessor, und zur Anzeige dient ein 4,8 Zoll Display – im Milestone waren es noch eine Single-Core CPU mit 550 MHz und ein 3,7 Zoll Display.



Tipps und Tools für
eine bessere
Akkulaufzeit unter
Android



c't August 2012:
Durchhaltetraining

Betriebszustand	zusätzliche Leistungsaufnahme	
	Motorola Milestone	Samsung Galaxy S3
Videoaufnahme1	1557 mW	1683 mW
UMTS Upload	1410 mW	1033 mW
UMTS Download	1349 mW	1074 mW
EGDE Upload	1179 mW	
WLAN Download	1158 mW	549 mW
Video abspielen (fullscreen)1	1135 mW	597 mW
UMTS-Telefonat	983 mW	637 mW
Kamera1	934 mW	1460 mW
EGDE Download	853 mW	
Bluetooth empfangen	751 mW	487 mW
Display (höchste Stufe)	730 mW	1568 mW*
GPS Suche	550 mW	263 mW

530. <http://www.heise.de/-1145579>

531. http://www.heise.de/artikel-archiv/ct/2012/17/124_kiosk

Betriebszustand	zusätzliche Leistungsaufnahme	
	Motorola Milestone	Samsung Galaxy S3
GSM-Telefonat	511 mW	297 mW
Bluetooth senden	487 mW	454 mW
WLAN Upload	479 mW	488 mW
Display (niedrigste Stufe)	310 mW	567 mW
WLAN Tether 2		372 mW
MP3 abspielen über Bluetooth		296 mW
MP3 abspielen	160 mW	153 mW
UMTS Standby	18,3 mW	13,8 mW
GSM/EDGE Standby	11,6 mW	9,5 mW
WLAN Standby 2,4 GHz	7,8 mW	9,3 mW
WLAN Standby 5 GHz	N/A	14,6 mW
NFC Standby	N/A	4 mW
Bluetooth Standby	2,8 mW	1,8 mW
GPS Standby	0,4 mW	0,7 mW
WLAN Tether Download ³		1254 mW

1 Leistungsaufnahme des Displays bereits abgezogen

2 Tethering aktiv, 1 Benutzer

3 Download vom Notebook per WLAN-Tether

Für die Gesamt-Leistungsaufnahme muss natürlich noch die Grundlast hinzugerechnet werden (was das Gerät verbraucht, wenn alles in der Tabelle genannte abgeschaltet ist; also „Flugmodus“). Das wären ganze fette 6,4mW. Wird das Gerät also des Nachts in diesen versetzt, spart das bereits enorm (Werte des Motorola Milestone):

Betriebszustand	zusätzliche Leistungsaufnahme	
	Motorola Milestone	Samsung Galaxy S3
Flugmodus	6,4 mW	6,4 mW
GSM Bereitschaft	18 mW	9,5 mW
GSM Bereitschaft + WLAN Standby	25,8 mW	18,8 mW
GSM Bereitschaft + WLAN Standby + Bluetooth Standby	28,6 mW	20,6 mW
UMTS Bereitschaft	24,7 mW	10,9 mW
UMTS Bereitschaft + mobile Daten aktiv		13,8 mW
UMTS Bereitschaft + WLAN Standby	32,5 mW	20,2 mW
UMTS Bereitschaft + WLAN Standby + Bluetooth Standby	35,3 mW	22,0 mW

Die Werte sind natürlich alle spezifisch für o. g. Motorola Milestone bzw. das Samsung Galaxy S3, und können auf anderen Geräten abweichen. Die Größenordnungen sollten aber zumindest ähnlich sein.

Nur so nochmal gesagt: Selbst wenn WLAN etc. alles aus sind, und nur Telefonate (und SMS) noch durchkommen (den Datenverkehr also mal ganz außen vorgelassen), reduziert sich der Verbrauch im Flugmodus auf ein Drittel (GSM) oder gar ein Viertel (UMTS). Einmal 6 Stunden angenommen (von 0 Uhr bis 6 Uhr), hält der Akku damit (theoretisch) für bis zu gut 2 Stunden länger. Natürlich wieder nur, wenn man dann anschließend auch nichts damit macht – also wieder so ein „Laborwert“. Trotzdem kann man sich leicht ausrechnen: Sechs Stunden Flugmodus (statt GSM Bereitschaft) schaffen Raum für zusätzliche ca. 8 Minuten GSM Telefonat...

Begriffserklärungen

An dieser Stelle möchte ich einige Begriffe kurz erklären, da ich danach des Öfteren gefragt wurde. Wie gewohnt, versuche ich mich dabei kurz zu fassen – und verweise für Details auf „externe Quellen“.

2G

Gemeint ist damit die Datenübertragung der „zweiten Generation“ (die erste ist bereits nicht mehr verfügbar). Hierzu gehören sowohl GPRS als auch EDGE.

3G

Da dies ein Android-Handbuch ist, denken wir jetzt mal nicht an das iPhone 3G. Obwohl der Zusatz auch hier bedeutet: Dritte Generation. Gedacht ist in unserem Fall jedoch an die Datenübertragung mittels UMTS bzw. CDMA.

ADB

Die **Android Debug Bridge** ist Bestandteil des Android-SDK. Anders als der Name es nahelegt, ist ADB für mehr als nur das Debuggen gut. so lässt sich hiermit ein Android-Gerät steuern und kontrollieren, Dateien können übertragen, installiert oder auch gelöscht werden, und mehr. Details finden sich im Kapitel [ADB: Die Android Debug Bridge](#).

Android

Hierfür zitiere ich einmal kurz das Wesentliche aus dem passenden [Wikipedia-Artikel](#)⁵³²:

Android ist ein Betriebssystem wie auch eine Software-Plattform für mobile Geräte wie Smartphones, Mobiltelefone, Netbooks und Tablets, die von der Open Handset Alliance⁵³³ entwickelt wird. Basis ist der Linux-Kernel 2.6. Android ist freie Software und quelloffen.

[...]



Wikipedia: Open Handset Alliance

Als erstes Gerät mit Android als Betriebssystem kam am 22. Oktober 2008 das HTC Dream unter dem Namen T-Mobile G1 in den Vereinigten Staaten auf den Markt. Dass bereits dieses erste Gerät auf das Global Positioning System zugreifen konnte und mit

532. http://de.wikipedia.org/wiki/Android_%28Betriebssystem%29

533. http://de.wikipedia.org/wiki/Open_Handset_Alliance

Bewegungssensoren ausgestattet war, gehörte zum Konzept von Android.

Also grob zusammengefasst: Speziell für mobile Geräte – Linux unten drunter, „eine Art Java“ obendrauf, und ganz zuoberst laufen unsere lieben Apps.

Android Versionen

Der kleine Andy (der grüne Roboter Androide) ist ein Süßer. Entsprechend hat Google auch die Namen der Android-Versionen recht süß gewählt. Kommen also entsprechende Kuchen-Varianten ins Spiel, zielt das i. d. R. auf eine Android-Version:

Version	Name	wichtigste Neuerungen
1.1		(Full Version notes ⁵³⁴)
1.5	Cupcake	AutoRotate, Bildschirmtastatur, Videos (Full Changelog ⁵³⁵)
1.6	Donut	VPN, Verbesserungen bei der Energieverbrauchssteuerung, TTS, Gesten (Full Changelog ⁵³⁶)
2.0	Eclair	Digitalzoom, Blitzlicht, Exchange, Bluetooth 2.1 (Full Changelog ⁵³⁷)
2.1	Eclair	Webkit-Erweiterungen (HTML5, GeoLocation, u. a.) (Full Changelog ⁵³⁸)
2.2	Froyo (Frozen Yoghurt)	sparsamerer Kernel, mehr RAM nutzbar, Hotspot und Tethering , App2SD , Bluetooth-Sprachwahl (Full Changelog ⁵³⁹)
2.3	Gingerbread	DualCore, NFC , SIP (Full Changelog ⁵⁴⁰)
3.0	Honeycomb	Optimierungen für Tablets (Full Changelog ⁵⁴¹)
3.1	Honeycomb	USB Host-Modus (Full Changelog ⁵⁴²)
3.2	Honeycomb	Optimierungen für Tablets (Full Changelog ⁵⁴³)
4.0	Ice Cream Sandwich	Data-Tracking, Screenshots (Full Changelog ⁵⁴⁴)

534. <http://developer.android.com/about/versions/android-1.1.html>

535. <http://developer.android.com/about/versions/android-1.5.html>

536. <http://developer.android.com/about/versions/android-1.6.html>

537. <http://developer.android.com/about/versions/android-2.0.html>

538. <http://developer.android.com/about/versions/android-2.1.html>

539. <http://developer.android.com/about/versions/android-2.2.html>

540. <http://developer.android.com/about/versions/android-2.3-highlights.html>

541. <http://developer.android.com/about/versions/android-3.0-highlights.html>

542. <http://developer.android.com/about/versions/android-3.1-highlights.html>

543. <http://developer.android.com/about/versions/android-3.2.html>

544. <http://developer.android.com/about/versions/android-4.0-highlights.html>

Version	Name	wichtigste Neuerungen
4.1	Jelly Bean	Offline-Sprachsteuerung, Notification Bar Actions (Ursprung ermitteln, direkter Rückruf...) (Full Changelog⁵⁴⁵)
4.2	Jelly Bean	Daydream (interaktiver Bildschirmschoner mit API), Unterstützung für „Secondary Screens“ (gleicher Inhalt auf zwei Bildschirmen), Lockscreen-Widgets, MultiUser, Swype (Full Changelog⁵⁴⁶)
4.3	Jelly Bean	Erweiterter Bluetooth-Support (BTLE, AVRCP), Support für OpenGL ES 3.0, eingeschränkte Profile für Benutzerkonten, (versteckte) Rechte-Verwaltung für Apps (Full Changelog⁵⁴⁷)
4.4	KitKat	Reduzierter Speicherverbrauch, SMS in Hangouts integriert, Drucker-Unterstützung, Storage Access Framework, CallerID via Google, Screencast, Miracast, NFC HCE, IR (Changelog⁵⁴⁸)
5.0	Lollipop	ART statt Dalvik als Default, Support für 64-bit Architektur, Floating Notifications, beschleunigte Grafik mit OpenGL ES, Audio bis 7.1 Kanäle, RAW Kamera Format, 4K Video, Screen Recording, verbesserte Akku-Laufzeit und Statistik (Changelog⁵⁴⁹)
5.1	Lollipop	Multiple SIM Card Support, HD Voice, Captive Portal Erkennung (Changelog⁵⁵⁰)
6.0	Marshmallow	Neues Berechtigungssystem, Unterstützung für Fingerabdruckscanner, Android Pay, Multi-Fenster-Modus, Doze (verbesserte Akkuleistung), SD-Karte als interner Speicher, Unterstützung für USB Typ-C



Wikipedia: Liste von
Android-Versionen

Wen die Unterschiede zwischen den einzelnen Versionen interessieren, der schaue doch einfach mal bei [Wikipedia](#)⁵⁵¹ vorbei...

AndroidPIT



AndroidPIT

545. <http://developer.android.com/about/versions/android-4.1.html>

546. <http://developer.android.com/about/versions/android-4.2.html>

547. <http://developer.android.com/about/versions/jelly-bean.html#android-43>

548. <http://developer.android.com/about/versions/kitkat.html>

549. <http://developer.android.com/about/versions/lollipop.html>

550. <http://developer.android.com/about/versions/android-5.1.html>

551. http://de.wikipedia.org/wiki/Liste_von_Android-Versionen

[AndroidPIT](#)⁵⁵² ist eine [Community](#)⁵⁵³, in der sich Android-Nutzer austauschen. Wo sie aber auch in Blogs, Wiki, Foren und Testberichten fundierte Informationen erhalten. Kurzum: Eine Anlaufstelle für alle Fragen rund um Android. Aber was verbirgt sich hinter diesem seltsamen Namen?

Da steckt der Name „Android“ drin, den wir ja gerade geklärt haben. Bleiben die drei Großbuchstaben am Ende: PIT. Was heißen die denn nun? Programme, Informationen, Tests? Pizza Im Tiefkühlfach? (Unserem Apple-Stevie zufolge müsste das „P“ ja sicher für „Porno“ stehen – doch findet sich eh kein Regisseur, der einen solchen im Tiefkühlfach dreht). Ich bin sicher, daraus ließe sich eine Preisfrage machen – und der Gewinner bekommt dieses Buch...

Na? Erraten? Ich helfe mal ein wenig nach: Denken wir mal an Autos. Ganz schnelle. F1 (Hilfe? Nee, Formel-1 meine ich). Und genau: „Pit stop“, der Boxen-Stop. Auftanken, und weiter geht's!

Weitere Informationen gibt es u. a. in der [FAQ](#)⁵⁵⁴.



Wikipedia: Online Community



AndroidPIT FAQ

AOSP

Zwar heißt es immer, Android würde von Google entwickelt – doch so ganz korrekt ist das nicht: Hinter der Entwicklung steht das [Android Open Source Project](#). Hier laufen die Fäden der Entwicklung des Android-Systems zusammen: Interessierte Entwickler können den Sourcecode herunterladen, aber auch eigene Anpassungen einreichen. Das Projekt wacht dabei darüber, dass am Ende alles auch wieder zusammenpasst.



AOSP Homepage

API

Kurzform (oft gesprochen, wie man es schreibt – aber auch als Abkürzung buchstabiert) steht für Application Programmers Interface. Gemeint ist hier eine definierte Schnittstelle, über die Informationen bezogen werden können. Die Android-API bietet auf diese Weise z. B. Informationen über Akkustand u. a. m. So muss nicht jeder Programmierer das Rad neu erfinden.

APK-Datei

Die Abkürzung steht für Android Package, und da drin befindet sich in der Regel eine App zur Installation unter Android. Da diese Apps ja in Java geschrieben sind, verwundert es sicher nicht, dass das APK Format eine „Abwandlung“ des JAR (Java ARchive) ist, und sich somit mittels WinZip & Co. ein Blick in selbige werfen lässt...

552. <http://www.androidpit.de/>

553. <http://de.wikipedia.org/wiki/Online-Community>

554. <http://www.androidpit.de/de/android/faq>

Datei-Manager unter Android erkennen diese Packages natürlich, und bieten an, die enthaltene App zu installieren.

APN



IzzyOnDroid: APN-Einstellungen ausgewählter Netzbetreiber

Kürzel für **Access Point Name** (zu Deutsch: Name des Zugangspunkts). Gemeint ist in der Praxis mitnichten nur der Name, sondern vielmehr der komplette Datensatz. Siehe auch [mobiles Datennetz](#) für eine detailliertere Beschreibung, sowie [bei IzzyOnDroid](#)⁵⁵⁵ für eine nach Netzanbieter sortierte Liste von APN-Definitionen.

App



Wikipedia: Neudeutsch

Kurzform für *Application*. Wird auch im Deutschen („[Neudeutsch](#)⁵⁵⁶“. Applikation) verwendet, da „Anw“ einfach blöd klingt. Denn nichts anderes bedeutet das englische Wort *Application*: Anwendung.

Im Zusammenhang mit Smartphones aller „Coeur“ (also Früchte wie auch KGMs, kleine grüne Männchen) hat sich die Kurzform „App“ eingebürgert – „Application“ (oder im Deutschen „Anwendung“) wird hier eher selten verwendet.

App2SD

Das hat nix mit dem abendlichen „Jezz abba App ins Bett“ zu tun – sondern vielmehr mit der Frage: „Wie kann ich mehr Apps installieren, als in den internen Speicher passen?“. Dazu gibt es mehrere Ansätze, die unter dem Begriff „App2SD“ zusammengefasst sind:

App2SD: Mit [Froyo](#) eingeführt. *App2SD* verschiebt Teile der App auf die (einige) [Partition](#) der SD-Karte, wobei die App dies unterstützen muss. Mit Widgets klappt dies in der Regel nicht – hier kommt es zu Abstürzen, da die SD-Karten-Partition bei Anschluss an den PC via USB auf dem Androiden nicht mehr zur Verfügung steht, wenn der [USB-Massenspeicher](#)⁵⁵⁷ Modus verwendet wird. Dies war ursprünglich der Standard; ab [Android 4.0](#) wurde u. a. aus diesem Grund auf [MTP](#) umgestellt.

App2SD+: Gibt es mit einigen Custom-ROMs. Hier wird das Widget-Problem dadurch umgangen, dass eine eigene Partition auf der SD-Karte verwendet wird. Android gibt bei USB-Anschluss lediglich die erste Partition frei, die zweite mit den Apps bleibt somit unangetastet. Wie bereits geschrieben: Benötigt [root](#) und [Custom-ROM](#), wobei auch nicht jedes Custom-ROM App2SD+ anbietet (CyanogenMod zum Beispiel nicht).

555. http://android.izzysoft.de/books.php?topic=apn_list

556. <http://de.wikipedia.org/wiki/Neudeutsch>

557. <http://de.wikipedia.org/wiki/USB-Massenspeicher>

Link2SD: Im Prinzip wie App2SD+ – nur bedarf es keines Custom-ROMs: Die App wird dabei zunächst auf die zusätzliche Partition verschoben, und sodann ein sogenannter „symbolischer Link“ dorthin im internen Speicher angelegt. Somit wird die App gefunden, als wäre sie im internen Speicher – obwohl sie ganz woanders steckt... Benötigt [root](#) und eine zweite Partition auf der SD-Karte.

Appbrain

Ein alternatives „Front-End“ für den *Google Play Store*, der sowohl als App als auch als Website verfügbar ist. Nicht so bunt, dafür aber m. E. weitaus übersichtlicher. Eine ausführlichere Beschreibung findet sich im Kapitel *Playstore Alternativen* unter [Appbrain](#).

Aptoide

Ein alternativer Android-Markt, über den man Apps beziehen kann. Näher beschrieben ist er unter [Öffentliche Märkte](#).

ART

Die **Android RunTime** löst mit Android 5 [Dalvik](#) ab. Experimentell lässt sich sich bereits ab [Android 4.4](#) aktivieren, damit Entwickler ihre Apps im Vorfeld auf Kompatibilitäts-Probleme überprüfen können. Für den täglichen Einsatz sei davon jedoch (zumindest bei Android 4.4) noch abgeraten. Ab Android 5.0 wird ausschließlich ART verwendet.

Doch was ist mit *ART* denn nun anders? Es ist ein ganzes Stück schneller (bis um das Doppelte). Statt wie im Falle von *Dalvik* „Bytecode“ zu erzeugen, wird hier gleich auf nativen Code gesetzt. Das benötigt meist etwas mehr Speicherplatz (durchschnittlich etwa 25% zusätzlich), sowie ein wenig länger bei der Installation – arbeitet dafür aber performanter und ressourcenschonender.



AndroidNext: ART statt Dalvik – Experimenteller Runtime-Compiler macht Android doppelt so schnell

Wer es genauer wissen möchte, wirft am Besten einen Blick in den Artikel [ART statt Dalvik](#)⁵⁵⁸ bei AndroidNext.

Baseband

Auch „Radio-ROM“ bzw. „Radio-Image“ wird es gern genannt. Das ist quasi die eigentliche Geräte-Firmware. Hat weniger direkt mit Android, als vielmehr mit der Hardware zu tun – und initialisiert letztere, so dass sie von ersterem genutzt werden kann. Also sowas ähnliches wie das BIOS beim PC. Und genau wie dieses, befindet es sich i. d. R. auf einem separaten Chip.

Das Teil bootet also die Hardware, und übergibt dann an den eigentlichen Bootloader, der sich dann um Android kümmert. Daher muss das Radio-Image auch zum verwendeten [SPL](#) passen. Flasht man das falsche Image,

558. <http://www.androidnext.de/schwerpunkt/art-statt-dalvik-android-doppelt-so-schnell/>

hat man einen Ziegelstein (engl.: „[Brick](#)“) oder Briefbeschwerer, aber kein brauchbares Telefon mehr...

Bloatware

Vom Hersteller und/oder [Provider](#) zusätzlich auf dem Gerät fest vorinstallierte, und oftmals völlig unerwünschte (oder gar unnötige) Apps. Da diese Installation i. d. R. Im [ROM](#) erfolgt, kann Otto Normalnutzer diese Apps auch nicht einfach de-installieren (ab Android 4.0 aka [Ice Cream Sandwich](#) allerdings zumindest deaktivieren). Das Vorhandensein dieser Apps stellt für sich nicht das große Problem dar – nur lassen diese häufig Hintergrunddienste laufen, die natürlich Systemressourcen verbrauchen.

Bootloader

Sozusagen der „zweite Teil“ nach dem [Baseband](#) (daher auch „SPL“ oder „Secondary Program Loader“ genannt). Bleiben wir weiter bei den Hinke-Vergleichen, sind wir hier etwa im „Boot-Manager“ (Lilo, Grub) gelandet (davor wäre noch der „MBR“ oder „Master Boot Record“ auf dem PC – das wäre in diesem Fall der „IPL“, der „Initial Program Loader“ – der ist bei Androiden in Hardware gegossen, und daher nicht veränderbar).

Aber das wäre jetzt nur sehr grob und ungenau, denn hier steckt mehr drin: Der Android-Bootloader, sowie weitere Boot-Optionen wie das Recovery-Menü, Fastboot, u. a. m.

Boot-Loop

Eine Dauer-Boot-Schleife beim Starten des Androiden: Das Gerät fährt hoch, man sieht das Boot-Logo, eventuell sogar noch kurz den Homescreen oder Sperrbildschirm – dann wird es plötzlich Schwarz, und fängt wieder von vorn an. Bis man das Gerät ausschaltet. Erste Hilfe zur Behebung dieses Problems findet sich u. a. im [Boot-Loop Tag-Wiki](#)⁵⁵⁹ bei Stack Exchange.



Android.SE: Boot-
Loop Tag-Wiki

Branding

[Provider](#)-spezifische Anpassungen am [ROM](#). Das kleinste Ärgernis dabei ist evtl. noch die Boot-Animation, die ggf. das Provider-Logo anzeigt. Haarsträubender sind oftmals die zusätzlich installierten Apps ([Bloatware](#)), die der Benutzer ohne [root](#)-Rechte nicht entfernen kann. Darüber hinaus stellt das Branding gelegentlich wünschenswerte zusätzliche Funktionalitäten bereit – wie etwa das Bezahlen von Apps im Playstore über die Mobilfunkrechnung (T-Mobile).

Brick

In der Regel das Lebensende eines Androiden – der dann nur noch als Briefbeschwerer o. ä. herhalten kann. Wörtlich heißt das zwar „Ziegelstein“,

⁵⁵⁹ <http://android.stackexchange.com/tags/boot-loop/info>

aber das würde im Deutschen u. U. zu meilenweisen Verwechslungen mit gewissen Androiden aus dem Hause Motorola führen...

Was sich die Entwickler dabei dachten, als sie die gleichnamige [Permission](#) einführten, sei der Fantasie anheim gestellt ...

Wie verwandelt man einen Androiden nun in einen „Brick“? Dazu werde ich keine Schritt-für-Schritt-Anweisung geben (da wenig sinnvoll). Nur soviel sei gesagt: In etwa 95% aller Fälle hängt das mit dem [Flashen](#) eines zum Gerät inkompatiblen [RUU](#) zusammen. Vermeiden lässt sich solches also durch gründliches Lesen der Anleitungen und prüfen des „Zubehörs“ **vor** dem „Brutzeln“.

CalDAV

Ein Standard für die Synchronisation von Kalendern („Cal“) über [WebDAV](#)⁵⁶⁰. Ermöglicht es, ohne größeren technischen Aufwand Kalenderdaten mit eigenen Ressourcen abzugleichen.



Wikipedia: WebDAV

CardDAV

Das Gegenstück zu [CalDav](#) für Kontaktdaten.

CDMA

[CDMA](#)⁵⁶¹ (**C**ode **D**ivision **M**ultiple **A**ccess) ist ein Mobilfunkstandard der dritten Generation (3G), der primär in Amerika und Teilen von Asien und Afrika für den Betrieb von Mobilfunknetzen Anwendung findet. Die maximal möglichen Datenraten reichen hier fast bis an die 5 MBit/s.



Wikipedia:
CDMA2000

CupCake

Eine Tasse Kuchen, klar. Oder doch eher eine [Android-Version](#)? Genau, nämlich 1.5.*.

Daemon

Kein „Teufelchen“, sondern unter Unix-artigen Systemen ein im Hintergrund laufender Prozess, der bestimmte Dienste zur Verfügung stellt. Auf Android-Systemen läuft beispielsweise bei aktiviertem *USB Debugging* ein [ADB](#) Daemon, der mit einem speziellen Client den Zugriff vom Computer aus ermöglicht. Vertiefende Informationen finden sich u. a. bei [Wikipedia](#)⁵⁶².

Dalvik

Dafür muss ich ein klein wenig ausholen: Android besteht, vereinfacht gesagt, aus einem Linux-Kernel, auf dem eine spezielle Java-Version läuft.

560. <http://de.wikipedia.org/wiki/WebDAV>

561. <http://de.wikipedia.org/wiki/CDMA2000>

562. <http://de.wikipedia.org/wiki/Daemon>

Letzteres ist die sogenannte *Dalvik VM* (wobei „VM“ für „Virtual Machine“ steht). Android Apps sind also in Java geschrieben.

Für die Ausführung der App wird der Java Code in einen sogenannten „Byte Code“ übersetzt, der optimal auf die Hardware (und Android-Version) angepasst ist. Damit dies nicht bei jeder Ausführung der App geschehen muss, passiert diese „Übersetzung“ unmittelbar nach der Installation der App – und der Byte-Code wird im sogenannten *Dalvik Cache* abgelegt. Da dies nach der Installation eines „neuen Systems“ für alle Apps geschehen muss, dauert auch der erste Start nach der Neuinstallation (bzw. bei Erst-Inbetriebnahme) ein wenig länger (dafür geht die Ausführung der Apps nachher entsprechend schneller).

Bei der Installation eines neuen [ROMs](#) muss aus genannten Gründen (optimale Anpassung ans System) der *Dalvik Cache* neu aufgebaut werden. Dafür gibt es im [Recovery-Menü](#) einen extra Menüpunkt – aber das ist im entsprechenden Kapitel auch erklärt.

Debuggen

Wörtlich „entkäfern“. Ein Computer-antiker Begriff, der noch aus einer Zeit stammt, in der „Programmierung“ durch das Ziehen von Drähten, Stecken von Röhren und Löten von Leiterbahnen stattfand. Da war der „Bug“ im „Programm“ nämlich durchaus wörtlich zu nehmen – wenn ein verbrutzelter Käfer für einen Kurzschluss sorgte...

Die Käfer sind mittlerweile zu groß geworden (oder vielmehr die Chips zu klein), trotzdem haben sich beide Begriffe gehalten: „Bug“ für einen Fehler im Programm, und „Debuggen“ für die Suche nach und das Entfernen desselben.

Derivat

Eine Abspaltung (auch *Fork*; englisch *fork* = Gabel, üblicherweise als Maskulinum verwendet) ist in der Softwareentwicklung ein Entwicklungszweig nach der Aufspaltung eines Projektes in zwei oder mehr Folgeprojekte, wobei Teile des Quelltextes und seiner Historie kopiert werden und dann unabhängig von dem ursprünglichen Projekt weiterentwickelt werden. Mit Bezug auf das Urheberrecht wird auch von einem Derivat (*derivativ*, lat.: *derivare* = ableiten) gesprochen. ([Wikipedia](#)⁵⁶³).

563. http://de.wikipedia.org/wiki/Derivat_%28Software%29

DLNA

Die [DLNA](#)⁵⁶⁴ (**Digital Living Network Alliance**) ist eine internationale Vereinigung von Herstellern von Computern, Unterhaltungselektronik und Mobiltelefonen mit dem Ziel, die Interoperabilität von informationstechnischen Geräten unterschiedlicher Hersteller aus dem Bereich Heim- und Eigengebrauch sicherzustellen. (Wikipedia). Klingt nach viel („Oh, da kann ich meine Waschmaschine, das Licht, den Rollladen ...“) – beschränkt sich meist jedoch auf die Steuerung von Bild- und Tonkonserven kompatibler Geräte (auf denen dann auch „DLNA“ steht).



Wikipedia: Digital Living Network Alliance

Donut

Die Lieblingsspeise eines gewissen Homer Simpson – aber was soll das hier? Na klar, süßes Backwerk, und daher folgerichtig eine [Android-Version](#) (1.6).

Downgrade

Kurz und schmerzlos: Das Gegenteil eines [Upgrades](#). Also ein (oder mehrere) Versionsschritt(e) rückwärts.

Warum man so etwas tun sollte/möchte? In der Regel natürlich gar nicht. Aber wenn man nach einem Upgrade wesentlich schlechter dran ist als davor, und auch kein [Update](#) zur Abhilfe in Sicht – dann bleibt einem nicht viel anderes übrig.

Ein weiterer Grund ist häufig, dass man sein Gerät [rooten](#) möchte, das aber mit der aktuellen Firmware nicht möglich ist.

DroidDream

DroidDream ist ein Trojaner, der eine Sicherheitslücke in Android-Versionen vor [Froyo](#) ausnutzt. Er sendet private Daten an einen Server im Netz und installiert eine „Hintertür“ im Android-Gerät, durch die Code aus dem Netz nachgeladen werden kann.



AndroidNews.DE
Droiddream
Trojaner

Bekannt wurde dies Anfang März 2011: Etwa 50 infizierte Apps wurden im *Google Play Store* aufgespürt und aus diesem entfernt. Dies war auch einer der seltenen Fälle, dass Google von der Möglichkeit Gebrauch machte, diese Schadsoft aus der Ferne von betroffenen Android-Geräten zu löschen.

Weitere Informationen finden sich im Netz – u. a. [hier](#)⁵⁶⁵.

DroidSheep

[DroidSheep](#)⁵⁶⁶ ist eine Android-App zur Sicherheitsanalyse des verbundenen WLAN und zum Abfangen offener Facebook, Twitter und anderer Sitzungen. Wie häufig, lässt sich eine derartige Lösung allerdings auch missbrauchen –

564. http://de.wikipedia.org/wiki/Digital_Living_Network_Alliance

565. <http://www.androidnews.de/droiddream-trojaner>

566. http://droidsheep.de/?page_id=263

 wogegen man sich unter Android z. B. mit [DroidSheep Guard](#)⁵⁶⁷ schützen kann.

Eclair

Klingt nach einem süßen Backwerk – und lässt daher korrekt auf eine [Android-Version](#) (2.0.*/2.1.*) schließen.

EDGE



[EDGE](#)⁵⁶⁸ steht für Enhanced Data Rates for GSM Evolution. Als Erweiterung zu [GPRS](#) dient es der Datenübertragung, wobei es die maximal mögliche Datenrate auf 384 kbit/s mehr als verdoppelt. Dennoch gehört es noch zur Kategorie [2G](#). In der Statusleiste von Android-Geräten macht es sich durch

Wikipedia: Enhanced Data Rates for GSM Evolution

F-Droid

Ein alternativer Android-Market, der sich auf [Open Source](#) Apps spezialisiert hat. Details finden sich unter [Öffentliche Märkte](#).

FaceNiff



FaceNiff

[FaceNiff](#)⁵⁶⁹ ist ein Tool ähnlich [DroidSheep](#). Sein Name setzt sich zusammen aus „Facebook“ und „Sniff“ (schnüffeln) – obwohl sich das „Schnüffeln“ dieser Android-App nicht auf Facebook allein beschränkt. Schutz bietet auch hier bereits benannter [DroidSheep Guard](#)⁵⁷⁰.

Fastboot

Der Name ist zunächst ein wenig irreführend – handelt es sich dabei doch nicht um die Möglichkeit, das Android-Gerät schneller einsatzbereit zu haben. *Fastboot* hat eigentlich mit dem installierten Android-Betriebssystem nicht einmal direkt etwas zu tun.

Zu finden ist ein Fastboot-Eintrag gelegentlich im [Boot-Menü](#). Und gedacht ist es in erster Linie zum schnellen Bearbeiten von [Partitionen](#) via USB. Dazu wählt man am Android-Gerät diesen Punkt aus, und kann dann vom PC aus mit der entsprechenden Software passende Befehle absetzen – etwa um die Daten auf einer Partition zu löschen, mit einer Image-Datei zu überschreiben, oder schlicht das Gerät neu zu starten.

Factory-Reset

Auch *Rücksetzen auf Werkseinstellungen* genannt: Wiederherstellung des Auslieferungs-Zustandes. Stimmt natürlich nicht so ganz, denn die ursprüngliche Firmware wird dabei nach einem Update nicht wieder

567. http://droidsheep.de/?page_id=265

568. http://de.wikipedia.org/wiki/Enhanced_Data_Rates_for_GSM_Evolution

569. <http://faceniff.ponury.net/>

570. http://droidsheep.de/?page_id=265

hergestellt; es werden lediglich alle Nutzerdaten einschließlich vom Anwender installierter Apps etc. gelöscht.

Flaschen

A: Behälter für Bier, Cola, Wein, u. a. m.

B: Taugenichtse, Tagediebe, Apfeldiebe...

C: Gesucht war bestimmt eher der Begriff [Flashen](#), gelle?

Flashen

Foto: Benutzung des Blitzlichts (engl.: „flash“)

MIB: Löschen von Teilen des biologischen Speichers, sodass Gedächtnis-Lücken entstehen. Der Vorgang wird auch als „Blitzdingen“ bezeichnet.

Android: Den Androiden mit einem neuen [ROM](#) versehen, indem entweder ein reguläres [Update](#), oder ein Custom-ROM installiert wird. Der Name röhrt daher, dass hier die Daten größtenteils im „internen Speicher“, dem sogenannten „Flash Speicher“, landen. Oft geht damit auch ein teilweises „Blitzdingen“ einher, etwa für den [Dalvik-Cache](#).

Es muss jedoch nicht gleich ein komplettes neues System sein – auch kleinere Updates gelangen hin und wieder auf diese Weise in die Geräte.

FOTA

Ein [OTA](#)-Update der Firmware. Auf einigen [Custom-ROMs](#) läuft auch ein Prozess namens *FOTA kill*, der eben selbiges (insbesondere seine Verfügbarkeits-Meldungen) verhindern soll. Die machen ja da auch keinen Sinn...

Froyo

Eigentlich *Frozen Yoghurt*: Eine [Android-Version](#) (2.2.*)

GingerBread

Eine [Android-Version](#) (2.3.*)

GingerBreak

In Anlehnung an den Namen [GingerBread](#) ist dies zum einen der Name eines Algorithmus als auch einer App zur Erlangung von [Root](#)-Rechten unter Android > 2.2.1.

GPRS

Wikipedia: General Packet Radio Service

GPRS⁵⁷¹ steht für **General Packet Radio Service**. Es dient der Datenübertragung, und ist aktuell wohl die langsamste Fassung davon. GPRS gehört in die Kategorie **2G**, zusammen mit der Erweiterung **EDGE**. Die maximal mögliche Datenrate beträgt bei **GPRS** 171,2 kbit/s. Zeigt die Statusleiste eines Androiden ein „G“, muss sich dessen Besitzer hiermit begnügen...

Hardreset

Wenn der **Softreset**, also das „weiche Herunterfahren“, nicht mehr funktioniert, muss dem Gerät zum Ausschalten die Stromzuführ entzogen werden. Dies geschieht i. d. R. durch Entfernen der Batterie, oder im Falle einer fest verbauten selbigen durch das „Anpieksen“ des Reset-Löchleins. Fehlt auch dieses, schafft häufig das Gedrückt-Halten des Power-Buttons für ca. 30 Sekunden Abhilfe.

Nicht selten wird dieser Begriff als Synonym für den **Factory-Reset** verwendet, der jedoch etwas ganz anderes ist (siehe dort).

HBoot

So nennt HTC seinen **Bootloader**.

HSDPA

Wikipedia: High Speed Downlink Packet Access

HSDPA⁵⁷² (**High Speed Downlink Packet Access**) ist eine Erweiterung des **3G** Mobilfunkstandards **UMTS**, der Datenraten von bis zu 14 MBit/s ermöglicht. Wird manchmal auch gern als „3,5G“ oder „3G+“ bezeichnet, und macht sich in der Statusleiste eines Androiden durch ein „H“ bemerkbar.

HTTPS

Wikipedia: HTTPS

Dieses Kürzel steht für **HyperText Transport Protocol Secure**, was sich auf Deutsch am besten mit „sicheres Hypertext-Übertragungsprotokoll“ wiedergeben lässt. Sämtliche Daten werden hierbei verschlüsselt über das Netzwerk übertragen. Nähere Details finden sich u. a. bei [Wikipedia](#)⁵⁷³.

Ice Cream Sandwich

Wikipedia: Ice Cream Sandwich

Hmmm, wieder etwas Süßes? Richtig: Eine **Android-Version**, nämlich 4.0

Image

Die Fotos, die mit der Kamera des Androiden gemacht wurden, sind damit *nicht* gemeint. Die nennt man nämlich *Pictures*.

571. http://de.wikipedia.org/wiki/General_Packet_Radio_Service

572. http://de.wikipedia.org/wiki/High_Speed_Downlink_Packet_Access

573. <http://de.wikipedia.org/wiki/Https>

Ein **Image** nennt man das Speicher-Abbild einer Partition (für Windows-User: Das, was sich hinter einem Laufwerks-Buchstaben verbirgt; eine Partition kann schließlich wie ein eigenständiges Laufwerk behandelt werden). Im Zusammenhang mit Android werden Images häufig in folgenden Kontexten genannt:

- **Nandroid-Backup:** Erstellt Images von allen Partitionen
- **Update.zip:** Enthält nicht selten ein (oder mehrere) Image(s)
- **Custom-ROMs** kommen auch oft als Images daher
- **Fastboot** kann zum Flaschen verschiedener Images verwendet werden

IMEI

Die *International Mobile Station Equipment Identity (IMEI)* ist eine eindeutige 15-stellige Seriennummer, anhand derer jedes GSM- oder UMTS-Endgerät eindeutig identifiziert werden kann. ([Wikipedia](#)⁵⁷⁴). Diese Nummer ist also Geräte-spezifisch, und wird von diversen Werbe-Modulen gern zur Identifizierung herangezogen (was jedoch die [Playstore Richtlinien](#)⁵⁷⁵ seit August 2014 verbieten). Mit ihr lässt sich aber auch ein Gerät beim Netzanbieter sperren, sodass ein Dieb es nicht mehr verwenden kann (zumindest nicht im gesperrten Netz – dies weltweit durchzusetzen, dürfte ein wenig aufwendig sein). Genaue Details finden sich u. a. bei den [XDA-Developers](#)⁵⁷⁶.



Wikipedia: IMEI



XDA Developers
Guide: Why my IMEI
missing & NO
network?

IMSI

Abkürzung für *International Mobile Subscriber Identity*. Diese aus 15 Ziffern bestehende interne Teilnehmerkennung dient in GSM- und UMTS-Mobilfunknetzen der eindeutigen Identifizierung von Netzteilnehmern, und wird auf der SIM-Karte gespeichert.



Jelly Bean

Code-Name der [Android-Versionen](#) 4.1 bis 4.3

Kernel

Da steckt das Wort „Kern“ drin, genau. Wenn wir hier vom „Kernel“ sprechen, meinen wir den „Betriebssystem-Kern“, den „[Linux-Kernel](#)⁵⁷⁷“. Das ist, vereinfacht ausgedrückt, eine Abstraktions-Schicht: Unten speziell an die jeweilige Hardware angepasst, stellt der *Kernel* „oben“ eine einheitliche Schnittstelle ([API](#)) für die Software zur Verfügung.

574. <http://de.wikipedia.org/wiki/IMEI>

575. <https://play.google.com/intl/en/about/developer-content-policy.html#ADID>

576. <http://forum.xda-developers.com/showthread.php?t=1857054>

577. http://de.wikipedia.org/wiki/Linux_%28Kernel%29

Bei Android läuft auf dem *Linux-Kernel* die [Dalvik-VM](#) (eigentlich je eine pro App), und in der *Dalvik-VM* sodann die [App](#).

KitKat

Ein Schoko-Riegel. Aber auch Code-Name der [Android-Version](#) 4.4

KML

Die *Keyhole Markup Language* ist ein spezieller [XML](#)-Dialekt, der zum Austausch geografischer Informationen verwendet wird. Die bekanntesten Vertreter, die dieses Format verwenden, sind *Google Maps* und *Google Earth*.

Eine Spezialform davon ist *KMZ*: Hier steht das "Z" für "ZIP", die Datei ist also komprimiert. Auf diese Weise ist es möglich, zusätzliche Elemente wie Icons mit der KML-Datei beisammen zu halten.

Launcher

Der *Launcher* ist sozusagen die „Grafische Benutzeroberfläche“ (GUI) von Android – das, was nach dem Entsperren des Bildschirms angezeigt wird. Anders als bei Windows (und eher ähnlich wie bei Linux) gibt es bei Android nicht *den* Launcher, sondern eine Reihe von Alternativen: Angefangen von „besonders Ressourcen-schonend“ bis hin zu „mit allen (un)möglichen Features“. Mehr Details dazu finden sich im Kapitel zum [Home-Screen](#).

Lollipop

In diesem Fall nichts zum Lutschen, sondern die [Android-Version](#) 5.x

MTP



Wikipedia: Media Transfer Protokoll

Das [Media Transfer Protokoll](#)⁵⁷⁸ löst ab [Android 4.0](#) den [USB Massenspeicher](#) ab. Damit kann bei bestehender USB-Verbindung gleichzeitig vom Androiden und vom Computer auf die SD-Karte (und ggf. weitere Speichermedien) zugegriffen werden – was allerdings auf dem Computer passende Treiber-Software voraussetzt. Auch ist vom Computer ein Direktzugriff auf die Speicherkarte (z. B. zum Formatieren) damit nicht länger möglich.

Nandroid Backup

Ein vollständiges System-Backup, welches sich z. B. aus dem [Custom-Recovery-Menü](#) heraus erstellen und auch wieder herstellen lässt. Hier werden nicht einzelne Dateien gesichert, sondern ein Abbild („Image“) des gesamten Systems wird angelegt. Es ist also ein „Alles-oder-Nichts“: Die Wiederherstellung einzelner Dateien ist hier nicht vorgesehen (jedoch [mit speziellen Apps möglich](#)).

578. http://de.wikipedia.org/wiki/Media_Transfer_Protocol

Insbesondere bevor man ein [Custom-ROM](#) einspielt, aber auch generell vor einem System-Update sollte ein Nandroid-Backup angelegt werden. Es ist natürlich auch sonst immer eine gute Idee, ein komplettes Backup zur Hand zu haben.

NFC

Nein, nicht **Nashville Fried Chicken**, sondern [Near Field Communication](#). Dient zum Ultra-Kurzstrecken-Datenaustausch (Reichweite also noch kleiner als bei Bluetooth), und soll u. a. zum „Bezahlen mit dem Handy“ genutzt werden. Aber auch Dinge wie „Handy als Autoschlüssel“ oder „Handy als Fahrkarte“ etc. sind denkbar (und schon gedacht worden).



Wikipedia: Near Field Communication

Open Source

Bei [Open Source](#)⁵⁷⁹ liegen die Quellen offen: Hier handelt es sich um Software, bei der jeder Zugriff auf den Quelltext haben kann. So lässt sich nicht nur herausfinden, wie einzelne Funktionalitäten umgesetzt wurden – sondern auch beispielsweise prüfen, ob eventuell [Hintertüren](#)⁵⁸⁰ eingebaut, oder andere Sicherheitsmängel vorhanden sind. Derartige Prüfungen werden etwa bei [F-Droid](#) durchgeführt, und die Apps dann direkt aus dem geprüften Code erstellt.



Wikipedia: Open Source

OTA

Da liegt was in der Luft. Denn OTA steht für „Over The Air“. Ja was denn? Beim Rundfunk ist es „On The Air“ und heißt Musik. Bei Phil Collins „In The Air Tonight“. Und bei Android ein „(komplettes) Over The Air Update“ – also ein [Kotau](#)⁵⁸¹, sozusagen. Die Frage wäre da nur, wer dabei der Kaiser ist ...



Wikipedia: Backdoor

Also, kurz gefasst: Beim *OTA* werden Software-Updates des Herstellers/ Providers über das Funknetz des letzteren verteilt.



Wikipedia: Kotau

Partition

Eine Partition ist ein zusammenhängender Bereich auf einem Datenträger (unter Windows häufig mit einem Laufwerksbuchstaben verbunden).

Auf einem Android-System sind immer mehrere Partitionen in Benutzung, auch wenn nur der interne Speicher zur Verfügung steht (und keine SD-Karte eingelegt ist): So ist das „/system“ in der Regel nur lesend eingebunden (um Änderungen im Betrieb zu verhindern), während für [Apps](#) und Daten eine eigene Partition („/data“) bereitsteht.

Die SD-Karte beinhaltet meist nur eine (in der Regel unter „/sdcard“ eingebundene) Partition. Es sind aber auch hier mehrere Partitionen

579. http://de.wikipedia.org/wiki/Open_Source

580. <http://de.wikipedia.org/wiki/Backdoor>

581. <http://de.wikipedia.org/wiki/Kotau>

möglich (und werden z. B. mit [App2SD+/Link2SD](#) verwendet) – wobei Android bei Anschluss an den PC via USB nur jeweils die erste Partition davon freigibt.

Weitere Details können z. B. bei [Wikipedia](#)⁵⁸² nachgelesen werden.

Provider

In der Regel ist damit der „Netzanbieter“ gemeint (also E-Plus, T-Mobile, A1 & Co).

Zum anderen könnte aber auch ein Dienst des Android-Systems gemeint sein: So stellt etwa der *Location Provider* über eine [API](#) den Apps die aktuelle Position bereit...

RAM

Diese drei Buchstaben stehen für **R**andom **A**ccess **M**emory – also Speicher, auf den man nach belieben an beliebiger Stelle zugreifen kann. Im Gegensatz nicht etwa zu [ROM](#), sondern zu Dingern wie Bandlaufwerken (jaja, so alt ist der Begriff schon), bei denen man sich erst mühsam vom Start zur gewünschten Position (linear) vortasten muss.

Sowohl auf PCs wie auch auf unseren Androiden ist damit meist der Arbeitsspeicher gemeint, in den die Programme/Apps zur Ausführung geladen werden. Üblicherweise ist dies der Bereich, der generell zu klein ist – oder von dem man halt nie genug haben kann ...

Recovery Menü

Ein separater Bereich des Boot-Menüs (siehe [Bootloader](#)), aus dem heraus verschiedene Operationen wie [Nandroid-Backup](#) oder auch das Bereinigen des [Dalvik-Caches](#) möglich sind.

In das *Recovery Menü* gelangt man in der Regel durch eine spezielle Tastenkombination beim Einschalten. Diese ist aber zum mindesten von Hersteller zu Hersteller unterschiedlich. Bei HTC ist es üblicherweise das Halten der „Leiser-Taste“ während des Einschaltens. Bei Motorolas Milestone muss die Kamera-Taste beim Einschalten gedrückt gehalten, und anschließend die „Lauter-Taste“ betätigt werden. Und so weiter. Bei Bedarf also am besten im Forum erlesen/erfragen.

Einfacher geht es mit dem bereits im Kapitel [Custom-ROM](#) genannten *ROM Manager*: Diese App erlaubt auch einen direkten Boot ins Recovery-Menü. Ebenso integrieren einige Custom-ROMs einen entsprechenden Punkt in dem Menü, welches sich bei langem Drücken der Power-Taste öffnet.

582. http://de.wikipedia.org/wiki/Partition_%28Informatik%29

Repository

Ein Repository (engl. für Lager, Depot, Quellen oder Archiv), auch Repotorium, ist ein verwaltetes Verzeichnis zur Speicherung und Beschreibung von digitalen Objekten. Bei den verwalteten Objekten kann es sich beispielsweise um Programme (Software-Repository), Publikationen (Dokumentenserver), Datenmodelle (Metadaten-Repository) oder betriebswirtschaftliche Verfahren handeln. Häufig beinhaltet ein Repository auch Funktionen zur Versionsverwaltung der verwalteten Objekte. ([Wikipedia](#)⁵⁸³)



Wikipedia:
Repository

Reset

Je nachdem: Siehe [Softreset](#), [Hardreset](#), [Factory-Reset](#), sowie [Wipe](#).

ROM

Richtig offensichtlicher Mist ist diese real offerierte Mehrdeutigkeit: Manchmal hat man den Eindruck, er wurde nur zur Verwirrung der Massen eingeführt. Wer einmal den Namen für eine Android-Komponente nicht kennt, sagt einfach „ROM“. Klingt, als wüsste man voll Bescheid – und die Chance, dass das auch noch Sinn ergibt, ist verdammt hoch ...

Aber im Ernst: Worum geht es hier? Eigentlich steht der Begriff „ROM“ für Read Only Memory – also Speicher, auf den ausschließlich lesend zugegriffen werden kann. Ja, richtig: So wie bei CD-ROM, da steckt das ja auch drin. Nur bei Android, da kann das alles mögliche sein. Nicht selten sachlich falsch – aber wen kümmert's? Schauen wir uns also die einzelnen Bedeutungen einmal an:

Systemspeicher: Teile des Android-Systems werden in der Tat „nur lesend“ eingebunden. Unter anderem eine Schutzmaßnahme, um Veränderungen zu erschweren (damit wir die dusseligen Apps, mit denen uns die Hersteller/Provider „beglücken“, nicht einfach löschen können). So heißt es z. B. in den Spezifikationen des HTC Wildfire: „384 MB RAM; 512 MB ROM“. Nonsense(s): Ein Blick hinter die Kulissen offenbart, dass nur 250MB read-only (/system) eingebunden sind. Die restlichen 250MB „ROM“ stehen zur Installation von Anwendungen zur Verfügung. Read-only? Mitnichten. De facto kann der ganze Bereich jederzeit schreibbar gemacht werden, sofern man [root](#) hat. Also eher irreführend – richtiger müsste es hier heißen: „interner Speicher“, oder – in Abgrenzung vom [RAM](#) – „interner Flash-Speicher“.

Das System selbst: Um die Verwirrung komplett zu machen, wird auch hier gern von „ROMs“ gesprochen (siehe [Custom ROMs](#)). Das hat schon Tradition: Auch bei älteren Spiele-Konsolen sprach man davon, „ein ROM zu laden“. Dabei handelte es sich aber „seinerzeit“ tatsächlich um eine Cartridge –

583. <http://de.wikipedia.org/wiki/Repository>

also einen Speicher-Chip, den man an das Gerät ansteckte. Später kamen dann die Emulatoren, welche die „antiken Geräte“ auf moderner Hardware emulieren können. Und bei diesen kommt die „Cartridge“ natürlich in Form einer Datei daher. Den Begriff „ein ROM laden“ hat man beibehalten. Und schließlich weiter übertragen.

ROM Kitchen

ROM-Knast? Nein, gemeint ist der englische Begriff „Küche“ (also wenn man schon „Knast“ hat). In einem *ROM Kitchen* kann der Anwender sich sein eigenes [ROM](#) kochen. Natürlich setzt dies [root](#) voraus (sonst kann man das fertige „Gericht“ nicht servieren/installieren), und ist etwas Gerätespezifisches.



Ultimate Online
Kitchen

Neben „vollständigen ROM-Küchen“ gibt es auch noch welche, die sich auf Elemente der Gestaltung beschränken, wie z. B. das [UOT](#)⁵⁸⁴ (Ultimate Online Kitchen, Beschreibung und Video findet sich bei den [XDA-Developers](#)⁵⁸⁵). Hier tauscht man lediglich einzelne Elemente (Batterie-Anzeige in der Statusbar, Icons, etc.) eines bereits installierten ROMs aus. Natürlich braucht auch dieses root, da sonst kein Austausch von Systemdateien möglich ist.

root

Aus Herstellersicht: Die Wurzel allen Übels. Objektiv betrachtet: Der Administrator (auch „SuperUser“) eines Linux-Systems. Der darf alles, und kann alles (kaputtmachen auch, ja). Näheres ist im Kapitel [root](#) ausführlicher beschrieben.

RUU

Radio Unit Update: Eine Art [update.zip](#) für das [Radio-Image](#), also ein „Firmware-Upgrade“.

ROM Upgrade Utility: Wie der Name bereits sagt, ein Utility zum Upgrade des [ROM](#), welches entweder vom Hersteller oder von Drittanbietern zur Verfügung gestellt und vom PC aus installiert wird.

Wenn nicht ganz klar ist, was von beidem gemeint ist, ist es in der Regel das erste – wobei das durchaus mit dem zweiten identisch sein kann, da die Begriffe oftmals gleichbedeutend verwendet werden. Was mancherorts als „ROM Upgrade Utility“ bezeichnet wird, ist nämlich nichts anderes als das Update des Radio-Images...

Safe Mode

Vielen aus der Windows-Welt als „Abgesicherter Modus“ bekannt. Bootet man in diesen, werden beim Systemstart alle Anwender-Apps ignoriert. Dies

584. <http://uot.dakra.lt/>

585. <http://forum.xda-developers.com/showthread.php?t=990829>

ist beispielsweise hilfreich, wenn das Gerät bei einem normalen Start in einer Force-Close-Schleife festhängt.

SDK

Das SDK (**S**oftware **D**evelopment **K**it) ist die Grundlage für die Android App Entwicklung und liegt für die jeweilige Version von Android vor. Es ist aber nicht nur dafür verwendbar, sondern bringt auch einige brauchbare Tools wie Fastboot, den Dalvik Debug Monitor sowie ADB mit.

Ergänzt man das Ganze noch um Eclipse⁵⁸⁶, kann es mit der Entwicklung von Apps losgehen!

SIP

Das Kürzel SIP steht für Session Initiation Protocol⁵⁸⁷. In unserem Zusammenhang kann es quasi als Alias für „Internet-Telefonie“ bzw. „IP-Telefonie“ betrachtet werden, da es bei dieser zum Einsatz kommt.



Wikipedia: Session Initiation Protocol

S-OFF

Mit S-OFF (kurz für Security off) ist der volle Zugriff (lesen und schreiben) auf den Bootloader gemeint. Üblich ist dieses „secu_flag“ nur bei HTC, somit kann man es bei nicht-HTC-Geräten ignorieren. Eine ausführliche Beschreibung findet sich u. a. bei AddictiveTips⁵⁸⁸ in englischer Sprache.



AddictiveTips: What Is S-OFF & How To Gain It On HTC Android Phones?

Softreset

Ein „weiches“ herunterfahren des Systems, wenn nichts mehr geht – vergleichbar mit Strg-Alt-Entf am PC. Siehe Weiches Zurücksetzen für Details.



Wikipedia: Stack Exchange Network

Stack Exchange

Das Stack Exchange Network⁵⁸⁹ ist eine vernetzte Gruppe von Communities, auf denen sich im Frage-und-Antwort Stil über diverse Probleme (und

586. http://de.wikipedia.org/wiki/Eclipse_%28IDE%29

587. http://de.wikipedia.org/wiki/Session_Initiation_Protocol

588. <http://www.addictivetips.com/mobile/what-is-s-off-how-to-gain-it-on-htc-android-phones-with-unrevoked-forever/>

589. http://en.wikipedia.org/wiki/Stack_Exchange_Network

natürlich deren Lösungen) ausgetauscht wird. Die Inhalte stehen unter einer [Creative Commons](#)⁵⁹⁰ Lizenz, wie auch dieses eBook.

Der Name setzt sich aus zwei Worten zusammen: „Stack“ (Stapel, Packen, Aufschichtung) und „Exchange“ (Austausch) – was den Charakter des Netzwerks recht gut beschreibt: Zum Austausch steht ein ganzer Stapel verschiedener themenspezifischer Sites zur Verfügung, die untereinander vernetzt sind. Sei es zu technischen Aspekten wie [Android](#)⁵⁹¹, [Ubuntu](#)⁵⁹², [Software Recommendations](#)⁵⁹³ und [Datenbanken](#)⁵⁹⁴, zu diversen wissenschaftlichen oder kulturellen Themengebieten, oder gar zu Lifestyle-Themen; für jeden ist etwas dabei. Über 100 Sites sind es mittlerweile, und es kommen ständig neue hinzu. Allen „Stacks“ gemein ist die Sprache zur Verständigung: Englisch. Ebenfalls gemein ist das Benutzer-Konto, will man sich aktiv beteiligen: Ein Account für alle beteiligten Sites.

Viele Verweise aus diesem Buch gehen in dieses Netzwerk, in dem ich selbst auch recht aktiv bin.



Wikipedia: Creative Commons



Android.StackExchange.com

Tar Archiv



Wikipedia: tar

Die Abkürzung steht für **Tape Archiver**, da ursprünglich für Backups auf Bandlaufwerke gedacht. Doch auch heute noch ist [tar](#)⁵⁹⁵ unter Unix/Linux ein oft genutztes Packprogramm.



Tethering

Die Leinen los! Ja, so kam es einigen vor, als das mit *Froyo* zur Standard-Funktionalität wurde: Mittels [Tethering](#) lässt sich die mobile Internet-Verbindung mit anderen Geräten teilen.



Wikipedia: UMTS

[UMTS](#)⁵⁹⁶ (**Universal Mobile Telecommunications System**) ist ein Mobilfunkstandard der dritten Generation, und wird deshalb umgangssprachlich auch einfach [3G](#) genannt. Seine maximale Datenrate liegt – wie übrigens auch die der [2G](#)-Erweiterung [EDGE](#) – bei 384kBit/s. Schneller wird es mit der Erweiterung [HSDPA](#).

Unroot

Rückgängigmachen des sogenannten „rootens“ (siehe [root](#)), also das System wieder vom root-Zugang befreien (und in den Hersteller-konformen Zustand zurücküberführen). Dies geschieht in der Regel durch das [Flaschen](#) einer

590. http://de.wikipedia.org/wiki/Creative_Commons

591. <http://android.stackexchange.com/>

592. <http://askubuntu.com/>

593. <https://softwarerecs.stackexchange.com/>

594. <http://dba.stackexchange.com/>

595. <https://de.wikipedia.org/wiki/Tar>

596. <http://de.wikipedia.org/wiki/UMTS>

Stock-Firmware (sauberste Variante) – wobei mittlerweile auch etliche rooting-Tools eine entsprechende „Undo-Funktion“ anbieten.

Update

Das Ersetzen etwas Bestehenden durch etwas Neueres. Meist handelt es sich hier um eine Fehlerbereinigung von System oder Apps, aber auch neue/zusätzliche Features können damit einhergehen.

Update.Zip

Ganz offensichtlich eine Datei. Und ebenso offensichtlich will diese etwas aktualisieren – nur was?

Es handelt sich hier um ein „flashbares Update“. Offizielle Firmware-Updates kommen oft unter diesem Namen daher. Und da das System eine solche Datei, so sie im Wurzel-Verzeichnis der SD-Karte liegt, als ein solches betrachtet, lässt sich auf diese Weise auch so einiges anderes ins System mogeln. Das nutzt z. B. Titanium Backup aus, wenn es ein update.zip erstellt.

Einspielen lässt es sich zum Beispiel über das Recovery-Menü.

Upgrade

Ist mit einem Update ein Versionssprung verbunden (etwa von Eclair auf Froyo, oder von Froyo auf Gingerbread, oder gleich von Eclair auf Gingerbread), spricht man von einem Upgrade.



Wikipedia:

USB Massenspeicher

Taucht der Speicher des Androiden nach dem Verbinden mit dem Computer per USB-Kabel wie von Zauberhand als Wechseldatenträger⁵⁹⁷ auf letzterem auf, handelt es sich dabei um USB-Massenspeicher⁵⁹⁸. Ein Teil des Androiden wird quasi zum „Kartenleser“ umfunktioniert, der Computer erhält somit „Vollzugriff“ – was sich auch zum Formatieren der Karte (oder zum Retten versehentlich gelöschter Daten) nutzen lässt. Da dies jedoch eine dedizierte Partition voraussetzt, die entweder nur am Computer oder nur auf dem Androiden bereitstehen kann, setzt Android ab Version 4.0 stattdessen auf MTP.



Wechseldatenträger



Wikipedia: USB-Massenspeicher

VoIP

Abkürzung für **Voice over IP**, oder auch „Internet-Telefonie“. Aber so wie man heute simst (SMS schickt), googelt (mit Google sucht), und skypet (mit Skype chattet oder telefoniert), so voipt man auch.

597. <http://de.wikipedia.org/wiki/Wechseldatentr%C3%A4ger>

598. <http://de.wikipedia.org/wiki/USB-Massenspeicher>

VPN



Wikipedia: Virtual Private Network

Steht für [Virtual Private Network](#)⁵⁹⁹ – zu Deutsch: Virtuelles Privates Netzwerk. Hier wird vom Client eine verschlüsselte Verbindung zu einem VPN-Server aufgebaut, der ersterein sodann in sein eigenes Netz integriert. Aller Datenverkehr lässt sich somit über diese verschlüsselte Strecke leiten – sodass die übertragenen Daten vor Schnüfflern (etwa in offenen WLANs) relativ sicher sind.

WebDAV



Wikipedia: WebDAV

Die Abkürzung [WebDAV](#)⁶⁰⁰ steht für „**Web-based Distributed Authoring and Versioning**“, und ist ein offener Standard zur Bereitstellung von Dateien im Internet – kann also quasi für „Online-Festplatten“ genutzt werden. Doch nicht nur das, auch einige andere Dienste setzen darauf auf. Beispiele wären etwa [CalDAV](#) und [CardDAV](#).

Wipe

Wörtlich übersetzt: Löschen. Das „was“ ist hier allerdings die Frage. Und da kommt es darauf an, wen man fragt bzw. wie man den Wipe initialisiert.

Factory-Reset: Eine Form des Wipe ist mit diesem gleichbedeutend, denn sie löscht lediglich die „/data“ [Partition](#) und den Cache. Das ist der Bereich, in dem die selbst installierten Apps sowie die Daten abgelegt werden.

Dalvik-Cache: Mit dem Wipe/Löschen des Dalvik-Caches erzwingt man eine Neu-Übersetzung des Programmcodes aller Apps. Damit geht kein Datenverlust einher: Lediglich der nächste Gerätestart dauert etwas länger.

Komplett-Wipe: Den gibt es in verschiedenen Versionen des [Recovery-Menüs](#). Sollte nur ausgeführt werden, wenn man auch wirklich weiß, was man da tut: Der löscht nämlich alle Daten von allen Partitionen, auch von „/system“. Danach geht dann nichts mehr – es lässt sich lediglich ein neues [ROM](#) einspielen.

XML

Die *eXtensible Markup Language* stellt hierarchisch strukturierte Daten in Form von Textdaten dar, und wird u. a. für den plattform- und implementationsunabhängigen Austausch von Daten zwischen Computersystemen eingesetzt. Normalerweise enthalten XML-Dateien keinen Binärkode, auch wenn dies aufgrund der Erweiterbarkeit durchaus möglich ist. Daher kann man selbige notfalls auch im "Texteditor" betrachten – wenn auch nicht unbedingt sehr komfortabel.

Weiterführende Informationen finden sich bei der [Wikipedia](#)⁶⁰¹.

599. http://de.wikipedia.org/wiki/Virtual_Private_Network

600. <http://de.wikipedia.org/wiki/WebDAV>

601. http://de.wikipedia.org/wiki/Extensible_Markup_Language