TECHNOIDS

The free BeOS magazine

Ausgabe 2 - August 2003 www.technoids.tk

ProEdition | PeEdition | DevEdition | MaxEdition | Zeta | OpenBeOS ...

in dieser Ausgabe findet ihr ....

- ein Interview mit Axel Dörfler ...

- Berichte über Programme ...

- Workshops ...

- Tips & Tricks ...

Liebe BeOS - Freunde

Wir hoffen ihr habt die Zeit bis zur neuen Auflage von Technoids gut überstanden.

Wir waren positiv überrascht, als man auf Bebits lesen konnte, dass das erste Magazin über 400 mal heruntergeladen wurde. Ebenfalls gefreut hat uns, dass sich einige BeOS Mitglieder bei uns persönlich gemeldet haben, sei es wegen Hilfe bei Übersetzungen, als Testleser oder als Artikelschreiber.

Vielen Dank dafür!

Des weiteren möchten wir euch für die Feedbacks danken, die ihr zu der ersten Ausgabe gepostet habt. Ihr habt uns mit eueren Vorschlägen und Kritiken sehr geholfen, sodass wir guten Grundes hoffen können, dass diese Ausgabe um ein bedeutendes Stück professioneller wirkt.

Besonderer Dank geht an die fleißigen Übersetzer, die die englische Version in die jeweilige Muttersprache übersetzt haben, sprich in Polnisch, Französisch und Griechisch.

Ebenso geht ein großer Dank an den Portugiesen Joao Carvalho, der uns einen netten Vorschlag für das Technoids Layout gemacht hat.
Wir haben davon abgeschaut :-)

Auch den Personen einen Dank, die eigentlich besseres zu tun hätten oder anderweitig ziemlich beschäftigt sind und sich trotzdem für uns Zeit genommen haben.

So, nun wünschen wir euch gute Unterhaltung mit der zweiten Ausgabe von Technoids und sind gespannt auf eure Reaktion!
Und vergesst nicht weiterhin Feedbacks zu senden bzw. zu posten, denn ohne Feedback wissen wir nicht was wir besser machen könnten bzw sollten.

Euer Technoids Team Matthias,
Lelldorin und Florian mit Gehilfen

Interview

Berichte/Vorstellungen

Gedanken

Workshop

Tips & Tricks

Impressum

Interview mit Axel Dörfler von OpenBeOS

von Matthias Breiter

Das Interview wurde von Matthias Breiter per email geführt. Trotz (oder gerade wegen) des großen Umfanges gibt es einen sehr interessanten Einblick in das freie BeOS - System. Wir danken Axel Dörfler, einem der wichtigen Programmierer und erstem Ansprechpartner des viel versprechenden Projektes.

1. Kannst Du Dich unsern Lesern kurz vorstellen?

Ich heiße Axel Dörfler, bin Jahrgang 1976 und studiere z.Zt. (noch) "Computerlinguistik & Künstliche Intelligenz" an der Universität Osnabrück.

2. Wie und wann bist Du zu BeOS gekommen?

Vermutlich hauptsächlich durch meine Abneigung gegenüber dem Betriebssystem Windows - ich war lange Zeit (und bin es immer noch) ein Amiga-Benutzer, und konnte als solcher die Windows-Versionen vor NT nicht so ganz ernst nehmen.
Bis zum Jahre 2000 besaß ich auch keinen eigenen "IBM-kompatiblen" PC; ich hatte mir mal eine Demo von BeOS R3 auf dem Computer meines Vaters angesehen, als es damals mal auf der CD irgendeines PC-Magazins enthalten war. Leider war es nur in schwarz/weiß mit 640x480 und sogar die mitgelieferten Programme stürzten z.T. bei meinem oberflächlichen Test ab, daher verfolgte ich seinen Werdegang auch erstmal nicht weiter, auch wenn ich mich davor mal als Entwickler bei Be registrieren ließ - aber das war wohl noch vor der BeBox, und ich wurde irgendwann wegen Nichtaktivität nicht weiter von den Be Newslettern behelligt. Als dann aber, ca. 3 Monate nach dem Erwerb eines PCs BeOS R5 PE herauskam, und ich es dann auch einfach mal direkt ausprobierte, war es auch schon zu spät - bis zu diesem Zeitpunkt fristete der PC ein Schattendasein, der sich mit Windows 98 langsamer anfühlte als mein alter Amiga (immerhin ein Athlon 600). Aber es war nicht nur die überragende Geschwindigkeit mit der BeOS das änderte; es passte einfach alles (und sogar meine Hardware wurde zufälligerweise komplett unterstützt).

3. Was hat Dich dazu bewegt, OpenBeOS zu initiieren?

Nun, initiiert habe ich es nicht - das war "damals" Michael Phipps mit einer kleinen Schar von Mitstreitern. Ich habe noch ein wenig abgewartet, und hatte dem kleinen Projekt eigentlich keine großen Chancen eingeräumt, in den nächsten paar Jahren eine Alternative zu erschaffen.
Geändert hat sich meine abwartende Position eigentlich erst, als Be offiziell verkauft wurde, und es klar war, das BeOS selbst nicht mehr weiterentwickelt werden würde (YellowTabs Zeta hat ja auch noch ein wenig gebraucht, bis es soweit kam wie es heute ist) - und da alle anderen alternativen Betriebssysteme (von denen ich zu dieser Zeit einige ausprobierte) nicht meinen hohen Standards entsprachen, war OpenBeOS der für mich folgerichtige Schritt, die Zukunft von BeOS selbst in die

Hand zu nehmen. Ich schloss mich dann mit Bruno G. Albuquerque (auch bekannt als BGA) zusammen dem Team an, und habe es immer noch nicht bereut - arbeite ich doch mit den fähigsten und nettesten Programmierern zusammen, die ich bislang kennen lernen durfte.

4. Bist Du mit den Programmiermöglichkeiten (API, Schnittstellen) aus heutiger Sicht noch mit BeOS zufrieden?

Das bin ich zum größten Teil - mal abgesehen von den vielen Fehlern und Unzulänglichkeiten in deren Implementation (wie z.B. im Media Kit). Was mir als Programmierer fehlt, ist eine vernünftige Art, GUIs zu erstellen (siehe Marco Nelissens hervorragende liblayout.so), ein Locale Kit (in Arbeit), und ein vernünftiger Netzwerkstack.
Ich bin da also relativ anspruchslos - darüber hinaus könnte man aber natürlich noch viele Kleinigkeiten verbessern, und einige Design-Pattern aus aktuellen Klassenbibliotheken integrieren; aber der diesbezüglich aktuelle Status schränkt mich nicht ein.

Da setzt ja auch OpenBeOS erstmal an: im ersten Schritt wird versucht, die Implementation zu verbessern, und danach kommen dann die "Schmankerl" in der API - was interessiert es mich denn, wie kompliziert (oder einfach) das Media Kit zu benutzen sein mag, wenn das blöde Ding meine Filme nicht synchron abspielen möchte?

5. Falls Du andere Betriebssysteme auch kennst, würdest Du sagen, Be hat das Ziel der leichten Programmierbarkeit erreicht?

Definitiv, es kann von der Programmierbarkeit her durchaus mit anderen aktuellen Systemen mithalten (Java, OpenStep/MacOS X). Windows kann man zumindest in der pre-.NET Ära in diesem Zusammenhang eigentlich nur als abschreckendes Beispiel nennen.

6. Zurück zu deinem OpenBeOS. Wollt Ihr die APIs usw... überarbeiten oder hauptsächlich BeOS als freie Variante "Nachprogrammieren"?

Der Plan ist, BeOS R5 mit allen seinen APIs neu zu implementieren, d.h. die APIs werden grundsätzlich erstmal nicht verändert. Das wird zum größten Teil erst mit den nachfolgenden Revisionen passieren, auch wenn wir unter der Haube einige Dinge anders erledigen, von denen zwar Anwender wie Entwickler profitieren werden, aber keine (großen) API Änderungen erfordern.
Das hat dann auch den Vorteil, dass wir mit anderen BeOS Derivaten zusammen an Änderungen der API arbeiten können, und die Folgen von Änderungen direkt absehen können.

7. Im Gegensatz zu anderen freien Be Derivaten, setzt Du ja nicht auf ein Linux System sondern auf einen "echten" BeOS Kern. Warum?

Nun, das hat mehrere Gründe. Einer von ihnen ist z.B. schon die GPL - obwohl ich natürlich keine grundsätzliche Abneigung dagegen habe, halte ich sie im Bereich Betriebssysteme für nur bedingt geeignet. Wäre z.B. neben dem Kernel auch noch XFree unter der GPL würden heute viele Linux Systeme auf VESA angewiesen sein - auch wenn das nicht unbedingt offensichtlich ist, halte ich die Treibersituation unter Linux für alles andere als optimal; viele Firmen scheuen sich, öffentliche Treiber für Linux zu entwickeln. Außerdem schafft die GPL meiner Meinung nach die von uns nicht ganz gewollte Atmosphäre, das alles kostenlos sein müsse; sie setzt für einen kleinen Markt wie den unsrigen möglicherweise die falschen Signale.
Aber auch technische Gründe würden mich eher im Bereich FreeBSD/ Darwin/ etc. als in Linux nach einem adäquaten Kernel suchen lassen.

Aber warum haben wir uns nun entschlossen, einen eigenen Kernel zu entwickeln, bzw. NewOS als Basis zu benutzen? Hätten wir uns damals für FreeBSD entschieden, wäre unser Kernel vielleicht schon fertig, möglicherweise mit Eigenschaften, für deren erreichen wir nun mit Jahren rechnen können. Nun ganz so ist es auch nicht: die Anpassung eines fertigen Kernels um alle von uns gewünschten Leistungen zu erreichen wären z.T. erheblich gewesen - eine Neuimplementierung dieser Teile ist in vielen Fällen schneller und vor allen Dingen sauberer möglich. Wir haben die Möglichkeit, ein System komplett neu zu entwerfen, und nach unseren Bedürfnissen zu gestalten - und alles kommt, perfekt aufeinander abgestimmt, aus einem Guss. Wer schonmal in Linux oder BSD Sourcecode geschaut hat, der wird problemlos feststellen können, dass unser Code deutlich einfacher zu lesen und zu verstehen ist - und allein diese Tatsache ist mir sehr wichtig. Alles ist einheitlich und nach modernen Kriterien entwickelt - ein System, das so einfach und transparent ist, daß es sich niemandem verschließt.
Das z.B. erhöht meine Motivation erheblich, an diesem Projekt zu arbeiten - und Motivation ist eigentlich alles, was ein OpenSource Projekt benötigt - und Zeit, natürlich. Natürlich wird z.B. B.E.OS wegen dieser Unterschiede schneller Resultate vorweisen können als wir - aber wer schonmal mit Linux gearbeitet hat, wird schnell verstehen, daß Transparenz und Einfachheit sicher keine Entwicklungskriterien gewesen sein können, ganz unabhängig von dem, was Linux für den Power-User zu leisten imstande ist.

8. Wird OpenBeOS 100% Be- Kompatibel sein oder sind Rekompilierungen nötig?

OpenBeOS wird 99,8% kompatibel sein - d.h. das alle Applikationen, die auf *öffentlichen* APIs basieren, direkt funktionieren werden.
Einige private und undokumentierte APIs werden wir aber nicht oder anders umsetzen, weswegen z.B. das originale DriveSetup nicht auf OpenBeOS laufen wird - aber natürlich werden wir für solche Applikationen Ersatz liefern.
Wir drehen aber z.B. auch den Adreßraum um, was die Umsetzung von Projekten wie Wine deutlich vereinfachen wird, aber eben einige Änderungen mit sich bringt - glücklicherweise macht es wenig Sinn, solche Dinge im Programm abzufragen.
D.h. man kann grundsätzlich sagen: entweder es funktioniert (99,8%), oder es funktioniert nicht (0,2%) - daran kann eine Neukompilierung aber nichts ändern.
Ausnahme ist hier die PPC Version, die nicht binärkompatibel zu BeOS R5 PPC sein wird - hier ist eine Neukompilierung Voraussetzung.

9. In wie weit würde echte Kompatibilität aus technischer Sicht denn Sinn machen, insbesondere auf das reine Software- OpenGL bezogen?

Das ist ja wieder eine Implementierungsfrage. Die aktuelle MESA Version kann z.B. als drop-in Ersatz für Be's Software OpenGL benutzt werden (auch wenn es da noch ein paar Probleme gibt) - aber niemand kann in diesem Fall verlangen, daß wir künstlich die Funktionalität auf Software-Rendering beschränken würden (wenn wir denn derzeit eine andere Wahl hätten); solche Dinge sind aber nicht unbedingt für eine Erhaltung der Kompatibilität erforderlich.
Oder um ein anderes Beispiel zu geben: nur weil BeOS nicht vernünftig mit erweiterten und logischen Partitionen arbeiten kann, heißt das nicht, daß wir das Verhalten reproduzieren müßten, da die Implementation in diesem wie in den meisten anderen Fällen unabhängig von der Kompatibilität ist.

10. Was hältst Du von Bernd Kortz' ZETA Betriebssystem, das ja nicht open source ist?

Das einzige, was ich momentan daran auszusetzen habe, ist, daß es noch nicht fertig ist, und noch nicht auf dem Markt verfügbar. Ich hoffe, daß es den BeOS Markt wieder ein bißchen belebt, von dem wir schließlich auch ein Teil sind. Ich wünsche Bernd jedenfalls viel Glück!

11. Glaubst Du, das es auf Dauer Sinn macht, 3 freie Be- Varianten + eine Kommerzielle anzubieten bzw. zu entwickeln?

Definitiv nicht - aber ganz so schlimm ist es ja auch nicht. Wenn sich nicht alle Entwickler auf einen Kurs einigen können, ist es doch natürlich das es verschiedene Wege gibt - keines unserer Projekte (mal abgesehen von Zeta) ist auf finanziellen Erfolg aus, wir haben keinen Businessplan zu erfüllen und auch keinen Zeitdruck etc. - warum sollte sich also jemand freiwillig einem Projekt zuwenden hinter dem er nicht vollständig steht (er wird dafür schließlich nicht bezahlt)? Für das Wohl aller passiert das jedenfalls offensichtlich leider nicht.
Aber von den unterschiedlichen Zielen der Projekte abgesehen, werden sie z.B. alle von Marcus Overhagens Media Kit Implementierung profitieren - und wir lassen sie davon profitieren; unsere Lizenz-bedingungen erlauben ihnen diese Nutzung ausdrücklich. D.h. auch wenn unsere unterschiedlichen Ziele zu verschiedenen Projekten geführt haben, und einige Teile daher doppelt geleistet werden müssen, arbeiten wir doch automatisch auch zusammen. Es ist zwar bislang noch die Ausnahme, da wir alle noch relativ am Anfang sind, aber die Zusammenarbeit wird definitiv zunehmen. So hat z.B. Bill Hayden, der Entwickler von COSMOE auch Schreibrechte in unserem Repository.
Lange Rede kurzer Sinn: unsere potentiellen Benutzer bringen diese Möglichkeiten natürlich nicht unbedingt weiter, aber ich rechne schon damit, dass wir uns irgendwann auf einem gemeinsamen Weg wiederfinden werden.

12. Zeigt uns die Geschichte nicht, dass Betriebssysteme sich über die "eigene" Hardware am erfolgreichsten durchsetzen? (AMIGA, Commodore, Apple, Microsoft + IBM-PC XT/AT)

Nein, eigentlich zeigt sie genau das Gegenteil - ein gewisses Betriebssystem verteilte sich damals am besten, weil es keine eigene Hardware mitbrachte, sondern überall lief. Und auch Linux tritt in diese Fußstapfen. Ob wir Erfolg haben werden, oder nicht, wird sicherlich keine Hardware-Frage werden (zumindest nicht in diesem Sinne).

13. Würdest Du sagen, das ein Computer wie die BeBox mit höherer Auflage mehr Sinn gemacht hätte? Oder ist BeOS Heute noch aktuell, weil es auf normalen PCs läuft?

Letzteres; würde die BeOS Community nur aus denjenigen bestehen, die eine BeBox ihr Eigen nennen können, gäbe es heute wahrscheinlich keine mehr.

14. Kannst Du uns etwas zum Fortschritt des OpenBeOS Systems sagen? Gibt es etwas, was schon gut funktioniert?

Meiner Meinung nach schreiten wir gut voran. Eigentlich alle Bereiche haben schon etwas funktionierendes vorzuweisen - und wenn es nur unsere Unit-Tests sind, die übrigens zeigen, daß unsere Implementierung deutlich fehlerfreier und daher auch stabiler ist als das Original. Auch so komplexe Dinge wie das Media Kit oder BFS sind bereits zum Ausprobieren (auf eigene Gefahr) vorhanden; Marcus hat die Alpha 1 des Media Kits erst vor kurzem veröffentlicht, und allein die

Fortschritte seit diesem Zeitpunkt sprechen eine deutliche Sprache.

15. Du arbeitest ja an einem PowerPC- Kern. Wird es wieder eigene Software für PPC und x86 geben, oder läuft die OpenBeOS-Software auf beiden Plattformen?

Die Software muss für den Gebrauch auf einer anderen Plattform neu kompiliert werden - andere Anforderungen wird es keine geben. Es wird allerdings derselbe Compiler benutzt, was diesen Schritt im Vergleich zum originalen BeOS erheblich vereinfachen wird. Der PowerPC Kernel ist eine Portierung unseres Kernels auf den Pegasos, eine neue und aktuelle PPC Hardware die gerade von der Firma Genesi entwickelt wird. Sie wird allerdings voraussichtlich auch die Erste sein, die komplett selbständig von Festplatte booten wird; je nachdem, wie viele Fehler die OpenFirmware Version in meinem Rechner besitzt.

16. Wird dein PowerPC- Port auch moderne CPUs unterstützen? Beim echten BeOS war ja mit dem 604ten Schicht im Schacht.

Schlimmer noch, mein Port wird erstmal ausschließlich moderne CPUs unterstützen - ältere Modelle werden aber evtl. später nachkommen. Ich werde OpenBeOS nur auf der Hardware lauffähig machen können, die ich auch mein Eigen nennen darf.

17. Wie bist Du mit dem Support aus der Community zu frieden? Bekommst Du Feedback? Wird regelmäßig an OpenBeOS gearbeitet?

Das sind jetzt aber zwei verschiedene Dinge, oder? Natürlich wird regelmäßig an OpenBeOS gearbeitet - Dank der weltweiten Verbreitung der Entwickler kann man sogar ohne (viel) Übertreibung sagen, daß rund um die Uhr daran entwickelt wird. Ich fühle mich als Teil der (Open)BeOS Community sehr wohl.

18. Glaubst Du, das BeUnited es schaffen kann, alle Be- Richtungen unter einem Hut zu halten?

Das werden wir sehen, auch ob es überhaupt notwendig sein wird. Ich hoffe jedenfalls, daß wir insgesamt zu einer solchen Lösung kommen.

19. Kurze Frage, arbeitest Du an OpenBeOS fulltime oder in deiner Freizeit?

Haha, wenn mich jemand bezahlen würde, würde ich es gerne Vollzeit machen. Aber ich studiere noch, und wende nur einen Teil meiner knappen Freizeit für dieses Projekt auf.

20. Was wünscht Du Dir von BeOS für die Zukunft? Wird es noch mal zurück kommen?

Ich wünsche mir zumindest, daß es eine sinnvolle Alternative zu der am Markt befindlichen Produkte wird. Mir persönlich ist es nicht unbedingt wichtig, wie viele dieses für sich erkennen - auch ein kleiner Markt hat seine Vorteile. Aber wichtig wäre schon, daß ein Markt entsteht, d.h. das BeOS so groß wird, daß sich Firmen, die sich der BeOS-Programmierung widmen, auch am Leben halten können. Für meine persönliche Anteilnahme an diesem System ist das zwar nicht ausschlaggebend, aber ich würde es sehr begrüßen, damit irgendwann einmal meinen Lebensunterhalt finanzieren zu können - ich habe zumindest noch sehr viele Pläne, die ich mal umgesetzt wissen möchte.

21. Ich Danke Dir für dieses Interview.

Keine Ursache.

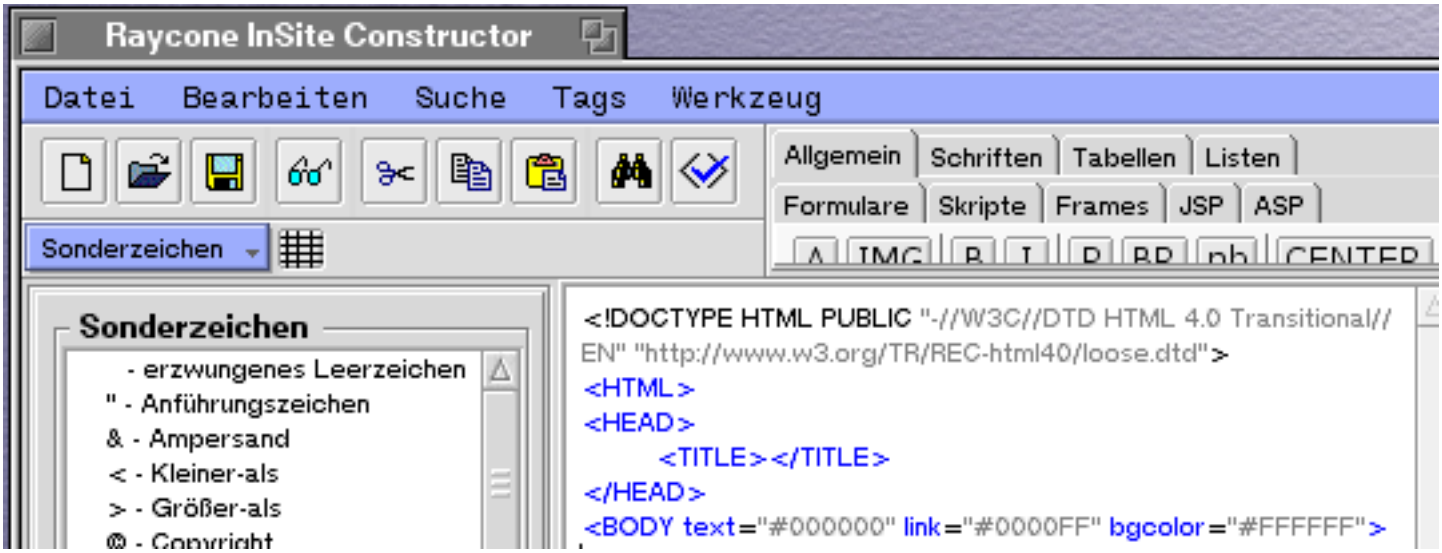
Mehr über das Thema OpenBeOS siehe unter: http://www.openbeos.org

Raycone mit "Insite" - Programmen wieder dabei

von Florian Thaler

Nach einem Stillstand der Weiterentwicklung von den bekannten Web-Design Programmen InsiteConstructor und InsiteDesigner seit Anfang 2002 wurde diese nun wieder aufgenommen. Der Grund der Wiederaufnahme ist der, dass der Verkauf von raycone-Software-Lizenzen vor acht Monaten zwar auf 0 gesunken ist, neulich aber wieder einige Lizenzen verkauft wurden. Deshalb arbeiten die Jungs weiter. Wie lange die Entwicklung weitergeht, so Ales von raycone, hängt davon ab, wie lange bzw. wie viele Lizenzen verkauft werden. Auf meine Frage, ob die Entwickler vorhätten, ein komplett neues Produkt für BeOS zu entwickeln, wurde geantwortet, dass man viele Lizenzen verkaufen müsste um ein neues Projekt zu starten. Wie ich weiter in der Mail von Ales Nedomlel erfahren habe, wird derzeit der InSiteConstructor um Tools erweitert und im Juli/September soll eine neue Version auf den Markt kommen. Eine neue Version von InsiteDesigner können wir im Herbst erwarten. Der neue InsiteConstructor soll alle Features enthalten, die ein Web-Tool bereitstellen sollte. Es werden ziemlich viele Änderungen vorgenommen, jedoch eher kleinere Verbesserungen, Dinge, die das Leben des Webdesigners noch einfacher machen sollen. Zeilennummern können ausgeschalten werden, das linke Panel kann ausgeblendet werden usw., auf alle Fälle soll das verbessert werden, was in der letzten Ausgabe gefehlt hat und InsiteConstructor-Käufer an Verbesserungsvorschlägen eingereicht haben. Was das Angebot an verschiedenen Sprachen der Programme angeht, dürfte es keine Probleme geben. Derzeit reicht das Angebot von Tschechisch, Englisch, Polnisch bis zu Russisch, Deutsch, Französisch und Italienisch (Version 1.1.4), außerdem ist es sehr einfach, andere Sprachen in das Programm zu integrieren (localedit).
Billiger werden die Programme wohl nicht werden, für BeOS- Verhältnisse sind 49$ zwar hoch, aber sie sind es wert. Auf anderen Plattformen werden vergleichbare Programme viel teurer verkauft.
Eine kostenlose, eingeschränkte Demo-Version kann unter http://www.raycone.com heruntergeladen werden. Es lohnt sich auf jeden Fall, das Programm auszuprobieren!

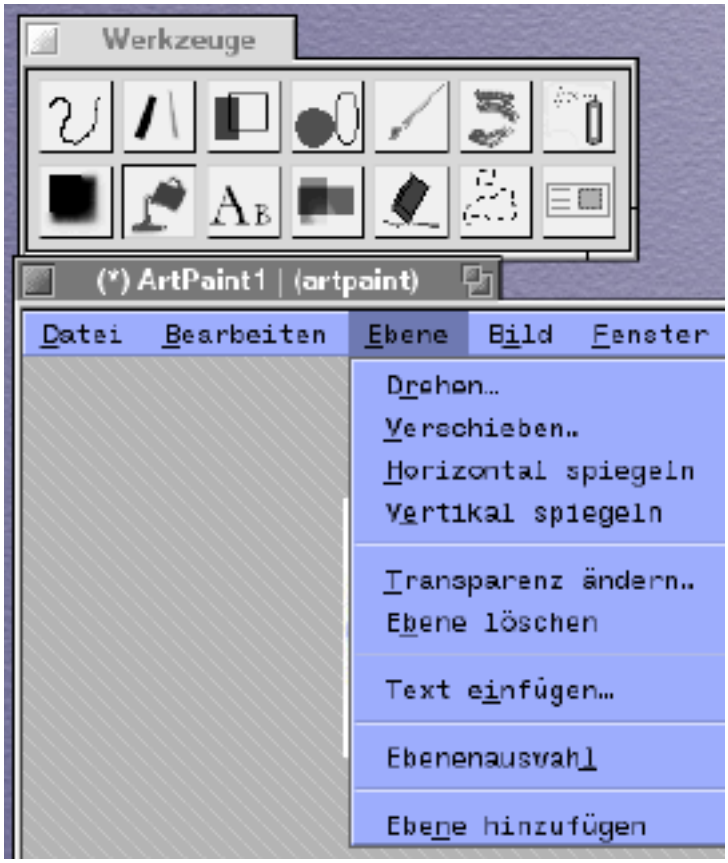
Weitere Informationen sowie Screenshots findet man auf der Homepage von raycone:
http://www.raycone.com



BeOS und vor mir laufen 1000 Bilder ab

Holger Wendenburg

Auch auf dem Gebiet der Bildbetrachtung und Bildbearbeitung hat BeOS einiges zu bieten. Leider fehlt es etwas an den notwendigen Treibern um diesen wirklich hervorragenden Programmen den nötigen Input zu liefern. Es wäre schon schön, wenn unter BeOS mehr externe Peripherie Geräte wie Scanner, Digitalkameras und Webcams unterstützt würden. Die Programme die es zur Weiterbearbeitung bereits gibt, haben einiges zu bieten. Auch hier werde ich die Abstimmungsergebnisse meiner BeOS Umfrage (siehe http://beos.holgerwendenburg.de) einfließen lassen.



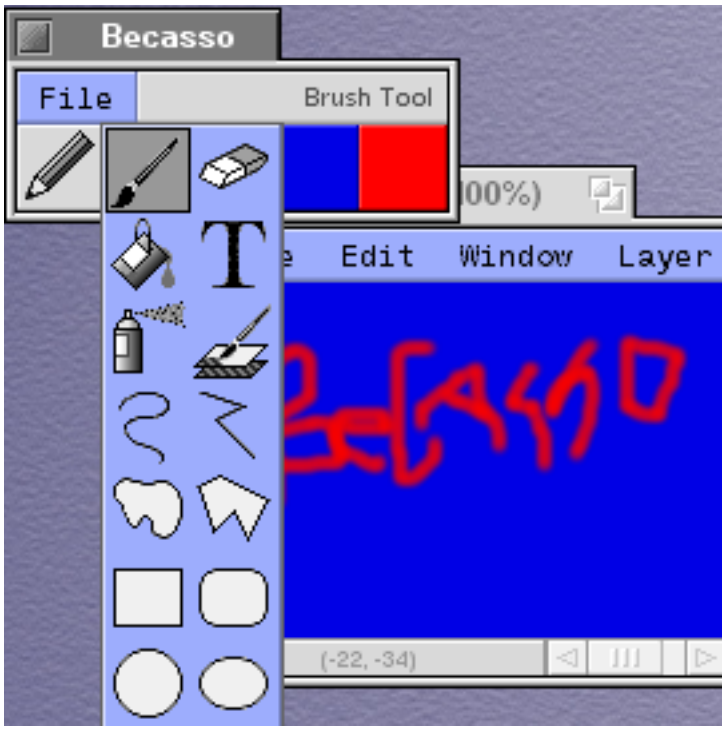
"Soviel Kunst muß sein"

Doch zunächst zu den Bildbetrachtern. Schon mal mit dem Mausrad durch ein Photoalbum mit weit über 100 Bildern gescrollt ohne nachzuladen? Wenn nicht sollten Sie mal einen Blick auf das Programm Butterfly wagen. BeOS spielt auch hier seine Geschwindigkeit voll aus.

http://www.chez.com/bthery/beos.html

Auch Peek ist einen Blick wert und erfüllt seine Aufgaben tadellos. Ebenso stellvertretend für weitere Programme dieses Genres seien ImageViewer und ImageMounter erwähnt.

Nun zu den Bildbearbeitungs- und Zeichenprogrammen. Wegen der Fülle der Programme hier in alphabetischer Reihenfolge.
Beginnen wollen wir mit Artpaint (Platz 25) das nun in der zweiten Version vorliegt und damit an Funktionsumfang gegenüber seinem Vorgänger entscheidend zulegen konnte.

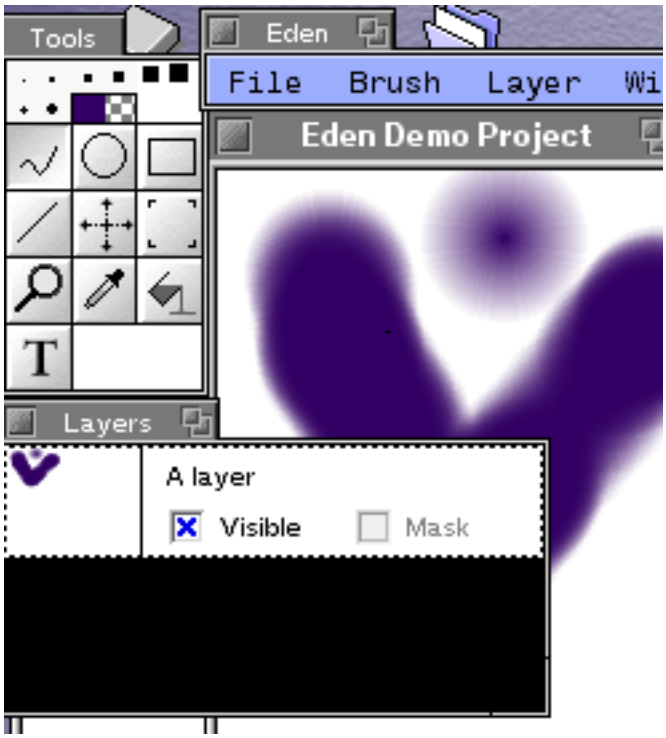


"Picasso hätte Freude dran"

Hier Becasso, das sich mit Artpaint den 25.ten Platz teilt und seinem Namen alle Ehre macht.

BePhoto Magic: auch hier steht der Bedienkomfort im Vordergrund:

"It is magic"

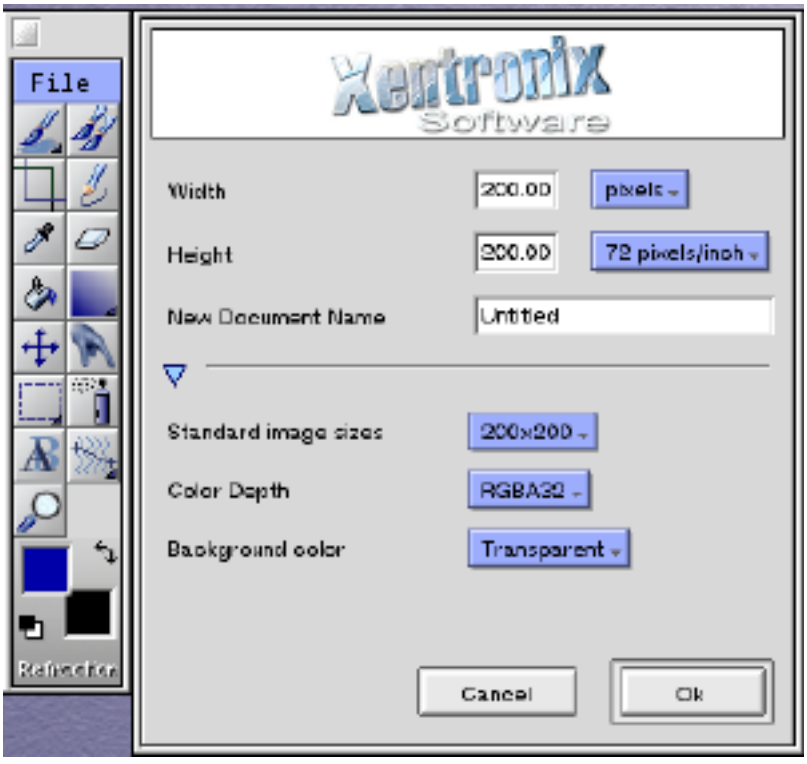


"Eden"

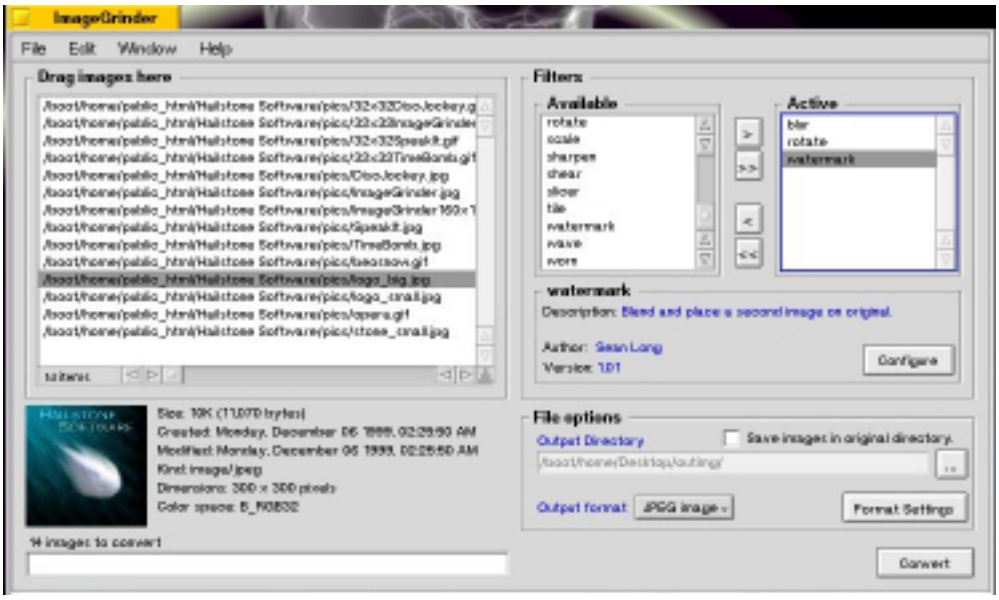
Schon etwas älter aber nicht minder gut - Easel.
http://www.bebits.com/app/2170

Für viele Bildbearbeitungsprofis der Himmel auf E(r)den.

Abschließend kommen wir zu einem Highlight unter den Bildbearbeitungs-programmen was auch die gute Bewertung bei der Umfrage (Platz 17) belegt. Refraction (das frühere Inferno) steht in der Gunst der Benutzer zu recht ganz oben. (hier rechts im Bild)



BeOS hat auch recht lustige Seiten, wie das Programm TuxPaint beweist, das von Linux portiert wurde und sich zum Zeichnen recht gut eignet. Begleitet werden Sie dabei von Sounds und anderen Extras. Hiermit macht es bestimmt auch jüngeren BeOS Benutzern und Benutzerinnen Spaß eigene Zeichnungen anzufertigen.



"Kleines Quadrat ganz groß"

Abschließend kommen wir noch zu Sonderprogrammen wie z.B. Tools mit denen Sie einfach Ihre Bildern in andere Formate umwandeln können. ImageGrinder(Bild oben) zeigt eine kleine Vorschau und ist einfach zu bedienen.

Für wahre Massen dieser Aufgabe ist das folgende Programm gut geeignet daher auch sein treffender Name: MassivePictureConverter. Es steht auch PicConvert als TrackerAdd-On zur Verfügung.

Dies war natürlich nur ein kleiner Ausflug in die bunte Welt von BeOS. BeOS hat auf diesem Gebiet ganz viel zu bieten, man denke da noch an Programme wie Blender (Platz 17) oder bmodeler mit denen sich 3dimensionale Bilder erschaffen lassen. Um Blender zum Laufen zu bekommen hier ebenfalls noch ein Tipp. Blender verlangt beim Start immer nach eine libpython1.5.so die sich in der home/config/lib befinden muss. Am Besten Python 2.2.2 installieren, da ist die gewünschte .so nämlich mit dabei (allerdings handelt es sich um eine neuere Version daher ist diese nicht mit der von Blender geforderten Bezeichnung ausgewiesen). Diese .so muss folglich noch dupliziert und umbenannt werden. Hierzu den ganzen Namen der Kopie löschen und in den von Blender gewünschten umbenennen.

Bei Fragen stehe ich natürlich gern zur Verfügung

post@holgerwendenburg.de

Der PackageBuilder v4.5

von Christian Albrecht

Mit Hilfe dieses Programmes ist es möglich Installationspakete zu erstellen. Das bedeutet, daß es nicht nur möglich ist, zusammengehörende Teile in einem Paket zu verbinden, sondern es ist auch möglich, die dazugehörenden Teile an den jeweiligen Orten zu installieren, wo diese benötigt werden. Dabei sind die Library (Bibliotheken, Treiber) im besonderen zu erwähnen, ohne deren Anwesenheit kein Programm funktioniert.

Was bei normalen Komprimierungsdateien wie z.B. BeZip oder BeRar noch ein von Hand durchgeführter Arbeitsgang mit Nachlesen einer Informationsdatei ist, wird hier automatisch durchgeführt.
Zusätzlich verringert der PackageBuilder die Dateigröße, indem dieser die Paketinhalte komprimiert. Der PackageBuilder ist in jeder ProEdition enthalten, muss aber nachträglich installiert werden.

Die Funktionen im Überblick:

Package
- New Package
Öffnet ein neues Projektfenster, ohne ein eventuell geladenes Projekt zu zerstören.

-Open...
Über diesen Menüpunkt kann ein bestehendes Projekt oder fertiges Package in den Builder geladen werden.

- Edit Groups
In diesem Bereich legen sie bestimmte Gruppen an, die später bei der Installation für die Zuweisung der einzelnen Paketinhalte nötig sind. Ohne diese Zuweisung weiß der Installer nicht, welche Paketteile mit installiert werden sollen.

- EditDestination
Möchten sie bestimmte Dateien in Systemordner installieren, die nicht unter den vorgefertigten Speicherpunkten (Destination DropDown Menü) angelegt sind, können sie Eigene anlegen, indem sie den genauen Pfad angeben.

-Save Package
Hier können sie ihr Projekt als Projektfile abspeichern. Diese Datei ist nicht als Installer ausführbar, sondern lädt automatisch den Builder. Diese Funktion speichert das gerade bearbeitete Projekt unter den Projektnamen ab.

-Save Package As
Über diese Funktion können sie das Projekt unter einen anderen Namen abspeichern. Dies ist besonders zum testen geeignet, da auf diese Weise eine schnelle Rückstellung der vorherigen Einstellungen über das Laden der Originaldatei möglich ist.

-Close
Schließt den PackageBuilder. Haben sie gerade ein Projekt in Arbeit, das noch nicht gespeichert wurde, fragt der Builder nach, ob die Datei vorher gespeichert werden soll.

-Preferences
Hier können am PackageBuilder einige Einstellungen vorgenommen werden.

-About PB...
Infos über den PackageBuilder

-Quit
Schließt den PackageBuilder

Items

-Add Files

Über diesen Menüpunkt fügen Sie Ihre Programmdateien in das Paket ein.

-AddFolders

Über diesen Menüpunkt fügen Sie Ihre zum Programm gehörenden Ordner ein.

-AddPatch

Mit AddPatch ist es möglich Updates zu intregieren. Das heißt sie können alte Paketinhalte durch neue ersetzten, indem sie die alte und die neue Version angeben.

-Add Shell Script

Durch diesen Menüpunkt ist es möglich, Startdateien (#!bin/sh) zu intregieren. Diese Startdatei wird nach Beendigung der Installation des Paketes ausgeführt und kann z.B. Ihre Anwendung starten.

-Extract Items

Mit diesem Befehl können sie Paketinhalte vom Builder aus entpacken. Dies ist nützlich, wenn sie die Paketinhalte nicht mehr anderweitig gesichert haben.

-Delete

Durch delete löschen sie ausgewählte Paketinhalte.

-New Folder

Mit dieser Funktion können sie neue Ordner in ihrem Paket erstellen.

-Rename Item

Mit Rename Item können die Namen der Paketinhalte geändert werden.

-Select all

Mit dieser Funktion können sie alle Paketinhalte markieren.

Installation

-Generate R4...

Durch das Markieren dieser Auswahl machen sie Ihr Paket auch für die BeOS R4 Version ausführbar.

-SplashScreen

Mit SplashScreen können sie ein Logo, Cover oder Ähnliches integieren.
Es werden alle Grafikarten unterstützt, die das System anhand von Translators kennt. Um eine Grafik zu intregieren, öffnen sie SplashScreen und schieben Ihre Grafik mit Hilfe von Drag and Drop (mit der Maus anwählen und verschieben) in das Fenster des SpashScreen.

-Install Settings

In den Installations Settings können sie verschiedene Angaben zu ihrem Paket machen.

-Install Folder

Am wichtigsten in diesem Bereich wäre der Install Folder, durch dessen Aktivierung alle Paketinhalte in diesen Ordner kopiert werden (Vorraussetzung ist dabei die Einstellung unter Destination: Install Folder). Wenn sie für alle Paketinhalte direkte Installationspfade angegeben haben, sollten sie diese Einstellung deaktivieren.

-Display Folder Selection Menu überlässt dem Benutzer die Wahl des Installationsverzeichnisses. Sind Ihre Paketinhalte aber durch ihren Standort und Beschaffenheit darauf angewiesen, das nur sie die Installationsstandorte bestimmen, sollten sie diese Einstellung deaktivieren.

-Install Description

In dem zu dieser Einstellung gehörigen Textfeld können sie eine Kurzbeschreibung Ihres Paketes einfügen.

-Display Pkg Help

Durch aktivieren dieser Einstellung, wird dem User die Möglichkeit gegeben, die Hilfestellungen zu dem Paket zu lesen. Hier können sie

entweder den vorgefertigten Hilfetext verwenden, oder einen eigenen schreiben.

-Display Text...
Mit Display Text at Installer Open, wird die Install Description beim Öffnen des Paketes angezeigt.

-Package Settings
Auch in diesem Menübereich können sie verschiedene Angaben zu Ihrem Paket machen.

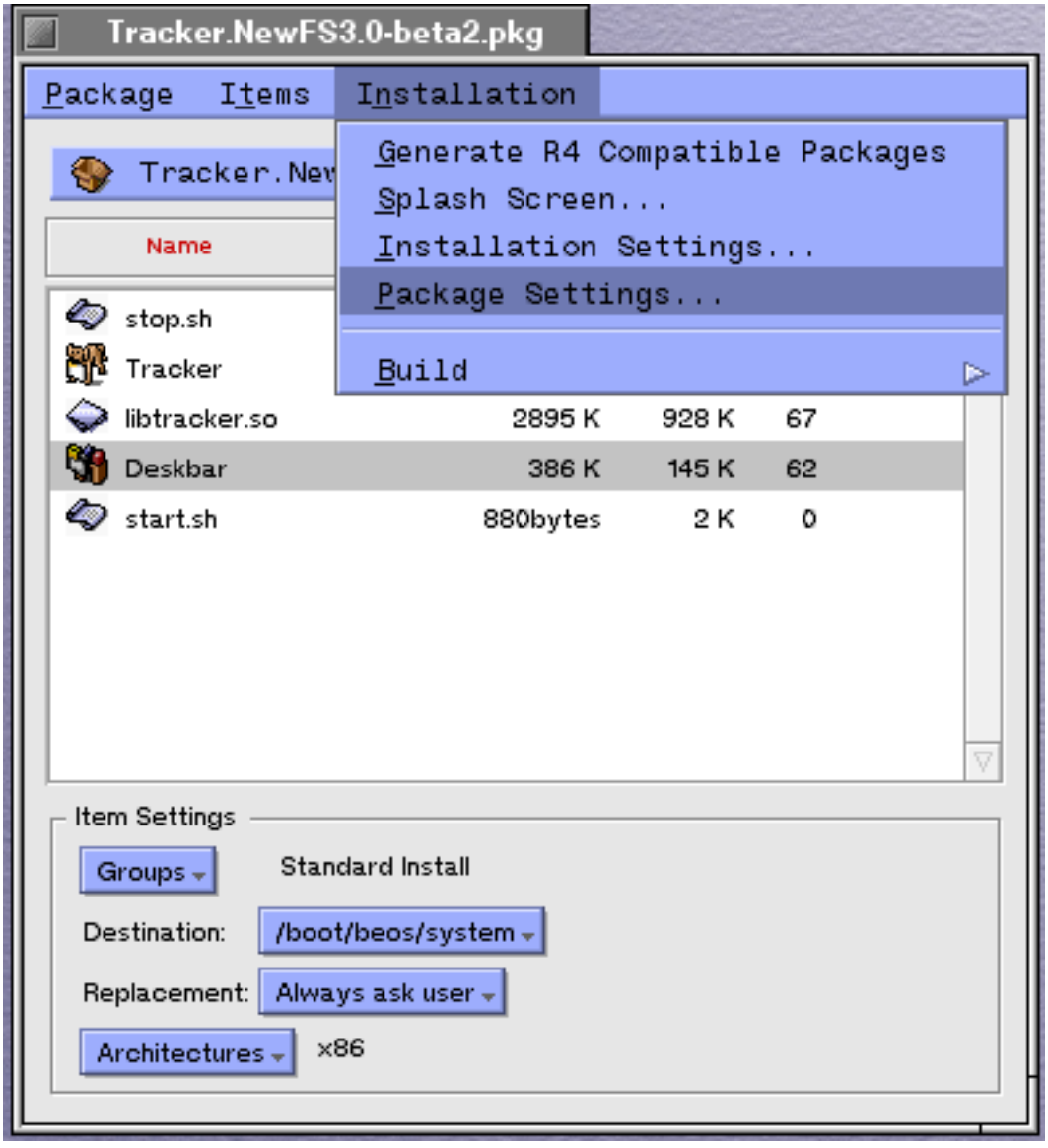
-Package Name
Name ihres Installationspaketes Version Die Version Ihres Paketes (z.B. V1.0)

-Developer
Hier können sie den Entwickler des Programminhaltes angeben.

-Release Date
Angaben zum Zeitpunkt der Erstellung dieses Paketes.

-Description
Hier können Sie eine Kurzbeschreibung Ihres Paketes machen.

-Software Type
Über diese Einstellung, können sie die Rechte Ihres Paketes angeben (z.B. Commercial, Freeware...).



Es sind noch weitere Einstellungen möglich, auf die ich in diesem Bericht nicht weiter eingehen werde. Diese dienen der genaueren Identifikation des Paketes, Registierung und Bezeichnung. Der Package Builder ist ein sehr starkes und qualitativ hohes Programm, das durch seinen enormen Funktionsumfang überzeugt.

Von Generation zu Generation

von Matthias Breiter

In der letzten Ausgabe hatte ich Euch etwas über die nahe Zukunft erzählt. Und wie wir daraus einen Gewinn ziehen könnten. Seit der ersten Technoids habe ich noch ein paar neue Ansätze gefunden, die sehr einfach verdeutlichen, worum es hier geht. Das Problem, das der PC hat (und das betrifft alle PC Systeme wie BeOS, MacOS, Windows, Linux usw...) mag nicht jedem einleuchten. Viele werden auch sagen "Heh? Welches Problem?".
Der Schlüssel liegt wie so oft in der Vergangenheit begraben, in der jungen Geschichte der Computer. Und obwohl sie noch jung ist, können wir schon jetzt interessante Gesetze beobachten.

Der Wettlauf

Keine andere Branche hat sich so schnell entwickelt wie die Computerwelt. Gestern noch Vision, Heute Realität und Morgen schon wieder veraltet. Dieses Schicksal bestimmt uns seit dem 1. April 1976. An diesem Tag nimmt die Firma Apple offziell ihren Betrieb auf und verkauft ihren Computerbausatz "Apple I" in Rekordstückzahlen. Seit diesen Stunden herrscht Krieg. Wie am Fließband erscheint ein neues Produkt nach dem anderen und nach nur einem Jahr bringt Apple den "Apple II". Computer sind die neue Lizenz zum Geld drucken und das Silicon Valley wächst wie nie zuvor.
Ständig entstehen neue Firmen, manche von ihnen werden ganz groß. So wie Commodore, wie AMIGA, wie ATARI, wie Microsoft, wie Apple, wie HP und all die anderen.
Bis 1993 entstehen jährlich neue Computersysteme, bringen neue Ideen mit, neue Lösungen, bilden neue Geräteklassen, werden zu Alltagsgegenständen oder gehen so schnell unter wie sie gekommen sind. Jahrelang zeichnet dies den Computermarkt aus. Neue Systeme bringen Innovationen, verschmelzen gute Ideen früherer Systeme zu einem neuen Ganzen. Aber auch Anwendungen profitieren. Man kopiert beim Vorgänger oder bei der Konkurrenz - den User freut es. Dokumente werden kompatibel, Netzwerke überbrücken Welten und Ideologien.
Und was für den Computer gilt, kann für Konsolen ja nicht schlecht sein. Und so preschen auch hier seit 1977 immer neue Systeme auf den Markt. Hersteller kommen und gehen, Ideen werden zu Spielen, neue Spiele bringen neue Ideen ...
Dieser Wettlauf hat die Computer und die Konsolen immer wieder neu erfunden, neu definiert und immer neue Märkte erschlossen.

Ist das so?

Am besten einfach mal kurz die Zeitung weglegen und darüber nachdenken. Schaut Euch auch mal die Tabelle an (Seite 18), in der die zeitliche Entwicklung verschiedener Systeme aufgeführt ist.
Klar, es ist so, weil es wirklich so passiert ist- das ist die Realität. Davon allein haben wir noch nichts. Aber wir können etwas daraus lernen. Halten wir einfach mal die prinzipiellen Erfahrungen fest:
- Hersteller XY entwickelt neues Produkt
- lässt eigene Erfahrungen und Ideen einfließen
- übernimmt Ideen aus älteren Produkten
- baut neues, besseres Produkt

Ungeachtet davon, ob Hersteller XY nun damit Erfolg hat oder nicht, hinterlässt er doch ein Erbe. Denn die Ideen bleiben im "Produkt manifestiert" erhalten. Je einleuchtender seine Ideen waren, desto öfter werden sie übernommen. Manchmal sind es große Dinge, wie z.B. die grafische Oberfläche des Apple

Macintosh. Oft sind es aber auch nur Kleinigkeiten oder liebevolle Details.

So, nun kann man sagen: "Lieber Matthias. Das ist doch logisch. Es läuft doch immer und überall so ab."ssss Wirklich? Darauf kommen wir an anderer Stelle nochmal zu sprechen...
Jetzt müssen wir aber erst noch ein typisches IT- Element aufgreifen. Denn ein Computer oder eine Konsole ist ja (im Gegensatz zu einem Auto) nicht nur von einem Hersteller abhängig, sondern von vielen.

Einer für Alle - Alle für Einen

Jedes neue System ist ein neuer Anfang. Ein unbeschriebenes Blatt Papier. Der Hersteller liefert erste Vorschläge, Programme und vielleicht auch Spiele. Viel mehr kann er nicht tun. Natürlich gibt es Firmen, die immer wieder mal selbst neue Software liefern, aber den Hauptanteil haben stets die Dritthersteller. Auf jedem System setzen sich diese aus erfahren Veteranen und neuen, kleinen Studios zusammen. Manchmal sind es gar nur einzelne Personen. Dadurch, das neue Plattformen wie ein unbeschriebenes Blatt Papier sind, können neue Entwickler ebenso gut einen Hit landen wie die "alten Hasen".
Ein neues System ist also folglich eine Chance für junge Talente und kleine Firmen. Ideen lassen sich unvoreingenommen umsetzen und man profitiert davon, das auch "die Großen" Zeit brauchen, um wirklich überzeugende Arbeit abzuliefern. Das lässt sich auch historisch belegen. Jedes neue System hat neue Spiele, neue Anwendungen und neue Firmen hervorgebracht, die es vorher nicht gab.
Und selbst wenn das ganze System floppt- wer sich als Entwickler einen guten Namen gemacht hat, ist auf anderen Plattformen gern willkommen. Und seine Ideen wandern so auf andere Produkte über, werden wieder von anderen aufgegriffen und alles wird von Generation zu Generation vererbt.

Ist es wirklich immer so?

Um auf den Punkt zurückzukommen, das sei doch immer so. Mhh, schaut Euch mal die Tabelle an (nächste Seite). Fällt Euch nix auf? In der Tat, bei Konsolen ist das immer so. 1993 hört die Entwicklung im Computerbereich (mit Ausnahme von BeOS) plötzlich auf. Das war zu dem Zeitpunkt, als der PC den Computermarkt aufgerollt hat. Seit gut 10 Jahren geht es im Computerbereich nicht vorwärts. Die Grundlagen des PC stammen noch aus den 80'ern, sind also über 20 Jahre alt. Was hat sich denn schon groß verändert? Ein PC fühlt sich seit eh und je an wie ein PC. Die wichtige Erneuerungen hat er seit über 10 Jahren nicht mehr erfahren.

Am Anfang der 90er war der PC ein Hammer. Spiele wie Wing Commander III, Need for Speed, X-WING, Lucas Adventures, Command & Conquer usw... brachten den PC unglaublich weiter. Doch dann bildete sich eine Stagnation. Keine neue Plattform, keine neue Software, keine neuen Ideen. Es wird Zeit für ein neues Produkt. Ob es scheitert oder ob es Erfolg hat ist irrelevant- Hauptsache die Welt bekommt endlich mal wieder einen wirklich neuen Computer. Und dann werden wieder neue Entwickler gewonnen, neue Firmen gegründet, neue Ideen weiter verfolgt und das Wissen wird weiter gegeben - von Generation zu Generation.

Und was hat das jetzt mit BeOS zu tun?

Technoids ist ja ein praktisches Magazin ohne ideologisches Bla Bla. Und das wird auch so bleiben. Jetzt hatten wir hier gerade ganz schön viel Bla Bla, da stellt sich die Frage, was man jetzt tun kann.
Wir halten ein mächtiges System in unseren Händen. YellowTAB verfügt mit ZETA über den (wahrscheinlich) letzten Stand der BeOS Entwicklung. Diese Tatsache macht es zum potentiell gefährlichsten System für den PC Markt. Aber, ein Betriebssystem alleine macht noch keinen neuen Computer. Das ist auch der

Grund, warum alle scheitern. Es gibt viele gute Systeme, die keine eigene Plattform und keinen Erfolg haben. Nehmen wir z.B. Apple. Trotz der geringen Verbreitung ist Apple ein reicher und großer Hersteller, der über viele Entwickler und Dritthersteller verfügt. Selbst die ACORN RiscPCs werden in England immer noch eingesetzt und diverse Hersteller produzieren munter weiter. Trotz des hohen Preises für die "feine, englische Art" hat das System viel mehr Anwender als BeOS.

Ich sage nicht, das wir jetzt bei Hersteller XY klingeln sollen "Hey Leute, baut uns doch mal nen Computer". Aber denkt mal darüber nach. Als M$ kaum größer war als YT hat man dort erfolgreich die MSX- Plattform vermarktet- obwohl die MSX Hardware aus handelsüblichen Komponenten bestand.

Warum heute keiner auf so eine Idee kommt, ist auch klar. Wenn ich 10 Jahre mit dem (vom Prinzip her) gleichen PC arbeite, habe ich mich so daran gewöhnt, das mein Horizont ziemlich schmal wird.

Jahr	Computer	Konsole
---	---	---
1976	**Apple I**	
	Commodore PET	
1977	**Apple II**	**ATARI VCS (2600)**
1978	ATARI 400/800	
1980	**Apple III**	**Colecovision**
	ACORN Atom	
	Sinclai ZX80	
1982	**ACORN BBC Micro**	**COLECO Colecovision**
1983	Apple LISA	ATARI 5200
	ASCII / MS MSX	
1984	**Apple Macintosh**	
1985	ATARI ST	Nintendo NES
	AMIGA A (1000, 500...)	
	MSX 2	
1986	**Apple IIgs**	**ATARI 7800**
	NeXT	SEGA Mastersystem
	Robotron KC-85	SNK Neogeo
1987	**ACORN Archimedes**	
1988	NeXT Cube	SEGA Megadrive
	ASCII/MS MSX 2+	NEC PC Engine
	Olivetti PC 1	
1989		**ATARI LYNX**
		Nintendo Gameboy
1990	AMIGA A+ (3000)	Nintendo SNES
		SEGA GameGear
1992	**AMIGA AGA (1200,4000)**	
	Linux	
1993	**Apple PowerMacintosh**	**AMIGA CD32**
		ATARI JAGUAR
1994	ACORN RiscPC	SONY Playstation
		3DO FZ-1
1995		**SEGA SATURN**
		Nintendo N64
1996	BeBox / BeOS	Apple Pipin
1998		**SEGA Dreamcast**
2000		Playstation 2
2001		**Microsoft X-BOX**
2002		Nintendo GameCube

CIFS Mount

Wie man auf Windows- und SAMBA- Freigaben übers Netzwerk zugreift

von Adam Szczech (Übersetzung aus dem Englischen von Matthias Breiter)

Hier beschreibe ich, wie man unter BeOS "Netzwerkfreigaben" von Windows (SAMBA) sehen und benutzen kann.
Viele werden schon wissen, dass man unter BeOS, ähnlich wie unter UNIX, Laufwerke erstmal mounten muss, um damit zu arbeiten. Das geht am einfachsten im "Drives"- Fenster, indem man dort die Rechte- Maustaste drückt und dann auf "Mount" klickt. Ebenso ist es möglich, entfernte Laufwerke über das Netzwerk zu mounten. Unter Windows wird dieser Vorgang "Netzlaufwerk verbinden..." genannt. Das gleiche kann man auch unter BeOS machen.

Anforderungen

Ganz elementar müssen wir folgende Dinge wissen, um auf das Laufwerk (der Freigabe) eines Kollegen zuzugreifen:

a. Unser Netzwerk muss natürlich konfiguriert sein (TCP/IP)!
1. IP Adresse (TCP/IP) unseres Kollegen
2. Name seines Computers (kann im Windows unter "Arbeitsplatz: Eigenschaften" nachgelesen werden, z.B. "JOHN SMITH")
3. Name der Arbeitsgruppe ("Workgroup"), ebenfalls dort nachzulesen
4. Name der Freigabe (z.B. MUSIK, GAMES, FILES). Dieser Name wird unter Windows in der "Freigabe" eingestellt
5. Einen Ordner, in dem wir die Dateien aus der Freigabe unter BeOS anzeigen wollen (z.B. /boot/home/Dektop/Files)

Unser Kollege kann uns diese Daten bereit stellen und sollte folgende Dinge beachten:
1. TCP/IP muss als Protokoll aktiviert und konfiguriert sein. Auf NetBIOS (NetBEUI) allein können wir nicht zugreifen.
2. Er muss seine IP Adresse fest einstellen bzw. kennen, falls er mit DHCP arbeitet (am einfachsten in der DOS- Eingabeaufforderung "ipconfig" eingeben).
3. Die Option "Datei und Druckerfreigabe" muss aktiviert sein und es muss natürlich mindestens eine Freigabe geben.

Allgemeines zu CIFS Mount

Über das Common Internet File System CIFS (allgemeines Internet Dateisystem) haben wir die Möglichkeit, Laufwerke und Ordner über das Netzwerk unter BeOS zu öffnen. Auf anderen Systemen heißt das auch SMB oder NetBIOS. Selbstverständlich kann man damit auch SAMBA- Freigaben öffnen, die unter UNIX, Linux, MacOS X usw... erstellt wurden.

Um nun eine Freigabe unter BeOS zu mounten, benötigen wir das kleine Programm "cifsmount". Es ist eigentlich immer bei BeOS dabei und unter "/boot/beos/bin" auf unserem Rechner zu finden. Sollte dies nicht der Fall sein, einfach schnell von www.bebits.com runterladen.
Es empfiehlt sich zu prüfen, ob auf unserem System ebenfalls "ksocketd" vorhanden ist. Die Datei muß unter "/boot/home/config/bin/" oder "/boot/beos/bin/" vorhanden sein, sonst funktioniert CIFS Mount nicht.

Schritt für Schritt

"cifsmount" ist ein Terminalprogramm. Keine Angst, trotz der antiquiert erscheinenden Texteingabe ist cifsmount ganz einfach zu benutzen.
Zunächst öffnen wir erstmal ein Terminal. Z.B. über das Be- Menü, dann auf "Applications" und dann auf "Terminal" drücken. Ein neues Fenster mit einer DOS- ähnlichen "Oberfläche"

geht auf. Bevor wir jetzt blöd drauf los tippen, hier mal die grundlegende Syntax des cifsmount- Befehls:

cifsmount -I <IP Adresse> -W <WORKGROUPNAME> -d \\\\ <computer_name> \\<FREIGABENAME> <login> <password> /<mountFolderPath>

Die Werte in den <> Klammern sind Variablen, die wir gegen unsere Werte austauschen müssen. Die Klammern im echten cifsmount- Befehl bitte weglassen! Hier mal im Detail:

<IP Adresse>- die IP Adresse des Computer, den wir ansprechen wollen (von dem wir ein Laufwerk mounten möchten)

<WORKGROUPNAME> - der Name der WORKGROUP bzw. Arbeitsgruppe, die auf dem Windows- Rechner eingestellt ist. Wichtig: Alle Buchstaben groß schreiben! Normalerweise ist die Angabe von WORKGROUPNAME nicht erforderlich.

<computer_name>- Name des Computers. Auch der ist auf dem Windows- Computer eingestellt.

<FREIGABENAME>- Der Name der Freigabe auf dem Windows- Computer. Nicht der Laufwerksbuchstabe, der dort zugewiesen wurde!!!
Wichtig: alle Buchstaben groß schreiben, auch wenn unter Windows der Name klein geschrieben wurde!

<login>- der Benutzername, mit dem wir uns anmelden. Kann der Benutzername des Eigentümers aber auch jeder andere, unter Windows eingestellte Benutzer sein.

<password>- das zum Benutzernamen zugehörige Passwort
<mountFolderPath>- das Verzeichnis unter BeOS(!), in dem die Freigab erscheinen soll. Dieses Verzeichnis muss bereits existieren!

Beispiel

So, nun haben wir CIFS Mount ausführlich kennen gelernt. Nun ein praktisches Beispiel. Der Computer des Kollegen hat die IP Adresse 192.168.2.44, steht in der Arbeitsgruppe WORKGROUP und heißt "TOM". Die Freigabe heißt "GAMES". Als Benutzernamen hat er "Paul" für uns angelegt, als Passwort "1234". Wir haben als Beispiel einen Ordner "Files" auf dem BeOS Desktop angelegt, den man unter "/boot/home/Desktop/Files" erreicht. In dieses Verzeichnis wollen wir die Freigabe "GAMES" unter BeOS sichtbar machen.

cifsmount -I 192.168.2.44 -W WORKGROUP -d \\\\TOM\\GAMES Paul 1234 /boot/home/Desktop/Files

Wunderbar. In dem Ordner "Files" auf unserem BeOS- Desktop erscheinen nun die Dateien aus der "GAMES"- Freigabe. Wie bereits erwähnt, ist es eigentlich nicht erforderlich, die Arbeitsgruppe (hier "WORKGROUP") mit anzugeben. Es kann auch sein, das der Kollege unter Windows gar kein Passwort verlangt. Dann können Sie anstelle des Benutzers und des Passwortes irgendetwas eingeben, z.b. jeweils "egal". Das müssen Sie dann aber mit Ihrem Windows- User ausmachen.

Würden wir folgendes schreiben:

cifsmount -I 192.168.2.44 -d \\\\TOM\\Games Paul 1234 /boot/home/Desktop/files

bekämen wir einen Fehler, da der Freigabename *immer* groß geschrieben werden muß, hier also "GAMES". Das weglassen der WORKGROUP hat hingegen keinen Einfluss auf die Funktion.

Vereinfachungen mit Host- Datei

Wer mehrere Laufwerke (bzw. Freigaben) eines Nutzers mounten möchte, oder immer wieder auf die gleichen Computer zugreift, wird sich wünschen, den cifsmount vereinfachen zu können. Auch dies ist mit einfachen Mitteln möglich. Unter "/boot/beos/etc/" findet sich die Datei "hosts". Öffnen Sie die Datei mit einem beliebigen Editor, z.B. mit dem BeOS eigenen "StyledEdit". Eine Hostdatei kann so aussehen (beispielhafter Auszug!):

192. 168. 2. 22 frank
192. 168. 2. 34 george
217. 117. 128. 4 elena

Neben den IP Adressen wird hier auch der Hostname abgesichert. Der Hostname ist für cifsmount gleichbedeutend mit dem Computernamen. Wenn wir nun die Zeile "192.168.2.44 TOM" ergänzen, kann cifsmount automatisch auf diesen Host zugreifen. Sie können also beliebig viele Hosts eintragen, in dem Sie die IP- Adresse und den Namen eingeben und die Datei "hosts" dann abspeichern und schließen.

WARNUNG: Wenn Sie in Ihrem Netzwerk DHCP benutzen, das IP Adressen automatisch vergibt, können Sie nicht mit der Hostdatei arbeiten. Nach jedem Neustart stimmt zu 99% die IP Adresse nicht mehr.

```
cifsmount \\\\TOM\\GAMES Paul 1234
/boot/home/Desktop/files
```

Jetzt greift cifsmount automatisch auf die Datei "hosts" zu und wir können den kompletten Anfang der Variablen (IP Adresse) weglassen. Falls Sie darauf angewiesen sind, Arbeitsgruppen anzugeben, können auch diese in der Datei "/boot/home/config/settings/network" gespeichert werden. Dies ist aber nur in Ausnahmen erforderlich.

Vereinfachung mit ezmount

ezmount (sprich Isi- Mount, also englisch für "einfaches mounten") benutzt cifsmount und kann den Befehl drastisch vereinfachen:

ezmount <IP Adresse> <FREIGABENAME> <login> <password>, in unserem Beispiel also:

```
ezmount 192.168.2.44 GAMES Paul 1234
```

Bei ezmount können wir jedoch keinen Ordner angeben, in dem die Daten der Freigabe erscheinen sollen. Sie tauchen immer im Verzeichnis "/ezmount" (im BeoS Stammverzeichnis "/") auf. Auch ezmount wird vom Terminal aus aufgerufen.

Freigaben abmelden (unmount)

Ebenso wie man sich an Freigaben anmelden kann, kann man sich natürlich auch wieder abmelden, wenn die Daten nicht mehr benötigt werden oder der Computer (oder die Freigabe) im Netzwerk nicht mehr zur Verfügung stehen. Der Befehl dafür ist kinderleicht und wird ebenfalls im Terminal eingegeben:

```
unmount <mountFolderPath>
```

<mountFolderPath> gibt ja das Verzeichnis an, in dem die Freigabe erscheint. In unserem Beispiel führt der Befehl

```
unmount /boot/home/Desktop/files
```

dazu, dass die Freigabe abgemeldet wird und die Dateien wieder verschwinden. Dies sollte man wirklich immer machen, sobald man die Freigabe nicht mehr benötigt, um somit das Netzwerk zu entlasten.

Grafische Tools

Von der Terminal- Eingabe sollten sich auch Anfänger nicht abschrecken lassen. Wenn man die verfügbaren Daten hat, ist das Verbinden zu Windows- Freigaben ganz einfach. Dennoch

möchte ich hier die grafischen "Front- Ends" nicht unerwähnt lassen. Sie können mit der Maus bedient werden. In der Praxis zeigen sie sich jedoch immer wieder als unausgereift, fehleranfällig und unzuverlässig. Auch das an sich tolle Programm "World o' Networking (WON)", das automatisch alle Windowsfreigaben (und auch SAMBA Freigaben) im Netzwerk anzeigt, ist beim kopieren von Dateien leider viel zu instabil. Daher meine Empfehlung: cifsmount benutzen- funktioniert immer.

Zu guter Letzt ein Script

Die mächtigen Scripting- Fähigkeiten haben wir ja schon in der ersten Technoids kennen gelernt. Ähnlich können wir auch cifsmount automatisieren. Dies funktioniert natürlich nur, wenn wir feste IP Adressen verwenden!!! Zunächst brauchen wir wieder einen Ordner, in den wir die gemounteten Daten legen. Für das Beispiel dient hier wieder "Files" auf dem Desktop. Nun erstellen Sie in einem beliebigen Textprogramm folgendes Script:

```
#!/bin/sh
echo Verbinde zum Windows- Netzwerk ...
cifsmount -I 192.168.2.44 -W WORKGROUP -d
\\\\TOM\\GAMES Paul 1234
/boot/home/Desktop/files
echo Fertig!
```

Diese Datei müssen Sie natürlich an Ihre eigene Netzwerkadresse anpassen. Ebenfalls können Sie mehrere cifsmount hintereinander schreiben, oder verschiedene Dateien für verschiedene Windows Freigaben erstellen. Speichern Sie diese Datei nun, z.B. als mount_tom.txt. Merken Sie sich das Verzeichnis- für schnellen Zugriff können Sie es auch auf dem BeOS- Desktop speichern. Im Terminal navigieren Sie nun in das Verzeichnis, in dem sich die so eben erstellte Date befindet.

Geben Sie

```
chmod +x mount_tom.txt
```

ein. Ab sofort ist die Datei ausführbar (das x steht für executeable) und beim Aufruf der Datei wird die angegebene Freigabe verbunden.

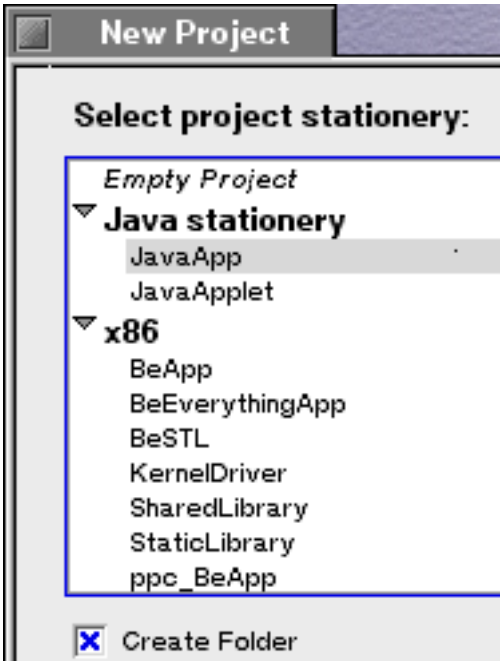
Java -Tutorial

von Florian Thaler

Mit diesemTutorial möchte ich diejenigen, die unter BeOS mit Java programmieren möchten, helfen. Voraussetzung für das java-Programmieren ist bekaffe, ein java compiler. Man findet ihn unter www.bebits.com. Nach dem Runterladen entpackt man die bekaffeDR.zip Datei ins /boot/home Verzeichnis. Dabei wird automatisch der CLASSPATH gesetzt. Ab nun kann man überall im Terminal "javac" und "java" aufrufen. Hat man die BeIDE installiert (empfohlen), kann man sich das JikesDevKit (981kb) von bebits downloaden und installieren. Damit kann man beim Öffnen der BeIDE das Plugin JavaApp oder JavaApplet auswählen. Während dem schreiben des Codes unter BeIDE ist somit syntax-highlighting möglich und der Quelltext wird übersichtlicher. Leider sind wir, was die Java-Version anbelangt, noch im letzten Jahrtausend, denn die Version 1.1 ist von 1999. Das schränkt uns zwar in einigen Dingen ein, soll uns aber nicht davon abhalten ein kleines Programm zu schreiben.
Wir schreiben den Quelltext in BeIDE und speichern die Datei mit der Endung .java in einem beliebigen Verzeichnis ab.
Dann öffnen wir eine Konsole und wechseln in das Verzeichnis, wo wir unsere Java-Datei (beispielsweise MyJavaProg.java) abgespeichert haben. Nun führen wir den Befehl "javac MyJavaProg.java" aus. Achtung auf die korrekte Schreibweise, Groß- und Kleinbuchstaben werden unterschieden. "javac" kompiliert den vorhandenen Quelltext und wandelt ihn in ByteCode um, der für die Java Virtual Machine(JVM) verständlich ist. Diese Java Virtual Machine ist bekaffe unter BeOS. nun können wir unser Programm mit "java MyJavaProg" aufrufen und es wird über die JVM gestartet. Ebenso könnten wir unter /boot/home/bekaffe/ das Programm BeKaffe aufrufen und mit "file", "open" unsere Datei suchen. Achtung: immer eine Datei mit Endung -class auswählen, sonst passiert nichts. Das Programm öffnet dann das gewünschte eigene Programm. Sollte sich keine Datei mit der Endung .class im vermeintlichen Verzeichnis befinden, wurde die .java Datei wohl noch nicht kompiliert.

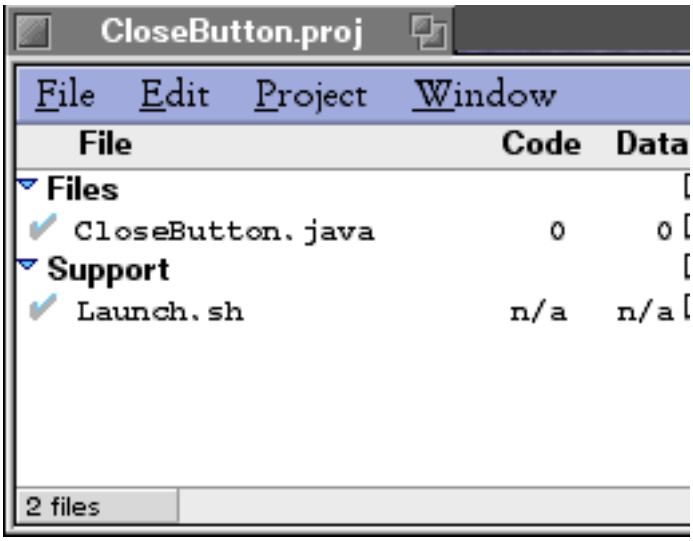
Kurz und klar nochmal die Schritte zu einem Java-Programm:
- Benötigte Programme installieren (BeIDE, bekaffe, JikesDevKit)
- Quelltext schreiben
- Quelltext mit der Endung .java abspeichern
- Terminal öffnen, ins Verzeichnis wechseln, wo die Datei gespeichert ist
- Quelltext im Terminal mit "javac MyJavaProg.java" kompilieren
- ByteCode mit "java MyJavaProg" ausführen

So, und nun erstellen wir ein kleines Programm, das ein Fenster erzeugt und einen Button zum Beenden des Programmes besitzt. Wir öffnen die BeIDE und wählen JavaApp aus der Liste aus.(wie im Bild)



Nach dem Klicken auf "Create" werden wir aufgefordert, einen Namen für das Projekt einzugeben. Geben wir dem Projekt auch den Namen CloseButton.

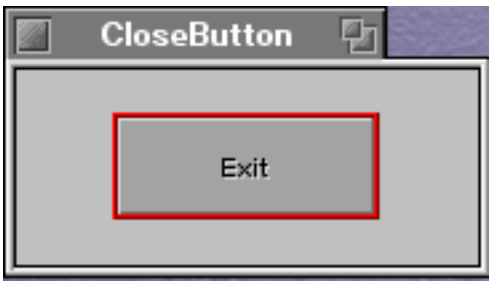
Nach dem Speichern erscheint ein Fenster mit dem Namen CloseButton.proj. Wir löschen unter "Files" die bereits vorhandene Datei und erzeugen mit "File", "new text" eine neue Datei, die wir unter CloseButton.java abspeichern. Nicht das Häkchen bei "add to project" vergessen!!
Nun sollte das Fenster folgendermaßen aussehen.



Nun schreiben wir folgenden Quelltext in die eben erstellte Textdatei: (grüner Text ist "nur" Kommentar) SIEHE SEITE 26!

Mit Rechtsklick auf die Dateien im CloseButton Fenster können wir "compile" auswählen, so wird diejenige Datei kompiliert und bei eventuellen Fehlern eine Meldung angezeigt.
In der Datei Launch.sh, die wir mit einem Linksklick öffnen, ändern wir in der zweiten Zeile das HelloWorldApp in CloseButton.
Kompilieren wir nun Launch.sh, wird unser Programm ausgeführt. Die andere Methode zum kompilieren und ausführen von Java Programmen habe ich bereits weiter oben beschrieben (durchs Termninal).

Unser CloseButton-Programm sollte nach dem Ausführen so aussehen:



Beim Drücken des "Exit" Buttons sollte sich das Fenster schließen.

(Erklärung nächstes Seite)

Noch einiges zur Verständlichkeit des Kodes:

// = der Rest der Zeile wird vom Kompiler ignoriert

/* Kommentar */ = mehrzeiliger Kommentar steht zwischen solchen Zeichen

import = wie include in C/C++, die Dateien die wir benötigen werden importiert

void = definieren einer Funktion

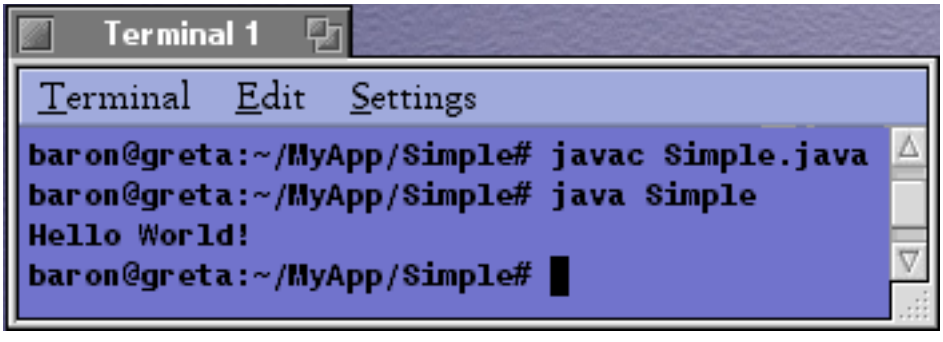
private/public = Sichtbarkeit der Funktion bzw. Variablen, in unserem Programm nicht relevant, da CloseButton nur aus einer Datei besteht

public static void main (String [] args) {
...
} = Hauptklasse, ohne die läuft nichts

Das allereinfachste Programm ist:

```
class Simple {
  public static void main (String [] args) {
  System.out.println ("Hello World!");
  }
}
```

einfach in eine StyledEdit TextDatei schreiben und unter Simple.java abspeichern, im Terminal kompilieren und ausführen.
Also z.B.
- cd /boot/home/MyJavaApps
- javac Simple.java
- java Simple



Nun sollte man die Ausgabe "Hello World!" sehen.

Ich hoffe es hat euch Spaß gemacht und ihr habt alles verstanden!

```
-----------------------------------------------------------
import java.awt.*; //zum Erstellen von Fenstern
import java.awt.event.*; //zum Behandeln von Fensterereignissen

public class CloseButton extends Frame { //wir erzeugen eine Klasse CloseButton und erweitern sie
mit Frame, d.h. es wird eine Fensteranwendung
        Frame f = new Frame("CloseButton"); //erstellen eines neuen Fensters mit Fenster-Namen
        Button button = new Button("Exit"); //wir erzeugen einen neuen Button mit dem label "Exit"

        public CloseButton() {      //erzeugen eines Konstruktors, dieser Konstruktor wird durch
                                    "new CloseButton()" in der main-Funktion aufgerufen
                System.out.println ("Starting CloseButton..."); //auf der Standardausgabe wird
                "Starting CloseButton..." ausgegeben; wenn wir das Programm aus dem Terminal
                starten, sehen wir diese Zeile, öffnen wir das Programm unter Bekaffe, nicht;
                components(); //Aufruf der Funktion components,Fenster+Button werd.gezeichnet
        }
        public void components () { //Die Funktion components, hier deklarieren wir die
        verschiedenen Komponenten des Programmes;
        f.setLayout (null); //Java beinhaltet vorgegebene layouts, hier setzen wir es auf NULL, weil
        wir unsere Objekte selber positionieren wollen.
        f.setBounds (200,200,180,100); // so setzen wir Position und Größe des Fensters f, und
        zwar x,y,breite,höhe (x+y = wo auf dem Bildschirm das Fenster begonnen werden soll
        (linke obere Ecke)) Einheiten sind in Pixel gehalten.
        button.setBounds (40,40,100,40); // so setzen wir Position und Größe des Buttons, x+y
        stehen hier für die Position innerhalb des Fensters
        f.setResizable (false); // setzen wir diese Funktion auf "false", ist unser Fenster nicht in der
        Größe veränderbar; schreiben wir diese Zeile nicht oder ersetzen "false" mit "true", ist das
        Fenster skalierbar
        f.add(button); // hier führen wir unserem Fenster unseren Button zu; wenn diese Zeile fehlt,
        bleibt das Fenster ohne Button!
                button.addActionListener ( new ActionListener() {
                        public void actionPerformed (ActionEvent aev) {
                                System.exit (0);
                }}); // So weisen wir dem Button eine Funktion zu (addActionListener) und teilen ihm
                mit, dass bei Betätigung des Buttons (actionPerformed) ---> System.exit(0);
                ausgeführt werden soll. Das bedeutet das Ende des Programms (mit System ist das
                laufende Programm gemeint, nicht das Betriebssystem)
        f.setVisible (true); // erst hier lassen wir unser Fenster sichtbar werden (nachdem alle
        Komponenten, bei uns nur ein Button, dem Fenster hinzugefügt wurden)
        }
        public static void main (String [] args) {
                new CloseButton();
        } //unerlässlich ist die Methode main(), und sie muss so defnniert werden wie hier
vorgegeben; Bei Aufruf des Programmes wird nur diese Funkion aufgerufen, was hier drinnen
steht wird ausgeführt. In unserem Fall erzeugen wir ein neues Objekt der Klasse CloseButton();
}
-----------------------------------------------------------
```

Tips und Tricks

von Florian Thaler

Fenster und Workspaces

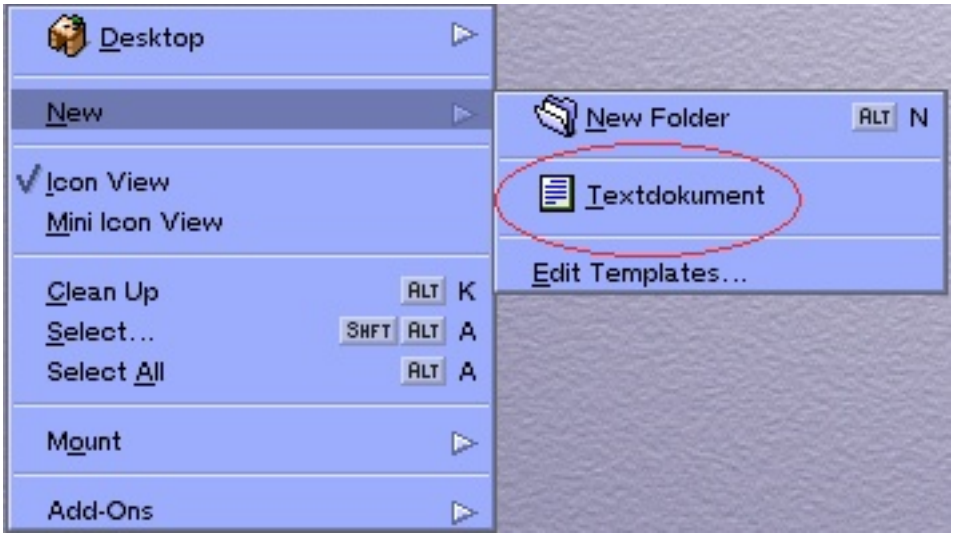
Will man zwei Programme, die in verschiedenen Workspaces laufen, in einem Workspace zusammenführen, ohne eines schließen zu wollen, macht man folgendes: Man hält ein Fenster fest (mit der linken Maustaste gedrückt) und wechselt einfach in den Workspace wo das andere Programm läuft. Beispiel: Das BeShare-Fenster in WS (Workspace)1 soll zum Net+ Fenster in WS 2. Also wechselt man zu WS 1, hält das BeShare Fenster mit der linken Maustaste fest und drückt "ALT"+"F2".
(ich benutze die Developer Ed.1)

BeOS Radio

BeosRadio ist ein Internet-Radiosender, der ausschließlich Lieder von BeOS-Mitglieder spielt. Stündliche News machen den Sender auf jedenfall hörenswert. Um nicht jedesmal auf die Homepage des Senders http://www.beosradio.com zu surfen, um den Sender zu hören, gibt es "Probe for Beos Radio". Man wählt einfach die Verbindungsgeschwindigkeit und klickt auf den Link daneben. Es öffnet sich CL-Amp (sollte vorhanden sein) und schon bald hört man die ersten Klänge.

Tracker

Oft findet man beim Rechtsclick der Maustaste im Tracker oder auf dem Desktop unter dem Menü "New..." nur den Eintrag "New Folder" und "Edit Templates...". Um zum Beispiel auch "Neu", "Textdokument" auswählen zu können, was wirklich seinen Nutzen hat, öffnet man StyledEdit und speichert das leere Dokument unter "/boot/home/config/settings/Tracker/Tracker New Templates" als "TextDokument" (oder was auch immer man will) ab. So kann man beliebig viele "New"-Einträge anlegen. Erstellt man so ein neues Textdokument (wie im Bild), so wird diese Datei auch gleich mit dem Standardprogramm geöffnet (festgelegt unter "Preferences", "File Types").



Impressum

Hier würde die Vorschau für die nächste Ausgabe stehen, da wir die Artikel aber noch nicht erahnen können, es sind immerhin zwei Monate bis zur nächsten Ausgabe, werden wir diesmal keine Vorschau geben, sondern Euch Ende des Monats auf http://www.technoids.tk einen News-Artikel schreiben, der den wichtigsten Inhalt der Technoids 03/2003 beinhaltet.
Jeder von Euch kann uns gerne einen Artikel schreiben, muss nicht in deutscher Sprache sein.

Bei Interesse bitte per e-mail melden.

Schreibt uns ebenfalls, welche Artikel Euch an der Ausgabe gut gefallen haben, welche weniger, und Eure Lieblingsthemen zu BeOS. Wir möchten das Magazin so interessant wie möglich hinbekommen, und auf jeden Fall unser Ziel erreichen, das wäre, euch zu informieren, nicht zu langweilen mit Dingen die ihr eh schon wisst. Also etwas Neues sollte für jeden BeOS Benutzer in der Ausgabe stehen.

So können wir uns auch sicher sein, dass ihr auch nächsten Mal wieder die Technoids downloaden werdet, und das schätzen wir sehr.

Also, ran an Mail-It, BeMail, Scooby etc. und schreibt uns in ein paar Zeilen als Mail. Oder postet was auf die Technoids-Homepage, damit jeder Eure Meinung lesen und dazu Stellung nehmen kann.

Vielen Dank für Euer Interesse,
Euer Technoids Team

Technoids-Team

Chefredakteur: Matthias Breiter

Feste Redakteure: Christian Albrecht, Florian Thaler

Beiträge zu dieser Ausgabe leisteten ebenfalls:
- Holger Wendenburg
- Adam Szczech
- Joao Carvalho (Layout-Vorschlag)

Grafik und Layout: Florian Thaler, Matthias Breiter

Testleser: Tonio und Hendryk

Danke an die Personen, die uns per e-mail Fragen beantwortet haben sowie den Testlesern für das Heraussuchen der 1001 Fehler! :-)

Technoids erscheint 2monatlich jeweils zum ersten des Monats. Technoids ist Freeware. Das Heft darf von privat zu privat kostenlos weitergegeben werden.

Matthias Breiter

Kontakt:
technoids@morgentau.org