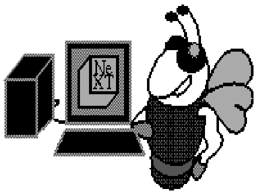


BuzzNUG
Buzzings
April 1990 • ISSUE 5
Journal of the NeXT Programmer's Community

WELCOME

Erica J. Liebman

Welcome to, potentially, the last issue of Buzzings. As of next month, we are hoping to change our name, and as you can see already, our style. We BuzzNUGians are grateful to Frame Inc who have graciously donated a copy of FrameMaker to the cause. When you have any reason to contact Frame, don't forget to pass on our thanks. Special thank yous go out to Bill Edwards of Frame and Steve Sieczko of NeXT.

On a more serious note, a number of articles were editorially mangled in the March issue. In several cases, the original intent and meaning of the authors were changed to the detriment of the articles. Let me emphasize, our authors are our single **most** valuable resource. I have taken steps to assure that this should not happen again. I sincerely regret the disservice to the authors and our readers. I am counting on each and every one of you out there to help contribute to the BuzzNUG effort. Your time, your effort and your labor is both appreciated and vitally needed.

Buzzings is going through a number of changes in its metamorphosis to our new form. We are hoping to add some regular columns. First, a third-party developer watch, where developers can list their ongoing projects and status in addition to the regular listings of shipping products in *Market View.*

Second, we are looking to add an intermittent "Government Watch" column. As far I know, currently three vendors are vying for NeXT Specific Government contracts. We hope to list vendors and target markets.

As you are well aware, one of our targets is to go "paper" in the months ahead. With this new category of readers, we intend to add a third column: "Best of the Net", which details the most interesting fixes and hints and rumors from comp.sys.next. Although getting author permissions for this makes my hair stand on end, I think a non-net readership will greatly appreciate the wisdom, knowledge and bon-mot's of our networking gurus and guruettes. In the course of a month a lot of words go by on the network, I hope this will help to weed out some of the most valuable information passed.

For the time being, I will also be continuing the new column "the myopic eye", started last month focussing on brief reviews and rumors.

As always, keep in touch. Let me know what is going well and what isn't.

HEADLINE NEWS

- The NeXTStep development team won the Andrew Fluegelman Award sponsored by PC World, MacWorld and SPA.
- Our BuzzNUG Friend, Conrad Geiger, has been promoted to director of Western Sales -- Atta boy, Conrad
- Another friend, Tom Poiker, has put out the first issue of a new Newsletter, **NeXTView**. We even got a mention.
- Awesome BaNG User Group Mtg, for May see User Group Section

INDEX

Contributors:

Erica J. Liebman, Pat McGee, Andrew Stone, Austin Meredith, Jiro Nakamura, Mark ^Zimmerman, Dave Joerg, Daniel A. Powers, Dan Port, Bryce Jasmer, Eric P. Scott, Doug Kieslar

•Editorial Matters•
1150 Collier Road NW/L-12
Atlanta, GA
30318

Editorial Stuff

Articles for Buzzings are accepted in various forms, NeXT mail enclosures and Internet .wn.tar.Z forms are preferred but ascii text via the net, IBM and Mac Disks via USMail and (yikes) written text via the same USMail are happily accepted. We can guarantee no return of materials without SASEs, sorry. Our focus is how-to articles, especially with sample code. All articles are subject to editorial review.

We also welcome copies of new (and old) software for review from third party vendors. Again, we can not guarantee material return without SASE or guarantee publication dates, if at all, although we try to be prompt.

"Feedback from the Trenches" is open for comments/ letters of limited length from all readers. Please write and tell us what you liked and what you disliked.

Mailed subscriptions should start real soon now, with any luck. Please write for information.

Feature Submission Guidelines

If you have NeXT mail, simply drag your icon into next mail and post it off to me. Don't worry about tar'ing and compressing.

•Avoid passive voice. Please.

•Spellcheck.

•Wit is welcome, overwhelming cuteness and obscen ity are not.

•Preformatting with the following guides will aid us ENORMOUSLY:

•Paragraphs: No tabs, single return at end of paragraph, no hard-returns at ends of lines

•Title on one line, author information on one line.

•Avoid blank lines between paragraphs, but use bl ank lines to denote sections.

•Start sections on first line after section heade r.

Editorial matters to:

BuzzNUG c/o EJ Liebman

1150 Collier Rd./NW L-12

Atlanta, GA 30318.

1-404-352-5551.

There is always an answering machine, but please respect relatively normal hours. Long distance phone calls may not be returned by the impoverished student at the other end -- if you leave a message, indicate if it is acceptable to call you collect. Please send any deliveries of items that will not fit within a tiny mailbox care of the Leasing Office. To contact me directly for subscription information, corrections, requests, or just to say hi, write via internet: erica@kong.gatech.edu.

FEEDBACK FROM THE TRENCHES

I want to join! (BuzzNUG). I don't know anyone else who is interested in the NeXT as more than a novelty item. Myself, I am using it full-time for the development of software, some of which is exclusively for the NeXT, and most of which will be ported to popular PC's. My company (MindShock, Inc.) is working on two big projects right now: a set of C library routines for natural language analysis (accessible through Speaker-Listener, of course) and a big next-generation fantasy role-playing game for fast computers with lots of disk space. Thanks for running such a cool publication. *Dave S. Joerg*

Congrats on a terrific magazine!!! Looks good... I would certainly like to receive the OFFICIAL membership card you mentioned in issue two. *Bob McGill*

Hi! Just finished reading the February issue of BuzzNuG newsletter. Looks really good! I saw something about ID cards, etc., what is the deal? Are you going to have a registry, and have cards to ID NeXT users, like a secret society? That would be neat. *Scott Hess*.

Since the Subscriber Database is STILL not written, I have come up with an alternate distribution scheme -- I will be loading the ID Card and a "subscription form" into the archives by the end of April. You can pick up and customize your ID card and will be assigned a BuzzNUG Number when you submit your form. -- Erica

I also wanted to let you know that <a certain user's group> president, was "extremely jealous" of your fine work. I think the age of the virtual user's group has finally come. I haven't had a chance to play with the convex hull stuff, looks neat. one comment though: the lockfocus mechanism is a pig, and you call it from within the procs [connect, etc.] on an order 0(n^3): You could probably seriously optimize this by placing the lockfocus stuff in animate instead of in the called functions. After all, you don't draw into any other view do you? Or I am confused and babbling....;-) Take care, *Andrew Stone*

The reason I used so darned many locks is that when I left them out, system events would interrupt my algorithm processing and cause me to start drawing on my buttons, etc. - Erica

It probably doesn't hurt to say once again that Buzzings is a marvelous magazine that seems to be getting better and better. I really liked the third issue. As ever, *Bob Solovay*

As always, telling me that Buzzings is great is a terrific way to get into the Feedback Column - Erica

BEGINNERS' GUIDE TO BROWSERS

Pat McGee

Computer Graphics Group

Los Alamos Nat. Lab.

Recently, I needed a way of selecting one of a list of objects for an application I was writing. I didn't want to use radio buttons, because the list of objects could change from time to time. Using radio buttons would mean going back to IB every time it changed. I preferred just editing a text file or something equally easy.

NeXT had the perfect solution, in the Browser object. The only problem was, they didn't tell people how to use it. I remembered a discussion on comp.sys.next about this, but our news server had long since purged the messages. Being too dumb to realize that the directory news on the Purdue archives might contain old news messages, I posted a request for info.

William Shipley (wjs@fred.cs.washington.edu) responded, and pointed me in the right direction. He also sent me a necessary file which was not supplied by NeXT. This file had been generated by Edward Jung of the Deep Thought Group by using a Class Disassembler on some NeXT code. My thanks to both of these gentlemen. I am writing and distributing this document in partial payment of my debt to them.

NeXT had the perfect solution, in the Browser object. The only problem was, they didn't tell people how to use it.

The first thing you need to do is to get a Browser object. This is easy since there is one in /usr/lib/NextStep/FontPanel.nib. Open this file with IB, select one of the browser objects, and "Copy". Then close the file, open your own application, and paste the browser object into your window.

Next, get the Browser.h file -- some are floating around the archives.

Now, you need to tell IB about this class. Open the Classes icon in the IB file window. Click on Object, Responder, View, Control. Then, under the operations pulldown, select Subclass. Rename the new class to "Browser" (be careful of the capitalization). Now, from the same pulldown, select "Parse". You will get an alert panel saying "Parsed class has different actions or outlets" t. Click on "Replace". Now, IB knows about browsers.

You will see something interesting in the inspector panel when you select a browser object, then select the "Attribute" inspector. It seems that IB already knows about Browsers! (Wonder what other goodies are hidden here?) You can change several things about your browser here. Some are pretty obvious, like "Title", and the "Titles" check box. Others allow you to use multiple columns, which I haven't figured out yet, and to select between "List Mode" and "Radio Mode". I haven't needed to figure these out either.

You can also change these attributes by sending messages to the browser object. For example,

```
[browserWindow setTitle:"changed" at: 0];
```

changes the title on the first column (the only column if not using multiple columns).

To use your new browser, you need to connect it to something, probably your heavyweight object. Create an outlet in your heavyweight object, or if you prefer, in the "File's Owner" object, to connect it to. Control drag from your instance of your heavyweight object, or the "File's Owner" object to the browser object.

This allows your heavyweight class to send messages to the browser. You will need to send various messages to set up the browser and to initially fill it with items to be selected among.

You will also need to create an action in your heavyweight class or somewhere else to do something when an item is selected in the browser. This action method will be called when the user clicks on an item in the browser, and gets passed only the id of the sender, that is the browser. It needs to inquire of the browser which item was selected.

Now, you need to connect the browser object to the action method. Control drag from the browser object to whatever object implements the desired action method. Select the "target" outlet from the browser and whatever method you set up from the destination.

So much for the setup - now how do you code things so that you can actually use the browser? What do all of those methods defined in the header file do? Well, I don't know about most of them. I've found a few that allow me to do what I want. The rest, I ignore, at least for the moment.

The first thing you need to do is to put a bunch of items into the browser. I do this by sending a bunch of messages of the form:

```
[browserWindow addString:”blue” at: 0];
```

I don't know what the 0 is for; my guess is that it relates to the multiple column stuff. My test program does this in the setBrowserWindow: method, but you should probably do this in some other initialization code.

Sample code to exercise browser.h

```
/* heavy.m */
/* this test program assumes that you have used IB to
 * connect this class to
 * a Browser object and a TextField object.
 */
#import “heavy.h”
#import “Browser.h”

@implementation heavy

- setBrowserWindow:anObject
{
   browserWindow = anObject;
   [browserWindow setTitle:”changed” at: 0];
   [browserWindow addString:”blue” at: 0];
   [browserWindow addString: “green” at: 0];
   [browserWindow addString: “red” at: 0];
   [browserWindow addString: “black” at: 0];
   return self;
}

- setTextWindow: anObject
{
   textWindow = anObject;
   [textWindow setStringValue: “not clicked”];
   return self;
}

- browserClicked:sender
{
(int) col, row;
char   temp[100];


   col = [sender selectedCol];
   row = [sender selectedLine];
   sprintf (temp, “clicked on row=%d col=%d”, row,
col);

   [textWindow setStringValue: temp];

  [sender setLastCol: -1];

   return self;
}


@end
```

After you've put a list into the browser, just let the program run. When the user clicks on something in the browser, your target action method will be called with the id of the browser as its argument. You can then inquire from the browser which item was selected. Do this with the following:

```
col = [sender selectedCol];
row = [sender selectedLine];
```

If you didn't select “Multiple Columns” from the Browser Inspector, col will always be 0. Row will be 0 for the first thing in the list, and increases by 1 for each succeeding item.

You may decide at some point that you want to discard the contents of a column so you can put something else there. Do this with the setLastCol: method. If you are only using single columns, do this:

```
[browserWindow setLastCol: -1];
```

Well, that about exhausts my current knowledge of browsers (and my time available for writing this). I hope that you find this useful. I also hope that you will experiment with more of the methods to discover how to do more things with what we've got. Send me what you discover, and I'll write it up for the next version of this.

Comments from other people:

Andrew Stone (stone@hydra.umn.edu) says

List Mode is a list where multiple selections are permissible.

Radio Mode, naturally, is mutually exclusive. Moreover, if you do not send your browser the allowEmptySel:YES method, one cell will always be selected.

Since I have hacked browsers by subClassing matrix, I recognize all of thewindow dressing. I deal with matrices often because it allows generalizationof code, which you can switch on the “selectedTag”. Note that with matrices, you always feed arguments row, col; so as you guessed, the “0” refers to col.0 or the only column in a single list. Old fortran hackers just have to learn that arrays start with 0 not 1!

The rationale behind using selectedTag as opposed to selectedCol or selectedRow in querying a matrix for who was clicked on is that it frees you from worry if you switch a matrix from being horizontally oriented to vertically. There is a checkBox in the Attributes window of Matrix that allows he tag to match to the cell position.

Browser.h

```
/*
* $RCSfile: Browser.h,v $
* $Revision: 1.1 $
* $Author: ed $
* $Locker:  $
* $Date: 89/11/17 16:47:50 $
 * $Log:Browser.h,v $
 * Revision 1.1  89/11/17  16:47:50  ed
 * Initial revision
 * *
 * Documentation: -NONE-
 * Mr Yuck (Bugs): romper
 * Resolver: /
 * Configuration cache: /prefect
 * Project context: External/NONE
 * Law file: External/NONE
 * Tester: External/NONE
 * Probe: External/NONE
 * Generator: External/RunTimeClassDisassembler(1.3)
 *
* Copyright: 1989 by Edward Jung of the Deep Thought Group, L.P.
* Copyright: Permission is hereby granted to freely utilize this
* Copyright: code for any legal purpose provided that copyrights
* Copyright: and notices are preserved.
*/



#import <appkit/Control.h>

/*
 * SYMBOLS AND MACROS INTERFACE
 */
 /*  *  This "false" struct is an unknown, so we set up some
  *  random definitions, conservatively padded, based in
  *  part from disassembling the library.
  */

struct _BrowserFalseStruct {
     unsigned int iv21_1:2;
     unsigned int iv21_2:3;
     unsigned int iv21_3:4;
     unsigned int iv21_4:3;
     unsigned int iv21_5:2;
};
```

```
/*
 * MAIN CLASS INTERFACE
 */

@interface Browser:Control
{
  idprototype; /* Cell prototype, default is SelectionCell */
  id target;
  SELaction;
  SELdoubleAction;
  floatlineHeight;
  floatinterCellHeight;
  floatcolWidth;
  id leftButton;
  id rightButton;
  id contentView;
  intselectedCol;
  intselectedLine;
  intlastCol;
  intfirstVisibleCol;
  intnumVisibleLines;
  intnumVisibleCols;
  inthitPart;
  intmode;
  id root;/* for consumer */
  id current;/* for consumer */
  intmouseDownFlags;
  struct _BrowserFalseStruct
     bFlags;
}

/* CLASS METHODS */
+ getSize:(const NXSize *)aSize
     forNumCols:(int)c numLines:(int)l
     colWidth:(float)w titles:(BOOL)t multiColumn:(BOOL)m;

+ newFrame:(const NXRect *)aRect
     colWidth:(float)w titles:(BOOL)t multiColumn:(BOOL)m;

+ newFrame:(const NXRect *)frameRect;



/* INSTANCE METHODS */

- free;

- cellClass;
- setCellClass:anObject;// sets all matrices cellClass, see Matrix docs
- prototype;
- setPrototype:anObject;// sets all matrices prototype, see Matrix docs

- (int) selectedCol;
- (int) selectedLine;

- target;
- setTarget:anObject;

- (SEL) action;
- setAction:(SEL) sel;
- (SEL) doubleAction;
- setDoubleAction:(SEL) sel;

- setMultiColumn:(BOOL) yesOrNo;
- (float) columnWidth;
- setColumnWidth:(float) width;
- (int) columnOf:anObject;

- setTitled:(BOOL) yesOrNo;
- setTitle:(const char *)aString at:(int) offset;// browser well title
- (char *) titleAt:(int) offset;

- matrixAt:(int) offset;

- setMode:(int) modeNum;// mode of matrices, see Matrix docs

- addString:(const char *)aString at:(unsigned int) offset;// append cell to matrix at offset initialized with aString

- sizeTo:(float)w :(float) h;
- sizeToFit;

- growTo:(unsigned int) offset;
- setLastCol:(int) offset;// toss matrices above offset
- (int) lastCol;

- showNthCol:(int) offset;
- showLastCol;

- selectCol:(int)x line:(int) y;
- displayColumn:(unsigned int) offset;
- displayFrom:(int)x_low to:(int) x_high;
- redisplayCol:(int)x line:(int) y;

- renameCol:(int)x line:(int)y to:(const char *) aString;// rename cell at line y of matrix at x to aString
- shiftColsBy:(int) offsry;
- clickMatrix:sender;// received from matrix when click occurs
- doubleClickMatrix:sender;// received from matrix when doubleClick occurs

- scrollUp:anObject;
- scrollDown:anObject;
- scrollLeft:anObject;
- scrollRight:anObject;

- (int) mouseDownFlags;
- (int) hitPart;

- drawSelf:(NXRect *)r :(int) c;
- (char) acceptsFirstMouse;

- write:(NXTypedStream *) stream;
- read:(NXTypedStream *) stream;

- _buildColumn;
- _buildSubviews;
- _grow;
- _tile;// called by drawSelf::
- _tileColumn:anObject;
- _enableLeftButton:(BOOL)yesOrNo;
- _enableRightButton:(BOOL)yesOrNo;
- _updateHScrollButtons:(BOOL)yesOrNo;
- (int) _lastVisibleCol;
- _incrLastColTo:(int)col andDisplay:(BOOL)yesOrNo;
- _decrLastColTo:(int)col andDisplay:(BOOL)yesOrNo;

@end
```

Our Thoreau Project

Austin Meredith, meredith@jeckle.cs.umn.edu

I am at a NeXT-owned workstation in the basement of the Computer Sciences building at the University of Minnesota. By special arrangement, since I have no connection with the University or with NeXT, I am being allowed the use of this machine in order to put the complete works of Henry David Thoreau on your magneto-optical disk. Our project, which we call “The Stack of the Artist of Kouroo” because of a fable told in WALDEN, is preparing this hypertext textbase by use of donated software from Frame Technologies. We intend this disk to constitute a quarterly scholarly journal circulated among Thoreau scholars nationwide.

The disk will have on it not merely Thoreau's published works, including his 18-volume journal and his letters, but also all the scriptures to which he habitually referred, such as the Bible, the Bhagavad Gita, the Qur'an, the Four Books of China, etc. All connected by hypertext links so you can pop back and forth. The disk will also contain some seventy volumes of scholarship written in the past ten years, and quite a number of scholarly papers. The disk will also have bird calls and illustrations, and plant trinomens and illustrations.

We are sponsoring a music contest. The contest is not for a composition but for a program. The composer John Cage is allowing us to use his name in connection with this programming contest, which is to be called "Walden: The Program." We are asking NeXT to donate a machine, with accessories, as the prize. The winning program will provide an aural environment for the Thoreau scholar as he/she uses this disk. The criterion for the program is that it must be as much as possible in the spirit of Henry David Thoreau and John Cage.

Austin Meredith, "Stack of the Artist of Kouroo" Project,
2860 Kenwood Isles Drive, Minneapolis MN 55408-1981

Austin sent me a more detailed write-up which was corrupted in the mail. I shall try to include his full write-up next issue - EJL.

Accessing Unix Tools from the Workspace Manager:

Uncompress

by Jiro Nakamura

Abstract

The NeXT's graphical user-interface has gained much acclaim as a user-friendly front-end to UNIX. An unfortunate side-effect, however is that advanced-users cannot easily access standard UNIX tools such as uncompress (1) from the Workspace Manager. This article shows how to easily create a pseudo-application that can be launched from the WM and access the underlying UNIX facilities.

Unix from the Workspace Manager

The NeXT workstation has achieved great fame with its sophisticated graphical user-interface that overlies the UNIX operating system. The user-interface hides many details of UNIX that end-users are not interested in. Unfortunately, because it provides a layer of abstraction, it becomes more difficult to access standard UNIX tools and we have to open a Terminal or Shell window to directly access them. I would like to show in this article an easy way to make a pseudo-application that can access UNIX facilities from the Workspace Manager (WM).

I will use as an example and foundation the uncompress (1) utility. The uncompress utility takes a file that has been compressed using compress (1) and expands it back into its original form. What we would like to be able to do is to double-click on compressed files with a suffix of *.Z,* launch an uncompress program, and end up with the original uncompressed documents.

We actually have two problems in implementing this:

1) We have to get the WM to recognize the ™.Zsuff ix and call a uncompress-ing program.

2) The uncompress-ing program has to recognize that it's being called by the WM, figure out what the WM wants it to do (the files it is supposed to uncompress) and uncompress them.

Ideally, NeXT should have provided us with an easy way to associate pathname suffixes to applications. Even the Macintoy allows us to do this by changing parameters in ResEdit. But the NeXT is still young, so someone else has had to write a program that will let us do this. This program is Unknown, written by Christopher Lane (lane@sumex-aim.stanford.edu). It can be found in the cs.orst.edu or j.cc.purdue.edu NeXT ftp archives. If you do not have ftp access, please e-mail me at jiro@heights.cit.cornell.edu and I will see if I can get you a copy.

What we would like to be able to do is to double-click on compressed files with a suffix of.Z, launch an uncompress program, and end up with the original uncompressed documents.

Unknown associates pathname suffixes with their appropriate applications and icons. It already has an entry and icon for files with*.Z*suffixes. However, the application field for *.Z*is unspecified (it points to Unknown which will simply return without doing anything). Let's change it so that it will launch a program called Uncompress (notice that capital letter). To do this, we have to change the Unknown.iconinfo file:

```
            UnknownUnknown
      ZUncompress<----=== Change this line
                 ......
```

Now when we double-click on a file such as *foo.Z,* the Workspace Manager will try to launch an application called Uncompress (big U) telling it to uncompress *foo.Z* You may be asking why we don't just tell it to launch the UNIX utility uncompress (small u). If things were that simple, I wouldn't have to write an article.:-)

Unfortunately, the way the WM specifies its arguments and the way UNIX tools want their arguments do not match

The Workspace Manager opens double-clicked files by launching their application with special arguments that 1) tell the application it is being launched from Mach and 2) tell it to open certain files. Unfortunately, the way the WM specifies its arguments and the way UNIX tools want their arguments do not match. If we select a .Z file - say thisFile.Z - and then Open or double-click it, the WM will launch Uncompress with the following arguments:

Argument 0:/LocalApps/Uncompress<- The program's own name

Argument 1:-NXOpen<- The WM wants the following

Argument 2: thisFile.Z<- file opened.

Argument 3:-MachLaunch<- Tells us Mach is launching it

We have to translate these arguments to something uncompress can understand. This is pretty easy to do. First, Uncompress (big U) figures out what its arguments are and then calls uncompress (small u) with the suitable changes. In other words, if the WM is calling Uncompress with:

```
/LocalApps/Uncompress -NXOpen thisFile.Z -MachLaunch
```

then Uncompress (big U) will call uncompress (small u) with:

```
/usr/ucb/uncompress thisFile.Z
```

The source code for Uncompress is at the end of this article. This program can easily be modified for other utilities such as tar (1), uudecode (1C), etc. Basically all you have to do is change the name of the program and the MACHPROGRAM definition. To aid in debugging, you can define the DEBUG definition. This will print some debugging aids onto the standard error output which for WM-launched programs is the Console window.

You might ask what happens if we select more than one file to open. Well it turns out that the WM will repeated run Uncompress for each of the files, rather than try to open them all at once. This is why Uncompress is only written to handle one NXOpen parameter.

Ideally, the WM itself should be able to call the appropriate UNIX utilities itself. Someone can also try to write a single program that handles each and every suffix and link that to Unknown. (Maybe Chris Lane will write extend Unknown to do it:-) Alternately, someone can extend the Uncompress utility so that it is a full-blown user-friendly GUI application. These are all left as exercises to my readers (standard article writer's cop-out). Any questions about this program or about NeXT's, the WM, or UNIX can be directed can be sent to me at jiro@-heights.cit.cornell.edu. I'll be more than willing to help anyone if they have any problems.

The modified Unknown.iconinfo file for Unknown

```
UnknownUnknown
Z    Uncompress<-- This line changed
a    Unknown
# afmEdit
# asmEdit
# awkEdit
c    Edit
cl   Edit
# cshrcEdit
doc  Edit
# dockUnknown
el   Edit
elc  Unknown
fasl Unknown
h    Edit
# iconheaderEdit
# iconlistUnknown
# listUnknown
log  Edit
# loginEdit
m    Edit
o    Unknown
psw  Edit
pswm Edit
# rc Edit
# rtfEdit
# shlibUnknown
# tarUnknown
tex  Edit
```

The source code for the Uncompress application

The source-code to Uncompress and Unknown (along with modifications) can be obtained by e-mail to the author at jiro@heights.cit.cornell.edu.

```
//
// Program Name:Uncompress
// Files: Uncompress.c
// Created:March 1, 1990
// Last Modified:March 17, 1990
// Author:Jiro Nakamura
//
// This program is Copyright (c) 1990 by Jiro Nakamura. All
rights
// reserved. Use for non-commercial purposes granted.
//
// This is a simple program that shows how you can turn a pure
// Unix(tm) command into one that can be called by the Workspace
// Manager (tm) by double-clicking on its files.
//
// It must be used in conjunction with "Unknown." This program's
// function is to parse the arguments passed by Workspace manager
// when someone double-clicks on one of its files into something
// that the original Unix program can understand.
//
// See the original article in BuzzNug #5 for details.
//
// Compile me with:
//   % cc -O -g Uncompress.c -o Uncompress -lNeXT_s -lsys_s
//   % install -s -m /LocalApps Uncompress
//

#include <stdio.h>// for fprintf, stderr, et al.
#include <libc.h>// for access, system, wait, et al.
#include <strings.h>// for strcmp, strcpy

#defineOPENFILE"-NXOpen"
#defineMACHLAUNCH"-MachLaunch"

// Name of program to call eventually
#defineMACHPROGRAM"/usr/ucb/uncompress"

// Access(2) mode needed for the file we are to open
// Uncompress only needs to be able to read the input file
#define ACCESSMODER_OK

void main(int argc, char *argv[])
{
int  a;   // A simple loop variable
static char progName[255], // This program's name as called by
the WM
    openFile[255],// File to be opened as called by the WM
    launchMode[255],// Launch mode (not utilized)
    systemCall[300];// Buffer space for system(3) call

    #ifdef DEBUG
        fprintf(stderr,
        "Launched with %d arguments.\n",argc);
        for( a= 0; a< argc; a++)
        fprintf(stderr, "Argument[%d] is <%s>\n",
        a,argv[a]);

    #endif



    // The program name is argument 0 by default
    strcpy( progName, argv[0]);



    // The rest of the arguments we will have to parse using
    // strcmp since there aren't that many arguments.
    for( a= 1; a< argc; a++)
        {
        if( strcmp( argv[a], OPENFILE) == 0)
        strcpy(openFile, argv[++a]);

        if( strcmp( argv[a], MACHLAUNCH) == 0)
        strcpy(launchMode, argv[a]);
        }



    #ifdef DEBUG
        fprintf(stderr,
        "%s called with NXOpen <%s> and MachLaunch <%s>.\n",
        progName, openFile, launchMode);
    #endif

    // Make sure we have a valid openFile argument
    if( access( openFile, ACCESSMODE) == -1)
        {
        fprintf(stderr,
        "Uncompress: File <%s> not found error.\n",
        openFile);
        exit(1);
        }

    // Make the system call using the name of the mach program
    // we want to call and the file we want to open as the
    // argument.
    sprintf(systemCall,"%s %s",MACHPROGRAM, openFile);
    #ifdef DEBUG
        fprintf(stderr,"Making system call: <%s>",system-
Call);
```

```
    #endif
    system( systemCall);// Make the system call
    wait(0);// Wait for it to complete

    exit(0);// and exit
}
```

```
  Finis.
```

Jiro Nakamura

Independent NeXT Developer

jiro@heights.cit.cornell.edu

FreeText IR Philosophy

Mark Zimmermann

Abstract

This essay describes in general-audiences fashion my free text information, retrieval philosophy, and gives some technical background on the design decisions that I made while writing my various indexer/browser programs; it also outlines some of the extensions and enhancements that are necessary to handle 100 MB/day of incoming text... various folks have asked me to write this sort of thing, and now that I'm done I'll try to stop talking and get back to writing some code to actually *do* it! ^z

NOTES ON FREE TEXT INFORMATION RETRIEVAL

I've been thinking recently about some fundamental issues in big free-text databases... specifically, whether it will be feasible to handle 100 MB/day or so of unstructured text (from wire services, USENET, database downloads, scanned-in magazines, etc.) and do useful research on it, the way that my FREE TEXT indexer/browser programs on the Mac let you work with single files of tens of megabytes.

I think that the answer is “yes”, 100 MB/day is manageable (for how long? -- a month, at least?). But it will require some (straightforward, I hope) extensions to my current indexer/browser systems. This informal memorandum is a summary of my thoughts. It includes comments on required features of the software, the technical design decisions behind my current information retrieval (IR) codes, key problems that must be solved to handle bigger free-text databases, and some suggestions for how to implement solutions to those problems.

My goal is to provide a record of where my general free text IR thoughts stand today, to help people who want to work with me on enhanced text handling tools for researchers. I'd appreciate any comments or suggestions or corrections that anybody wants to make!

REQUIREMENTS

I want real-time high-bandwidth responses to simple, common user queries of the free-text database. By “real-time” I mean that single-term retrieval requests should be answered in less than a second. (Complex Boolean proximity searches can take a bit longer to set up and execute, but even there, speed is important.) By “high-bandwidth” I mean that the human has to get information back from the computer in a useful form which can be evaluated quickly -- it's not good enough to return a list of documents or files, each of which has to be paged through in order to find the few relevant items. By “free- text” I mean that the input stream can't be assumed to include any regular structure beyond words separated by delimiters (spaces, punctuation, etc.).

By “high-bandwidth” I mean that the human has to get information back from the computer in a useful form which can be evaluated quickly -- it's not good enough to return a list of documents or files

I want tools for the earliest stages of research, when a person needs to be able to browse and free-associate without even knowing what are the right questions to ask. Users have to be able to work with the database without intimate knowledge of the details of what's in it (since nobody will have time to read much of a 100 MB/day flood of information). A big database in a particular area has to be a “corporate memory” for groups of scholars who are working on related topics. A new person has to be able to come in and do research without extensive training, and an experienced user has to find the system transparent, so that it becomes an extension of her memory, not a barrier to getting at the data and thinking with it. I see free-text IR tools as coexisting with other types of tools, such as structured databases which are appropriate for answering well-formulated queries at later phases of the research effort.

I know of four fundamental end-user operations that a good real-time high-bandwidth free-text IR system must support:

- viewing lists of words that occur in a database;
- selecting subsets of a big database to work within;
- browsing through candidate items efficiently; and
- reading and taking notes on retrieved information.

I'll discuss each of these four tasks below.

WORD LISTS

Viewing lists of words that occur in a database is exceedingly useful, for many reasons. An alphabetized word list makes it easy to spot many misspellings and variant suffixes for search terms. Word lists are surprisingly useful in working with mixed-language databases, particularly in spotting cognates and words that begin similarly to related native-language terms. The ability to view a word list, along with numerical information about occurrence rates in the whole database and in subsets, is valuable in formulating better queries that maximize good data retrieval. Perhaps most importantly, just being able to scroll around in a list of terms is a superb aide-memoir in thinking of words worth looking for in a database. Visible word lists are much better than blind searching, even when wild cards are permitted. It's precisely analogous to the “recognize vs. remember” dichotomy between menu-driven and command-line user interfaces.

Word lists are easy to generate, provided that one has built a complete inverted index to the database being browsed. For example:

1	8805251042
1	9511
6774	A
1	AA01693
1	AA03935
2	AA13961
1	AA29900
34	ABLE
291	ABOUT
29	ABOVE

Each word is preceded by the number of times it occurs in the database as a whole. Note the presence of numbers and document identifiers as well as ordinary words.

A word list doesn't need to take up much storage space as files get big, since the number of distinct words grows much slower than the file size. There are various rules of thumb and theoretical formulae, such as Zipf's Law, for estimating how big the word lists will get. In my experience, unique dates, other numbers, garbles, typos, etc. tend to keep the word lists growing more than expected, but not at an unmanageable rate.

My free text IR programs use an extremely simple file structure, which makes generating word lists nearly trivial. In the “keys” index file, I store as an array the words found in the database along with count information. Thus, a single disk access can fill a small window with an alphabetized word list along with each word's frequency of occurrence in the database as a whole. (See the discussion under BROWSING below for information on how counts are displayed when working in a subset of the database.)

In my programs, I currently truncate words rather arbitrarily after 28 characters; I keep 4 bytes of cumulative count as a long integer; I default to a standard character mapping from lower-case to upper-case; I default to break words apart at punctuation marks; and I default to alphabetizing according to the ASCII character set order. These choices work well for English-language texts in my experience, and they're easily changed if necessary by recompiling the indexing and browsing programs. (In a future version of my programs, I intend to allow at least the character mapping and alphabetization order to be defined separately for each database; see the EXTENSIONS AND ENHANCEMENTS section below.)

I index every word in the database, even common words such as “a” and “the”. Many free-text IR systems call such terms “stop words” and omit them. The cost, however, is too great -- it becomes impossible to search for common terms in proximity to each other or as components of names, part numbers, dates, and so forth

These design choices mean that each distinct word in a database occupies 32 bytes in the “keys” index file. It is possible to save a little space by using a more complex data structure, but it would be at the cost of speed and simplicity in the indexing and browsing programs. In fact, as databases get larger the storage used by the “keys” file becomes increasingly insignificant compared to the database and the other part of the index (the “pointers” file, discussed under SUBSETS and elsewhere below). The four-byte cumulative count element of the data structure will limit the indexed databases to 2^32 total words. That's about 4 billion words, or roughly 25 GB total file size, given a typical word length of 5 letters plus 1 delimiter. Modifications to this part of the index data structure will thus be necessary for very large free-text databases (see EXTENSIONS AND ENHANCEMENTS).

I index every word in the database, even common words such as “a” and “the”. Many free-text IR systems call such terms “stop words” and omit them. The result is a saving of perhaps 20% - 40% in final index file size, depending on how long the stop word list is. The cost, however, is too great -- it becomes impossible to search for common terms in proximity to each other or as components of names, part numbers, dates, and so forth. A user can never find “To be, or not to be...” in Shakespeare, since every word in that phrase is a stop word! It also becomes impossible to look for “A. I. Smith” in a long name list, or to find many short Chinese or Japanese words at all.

I am inalterably against the use of stop words; I also oppose “stemming” or “truncation” in free-text indexing. Many conventional systems attempt to remove suffixes so that they can index various forms of a word under a single entry; for instance, “compute”, “computer”, “computers”, “computational”, and “computing” might all be merged into a single “comput” entry. This saves a small amount of space, but at the expense of simplicity and control for the user. (At times, I may need to retrieve “computers” but not “computer”.) Many stemming systems also get confused easily and garble words (particularly proper nouns) -- occasionally with disastrous results when a truncated term lands on the stop word list and is left out of the index entirely! Foreign terms and oriental names are particularly vulnerable to butchery by such systems.

SUBSETS

Researchers need to be able to select subsets of a big database to work within. As information collections get very large, there will still be unique or nearly-unique terms that people will want to retrieve quickly -- the last occurrence of a particular name, a bizarre word or document identifier that only occurs once, etc. But increasingly often, there will be too many instances of an important word for convenient browsing (but see the BROWSING section below for methods to delay that point). It thus becomes necessary to filter the input stream in a fast and flexible way, so that a human- usable volume of information is left to study.

Proximity search is a common free-text IR approach to data filtering. The usual proximity criteria, however, tend to be narrow, inflexible, and (in my experience) extremely hazardous to retrieval when applied to real-world databases. A conventional form of proximity search is to enter a boolean query such as “FREE(w2)TEXT AND (INFORMATION OR RETRIEV*)” which translates into “give me any document with FREE occurring within two words before TEXT, which also contains INFORMATION or any word beginning with RETRIEV”. Of course, this example can't work at all if the database isn't already divided up into “document” units. Even if the query appears to work, the user will fail to see an unknowable number of near-miss items which contain some but not all of the specified words, or which use alternative spellings (including typographical errors).

The user interface for conventional boolean proximity search is simply terrible. It doesn't provide enough feedback to let the user avoid subtle (or obvious) mistakes, it requires too much human memory and creative energy in query formulation, and it gets in the way of seeing the data. Often, an apparently-innocuous term in a parenthesized boolean search statement actually distorts the entire meaning of the query and eliminates all the useful information that was to be retrieved. Expert searchers can overcome some of the barriers that a command-line proximity search interface imposes, but it's never easy. In my browser programs, I take a different approach. I let a user define a neighborhood of interest in a free-text collection based on loose proximity to words chosen from the full database word list (see WORD LISTS above). For reasons of speed, simplicity, and convenience, the subset proximity criterion is in bytes (characters of text), with a default resolution of 32 bytes. Thus, a searcher could ask a fuzzy equivalent of the above sample boolean query by starting out with an empty subset and then adding the neighborhood of the word “FREE”. This is actually done by clicking on the word “FREE” as it is displayed in an Index View window -- a perfect opportunity to see nearby alternative search terms! The default neighborhood is “within-a-few-words”, roughly plus or minus 50 bytes. At this point, an excerpt from the database subset's word list might be displayed as:

```
5/34     ABLE
0/291    ABOUT
0/29     ABOVE
0/5      ABSTRACTS
9/41     ACCESS
```

(Here, the term “ABLE” occurs 34 times in the database as a whole, of which 5 instances are in the proximity neighborhood of “FREE”.) Next, after making a similar selection of the neighborhood around the word “TEXT”, the user can narrow down the subset of interest to the intersection of the “FREE” and “TEXT” subsets. (This equivalent of the boolean AND operation is again done with a mouse click.) Then, broadening the neighborhood setting to 500 bytes or so, the user can with a few more clicks define another subset of words very loosely near INFORMATION or RETRIEVAL or RETRIEVALS or RETRIEVING, etc. Intersecting that subset with the FREE TEXT subset gives the final subset of interest for interactive browsing.

The user interface for conventional boolean proximity search is simply terrible. It doesn't provide enough feedback to let the user avoid subtle (or obvious) mistakes

My experience, and that of many users, is that my kind of fuzzy proximity search is far better than classical boolean search for finding the information that one really wants. It's faster, safer, and far more aesthetic to use as well. The mental model that the user has is simple and powerful: neighborhoods of influence that radiate out from interesting words, and the possible combinations of those neighborhoods. The proximity search interface, after a few minutes of use, becomes transparent and doesn't get in the way of thinking with the data. My current indexer/browser programs assume that the free-text database is a single file. That assumption is straightforward to lift, and I plan to do so soon (see EXTENSIONS AND ENHANCEMENTS). But it is very convenient, for many purposes, to treat the entire database as a linear entity, and refer to byte offsets from the beginning of the database. Databases which have additional structure (sections, chapters, pages, etc.) can easily be handled within such a framework.

My current implementation of fuzzy proximity search uses vectors, linear arrays, to represent the database subsets. Each bit of a subset vector stands for a 32-byte (default value) section of free text. A bit is cleared when its section is not included in a given neighborhood, and a bit is set when its section is part of the subset. This approach of representing 32 bytes with 1 bit results in a compression factor of 256, so a gigabyte database can have a proximity neighborhood defined using a 4 MB vector. Subset operations are trivially simple with this data structure. (See the EXTENSIONS AND ENHANCEMENTS section for a discussion of what changes may be needed to handle multi-gigabyte databases.)

In order to set or clear bits in defining subsets, it is obviously necessary to have pointers to the locations of every word in the database. (It is necessary to have such pointers in any event to retrieve information quickly, of course.) The second index file that my programs use, the “pointers” file, is precisely that. In the current implementation, it simply consists of 32-bit integers giving the offset from the beginning of the database to each indexed word. Entries in the pointer file map directly to proximity subset vectors, making it easy to set and clear bits as needed to define regions of interest in the database. (See the EXTENSIONS AND ENHANCEMENTS section for more on how this model can grow to handle larger data collections.)

Each instance of a word thus requires 4 bytes of space in the “pointers” file. Since an average word is 5 letters long, plus one or more terminal spaces or other delimiters, the average “pointers” file is less than 80% of the size of the original database text file. It would certainly be possible to save some disk space by compressing the “pointers” file, but I doubt that more than half of the original file size could be recovered. There would also be a price to pay, in terms of slower retrieval, more severe database size limitations, and increasingly complex (and less reliable) software. I thus chose to keep the “pointers” file structure as simple as possible.

Many conventional free-text IR systems, I suspect, generate in-memory lists in order to do proximity search operations. They tend to fail when looking for common words in conjunction with each other, probably when their data structures overflow. (I don't know for sure, as their algorithms are proprietary and unpublished; I deduce their design by observing their common fail cases.) I much prefer my subset vector approach. It is simpler, it has guaranteed good run time behavior, and it never breaks down when working with dense database subsets.

Conventional free-text IR systems also (I suspect) keep lists of sentence and paragraph and document delimiters, in order to handle some types of proximity search requests. I have never found it necessary (or even desirable) to pay much attention to rigidly-defined boundaries such as sentences or paragraphs in actually performing proximity searches. The fuzzier proximity search that I have implemented is, in actual experience, much safer and more likely to find the key data items that are being sought. A user of my system will never lose a good piece of information due to an unfortunate sentence division or paragraphia choice by an author. As a secondary benefit, there is no need to pre-process items flowing into the database in order to insert paragraph, section, or document boundary markers before indexing.

BROWSING

The key to effective use of a free-text database is the ability to browse rapidly through many items in order to find the few that are of greatest interest. A good free-text IR system has to provide lists of indexed words (see WORD LISTS above) and it has to fetch the full text of documents rapidly upon demand (see READING below). But a great free-text IR system must also provide facilities to bridge the gap between word lists and full text retrieval. I do this in my programs by applying an old concept, the “key word in context” display, in a new way which I call a “context view”.

A “key word in context” listing, sometimes called a KWIC, is a set of lines extracted from a database which show the instances of a word's occurrence, each with some context on either side. The “key word” is lined up in a column down the middle of the page. Static, printed KWIC indices have been around for many years, and are better than no index at all, but they don't provide fast, transparent access to data. A user still has to track down the actual reference in order to get back to the full text of the item, and she still has no way to look at a subset of the original database.

My context views are key-word-in-context displays which are generated on the fly, at run time, in response to user requests. They look like a common KWIC -- for example, a context view of six occurrences of “Macintosh” in a small file might reveal:

```
rogram, without the Macintosh user interface features.  Ho
her features of the Macintosh Browser:   - easy access to
megabytes/hour on a Macintosh Plus, and over 60 megabytes/
[75066,2044].  The Macintosh Browser program is available
O-MAC> and the MAUG Macintosh Users' Forum on CompuServe.
send me a formatted Macintosh disk and a self-addressed st
```

Tabs, carriage returns, line feeds, etc. are all suppressed in a context view, to allow the user to see the maximum amount of information on each line.

A person can scroll around in a dynamic context view in a window on the computer's screen, and quickly eliminate many of the uninteresting usages of a word. When an item looks promising, a mouse click immediately calls up the actual text from the original document for reading and possible further use. (See READING below for further comments.) When working in a proximity subset of the whole database, the context view only shows the lines surrounding word instances in the subset of interest.

My experience indicates that a human can comfortably browse through hundreds of lines in a context view in order to locate a few good ones to retrieve. It's at least an order of magnitude easier (and more pleasant!) than wading through full text documents as they are conventionally retrieved by IR systems.

The current implementation of context views in my FREE TEXT indexer/browser programs is straightforward. For each line of context to be displayed, a single disk access is needed to fetch the characters from the appropriate point in the database file. (Since the entries in the “pointers” file are already alphabetized and sorted together, one access to that file provides many windows worth of context view pointers; the cost of that access is thus negligible compared to the fetching that has to be done from the text database file.) With some operating-system caching of disk sectors as they are used, scrolling of a context view window is reasonably smooth.

In order to guarantee a fast response to user requests when working in a subset of the entire database, I currently apply a couple of simple tricks. In generating the word list, for example, I give exact counts of the number of words in a subset only if that word occurs less than 100 times (default value). Thus, for example, an entry in a word list might be:

~11%   A

I take a sample of the occurrences of “A” in the database and report back a percentage estimate of the fraction in the subset. (To be precise, I default to counting the last 100 instances in the database.) In generating a context view, when more than 100 (default value) instances of a word are skipped over because they fall outside the subset of interest, I display an indication of that as a line of dots in the context window; for example:

```
.........................................................
.........................................................
megabytes/hour on a Macintosh Plus, and over 60 megabytes/
.........................................................
```

If I didn't use this tactic, a user might try to retrieve in an empty subset and have to wait for the entire database to be scanned before a “no valid instances” message could come back. I find the quicker response of my approach superior, and I also find it useful to see an indication (by lines of dots) when I'm working in a “desert”, an almost-empty subset.

READING

The bottom line for a free-text IR system is to get a desired document (or excerpt) in front of the user, so that it can be read and worked with. It goes without saying that a good IR system must make it easy to scroll around in a database, copy out selections, and paste them into working papers, other programs, or anyplace else that the user desires. Handling this interprocess communication is a job for the operating system, but the IR program must be at least open enough to make it very easy to export data to other programs. If the original document in the database has some structure (and thus isn't purely “free text”) it may be valuable to reflect that structure in the retrieved data. Chapter or section titles can be displayed, for instance, or if there are graphics or font style changes they can be shown properly. The ultimate goal is to have the original document (or a high-resolution image of it) appear in front of the user, open to the right page, with a highlight to draw attention to the key part. That's unfortunately not easy to do today!

My current systems go only partway toward meeting this goal. I display text and highlight the index term that was the original basis for the retrieval. But I only extract a small chunk (a few thousand characters, by default) of the text document and let the user scroll around in that. It's hard to do more and still maintain the speed and simplicity of the software. I don't handle non-textual elements of the database at all, and I only display a single font.

If the original document in the database has some structure (and thus isn't purely “free text”) it may be valuable to reflect that structure in the retrieved data.

Beyond mere retrieval, an ideal free-text IR system would make it easy to annotate retrieved information, add valuable comments and leave a trail behind so that other explorers (or the same explorer, at a later date) could profit from previous experience. Hypertext links between documents should be easy to create (as one type of annotation, perhaps) and should be easy to see and follow. They would then provide another set of paths through the data space. All of these topics are under active research today, and perhaps within a few years robust, transportable systems to do such jobs will be available. In the meantime, however, my top priority is to do a better job at the fundamental task -- fast, easy, open, good real-time high-bandwidth free-text information retrieval on big databases.

EXTENSIONS AND ENHANCEMENTS

As mentioned repeatedly above, some significant yet straightforward extensions to my current free text IR systems are necessary in order to properly handle 100 MB/day of data. Here, I will briefly sketch out how I plan to attack the key problems during the coming months. I assume that ample physical storage is available to hold the influx of information on-line, in a form which allows access to any item in a fraction of a second.

My systems have to be modified handle multiple text files as a single database. I propose to do this by adding a third index file to the “keys” and “pointers” files -- a “filelist” index file which will simply contain a list of the database document files along with their lengths. The structure of the “keys” and “pointers” files will remain unchanged (which should maximize compatibility with earlier index programs and minimize the number of new bugs introduced in this step). The index building programs will treat each of the documents in the “filelist” file as part of a single big document for indexing purposes, and the index browsing programs will consult the “filelist” in order to know where to go to retrieve lines of context or chunks of full text for display.

A drawback of this multifile “virtual merge” approach is that it will at times be necessary to open and close files during browsing operations. I have not yet run tests, and do not know what penalties the operating system will impose (one hopes only a few milliseconds?) every time a file is opened and closed. With the use of modern operating system RAM caches, I hope that speed will not be a problem. During typical browsing operations, I believe that most database references will be predictable and localized, so caching should help average performance a lot.

Another extension which I plan to implement is to add facilities to rapidly merge separate index files upon user demand. Merging already-sorted index files is a very fast operation which should be limited only by disk I/O rates. It will then be possible to keep indices for each day's (or hour's, or whatever) collection of data separate until the time at which a user wants to browse a chosen set of files. The delay to merge the selected separate indices will be about equal to the time required for a single sequential scan through the chosen database. After that start-up delay, searches will progress at the normal full speed (subsecond response time for simple queries). Many data collections which are commonly referred to as a unit can have their merged index files kept on-line to avoid any search delays.

My current browser programs can already run on one computer while searching files resident elsewhere on a network. The generic UNIX version of my browser can also already be used as a “server” process and run on one host, sending back only the (relatively) small amounts of retrieved information that the user wants to see on a local machine. I plan to rewrite some parts of my browser programs to make their use as servers simpler and more efficient; this rewrite will take place along with the other revisions to introduce new features.

Index building itself should not be a computationally infeasible operation at a 100 MB/day data rate. My indexer programs already run at 15-20 MB/hour on a 16 MHz 68030 (Mac IIcx), [It flies on the NeXT -- EJL] and I have had reports of 60 MB/hour or better performance on faster machines with more memory and higher performance disk drives. I also believe that there is room for a 20% - 50% speed improvement in my indexing algorithms, by applying some of the standard quicksort enhancements discussed in many textbooks. For storing the index files, a simple and obvious modification that I plan to make is to give the user complete freedom to put databases and index files in any directory, on any volume (online storage device) that is desired. This will allow archival databases to reside on high-density optical read-only media, while indices can be built on faster magnetic media and can be moved to the archive when appropriate.

To handle databases larger than about 4 GB (2^32) will require modifications to my programs, but (assuming that disk space is available) not major trauma. If the pointers and counters in the index data structures are redeclared to be 6 bytes instead of 4 bytes, for example, it should be possible to handle up to 256 TB of text in theory. Index file overhead will go up to about 120% instead of the current 80% of the database text size. At this point, some simple compression routines might be worth exploring to increase storage efficiency, if it can be done without slowing down the retrieval process.

Proximity subset searching should still be straightforward to do using my simple vector representation of the database, but as files get bigger it may be necessary to hold the large subset vectors on disk and page them into memory only as needed. Modern operating systems with large virtual memory spaces should be able to handle that for the program automatically. If I keep the current default 32-byte quantization, then the subset vectors will still be only 0.4% as big as the total database text size, and so even in the multigigabyte zone each subset will only require a few megabytes of space.

CONCLUSIONS

My bottom-line evaluation is that a free-text IR system such as I have built, plus anticipated extensions, will surely break down -- but not until reaching file sizes in the 10 GB or larger range, or with information arriving at a rate greater than 1 GB of text per day. Then, problems with data transfer rates and with indexing speed may force one to find alternative solutions. Probably a multiprocessor approach using a partitioned database is the best tactic to take at that point. Meanwhile, I see a lot of value still to be derived from my real-time high-bandwidth free-text information retrieval tools, particularly as the costs of data storage continue to decline.

[*Browser and Qndxr were the very first programs, I ported to the NeXT. Sources & Executables will be uploaded to the nets by the end of April -- EJL]*

TECHNOLOGY WATCH

Dave Joerg

NeXT OFFICIALLY ANNOUNCES HIGH-END COLOR WORKSTATION

NeXT, Inc., in response to Apple Computer, Inc.'s recent unveiling of its “wicked-fast” MacIIfx, boasting a 40-MHz 68030, unveiled its own high-power workstation months ahead of schedule. Most so-called “experts” were shocked by what NeXT had developed so quickly. The “next NeXT” as people are calling it, sports a tremendous range of improvements:

•As everyone expected, it has a 25-MHz 68040, which combines the functionality of the 68881 and the 68030, and then some.

• A full two-page color display, “even crisper” then the original NeXT's 2-bit mono. You have to see it to believe it.

• A 32-bit color board, with a dedicated i860 processor, contains its own PostScript and Renderman interpreter (in ROM) and memory slots. It comes with 4 Megs, expandable up to 64. Users can set a “virtual screen size”, which lets users work within screens that are larger than their physical monitor. An extra compositing chip makes the requisite scrolling “damn near perfect”. The visible area of the “virtual screen” follows the mouse at all times. A wide range of ports are on the back of this board to allow for real-time live video display in any software-defined rectangle. Look for videotape and videodisc controllers, as well as live multi-party teleconferencing software in late summer of 1990.

• The highly touted 20 ms Canon floptical drives have arrived: all NeXT's will now be shipped with them. An added bonus: Canon's floptical technology has improved to the point where polarization of crystalline substrate can be recorded with twice the precision, meaning that there can be 2 bits per “dot”, double the original value. Also, the new floptical drives write to both sides of the disk, resulting in a total floptical capacity of 1 Gigabyte. Of course, nothing's perfect: you will have to go out and buy new flopticals. But because of NeXT's staggering secret sales growth (see “SALES FIGURES” below), Canon has given NeXT a break on floptical prices: $25 for developers and academic institutions. It is reassuring to note that the new drives can read both the old floptical format, and, using a special cartridge and laser head, even CD's.

• The “secret weapon” in the hardware wars nobody predicted: Canon's color 400 dpi imagesetter. It's multi-channel redundant-laser paper-latency technology makes it possible to produce glossy's, transparencies, and even color-separations, all within the same housing your old Laser-printer fits into.

•When asked by a skeptic, “Mr. Jobs, where does all this high-powered, low-cost technology put NeXT?” Steve replied, “Right on everyone's desktop.”

NeXT RACES AHEAD IN THE “SERVICE WARS”: PASSIVE-AGRESSIVE UPGRADES

NeXT Inc. once again flew ahead of everyone's expectations in announcing its new “passive-aggressive” free upgrade policy, even more remarkable than NeXT's technological improvements.

> *“In addition, we've sent out the 'NeXT upgrade swat team: a group of hardened professionals who seek out timid NeXT's that don't want to get upgraded, and we UPGRADE THEM ANYWAY. That's the 'aggressive' part. We will leave no NeXT un-upgraded!”*

As explained by Steve: “It's simply not in our best interest to have NeXT's out there that have sluggish 68030's, 92ms floptical drives, with monochrome monitors and printers. So we are upgrading them all. So whenever we get our hands on your computer, be it for servicing, installation of peripherals and memory, or even just to replace a toner cartridge, we're going to upgrade your machine. That's the 'passive' part. In addition, we've sent out the 'NeXT upgrade swat team': a group of hardened professionals who seek out timid NeXT's that don't want to get upgraded, and we UPGRADE THEM ANYWAY. That's the 'aggressive' part. We will leave no NeXT un-upgraded!”

> *“Well”, Steve thought, “what in the hell are we going to put in all that extra disk space?”*

NeXT RELEASES NEW SYSTEM SOFTWARE, SOURCES

Of course, what is new hardware without new software? NeXT began shipping version 2.0 of the system software yesterday, on its Gigabyte flopticals. Enhancements include:7 The on-line documentation is now complete. Never again will you see <<This section isn't available on-line.>>

•Lots and lots of demos. Basically everything a NeXT employee has ever written is in that handy /NeXTDeveloper folder

• More fonts. 32 font families to be exact.

•A new utility, DriverBuilder, that lets anyone use his SCSI disk with the NeXT. NeXT warns, however, that the drivers created by DriverBuilder are not optimized for caching, speed, or maximum capacity.

•All the graphics in the unabridged Webster's.

• Improvements made in the design of Webster's make it easily accessible through the Speaker-Listener Inter-Application Interface. What this means is that anyone can spell-check a document, an entire document, with their NeXT.

Even with all this new stuff included with the system, there was about 450 Megabytes empty. “Well”, Steve thought, “what in the hell are we going to put in all that extra disk space?” Before he even knew what he was doing, he put all of NeXT's software sources on the disk.

NeXT RELEASES OFFICIAL SALES AUDIT -- EVERYONE BUT STEVE IS AMAZED

Representatives from Sun Microsystems, Inc. and Apple Computer, Inc. struggled to retain a grip on reality when an independent auditing agency released NeXT's sales figures to-date. “I thought they had misplaced a decimal point at first,” reported one of the survivors. “I mean, we knew there were some NeXT's out there, but this... there's something unholy about it.” NeXT officials attribute a sizable percentage of its “surprise” sales to foreign wholesalers. “NeXT has made great inroads in places you wouldn't think wanted computers, such as Malaysia, Libya, and Finland. And the U.S.S.R.'s purchase of 25,000 workstations helped push NeXT's sales beyond that of all Macintosh's and Sun's combined.” reported one of the auditors.

NeXT PARAPHENELIA OFFER!

To celebrate NeXT's triumphant product releases, NeXT is now taking orders for a wide range of classy NeXT goods:

- NeXT tote bag. $3.50
- NeXT 3-ring binders. $5.00 / 3 pcs.
- NeXT sunglasses. Dark black, NeXT logo on sidebar. Filters out UV, infrared, magnetic pulses, and electrostatic waves. “Only good photons get through.” says Steve. Suitable for wearing while using a NeXT. $12.50
- NeXT turtleneck. Classic. $16.00
- NeXT “racecar gloves”. Black, sinister, leather. $22.95
- NeXT boxer shorts. Steve says, “Heck, I wear 'em every day!” $9.50

WHAT TO LOOK FOR IN THE FUTURE

A very reliable source at NeXT has secured this list of planned product releases for late 1990. Keep your eyes peeled.

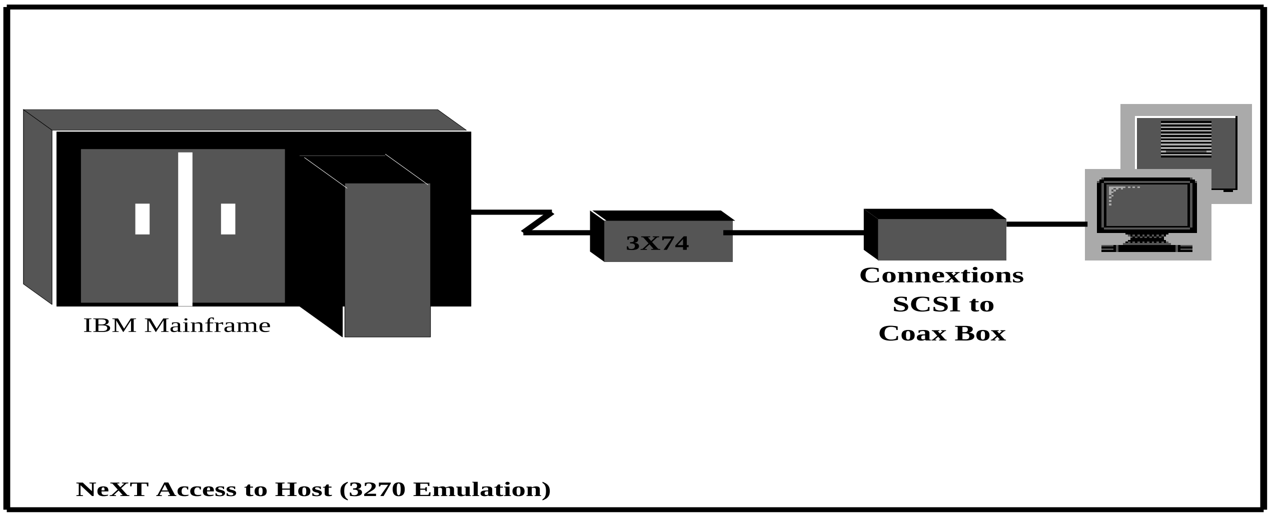
- Macintosh and IBM sub-simulations. BuzzNUG 's technical staff, while getting a sneak preview of the “next NeXT”'s logic board, noticed an empty space for a chip, suspiciously close to the size of the Macintosh ROM chip. It seems that if one can obtain a Mac ROM (not too hard nowadays), NeXT is developing the proper software to run a Mac simulation inside a NextStep window. Similar efforts are reportedly underway for the IBM-compatibles, and NeXT has apparently found a technologically and legally feasible way to simulate an IBM-compatible with software only. This would make NeXT the truly over-arching OS standard for PC's, being capable of running X Windows, Mac OS, MS-DOS and OS/2, and NextStep all from one platform. The public probably won't get this until summer of 1990.
- Advanced hardware support for Renderman. Jobs' close ties with Pixar finally pay off. The new two-page color monitors being distributed have a latent polarizing filter, meaning that for double the graphics memory space, one can easily create true stereoscopic images on the NeXT. The NeXT sunglasses (see “NeXT Paraphenelia” above) come with cross-polarized lenses, and are all one needs to get the 3-D effect. Of course, the Renderman driver in ROM on the 2-page color displays has its own algorithms for translating 3-D data into stereoscopic fields, or you can write your own. Look out for a 3-D windowing environment in the 3rd quarter of 1990.
- A “CPU board” for the NeXT that augments the “next NeXT”'s processing power by at least a factor of 4. High-bandwidth NanoBus communications allows the monetarily endowed user to install up to 3 of these boards in the 3 empty slots. Apparently the processors have to work independently; that is, a single Mach thread can be executed by only one processor, but the presence of extra processors reduces the total load on any one processor. NeXT will be releasing both the “CPU boards” and technical documentation on how to write good multi-thread applications to developers three months previous to the general release. We're really not allowed to say when this will be available, or even what kind of processors it will use.

In the 21st century, there will be only ten truly revolutionary advances in the way we use computers. Here are seven

WHAT TO LOOK FOR IN THE DISTANT FUTURE

The following was “liberated” from a top-secret NeXT internal document, titled “In the 21st century, there will be only 10 truly revolutionary advances in the way we use computers. Here are 7:”

- Users will be able to choose from a wide variety of input devices: keyboard, mouse, infrared gesture scanner, eye motion tracker, voice-activated command shells, or even a trackball.
- Output devices will be both powerful and unimposing: “super-twist” ultra-light LCD's, multiple-depth electroactivated color hologram blocks, and a “sliding-pin-matrix” for the blind.
- New mass storage media: a newly discovered strain of blue-green algae has a pigment that, when frozen, will “flip-flop” its chemical structure when struck be certain frequencies of laser light. This same pigment will make effect phase transforms upon laser light of a different frequency, and the amount of phase transform depends on the “flip-flop” state of the pigment. We're talking terabytes of storage per square inch here. Wrapped on a cylinder with multiple floating read-write heads, this system achieves under 10ns average seek time, with a sustained transfer rate of over 2GBits/sec. Using hydraulic mechanics techniques not unlike that in a centrifuge, this new hard drive under development at NeXT runs at over 100,000 rpm. Eerily enough, the tremendous gyroscopic force exerted by the drive, which is not placed at the exact center of mass of the cube, makes the cube's corner gradually rise up until the cube “hovers” at exactly 28 degrees to the horizontal. Will Steve's vision ever fail?



NeXT Access to Host (3270 Emulation)

Cheers, Dave

NeXT and Communications in the IBM World

Daniel A. Powers

This report will identify possible solutions for getting NeXT to talk to the IBM world. If NeXT is to be used as a business machine, many companies will want to be able to plug NeXT machines into their existing networks. In many cases these networks are completely IBM, no Ethernet.

Access to Host (3270 Emulation)

It is a challenge integrating the NeXT into the IBM communications environment. The NeXT, in order to be integrated in the IBM environment, needs host connectivity. This connectivity could be via coax or through a Token-Ring. The NeXT would also need a way to communicate with other IBM PCs and IBM RTs on the Token Ring Network and a software package that does 3270 emulation.

Conextions, a NeXT development company from Nashua, NH., has helped identify possible networking/ communication solutions. Conextions has a software product called 3270Vision. 3270Vision is a 3270 emulation program for the NeXT Computer.

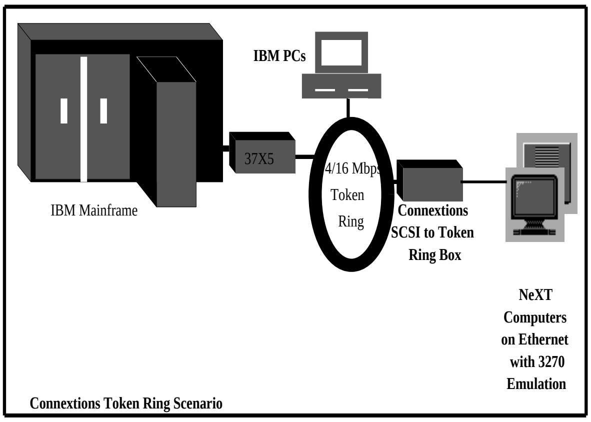
Conextions is coming out with a product this summer that will allow a NeXT to connect to an IBM mainframe via a coax connection to a 3x74. This hardware device, used in conjunction with 3270Vision, enables the user to have full screen IBM terminal emulation. There will be no way for the NeXT Computer to communicate with other IBM PCs and IBM RTs, since the NeXT will not be connected to the Token Ring. The diagram below shows Conextions solution for host access:

The new coax hardware device will be a little black box with a cable coming out from each end. One end will be the actual coax connection, the other will hook up to the NeXT's SCSI port.

This solution is similar to putting an IRMA card in a IBM PC. The NeXT, with this solution, will be able to have up to 5 host sessions active at once.

Other Scenarios

The scenario, shown in the diagram above, is possible with hardware and software products available today. It would be a complete solution that would give the NeXT computers access to the host, and allow the NeXTs to communicate with IBM PCs, and IBM RTs through the Token Ring.


IBM PCs
37X5
4/16 Mbps Token Ring
IBM Mainframe
Connextions SCSI to Token Ring Box
NeXT Computers on Ethernet with 3270 Emulation

Connextions Token Ring Scenario

* Ethernet to Token Ring Bridge with Transparent Bridging and Source Routing.

* 3270 Emulator on NeXT Sees IBM Mainframe on Ethernet.

* IBM Mainframe Sees 3270 Emulator on Token Ring.

* All TCP/IP Traffic thru IBM 8232.

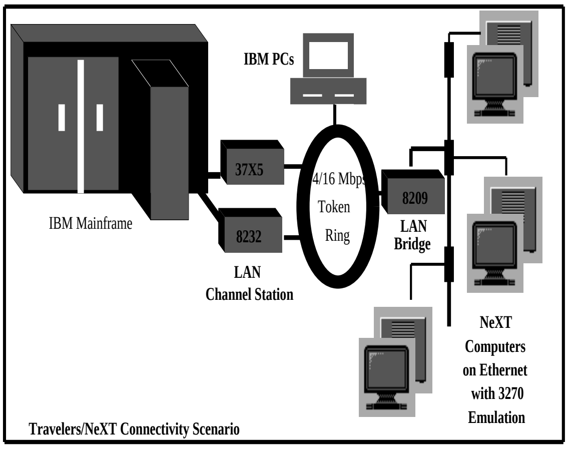
* All SNA Traffic thru 37X5 Gateway.

In this scenario, the 8232 LAN Channel Station, and the 8209 LAN Bridge (both IBM hardware devices) have to be purchased. Software for the IBM mainframe would also have to be purchased (TCP/IP for MVS). This software, combined with the 8232 LAN Channel Station, allows an IBM mainframe to connect to a Token Ring network and use the TCP/IP network protocol.

The IBM 8209 LAN Bridge connects an Ethernet network to a Token Ring network. With Conexion's 3270Vision program installed on the NeXT computer, the user will have full screen IBM terminal emulation to the Token Ring-attached IBM mainframe.

With this scenario the IBM PC, IBM RT, and NeXT computers on the Token Ring can all communicate with each other through TCP/IP programs (Telnet).

The following scenario will be possible at the end of 1990. Conextions will be completed with another hardware device similar in design to the SCSI to coax connector discussed previously. This device will be an SCSI to Token Ring connector.



This scenario will allow the NeXT Computer to achieve host access through Token Ring, by using Conexion's SCSI to Token Ring hardware device. 3270Vision software will again be used to give the NeXT full screen 3270 emulation with up to five concurrent host sessions. The NeXT Computers will be able to communicate with other IBM PC and RT computers connected to the Token Ring. The cost of this solution would be about $1500 for Conexion's hardware and software.

Networking/Communications Summary

These scenarios are all possible within the year. Although Conextions is the only company reported on, hopefully there will also be many other companies coming up with the above mentioned solutions by years end.

DUST AND YOUR NeXT

Dan Port, NeXTessories

Have you noticed a bit of extra noise coming from your cube of late? Do some, or perhaps all, of your optical disks have problems mounting, spinning up, down, up, down. Or do you get the dreaded “inconsistent file system, partition dirty”? How about optical disk read errors? If any of the above problems sound familiar then it is likely that your cube has been sucking up too much dust. In this article we will tell the story of dust and the NeXT: what the problem is, how it came about, what you can do about it, and what NeXT is going to do about it. The main purpose of this article is to clear up misconceptions regarding this problem and help you, the NeXT owner, keep a clean and working machine.

The cube is designed to have more than adequate ventilation to keep the power supply, motherboard, and storage devices within normal operating temperature

The problem: Dust builds up inside the optical drive, causing drive failures.

What is happening: The cube is designed to have more than adequate ventilation to keep the power supply, motherboard, and storage devices within normal operating temperature. To accomplish this there is a vent at the bottom front and a fan in the backplane. Ideally air should flow in through the bottom vent along a track inside the cube (under a wire mesh) and up through the back and out the fan. This works well as the heat gets drawn into the track and outside air is not directly blown into the power supply or motherboard, thus having the advantage of reducing dust buildup.

All of this works fine until an optical drive is added. The optical drive adds a third opening which air can flow through. Ideally the air will flow into the vents at the lip of the OD, but unfortunately the optical drive is open in the back, so air will flow through the OD. It does not flow through completely, however, because the flow is blocked by the things inside the OD. Thus the OD acts like a filter and dust becomes trapped inside it, settling on the laser platform, stepper, and other critical functioning parts. Eventually this leads to failure. An inserted optical disk actually aggravates the problem by increasing air flow through the OD.

Why: Although the cube is splendidly engineered, it is well known that the optical drive (built by Cannon) was designed independently from the rest of the cube. We speculate that the Cannon engineers, in their rush to produce the first read-write-erase OD, simply neglected to consider the cooling scheme of the cube. Further, most of the testing of the cube was probably done in relatively dust free environments. These facts, combined with a relatively long latency period for OD failure, may have contributed to this oversight during testing.

One popular rumor regarding this problem is that it can be solved by reversing the fan, so that air will flow through the cube from back to front

What NeXT is doing about it: At present they are just replacing the OD's. They have been working on a retrofit solution to the problem involving a filter placed in the rear of the OD and a preventative maintenance/cleaning kit. As of this writing, NeXT has no projected release date. NeXTessories (that's us) has also been communicating solutions and preventative maintenance/cleaning kit ideas to NeXT. Even further down the line, it is speculated that the next generation OD's (higher density, perhaps double sided) will be re-designed to be immune to the dust problem.

What you should not do: One popular rumor regarding this problem is that it can be solved by reversing the fan, so that air will flow through the cube from back to front. It fact, it has even been rumored that NeXT is actually doing this on their newer models. None of this is true. Reversing the fan destroys the carefully planned airflow patterns in the NeXT, and can lead to overheating and other severely dangerous problems. Many other rumors are similarly pernicious, be cautious before acting on rumors.

What you can do about it: Obtain the document “Getting the Dust out of your NeXT” available via the ftp archives or from NeXTessories. This document describes several methods for cleaning the dust from your OD and gives advice for maintaining a clean cube. You might also consider purchasing a dust filter for your OD. One major reason NeXT simply replaces failed OD's (rather than having the service center clean them) is that it is thought that the dust that gets trapped inside the laser platform cannot be removed by simple cleaning (or otherwise). This eventually leading to an OD failure that is unresponsive to cleaning.

In the meantime, NeXTessories has experimented with various methods of retarding dust buildup and preventative measures. Although there are several possible solutions, some are risky and could cause damage. NeXTessories offers to the public what it considers “safe”, foam filters specifically designed to deal with this problem. These filters have sufficient flow-through to insure that the cube remains properly cooled, but they do reduce the amount of dust that enters the OD, both through the front entrance and from the vents underneath the cube. They are sold nearly at cost as a service to NeXT owners. For more information of dust filters get the document “About OD dust filters” from an ftp archive or write NeXTessories at:

s-mail: NeXTessories, MIT Branch P.O. Box 128, Cambridge, MA 02139

e-mail: dnp@math.mit.edu, mcarthur@-math.ucla.edu

The Right Cursor for the Right Job

c1990 Andrew C. Stone

Recently, I had the task of including a graphic view in a window with other tools and controls in it. Since I wanted the cursor to change to the cursor appropriate for this view, a cross hair, I had to install my own cursor rect to let the window server know to change the cursor to a cross when entering the drawing view. Had this been a scrollView or a clipView, it is a simple one line task:

```
[scrollView setDocCursor:crossCursor];
```

This is very useful if you are using the scrolling text view from IB, and want it to be read only: you might want to set the cursor to an arrow to show it is not editable.[Downside: users can't conveniently grab the text for copying.]

```
[scrollView setDocCursor:NXArrow];
```

But my challenge was for a non-scrollable, what you see is all you get, view. The window will maintain a list of the cursor rects in it, and you could call the [window resetCursorRects] method to reset the cursor in your view. However, this is horribly inefficient and according to the 1.0p docs, “To be optimized...” It turned out to be quite simple, and most elegantly terse:

First, override the resetCursorRects method in your custom view. This is called for each subview in the window when the application sends the window a resetCursorRects message. Upon receiving this message, we respond with the visible portion of our view and the desired cursor:

```
- resetCursorRects
{
     return [self addCursorRect:&bounds cursor:[currentGraphic
cursor]];
}
```

And, then alert the window when things have changed. In the case of a subClass of GraphicView, when we change tools on the tool palette:

```
- setCurrentGraphic:sender
{
     [super setCurrentGraphic:sender];
     [window invalidateCursorRectsForView:self];
     return self;
}
```

The beauty of the NeXTStep Environment is that after spending hours debugging some bizarre postscript anomaly and getting tremendously concerned for the future of our providers of black equipment, you tackle a task in six lines and six minutes what you feared would take several hundreds of each.

Andrew Stone

Accessing Files on the Archive Sites

Bryce Jasmer, jasmerb@cs.orst.edu

The NeXT is bringing UNIX to mere mortals and with the birth of a new computer usually comes the birth of a different way of transferring files. This article will try to explain how one can transfer files from the archive sites to a NeXT that is connected to a remote UNIX machine over phone lines. (see figure 1.)

```
     NeXT MachineUNIX Machine UNIX Machine
        name:       =====>    name:    =======>Archive Server
     localnext     phone remoteunix ethernet    name:
          cs.orst.edu
```

figure 1.

I. Important Programs

There are several important programs that you need to be aware of that will allow you to access the archive sites. The following is a brief description of each.

A. kermit

Kermit provides error free transmission over serial (phone) lines. It needs to be installed on both “localnext” and “remoteunix” for you to be able to transfer any files. Getting kermit onto your NeXT is a little bit of a Catch-22. It is recommended that you find someone who has it on an Optical Disk.

B. ftp

Ftp provides error free transmission over ethernet and is the program that you will use to access the NeXT archives at “cs.orst.edu” from “remoteunix”.

C. tar

Tar allows you to group several files and directories into one single archive for ease of management. Tar stands for “Tape ARchive” and was used extensively for backing up files to a tape drive. There are lots of options to tar and you will be shown the essential ones in the example transfer session below. Files that have been tar'd together traditionally have the file name suffix '.tar'.

D. compress/uncompress

Compress and uncompress are standard UNIX utilities for compression of files using the Lempel-Ziv algorithm. Most (if not all) files that you find on the archives will be compressed to save as much disk space and transfer times as possible. Files that are compressed have the file name suffix '.Z'.

E. uuencode/uudecode

Some equipment that you might be transferring your file through could have difficulty dealing with binary files. The modems that I dial in through have this difficulty so I must convert all files into ASCII text before transferring them. The program to convert a binary file into a text file is uuencode and the program to convert that text file back into a binary file is uudecode. The file name suffix that I use for uuencoded files is '.uu'.

II. Example Session

From “remoteunix”, you will need to connect to cs.orst.edu using ftp.

```
remoteunix> ftp cs.orst.edu
```

and you will log in as the user “anonymous” and by convention will use your login name as the password. You can use the same basic set of commands within ftp as you can within your normal shell (sh/csh). The two most basic commands are 'cd' and 'ls'.

To transfer binary files you must set ftp into binary mode with the 'binary' command. Another useful command is 'hash' which prints out a hash mark for every 1024 bytes transferred. And to do the actual transmission you will use the 'get' command.

```
ftp> binary
ftp> hash
ftp> get LockScreen.tar.Z
```

When the transfer is finished you can simply issue 'quit' to quit the ftp session.

Now you will need to convert the binary file into a text file before transferring the file over a phone line to your NeXT.

```
 remoteunix> uuencode LockScreen.tar.Z < LockScreen.tar.Z >
     LockScreen.tar.Z.uu
```

The first argument to uuencode tells what you want the file to be named after you uudecode it. It will usually be the same name as the filename you are uuencoding.

Now you are ready to transfer from “remoteunix” to “localnext” using kermit. You will use the -s (send) option in kermit to send the file.

```
 remoteunix> kermit -s LockScreen.tar.Z.uu
```

this puts the remote kermit into send mode and you need to escape back to your local kermit (running on localnext) and give the command to receive. To do this you need to type a “Control-\ c” and then type 'receive' at the local kermit. Kermit has a lot of options to it and can be confusing at times. It is recommended that you read the manual to kermit for more information.

After the transfer is finished you need to give the escape sequence again to get back to “remoteunix”. You can now log out if you want now that you have the file on your NeXT. Kermit will usually rename the file and change uppercase to lowercase and replace all the periods except the first one with an 'x'. So the file we just transferred will be “lockscreen.tarxzxuu”.

Now that you have the uuencoded file on your NeXT, you now need to uudecode it. The (binary) file that will be created when you uudecode the file will be have the name that was given to uuencode earlier.

```
localnext> uudecode lockscreen.tarxzxuu
```

This will create the compressed file “LockScreen.-tar.Z” which needs to be uncompressed. Note that the uuencoded file will still be there.

```
localnext> uncompress LockScreen.tar.Z
```

This uncompresses the file and replaces the compressed file with the new uncompressed file. You are now at the last step and all that is left is to un-tar the file into its respective files and directories.

```
localnext> tar xvf LockScreen
```

The 'x' option tells tar to eXtract from the file instead of adding more onto it. The 'v' option is for Verbose which tells tar to print the name of each file and other information as it is extracted. The 'f' option tells tar which File to extract from.

After this command you will normally have a new directory created with all the files in that directory. Sometimes the files will extract into the current directory instead of creating a new directory. This all depends on how the file is originally tar'd up.

If you have any questions, feel free to ask.

April Phiel’s NeXTopoly Game

Eric P. Scott, San Francisco State University

New for 2Q1990: NeXTopoly Yet another of the ***opoly derivatives licensed from Parker Bros. (The first board game whose camera-ready artwork was produced on a NeXT!) Plays the same as the usual ***opoly games, but has an appropriately themed board, tokens (the little MegaPixel is sooo cute), black dice (what else?), and cards. Here's a sample:

“Take a spin to Optical Drive. If you pass `Boot' collect $200. If the property is unowned, you may buy it. If the property has been owned for more than 3 turns, you don't owe any rent; it's been condemned.”

Basically, it's ***opoly for the 1990s. Buses have replaced railroads, and here's a switch--there are more utilities than buses. But hey, that's UNIX for you.

The game is full of references to NeXT's colorful (er, grayscale?) history and locale. (e.g. the “Get out of jail free” card is now a “Get out of Fremont jail.” The cubes are assembled in Fremont, California.) But many familiar elements of the traditional game are still present; for example, the “Big Blue” properties are the most expensive.*:) -- EPS*

REVIEWS, RUMORS and STUFF

A Myopic Eye into the NeXT World

Erica J. Liebman, erica@kong.gatech.edu

What's New at NeXT

First of all, congratulations are due to our regular correspondent, Conrad Geiger, who was just promoted to the director of Western Sales!.

Patty Kammerer of NeXT is finishing a NeXT quarterly newsletter which is going to be mailed out to the NeXT Community, real soon now.

New Technical Support Notes: "NeXT Serial Port Connectivity", "So you want your Applications to Fly", and "Changes in Font Structures" were mailed out last month.

Adobe

Adobe Sent out their own set of Technical Support Notes, including a dandy IB/DPS tutorial, a DPS Overview, DPS Operators: Single Calls vs Wraps, Coordinates and Stroke Adjustments and an invitation to join their Developers Association. Alas, I can't afford it. A gratis membership will be *very* welcome.

Adobe has provided the Adobe File Server. This is an automatic mail-response program. That means that you mail it a request, and it mails back the response. To get in touch with the server, send a message to:

"ps-file-server@adobe.COM" OR

adobe!ps-file-server

NewsLetter Alert.

Tom Poiker has put out a new Newsletter, NeXTView. Articles in the first issue included "Graphics on the NeXT", "New Products", "Technical Talks", "Literature Search" and "NeXTTips & Questions". Boston Computer Society, Washington Apple Pi and the Bay Area NeXT Group are all planning their own newsletters.

New Book

The **NeXT Bible**, written by Douglas Klaap is due on your bookstore shelves this month. Initial reaction has been mixed. I haven't bought my copy yet. The consensus is that it is very good for the naive user and has some handy information for the more experienced NeXTer.

FTP Sites

The University of Texas/Arlington has started a new NeXT Archive site: eesun1.arl.utexas.edu/129.107.2.51. For More information contact Travis TeSelle at (817)794-5024, teselle@eesun1.arl.utexas.edu.

Don't forget:

cs.orst.edu

j.cc.purdue.edu

uvaarpa.acc.virginia.edu

and umd5.umd.edu

TERM Alternative: STUART

Sick of *Term*? You may want to try **Stuart** instead. I did, this week. Stuart is available from all the major ftp sites. From the very first, I was impressed by its improvements. First, you can create multiple session windows.

Second, you can alter the font with ease -- without running any silly shell scripts, I was able to get readable screens in seconds. On the other hand, while the menu supported "copy" and "cut", I was unable to use any of the standard menu features. I found this annoying, along with my inability to paste input from one window into my Stuart window. VT100 emulation through playing nethack, running readnews, etc worked flawlessly, but I found the program swallowed my tildas -- making it very hard to leave my tip session. I ended up having to use the "mystery-tilda", which is the shift-ESCape key.

On the whole, with the above exceptions, I found this to be a well behaved program that I will not hesitate to use again. Recommended.

Frame: The first week

Ok Dudes. The issue is late. The big reason was my learning phase for my brand new **Framemaker** program. The kind folks at Frame were gracious enough to donate a copy of Framemaker to the newsletter, for which I am very grateful.

Framemaker has a hell of a lot going for it -- and a few things which just drove me nuts. Putting together a professional looking brochure or newsletter can become a matter of hours rather than day -- my first newsletter attempt was done in under an hour with only a few hours of reading manuals. Before you get worried, I need to explain something right off. I **like** reading manuals. It is the first thing I do when I get a package. Sneer at me, ostracize me or make fun of me behind my back. I've stood though it all. I simply enjoy reading manuals. The Frame manuals were especially good reading -- clear directions, well designed presentation, strong spiral bindings, etc. I was impressed from the start.

The first obstacle which I had to pass is the understanding the underlying metaphor of the program. Framemaker is primarily a page layout program -- not a drawing program, not a word processor although it does those too. Framemaker's raison d'etre is personal publishing. It is best suited to inhouse publishing of brochures and publicity materials of perhaps 2-10 pages in length.

> *The best feature of Framemaker is the Paragraph Catalog - I loved it from the very start*

I found it hard to add graphics initially, although I am getting better at it. (Take a look at some of the pictures included in this issue -- some of them are a tad warped. Alas.) Graphics are added in several different ways: you can't enter EPS exactly in the same moves as TIFF. Also, you can't just do a cut/paste from draw or textart, etc. You must work at the file level -- even if you don't want to use references.

Laying out columns was also a mystery that I had to learn, although once I started to pick up on the ideas, I began to really like this. Master pages lay down default column setups. The columns of these setups can be edited directly on the body page level. You can not, however edit the layout "art", such as the grey bars on the left of each page or the column separation line.

Another mystery is the use of the "frame above/below" in the advanced section of the paragraph layout -- although I created a number of specialty frames on the reference page, these weren't reflected in the paragraph layout dialog box.

Quotes were a delight. You may have noticed the bold standout quote sections -- simple to do and very attractive. Again, though, I had to copy sections of text to where I wanted the quote placed and format the paragraph accordingly.

When the document size started to exceed fifteen pages, the response time really started to crawl -- the way to avoid this is have your copy proofread and pasteup ready in another program - writenow or edit - and use Framemaker for strict layout. The time will still be slow, but you avoid the five second delay in typing a return and waiting for the screen to advance to the next line.

The best feature of Framemaker is the Paragraph Catalog - I loved it from the very start. With a mere click of the button, I could format my first paragraphs flush left, my following paragraphs with an indent. Titles were a mere click away, with a strong header bar and an italicized author section.

Another lovely feature is the column connections. Text flowed correctly no matter how you set up your columns. It is a little weird trying to figure out that you need to click on column a, then on column b and then select connections. There is no indication that column a is 'remembered', but I assure you it is and it works wonderfully.

I do not hesitate to recommend Framemaker to the corporate market in particular. Promotional materials can be easily put together in an attractive, but cost effective, manner.

I'll be continuing my work with Framemaker over the next few months -- I hope to include tips in our hints columns. With some feedback from our readers, I'll be putting this newsletter in fine form indeed.

Market View

New This Month

THIS SECTION IS PROVIDED AS A COURTESY TO OUR READERS AND THIRD PARTY FRIENDS. BuzzNUG IS NOT RESPONSIBLE FOR THE CONTENT OF THESE OR ACCURACY OF INFORMATION PRESENTED IN THIS SECTION.

•Digital Instrumentation Technology Inc (505)662-1459 is shipping beta copies of **Cube Floppy 1.4**, a 3.5" floppy disk drive connecting to the SCSI port, reading and writing MS-DOS(720K and 1.44MB disks), UNIX and Macintosh (1.4MB disks) file formats.

• WeDesign Inc (415)-479-1105 sent a flyer on **TheLibrary**., an on-line information system with Objective C references and authoring tools.

• Pacific Micro (415)948-6200 is shipping the **PM1.44** and **PM HDE** external 3.5" floppy drive and hard disk enclosure. Special order line is 1-800-628-DISK.

• BYTE's **BIX** service has a special NeXT interest group. Contact 1-800-227-2983 (BIX Customer Service) if interested. Please mention both BuzzNUG and Dave Andrews (of Byte) as contact names.

• **Cassandra Version 1.0**

Jiro writes "The changes to it are mostly in the User Interface. It took me a while to decode the Rich Text Format manual that comes with the 0.9 docs (Microsoft writes terrible manuals), and NeXT doesn't fully support RTF in its Text view, but because I am now using RTF, a whole avalanche of display options are now open to me. I therefore changed the display characteristics of both the Overview window and Today window.

I think you will find the new format MUCH easier to read. It also support proper highlighting (boldface) of high priority events.You can now change the fonts in the notepad and event log windows. A font panel and submenu has been provided for this. You can also cut and paste font-ed documents into these windows. Because of this change, however, users of v0.9.2a and below will have to convert their notepads and event logs over to 1.0. Please read the READM09USERS document for more information

I was a bit hurried and didn't have time to rewrite the Events database as I had originally planned. It didn't help that my NeXT was down for most of Spring Break. Therefore, I wasn't able to implement half as many things as I promised I would. I sincerely apologize for this.... It's inexcusable to not live up to one's promises.... On a more superficial level, the user now has the ability to disable the display of the alarm panel and only play the alarm sound. The main display windows now remember their positions between sessions (something a lot of users asked for). For more information on changes, see Section 6.0 of the manual.

It's taken a long time to get this far and sometimes I wondered if it was worth it. But doubts aside, it's here. Enjoy Cassandra's first real release! *- Jiro Nakamura* - *jiro@heights.cit.cornell.edu*

• **NeXTessories**

NeXTessories (pronounced next-cessories) is a coalition of programmers and technicians dedicated to providing low cost, innovative, high quality accessories and software for the NeXT computer. We believe that the NeXT is the ideal platform for this purpose. Here is a list of our current projects:

SOFTWARE:

NeXTtime: A utility that uses your modem to set your clock to the national standard.

ScreenSaver: An alternative to the standard dim feature. A variety of dynamic graphics automatically appear when the cube is idle.

GIF, or Generic Inter-Face: A NeXTstep interface for those dull command line programs such as bc, compress, vi, etc.

Alarm Clock: A fancy alarm utility.

The above will be bundled into a shareware package, each program will cost $5 (this buys you free upgrades and full technical support). Available April 20, 1990

[My Birthday is the 21'st. Hint hint.].

HARDWARE:

Answer-Mate: A very low cost (around $100) fully programmable phone device. Will work as a standard answering machine, i.e. pick up phone, play messages, record messages, DTMF remote operation. Additional features include, call forwarding, recognition of Fax and Modem, multiple (including random) outgoing messages, “voice mailbox” with personalized security codes, time and date stamp, interactive messaging (like the 976 services), message editing, computer phone, and whatever you can program yourself using the new “phone-talk” script language. Beta test versions will be released around June 1, 1990.

NeXT-sentry: A low cost, fully programmable digital alarm system. Can be used to monitor windows, doors, even the cube itself!

NeXTplexor: The NeXTplexor gives you eight more high speed RS-232C or RS-422 serial lines for terminals, printers, etc. via the ethernet interface (does not interfere with normal ethernet operation). If there is enough interest we will offer an SCSI version.

ACCESSORIES:

Dust filters, retards dust buildup inside the optical drive and cube that may cause OD failures

Dust filter mats: Place under the bottom vent or inside the cube to retard dust buildup inside the cube and on the fan.

Dust Covers: All black custom covers for the cube, monitor, and printer. Call for price.

Cables, All black cables, RS-422 to RS-232, RS-422 to RS-422, either null modem or straight through.

For more information contact:

s-mail:NeXTessories/MIT Branch P.O. Box 128/Cambridge, MA /02139

e-mail:dnp@math.mit.edu, mcarthur@math.ucla.edu

Product Listings

I've received literature from NeXT or Third Party Developers on the following products. No warranty, express or implied, is given. The quotes are mostly the developers' and may not reflect reality. Please send up-to-date info and review copies for new products.

•Communicae - Active Systems 1-617-576-2000 "high performance communications package. VT240 emulation" *SHIPPING*

•Wingz Informix Software, Inc. 1-913-599-7100 "graphic spreadsheet featuring advanced charting, desktop presentation capabilities, and HyperScript"

•Scan 300/GS Abaton 1-415-683-2226 "300 dpi flatbed scanner with TIFF compatility"

•DM-N Digital Microphone Ariel Corporation 1-201-249-2900 "software-selectable sample rates from 88.2 kHz to 5.5 kHz per channel" *SHIPPING*

• BUG-56 : DSP debugger, Ariel Corporation 1-408-982-0400/1-201-249-2900, *SHIPPING*

•DaynaFILE Dayna Communications, Inc. 1-801-531-0600 "external, SCSI floppy disk drive to write to standard UNIX-formatted diskettes, as well as MS-DOS formats" *SHIPPING*

•Smart Art Emerald City Software, Inc. 1-800-223-0417 "50 text and graphics effects and easily customized in any NeXT word processor, desktop presentation, or page layout program *SHIPPING, -- I GOT A THIRD PARTY LISTING OF THIS FROM NeXT WHICH SAYS THIS IS NOW OFFERED BY ADOBE! (415)962-2045*

•FrameMaker 2.0 Frame Technology Corporation 1-408-433-3311 "powerful, cost-effective workstation publishing software" *SHIPPING, RECOMMENDED FOR DESKTOP PUBLISHING, PAGE LAYOUT FOR IN-HOUSE PUBLICATIONS, BROCHURES.*

•Artisan Media Logic Incorporated 1-213-453-7744 "high-resolution paint and image processing system"

•TopDraw Media Logic Incorporated 1-213-453-7744 "complete and advanced page-based graphics software" *SHIPPING*

• HSD incorporated US Scan-X 1600/600 415-964-1400. Scanners for “line art and grayscale” *SHIPPING*

•TextArt Stone Design Corporation 1-505-345-4800 "array of tools that allow immediate creation of outstanding PostScript images" *SHIPPING, RECOMMENDED FOR “SPLASH” APPEAL, PROGRAMMING WITH ICONS.*

•Encapsulated PostScript ClickArt T/Maker Company 1-415-962-0195 "combines ClickArt EPS portfolios into a collection of high-quality Encapsulated PostScript (EPS) artwork" *SHIPPING*

•Public Domain Disk #1 - Lighthouse Design 1-800-FOOBAR9 "Public Domain Software & More" (I'm in on this one. Buy it. Please!) *SHIPPING*

•Scematic Entry - Lighthouse Design 1-800-FOOBAR9 "CAD Tool for designing electrical circuit schematics"

•Media Station - Imagine Inc 1-313-434-1970 "archival, retrieval and processing of multi-media information" *SHIPPING*

•Fortran 77 -- Absoft 1-313-853-0050 "Objective Fortran-77"

•DisplayTalk - Emerald City Software - 1-800-223-0417 "Complete development environment for Display PostScript programming" *SHIPPING-- I GOT A THIRD PARTY LISTING OF THIS FROM NeXT WHICH SAYS THIS IS NOW OFFERED BY ADOBE! (415)962-2045*

• *SmartArt*, Adobe Systems Inc. 415-962-2045 “Graphics and headline type effects using Display PostScript”.

•Video Monitor and Projector Interfaces Extron Electronics 1-800-633-9876 "offers three video monitor and projector interfaces"

•Digital Ears Metaresearch, Inc. 1-503-238-5728 ""allows entering and recording compact disc-quality sounds" *SHIPPING, REVIEWED IN ISSUE 4*

•Digital Eye Metaresearch, Incorporated 1-503-238-5728 "allows entering and recording NTSC video images"

•NVT High Density Video Drive New Vision Technologies, Inc. 1-415-285-8744 ""video playback device for interactive multi-media applications"

•JETSTREAM Tape Backup System Personal Computer Peripherals Corporation 1-813-884-3092 "high performance tape backup system"

•A/D64x Analog/Digital Interface Singular Solutions 1-818-792-9567 "a low-cost platform for sound recording, experimentation, and analysis"

•Who's Calling Adamation, Inc. 1-415-452-5252 "lets sales & business professionals keep track of phone calls and other client information" *SHIPPING, THEIR BROCHURE IS REALLY COOL, THIS IS ONE I WANT TO REVIEW.*

•GEMS (Generalized Equilibrium Modeling System) Data Transforms, Inc. 1-303-832-1501 "a flexible way to model economic systems"

•InDia (Influence Diagram Processor) Data Transforms, Inc. 1-303-832-1501 "graphical application for representing complex decision-making"

•Knowledge Retrieval System (KRS) KnowledgeSet, Corporation 1-415-968-9888 "rapidly searches and retrieves information from large databases of text and graphics"

•OMEN III Microstat Development Corporation 1-604-228-1612 "stock quotation and financial system"

•TACTICIAN Plus SouthWind Software, Inc. 1-316-636-5100 "multi-user spreadsheet that supports high-level functions and adds built-in presentation graphics" *I'VE SEEN A DEMO VERSION, BASIC SPREADSHEET FUNCTIONALITY -- IT'S IN THERE.*

•Adobe Illustrator Adobe Systems Incorporated 1-415-961-4400 "graphic design and illustration program for generating high-quality artwork"

•Adobe Type Library Adobe Systems Incorporated 1-415-961-4400 "offers more than 500 different typefaces" *I WANT A COPY OF THIS!*

•Flash Graphics Flash Graphics 1-415-331-7700 "extensive charting, illustration, and text functions in a graphics package for screen, slide and paper presentations"

•InterFax 24/96N Abaton 1-415-683-2226 "combines a 9600 bps Group 3 fax modem with a 2400 bps MNP 5, Hayes-compatible data modem"

•GatorBox Cayman Systems, Inc. 1-617-494-1999 "LocalTalk to Ethernet gateway that translates the Network File System (NFS) protocol into Apple Filing Protocol (AFP)" *SHIPPING*

•MacLinkPlus/PC DataViz Inc. 1-203-268-0030 "kit for transferring and translating files between NeXT and Macintosh environments" *SHIPPING*

•Ethernet PhoneNET, Sound and Interpersonal Communications Farallon Computing, Inc. 1-415-849-2331 "used to build LANs over standard telephone cables"

•Etherport NL Kinetics 1-415-947-0998 "allows the NeXT computer to connect directly to standard twisted-pair Ethernet networks"

•INFORMIX-TURBO Informix Software, Inc. 1-415-926-6300 "database engine for on-line transaction processing (OLTP)"

•INGRES Relational Database Management System Relational Technology, Inc. 1-800-4-INGRES "SQL database engine provides on-line transaction processing (OLTP) in single- or multi- CPU and distributed environments"

•DAN - The Data Analyzer Triakis Inc 1-505-672-3180 "data analysis package for reducing data and generating presentation-quality plots"

•Math++ - Triakis Inc 1-505-672-3180 "C-language math library. Approx 100 math functions"

•Dreams - Innovated Data Design 1-415-680-6818 "Frm the makers of MacDraft, drawing and drafting tools"

•Cross Assember/Simulator Programs - Motorola 1-512-891-2030 "for the 56000 and 96000"

•Fortran, C and Pascal Compilers - OASYS 1-617-890-7889 *SHIPPING*

User Groups

Here are some pointers to Users Groups that may be in your area. Send info if you don't see your group here.

Maryland/Northern Virginia/DC

Washington Apple Pi,

Hugh V. O'Neill, (301)-328-9510, Chairperson

POB 39036, Washington, DC 20016

The purpose of the SIG is to exchange information from an end-user's viewpoint on the capabilities (present, planned, and potential) of the NeXT computer/information system. The goals and objectives include the following:

1. Presentation of relevant, timely, accurate and complete information on the performance characteristics, hardware, and software of the entire system.

2. Discussion of applications in various areas including: education, research, medicine, law, decision making/management, policy making/analysis, science, engineering, mathematics, statistics, humanities, arts, business, governmental activity (federal, state, and local), executive information systems, artificial intelligence (AI), operations research/management science, systems analysis, desktop publishing, office use and home use.

3. Cost-effectiveness/benefits/performance evaluation from the viewpoint of the end-user.

4. Special topics such as networks, information system security/integrity, speech recognition, signal recognition, array processing, digital signal processing, and sound/music.

The NeXT SIG usually meets at 7:30 p.m. on the 2nd Wednesday of the month at the National Institutes of Health (NIH) in Bethesda.

In addition, we have special joint meetings with other organization groups and SIGs at mutually acceptable times and places.

[Thanks **Keith & Lisa** for getting this to me! Hugh also mentioned on the phone that he's going to try to put together a Newsletter focussed on the Applications end-user.]

Phil Phuster ffuster@next.com is listed as another DC contact.

Georgia

BuzzNUG is sponsoring local demos and talks. Contact **Erica Liebman** at erica@kong.gatech.edu. (404)-352-5551 Our next event is an Applications/Programming Clinic, at the end of April. Call me for more info as plans are settled. We **desperately** need a local activities organizer. Please contact me if you are interested in pitching in for things Buzz.

Massachusetts

The ***Boston Computer Society*** (BCS) has a NeXT special interest group. Contact **Dan Lavin** at 1-617-969-6555, or **Jan McPeek** at 1-617-926-4027. BCS's NeXT SIG has 750 members. Contact Dan at BCS 1 Center Plaza, Boston, Mass 02108. They've been meeting monthly since January 1989 and have a Bulletin Board and News-Letter aimed at the Business Community. There are member fees.

Erik Kay reports via Conrad Geiger: BCS NeXT is an affiliate of BCS which has members in all 48 continental states and many foreign countries. The leader of the NeXT group is Dan Lavin. I've been working closely with Dan to set up a combined MIT/BCS NeXT FTP archive server (yes, I know there are 3 big ones already, but I think that because of BCS and MIT, we will have a lot to offer. We'd like to make it sort of a developers archive site, with the GNU source, lots of programming tools and classes, X Windows, etc.). Dan's phone number is (617)969-6555 if you'd like to get in touch with him yourself. The user group has a fairly large turnout which ranges from an average of 30-50 and has exceeded 100 at times. Partly because of it being BCS, it has attracted many commercial developers to come talk. People who have been there have included Cayman, the people who make the Scan-X scanner, Media-Logic, Stone Design (Andy Stone came himself), the people who do Digital Eyes/Ears, and others. ... Next month's meeting will have WingZ making it's world product release announcement. Erik Kay

.California

Robert D. Nielsen, nielsen@everest.sjsu.edu, 1-408-995-5775 coordinates ***BaNG***! out of San Jose, loosely associated with Stanford. Robert is the San Jose State University NeXT Campus Consultant. (Atta-boy!) Definitely call if you live within fifty miles of San Jose. Or are willing to drive further. Or have your own 'copter. A newsletter is planned.

The first big meeting will be on 3 May from 6-10 pm. Robert Poor and Julius Smith of NeXT will be talking on “Sound & Music using DSP”. The meeting will start with a catered social hour, compliments of NeXT themselves, followed at 7pm by the lecture. An open forum follows, with numerous third party vendors participating. THIS is the meeting to go to. **HEY GUYS, fly me out there for this one!**

Where: Braun Music Center, Room 110 (rehearsal hall) Stanford University, Free Parking. Ask Robert for Free Map or get it directly from the archives. It is in postscript form. Robert says that the map, done by Eric Ly, is “REALLY hot”.

The meeting will start with a catered social hour, compliments of NeXT themselves, followed at 7pm by the lecture. An open forum follows, with numerous third party vendors participating. THIS is the meeting to go to. ***HEY GUYS, fly me out there for this one!***

Paul Lowe (714)787-3883 at the University of California at Riverside is interested in seeing what others are doing with their NeXT Cubes to distribute to the other NeXT Users (six so far) on campus. write to: plowe@ucrac1.ucr.edu

Texas (yee-hah!)

The report goes that there “is some massive new N eXT user group down there”. Says **Jerry Goode**: “Hi there! Just wanted to get back to you with some info about the NeXT user's group down here in Dallas, Texas...

here is the name of the fellow who is heading up the group down here: **Dirk Hardy**/Hofbauer Information Systems/5080 Spectrum Drive/Suite 912W (Lock Box 21)/Dallas, Texas 75248/Phone: 214-385-2991 *(Late breaking news, additional contact name, no guarantees:* ***Charlie Lindahl - Dallas- Ft. Worth Texas....lindahl@evax.arl.utexas.edu***)

Hofbauer is a NeXT registered developer doing courseware authoring tools, and Dirk is working in conjunction with Dr. Ali from North Texas State U. to get the group going. I encourage you to get in touch with him - he's got a lot of creative ideas! Hope to have an email address for him soon.....Finally, we plan to meet the third Thursday of every month at 7 pm, somewhere! The logistics are still a bit up in the air as we try to figure out how much this thing will grow. If the response after our first meeting is any indication, we could settle in at around 50 people. By the way, I have had Buzzings forwarded to me and it's GREAT. Dirk has copies of both issues so far and was really impressed. You may want to talk with him about organizing a Texas contingent contribution as we try to get things rolling here. "

Washington (the state)

University of Washington (with over 110 cubes) ha s a NeXT User group dating back to March 1989. Contact **Corey Satten** (corey@cac.washington.edu).

Oregon

It looks like Bryce may be starting something up there, contact: **Bryce Jasmer** at jasmerb@cs.orst.edu

Minnesota

Scott Hess,scott@gacvax1.bitnet says: "I was wondering if you have had anything about the Minnesota Next User's group in Buzzings. I can talk to them and see what they want to put in, as we have our March meeting on Tuesday (the 12th). It would be nice to get our name out there". *Hopefully, there will be more information, next issue.*

Canada (O!)

Vancouver NeXT User's Group (in BC, CA) is headed up by **Lionel Tolan** (lionel_tolan@cc.sfu.ca). **Tom Poiker** of the University of British Columbia is starting a new newsletter NeXTVieW (name tentative) with **Shirley Chan**. (poiker@whistler.sfu.ca) Tom's particular interests are geographic imaging on the NeXT. [Pretty cool, eh?] The first issue is out and available via the nets -- a very nice job, focussing on the applications level of NeXT use. **Tom, by the way, owes me a review of TopDraw!**

Colorado

Up Denver way, we've got the contact name of **David Hieb** -.davehieb@alumni.Colorado.EDU

Utah

Mitch Green of NeXT reports of the emerging Salt Lake City group, "The contact in Salt Lake City is probably going to be Tom Pier, but I don't have an email for him right now."

Ohio?

Gary T. Lang (well!glang@apple.com, mips!apple!well!glang@cis.ohio-state.edu) says he's trying to put together a newsletter aimed at the business market.

SCENES FROM THE NEXT ISSUE

Well, we're a little low on committed stuff for next month -- now is the time to start contacting me, dudes. Hopefully we'll have the long delayed report from John Corey and an update from Morris Meyer on superior-computing! (The california version of the cray).

I'll be doing another "adventure in graphics education" and will document my "dock-printer" if it gets debugged by then.

Jiro Writes:

Hit upon an (actually two, but one is only "eh") idea for next month's issue:

- An ongoing seminar about programming in IB with the ubiquitous 'calculator' example. Actually I've wanted an all-purpose calculator for some time now. 'Bout time I programmed one, huh?

- jiro -

p.s. when is the next deadline?

Please Get me the final version of articles no later than the 1st of a Month.

We'll have some more reviews & stuff, but PLEASE GUYS, get me articles as soon as possible. Thanks!

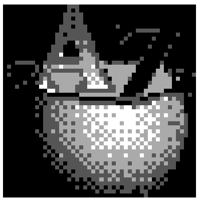
Buzz's Hint Corner

• (Hint? or April Fools.).Check this out: in Icon, use the "Grab" menu item [command-g]. You get the black cursors [actually small thin windows]. Now, attempt to grab the bits of the NeXT WorkSpace Icon [the cube]. Surprise! Reluctant bits! *Andrew Stone*

• The upper right hand corner close box goes to an incomplete "X" in most applications when a document has not yet been saved. The mini-window may also be grayed *EJL*.

• If you remove localhost from /etc/hosts, the net info manager will not run, you can't print and everything else will slow down on the order of 300%, for example password lookups, etc. So, be careful out there! *Ron Hutchins*

- Thanks to Buzzings [Journal] for Librarian custom icon addition tips. I got carried away, and designed one for my “misc” directory, which is where everything seems to go. For your collection is misc.tiff. *Andrew Stone*.



- Forgot a Unix Command? Try apropos from the shell. For example, try “apropos dsp” and see all the dsp utilities that are included from the shell.

NeXT Support Answers,
Care of Doug Kieslar

Q: My system hangs early on in the boot process when booting from the hard disk [or optical; see note below].

Q: I heard that there's a bug that corrupts the kernel. It hasn't affected me yet, but how do I keep this from happening to my machine?

A: See the bottom of this page for the answer to the second question. Here's the answer to the first:

Immediately after powering up, get to the ROM monitor. To get there, hold down both Command keys and press tilde (the upper left key on the numeric keypad). You should see a window with a “NeXT>” prompt. Then type “bsd” to boot from the hard disk and to see the output that normally isn't visible. If the last thing the system prints out is something like:

```
<date><hostname> mach: physical memory = 7.99 megabytes.
<date><hostname> mach: available memory = 7.14 megabytes.
<date><hostname> mach: using 16 buffers containing 0.12 megabytes of memory
```

you probably have a corrupted kernel on the hard disk. If so, typing Command-Command-tilde at this point will probably do nothing, instead of getting you back to the ROM monitor as it usually would.

If you have a corrupted kernel, it was probably caused by someone trying to run /sdmach (either by typing its name in a shell or by double-clicking on the file in the Workspace Manager).

The easiest way to recover is to boot off of an optical (or off the net, if you're in an environment with networked NeXTs), mount the hard disk, and restore the sdmach file. You'll also want to make /sdmach non-executable to prevent the bug from recurring. Follow these instructions:

[1] Boot off of a release1.0 optical disk that doesn't exhibit the problem:

```
NeXT> bod
```

[If booting off the network, type “ben” instead of “bod” and replace “optical” with “net” in the following instructions.]

[2] Log in, start a Shell window (launch /NextApps/Shell), and su as needed to get

root privileges:

```
localhost> su
Password:
localhost#
```

[3] Mount the hard disk as follows:

```
# mount /dev/sd0a /mnt
```

[4] Fix the protections on the optical so that this won't happen again:

```
localhost# chmod a-x /sdmach
```

[5] Copy the kernel from the optical to the hard disk. (The -p switch preserves

the dates.)

```
# cp -p /sdmach /mnt/sdmach
```

[6] Reboot off the hard disk.

[7] If you have other opticals you want to protect, do the chmod step on these too. (See the second question, below.)

Note: These instructions assume your problem is booting from the hard disk. If the problem occurs when booting from an optical and you have a good hard disk or a NeXT network, adjust the instructions accordingly. If you have an optical-only, standalone system, you'll have to copy from a good optical to the bad one; please see the instructions in the User's Manual, Chapter 8, pages 138-139.

Q: [From above] I heard that there's a bug that corrupts the kernel. It hasn't affected me yet, but how do I keep this from happening to my machine?

A:

[1] Log in, start a Shell window (launch /NextApps/Shell), and su as needed to get

root privileges:

```
localhost> su
Password:
localhost#
```

[2] Fix the protections on the disk so that this won't happen again:

```
localhost# chmod a-x /sdmach
```

(Note that “localhost#” represents the prompt; type only what follows the prompt.)

[3] Do the same for each optical you have that has the 1.0 system software on it. Insert the optical, remember where it's automounted (e.g. ~/OpticalDisk), and in the same Shell window su'd to root, type:

```
localhost# chmod a-x ~/OpticalDisk/sdmach
```

(Again, don't type the “localhost#”, and replace “~/OpticalDisk” with the pathname of the optical.) Unmount the optical and repeat step [3] for any other opticals.

Q: It seems like the scrollView ignores the frame.origin of its docView. Also, does a ScrollView clip to its update rect(s)?

A: A ScrollView translates the coordinate system of its contentView in response to user movement of the scrollers. Since the frame of the docView is defined in the coordinate system of the contentView (i.e. it's a subview of the contentView), translating the coordinate system of the contentView has the effect of moving the frame, and hence the visible portion of the docView.

When you do a setDocView: the system translates the coordinate system of the contentView so that the contentView's bounds.origin is the same as the frame.origin of the docView. For example: if the frame.origin of your docView is {100.,100.}, the bounds.origin of the contentView will be {100.,100.} as well.

So the ScrollView does pay attention to what the frame.origin of the docView is, but it doesn't really matter what it is, at least initially.

No, the drawing is not clipped to the update rects; however, you can restrict the area being redrawn yourself by drawSelf:: method you can use PSrectclip() or NXRectClip().

FINIS