Mac OS X Consoliero
Einstieg und Gebrauch des Terminals für Power User.

Mac OS X Consoliero – Teil 1: Darwin, der Mach Kernel und das BSD Subsystem.

Christoph Müller, PTS

Mac OS X Consoliero – Teil 1:
Darwin, der Mach Kernel und das BSD Subsystem.

Inhaltsverzeichnis

Mac OS X Consoliero – Teil 1:

Darwin, der Mach Kernel und das BSD Subsystem.

Max OS X Übersicht

Die Mac OS X 10.2 Architektur ist auf elf Ebenen aufgebaut (Abbildung 1.1). Eine davon ist Darwin. Darwin ist ein Open Source Core Operation System. Dies bedeutet, dass der Code welcher Darwin zugrunde liegt, öffentlich verfügbar ist. Mehr Informationen dazu unter: http://developer.apple.com/macosx/architecture/index.html /. In der Standard OS X Architektur Modell wird Darwin als eine Ebene dargestellt. Tatsächlich ist Darwin aber aus zwei Teilen zusammengesetzt. Zum einen aus dem Mach Kernel und zum andern aus dem BSD Subsystem. Dies ist insofern wichtig, weil es das erste mal ist, dass ein Mac OS Kernel basiert ist. Schauen wir uns diese Technologie einmal an und betrachten welchen Einfluss es auf die „total neue Benutzer- freundlichkeit von Mac OS X" hat. Das meiste was hier entdeckt wird, ist vermutlich neues Territorium. Dies auch für die grössten Mac.

Abbildung 1.1

Der Mach Kernel

Ein Kernel ist ein kleines Stück Code welches die Steuerung zwischen Software und Hardware kontrolliert. Der Kernel fungiert als eine Art Torwächter für alle Prozesse und Programme. In einem Kernel basierten System, kann nur der Kernel direkt auf die Hardware zugreifen (I/O Systeme, RAM, Peripherie, usw.). Weil es nur einem Stück Software erlaubt ist kritische Aufgaben auszuführen, können einzelne Programme durch ihr Abstürzen nicht mehr das ganze System blockieren. Wie wir alle wissen, war das bei älteren Mac OS Releases nicht der Fall. Natürlich kann auch der Mach Kernel des Mac OS X abstürzen. Dies manifestiert sich dann in folgender legenderen Fehlermeldung: „Kernelpanic".

Darwin benutzt den Mach V.3 Kernel, welcher hoch optimiert und unter extremen Bedingungen getestet wurde. So ist sichergestellt, dass der Kernel so stabil wie möglich ist. Dies ist deshalb so wichtig, weil alle andern Komponenten auf ihm aufbauen (Abbildung 1.1). Bei älteren Mac OS Versionen, konnte jede Applikation direkt auf den Systemspeicher zugreifen. Unter OS X laufen solche Vorgänge über den Mach Kernel und bieten so die nötige Stabilität und Kontrolle.
Zum Vorteil der Stabilität kommt ein weiter Benefit für alle anderen OS Komponenten hinzu. Werden neue Kernelfunktionen implementiert, sind die Vorteile allen anderen Komponenten des Systems zugänglich, ohne das die Programme des Endbenutzer eines Updates bedürfen. Hier ein paar Technologien welche der Mach Kernel dem Rest des Mac OS X Systems zur Verfügung stellt:

Protected Memory

In älteren Mac OS Versionen konnte jedes Programm Speicher vom System anfordern. Es war dem Programm überlassen zu kontrollieren, das es nicht in den Speicher ausserhalb des verfügbaren Bereichs kommt. Das System konnte nicht verhindern, dass ein Programm quasi im Speicher wildert. In Mac OS X läuft jedes Programm in seinem eigenen „Memory Space". Es hat keinen Zugriff auf Speicher von anderen Programmen oder des Systems (ausser es läuft mit erweiteter Berechtigung).

Virtual Memory
Der Mach Kernel organisiert alle Speicherbedürfnisse und kann dynamisch Speicher zuweisen. Dies ersetzt das benutzerdefinierte „Swap File" von Mac OS 9 und frühere Versionen. Deswegen fehlt auch ein Kontrollfeld „Speicher".

Pre-emptiv Multitasking
Unter Mac OS X können Programme den Computer nicht mehr blockieren indem sie den Prozessor alleine auslasten.

Network Kernel Extension
Das fortschritliche Netzwerkmodell des Darwin Kernel lässt es zu dass „Protokol Stacks" dynamisch geladen werden können. Ebenso kann der Netzwerkverkehr in „Real Time" beobachtet und modifiziert werden. So kann man unter Mac OS X komplexe Netzwerkapplikationen wie etwa Firewalls oder Netzwerkmonitor einfach erstellen und ausführen.

File System Support
HFS+ ist das Standard File System von Mac OS 9 und Mac OS X. Zusätzliche File Systeme können dem Kernel als Module hinzugefügt werden. und sind allen Programmen innerhalb des Betriebssystem zugänglich. Dabei spielt es keine Rolle ob ein Programm die fremde File Struktur versteht.

Ein möglicher Benefit des kernelbasierten OS ist in dieser Aufzählung nicht genannt. Er könnte aber das Gesicht von Macintosh für immer ändern. Seit beginn wurde der Macintosh mit Motorola Prozessoren bestückt und der Code auf diese Prozessoren optimiert. Warum nicht mit Intel Prozessoren arbeiten? Der Darwin Kernel ist bereits auf Intel Plattformen portiert. Es ist technologisch gut denkbar, dass Mac OS X auf Intel oder AMD Prozessoren laufen kann, da nur der Kernel ersetzt werden muss. Fact ist: Der erste „final release" des Mac OS X Server, der „Rhapsody developer release" lief auch auf Intel Plattformen. Allerdings, von Cupertino hört man nichts offizielles zu diesem Thema.

Das BSD Subsystem

Über dem Darwin Kernel liegt das BSD (Berkley Software Distribution) Subsystem. Dieses Stück des Mac OS X ist so konzipiert worden, dass es der Benutzer nie zu Gesicht bekommt. Ausser er will es. Die Idee dieser Serie ist aber dieses BSD genauer zu Gesicht zu bekommen.
Also was ist BSD? Die einfachste Antwort ist: Eine Sammlung von kleinen Programmen welche das Ganze wie eine Art Unix aussehen lässt. Dies führt zur Frage: was ist Unix? Dies ist schon ein bisschen komplizierter zu beantworten.
Unix ist ein Multitasking Betriebssystem, welches von Bell Labs in den 70er Jahren entwickelt wurde. Es wurde mit dem Gedanken ein stabiles und für Entwicklungen offenes Betriebssystem zu sein, entwickelt. Heute ist Unix in weiten Kreisen als hoch kryptisches OS angesehen und die meisten Windows System Administratoren suchen das Weite wenn sie nur den Namen Unix hören. Unix ist eine Zusammenarbeit aus hunderten von kleinen Programmen welche Zugriff auf die Festplatten, auf Files und andere OS Funktion erlauben.
Es gibt zwei wichtige Unix Distributionen. Das erste ist das BSD, das zweite ist das System V. Welches durch AT&T entwickelt wurde und im Gegensatz zu BSD als kommerzielles Unix augelegt wurde. Die meisten Unterschiede sind klein, die Philosophie hinter den beiden liegt aber diametral zueinander. BSD wird bevorzugt von Wissenschaftlern, während System V in kommerziellen Unix Distributionen benutzt wird. Apple wurde viel gefragt, warum es als Basis nicht Linux gwählt hat. Hierzu muss gesagt

sein, dass Linux eigentlich nur ein Kernel ist und nicht ein Betriebssystem. Der Linux Kernel ist extrem instabil aus Sicht der Entwickler. Linux Benutzer müssen ihr System mehrmals im Jahr updaten. Für einen Poweruser ist das so in Ordnung. Hingegen für den profesionellen Gebrauch in Firmen wie auch für den unbedarften Benutzer zuhause ist dies ein Problem. Linux Benutzer finden sich jedoch schnell im Mac OS X zurecht, weil die Software welche auf dem Kernel liegt in weiten Teilen die gleiche ist wie sie in vielen Linux Distributionen verwendet wird.

Unter BSD laufen gewisse Dinge anders als im klassichen Mac OS. Beispielsweise das Anzeigen von Ordnerinhalten.
Schauen wir uns das ganze mal an. Öffnen wir unsere Konsole und gedenken dem Moment. Dieser Schritt markiert das erste Mal dass ein Mac OS ein Kommando Zeilen Interface hat. Ja, ja.. früher habt ihr über die Windows Benutzer gelacht, ihre command.com Fenster, ihren Bootvorgang mit weissen Ziffern auf schwarzem Hintergrund :) und so weiter. Und die Windows Benutzer können sich trotzdem nicht ins Fäustchen lachen. Das gute alte command.com und das Mac OS X Terminal haben nur eines gemeinsam: man tippt bei beiden auf der Tastatur. Das Terminal ist sehr, sehr, sehr viel mächtiger als das command.com. Es ist eigentlich kein fairer Vergleich.

Das Terminal

Für die, welche es noch nicht gesehen haben, das Terminal befindet sich unter:

Programme/Dienstprogramme/Terminal.

Das Terminal ist langweilig weiss und öde. Deswegen empfehle ich hier schnell die Einstellung zu ändern. Im Menu Terminal -> Fenstereinstellungen findet man im Popup Menu die Einstellung „Farbe". Ich habe während langen Terminal Nächten folgende Einstellung als gut „für mich" empfunden (Abbildung 1.2).

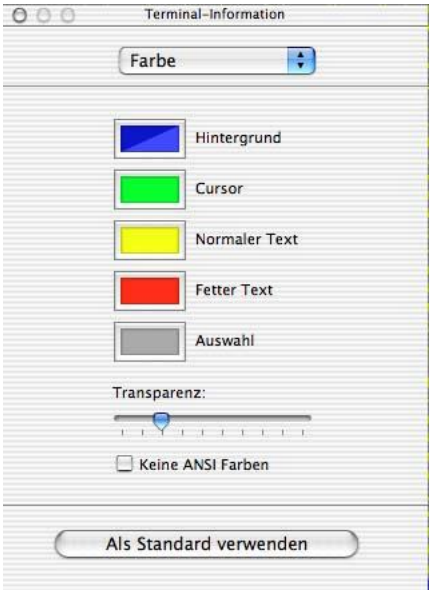

Abbildung 1.2

Hintergrund: blau
Cursor: grün
Normaler Text: gelb
Fetter Text: rot
Auswahl: grau
Transparenz: ein wenig.
Der Einfluss meiner Befehle in der Konsole auf die Programme kann so besser beobachten werden.

In einem weiteren Menu im gleichen Popup Menu findet man noch die „Fenster Einstellung". Hier habe ich das Fenster als Standard ein bisschen grösser gemacht (Abbildung 1.3). Dies hilft vorallem bei langen Pfaden oder bei der Kontrolle von Prozessen, da diese abgeschnitten sind wenn man das Fenster später öffnet.
Nicht vergessen am Ende der Einstellungen den Button „Als Standard verwenden" zu betätigen sonst gelten die Einstellungen nur für das aktuelle Fenster.

Die Unix Shell

Wie wir uns vorstellen können, reicht ein Terminal alleine noch nicht um mit der Rechner zu kommunizieren. Das Terminal braucht jemand um zu reden, und dieses Ding wird Shell genannt. Unix Shell's ermöglichen die textbasierte Interaktion zwischen dem Benutzer und dem Rest des Betriebssystems. Es gibt verschiedene Shell's und die meisten Unix Distributionen kommen mit einigen davon. Auch Mac OS X macht hier keine Ausnahme. Welche Shell man wählt ist seinem persönlichem „Geschmack" überlassen. Alle Shell's unterscheiden sich durch den Syntax durch welche Befehle eingegeben werden. Einige haben einen erweiterten Befehlssatz oder sonst erweiterte Funktionalität. Apple hat als Standard die „tcsh" Shell gesetzt.

Das Unix File System

Also legen wir los. Wir gehen wieder in unser Terminal Fenster und sehen dort schon, dass wir uns mit „Welcome to Darwin!" begrüsst werden und nicht etwa „Willkommen zu Mac OS X". Zur Erinnerung: Wir befinden uns mit unseren Interaktionen oberhalb des Kernels.

Wer jetzt erwartet ich würde über HFS+, Ordner und Files eine Einführung geben hat sich getäuscht, aber manchmal muss ich doch auf solch auf den ersten Blick trivial wirkenden Dinge eingehen. Da sie sich manchmal ziemlich von früheren Mac OS Versionen unterscheiden. Hier kurz ein paar kleine Basic's:

Wenn man das Terminal startet und ein neues Fenster öffnet startet man im sogenanten „home directory". Das ist der Ordner in dem der Finder seinen eigenen Ordner öffnet wenn man auf „Privat" klickt.

Das „working directory" ist das Verzeichniss in dem man arbeitet. Alle Befehle die man eingibt beziehen sich immer auf das aktuelle „working directory".

Hat man mehrer Terminal Fenster offen, können alle Fenster ein eigenes, voneinander verschiedenes „working directory" haben.

Alle „directorys" haben einen hierarchischen Aufbau. Also so wie früher: ein Ordner indem ein Ordner ist und so weiter. Der oberste Ordner, also quasi die Festplatte beim klassichen Mac OS. Bei Unix ist es das sogenannte „root directory" geschrieben als slash „/".

Absolute und relative Pfadnamen

Der Weg zu einem Ordner oder einem File innerhalb des Filesystems wird als Pfad bezeichnet. Dieser Weg ist ein „absoluter Pfadname" der den Weg beschreibt vom „root directory" zu dem File zu dem man möchte. Ordner und Subordner werden mit „Slash" also „/" voneinander getrennt. Es dürfen keine Leerschläge in einem Pfad sein.

/Users/pts/Documents (Bsp. Absoluter Pfadname)

Ein relativer Pfadname gibt das Zielort an und zeigt den Weg vom Standort aus. Also nicht vom „root diretory" aus. Oder anders augedrückt, relative Pfadnamen beginnen im „working directory". Sie haben kein „slash" da es kein „root directory" braucht.

Documents/Fotos (Bsp. Relativer Pfadname)

Das „working directory" anzeigen und wechseln
Hier folgen ein paar Befehle für die Navigation im „directory tree". Jeder Befehl in der Shell wird mit „Return" ausgeführt. Bitte beachten Sie, dass alle in Courier geschriebenen Zeilen, Kopien vom Terminal sind. Wobei „TitanSurfer" der Name des Rechners und „pts" der Benutzername ist. Dies wird bei Ihnen definitiv anders aussehen (Ausser Ihr Rechner heisst gleich und sie haben den selben Benutzername wie ich angegeben habe).

pwd
Um herauszufinden in welchem directory man ist, benutzt man den pwd Befehl. Der Befehl pwd ist eine Abkürzung für „print working directory"

```
[TitanSurfer:~] pts% pwd
/Users/pts
```

Aufmerksame Leser haben jetzt gemerkt, dass pwd ein absoluter Pfadname von meinem working directory ausgibt. Also: „/" root directory, Ordner „Users", Ordner „pts".

Wenn man jetzt im Finder schaut sieht man, dass der Finder hier etwas korrigiert (Abbildung 1.3). Er ersetzt nämlich das directory „Users" durch die deutsche Übersetztung „Benutzer". Eher unsympatisch aber ist halt so.

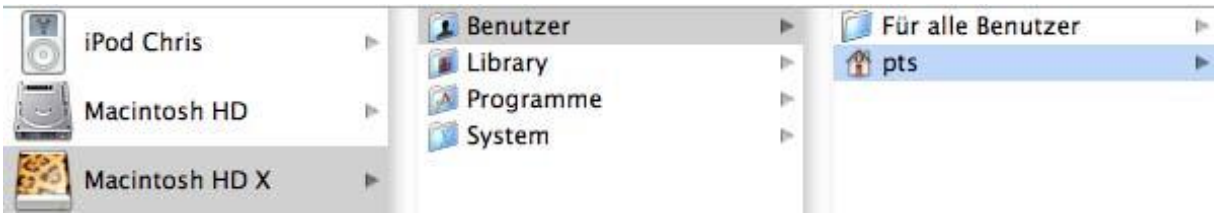

Abbildung 1.3

cd
Der Befehl cd ist eine Abkürzung für „change directory". Mit diesem Befehl können wir in jedes directory auf der Festplatte erreichen solange wir die Berechtigung haben.

```
[TitanSurfer:/] pts% pwd
/
[TitanSurfer:/] pts% cd Users/pts/Documents/
[TitanSurfer:~/Documents] pts% pwd
/Users/pts/Documents
[TitanSurfer:~/Documents] pts%
```

Was wurde gemacht:
1 Zeile: wo sind wir (pwd).
2 Zeile: Ausgabe: „/" (also auf dem root directory).
3 Zeile: Dem Terminal gesagt ich möchte das working directory auf meinen Dokument Ordner setzen.
4 Zeile: Das wurde gemacht und das Terminal wartet auf eine Eingabe. Ich überprüfe mit pwd.
5 Zeile: Die Ausgabe vom Terminal zeigt mir den absoluten Pfadnamen an.
6 Zeile: Das Terminal wartet auf eine neue Eingabe.

Man kann auch einfach in einer Ordnerhierarchie eine Ebene nach oben gehen, dies funktioniert mit dem Befehl:

cd .. (Nach cd folgt ein Leerschlag, erst dann die beiden Punkte)

```
[TitanSurfer:~/Documents] pts% pwd
/Users/pts/Documents
[TitanSurfer:~/Documents] pts% cd ..
[TitanSurfer:~] pts% pwd
/Users/pts
```

Hier ist ersichtlich das mit „cd ..“ das working directory eine Ebene nach oben verschoben wurde. Jeweils mit pwd überprüft.

Noch ein kleiner Tip, beim manövrieren im directory tree, wenn man die ersten Buchstaben des directorys, zu welchem man wechseln möchte eingeben hat, kann man mit dem Tabulator das Terminal den Rest ausfüllen lassen. Also ich möchte zum

/Users/pts/Documents

und ich bin im root directory, dann gebe ich ein:

cd Us „Tabulator“ „/“p „Tabulator“ „/“Us „Tabulator“ „/“

Eine Übung:

Jedes File welches man im Finder sieht ist über die Unix Shell erreichbar. Änderungen werden sofort ausgeführt. So wollen wir ein File auf dem Desktop erstellen und löschen.

Wechseln Sie in Ihrem User directory zum Desktop Ordner.

```
[TitanSurfer:~] pts% cd Desktop/
```

Erstellen Sie mit dem Befehl touch eine neue, leere Datei.

```
[TitanSurfer:~/Desktop] pts% touch testfile
```

Eine generische Datei ohne Symbol erscheint nun auf dem Desktop (Abbildung 1.4).

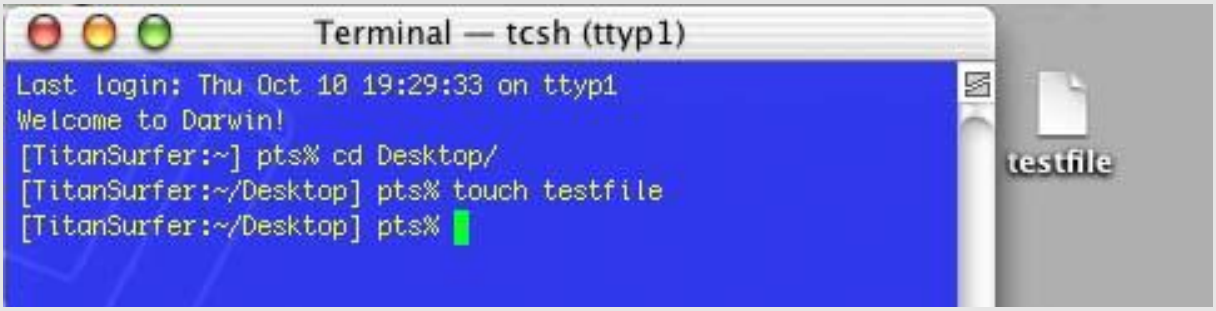

Abbildung 1.4

Löschen Sie diese per Finder, oder in der Shell mit dem Befehl rm.

```
[TitanSurfer:~/Desktop] pts% rm testfile
```

Dateien in einem Ordner anzeigen
Wollen Sie den Befehl cd nutzen, müssen Sie wissen was sich alles in einem directory befindet. Sie müssen aber auch wissen was ein Verzeichnis und welches eine Datei ist. Der Befehl ls wird benutzt, um Einträge in Verzeichnissen (directorys) anzuzeigen.

ls
Wenn man den Befehl ls eingibt erhält man eine Liste der Dateien und Unterverzeichnisse im Arbeitsverzeichnis (working directory). Der Syntax ist:

ls *-Optionen Verzeichnis_oder_Dateinamen*

```
[TitanSurfer:~/Documents] pts% ls
Microsoft User Data  Palm                Visage 1.2.3         pts10%.pdf
```

ls -a
Der Befehl ls kennt eine ganze Reihe von Optionen. Die Option –a (für -all) zeigt garantiert mehr Dateien.

```
[TitanSurfer:~/Documents] pts% ls -a
.                    .DS_Store           Microsoft User Data  Visage 1.2.3
..                   .localized          Palm                 pts10%.pdf
```

In welchem Verzeichnis Sie das jetzt auch testen, zumindest zwei neue Dateien sind aufgetaucht. Sie heissen „." (Punkt) und „.." (zwei Punkte). Zwei Punkte sind immer der relative Pfadname zum übergeordneten Verzeichnis. Ein einzelner bezeichnet immer das Arbeitsverzeichnis.
Alle Dateien die mit einem Punkt beginnen, sind versteckte Dateien. Sie werden nur angezeigt wenn man ls –a eingibt.

ls –al
Für weitergehende Informationen zu allen Dateien fügt man die Option „l" hinzu. Dies steht für „long". Diese kann alleine oder in Kombination mit der Option „a" stehen.

```
[TitanSurfer:~/Documents] pts% ls -al
total 40
drwx------   8 pts  staff   272 Oct  2 01:28 .
drwxr-xr-x  14 pts  staff   476 Oct  4 01:04 ..
-rwx------   1 pts  staff  6148 Oct  2 01:06 .DS_Store
-rw-r--r--   1 pts  staff     0 Sep 17 15:15 .localized
drwxr-xr-x   7 pts  staff   238 Oct 10 14:20 Microsoft User Data
drwxr-xr-x   3 pts  staff   102 Sep 17 21:49 Palm
drwxr-xr-x   9 pts  staff   306 Oct  2 01:12 Visage 1.2.3
-rw-r--r--   1 pts  staff  7137 Oct  2 01:28 pts10%.pdf
```

Das lange Format liefert zu jeder Datei die folgenden Informationen (Abbildung 1.5).

Total n
Der Speicherplatz, der von allen Dateien in diesem Verzeichnis benötigt wird.

Typ
Sagt aus, ob es sich um ein Verzeichnis (d), um eine reguläre Datei (-) oder um einen Link (l) handelt.

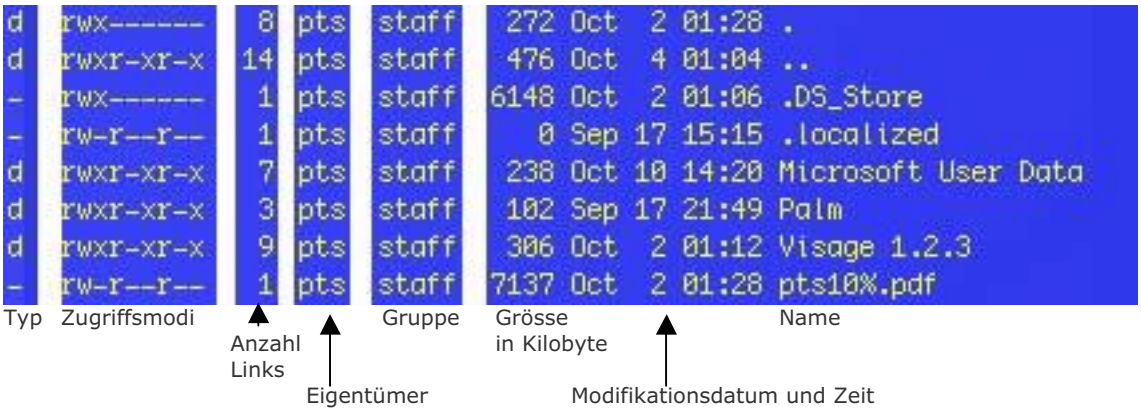

Abbildung 1.5

Zugriffsmodi
Sie geben drei Typen von Benutzern an (Sie selbst, Ihre Gruppe, alle anderen), denen es erlaubt ist, Ihre Dateien zu lesen (r), in Ihre Dateien zu schreiben (r) bzw. sie auszuführen (x).

Links
Die Anzahl der Dateien und Verzeichnisse, die auf die Dateien „gelinkt" sind. Diese Links haben nichts mit Weblinks zu tun. Nur eine kurze Idee um was es geht. Es gibt „hard links" hlink und „symbolic links" slink. Slink sind vergleichbar mit Verknüpfungen im Finder. Und hlink sind zweimal die gleichen Dateien in zwei verschieden Orten. Wenn ich nun die eine verändere, ändert sich die andere auch. Alles klar? Wenn nicht ist es für den Moment auch egal.

Eigentümer
Erzeugte oder besitzt die Datei.

Gruppe
Die Gruppe welche die Datei besitzt.

Grösse
Die Grösse der Datei in Kilobyte.

Modifikationsdatum
Gibt an wann die Datei zum letzten mal geändert wurde.

Name
Name der Datei oder des Verzeichnisses.

Nochmals an einem Beispiel bitte
Schauen wir uns zwei der Ausgaben von ls –al von vorher nochmals an:

```
-rw-r--r--   1 pts  staff     0 Sep 17 15:15 .localized
```

Auf den ersten Blick lässt sich sagen, dass es sich um eine versteckte Datei handelt. Der Typ ist (-) und der Name beginnt mit einem „.". Ich als Besitzer habe sowohl Lese- als auch Schreibberechtigung (rw-). Andere Benutzer des System können die Datei nur lesen (r--). Keiner von allen hat das Recht die Datei auszuführen. Das kann natürlich nur bei Applikationen der Fall sein und da „.localized" eine Datei ist, fällt das weg.

```
drwxr-xr-x   3 pts  staff   102 Sep 17 21:49 Palm
```

Auch hier ist auf den ersten Blick: Ein Verzeichnis mit dem Namen Palm, sichtbar. Aber was ist mit den Zugriffsberechtigungen. Hier steht ein (rwx) für mich, sowie alle anderen. Bei einem Verzeichnis wird mit einem (x) symbolisiert, dass man Zugriffserlaubnis auf das Verzeichnis hat.

ls –F
Die Option (-F) versieht alle Verzeichnisse am Ende mit einem (Slash) so ist es einfacher ein Verzeichnis zu erkennen. Vor allem wenn einem die Details von ls –al nicht interessieren. Die Option „F" kann beliebig kombiniert werden. Der Slash gehört aber nicht zum Namen, sondern ist nur eine Abkürzung welche ls –F verwendet.

```
[TitanSurfer:~/Documents] pts% ls -aF
./                      .DS_Store*              Microsoft User Data/ Visage
1.2.3/
../                  .localized          Palm/                pts10%.pdf
```

Hier die Option –a und –F kombiniert.

Wer kann sich das alles merken
Alle guten Unix Distributionen haben eine Sammlung von Online Handbüchern zu den Befehlen. Diese wird mit dem „man" Befehl formatiert.

man
Der Befehl man steht für „manual" und formatiert sich einfach mit:

man *Befehl den man abfragen möchte*

```
man pwd
PWD(1)              System General Commands Manual                    PWD(1)

DESCRIPTION
    pwd writes the absolute pathname of the current working directory to the
    standard output.

     The pwd utility exits 0 on success, and >0 if an error occurs.

     The following options are available:

     -L      Print the logical path to the current working directory, as
             defined by the shell in the environment variable PWD.
```

Der Handbuch Text wird dann bis zum Terminalfenster angezeigt. Möchte man weiter lesen kann man mit der Taste „Return" bestätigen, oder mit der Taste „q" abbrechen. Die Taste „q" funktioniert seit OS 10.2. Bei älteren Versionen von OS X muss man das Handbuch jeweils mit „ctrl" und „c" abbrechen um zur Shell zurückzukehren. Die Tastenkombination „ctrl" und „c" sind die Unix Unterbrechungstasten und funktionieren im ganzen Shell Bereich.

Übung: Entdecken der Harddisk
Mit diesen ersten Befehlen können Sie bereits erste Reisen durch ihre Harddisk unternehmen. Machen Sie auch probeweise einen Mix aus cd und pwd Befehlen usw.. Hier nochmals eine Tour durch Ihre Harddsik:

Gehen Sie zu Ihrem home directory:	**cd** eingeben
Kontrollieren Sie ihr working directory:	**pwd** eingeben
Wechseln in ein anderes Verzeichnis mit absolutem Pfadname:	**cd /bin** eingeben
Files im neuen working directory anzeigen lassen:	**ls** eingeben
Gehen Sie zum root Verzeichnis und sehen den Inhalt an:	**cd /;ls** eingeben
(Als Befehlsseparator wird das „;" Semikolon verwendet)	
Kontrollieren Sie ihr working directory:	**pwd** eingeben
Wechseln Sie in ein neues subdirecory:	**cd usr** eingeben
Kontrollieren Sie ihr working directory:	**pwd** eingeben
Wechseln Sie in ein neues subdirecory:	**cd lib** eingeben
Kontrollieren Sie ihr working directory:	**pwd** eingeben
Wechseln Sie in eine nicht vorhandenes subdirecory:	**cd xyz**
Lassen Sie sich die Files in einem anderen directory anzeigen:	**ls /bin**
Kontrollieren Sie ihr working directory:	**pwd** eingeben
(Sollte sich nicht verändert haben)	
Gehen Sie zu Ihrem home directory:	**cd** eingeben

Vergleichen Sie auch mal die Ansicht des „root directory" im Finder, mit der Ansicht im Terminal.

Den Inhalt einer Datei betrachten
Im File System herum zu turnen ist ja eigentlich nicht so lustig, wenn ich die Dateien nicht anschauen kann. Unix stellt da einige Möglichkeiten zur Verfügung um in die Dateien zu schauen. BBEdit zum Beispiel ist ein kleiner, feiner Texteditor für den Finder. Wenn ich einfach mal schnell sicher sein will ob ich die richtige Datei lösche, kann ich sie in BBEdit aus dem GUI öffnen oder sie mir schnell im Terminal anschauen. Das Terminal bietet mir einige schnelle Wege um wesentlich schneller als BBEdit zu sein. Wie gesagt unter Unix gibt es einige Programme welche Dateien lesen und in der Konsole dargestellt werden können. Wir betrachten hier nur zwei. Nämlich das Programm: cat und more.

cat
Der Syntax von cat lautet wie folgt:

cat -*Optionen_Dateinamen*

Vergessen Sie nicht, dass Sie jederzeit das Handbuch zu cat via dem Befehl:

man cat

lesen können. Dort sind alle Option aufgeführt und erklärt.

Hier ein Beispiel einer kleinen Datei welche mit cat im Terminal betrachtet wurde.

```
[TitanSurfer:~/Documents] pts% ls -F
Microsoft User Data/ Palm/                  Visage 1.2.3/         pts10%.pdf
testfile
[TitanSurfer:~/Documents] pts% cat testfile
Das ist ein Testfile welches ich mit dem Unix Programm "vi" erstellt habe.
Aber dazu kommen wir im naechsten Teil.
[TitanSurfer:~/Documents] pts%
```

In der ersten Zeile habe ich schnell den Inhalt des Ordners überprüft um danach mit cat das Testfile anzuschauen. Wenn cat mit dem Anzeigen fertig ist, haben wir einfach den Prompt wieder und wir können weiter arbeiten. Haben wir eine sehr grosse Datei wie etwa ein langes logfile, müssen wir via dem Scrollbalken des Terminals zurück scrollen. Das ist nicht sehr effektiv wenn wir einen Eintrag suchen. Für solche Fälle eignet sich more einiges besser. Da es jedesmal stoppt, wenn das Terminalfenster voll ist und wir mit der „Leerschlag" Taste den Befehl geben, das Fenster erneut ganz zu füllen.

more
Der Syntax von more lautet wie der von cat:

cat -*Optionen_Dateinamen*

Ich brauche es nicht mehr zu sagen, auch hier mit dem Befehl man .... Klar!

Hier ein Beispiel von more:

```
[TitanSurfer:~/Library/Logs/CrashReporter] pts% ls
Adobe Illustrator 10.crash.log           Adobe Photoshop 7.0.crash.log
Microsoft Word.crash.log            StuffIt Expander.crash.log
[TitanSurfer:~/Library/Logs/CrashReporter] pts% more Adobe\ Illustrator\
10.crash.log
**********

Date/Time:  2002-10-09 22:10:04 +0200
OS Version: 10.2.1 (Build 6D52)
Host:       TitanSurfer.local.

Command:    Adobe Illustrator 10
PID:        634

Exception:  EXC_BAD_ACCESS (0x0001)
Codes:      KERN_PROTECTION_FAILURE (0x0002) at 0x00000000

Thread 0 Crashed:
 #0   0x002d8038 in 0x2d8038
 #1   0x002d7f34 in 0x2d7f34
 #2   0x002d7d4c in 0x2d7d4c
 #3   0x003ad7d4 in 0x3ad7d4
 #4   0x003ad72c in 0x3ad72c
 #5   0x003ad668 in 0x3ad668
 #6   0x00295b98 in 0x295b98
 #7   0x0029591c in 0x29591c
 #8   0x005d2750 in 0x5d2750
 #9   0x00295354 in 0x295354
 #10  0x0079c2ac in 0x79c2ac
```

```
 #11   0x0060e064 in 0x60e064
 #12   0x0021a37c in 0x21a37c

Thread 1:
 #0    0x9003f188 in semaphore_wait_signal_trap
 #1    0x9003efa4 in _pthread_cond_wait
 #2    0x90233454 in TSWaitOnSemaphoreCommon
 #3    0x9023c298 in TimerThread
 #4    0x90021428 in _pthread_body

Adobe Illustrator 10.crash.log (41%)
```

Auf der ersten Zeile, nach dem Inhalt des Ordners CrashReporter geschaut. Und dann mit dem Befehl more den Illustrator 10 Crash Report angezeigt. Auf der letzten Zeile sehen wir, dass 41% des Textes im File angezeigt wird. Wir könnten jetzt mit der Taste „Space" also dem Leerschlag eine weitere Seite des Terminals füllen, oder uns mit Return, Zeile um Zeile des folgenden Textes anzeigen lassen.

File Zugriff Berechtigungen

Wie wir früher im Text beim Befehl ls gesehen haben, hat jedes File für den Eigentümer, die Gruppe und alle andern ein Zugriffsystem für Lesen, Schreiben und Ausführen. Wenn wir uns die Informationen im Finder über ein File ansehen, sehen wir, dass der Finder das auch so sieht (Abbildung 1.6). Im Gegensatz zu OS 9 eine leicht veränderte Methode.

Wenn wir die Zugriffsattribute ändern wollen können wir das im Terminal oder im Finder machen. Das Ändern im Finder hat den Vorteil das ich keinen neuen Befehl lernen muss. Der Nachteil im Finder ist, wenn ich für den Finder unsichtbare Files ändern muss, um zum Beispiel den Telnet Zugriff auf einen andern Port umleiten möchte.

Allerings ist es unter Mac OS X möglich, im Gegensatz zu früheren Mac OS Versionen, die Attribute mehrerer Dateien miteinander zu verändern (Abbildung 1.7).

Abbildung 1.6

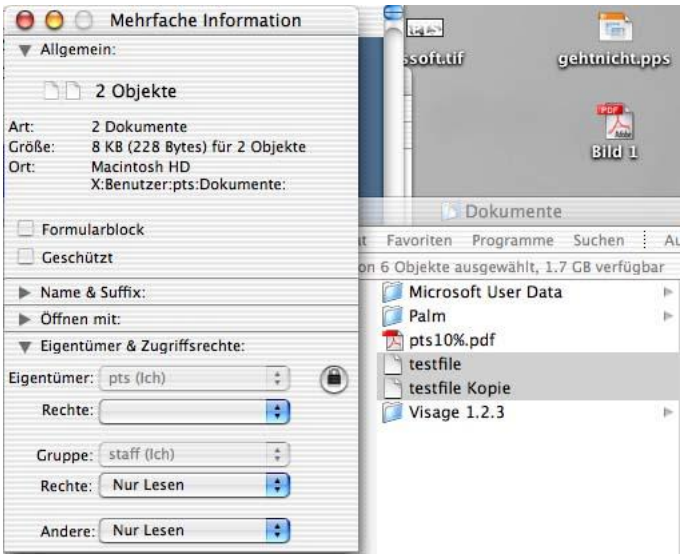

Abbildung 1.7

chmod
Es gibt streng genommen zwei Wege wie man die Berechtigungen ändern kann. Entweder man fügt einem Benutzer oder einer Gruppe Berechtigungen hinzu, oder man nimmt sie ihnen weg. Dies ist genau der Gedanke welche chmmod macht. Deswegen lautet der Syntax wie folgt:

chmod *(wer: ich, Gruppe, andere) (was kommt weg oder hinzu) <Filename>*

Das scheint doch ein bisschen komplexer. Ist es aber nicht. Hier die Spielregeln und ein Beispiel.

1. Wie schon früher besprochen (beim Befehl ls) gibt es drei Gruppen. Der Besitzer welcher hier user heisst und mit „u" abgekürzt wird Die Gruppenberechtigung welche „g" abgekürzt wird. Die andern werden schlicht others „o" genannt. Wenn man mehr als eine Kategorie ändern möchte schreibt man einfach die Lettern zusammen. Beispielsweise: „go" für Gruppe und die Anderen.
2. Möchte ich Rechte hinzufügen, dann schreibe ich „+", möchte ich sie wegnehmen dann schreibe ich „-„.
3. Die Berechtigung welche ich bearbeiten möchte wird mit der gleichen Schreibweise wie beim ls Befehl geschrieben. „r" read, „w" write, „x" execute.

Hier jetzt endlich ein Beispiel wie es dann aussieht:

```
[TitanSurfer:~/Documents] pts% ls -l testfile
-rw-r--r--  1 pts  staff  114 Oct 12 17:45 testfile
[TitanSurfer:~/Documents] pts% chmod go+w testfile
[TitanSurfer:~/Documents] pts% ls -l testfile
-rw-rw-rw-  1 pts  staff  114 Oct 12 17:45 testfile
[TitanSurfer:~/Documents] pts%
```

1 Zeile: Die Berechtigungen des Testfiles angeschaut. Die group und die others können nicht in das File schreiben. Sie können es nur lesen (2 Zeile)
3 Zeile: Mit chmod der group und den others (go) die Berechtigung zum schreiben (+w) hinzugefügt.
4 Zeile: Den Vorgang überprüft.

Christoph Müller
Rüschlikon, 18.10.2002

Bei Fragen oder Anmerkungen: Kontaktieren Sie mich bitte unter chm@pts.ch.