

Features, Hacks, Gadgets:

Open Source

Dieser Artikel ist ein Teil des Extra-Kapitels »Features, Hacks, Gadgets« aus dem »Praxisbuch Mac OS X Tiger«. Das Kapitel wurde in einzelne Artikel aufgeteilt, die in sich abgeschlossen sind und einzeln unter <http://www.galileodesign.de/959> und <http://www.galileocomputing.de/963> für Leser des Praxisbuchs und der zweiten Auflage von »UNIX für Mac OS X-Anwender« zum Download zur Verfügung stehen. Vorschläge und Anregungen für weitere Themen werden gerne entgegen genommen. Diese PDF-Datei wurde für den Ausdruck optimiert und verwendet das Papierformat DIN A4. (Die Seitenangaben beziehen sich auf das Praxisbuch.)



Mac OS X und Open Source

Mit Mac OS X hat Apple sich und seinen Anwendern das Tor zur Open Source-Welt weit aufgetoßen. Ihnen als Anwender steht damit eine große Anzahl an kostenlosen Programmen für fast alle Anwendungen zur Verfügung.

Inhalt

Mac OS X und Open Source	1
Einführung	2
Programme kompilieren und installieren	3
configure, make und make install	4
Pakete installieren mit DarwinPorts	6
DarwinPorts installieren und einrichten	6
Programme kompilieren und installieren	7
Der X11-Server	9
Der Hintergrund	9
GNU Network Object Model Environment – GNOME	12
Das K Desktop Environment – KDE	13
Weitere Window-Manager	14
Ein Best of Open Source	15
OpenOffice.org	15
The GIMP	16
Sodipodi	17
Blender3D	17

Einführung

Frei, nicht Freibier | Open Source hat sich mittlerweile zu einem schillernden Begriff entwickelt. Sehen es die einen als eine politische Bewegung, die im Zusammenhang mit Urheberrechten und der Freiheit von Informationen eine erhebliche Rolle spielt, so ist es für die anderen nicht mehr als ein vorübergehendes Phänomen, ein Spielzeug für Computerfreaks. Die Wahrheit liegt wahrscheinlich irgendwo in der Mitte. Mit Mac OS X hat sich Apple der Open Source-Bewegung weit geöffnet. Dabei gehört Apple nicht nur zu den Profiteuren einer quelloffenen Software, sondern gibt auch etwas an die Bewegung zurück. So steht der Kernel von Mac OS X im Kontext des Projekts OpenDarwin jedem Interessierten offen. Er kann frei verwendet werden. Außerdem stellt Apple seine Fähigkeiten als professionell geführte Firma einigen Projekten zur Verfügung.

Die Basis des Browsers Safari besteht aus einem Programmmodul zum Anzeigen von Hypertexten. Dies stammt keineswegs von Apple, sondern wurde von den Entwicklern des KDE-Projekts begonnen. Apple hat diesen Quellcode adaptiert, war aber im Gegenzug verpflichtet, die Änderungen frei zugänglich zu machen. Hiervon profitierte wiederum das KDE-Projekt, das die Entwicklungen von Apple für die eigene Arbeit verwenden durfte. Open Source-Software ähnelt manchmal ein wenig dem Basteln am Motor des Lieblingsautos. Nichtsdestotrotz gibt es auch in diesem Bereich professionelle Programme, die den Vergleich mit kommerzieller Software nicht scheuen müssen.

Forking bei Projekten | Da Open Source-Software sehr stark auf ehrenamtlicher und unbezahlter Arbeit und dem Engagement vieler Menschen in ihrer Freizeit beruht, treten hier zwischenmenschliche Faktoren deutlicher zu Tage, als es bei einer hierarchisch geleiteten Firma der Fall ist. Manchmal sind die Differenzen innerhalb eines Teams so groß, dass sich eine Gruppe entschließt, das Projekt zu verlassen und ein eigenes aufzubauen. Da ja der Quellcode offen liegt, bauen diese Splittergruppen oft auf der schon vorhandenen Basis auf und geben dem Projekt lediglich eine andere Zielrichtung. In diesem Fall spricht man von Forking, und bei mehreren Projekten kann man mittlerweile einen regelrechten Stammbaum erstellen, wer sich wann warum abgespalten hat. Dies muss nicht zwingend negativ sein, denn es belebt die Konkurrenz und appelliert auch an so etwas wie den Sportsgeist der Entwickler.

Der gcc | Das zentrale Element für die Arbeit mit Open Source-Programmen ist der GNU C Compiler. Er wird in den Versionen 3.3 und 4.0 mit Xcode installiert. Geläufig ist die Abkürzung gcc. Die Aufgabe dieses Compilers ist es, den Quellcode von Programmen in ausführbare Applikationen zu übersetzen. Letzteres wird »kompilieren« genannt. Die Sprachen C und C++ sind in diesem Bereich am beliebtesten – zum Teil, weil sie in der Abarbeitung enorm schnell sind. Aber es gibt auch in Java oder Python geschriebene Open Source-Programme, die dann nicht übersetzt werden müssen.

Einbindung von Libraries | Mit Xcode werden auch jede Menge Programm- und Funktionsbibliotheken installiert. Bei diesen Dateien handelt es sich um vorgefertigten Programmcode, der vom Compiler in die zu kompilierenden Programme eingefügt werden kann. Diese Bibliotheken stellen zum Beispiel die Schnittstelle zum NetInfo-System oder aber zur Ansprache der MySQL-Datenbank zur Verfügung. Beim Kompilieren, wenn der Quellcode in ein ausführbares Programm umgewandelt wird, greift der gcc auf diese Bibliotheken und Funktionssammlungen zurück. Viele Programme sind von etlichen Bibliotheken abhängig. Dies wird oft Dependency genannt und hat zur Konsequenz, dass das Fehlen einer Bibliothek das Kompilieren dieses Programms verhindern kann. Die Bibliotheken stammen nicht alle zwingend von Apple selbst. So gibt es auch von den Entwicklern des MySQL-Systems eigene Bibliotheken, mit deren Hilfe Programme direkt auf MySQL-Funktionen zugreifen können. Da es hier mittlerweile zu recht komplexen Abhängigkeitsverhältnissen gekommen

ist, gibt es mehrere Projekte, die dem Anwender die Arbeit und die Verwaltungsaufgaben abnehmen und Buch darüber führen, welches Softwarepaket von anderen abhängig ist. Unter Mac OS X sind Fink und DarwinPorts die bekanntesten Projekte in dieser Richtung. Es besteht aber auch die Möglichkeit, Programme selbstständig in ein lauffähiges Format zu übertragen.

Concurrent Versions System | Wenn Sie sich ein wenig im Bereich Open Source umschauen, werden Sie öfter auf das Kürzel CVS stoßen. CVS steht für »Concurrent Versions System«. Hierbei handelt es sich um ein System, das ein Team von Programmierern bei der Koordination ihrer Entwicklungsarbeit unterstützt. CVS besteht aus einer Client-Server-Architektur. Der aktuelle Stand des Quellcodes wird zentral auf einem Server gespeichert. Nimmt ein Entwickler Änderungen am Quellcode vor, so führt das System Buch über diese, und es ist auch möglich, Änderungen zurückzuverfolgen. Sie können sich außerdem – Mac OS X liefert auch die entsprechenden Programme hierfür – direkt von diesem Server die Quellcodes laden und aus ihnen ein Programm erstellen. Dies ist allerdings immer ein wenig Glückssache, da es ja die aktuellsten Änderungen sind, die oft noch nicht ausreichend getestet sind. Ebenso wie bei kommerzieller Software gilt auch hier, dass Sie half fertige Programme nicht auf einem Rechner einsetzen sollten, den Sie zur produktiven Arbeit unbedingt benötigen. Die Gefahr des Datenverlustes ist hier besonders hoch.

Nightly Builds | Die Nightly Builds sind ebenso wie CVS eine Möglichkeit, sich den letzten Stand der Entwicklungsarbeiten anzuschauen. Allerdings liegt hier nicht der Quellcode vor, sondern das schon kompilierte Programm. Das Kompilieren übernimmt in diesem Fall ein speziell dafür eingerichteter Rechner, der Ihnen die ausführbaren Programme zum Download bereitstellt. Bei diesen Builds ist es ähnlich wie beim aktuellsten Quellcode. Es kann sein, dass viele Fehler behoben sind und das Programm hervorragend seinen Dienst verrichtet. Es kann aber auch gar nicht funktionieren.

Programme kompilieren und installieren

Quellcode übersetzen | Liegt ein Programm im Quellcode vor, dann müssen Sie aus diesem Code ein ausführbares Programm erstellen. Sie finden unten ein kleines Programm in der Programmiersprache C.

```
/* hallo.c */
#include <stdio.h>
int main (void) {
    printf("Hallo Welt\\n");
    return 0;
}
```

Geben Sie diesen Code im Editor `nano` ein, und speichern Sie die Datei unter dem Namen `hallo.c`. Das kleine Programm hat lediglich die Aufgabe, Ihnen am Terminal den Text `Hallo Welt` auszugeben. Mit der Zeile `#include <stdio.h>` wird auf eine Funktionsbibliothek von Apple zurückgegriffen, die unter anderem die Ein- und Ausgabe von Text ermöglicht. Mit der Zeile `printf("Hallo Welt\\n");` wird der Text ausgegeben, und die Rückmeldung, dass das Programm erfolgreich seine Arbeit verrichtet hat, erfolgt mit der Zeile `return 0;`.

Haben Sie den Quelltext gespeichert, dann können Sie mit dem Befehl `gcc -v -o hallo hallo.c` den Compiler anweisen, aus dem Quellcode ein lauffähiges Programm zu erstellen. Die Option `-v` gibt Ihnen detailliert Auskunft über den Verlauf des Übersetzungsvorgangs (siehe Abbildung 1). Die Option `-o hallo` sorgt dafür, dass der Compiler das fertige Programm im aktuellen Verzeichnis mit dem Namen `hallo` speichert. Das Ergebnis dieses Aufrufs besteht in einer Datei **hallo**, die Sie mit `./hallo`

aufzurufen können. Das Programm wird ausgeführt, und am Terminal erhalten Sie die Nachricht `Hallo Welt`.

```

kai@localhost:~/nano-1.2.5 > make
make all-recursive
Making all in po
make[2]: Nothing to be done for `all'.
Making all in m4
make[2]: Nothing to be done for `all'.
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c color.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c cut.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c files.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c global.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c move.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c nano.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c rcfile.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/

```

Abbildung 1: Mit dem gcc wurde ein kleines Programm übersetzt.

gcc auswählen | Mit Xcode werden zwei Versionen des gcc-Compilers installiert. Beide finden Sie im Verzeichnis `/usr/bin`, und genau genommen handelt es sich bei dem Programm `/usr/bin/gcc` lediglich um einen Verweis auf das Programm `gcc-3.3` oder `gcc-4.0`. Wenn Sie ein Programm mit `gcc` übersetzen möchten, dann sollten Sie sich zuvor in dessen Dokumentation informieren, welche Version des Compilers benötigt wird. Bei einigen Programmen wird mittlerweile die aktuelle Version 4.0 vorausgesetzt, während andere noch nicht für diese optimiert wurden. Mit dem Programm `gcc_select` und einer Versionsnummer als Option können Sie zwischen den beiden Versionen des Compilers umschalten. Der Aufruf `sudo gcc_select 4.0` wählt die Version 4.0 aus, und Sie erhalten die Nachricht `Default compiler has been set to: gcc version 4.0.0` für einen erfolgreichen Wechsel. Der Aufruf von `sudo gcc_select 4.0` würde die ältere Version auswählen. Wenn Sie nicht wissen, welche Version des Programms benötigt wird, oder die Übersetzung fehlschlägt, dann kann ein Versuch mit der anderen Version lohnenswert sein.

```

kai@localhost:~ > ls -l /usr/bin/gcc*
lrwxr-xr-x 1 root wheel 7 2 Okt 12:22 /usr/bin/gcc -> gcc-3.3
-r-xr-xr-x 1 root wheel 135076 20 Mär 2005 /usr/bin/gcc-3.3
-rwxr-xr-x 1 root wheel 39524 20 Mär 2005 /usr/bin/gcc-4.0
-rwxr-xr-x 1 root wheel 139540 20 Mär 2005 /usr/bin/powerpc-apple-darwin8-gcc-4.0.8
kai@localhost:~ >

```

Abbildung 2: Im Verzeichnis `/usr/bin` finden sich mehrere Versionen des gcc.

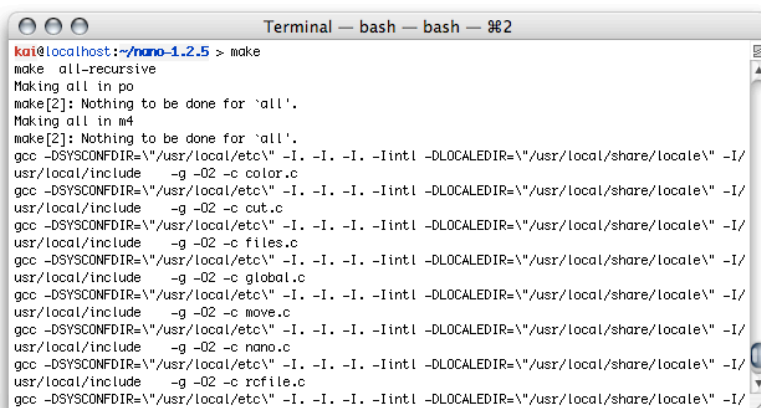
configure, make und make install

Der klassische Dreiklang | Schon in den Anfängen der quelloffenen Software gab es das Bestreben, die Installation möglichst einfach zu machen. Ein relativ frühes, aber auch heute noch gebräuchliches Werkzeug ist die Kombination der Befehle `configure`, `make` und `make install`.

Vorgaben definieren | Hinter `configure` steht oft ein Skript, das die Installation und die Kompilierung des Programms vorbereitet. Schon beim Übersetzen des Quelltextes in ausführbaren Code können bei vielen Programmen Entscheidungen getroffen werden. So ist es beim Apache-Webserver bereits bei der Übersetzung möglich zu entscheiden, ob er in der Lage sein soll, eine gesicherte Verbindung mittels HTTPS herzustellen, oder ob der Anwender dieses Modul nicht benötigt. Ein weiteres Beispiel ist die Unterstützung der NetInfo-Datenbank durch den Mailserver Postfix. Auch hier hätte dieses Modul nicht eingebunden werden müssen, damit Postfix seine Arbeit verrichten kann. Allerdings hätte der Anwender dann unter Mac OS X den Nachteil, dass auf die Benutzerinformatio-

nen nur über Umwege zugegriffen werden kann. Eine solche Konfiguration kann bei einigen Programmen Fehler ausschließen oder ihre Arbeitsgeschwindigkeit erhöhen, da der Quellcode dann von überflüssigem Ballast befreit wurde. Das Skript `configure` wird mit unterschiedlichen Optionen aufgerufen. Welche dies im konkreten Fall sind, ist hoffentlich der mitgelieferten Dokumentation zu entnehmen.

Quellcode kompilieren | Das Resultat von `configure` ist eine `makefile` genannte Datei. Diese enthält die mit `configure` vorgenommenen Einstellungen in einer Form, die um einiges abstrakter ist als die an `configure` zu übergebenden Parameter in der Form `--ssl=enable`. Dieses `makefile` wird von dem Programm `make` eingelesen. Es enthält die Informationen, die `make` benötigt, um die Software korrekt zu übersetzen und mit Hilfe des Compilers ein lauffähiges Programm zu erzeugen. Genau genommen ist das `makefile` also eine Steuerungsdatei, die `make` mitteilt, mit welchen Optionen und in welcher Reihenfolge der Quellcode an den Compiler zu übergeben ist. Im Fortgang der Übersetzung wird er von `make` mehrfach aufgerufen. Das Kompilieren von Programmen ist ein durchaus komplexer Vorgang, immerhin muss ja ein Konstrukt wie `if test -nt` in eine für den Prozessor des Computers verständliche binäre Form gebracht werden. Bei sehr großen Paketen, wie zum Beispiel der Office Suite OpenOffice.org, kann die Kompilierung auf einem langsamen Rechner durchaus auch einen ganzen Tag, wenn nicht sogar länger, dauern. Wundern Sie sich also nicht, wenn Ihr Rechner auf einmal bei der Kompilierung eines Programms kaum noch andere Befehle entgegennimmt. Der Compiler `gcc` kann gern durchaus einmal bis zu 90% der Rechenleistung in Anspruch nehmen. Die Ausgaben des Compilers während der Arbeit sind für technisch nicht so versierte Anwender eher kryptisch (siehe Abbildung 3). Solange aber der Vorgang nicht abbricht, ist lediglich Geduld gefragt.



```
kai@localhost:~/nano-1.2.5 > make
make all-recursive
Making all in po
make[2]: Nothing to be done for `all`.
Making all in m4
make[2]: Nothing to be done for `all`.
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c color.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c cut.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c files.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c global.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c move.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c nano.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
usr/local/include -g -O2 -c rcfile.c
gcc -DSYSCONFDIR=\"/usr/local/etc\" -I. -I. -I. -Iintl -DLOCALEDIR=\"/usr/local/share/locale\" -I/
```

Abbildung 3: Der Kompilervorgang wird am Terminal von `make` ausgegeben.

Ergebnisse ans Ziel bringen | Ist der Quellcode in ein ausführbares Programm übersetzt worden, so müssen die jetzt vorliegenden Programme an die richtige Stelle im Dateisystem, sprich in die richtigen Verzeichnisse gebracht werden. Je nachdem, was für ein Programm Sie installieren, wird es lediglich in das Verzeichnis `/usr/bin` kopiert. Handelt es sich aber um ein größeres Paket, das noch weitere Dateien wie Manpages enthält, dann sind hier mehrere Kopiervorgänge notwendig. Mit dem Befehl `make install`, der in der Regel vom Superuser ausgeführt werden muss, werden die Dateien an die von den Entwicklern vorgesehenen Stellen kopiert.

Dokumentation beachten | Wenn Sie Software aus dem Quellcode heraus installieren und kompilieren, sollten Sie die beiliegende Dokumentation auf jeden Fall gründlich lesen. Es gibt immer kleine Stolperstellen, die für Fehler sorgen und zu vermeiden sind, wenn man sich strikt an die Vorgaben der Entwickler hält. Insbesondere mit den beiden Versionen des `gcc` kann es immer mal wieder zu Fehlern kommen, und der Wechsel der Version mit `gcc_select` kann angeraten sein.

Pakete installieren mit DarwinPorts

Programme automatisch installieren | Die Kompilierung und Installation von Software ist, wenn es um recht große Programme wie zum Beispiel die Bildbearbeitung The GIMP geht, für den normalen Anwender ein recht mühsames Unterfangen. Eine Reihe von Abhängigkeiten müssen beachtet werden, und je nach verwendetem Betriebssystem müssen dem Compiler einige Optionen übergeben werden. Die Idee hinter den DarwinPorts besteht darin, dass ehrenamtliche Entwickler Ihnen einen Großteil dieser Arbeit abnehmen, herausfinden, welche Optionen benötigt werden und welche Abhängigkeiten zwischen einzelnen Programmen und Bibliotheken bestehen, und diese in zentralen Paketen zusammenfassen. Für den Anwender ist dieses Verfahren erheblich komfortabler, da er sich auf die Arbeit erfahrener Entwickler verlassen kann und nur ein paar Befehle eingeben muss, während sich das System der DarwinPorts um die korrekte Installation vom Download der Quelltexte bis zur Installation der fertigen Programme kümmert. Im Zentrum steht hierbei der Befehl `port`.

Ports | Der Name DarwinPorts rührt daher, dass der Vorgang, eine Software für ein anderes Betriebssystem aufzubereiten, auch als Portierung bezeichnet wird. Unter FreeBSD als Vorläufer von Mac OS X dient die so genannte Port-Verwaltung schon seit Jahren dazu, Software zu installieren und auch neue Versionen einzuspielen. Die Grundlage der DarwinPorts bilden die Port-Dateien. Sie enthalten neben einer kurzen Beschreibung des Programms auch die Quellen im Internet, die für die Kompilierung unter Mac OS X notwendigen Optionen und weitere Anweisungen, wie sich das betreffende Programm am besten unter Mac OS X installieren lässt. Diese Dateien werden von einer Reihe ehrenamtlicher Entwickler erstellt, verwaltet und aktualisiert. Sollte sich einmal eine Software nicht installieren lassen und der Installationsvorgang mit einem Fehler abbrechen, dann sollten Sie bei allem Ärger immer im Hinterkopf behalten, dass es sich hier um ehrenamtliche und unbezahlte Arbeit handelt und keine professionelle und gewinnorientierte Firma wie Apple das Projekt managt.

/opt/local | Ein weiterer Vorzug der DarwinPorts besteht darin, dass die mit dem System installierten Programme im Verzeichnis **/opt/local** installiert werden. Es wird somit eine scharfe Trennung zwischen der Grundinstallation von Mac OS X mit den Programmen in den Verzeichnissen **/bin**, **/sbin**, **/usr/sbin** und **/usr/bin** eingehalten und somit vermieden, dass ein Bestandteil des Systems der DarwinPorts überschrieben wird. So befinden sich die installierten Programme im Verzeichnis **/opt/local/bin**, von den Programmen benötigte Funktionsbibliotheken im Verzeichnis **/opt/local/lib** und die Dokumentationen im Verzeichnis **/opt/localshare/man**. Im Prinzip spiegelt dieses Verzeichnis ein wenig den Aufbau von **/usr** wieder.

DarwinPorts installieren und einrichten

Package | Für die Installation der DarwinPorts können Sie im Bereich **Get DarwinPorts** unter <http://darwinports.opendarwin.org> ein Installationspaket herunterladen. Haben Sie das Disk Image aktiviert, dann können Sie mit einem Doppelklick das Installationspaket **DarwinPorts.pkg** öffnen und damit das Installationspaket starten. Der Verlauf der Installation entspricht dem normaler Software unter Mac OS X. Anschließend befindet sich im Verzeichnis **/opt** die komplette Struktur der DarwinPorts.

PATH-Variable | Da auch der zentrale Befehl `port` im Verzeichnis **/opt/local/bin** installiert wurde, müssen Sie die **PATH**-Variable (siehe S. 75f) um den Pfad **/opt/local/bin** ergänzen. Die in der Dokumentation der DarwinPorts empfohlene Ergänzung der Variable `MANPATH` ist unter Mac OS X nicht notwendig. Der Befehl `man` durchsucht alle Verzeichnisse, die in der Variable `PATH` enthalten sind,

und sucht dann im korrespondierendem Verzeichnis **share/man** nach der Dokumentation. Befindet sich also der Pfad **/opt/local/bin** in der Umgebungsvariable `PATH`, dann sucht `man` automatisch im Verzeichnis **/opt/local/share/man** nach der passenden Dokumentation des Befehls und findet sie auch dort.

Aktualisierung | Da die Port-Dateien laufend aktualisiert und berichtigt werden, sollten Sie direkt nach der Installation das System der DarwinPorts mit dem Befehl `sudo port -d selfupdate` auf den aktuellen Stand bringen. Mit dieser Selbst-Aktualisierung werden sowohl die Port-Dateien als auch die eigentliche Infrastuktur aktualisiert. Eine (schnelle) Verbindung ins Internet ist hier wie auch bei den nachfolgend beschriebenen Aktionen erforderlich.

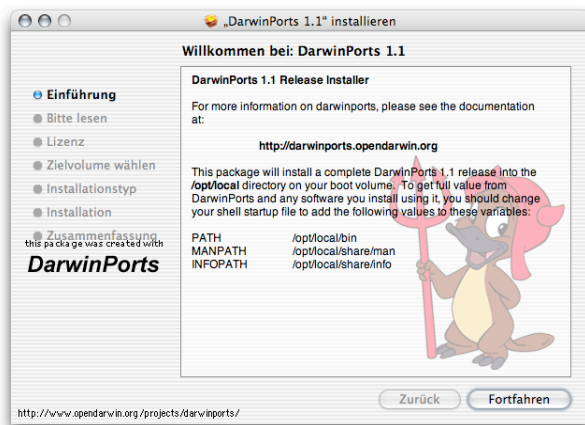


Abbildung 4: Die DarwinPorts werden mit dem Installationsprogramm von Mac OS X installiert.

Programme kompilieren und installieren

Der Befehl port | Die Verwaltung und der Zugriff auf DarwinPorts erfolgt durch den Befehl

`port Aktion Paket`

Um sich eine Liste aller verfügbaren Programme anzeigen zu lassen, können Sie `port` in Verbindung mit der Aktion `list` aufrufen. Es bietet sich an, die doch sehr lange Liste (siehe Abbildung 5) mit einer Pipe an `less` weiterzuleiten und mit Hilfe von `grep` gezielt nach einem Begriff zu suchen. Sie können auch `port search Begriff` verwenden, um die Ausgabe einzuschränken.

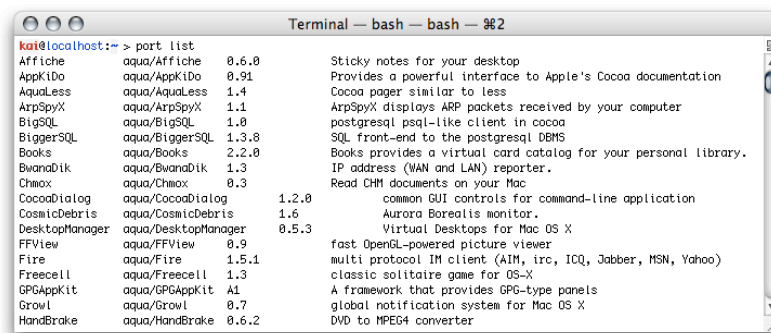
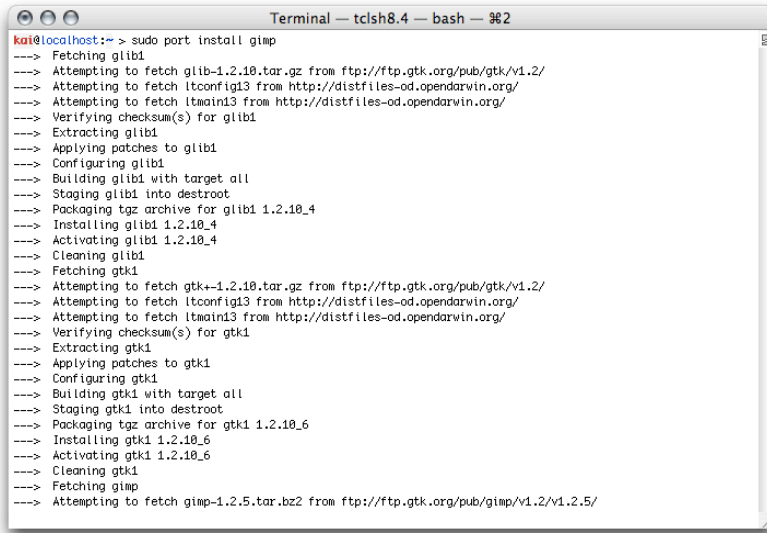


Abbildung 5: Der Befehl `port list` listet alle verfügbaren Pakete auf.

Abhängigkeiten aufzeigen | Bevor Sie ein Programm installieren, können Sie sich mit `port deps Name` die Abhängigkeiten dieses Programms anzeigen lassen. Sie können somit ungefähr ersehen, welche Programme zusätzlich installiert werden müssen und wie lange und wie umfangreich der Installationsvorgang sein wird. Hierbei werden allerdings Abhängigkeiten der zweiten Ebene – wenn also ein Paket ein Paket voraussetzt – das wiederum ein anderes benötigt, nicht detailliert aufgelistet. Bei einigen Ports stehen Ihnen mehrere Varianten zur Verfügung. Beispielsweise ist es bei eini-

gen Servern möglich, diese mit anderen zu koppeln, weitere Funktionen in dem Programm zu aktivieren oder eine speziell für Ihre Version von Mac OS X optimierte Fassung einzuspielen. Mit der Option `variants` können Sie sich die zur Verfügung stehenden Varianten anzeigen lassen. So steht beispielsweise der Server Zope in zwei Varianten zur Verfügung. Auf die Eingabe `port deps zope` hin erhalten Sie die Ausgabe `freebsd darwin`, wobei Sie wie nachfolgend beschrieben natürlich die Variante `darwin` auswählen sollten.



```

Terminal — tcsh8.4 — bash — %2
kai@localhost:~$ sudo port install gimp
--> Fetching glib
--> Attempting to fetch glib-1.2.10.tar.gz from ftp://ftp.gtk.org/pub/gtk/v1.2/
--> Attempting to fetch ltconfig13 from http://distfiles-od.opendarwin.org/
--> Attempting to fetch ltmain13 from http://distfiles-od.opendarwin.org/
--> Verifying checksum(s) for glib
--> Extracting glib
--> Applying patches to glib
--> Configuring glib
--> Building glib with target all
--> Staging glib into destroot
--> Packaging tgz archive for glib 1.2.10_4
--> Installing glib 1.2.10_4
--> Activating glib 1.2.10_4
--> Cleaning glib
--> Fetching gtk
--> Attempting to fetch gtk-1.2.10.tar.gz from ftp://ftp.gtk.org/pub/gtk/v1.2/
--> Attempting to fetch ltconfig13 from http://distfiles-od.opendarwin.org/
--> Attempting to fetch ltmain13 from http://distfiles-od.opendarwin.org/
--> Verifying checksum(s) for gtk
--> Extracting gtk
--> Applying patches to gtk
--> Configuring gtk
--> Building gtk with target all
--> Staging gtk into destroot
--> Packaging tgz archive for gtk 1.2.10_6
--> Installing gtk 1.2.10_6
--> Activating gtk 1.2.10_6
--> Cleaning gtk
--> Fetching gimp
--> Attempting to fetch gimp-1.2.5.tar.bz2 from ftp://ftp.gtk.org/pub/gimp/v1.2/v1.2.5/

```

Abbildung 6: Die Installation von The GIMP

Software installieren | Um einen Port zu installieren, müssen Sie mit dem Internet verbunden sein. Mit der Eingabe `sudo port install Name` wird der angegebene Port installiert. Hierbei geht `port` so vor, dass zuerst die Abhängigkeiten aufgelöst werden, also die zugrunde liegende Software installiert wird. Erst dann wird das eigentlich gewünschte Programm konfiguriert. Der Installationsvorgang beginnt mit dem Übertragen der Quellcodes. Hierbei versucht `port` mit den in dem `port file` angegebenen Servern Kontakt aufzunehmen. Diese Schritte werden mit den Ausgaben `Fetching` und `Attempting to fetch` am Terminal angezeigt. War `port` in der Lage, die Dateien herunterzuladen, so wird als Erstes die Prüfsumme (siehe S. 162) der heruntergeladenen Datei mit der in dem `port file` angegebenen Summe verglichen (`Verifying checksum`), um trojanische Pferde auszuschließen und die Integrität der installierten Software zu gewährleisten. Nachdem das Archiv entpackt (`Extracting`) wurde, werden zunächst notwendige Änderungen (`Applying patches to ...`) eingepflegt, und dann, das enthaltene Skript `configure` ausgeführt. Sobald der Text `Building ...` erscheint, tritt der Compiler in Aktion und das Programm wird übersetzt. Ist dies erfolgt und handelt es sich nicht um die eigentlich zu installierende Software, so wird das neu entstandene Programm in einem anderen Verzeichnis übergangsweise regelrecht geparkt. Dies wird `destroot` genannt. Darauf folgt die Installation des Programms intern im DarwinPorts-System. Es befindet sich hierbei noch nicht im Verzeichnis `/opt/local/bin`. Mit den DarwinPorts ist es möglich, mehrere Versionen eines Programms zu installieren und später mit den Aktionen `activate` und `deactivate` zwischen diesen zu wechseln. In den Standardeinstellungen wird das neu kompilierte Programm gleich aktiviert (`Activating ...`) und somit ins Verzeichnis `/opt/local/bin` kopiert.

Installationen entfernen | Wenn Sie ein installiertes Programm wieder löschen möchten, so können Sie es mit `sudo port uninstall Name` wieder entfernen. Sobald nach der Meldung `Uninstalling` der Prompt wieder zur Verfügung steht, ist das Programm gelöscht.

Fehler beheben | Es kann manchmal vorkommen, dass die Installation fehlschlägt und mit einer Fehlermeldung abbricht. In diesem Fall können Sie nachschauen, ob es vielleicht eine Variante (`port`

variants Name) gibt, die für Mac OS X oder Ihre Installation besser geeignet ist. Eine Variante können Sie mit einem Pluszeichen nach dem Namen des Ports angeben. So installieren Sie mit `sudo port install zope +darwin` die angepasste Version für den Darwin-Kernel. Auch kann der Fehler darin bestehen, dass die verwendete Version des `gcc` nicht korrekt ist. Wechseln Sie hierzu mittels `gcc_select` zunächst die Version des `gcc`. Dann gilt es, die durch den Fehlschlag bereits vorhandenen temporären Dateien mit `sudo port clean Name` zu entfernen. Um ein nicht installiertes Paket komplett zu entfernen, können Sie zusätzlich die Option `—all` verwenden. Die Eingabe `sudo port clean —all Name` würde alle Dateien des Pakets löschen. Dazu gehört auch der aus dem Internet heruntergeladene Quellcode.

Aktualisierung | Eine weitere Fehlerquelle kann darin bestehen, dass ein Paket, das zur Installation benötigt wird, in einer veralteten Version vorliegt. Mit der Option `upgrade` können Sie ein installiertes Programm oder eine installierte Bibliothek auf den neuesten Stand bringen. Die Eingabe `sudo port upgrade WindowMaker` würde den entsprechenden Fenster-Manager für X11 (siehe unten) aktualisieren. Die Eingabe von `sudo port -a upgrade` würde alle installierten Programme in einem Durchgang aktualisieren. Dies kann einige Zeit in Anspruch nehmen.

Konfiguration in /opt/local/etc | Wenn Sie für ein mit den DarwinPorts installiertes Programm Voreinstellungen in einer Datei, die im Verzeichnis `/etc` gespeichert sein sollte, ändern möchten, dann werden Sie diese vielleicht nicht finden. Die Voreinstellungen werden in diesem Fall im Verzeichnis `/opt/local/etc` gespeichert.

Der X11-Server

Der Hintergrund

Ein echter Server | Grafische Benutzeroberflächen fristeten auf UNIX-Systemen lange Zeit ein Schattendasein im Gegensatz zu den Produkten von Apple. UNIX war ursprünglich als Server-System gedacht, und bei einem Rechner, der nur Drucker ansteuern und Daten zur Verfügung stellen soll, ist eine grafische Oberfläche in erster Linie eine Verschwendung von Ressourcen. Später wurde dann ein System entwickelt, das auch der damals noch grundlegenden Struktur von UNIX entsprach. Der X11-Server ist in der Tat ein Server. Mit ihm werden die Berechnungen der Fenster vorgenommen, und die eigentliche Rechenarbeit findet nicht auf dem Rechner des Benutzers statt, sondern auf dem Server, an dem der Benutzer angemeldet ist. Es handelt sich also um eine Client-Server-Struktur, wie sie auch bei einem Abruf einer Webseite existiert. Der Client fordert Daten zur Darstellung an, die der X11-Server berechnet. Da sich die Leistungsfähigkeit der Rechner gegenüber dem Zeitpunkt der Entwicklung von X11 enorm entwickelt hat, werden heute der X11-Server und der Client auf ein und demselben Rechner betrieben. Welche Fenster dargestellt werden, wie diese aussehen und welche Funktionen sie haben, ist nicht Aufgabe des X11-Servers. Dies bestimmt ein so genannter Window-Manager, der auch für die grafische Oberfläche zuständig ist. Es gibt im Open Source-Bereich viele Window-Manager, die sich in der Leistungsfähigkeit und der Zielsetzung unterscheiden.

Ein wichtiges Projekt | Ein leistungsfähiges X11-System ist für Mac OS X ein sehr wichtiger Marktfaktor, denn gerade mit diesem System ist es leicht möglich, Software aus dem klassischen UNIX-Bereich unter Mac OS X zu verwenden. Der Aufwand für die Entwickler hält sich sehr in Grenzen, da sie auf die bekannten Bibliotheken zurückgreifen können und ihre Programme nicht so stark modifizieren müssen.

X11 konfigurieren

Unterstützung für Quartz | Zeitgleich mit der Markteinführung von Mac OS X 10.3 wurde auch von Apple die finale Fassung des eigenen X11-Servers bekannt gegeben. Es gibt mehrere Umsetzungen eines X11-Systems. Der Vorteil von Apples Eigenentwicklung besteht in erster Linie darin, dass der Window-Manager, der mitgeliefert wird, Unterstützung für die Quartz Engine von Mac OS X mitbringt. Somit fügt sich das System optisch etwas besser in die Aqua-Oberfläche ein, und vor allem ist das System schnell genug, um produktiv damit zu arbeiten. Wenn Sie mit Mac OS X 10.4 arbeiten, finden Sie das X11-System auf der Installations-CD und können es gegebenenfalls nachinstallieren. Den X11-Server finden Sie im Ordner **Dienstprogramme**, und Sie können ihn wie jedes andere Programm starten.

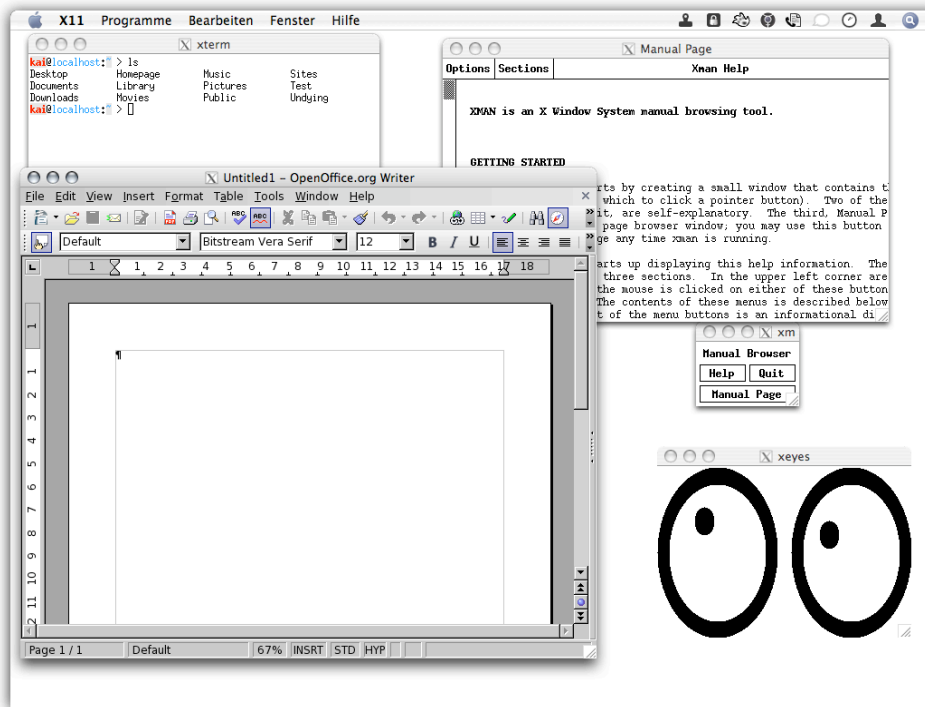


Abbildung 7: Der Quartz Window-Manager von Apple passt sich in die Optik von Mac OS X ein.

Die Datei .xinitrc | Die Voreinstellungen des X11-Servers werden in der Datei **xinitrc** vorgenommen. Sie können sich eine persönliche Datei erstellen und für jeden Benutzer individuelle Einstellungen vornehmen. Die Standardeinstellungen für X11 werden mit der Datei **xinitrc** im Verzeichnis **/etc/X11/xinit** festgelegt. Wenn Sie die dort vorhandene Datei **xinitrc** in Ihr persönliches Verzeichnis kopieren und in **.xinitrc** umbenennen (`cp /etc/X11/xinit/xinitrc ~/.xinitrc`), stehen Ihnen die in dieser Datei vorgenommenen Einstellungen beim nächsten Start von X11 zur Verfügung.

An und für sich gibt es nicht allzu viel, was Sie mit dieser Datei anfangen können. Die Konfiguration eines X11-Servers kann auf anderen Systemen zu einer etwas schwierigen Angelegenheit werden, insbesondere, da die angeschlossene Hardware sehr unterschiedlich ist. Da Apple über einen recht kleinen Marktanteil verfügt und außerdem die Entwicklung zentral von Apple gesteuert wird, erreichen Sie mit den Standardeinstellungen in der Regel die besten Ergebnisse. Die Einträge **xmodmap** in der Datei **xinitrc** konfigurieren angeschlossene Peripherie-Geräte wie die Maus oder die Tastatur für das X11-System. Mit dem Eintrag **xresources** können Sie das Aussehen des Mauszeigers und der Standardfarben für die dargestellten Fenster einstellen. Bei beiden Einträgen können Sie in Ihrem privaten Verzeichnis eigene Dateien anlegen, die die allgemeinen Einstellungen überschreiben. Dies ist aber, sollten Sie nicht über eine sehr exotische Hardwarezusammenstellung verfügen, selten notwendig.



Abbildung 8: Haben Sie bereits eine Datei `.xinitrc` erstellt, dann kann diese beim ersten Start von X11 übernommen werden.

Window-Manager | Von Relevanz können für Sie die letzten beiden Einträge der Datei `.xinitrc` sein. Mit dem Eintrag `xterm &` wird ein Terminal im X11-System gestartet, das die gleiche Aufgabe wie das Terminal von Apple hat. Kommentieren Sie diesen Eintrag aus, so wird beim Start von X11 kein Terminal mehr automatisch gestartet. Bei einem Update des X11-Systems von Apple gab es vereinzelt Probleme mit einer schon vorhandenen Datei `.xinitrc`. Sollten Sie nach einem Update auf Schwierigkeiten stoßen, kann eine Lösung darin bestehen, diese Datei zu löschen oder umzubenennen. Die letzte Zeile `exec quartz-wm` startet den von Apple entwickelten Window-Manager. Die Aufgabe von `quartz-wm` besteht darin, die Fenster der aktiven Programme auf dem Bildschirm zu positionieren. Der Vorteil von `quartz-wm` besteht darin, dass sich das Programm einigermaßen nahtlos in die Optik von Mac OS X einfügt. Es gibt noch eine ganze Reihe weiterer Window-Manager, die Sie anstelle von `quartz-wm` verwenden können.

.bashrc | Die Shell, die im Fenster `xterm` gestartet wird, entnimmt ihre Voreinstellungen nicht der Datei `~/bash_profile`, sondern `~/bashrc`. Sie können diese separat erstellen oder, sofern Sie die Einstellungen aus `~/bash_profile` verwenden möchten, mit `ln -s ~/bash_profile ~/.bashrc` einen symbolischen Link erstellen. Achten Sie ferner darauf, dass das Verzeichnis `/usr/X11R6/bin` in der Variable `PATH` enthalten ist, da Ihnen sonst die meisten grundlegenden Programme des X11-Servers nicht zur Verfügung stehen.

Weitere Einstellungen | In den Einstellungen des X11-Servers können Sie unter anderem festlegen, ob dieser den gesamten Bildschirm nutzen oder noch Platz für die Aqua-Oberfläche lassen soll. Bei einigen Programmen, die viele Fenster öffnen, kann die Darstellung über den gesamten Bildschirm die Arbeit erleichtern. Die Farbpalette, die der X11-Server verwenden soll, können Sie hier auch festlegen. Reduzieren Sie die Farbanzahl, so sinkt auch der Rechenbedarf für die Darstellung der Fenster ein wenig. Soll sich X11 an die Vorgaben der Systemeinstellungen halten, so belassen Sie die Einstellung bei **Vom Monitor**. Der Aufruf von Programmen erfolgt wie beim Terminal auch durch die Angabe des Pfades zum auszuführenden Programm. Sie können einigen Programmen auch Tastaturkürzel zuweisen.

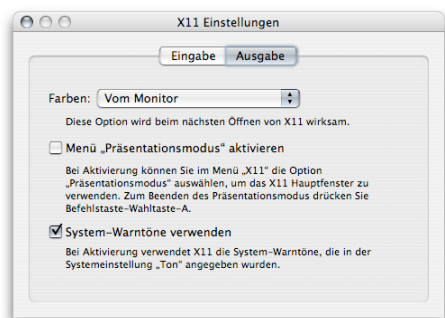


Abbildung 9: Der X11-Server kann auch den ganzen Bildschirm ausnutzen.

open-x11 | Die meisten Programme, die X11 benötigen, sollten Sie über das Menü des X11-Servers starten. Sollten Sie aber einmal ein Programm vom Terminal aus aufrufen wollen, so können Sie mit

dem Befehl `open-x11`, gefolgt von dem zu startenden Programm, dafür Sorge tragen, dass auch der X11-Server gestartet wird, bevor das Programm in Aktion tritt. Wurde X11 korrekt installiert, so gehört das Verzeichnis `/usr/X11R6/bin` zu Ihrer `PATH`-Variable. In diesem Verzeichnis finden sich neben den zu X11 direkt gehörenden Programmen auch einige nützliche oder weniger nützliche Applikationen. Das Programm `xeyes` ist ein Klassiker ohne Nutzwert und stellt lediglich zwei Augen dar, deren Pupillen dem Mauspfel folgen. Mit dem Programm `xman` lassen sich Manpages innerhalb des X11-Systems anzeigen, und es lässt sich vergleichsweise komfortabel navigieren. Ein eher funktionaler, wenn nicht gar spartanischer Editor wird mit `xedit` aufgerufen.

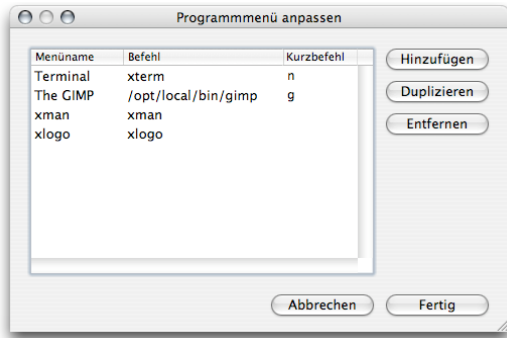


Abbildung 10: Programmen und Befehlen können Sie ein Tastenkürzel zuweisen.

GNU Network Object Model Environment – GNOME

GNOME verwenden | Bei GNOME handelt es sich um eine unter Linux sehr beliebte Benutzeroberfläche, die Sie auch mit den DarwinPorts oder Fink installieren können. Zu GNOME gehören auch eine Reihe leistungsfähiger Programme wie der Dateimanager Nautilus, die E-Mail- und Kontaktverwaltung Evolution oder der Browser Galeon. Haben Sie GNOME mit den DarwinPorts installiert, dann müssen Sie in der Datei `~/xinitrc` zwei Änderungen vornehmen. Zunächst müssen Sie die Zeile `exec quartz-wm in quartz-wm —only-proxy &` ändern. Dies führt dazu, dass der Window-Manager von Apple als Instanz zwischen dem X11-Server und dem Window-Manager von GNOME fungiert. Das Kaufmanns-Und führt dazu, dass der Prozess im Hintergrund arbeitet, und die Löschung von `exec` stellt sicher, dass auch die nachfolgenden Befehle abgearbeitet werden. Fügen Sie nun zunächst die Zeile `/opt/local/bin/gnome-wm &` am Ende ein, um den Window-Manager von GNOME im Hintergrund zu starten. Die Zeile `exec /opt/local/bin/gnome-session > /dev/null` aktiviert GNOME selbst, schreibt die Ausgaben des Startvorgangs hingegen nach `/dev/null`. Diese Beschreibung beruht natürlich auf einer Installation von GNOME, die Sie mit den DarwinPorts vorgenommen haben. Bei Fink müssen Sie die Pfade entsprechend anpassen.

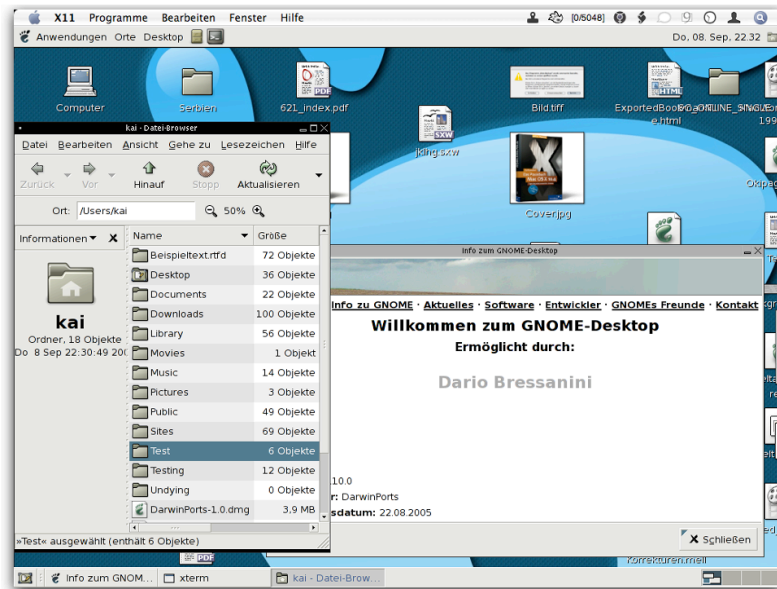


Abbildung 11: Mit GNOME steht eine leistungsfähige grafische Oberfläche Vlinebreak zur Verfügung.

Das K Desktop Environment – KDE

startkde | Eine ebenso umfangreiche wie flexible Alternative ist das K Desktop Environment, KDE. Ähnlich wie bei GNOME handelt es sich nicht nur um ein kleines Programm, sondern um ein umfangreiches Projekt.

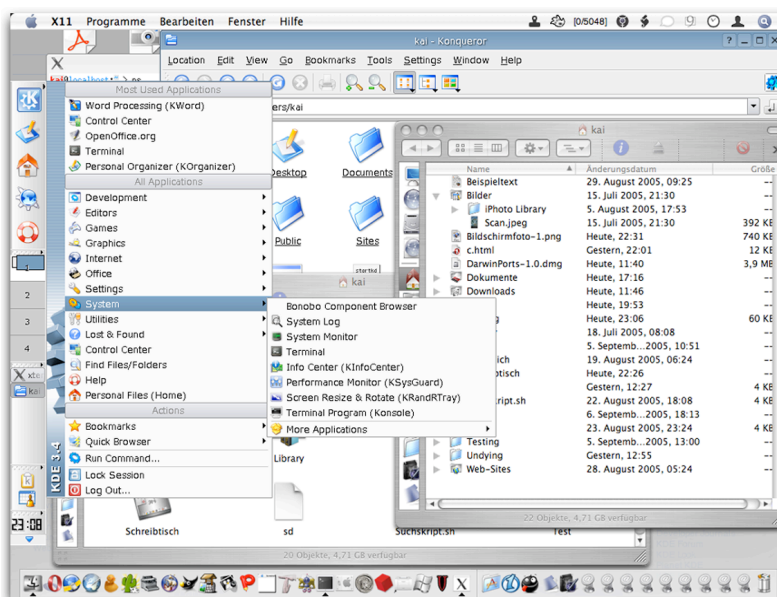


Abbildung 12: KDE bringt ein eigenes Dock zum Start von Programmen mit.

Im Zuge von KDE wurde nicht nur eine grafische Oberfläche, sondern auch ein eigenes Office-Paket, Koffice, entwickelt. Die Installation von KDE mit den DarwinPorts oder Fink ist etwas umfangreicher, und gegebenenfalls müssen Sie einige Einschränkungen akzeptieren. Insbesondere die Tonausgabe funktionierte zur Drucklegung dieses Buches nicht zufrieden stellend. Um KDE mit X11 nutzen zu können, müssen Sie auch hier den Window-Manager von Apple als Proxy starten (`quartz-wm — only-proxy &`). KDE bringt seinen eigenen Fenster-Manager mit, und Sie müssen ihn mit der Zeile `export KDEWM=kwin` in der Datei `~/xinitrc` für KDE vorgeben. Auf diese Umgebungsvariable greift das KDE-System zurück. Zum Start wird ein eigenes Skript verwendet. Dieses können Sie durch die Zeile `exec /opt/local/bin/startkde > /Users/kai/kde.log 2>&1` aufrufen. Etwaige Fehlermeldungen werden in der Datei **kde-log** im persönlichen Ordner des Benutzers **kai** gespeichert. Im Zusammenhang

mit dem KDE-Projekt sind neben der Office-Suite Koffice auch die Programme zur Erstellung von Webseiten erwähnenswert. Insbesondere der Editor Quanta ist eine sehr leistungsfähige Entwicklungsumgebung für die Erstellung von HTML-Dateien und PHP-Skripten. Am Rande: Der Browser Konqueror bildet das Grundgerüst von Apples Safari.

Weitere Window-Manager

Bunte Vielfalt | Neben KDE und GNOME gibt es eine ganze Reihe weiterer Window-Manager, die Ihnen die Verwaltung von Fenstern unter X11 ermöglichen. Sie unterscheiden sich von dem doch recht spartanischem `quartz-wm` durch eine große Anzahl an Funktionen und Fähigkeiten. Wenn Sie ein großes Programm wie OpenOffice.org unter X11 nutzen möchten, dann kann die Installation eines schlanken und schnellen Window-Managers wie zum Beispiel Fluxbox, der Ihnen sowohl bei den DarwinPorts als auch bei Fink zur Verfügung steht, nützlich sein. KDE und GNOME beanspruchen vergleichsweise viel Rechenzeit, und ein so umfangreiches Programm wie OpenOffice.org arbeitet vergleichsweise langsam. Mit Fluxbox oder Enlightenment (ebenfalls in den DarwinPorts und Fink) sparen Sie sowohl Arbeitsspeicher als auch Rechenzeit, und das eher spartanische Konzept dieser beiden Programme kommt Ihrer Arbeitsgeschwindigkeit unter OpenOffice.org zugute.

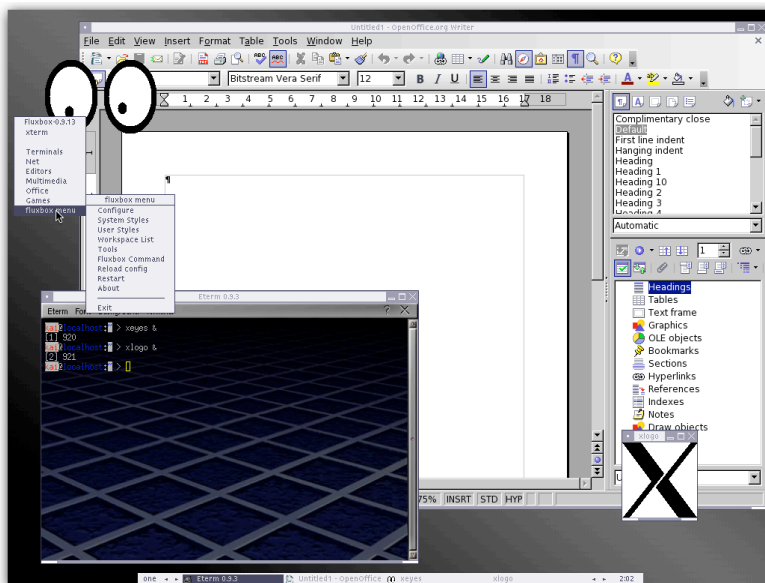


Abbildung 13: Mit Fluxbox steht ein sehr schlanker und schneller Window-Manager zur Verfügung.

Themes | Die Konfigurationsdateien dieser Programme finden Sie in unsichtbaren Verzeichnissen in Ihrem persönlichen Ordner. So können Sie durch die Anpassung der Datei `~/.fluxbox/menu` das Ausklappenmenü von Fluxbox ergänzen und so über die rechte Maustaste direkt Programme wie eben OpenOffice.org starten. Neben vielen kleinen Details in Bezug auf die Bedienung der Maus oder die Positionierung der Fenster können Sie auch das Erscheinungsbild mit Hilfe von so genannten Themes verändern. Auf der Webseite <http://www.themes.org> finden Sie für die meisten Window-Manager vorgefertigte Erscheinungsbilder. Diese meist sehr kreativen Themen können Sie beispielsweise für Fluxbox im Verzeichnis `~/.fluxbox/styles` installieren und anschließend die Optik der Fenster unter X11 verändern.

Ein Best of Open Source

Bei der Vielzahl der Programme geht der Überblick schnell verloren. Diese Auswahl stellt Ihnen einige der leistungsfähigsten Programme vor.

Ein weites Feld | Freie Software ist mittlerweile zu einem fast unüberschaubaren Feld geworden. Viele Programme und Projekte wurden von den Entwicklern in ihrer Freizeit begonnen, dann aber mangels dieser wieder eingestellt. Dieses Kapitel soll Ihnen einen Überblick über nützliche und hilfreiche Programme aus diesem Bereich verschaffen und bei dem einen oder anderen Programm ein wenig Hilfestellung geben.

OpenOffice.org

Ein Paradeponer unter X11 | OpenOffice.org ist wohl derzeit eines der zugkräftigsten Programme mit offenem Quellcode. Nach dem sehr umfangreichen Download können Sie das Programm installieren. Ein Installationsprogramm wird mitgeliefert. Dies ruft, so es notwendig ist, noch zwei weitere Installationsprogramme auf, die von OpenOffice.org benötigte Software installieren. OpenOffice.org ist noch nicht an die Aqua-Oberfläche angepasst und benötigt einen X11-Server. Im Ordner OpenOffice finden Sie ein Skript, das nach einem Doppelklick zuerst X11 und dann OpenOffice.org startet.

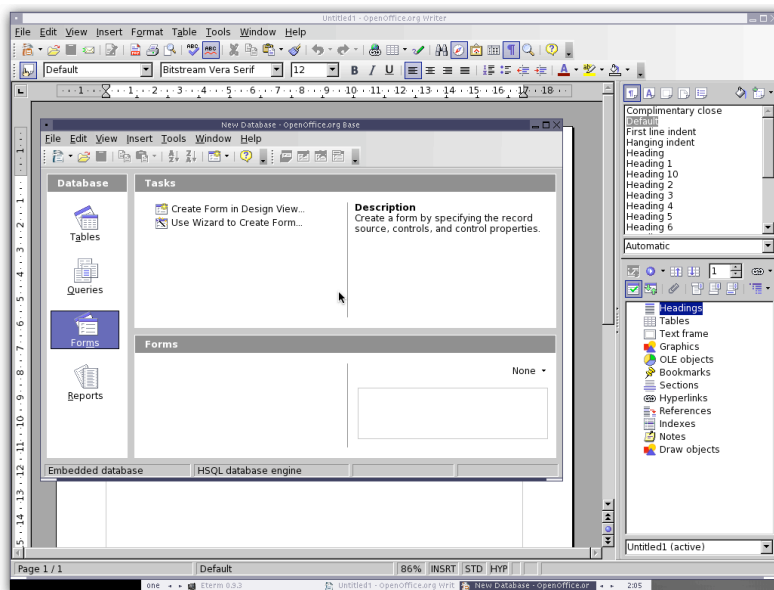


Abbildung 14: OpenOffice.org enthält eine umfangreiche Textverarbeitung und ab der Version 2.0 eine leistungsfähige SQL-Datenbank.

2.0 | Zu Drucklegung dieses Buches war die Version 2.0 von OpenOffice.org kurz vor der Fertigstellung. Die neue Version bietet neben vielen kleinen Aktualisierungen und Verbesserungen auch ein neues Datenbank-Modul. Dieses Modul integriert sich in die anderen Module, und Sie haben zum Beispiel dadurch von der Textverarbeitung aus direkten Zugriff auf die Inhalte der Datenbanken.

NeoOffice | Während die reguläre Version von OpenOffice.org auf X11 beruht, hat sich das Projekt NeoOffice das Ziel gesetzt, OpenOffice.org weitgehend nahtlos in die Aqua-Oberfläche von Mac OS X zu integrieren. Die Entwicklung dieses Projekts nimmt naturgemäß einen etwas anderen und auch langsameren Verlauf, bietet Ihnen aber eine kostenlose und fast vollständige Office-Suite.

Bezug: <http://www.openoffice.org>

Bezug: <http://www.neooffice.org>

The GIMP

Ein Ersatz für Photoshop | Das GNU Image Manipulation Program, also The GIMP, ist unter Linux zu einem ähnlichen De-facto-Standard geworden, wie Photoshop es wohl unter Mac OS X ist. Trotzdem kann sich der Einsatz von The GIMP unter Mac OS X lohnen, insbesondere da es kostenlos ist. Wenn Sie The GIMP installiert haben und zum ersten Mal unter X11 aufrufen, legt es in Ihrem persönlichen Verzeichnis ein Verzeichnis **.gimp** an, das Ihre persönlichen Einstellungen für das Programm enthält. Dieser Vorgang kann etwas dauern.

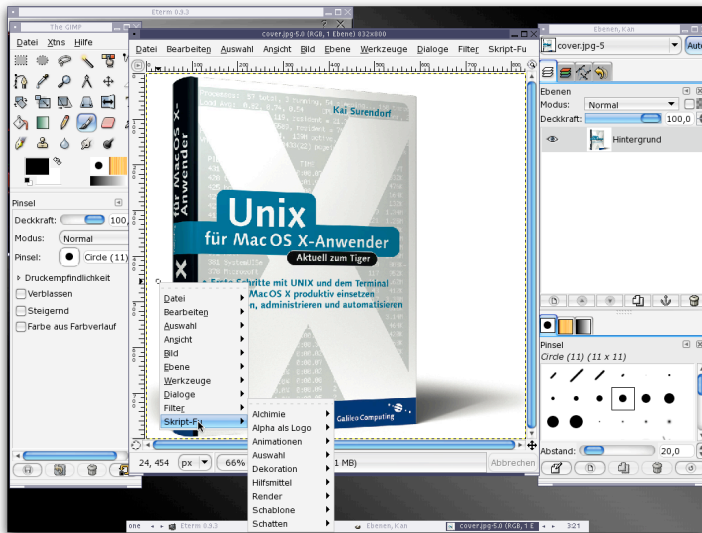


Abbildung 15: The GIMP kann durch die Vielzahl der Fenster und Paletten verwirrend wirken.

Die rechte Maustaste | Nach dem ersten Start kann die Bedienung von GIMP recht verwirrend wirken. Die Vielzahl der geöffneten Fenster und das Fehlen einer Menüleiste am oberen Bildschirmrand tragen nicht zur Bedienungsfreundlichkeit bei. GIMP nutzt die rechte Maustaste sehr stark. Es erscheint ein umfangreiches Menü, wenn Sie mit **File** ein Bild öffnen und mit der rechten Maustaste ins Bild klicken (oder mit **ctrl** und einem Mausklick bei der Verwendung einer Maus mit nur einer Taste). In diesem Menü finden Sie die Funktionen für die Zwischenablage, zum Speichern und Exportieren des Bildes und zur Anwendung von Filtern.

Einsatz von Ebenen | GIMP verfügt über die Funktion, ein Bild in mehrere Ebenen zu unterteilen und diese wie auch in Photoshop einzublenden oder über- und untereinander zu legen. Mit GIMP wird auch eine umfangreiche Sammlung von Mustern, Spritzen und Pinseln mitgeliefert. Weitere Muster und Pinsel finden Sie auf der Webseite des GIMP-Projekts <http://www.gimp.org>. GIMP verfügt eigentlich über alle Funktionen, die für ein Programm zur Bildbearbeitung heute Standard sind.

GIMP erweitern | Je nachdem, ob Sie GIMP mit Fink oder DarwinPorts erstellt haben, finden Sie unter **/sw** oder **/opt/local** im Verzeichnis **share** ein Verzeichnis **gimp**. In diesem ist noch einmal ein weiteres Verzeichnis angelegt, das als Namen die Nummer der von Ihnen installierten GIMP-Version trägt. In diesem Verzeichnis schließlich finden Sie, jeweils in Verzeichnisse unterteilt, die Pinsel und Muster. Ein besonderes Verzeichnis ist **scripts**.

Script-Fu | In diesem sind die Skripte abgelegt, mit denen Sie Bilder in GIMP ähnlich wie bei Photoshop mit Aktionen automatisch bearbeiten können. Diese Skripte sind in einer Script-Fu genannten Programmiersprache erstellt worden. Script-Fu lehnt sich an die Sprache Lisp an. Wenn Sie im Internet nach weiteren Skripten suchen und diese verwenden möchten, dann können Sie sie in diesem Verzeichnis ablegen, und sie stehen anschließend allen Benutzern Ihres Rechners zur Verfügung.

Wenn Sie zusätzliche Skripte und Pinsel nur für sich selbst nutzen möchten und sie nicht anderen Benutzern Ihres Rechners bereitstellen, speichern Sie diese im Verzeichnis **.gimp** in Ihrem persönlichen Verzeichnis ab. Script-Fu kann auch von der Shell aus genutzt werden, und Sie können sich deren Funktionen für Skripte zunutze machen.

Kaum Unterstützung für CMYK | GIMP kann ein Ersatz für kommerzielle Programme sein, wenn Sie sich in seine Bedienung einarbeiten und Bilder nur elektronisch publizieren möchten. Beim Einsatz in der Druckvorstufe hat GIMP den Nachteil, dass es das CMYK-Format kaum unterstützt und keine Farbsynchronisation mit Druckern und Scannern bietet. Zwar sind solche Funktionen in zukünftigen Versionen geplant, aber bis dahin wird es wohl noch einige Zeit dauern.

Bezug: **Darwinports**

Sodipodi

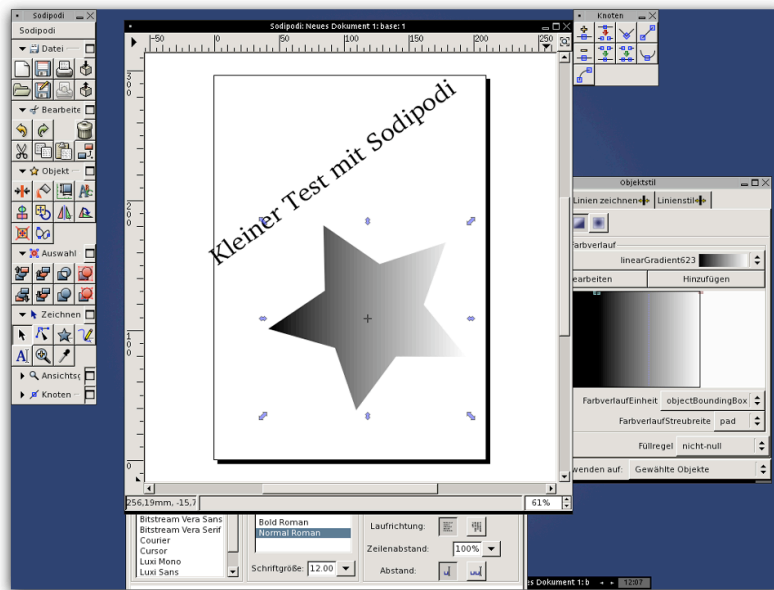


Abbildung 16: Mit Sodipodi steht ein leistungsfähiges Zeichenprogramm zur \linebreak Verfügung.

Scalable Vector Graphics | Während es sich bei GIMP um das Pendant zu Adobe Photoshop handelt, steht Ihnen mit Sodipodi ein kleiner Konkurrent zu Adobe Illustrator zur Verfügung. Bei Sodipodi handelt es sich um ein vektor-basiertes Zeichenprogramm, und Sie können Objekte frei auf der Zeichenfläche positionieren, manipulieren und kombinieren. Dabei verfügt Sodipodi gewiss nicht über die Funktionsvielfalt des Illustrator, es ist dafür aber etwas einfacher zu bedienen. Zur Gestaltung eines kleinen Titelblatts oder eines kleinen Werbe-Flyers ist das Programm gut geeignet. Es setzt wie auch GIMP das Kontextmenü und die rechte Maustaste intensiv ein, und Sie sollten sich nicht von den spartanischen Menüleisten abschrecken lassen. Beim Dateiformat setzt Sodipodi nicht auf eine Eigenentwicklung, sondern auf den offenen Standard Scalable Vector Graphics. Dieses Dateiformat wird auch von den Produkten von Adobe unterstützt, so dass Sie die mit Sodipodi erstellten Dateien auch mit anderen Programmen verwenden können. Bei dem Projekt Inkscape handelt es sich um eine Abspaltung von Sodipodi mit einer leicht modifizierten Zielsetzung.

Bezug: **DarwinPorts**

Blender3D

Animationen erstellen | Wenn Sie dreidimensionale Szenarien erstellen möchten, dann wird Ihnen Blender3D vielleicht gefallen. Das Programm ist bei weitem nicht so umfangreich wie Cinema4D,

aber es verfügt im Gegensatz zu POV-Ray über eine grafische Oberfläche und ist auch in der Lage, dreidimensionale Animationen zu erstellen. Auf der Webseite des Projekts finden Sie eine Version für Mac OS X, die nach dem Download sofort einsatzfähig ist. Blender3D unterstützt neben Quicktime auch die Möglichkeit, Szenen und Animationen als ausführbare Programme zu speichern, die ohne zusätzliche Software auf jedem Rechner abgespielt werden können.

Bezug: <http://www.blender3d.org>

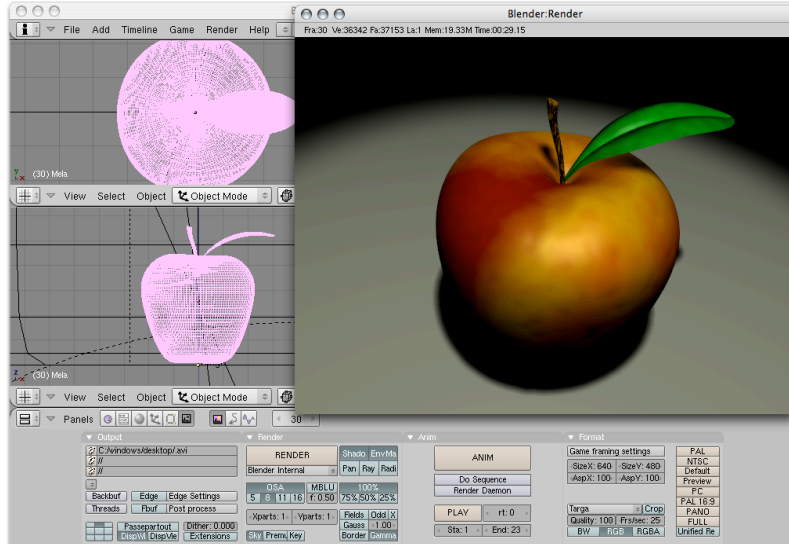


Abbildung 17: Mit Blender3D können anspruchsvolle Animationen erzeugt werden.