Newsletter of the NeXT Programmer's Community

NeXT Users' journal

JUNE 1990 • ISSUE 7

WELCOME

Erica J. Liebman

Gak. Another issue, another set of finals. At least that's all for the spring quarter. Now it is time to kick back, smell roses and remind y'all Znugies to send in some articles. Ah, summer.

I'm pretty happy to note that we've crossed the six-month line on getting out issues. BuzzNUG has existed for nearly nine months andseems pretty hale. All for the good.

I just sent out about three-score ID card numbers and am processing the next batch. Hope you guys are enjoying making the ID's. If you have any anecdotes about printing on file folders, drop me a line. I could use a good chuckle just now, recovering as I am from "post-final-phrenzia".

The SIGs have started off pretty nicely. This issue, we have a double-helping of Hardcore and Medical SIG reports. Contact names are included in the articles if you want to join.

Against better judgement, I've been continuing to evolve the "look" of Next Users' Journal by including some clip art. I am hoping (praying, fingers crossed, etc) that this will not cause more printer mortality. One of our most problematic, well, problems has been print and version errors arising from using Framemaker.

The problems have tended to be threefold : aforesaid printer errors, hardships of reading double columns of *very* small text and exporting issues into Digital Librarian.

Printer errors have been shown to be a little bit avoidable by printing only a few pages at a time and avoiding pictures wherever possible. (Surprisingly no one has had problems with Frame-specific clip-art, so I'm taking the chance on including these in the new column layouts).

For those who have problems with multi-column output, I suggest (a) you kill a tree or seven and print this newsletter directly to paper or (b) you create a single column stationary form and copy/paste the text from this multi-columned version to the single columned version. I have an open offer to anyone who can give me a better one-column layout than my current two-column one : I will switch to the (better) one-column version if you submit it to me with all the appropriate paragraph formats, master pages, pull-quotes, bar-lines and so forth.

As to the last problem, according to my documentation for Framemaker 2.0a, this newsletter **is** eompatible with Digital Librarian. I haven't tried it myself yet (no time at end of quarter), but doubt that Frame would be totally inaccurate in this regard. Again, if anyone out there has got this working, lets hear from you.

A big problem, you may have noticed is that the delivery date for each issue has been floating precariously forward from the fifteenth of each month. It is vitally important that you **deliver** your articles when promised. I count on *every single one of you* to submit articles & get them in on time. This is a group effort! Let's keep it going. Thanks again to everyone who has helped contribute said articles.

Finally, and importantly, if you want to see NeXT Users' Journal go "paper", please contact NeXT and let them know. Without groundswell support, (and the conviction that this move will help sell more machines), NeXT is not going to have incentive to help out.

Go get 'em. -- EJL.

INDEX

Contributors:

Erica J. Liebman,Ian Smith, Keith Edwards, Kris Jensen, Doug Kieslar, Bruce Henderson, Scott Hess, Joel McClung, Mike Dixon, Dave Joerg, Jacob Gore, Terrence J. Talbot, Bill Barker, Conrad Geiger

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

HEADLINE NEWS

• Stuart 1.0 now at FTP sites.

• HSD begins higher-ed pricing program.

• Ariel Digital Microphone now shipping

• Ariel QuintProcessor now shipping

• MediaStation version 1.11 now shipping

• Jacob is now Dr. “that means sir!” Gore. CONGRATS! Says he “ I just came back from Northwestern. I'm done!!! ("Wehhlcome to Chanel 31 Late Night Horror Movie... This is your host, Dr. Gore...":-)”

•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/or obscenity 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. Use blank lines to denote sections.

•Start sections on first line after section header.

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

This issue of NeXT Users’ Journal was prepared in FrameMaker, generously donated by Frame.

FEEDBACK FROM THE TRENCHES

••I would like to make a few comments on Dick Silbar's article in issue 6. The author laments the current lack of a NeXTStep interface for GNU Emacs. I have written such a beast--it is available in the NeXT archives under the name "Emacs1.0.tar.Z" and was mentioned in another article in the same issue. The interface provides rudimentary mouse support and uses the "Alternate" key as Meta.

When C-s is not being passed through to GNU Emacs, the keybinding C-x s (save-some-buffers) may be useful. It is certainly safer than C-x C-w or C-x C-c.

The global-set-key commands in the article also contained some serious typographical errors. I suggest the following replacements:

(global-set-key "\e." 'set-mark-command)

(global-set-key "\C-\\" 'isearch-forward)

(Note to editor: Please double-check the spelling of my

last name--it was botched in issue 6.) --*John G. Myers*, jgm@fed.expres.cs.cmu.edu

Dick's article was meant for an earlier issue (before the advent of Emacs 1.0) and was mislaid due to an oversight on ***my*** *part . We look forward to seeing more about Emacs1.0 -- hopefully about its development as well as hints for using it. - EJL*

•• I just downloaded #6 from j.cc.purdue.edu and printed it on the Data Products Laser printer on our VAX. Funny thing - it stopped printing after page 25! I tried to 'vi' the file, but it said that there were lines that were too long, so I couldn't strip out pages 1 - 25 to print the rest of the document. Has anyone else had a problem with the .ps file? - *Mike Brown*

Well, yes. A lot of people have been having problems with postscrip files ever since NUJ switched to using Frame. Our version of Framemaker is 2.0a. I have posted a second copy of issue #6 without pictures to the archives. We're working on the problem. Have patience. In the mean time, try printing only a few pages (say 5 - 10) at one go. Let me know if anyone out there has any better ideas or fixes. -- EJL.

•• I am Tully Hammill hammill@milton.u.washington.edu. I would like to subscribe to a paper copy of Buzz when you get it going.

Tully, and all the others of you who have written to me about "going paper", it is vitally important that you contact NeXT and let them know that you are interested in seeing NUJ in paper subscription. NeXT & I have tossed some ideas around, but unless they see groundswell support (and maybe some concrete reasons why this will help sell machines), they're not going to have any incentive to help out on the paper-issue. -- EJL

••I made a few boo-boos in the NeXT Users' Journal May 1990 issue 6 that I would like to clear up now.

In Buzz's Hint Corner: the 'open' command will simulate a double-click on the file in the Workspace manager. The 'openfile' command will tell Edit to open the named file(s), but only if Edit is already running.

In demandAppBitmap:, right before [appBitmap free], there should be something like:

```
*len = appBitmapSize*2;
```

In the Speaker/Listener article: the code fragment in the section 'Example: Sending and Receiving Byte Arrays' has a problem. What is:

```
- demandAppBitmap:(char **)bitmapData length:(int *)len
{
  id appBitmap;
  *bitmapData = malloc(2*2*48*48);
  <comments deleted>
  appBitmap = [Bitmap findBitmapFor:"app"];
  [appBitmap readImage:*bitmapData withAlpha:*(bitmapData +
2*48*48)];
  [appBitmap free];
  return 0;
}
```

should be:

```
- demandAppBitmap:(char **)bitmapData length:(int *)len
{
  id appBitmap;
  int appBitmapSize, width, height, bps, spp;
  appBitmap = [Bitmap findBitmapFor:"app"];
  [appBitmap imageSize:&appBitmapSize width:&width height:&-
height
            bps:&bps spp:&spp inRect:NULL];
  if ( (width!=48) || (height!=48) || (bps!=2) || (spp!=2) )
    return -1;
  *bitmapData = malloc(2*appBitmapSize);
  [appBitmap readImage:*bitmapData withAlpha:*(bitmapData +
appBitmapSize)];
  [appBitmap free];
  return 0;
}
```

of course, the technique described here still won't work for bitmaps that aren't 2 bps, 2 spp, and 48-by-48, but in the older demandAppBitmap:length: eight times too much space would be allocated, and the alpha data got put in the wrong place. *David S. Joerg, MindShock, Inc., Internet : joerg@alliant.mcs.anl.gov*

••Nice issue. Just wanted you to caution authors not to use dark shading such as on pages 7,8. The text inside the shaded rectangle is not legible.--*Roger Kirchner*

Good point! -- EJL

••I thought the two column format was ok to read online, but not great. The two column format was definitely better to read on paper, though. Maybe there could be a collection of the technical articles in WriteNow or just ascii text format for those who want to put them in a library. I will volunteer to do this, if people are interested.

All in all, you are doing a fantastic job. Your mother should be proud.--*Charles Neveu*

*Charles - **Yes**, please! We would all greatly, **greatly**, appreciate this gesture! -- EJL*

•• Some NeXT products I've run accross that are not listed in the NeXT User's Journal, issue 6...

Objective DB Toolket - 30 classes to links NextStep to Sybase. Professional Software, Inc./Lakeside Office Park,/599 North Avenue - Door 7/Wakefield, MA 01880/(617)246-2425

I know nothing about it, ad was seen in UNIX Today!

uni-REXX and uni-XEDIT - Unix version of IBM's mainframe/procedural language and general purpose editor./The Workstation Group, wrk/grp/6300 N. River Road/Rosemont, IL 60018/(708) 696-4800

NeXT version of these products should be shipping before your next newsletter. REXX is well known to people coming from the IBM VM and MVS world, and has also become well know to the Amiga world. . A version is also available for MS-DOS. REXX is a replacement for shell script programming language. As more user-friendly compared to UNIX standard scripts as NextStep is over X-Windows. XEDIT is the standard editor for IBM's VM on mainframes. I have both of these on order. -- *Paul Kunz*

••What's missing from the Example code distributed with the cube? What can we do to make it better? Help us improve the Examples by sending me your comments. Thanks! *__jayson adams*, NeXT Technical Support, Jayson_Adams@next.com

••In your article about Convex Hull generation in the March issue, you mentioned that you had to create your own node type because list would only take type id. Why not use the Storage class instead?

It seems to do the same thing as a list, except that its elements don't have to be arrays. I'm kind of new at NeXT stuff, so maybe I'm missing something here, but I was just wondering. (BTW- your program worked beautifully -- the first IB program I've ever entered in that compiled and ran on the first try!) Thanks! *Laura Fischer* (lfischer@previous.Princeton.edu)

*You are, of course, totally correct. I didn't even **know** about the storage class before you letter -- Thanks. - EJL.*

••What's this talk of membership cards? I don't get it. Why do I want one? Maybe there's something I'm missing. Discounts at Businessland? Admission to special secret meetings? Or some loony fraternity joke? *Jeff Adams*, adams@ucscl.ucsc.edu

You get a warm fuzzy feeling for helping me (a) organize my files better and (b) perform demographics on the group to get some funding and/or software for review. All in all, you get a spiffy ID card and I get some help. -- EJL.

NeXT on Campus Subscriptions

Jeff Wishnie is helping Ron Weissman & Patty Kammerer to organize the new quarterly NeXT Newsletter "Next on Campus". Subscriptions are free and you can get one easily. It's a great read -- I'm dying to get some of the people featured in N.o.C. to write some articles for NUJ. Let's cross our fingers. Anyway, if you want a copy, they'd like you to respond with at least the following information (send info to NeXT on Campus, NeXT, Inc., 900 Chesapeake Drive, Redwood City, CA 94063)

• Name:

• Title:

• Institution Name:

• Department:

• Address:

• City, State, Zip and Phone:

•Do you currently use a NeXT computer? (yes or no)

• Are you: [you can indicate more than one] Faculty/ Staff/Student/Developer/Reseller/Support Center/Service Center

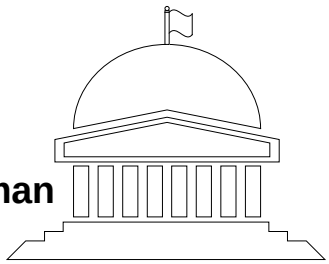
• Would you like to receive the current issue of NeXT on Campus?

•℃Second: Where did you first hear about NeXT on Campus (e.g. Usenet)?

•What types of articles would you like to see in the fall issue?

Your suggestions and subscription information can also be sent to NeXT on Campus electronically at next_on_-campus@next.com. Tell 'em Buzz sent ya! :)

Government Watch



coordinated by
Erica J. Liebman
erica@kong.gatech.edu

Each Month, we will focus on a different government contractor, allowing them space to describe their IR&D efforts, target customers and general efforts with respect to NeXT-based applications. Our first Government Contractor is ESL, followed in the next months by Compus, ADT and possibly Hughes.

ESL Serves Intelligence Community

Gary W. Fuller

ESL Incorporated of Sunnyvale, CA signed the first Federal System Integrator (FSI) Agreement with NeXT, Incorporated on 20 December 1989. As an FSI, ESL develops custom systems for its intelligence and reconnaissance community customers, and incorporates NeXT Computers as part of the system when appropriate. ESL does not develop commercial products.

ESL was first drawn to NeXT in the summer of 1988, when then ESL President Bob Kohler met with NeXT President Steve Jobs. During their discussions and a demo of the soon-to-be introduced NeXT machine, Kohler and Jobs discussed the similarities between the needs of an intelligence analyst and those of higher education. It was clear from this discussion that much of what NeXT offered--great user interface, effective development environment, vast data storage, text handling, etc.--were just what the analyst needed.

Discussions with NeXT continued throughout 1989, culminating in the signing of the agreement in December.

ESL believes that their strategic alliance with NeXT gives the intelligence and reconnaissance communities the best of both worlds--commercial and custom software and hardware. Combining NeXT products with ESL's custom software and hardware produces systems that allow the user to integrate text, graphics, images and signals--for effective multi-source data fusion.

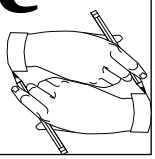
Recently, ESL has developed a prototype application which handles large volumes of message traffic [cables from overseas]. Incoming messages are first processed through a "Profiler" which determines the appropriate inboxes based on message content. The user can then view the contents of the inbox, file messages in nested folders, perform searches, and perform a variety of hypermedia functions, including annotation.

ESL is a full-service integrator, and thus conducts initial studies and analyses, performs system engineering and design, and develops custom hardware and software.

For further information contact Gary Fuller, ESL Workstation Products Manager at (408) 752-2525.

SIG HardCore

moderated by
Ian Smith
iansmith@warhol.gatech.edu



"Sockets: A quicker and dirtier primer"

by Ian Smith

Welcome to the initial installment of the the NeXT Hardcore SIG in the NeXT User's Journal! NeXT Hardcore is a SIG for people interested in systems/low-level programming on the NeXT workstation. It is the intention of the SIG to provide useful programming information for people who wish to create better tools for the NeXT and its environment. It is hoped that all NUJ readers can benefit from the information provided by the SIG.

In this first column of NeXT hardcore, I'll be discussing how to quickly create network interfaces for your programs. The primary mechanism for doing this is, of course, is a Objective C object called the TCP Network object. This object allows you to create a TCP network connection between two machines that are directly internet addressable. Each machine instantiates an object of this type, and then sends it messages regarding actions to perform. Currently supported are the sending of text (ascii), data, and files via the object.

"The title of this article?" you ask? Well, this article is based on another article written a couple of years back on sockets, by Jim Frost an employee of Saber Software. Via this paper many people (include the author) got their first taste of Inter Process Communication (IPC) and the BSD abstraction of "sockets." (This article was called, "Sockets: a Quick and Dirty Primer." and is still posted occasionally to newsgroups like comp.unix.questions or comp.unix.wizards.)

As the title implies, the critical implementation detail of this object is the "socket." Sockets are a Berkelyism used to make IPC easier. In general, sockets are a file-descriptor upon which most file-descriptor operations are valid. However, there are also special operations that can be performed on a socket to do communication, ie. bind, accept, and listen. Once a socket has been bound to (established on) a internet port it can accept or create connections from or to remote machines who have sockets bound to the same port. It does this via the accept and connect system calls. Both of these calls return an integer a file descriptor to read and write data to the remote machine. In fact, a socket can multiplex several connections over the same port via multiple calls to accept and connect.

What I have done is create another level of abstraction on top of sockets with the Objective-C object "network_tcp." The tcp is the connection type. TCP connections provide reliable connection based service between internet hosts. TCP was chosen over UDP or raw mode due to the ease of use and the reliable transport. (Hardcore hackers out there should have no trouble porting these ideas to UDP or raw mode.) These objects can be thought of black boxes that handle all the network details for you. However, I would rather see them viewed as building blocks for more sophisticated network tools or new/better implementations of existing ones. They also can be used for prototyping network applications before all the networking details are worked out.

This object allows you to create a TCP network connection between two machines that are directly internet addressable.

The objects as presented here, support the following methods:

```
- (int) initialize;
- (int) setup:(int)port;
- (int) receive;
- (int) call:(char *)target onport:(int)port;
- (int) send_text:(char *)string;
- (int) send_data:(void *)data ofsize: (int) size;
- (int) send_file:(char *)fname;
- (int) recv_text:(char *)string;
- (int) recv_data:(void *)data ofsize: (int *)size;
- (int) recv_file:(char *)fname;
- (int) call_waiting;
- (int) data_waiting;
- (int) close_tcp;
```

These function as follows:

```
- (int) initialize;
```

This method is called automagically by the objective c runtime system, and its only function is to initialize to zero the instance variables of the object. - (int) setu-

p:(int)port;

Setup takes a port number and will return a 1 if it can successfully configure itself to listen on that port for incoming connections. Note: Port numbers below 1024 are reserved for system use and are generally used for system daemons that need to listen on well-known ports.

```
- (int) receive;
```

This method receives an incoming connection on a port. If no incoming call is present, it will wait to return until one is present. (If you don't want the process to block, use the "call_waiting" method provided to determine when to receive a connection.) - (int) call:(char *)target onport:(int)port;

You call this method to inform the tcp_network object to call another machine on the specified port. The target machines name must be known to the machine that is making the call.

You call this method to inform the tcp_network object to call another machine on the specified port. The target machines name must be known to the machine that is making the call.

```
- (int) send_text:(char *)string;
- (int) send_data:(void *)data ofsize: (int) size;
- (int) send_file:(char *)fname;
- (int) recv_text:(char *)string;
- (int) recv_data:(void *)data ofsize: (int *)size;
- (int) recv_file:(char *)fname;
```

These methods perform exactly like you would expect them to. Each must be used with its mate, however, to perform their function successfully. The text methods work with null terminated strings as input, whereas the data methods work with binary-type data. Be careful that you pass recv_data a pointer as the second argument, so you can determine how many bytes were transmitted to you. The file transfer methods work with either full pathnames or names relative to the current directory. They also use null terminated strings as arguments. The recv_file method places the file it receives into the filename that is its argument.

```
- (int) call_waiting;
```

This method returns a 1 when there is call waiting to be received on the port that the object is listening on. It returns a 0 otherwise.

```
- (int) data_waiting;
```

This method is similar to call_waiting, in that it returns 1 when there is data pending to be read on a connection, 0 otherwise.

```
- (int) close_tcp;
```

You need to perform this method every time that you use a network_tcp object, after you are done with it, otherwise problems may arise. Due to the fact that tcp is a reliable protocol, data will remain in i/o buffers until delivered or the kernel decides that the data cannot be delivered. If you do not perform this method, there may be data left in buffers, and then when you try to re-establish control of the port (ie, running the program again) the system will not let you do so because it has data yet to be delivered on that port. I attempted to right this problem with the use of some socket options, but it still persists in some degree at present. Note: I did not implement any type of tcp multiplexing, as to do this properly would take considerably more space than can fit into this column in the NUJ. If you are interested, contact the NeXT hardcore mailing list at: hardcore-request@warhol.gatech.edu for more information.

I have written a couple of sample programs to show you how to use this class and its methods. They are : trans_tcp.m and rec_tcp.m. To compile them use:

```
%cc trans_tcp.m -o trans_tcp -lsys_s -lNeXT_s
%cc rec_tcp.m -o rec_tcp.m -lsys_s -lNeXT_s
```

You will need to make sure that you modify the machine name in trans_tcp.m to reflect the machine you want to call. After this just run the program rec_tcp on the machine you want to receive the connection, and trans_tcp on the machine you want to initiate the connection.

Be careful that you pass recv_data a pointer as the second argument, so you can determine how many bytes were transmitted to you.

Here are the files:

```
/* TCP Network Objects -- An Example of a Transmitter (trans_tcp.m)

   by Ian Smith

   of The Software Engineering Research Center's Multimedia Lab
*/


#import "tcp.m"

main ()

{
```

```
  id foo;
  int rt,mydata=3325;
  char c;

  foo = [network_tcp new];

/* substitute the receiving machine you wish to use below...
   it must directly internet addressable and have a well-known
   name to the host using the object
*/
  rt = [foo call:"escher" onport:3325];

  if (rt !=1) exit(1);
  rt=[foo send_text:"foo bar gronk fronk"];
  if (rt!=1) exit(1);
  printf("press <return> to continue...\n");
  scanf("%c",&c);
  rt=[foo send_data:(void *)&mydata ofsize:sizeof(int)];
  if (rt!=1) exit(1);
  rt=[foo send_file:"/etc/hosts"];
  if (rt!=1) exit(1);
  rt=[foo close_tcp];
}

/* TCP Network Objects -- An example of a receiver
   (rec_tcp.m) by Ian Smith of The Software Engineering
   Research Center's Multimedia Lab
*/

#import "tcp.m"
main()
{
  id bar;
  int rt,hisdata,size;
  char buff[80];

  bar = [network_tcp new];
  rt=[bar setup:3325];
  if (rt!=1) exit(1);
  while (![bar call_waiting]) {
    sleep(2);
    printf("waiting...\n");
  }
  rt=[bar receive];
  if (rt!=1) exit(1);
  rt=[bar recv_text:buff];
  printf("%s\n",buff);
  while (![bar data_waiting]) {
    sleep(2);
    printf("waiting for data...\n");
  }
  rt=[bar recv_data:&hisdata ofsize:&size];
  if (rt!=1) exit(1);
  printf("he sent %d which is size:%d\n",hisdata,size);
  rt=[bar recv_file:"local.file"];
  if (rt!=1) exit(1);
  printf("received the file.\n");
  [bar close_tcp];
}

/* TCP Network Objects -- Methods (tcp.h)
   by Ian Smith of The Software Engineering
   Research Center's Multimedia Lab
*/

@interface network_tcp:Object
{
  int sock;
  int fd;
}

- (int) initialize;
- (int) setup:(int)port;
- (int) receive;
- (int) call:(char *)target onport:(int)port;
- (int) send_text:(char *)string;
- (int) send_data:(void *)data ofsize: (int) size;
- (int) send_file:(char *)fname;
- (int) recv_text:(char *)string;
- (int) recv_data:(void *)data ofsize: (int *)size;
- (int) recv_file:(char *)fname;
- (int) call_waiting;
- (int) data_waiting;
- (int) close_tcp;

@end

/* TCP Network Objects -- Implementation (tcp.m)
   by Ian Smith
  of The Software Engineering Research Center's Multimedia Lab
*/

/* credit where credit is due: I must give a lot of the credit
for these objects to Jim Frost of saber software. A long time
ago, I read a paper called "Sockets: a Quick and Dirty Primer"
by an employee of that company and got some of the information
and motivation for this code.  In fact, some of this code may be
verbatim from the paper, but I don't know as I have hacked up
this so many times that it is hard to tell what is from where.
Also, to the people who wrote the BSD supplemental papers, thank
you, I'm in your debt. Nobody learns IPC without IPC primers from
the berkely docs.  A thank you to Keith Edwards for his infor-
mation and encouragment, without which nothing would get done.
Also, thanks to Henry Strickland for some info on socketops.  */
```

```
#import <objc/objc-runtime.h>
#import <objc/Object.h>
#import "tcp.h"
#import <stdio.h>
#import <sys/types.h>
#import <sys/time.h>
#import <sys/socket.h>
#import <string.h>
#import <netdb.h>
#import <netinet/in.h>
#import <errno.h>
#import <stdlib.h>

#define MAXHOSTNAME 64
#define FLAG 10000
#define BUFSIZE 1024

@implementation network_tcp

- (int) setup:(int) port;
{
char myname[MAXHOSTNAME+1];
    int s,len,one=1;
 struct sockaddr_in sa;
 struct hostent *hp;
        struct linger ling;

bzero(&sa,sizeof(struct sockaddr_in));
gethostname(myname,MAXHOSTNAME);
hp= gethostbyname(myname);
if (hp==NULL) {
  fprintf(stderr,"tcp error:gethostbyname\n");
  return(-1);
}
sa.sin_family=hp->h_addrtype;
sa.sin_port=htons(port);
if ( (s=socket(AF_INET,SOCK_STREAM,0)) < 0) {
  fprintf(stderr,"tcp error:socket\n");
  return(-1) ;
  }
if (bind(s,(struct sockaddr *)&sa,sizeof(sa)) < 0) {
  fprintf(stderr,"tcp error:bind\n");
  return(-1);
}
listen(s,5);
ling.l_onoff=0;
ling.l_linger=0;
        len=sizeof(ling);
        setsockopt(s,SOL_SOCKET,SO_LINGER,(char *)&ling,len);
len=sizeof(int);
setsockopt(s,SOL_SOCKET,SO_REUSEADDR,(char *)&one,len);
sock = s;
return(1);
}

- (int) receive;
{
struct sockaddr_in isan;
int i,t,rt;



i=sizeof(isan);
getsockname(sock,(struct sockaddr *)&isan,&i);
if ((t=accept(sock,(struct sockaddr *)&isan,&i)) < 0)  {
  fprintf(stderr,"tcp error:accept\n");
  return(-1);
}
fd = t;
return(1);

}
- (int) call:(char *)target onport :(int) port;
{
  struct sockaddr_in sa;
  struct hostent *hp;
  int a,s,f,rt;
  char buff[80];

  if ((hp=gethostbyname(target))==NULL) {
      errno=ECONNREFUSED;
      fprintf(stderr,"tcp error:gethostbyname\n");
      return(-1);
    }
  bzero(&sa,sizeof(sa));
  bcopy(hp->h_addr,(char *)&sa.sin_addr,hp->h_length);
  sa.sin_family=hp->h_addrtype;
  sa.sin_port=htons((u_short)port);
  s=socket(hp->h_addrtype,SOCK_STREAM,0);
  if ( s < 0 ) {
    fprintf(stderr,"tcp error:socket\n");
    return(-1);
  }
  if (connect(s,(struct sockaddr *)&sa,sizeof(sa)) < 0) {
      shutdown(s,2);
      fprintf(stderr,"tcp error:connection refused\n");
      return(-1);
    }
  fd =(int) s;
  return(1);
}
```

```
- (int) send_text:(char *)string;
{
  int ln,err;

  ln=strlen(string)+1;
  err=write(fd,&ln,sizeof (int) );
  if (err!=sizeof(int)) {
    fprintf(stderr,"tcp error:write\n");
    return(err);
  }
  err=write(fd,string,ln);
  if (err!=ln){
    fprintf(stderr,"tcp error:write\n");
    return(err);
  }
  return(1);
}
- (int) send_data:(void *)data ofsize: (int) size;
{
  int err;
  err=write(fd,&size,sizeof(int));
  if (err!=sizeof(int)) {
    fprintf(stderr,"tcp error:write\n");
    return(err);
  }
  err=write(fd,data,size);
  if (err!=size) {
    fprintf(stderr,"tcp error:write\n");
    return(err);
  }
  return(1);

}


- (int) send_file:(char *)fname;
{
  char buf[BUFSIZE];
  int ln,done=0,err;
  FILE *fp;

  fp=fopen(fname,"r");
  if (fp==NULL) {
   fprintf(stderr,"tcp error:unable to open file %s \n",fname);
    return(-1);
  }
  fseek(fp,0L,SEEK_SET);
  while (!done) {
    ln=fread(buf,1,BUFSIZE,fp);
    if (feof(fp)) done=1;
    else done=0;
    err=[self send_data:(void *)buf ofsize:ln];
    if (err==-1) {
      fprintf(stderr,"tcp error:send_data in file transfer\n");
      return(-1);
    }
    err=[self send_data:(void *)&done ofsize:sizeof(int)];
    if (err==-1) {
      fprintf(stderr,"tcp error:send_data in file transfer\n");
      return(-1);
    }
  }
  fclose(fp);
  return(1);
}
- (int) recv_text:(char *)string;
{
  int err,ln;

  err=read(fd,&ln,sizeof(int));
  if (err!=sizeof(int)) {
    fprintf(stderr,"tcp error:read\n");
    return(err);
  }
  err=read(fd,string,ln);
  if (err!=ln) {
    fprintf(stderr,"tcp error:read\n");
    return(err);
  }
  return(1);
}
- (int) recv_data:(void *)data ofsize: (int *)size;
{
  int err,amt;

  err=read(fd,&amt,sizeof(int));
  if (err!=sizeof(int)) {
    fprintf(stderr,"tcp error:read\n");
    return(err);
  }
  err=read(fd,data,amt);
  if (err!=amt) {
    fprintf(stderr,"tcp error:read\n");
    return(err);
  }
  *size=amt;
  return(1);
}
- (int) recv_file:(char *)fname;
{
```

```
  int ln,done=0,err,tmp;
  char buf[1024];
  FILE *fp;

  fp=fopen(fname,"w");
  if (fp==NULL) {
    fprintf(stderr,"tcp error: unable to open file for writ-
ing\n");
    return(-1);
  }
  fseek(fp,0L,SEEK_SET);
  while (!done) {
    err=[self recv_data:(void *)buf ofsize:&ln];
    if (err==-1) {
     fprintf(stderr,"tcp error: recv_data in file transfer\n");
      return(-1);
    }
    err=[self recv_data:(void *)&done ofsize:&tmp];
    if (err==-1) {
     fprintf(stderr,"tcp error: recv_data in file transfer\n");
      return(-1);
    }
    err=fwrite(buf,1,ln,fp);
    if (err==0) {
      fprintf(stderr,"tcp error: unable to write to disk\n");
      return(-1);
    }
  }
  fclose(fp);
  return(1);
}

- (int) call_waiting;
{
  fd_set fds;
  struct timeval to;
  int n;

  to.tv_sec=0;
  to.tv_usec=0;
  FD_ZERO(&fds);
  FD_SET(sock,&fds);
  n=select(FD_SETSIZE,&fds,0,0,&to);
  return(n);
}
- (int) data_waiting;
{
  fd_set fds;
  struct timeval to;
  int n;

  to.tv_sec=0;
  to.tv_usec=0;
  FD_ZERO(&fds);
  FD_SET(fd,&fds);
  n=select(FD_SETSIZE,&fds,0,0,&to);
  return(n);

}

- (int) initialize;
{
  fd=0;
  sock=0;
  return(1);
}

- (int) close_tcp ;
{
  if (fd!=0) close(fd);
  if (sock!=0) shutdown(sock,2);
  return(1);
}
```

--ian

iansmith@warhol.gatech.edu

READING NEXT MAIL MESSAGES ON NON-NEXT SYSTEMS

Keith Edwards

Software Engineering Research Center / Multimedia Group, Georgia Tech, Atlanta, GA 30332-0280, <keith@dali.gatech.edu>

Introduction

Although the NeXT mail system has many features to be admired, there are still many of us who do day-to-day work on machines that do not have the capability to play back NeXT multimedia mail messages. It is often very troublesome to have to resend your mail to a NeXT machine where it can be disected and played by the NeXT mail system.

While not a complete solution, this article presents a quick work-around. The software presented here will allow users to disect a NeXT mail message (complete with

attachments) and display the ASCII portions of that on the terminal. In addition, software is included for stripping off the NeXT soundfile header information so that CODEC sound data may be played on a Sun SPARCstation (or any other workstation that uses 8-bit muLaw encoding of sound).

First, lets delve into some history about mail systems and the mechanics of how the NeXT mail system works.

A Short History

One of the problems facing the designers of the NeXT mail system was that it needed to be compatible with existing mail systems in the transport mechanisms used to carry the mail between machines (or hosts). This is so that users on Sun workstations can send mail to users on NeXT workstations and vice versa. Note that if you send a simple NeXT message (with no font information or inclusions (``attachments", in the NeXT parlance)), then the mail is easily readable on any machine. It is only when one makes use of the ``special" features of the NeXT mailer that compatibility is lost.

For several years the Unix community has standardized on a protocol called SMTP, for Simple Mail Transfer Protocol (the ``Simple" in the name is something of a misnomer). SMTP was first defined by David H. Crocker at the University of Delaware in 1982 and is usually implemented on Unix systems by a program called ``sendmail.". One of the primary restrictions on the protocol is that it is impossible to send binary data via SMTP: only ASCII transfer is supported. Thus any multimedia mail system which wishes to be compatible with the rest of the world by using sendmail and SMTP must convert the data to be transfered to ASCII.

Enter NeXT.

An Overview of NeXT Mail

The NeXT workstation performs several steps to prepare mail messages before they are sent. First, the different components of the message are bundled together via the Unix program tar. Tar creates a single file which is an archive of all the various components of the mail message. One of these, called index.rtf, is present in every multimedia mail message and is special. We will discuss index.rtf in more detail shortly.

After this bundling process, the archive file contains binary data. Thus it is necessary to convert it to ASCII prior to shipping it out. The conversion to ASCII greatly increases the size of the file, however, so the next step of the preparation process is to run the Unix compress program on the archive. Compress is quite effective. Text will often be compressed by as much as 50-60%. Binary data is compressed somewhat less.

As the final step we convert our compressed archive to ASCII via the program uuencode. Uuencode performs a mapping from binary to ASCII on the file so that we may send it via SMTP. Generally the uuencode process seems to increase the size of mail message by roughly 50%. At this point the mail is ready to send.

For several years the Unix community has standardized on a protocol called SMTP, for Simple Mail Transfer Protocol (the ``Simple" in the name is something of a misnomer).

But how is the actual message arranged in this archive? Earlier we mentioned the index.rtf file. This file is included in every archive and is essentially the ``table of contents" for the message components. The .rtf extension means that the file is in the Rich Text Format, a file format for document interchange specified by Microsoft. Index.rtf contains all the textual data of the message, along with formatting commands and font and color information. Index.rtf also contains the filenames of the various attachments (such as voice mail clips, sounds, pictures, and the like) that are present in the archive.

Voice mail attachments are included in the message archive with the filename extension .vox. Other attachments (e.g., icons dropped into the mail window) keep their previous filename extensions.

Extracting the Information

Obviously if we wish to be able to read at least some of a NeXT mail message on a non-next machine, we need to be able to perform the inverse of this packaging process. Fortunately, the NeXT mail designers used standard Unix commands to prepare mail messages, so the reverse process is trivial. A shell script is presented that will take a NeXT mail message and perform the unpacking process on it.

But what about reading the RTF data, index.rtf? This file contains the textual information in the mail and in most mail messages (even with the NeXT mail system) the text component is the most important. RTF is quite complex but fortunately pretty much all we have to do to be able to view the text contained in index.rtf is strip out the RTF formatting instructions. RTF is an ASCII format

so there are many Unix tools available to us to do this task.

I have chosen to use Lex to strip out the RTF information. Lex is a tool for building lexical analyzers, and is primarily used by people writing parsers for simple grammers (such as some of the various Unix startup files). Another tool, called Yacc (for Yet Another Compiler Compiler) is more powerful and actually probably better suited for parsing the RTF language. Lex was used primarily for two reasons: (1) it's simpler, and (2) we're not doing any real semantic analysis of what the RTF information in index.rtf *means*, we're just stripping it out. For this Lex is sufficient.

Presented is a Lex grammar file called dertf.l, which should strip out most RTF information in a file. When references to attachments are found, it will print out a line like:

```
[ Attachment inserted here:  VoiceMail_keith0.vox ]
```

The author makes no claims of this grammar's completeness or accuracy.

Also included is a shell script called ``denext" that will perform all the unpacking operations, strip out the RTF information, and search for .vox (voice mail attachments) and play them (on a Sun SPARCstation) if found. The shell script makes use of a C program called stripnextsnd which strips off the SNDSoundStruct header from NeXT sound files. This program is not really necessary. If you don't want to use it you can simple change the shell script so that it cats the file onto /dev/audio.

As I said before, it's only a simple workaround, but it does allow you to get most of the information content out of most NeXT mail messages. What is needed now is a truly robust, extensible, multimedia mail system that can interchange multimedia mail messages between NeXTs and other systems. We are building such a system and hope to have prototypes available in a few months.

References

"RFC822: Standard for the Format of Arpa Internet Text Messages," David H. Crocker, University of Delaware. August 13, 1982.

"Microsoft Rich Text Format Specification," Microsoft Corporation. 1989.

Lex Grammar

```
/*
 * A simple Lex grammar for Microsoft RTF.
 *
 * Copyright (c) 1990, Keith Edwards
 * May be freely used, modified, distributed, or copied as long as
 * this notice stays intact.
 *
 * Keith Edwards
 * Georgia Tech / SERC / Multimedia Group
 * April 5, 1990
 * <keith@dali.gatech.edu>
 */


%e 4000
%p 9000
%n 1500
%a 4000
%o 5000


digit     [0-9]
num       [-]?[0-9]+
alpha     [a-zA-Z]
alphaspace[a-zA-Z ]
alphanum  [a-zA-Z0-9]
alphanumdot[a-zA-Z0-9.]
alphanumdotund[a-zA-Z0-9._]
alphanumseq{alphanum}+
nil       \\fnil
roman     \\froman
swiss     \\fswiss
modern    \\fmodern
script    \\fscript
decor     \\fdecor
tech      \\ftech
fctl      ({nil}|{roman}|{swiss}|{modern}|{script}|{decor}|{tech})


          int pos;/* so far used only for attachments */


%%
\\\{      printf("{");
\\\}      printf("}");
\\\       printf("\\");


\n        ;
\\deff{num};
\\defformat;
\\deftab{num};
\\\n      printf("\n");
\{        ;
\}        ;


\\ansi    ;
\\mac     ;
```

```
\\pc      ;
\\pca     ;


\\fonttbl\\f{num};
{fctl}\ [a-zA-Z]+\;;

\\sbasedon{num};
\\snext{num};

\\red{num};
\\green{num};
\\blue{num};
\\cf{num} ;
\\cb{num} ;
\\colortbl;

\\pict    ;
\\brdrs   ;
\\brdrdb  ;
\\brdrth  ;
\\brdrsh  ;
\\brdrdot ;
\\brdrhair;
\\macpict ;
\\wmetafile{num};
\\wbitmap{num};
\\picw{num};
\\pich{num};
\\picwgoal{num};
\\pichgoal{num};
\\picscalex{num};
\\picscaley{num};
\\picscaled;
\\piccropt{num};
\\piccropb{num};
\\piccropl{num};
\\piccropr{num};
\\wbmbitspixel{num};
\\wbmplanes{num};
\\wbmwidthbytes{num};
\\bin{num};

\\footnote;
\\chftn   ;

\\chatn   ;
\\annotation;
\\atnid   ;
```

```
\\headerl ;
\\headerr ;
\\headerf ;
\\footerl ;
\\footerr ;
\\footerf ;

\\title   ;
\\subject ;
\\author  ;
\\operator;
\\keywords;
\\comment ;
\\version{num};
\\doccomm ;
\\vern{num};
\\creatim ;
\\revtim  ;
\\printim ;
\\buptim  ;
\\edmins{num};
\\yr{num} ;
\\mo{num} ;
\\dy{num} ;
\\hr{num} ;
\\min{num};
\\nofpages{num};
\\nofwords{num};
\\nofchars{num};
\\id{num} ;

\\flddirty;
\\fldedit ;
\\fldlock ;
\\fldpriv ;
\\fldinst ;
\\fldrslt ;

\\bxe     ;
\\ixe     ;
\\txe[ ]{alphanumseq};
\\rxe[ ]{alphanumseq};

\\tc      ;
\\tcf{num};
\\tcl{num};

\\bkmkstart;
\\bkmkend ;
```

```
\\paperw{num};
\\paperh{num};
\\margl{num};
\\margr{num};
\\margt{num};
\\margb{num};
\\facingp ;
\\gutter{num};
\\deftab{num};
\\widowctrl;
\\hyphhotz;
\\ftnsep  ;
\\ftnsepc ;
\\ftncn   ;
\\endnotes;
\\enddoc  ;
\\ftntj   ;
\\ftnbj   ;
\\ftnstart{num};
\\ftnrestart;
\\pgnstart{num};
\\linestart{num};
\\landscape;
\\fracwidth;
\\nextfile;
\\template;
\\makebackup;
\\defformat;
\\revisions;
\\margmirror;
\\revprop{num};
\\revbar{num};

\\sectd   ;
\\sbknone ;
\\sbkcol  ;
\\sbkpage ;
\\sbkeven ;
\\sbkodd  ;
\\pgnstarts{num};
\\pgncont ;
\\pgnrestart;
\\pgndec  ;
\\pgnucrm ;
\\pgnlcrm ;
\\pgnucltr;
\\pgnlcltr;
\\pgnx{num};
\\pgny{num};
\\headery{num};
\\footery{num};
\\linemod{num};
\\linex{num};
\\linestarts{num};
\\linerestart;
\\lineppage;
\\linecont;
\\vertalt ;
\\vertal  ;
\\vertalc ;
\\vertalj ;
\\cols{num};
\\colsx{num};
\\linebetcol;
\\endnhere;
\\titlepg ;

\\brdrt   ;
\\brdrb   ;
\\brdrl   ;
\\brdrr   ;
\\box     ;
\\pard    ;
\\s{num}  ;
\\ql      ;
\\qr      ;
\\qj      ;
\\qc      ;
\\fi{num} ;
\\li{num} ;
\\ri{num} ;
\\sb{num} ;
\\sa{num} ;
\\sl{num} ;
\\intbl   ;
\\keep    ;
\\keepn   ;
\\sbys    ;
\\pagebb  ;
\\noline  ;
\\tx{num} ;
\\tqr     ;
\\tqc     ;
\\tqdec   ;
\\tb      ;
\\brdrbar ;
\\brdrbtw ;
\\brdrs   ;
\\brdrth  ;
\\brdrsh  ;
```

```
\\brdrdb  ;
\\brdrdot ;
\\brdrhair;
\\brsp{num};
\\tldot   ;
\\tlhyph  ;
\\tlul    ;
\\tlth    ;

\\posx{num};
\\posxc   ;
\\posxl   ;
\\posxo   ;
\\posxr   ;
\\posy{num};
\\posyil  ;
\\posyt   ;
\\posyc   ;
\\posyb   ;
\\absw{num};
\\dxfrtext{num};
\\pvmrg   ;
\\pvpg    ;
\\phmrg   ;
\\phpg    ;
\\phcol

\\clbrdrb ;
\\clbrdrt ;
\\clbrdrl ;
\\clbrdrr ;
\\trowd   ;
\\trql    ;
\\trqr    ;
\\trqc    ;
\\trgaph{num};
\\trrh{num};
\\trleft{num};
\\cellx{num};
\\clmgf   ;
\\clmrg   ;

\\plain   ;
\\b       ;
\\i       ;
\\strike  ;
\\outl    ;
\\shad    ;
\\scaps   ;
\\caps    ;
```

```
\\v       ;
\\f{num}  ;
\\fs{num} ;
\\expnd{num};
\\ul      ;
\\ulw     ;
\\uldb    ;
\\ulnone  ;
\\up{num} ;
\\dn{num} ;
\\revised ;

\\chdate  ;
\\chtime  ;
\\chpgn   ;
\\chftn   ;
\\chatn   ;
\\chftnsep;
\\\|      ;
\\~       ;
\\\-      ;
\\_       ;
\\'[0-9a-fA-F][0-9a-fA-F];
\\cell    ;
\\row     ;
\\par     ;
\\sect    ;
\\page    ;
\\column  ;
\\line    ;
\\tab     ;
\\:       ;
\\\*      ;

\\rtf{num};
\\gray{num};

\\attachment{num}[ ]{alphanumdotund}+ {
          pos=index( yytext, ' ');
          if (pos)
          printf("[ Attachment inserted here: %s ]\n", pos );
          }




%%
main()
{
          yylex();
}
```

Sound Header Stripping

```
/*
 * Strips the header off NeXT sound files for play on the Sparc
 *
 * Copyright (c) 1990, Keith Edwards
 * May be freely used, modified, distributed, or copied as long as
 * this notice stays intact.
 *
 * Keith Edwards
 * Georgia Tech / SERC / Multimedia Group
 * April 5, 1990
 * <keith@dali.gatech.edu>
 */


/
*************************************************************
**********
 stripnextsnd.c:
 If no args, reads from stdin and writes to stdout.  If one arg, opens
 that file and writes to stdout.  If two args, uses arg one as input and
 arg 2 as output.

*************************************************************
**********/


#include <stdio.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/uio.h>
#include "next_soundstruct.h"


#define BUFSIZE 1024


main( argc, argv, envp )
          int argc;
          char *argv[], *envp[];
{
          int in, out;
          unsigned char buf[BUFSIZE];
          SNDSoundStruct sndStruct;


        if (argc > 3) {
             fprintf( stderr, "Usage: %s [infile [outfile]]\n",
argv[0] );
                exit( 1 );
        }


          if (argc >= 2) {
          if ((in = open( argv[1], O_RDONLY, NULL )) == -1) {
               perror( argv[1] );
               exit( 2 );
          }
          close( 0 );
          dup2( in, 0 );
          }


        if (argc == 3) {
               if ((out = open( argv[2], (O_WRONLY | O_CREAT),
NULL )) == -1){
                        perror( argv[2] );
                        exit( 2 );
                }
                close( 1 );
                dup2( out, 1 );
        }


          read( 0, (char *) &sndStruct, sizeof( SNDSoundStruct
));
          if (sndStruct.magic != SND_MAGIC) {
          fprintf( stderr, "%s: bad magic number on sound in-
put\n",
          argv[0] );
          exit( 3 );
          }
          if (sndStruct.dataFormat != SND_FORMAT_MULAW_8) {
          fprintf( stderr, "%s: bad format on sound input (can
only read 8-bit mulaw)\n", argv[0] );
          exit( 4 );
          }
          if (sndStruct.samplingRate != (int) SND_RATE_CODEC)
          fprintf( stderr, "%s: warning: sampling rate not CO-
DEC\n",
          argv[0] );
          if (sndStruct.channelCount != 1)
          fprintf( stderr, "%s: warning: channel count not 1\n",
          argv[0] );


          while (read( 0, buf, BUFSIZE ) == BUFSIZE)
          write( 1, buf, BUFSIZE );
          write( 1, buf, BUFSIZE );


          close( in ) ; close( out );
          exit( 0 );
}
```

next_soundstruct.h: NeXT sound information needed by stripnextsnd

(this is copyright 1989-1990, NeXT, Inc.)

```
/*
```

```
 *        soundstruct.h
 *        Copyright 1988-89 NeXT, Inc.
 */
typedef struct {
    int magic;/* must be equal to SND_MAGIC */
    int dataLocation;/* Offset or pointer to the raw data */
    int dataSize;/* Number of bytes of data in the raw data */
    int dataFormat;/* The data format code */
    int samplingRate;/* The sampling rate */
    int channelCount;/* The number of channels */
    char info[4];/* Textual information relating to the sound.
*/
} SNDSoundStruct;



#define SND_MAGIC ((int)0x2e736e64)

#define SND_FORMAT_UNSPECIFIED(0)
#define SND_FORMAT_MULAW_8(1)
#define SND_FORMAT_LINEAR_8(2)
#define SND_FORMAT_LINEAR_16(3)
#define SND_FORMAT_LINEAR_24(4)
#define SND_FORMAT_LINEAR_32(5)
#define SND_FORMAT_FLOAT(6)
#define SND_FORMAT_DOUBLE(7)
#define SND_FORMAT_INDIRECT(8)
#define SND_FORMAT_NESTED(9)
#define SND_FORMAT_DSP_CORE(10)
#define SND_FORMAT_DSP_DATA_8(11)
#define SND_FORMAT_DSP_DATA_16(12)
#define SND_FORMAT_DSP_DATA_24(13)
#define SND_FORMAT_DSP_DATA_32(14)
#define SND_FORMAT_DISPLAY(16)
#define SND_FORMAT_MULAW_SQUELCH(17)

#define SND_RATE_CODEC(8012.8210513)
#define SND_RATE_LOW(22050.0)
#define SND_RATE_HIGH(44100.0)
```

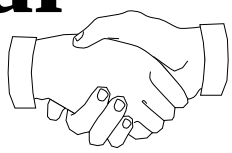
---

denext : shell script

```
#!/bin/sh
#
# Copyright (c) 1990, Keith Edwards
# May be freely used, modified, distributed, or copied as long
as this
# notice stays intact.
#
# <keith@dali.gatech.edu>
#

TMPDIR=/tmp/denext.$$
SPOOLDIR=/usr/spool/mail/`whoami`
DERTF=/hosts/dali/keith/rtf/dertf
STRIPNEXTSND=/hosts/dali/keith/mmmail/stripnextsnd

[ $# != 1 ] && echo "usage: $0 next_mail_file" && exit 1
[ ! -r $1 ] && echo "$0: can\`t read file $1" && exit 2

next=` grep -c "^Next-Attachment:" $1 `
[ x$next = x0 ] && echo $0: $1 is a normal mail message && exit 3

mkdir $TMPDIR
cp $1 $TMPDIR
cd $TMPDIR

start=` grep -n '^$' $1 | head -1 `
start=` expr $start : '^\(.*\):.*$' `

tail +$start $1 > msg.1
file=` grep begin msg.1 | head -1 | awk '{ print $3 }' `
uudecode msg.1

[ ! -f $file ] && echo "$0: uudecode failed...aborting." && exit
4

chmod 600 $file
mv $file $file.Z
uncompress $file.Z

[ ! -r $file ] && echo "$0: uncompress failed...aborting." &&
exit 5

tar xf $file 2> /dev/null
head -$start $1

( [ ! -r index.rtf ] && echo "$0: no index.rtf file." ) || \
                cat index.rtf | $DERTF

for i in *.vox ; do
        [ x$i != "x*.vox" ] && $STRIPNEXTSND $i >/dev/audio
done

rm -rf $TMPDIR
```

SIG Medical

moderated by
Bill Barker &
Conrad Geiger



Welcome to NextMed

the NeXT Medical Users Group mailing list.

NextMed intends to help NeXT users interested in medical uses of the NeXT communicate with one another. It is simply an e-mail alias set up to distribute any mail sent to NextMed to the members of the NeXT Medical Users Group. So be warned, any mail sent to NextMed will be distributed to many colleagues --the list grows by the hour!

I propose that we use this Group as a forum to exchange our project ideas and information. Please feel free to introduce yourself to the group and let us know what you are doing. Also feel free to send questions to this group.

How NextMed works:

Mail sent to ...

NextMed@ulnar.biostr.washington.edu

will be automatically distributed to the mailing list. Mail sent to ...

NextMed-request@ulnar.biostr.washington.edu

will be read only by the NextMed mailing list administrators. Please use this address for adding, deleting or changing member information, or for any other information you may want to pass along to the mailing list handlers. Please feel free to notify your colleagues about NextMed.

Who are we:

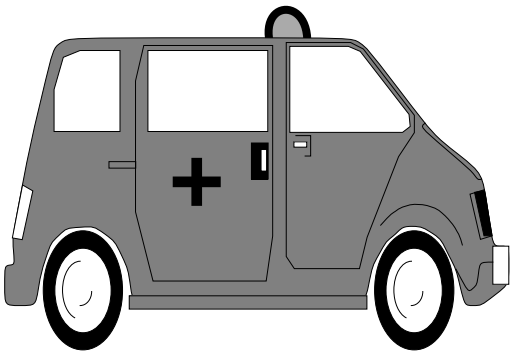
I am **Bill Barker**, Systems Administrator in the Department of Biological Structure, School of Medicine, University of Washington in Seattle, WA.

I am one of the mailing list administrators, along with **Conrad Geiger** of NeXT here in Seattle.

Please forgive any errant mail messages you may have received the past few days. Ah, the joys of debugging....

Cheers,

Bill Barker & Conrad Geiger



Stone Age *n* (1864)

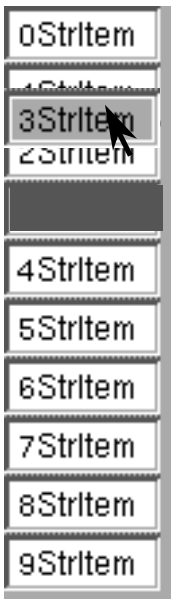
:the first known period of prehistoric human culture, characterized by the use of stone tools.

Moderated by
Andrew Stone

Subclassing Appkit Objects: A Matrix for ordering lists

by Kris Jensen

Here at Stone Design, we've come across several situations where we've wanted to allow the user to place items in a particular order. I've seen various solutions to this that usually involve having the user click on items in order. This works, unless the user makes a mistake or changes her mind. I wanted to provide something similar to the way menu items are ordered in the Interface Builder: the user should be able to directly manipulate the list items.



The Appkit provides the Matrix class for, among other things, visually representing a list of items. So I decided to subclass Matrix. And, since I'm a believer in reusing code whenever possible, I decided to use the drawing routines from the Draw sample program. Draw moves objects by drawing them into an offscreen window and then compositing the image back into the visible window. I planned to override Matrix's mouseDown: method and to add a move: and some subsidiary methods to composite the image of a selected cell. And...it worked! After a little bit of fine-tuning (to insert the moved cell in the "correct" place) , I had a functional MoveMatrix.

Now, the details. The Draw program composites an image from an offscreen window (the selectionCache). In MoveMatrix, I used the mousedown location to find the selected cell and then draw it into the selectionCache, using Cell's drawSelf:inView: method. I replace the cell itself with a placeholder cell that looks like an empty space. I subclassed Cell and overrode drawInside:inView: to make an empty-looking cell.

A Matrix is a flipped view, so in compositing, I had to do a little manipulation to draw the cell in the right place. I also wanted to be sure that the user didn't drag the cell outside the matrix (I have another class for dragging items from one matrix to another). Finally, when the user releases the mousebutton, I need to insert the cell into the right place and make sure the matrix looks good while I'm doing it. This was actually the trickiest part of this simple project; it makes a difference whether the cell is inserted before or after its original location.

To use MoveMatrix in your own applications, follow the instructions in last month's article: copy the files into your project directory, make a subclass of Matrix, rename it to MoveMatrix, and parse the MoveMatrix file. Create a custom view and make it a MoveMatrix. Connect it as appropriate to an outlet in your application or another object. You'll want to do some initialization to set the cellClass and control the appearance of the matrix; one advantage of subclassing Matrix is that we retain all the features of a Matrix. In fact, MoveMatrix is set up to send its action message to the matrix's target if a cell is clicked (or double-clicked) rather than dragged.

By itself, MoveMatrix just allows the user to reorder cells by direct manipulation. To be useful, you may want to build a composite object that will take an ordered list of items and return the list in a different order.

CODE: MoveMatrix.h

```
/* Generated by Interface Builder */

#import <appkit/Matrix.h>

@interface MoveMatrix:Matrix
{
}


+ newFrame: (const NXRect *) frameRect;

- mouseDown:(NXEvent *) event;
- (BOOL) move: (NXEvent *) event;
- selectionCache;
-(int)cacheCell:aCell in:(NXRect *)sbounds inView:aView;
- compositeSelection:(const NXRect *)sbounds from:(int)gstate;
@end
```

CODE: MoveMatrix.m

```
/* Generated by Interface Builder */

#import "MoveMatrix.h"
#import "MyCell.h"
#import <appkit/Cell.h>
#import <appkit/TextFieldCell.h>
#import <appkit/SelectionCell.h>
#import <appkit/MenuCell.h>
#import <appkit/defaults.h>
#import <appkit/nextstd.h>
#import <appkit/tiff.h>
#import <appkit/timer.h>
#import <dpsclient/wraps.h>

@implementation MoveMatrix

+ newFrame: (const NXRect *) frameRect
{
          [self setCellClass:[ActionCell class]];
          self = [super newFrame:frameRect];
          [self setBackgroundGray:0.5];
          return self;
}

- mouseDown:(NXEvent *) event
{
          BOOL gotHit;
          int oldMask;
    oldMask = [window addToEventMask:NX_MOUSEDRAGGED-
MASK|NX_MOUSEUPMASK];
          gotHit = [self move:event];


          [window setEventMask:oldMask];


          if (!gotHit) {
          [super mouseDown: event];
          }
          return self;
}


- selectionCache
/*
 * Shares an off-screen window used to draw the selection in so
that it
 * can be dragged around.  If the current off-screen window is
equal in
 * size or larger than the passed size, then it is simply re-
turned.
 * Otherwise, it is resized to be the specified size.
 */
{
    NXRect rect;
    static id selectioncache = nil;
    NXRect defaultRect = {{0.0, 0.0}, {300.0, 300.0}};


    if (!selectioncache) {
          rect = defaultRect;
          selectioncache = [Window newContent:&rect
            style:NX_PLAINSTYLE
          backing:NX_RETAINED
           buttonMask:0
            defer:NO];
          [selectioncache reenableDisplay];
    } else {
          [selectioncache getFrame:&rect];
          if (rect.size.width < bounds.size.width ||
          rect.size.height < bounds.size.height) {
          [selectioncache sizeWindow:MAX(rect.size.width,
bounds.size.width)
            :MAX(rect.size.height, bounds.size.height)];
          }
    }


    return selectioncache;
}



-(int)cacheCell:aCell in:(NXRect *)sbounds inView:aView
{
```

```
    int i;
    id selectionCache;

          if (!aView) aView = self;
    selectionCache = [aView selectionCache];
    [[selectionCache contentView] lockFocus];
    PSsetgray(NX_WHITE);
    PSsetalpha(0.0);/* fully transparent */
    PStranslate(- NX_X(sbounds), - NX_Y(sbounds));
    NX_WIDTH(sbounds) += 1.0;
    NX_HEIGHT(sbounds) += 1.0;
    NXRectFill(sbounds);
    NX_WIDTH(sbounds) -= 1.0;
    NX_HEIGHT(sbounds) -= 1.0;
    PSsetalpha(1.0);/* fully opaque */
          [aCell drawSelf:sbounds inView:[selectionCache con-
tentView]];
    PStranslate(NX_X(sbounds), NX_Y(sbounds));
    [[selectionCache contentView] unlockFocus];

    return [selectionCache gState];
}




- compositeSelection:(const NXRect *)sbounds from:(int)gstate
/*
 * Composites from the specified gstate whatever part of sbounds
is
 * currently visible in the View.
 */
{
          //y position modified to work with a flipped destina-
tion view
   PScomposite(0.0, 0.0, NX_WIDTH(sbounds), NX_HEIGHT(sbounds),
             gstate, NX_X(sbounds), NX_Y(sbounds)+NX_HEIGHT(s-
bounds), NX_SOVER);
    [window flushWindow];
    NXPing();
    return self;
}


#define stopTimer(timer) if (timer) { \
    NXEndTimer(timer); \
    timer = NULL; \
}


#define startTimer(timer) if (!timer) timer = NXBeginTimer(NULL,
0.1, 0.1);


#define MOVE_MASK NX_MOUSEUPMASK|NX_MOUSEDRAGGEDMASK


- (BOOL) move: (NXEvent *) event
{
    int gstate;
    int row, col;
          static int originalRow, originalCol;
    static id aCell;
    NXCoord dx, dy;
    NXEvent peek;
    NXPoint p, start, last;
    NXRect sbounds, visibleRect, cellFrame;
    NXTrackingTimer *timer = NULL;
    BOOL canScroll, tracking = YES;


    static MyCell *substituteCell = nil;
          if (!substituteCell) substituteCell = [MyCell new];
    last = event->location;
    [self convertPoint:&last fromView:nil];
          [self lockFocus];


          //unhighlight previously highlit cell (aCell is static
and is saved from
          //last entry into method
          if (aCell == nil) {
          [self highlightCellAt:originalRow:originalCol lit:-
NO];
          } else {
          [self getRow:&row andCol:&col ofCell:aCell];
          [self highlightCellAt:row:col lit:NO];
          }
          //highlight cell clicked on
          [self getRow:&originalRow andCol:&originalCol for-
Point:&last];
          [self highlightCellAt:originalRow:originalCol
lit:YES];


    event = [NXApp getNextEvent:MOVE_MASK];
          if (event->type == NX_MOUSEUP) {
          aCell = nil;
          [self unlockFocus];
          return NO;
          }



          // cache the image of the selected cell
          [self getCellFrame:&sbounds at:originalRow:original-
Col];
          aCell = [self cellAt:originalRow:originalCol];
    gstate = [self cacheCell:aCell in:&sbounds];
    [self compositeSelection:&sbounds from:gstate];
          [self putCell:substituteCell at:originalRow:original-
Col];
```

```
    [self getVisibleRect:&visibleRect];
    canScroll = !NXEqualRect(&visibleRect, &bounds);



    while (tracking) {
          p = event->location;
          [self convertPoint:&p fromView:nil];
          //only moving in vertical direction
          dy = p.y - last.y;
          if (dy) {
          [self drawSelf:&sbounds :1];
          NXOffsetRect(&sbounds, 0.0, dy);
          //set up so cell image stays within matrix bounds
          if (!canScroll ||
              (visibleRect.origin.y == 0.0) ||
              ((visibleRect.origin.y + visibleRect.size.-
height)==bounds.size.height))
              {
          //assuming a flipped view
          //above the matrix
          if (sbounds.origin.y < bounds.origin.y) {
          p.y += bounds.origin.y - sbounds.origin.y;
          sbounds.origin.y = bounds.origin.y;
          } else if ((sbounds.origin.y + sbounds.size.height) >
                     (bounds.origin.y + bounds.size.height)) {
          //below the matrix
          p.y -= sbounds.origin.y - ((bounds.origin.y +
          bounds.size.height) - sbounds.size.height);
          sbounds.origin.y = (bounds.origin.y + bounds.size.-
height)
          - sbounds.size.height + 1;
          }
          }
          if (!canScroll || NXContainsRect(&visibleRect,
&sbounds)) {
          [self compositeSelection:&sbounds from:gstate];
          stopTimer(timer);
          }
          last = p;
          }
          tracking = (event->type != NX_MOUSEUP);
          if (tracking) {
          if (canScroll && !NXContainsRect(&visibleRect,
&sbounds)) {
          [window disableFlushWindow];
          [self scrollRectToVisible:&sbounds];
          [self getVisibleRect:&visibleRect];
          [self compositeSelection:&sbounds from:gstate];
          [[window reenableFlushWindow] flushWindow];
          startTimer(timer);
          }
          p = event->location;
          if (![NXApp peekNextEvent:MOVE_MASK into:&peek]) {
          event = [NXApp getNextEvent:MOVE_MASK|NX_TIMERMASK];
          } else {
          event = [NXApp getNextEvent:MOVE_MASK];
          }
          if (event->type == NX_TIMER) event->location = p;
          }
    }


    if (canScroll) stopTimer(timer);


          //set up p so that the x dimension isn't affecting in-
clusion in matrix
          p.x = 0.0;
          // make sure we're dealing with a valid row
          if (![self getRow:&row andCol:&col forPoint:&p]) {
          col = 0;
          if (p.y < 0) row = 0;
          else row = [self cellCount]-1;
          }


          [self drawSelf:&sbounds :1];


          //check intersection of sbounds and the frame of the
cell


          //whole bunch of finetuning to get this to look right
          if (row == originalRow) {
          [self putCell:aCell at:row:col];
          } else {
          [self getCellFrame:&cellFrame at:row:col];
          NXIntersectionRect(&cellFrame,&sbounds);
          if ((sbounds.origin.y == cellFrame.origin.y) && (row
== originalRow+1))
          [self putCell:aCell at:originalRow:originalCol];
          else if ((sbounds.origin.y != cellFrame.origin.y) &&
(row == originalRow-1))
          [self putCell:aCell at:originalRow:originalCol];
          else {
          if (originalRow < row) {
          if (sbounds.origin.y == cellFrame.origin.y) row -= 1;
          [self removeRowAt:originalRow andFree:NO];
          [self insertRowAt:row];
          [[self putCell:aCell at:row:col] free];
          } else {
          if (sbounds.origin.y != cellFrame.origin.y) row += 1;
          [self insertRowAt:row];
          [[self putCell:aCell at:row:col] free];
```

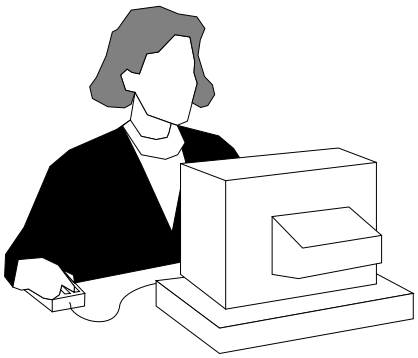
```
            [self getRow:&row andCol:&col ofCell: substitute-
Cell];
            [self removeRowAt:row andFree:NO];
            }
            }
            }
            [self display];
    [window flushWindow];
    [self unlockFocus];

    return YES;
}

@end
```

CODE: MyCell.h

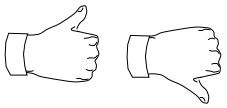
```
/* Generated by Interface Builder */

#import <appkit/Cell.h>

@interface MyCell:Cell
{
            float color;
}

+ new;
- (float) color;
- setColor:(float)newcolor;
- drawInside:(const NXRect *)cellFrame inView:controlView;

@end
```

CODE: MyCell.m

```
/* Generated by Interface Builder */

#import "MyCell.h"
#import <dpsclient/wraps.h>

@implementation MyCell

+ new
{
            printf("Version of MyCell = %d\n",[self version]);
            self = [super new];
            color = 0.333;
            return self;
}

- (float) color
{
            return color;
}

- setColor:(float)newcolor
{
            color = newcolor;
            return self;
}


- drawInside:(const NXRect *)cellFrame inView:controlView
{
            NXRect insideRect;
            NXSetRect(&insideRect,NX_X(cell-
Frame),NX_Y(cellFrame),
              NX_WIDTH(cellFrame),NX_HEIGHT(cellFrame));
            [self getDrawRect:&insideRect];
            PSsetgray(color);
            NXRectFill(&insideRect);
}
@end
```



Reviews



The NeXT Bible: Hardware and Software Systems for the NeXT Computer

By Doug Clapp. 683 pp. Copyright (c) 1990 by Brady Books (Simon & Schuster), New York. ISBN 0-13-620725-1 $29.95.

by Terrence J. Talbot

The back cover of Doug Clapp's new book, The NeXT Bible, declares that *the NeXT's sleek black box begs exploration.*This is not so much a statement for Clapp as it is a deep felt sentiment. The sheer breadth of information Doug Clapp wishes to present to his reader clearly shows that he has the goal of exploration in mind. Unfortunately, the information presented by Clapp is nothing more than a surface skim of the NeXT's features. Clapp redundantly extols the virtues of the NeXT without really giving an in-depth picture of how those virtues operate to give you ™the best computer ever made.As such, it is really a book for people who have hardly even seen a NeXT, and, unfortunately, not much more.

Overall, the Bible does not present much new information beyond what is already contained in the NeXT manuals included with every cube. Doug Clapp holds out a smorgasbord of info about the NeXT, but it is undetailed in many respects. Rather, Clapp tries to capture his own personal feeling and direction about the NeXT: to him, as to virtually everyone, it is a personal workstation of almost untold breadth. As such, Doug Clapp prefers to list and direct, rather than explain. Perhaps if he had titled his book **The NeXT Dictionary**rather than the self-admittedly hubristic title of **The NeXT Bibl**e,Clapp's book would be better received. It is in no way similar to the popular Macintosh Bible, which presents page after page of hints, tips, and tricks about the Macintosh operating system and some of its popular software. Instead, The NeXT Bible methodically explores the basic hardware and software of the NeXT computer, in much the same way as Bruce Webster's earlier work, The NeXT Book.

As a dictionary of terms or the most basic of starting places, Clapp's book will be most useful to those curious about the NeXT but uncommitted.

As a dictionary of terms or the most basic of starting places, Clapp's book will be most useful to those curious about the NeXT but uncommitted. That is, those people who know little of UNIX, but have heard of its power, or know little of the richness of the NeXT environment, but wish to learn more. Those of us who already have cubes, have studied the User's Reference Manual, and have at least browsed a few chapters of the System Reference Manual, already know most everything that is contained in Clapp's book. Still, the enormous scope of what Clapp presents makes the book well worth a look.

UNIX

If you are unfamiliar with UNIX, the Bible contains quite a rich presentation of the interesting aspects of UNIX. In a supplementary appendix Clapp lists those UNIX commands and shell scripts that best explicate his feeling about the power of UNIX and the sophistication of the NeXT. There are no detailed explanations here, no large examples, no tutorials, no exercises: just a simple listing of the command, perhaps a screen dump of what it does, and why he finds it unique or useful. Overall, it is a wonderful place to browse if you unfamiliar with scope of UNIX or curious to see what utilities have been created or added to by NeXT.

Clapp is right in thinking that a basic understanding of how UNIX works goes a long way toward clarifying one's understanding of how the features of the NeXT relate to each other and work together, particularly in the area of network protocols and networking applications. Understanding how UNIX both fits into the picture of the NeXT environment and guides it is fundamental to understanding the power of the system itself. Clapp properly captures this notion and invites his reader to explore UNIX by utilizing both the on-line manuals and the additional books listed in a subsequent appendix on further reading.

To supplement his listing, and really to set the table for his broader discussion, Doug Clapp begins his book by presenting a short history of UNIX, its creation, and the design concepts behind its particular NeXT Mach implementation on the cube. The Bible presents a clear but brief introduction to the Shell, UNIX files and directories, virtual memory, interprocess communication, and Mach itself.

NextStep

From there Clapp moves on to a presentation of NextStep, or the user interface of the NeXT computer. It is a well illustrated section, as are most all of Clapp's chapters, and cleanly lays out the concepts behind what he calls *the most elegant and most useful graphic interface ever to grace a computer.*

Clapp correctly describes the interface as *staunchly mouse-based*. He attacks the interface for not having the forethought to be able to do everything with the keyboard that one can do with the mouse. These are of course personal preferences. I agree that the use of the arrow keys could be weaved into more of the interface activity, especially when traversing browsers. Indeed, I would take it one step further and ask that arrow keys be functional whenever positioning any graphic object onscreen. This was something I looked for in Interface Builder and didn't find. Everyone has their own wish list–Clapp is no exception.

Clapp does, however, suggest one interface modification with which I disagree. He would prefer to have a keyboard way of cyclingthrough applications, as if on a rotating wheel or palette. This was an excellent idea back in early 1986 when it debuted in Andy Hertzfeld's amazing Macintosh application called Switcher: the forerunner of MultiFinder. I don't see the reimplementation of this as a feasible or useful tool in NeXT's multitasking environment. There would be no proper way to log applications into the *wheel*.This is the area where I think NeXT's invention of miniwindows really shines. Not only can you pull up the application you want, but a specific document window to work with. Clapp seems to miss this point, but it is inconsequential. I bring up my difference with Clapp only in the spirit of what he relays on page 102: ...NeXT is actively seeking your suggestions for improvement.

Whichever way you come out on this, the bottom line is simple and straightforward: NeXT offers you the power of UNIX without the pain of UNIX. The Workspace Manager and NextStep free you from needing to know UNIX, while still providing you with the majority of its functionality. However, understand UNIX, even a little UNIX, and suddenly much of NextStep and the NeXT computer's power, e.g. networking, multitasking, etc., begins to make sense. This is the reason that Clapp invites you to explore; and it is a sentiment with which I would agree.

Clapp rounds out his basic presentation of the NeXT with a description of the system's hardware. If you are curious about what's ™inside the cube,this is a good place to go for a general, but thorough, description of the motherboard, its various processors, and the periphery that surround it.

Networking

Clapp thoughtfully devotes and entire section (a chapter really) to networking, its standards and protocols, and, very generally, its implementation on the cube. If you are curious about the Client-Server model of distributed computing, Ethernet, filesharing, TCP/IP, and the like, Clapp provides a very good and clear discussion of these concepts and presents them in an interesting manner. If you are wondering about uucp, NFS, or the UserManager, the Bible presents the basic ideas behind the software and protocols. You won't come away an networking expert, but if you have no idea of what a network address is, or a network path, the the Bible's networking chapter is the place to begin.

Programming the NeXT: IB and the Kits

To begin his fourth section, Clapp begins by giving you the basic ideas behind object-oriented programming and then launches in to a quick discussion of the Window Server and the NeXT's use of Display Postscript.

From there, the Bible takes you through a complete, but undetailed, tour of Interface Builder: the main drafting table for constructing an object-oriented user interface. Clapp makes the insightful point that in the world of NextStep programming, i.e. utilizing objects, the user interface becomes the program. In no way is the application's interface added on or wrapped around a shell program. Virtually each object seen on the screen maintains its own functionality: it knows about itself and knows what to do with itself when a particular action is requested by another object. The power of Interface Builder (IB), and I'm not sure Clapp adequately stresses the point, is that once IB fully knows of the objects you wish to use in constructing your application, nothing but (visually) a flip of a switch is needed to test your application. You're not only testing whether you placed a button correctly, but also if you've connected it's action to the desired target. It's an amazing feeling to draw a connection from a button to a window, pick that window's printPSCode: method, hit command-r and actually watch your button work! (The window prints itself.) Try that with the Macintosh's toolbox! The power of IB, when more objects are fully fleshed out and fully linked with IB, will be much more expansive and powerful than Clapp's simple HyperCard analogy, and much more powerful than simply testing the placement of interface elements. If the interface truly is the application, and it is on

the NeXT, you'll be creating and testing the application itself. That truly is something to celebrate!

Beyond this, Clapp's discussion of the Application Kit is a bit thinner than I would like, even for a basic, novice-level book. Rather than give even the most meager of working examples, such as the simple but impressive Scribble program presented in Bruce Webster's The NeXT Book, Clapp merely hints about the wonder of object-oriented programming and the amazing feats of the NeXT software engineers. To take just one example, Clapp's discussion of the Speaker and Listener classes, contained in the Application Kit, is taken care of in three sentences: *Spend some time with the Speaker and Listener classes. NeXT has made them easy to use. Use `em*.Here I was looking for something more. Not a full blown tutorial, not a programming manual, but something more. Perhaps my own inability to define the level of detail appropriate for this type of book also stopped Clapp from elaborating in any depth. I can sympathize with that. It's hard to describe, in a few pages, the depth and power of the NeXT. Still, you feel sort of left out of the discussion and wonder what Clapp knows that you don't.

The Bible also quickly presents the Sound and Music Kits, something that The NeXT Book doesn't do, but then abruptly leaves the subject of programming to deal with the avalanche of applications bundled with the cube. The section on programming lacks a discussion of application design and philosophy that I think would have put the Kits in better perspective. Again, there is no great detail here, just a list of objects, or in this case the hierarchy of Kit classes.

NextApps

The final major section briefly discusses the main applications that are included in Release 1.0: WriteNow, Edit, Mail, Mathematica, Digital Webster, Quotations, Librarian, etc. The chapters are brief descriptions of the basic functionality of the applications and no more. No great tips in utilizing the applications are offered: that is not Clapp's intention.

Here I was looking for something more. Not a full blown tutorial, not a programming manual, but something more.

Rather than rehash what Clapp has already said about NeXT's bundled software, let me simply comment on a few of his remarks that really stuck out. Clapp decries Librarian as a less than powerful, less than useful application than it at first appears. I agree with Clapp's criticism of Digital Librarian, but can't endorse his recommendation of giving it up altogether for the standard UNIX index utility: it defeats the purpose of NextStep to head back into UNIX. Hopefully, Librarian will be improved in subsequent releases to incorporate Clapp's and other's suggestions. It is too easy to miss things in Librarian and I think this is what Clapp is getting at. I would be most impressed with the inclusion of some general free text searching capabilities: something that doesn't restrict you to simple Boolean searches, but more of a library tool. A good librarian should be able to negotiate through vast bodies of knowledge to find the two or three paragraphs you're really looking for, rather than just point out a few books that contain the words you're interested in. (See Mark ^Zimmerman's article in the April, 1990, NUJ.)

Strangely, in one page, Doug Clapp dismisses the Digital Quotations application as a throwback to the dark ages of literature. Of course, Clapp also begins each chapter with one or more useless quotes culled from the depths of DQ, so I guess it can't be all bad.

The Jewel of the Book

The rest of the book is given over to speculation about the future (color printers and the like) and a few useful appendices on third party products, listings of key codes and command keys, and the catalog of UNIX commands discussed above. Sandwiched in between these pages is an insightful interview with Brad Cox, the chief inventor of Objective-C, the object-oriented computer language that forms the basis of the NeXT's Application Kit. The interview itself is an interesting question and answer look into the future of programming and the creation of *software-ICs*.Ultimately, everyone will be a programmer. You'll only need to go to an Objective-C hardware store and buy the features, e.g. spell check, grammar check, dictionary, graph paper, that you need to give your application its necessary functionality. As Clapp himself might say: It's a fine interview. Well presented. Read it.

Wrapping It Up

The mechanics of the book are slightly less than top-notch. Contents pages do not always match up; index entries are sparse. Beyond this, Clapp spends over a hundred pages of his space listing every file contained in the standard Software Release 1.0 disk. It's a simple list, no more. He presents it, he says, because he *wanted to see The Thing Itself*.I would have appreciated at least a breakdown of the listing into categories so that I could grapple with it more easily. Unfortunately,

this was consciously not done. To me it seems to come off as mere padding, rather than the sweeping look at the greatness of the cube that was intended. You'll have to judge the merits of this appendix for yourself.

The thrust of Clapp's book rhetorically asks *What's NeXT*?In answering this, Doug Clapp presents a smorgasbord of info about the NeXT, but it is undetailed in any traditional respect. This may be a good thing for a general audience. Seen as a dictionary, rather than a bible, Clapp's book may have good, intrinsic value as a reference source. Bearing this in mind, and despite the faults listed above, this work belongs on the shelf of anyone interested in exploring the expansive world of NeXT computing.

Any publication, assuming that it is well written, is important for the success of the NeXT. During the cube's infancy, which is where I believe it is, however prodigious it may seem at this point, any book about the NeXT is welcome, if only to sustain interest in the machine. Clapp's book is certainly interesting and will, hopefully, lay the groundwork for deeper and more specialized publications.

The author can be reached electronically at 72307.1727@CompuServe.COM.

Yellow Porcupine

Doug Clapp. The NeXT Bible: Hardware and Software Systems for the NeXT Computer. Brady, New York, 1990. Paperback, xix, 684 pp.

Jacob Gore

The word *Bibl*ein the title of computer books has come to mean that the book provides details and insights that one needs once one gets past introductory material on the topic. The NeXT Bible does not: it is written for people new to NeXT and/or to Unix. It is also written by a person new to NeXT (but an experienced Mac GUI user) and to Unix.

When I scanned through The NeXT Bible at the book store, I was impressed with its easy-going presentation style and simple language. It did not seem like it presented any information that could not be found in NeXT online documentation, but I like seeing alternative presentations of material. I spotted one error, which I assumed typographical, in the appendix that lists Unix commands, but such errors are inevitable in the first edition of any book. I bought it, went home, marked the error with a Post-it note, and started reading.

Such errors are inevitable in the first edition of any book. I bought it, went home, marked the error with a Post-it note, and started reading... [marking errors with post-its]... I still went through two pads. My book looks like a yellow porcupine

I needed another Post-it on page 12. Then on page 23. Then two on page 24, one each on page 26, 27, 30, 32, 34 This is when I decided to contribute to the preservation of the world's forests by cutting the Post-it notes into strips. I still went through two pads. My book looks like a yellow porcupine.

There are all kinds of errors in the book, and very few of them are of the *harmless* typovariety. There are typos in examples that change their meaning (such as using grave apostrophes instead of apostrophes) and make them not work. There are statements that are just plain false (e.g., *UNIX* doesn't allow file names containing spaces. p. 24). There is a whole slew of errors stemming from the habit of the author to do all his work from the root account and use / as the home directory (two terrible things to do in themselves). There are very questionable statements (*The Mach kernel is object-orientedbecause the idea of `message passing' is at the lowest level of Mach*. p. 20). There is terrible advice (log in as root and the computer is all under your control; create new commands in /bin, p. 55), and very questionable advice (*if you are just starting to learn to program, learn assembly language first.* p. 250).

There is much confusion typical with Unix beginners that the author propagates to the reader. Examples are the instruction to use Control-c or Control-z (emphasis mine unless otherwise noted–JVG)to stop a running programand return to the shell's prompt (p. 48); the confusion between daemons and their clients (lpd vs. lpr, p. 57), file name paths vs. search paths(p. 27).

In addition, Mr. Clapp relays a set of misconceptions that are due to his inexperience with multi-user systems of any kind (*wall* sends a message to all users on the network.p. 523; *who* is a version of the whoami command.p. 523). The confusion between who and whoami is symptomatic of another type of mistakes in the book: trying a command, the author gets output that is coincidentally similar to that of another command, and jumps to the conclusion that the two commands are basically the same (pwd vs. dirs, p. 49).

Though the coverage of Unix is

bad enough, it is surpassed in profusion of confusion by just about anything that has to do with networking, and especially Chapter 12, *A Networking Primer*. If you have read it, and it served as your introduction to networking on a NeXT, to TCP/IP, to UUCP, to the Internet–go to a hypnotist immediately and have all that nonsense erased from your mind! If, on the other hand, you already know all this stuff, you may want to read Chapter 12: it is very entertaining.

Other sections of the book have a much lower frequency of errors, but they are not perfect either. Another frequent problem with those sections is redundancy between narration and illustrations.

The following table provides a rough idea of the distribution of the errors in the book. I intended to make it more specific by classifying the types of mistakes, but this edition of the book is just not worth that much trouble.

ChapterPagesPost-its

Section I: Software

1.In the Beginning: UNIX40

2.The Design of UNIX102

3.Mach62

4.Files and Directories249

5.Hands-on UNIX148

6.NextStep Fundamentals450

7.The Workspace Manager281

Section II: Hardware

8.The NeXT Display, Keyboard, and Mouse121

9.Inside the Cube221

10.Disks and Drives110

11.The NeXT Laser Printer116

Intermission: Images11

Section III: Networking and Communications

12.A Networking Primer3016*

Section IV: Programming the Cube

13.Programming the NeXT Computer105

14.The Window Server and Display Postscript82

15.Interface Builder160

16.The NeXT Application Kit, Part I: Non-Responsive Classes182

17.–, Part II: Responder and Its Subclasses164

18.Sound and Music130

Section IV: A Software Treasure Chest

19.WriteNow40

20.Edit150

21.Mail80

22.Mathematica201

23.Digital Webster60

24.Quotations10

25.Digital Librarian102

26.Terminal and Shell62

27.And More°131

Section VI: The Present and Future

28.Coming Soon°Maybe11

29.Interview: Brad Cox12

Appendices

A.Command-Key Combinations20

B.Keyboard Key Codes1

C.[Shell] Special Characters40

D.NeXT/UNIX Files and Utilities7725

E.Files in the NeXT 1.0 Software Release109

F.Third-Party Products18

G.Further Reading12

H.NeXT Hardware Specifications51

Plus many, many very, very wrong conclusions drawn from these errors are not thoroughly reviewed due to the reviewer's ignorance of the topic. Also, there is no sense debugging 1-year-old speculations which outdated, too volatile to be included in a book

One should note Appendix E. One hundred and eight pages of output from find / -print | sort. One hundred and eight pages! This from a guy who advises people to turn off their NeXTs after every use, to save resources. There is no excuse for this. There are other places where Mr. Clapp seems to go for the bulk: he includes whole system files, where a small portion could have served as a perfectly fine example. And this information is on-line, after all.

This edition of this book is not suitable for people new to Unix or to the NeXT. Its only possible use is to test experts: every sentence, every example, every claim is a true/false question. The majority is true, but too many are false.

This edition should have never been published. If this was software, it would not have made it out of alpha test. I have never heard of a book being recalled, but this one really should be. My copy can be returned with many useful Post-it stickers.

I hope Mr. Clapp does fix all the mistakes and produces a second edition. It could be such a delightful book.

Unixis a registered trademark of AT&T Bell Laboratories. ™Post-it is a trademark of Minnesota Mining and Manufacturing Company (3M).

Jacob Gore, <jacob@gore.-com>

Minimum Spanning Trees

Erica J. Liebman

After reading **Laura Fischer**'s letter about the **Storage** class,[see *Feedback from the Trenches* p.4] I set out to convert one of my *Adventures in Graphics Education* from List to Storage. After all, why have an extra wasteful class just for points when NeXT itself has supplied the useful and handy **NXPoint** as one of its standard ObjC classes? Fine.

*After all, why have an extra wasteful class just for points when NeXT itself has supplied the useful and handy **NXPoint** as one of its standard ObjC classes?*

Here's the goal of the effort : convert my minimum spanning tree program from using the **objc/List.h** implementation of linked lists to the **objc/Storage.h** implementation of growable arrays. The program is a simple demonstration which allows the user to enter data points into a custom view and calculates the set of edges that connect all points in the set, such that the sum of edge lengths is minimal. This is based on **Pat Flynn**'s implementation of MST, stolen I think from one of Ulman's books.

Let's take a look at these two data types. First, the official definitions from the online documentation

List : List allows easy manipulations of collections of objects. Collections can be manipulated as fixed or variable sized lists, sets, or ordered collections.

Storage :Storage implements a general storage allocator. Storage allows you to have an array with arbitrary type elements without having to subclass for each type desired. When creating a new instance of storage you specify the size of the array elements and a description of their type. The description is needed for archiving the class and must agree with the specified element size.

Now, ignoring the above, both List and Storage do approximately the same job : maintaining an ordered set of objects. Both have similar protocols. For List we find addObject:, objectAt:,removeObjectAt: and count. For Storage, the protocol mirrors with addElement:, elementAt:, removeAt:, and, again, count. These methods, for both types, allow you to create a random-access ordered linked-list set.

However, there is one major difference : lists handle Objective-C objects, storages handle pointers. This is real important. If you chose to go with a storage class rather than list, you are going to have to be really responsible when using memory. Be prepared to *malloc*, before you start to use storage.

In list, when you are through, you need only perform commands similar to the following :

```
[myList freeObjects];
[myList free];
```

These two steps take care of deallocating your entire structure including "client" objects of your list. Not so in Storage. You must specifically deallocate. In the example program which follows, I wimped out. I decided to skip this step and wait for the program to terminate before cleaning up (after all, how many data points can you enter in a single running of this program). This approach is neither recommended nor reflects good programming habits. If you play with memory, make sure that you, not the great core-dumper in the sky, is in control.

```
mLoc = (NXPoint *)malloc(sizeof(NXPoint));
*mLoc = ptr->location;
...
[dataPoints addElement:mLoc];
```

Allocation and creation are also importantly different. As mentioned before, you will have to allocate memory and then add pointers into the storage. You will also want to avoid the **new** message for storage and replace it with

```
+ newCount:elementSize:description:
```

I start with a count of zero, a size corresponding to sizeof(NXPoint) in this example, and an arbitrary description in this case "List of Point". I searched through digital librarian, but I never found a good justification for having a description field. I'd sure like to know why it is included.

```
dataPoints = [Storage newCount:0 elementSize:-
sizeof(NXPoint)   description:"List of Points"];
```

Next, you are going to have to start dealing with pointers. No more namby-pamby objects here. When you ask for an object you are going to get a vanilla pointer back that you will have to **cast** into something useful.

```
p1 = (NXPoint *)[points1 elementAt:i];
p2 = (NXPoint *)[points2 elementAt:j];
dx = p2->x - p1->x;
dy = p2->y - p1->y;
```

return(sqrt(dx*dx + dy*dy));

Since you are dealing with pointers, don't create extra local-variable space. If you must use temporary variables, don't forget you will need to deal with the *contents*, not the address of the results of **elementsAt:**.

Now, we have to ask ourselves, is this effort and potential danger from dealing directly with memory worth the effort. I maintain that if you're going to have a whole lot of very simple data types a la NXPoint, Storage isn't a bad approach but if you have objects to start with, for heaven's sake, stick with List.

How to Compile It

1. Create a fresh new directory, start IB , create a new application & save into that new directory. Create a new project into this directory. Put MSTView.[hm] in here.

2. Subclass View as MSTView. Parse in MSTView. Add the .[mh] files to the project. Save.

3. Add a custom view and two buttons to your main window. Label the buttons *clear* and *animate*. Set your custom view to MSTView (Command-1 and select). Connect the buttons to the view via alt-drag-- choose the appropriate messages (clear: and animateMST:) and make sure to click *connect* to finish the connections. Save.

4. Leave IB. Compile through *make*. Run the program. Questions & real deep problems to erica@kong.gatech.edu.

Source : MSTView.h

```
#import <appkit/appkit.h>
#import <objc/Object.h>
#import <objc/Storage.h>
@interface MSTView:View
{
    id dataPoints;
}
+ newFrame:(const NXRect *)tF;
- drawSelf:(NXRect *)tR:(int)c;
- mouseDown:(NXEvent *)ptr;
- clear:sender;
- animateMST:sender;
- line:(id)points1:(id)points2:(int)i:(int)j:(f-
loat)initColor:(float)finalColor;
@end
```

Source : MSTView.m

```
/* Generated by Interface Builder */
#import "MSTView.h"
@implementation MSTView
// Frame Creation for custom View
+ newFrame:(const NXRect *)tF
{
  self = [super newFrame:tF];
  dataPoints = [Storage newCount:0 elementSize:sizeof(NXPoint)
          description:"List of Points"];
  return self;
}
// Initialize the Frame View
- drawSelf:(NXRect *)tR:(int)c
{
  NXEraseRect(&bounds);
  PSsetgray(0.0);
  NXFrameRect(&bounds);
  return self;
}
// Deal with User Interaction
- mouseDown:(NXEvent *)ptr
{
  NXPoint *mLoc;
  [self lockFocus];
  PSsetgray(0.50);
  PSsetlinewidth(1.0);
  mLoc = (NXPoint *)malloc(sizeof(NXPoint));
  *mLoc = ptr->location;
  [self convertPoint:mLoc fromView:nil];
  PSarc(mLoc->x, mLoc->y, 2.0, 0.0, 360.0);
  PSstroke();
  [window flushWindow];
  [self unlockFocus];
  [dataPoints addElement:mLoc];
  return self;
}
- clear:sender
{
  [dataPoints free];
  dataPoints = [Storage newCount:0 elementSize:sizeof(NXPoint)
          description:"List of Points"];
  [self lockFocus];
  NXEraseRect(&bounds);
  NXFrameRect(&bounds);
  [self unlockFocus];
  return self;
}
float dist(id points1, id points2, int  i, int j)
{
  NXPoint *p1, *p2;
  float dx, dy;

  p1 = (NXPoint *)[points1 elementAt:i];
  p2 = (NXPoint *)[points2 elementAt:j];
```

```
  dx = p2->x - p1->x;
  dy = p2->y - p1->y;
  return(sqrt(dx*dx + dy*dy));
}
- line:(id)points1:(id)points2:(int)i:(int)j:(f-
loat)initColor:(float)finalColor;
{
  int k;
  NXPoint *p1, *p2;
  p1 = (NXPoint *)[points1 elementAt:i];
  p2 = (NXPoint *)[points2 elementAt:j];
  [self lockFocus];
 PSsetgray(initColor);
  PSmoveto(p1->x, p1->y);
  PSlineto(p2->x,p2->y);
  PSstroke();
  [window flushWindow];
  PSsetgray(finalColor);
  PSmoveto(p1->x, p1->y);
  PSlineto(p2->x,p2->y);
  PSstroke();
  [window flushWindow];
  [self unlockFocus];
}
// The Algorithm and the Animation for MST algo.
- animateMST:sender
{
  NXPoint curr, prev, next;
  int i,j,k, speed;
  int minP1, minP2;
  float minLength = 99999.9;
  id MST, NMST;
  id edges;
  if ([dataPoints count] < 2)
  {
     NXRunAlertPanel("Minimum Spanning Tree",
       "Enter Some Points",
       "OK", NULL, NULL);
     return self;
  }
  // turn off mouse
  [NXWait push];
  // Initialize MST & NMST
  MST = [Storage newCount:0 elementSize:sizeof(NXPoint)
          description:"List of Points"];
  NMST = [Storage newCount:0 elementSize:sizeof(NXPoint)
          description:"List of Points"];
  edges = [Storage newCount:0 elementSize:sizeof(NXPoint)
          description:"List of Points"];
  for (i = 0; i < [dataPoints count]; i++)
    [NMST addElement:[dataPoints elementAt:i]];
  // first find the shortest edge -- this is done in O(n^2)
  [self lockFocus];
  minP1 = 0; minP2 = 0; minLength = 99999.9;
  for (i = 0; i < [dataPoints count]; i++)
    for (j = i+1; j < [dataPoints count]; j++)
    {
     [self line:dataPoints:-
dataPoints:minP1:minP2:NX_DKGRAY:NX_DKGRAY];
     [self line:dataPoints:dataPoints: i: j:NX_LTGRAY:NX_WHITE];
     if ((k = dist(dataPoints, dataPoints, i,j)) < minLength)
      {
        [self line:dataPoints:-
dataPoints:minP1:minP2:NX_LTGRAY:NX_WHITE];
        minLength = k;
        minP1 = i;
        minP2 = j;
        [self line:dataPoints:-
dataPoints:minP1:minP2:NX_LTGRAY:NX_DKGRAY];
      }
    }
  // ok, we've found the shortest edge.  include it.
  [self line:dataPoints:-
dataPoints:minP1:minP2:NX_LTGRAY:NX_BLACK];
  [MST addElement:[dataPoints elementAt:minP1]];
  [MST addElement:[dataPoints elementAt:minP2]];
  [NMST removeAt:minP2];
  [NMST removeAt:minP1];
  [edges addElement:[dataPoints elementAt:minP1]];
  [edges addElement:[dataPoints elementAt:minP2]];
  // now add each "best" edge that spans from the MST
  //    to the set of free points
  while ([NMST count] > 0)
  {
    minLength = 99999.9; minP1 = 0; minP2 = 0;
    for (i = 0; i < ([edges count]/2); i++)
      [self line:edges:edges:i*2:(i*2+1):NX_BLACK:NX_BLACK];

    for (i = 0; i < [MST count]; i++)
      for (j = 0; j < [NMST count]; j++)
      {
        [self line:MST:NMST:i:j:NX_LTGRAY:NX_WHITE];
        if ((k = dist(MST,NMST,i,j)) < minLength)
        {
          [self line:MST:NMST:minP1:minP2:NX_LTGRAY:NX_WHITE];
          minLength = k;
          minP1 = i;
          minP2 = j;
      [self line:MST:NMST:minP1:minP2:NX_LTGRAY:NX_BLACK];
        }
      }
    [edges addElement:[MST elementAt:minP1]];
    [MST addElement:[NMST elementAt:minP2]];
    [edges addElement:[NMST elementAt:minP2]];
    [NMST removeAt:minP2];
  }
  // show them edges again.
  for (i = 0; i < ([edges count]/2); i++)
    [self line:edges:edges:i*2:(i*2+1):NX_BLACK:NX_BLACK];
  // restore cursor
  NXPing();
  [NXWait pop];
  [MST free];
  [NMST free];
  [edges free];
  return self;
}
@end
```

Reviews, Rumors & Stuff

A Myopic Eye

into the NeXT Community

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

TopDraw

I had the opportunity this month to sit down with a copy of TopDraw for a few hours -- for a test drive, wheel-kicking, you get the idea. Of course, I first sat through the Top Draw Tutorial -- all seventeen megabytes of it. What is it with NeXT and British women? Is there some cult that I'm missing out on? Maybe I should work on a cultured English accent. (Yoooour Newslettuh is out of Paypuh).

First off, TopDraw has a bunch of features. Lots of features. Hundreds of thousands of features. Well, maybe I am exaggerating a bit. As in TextArt, I was presented with one heck of a main control panel. Within a few minutes, I had created a shadowed, highlited, rather strange-looking little man, peeking into a manhole cover. I skewed, I flipped, I rotated. Overall, this program is loaded with features. Oh, yeah, I guess I already said that.

Ok, here's what I didn't like : 1. I couldn't find how to save only a part of the screen a la *Icon*. When I saved to EPS, I couldn't save just my man&manhole. I had to save a whole lot of white space.

2. Copy & paste didn't seem to work correctly between TopDraw and other programs like WriteNow and Frame. I tried to paste my Man&Manhole picture in this review, and couldn't.

3. I really hated that the special inspector panels were "modal dialogs", forcing my attention and then going away. I really prefer to keep my dialogs around when their context still makes sense. (Again, I really like the way TextArt does this).

4. I didn't like that I couldn't stick this thing into freehand drawing mode. For each stroke of my wobbly Man, I had to keep going back to click on freehand tool.

Here's what I liked : 1. skewing (especially circular objects) was a *breeze*. I always want to have general ellipses and TopDraw provided that for me.

2. The way it handles grouping & ungrouping is great. I really was impressed by the amount of information stored per object & group of objects. It remembered old groupings even after I regrouped.

3. I **loved** the pop forward/pop backwards -- this is GREAT. Simple, to the point, effective and a wonderful technique of handling precedence in an "object-oriented" drawing system.

4. The effects panel is great - I wish there were more effects. You get a really splashy look for very little effort. This also works on letters as well as objects and is just a totally neat feature. (You can quote me on that).

Overall, for a first peek? I'd say that this is a tool to be reckoned with, one that could use a little more organization & thought. It isn't a tool to be picked up in a day -- a lot of people are going to use simpler programs just because of the steep learning curve. I was very impressed by the quality of drawings I was able to produce in short order, but then again I love reading manuals -- most people don't.

As I continue working with TopDraw, I'll keep y'all informed on how it goes. It looks like its going to be fun.

Mathematica

I received a postcard in the mail the other day. Wolfram is looking for people to join and coordinate Mathematica special interest groups. If you think you could coordinate a BuzzNUG Mathematica SIG, please contact both Wolfram at mathuser@wri.com and Conrad_Geiger@next.-com (Conrad is helping to coordinate all our SIGs.)

For any of you who just want a quick reference on Mathematica contact info, here it is :

Wolfram Research Inc.

PO Box 6059

Champaign, IL 61826-6059

217-398-0700

(Tell 'em BuzzNUG sent ya).

FTP Fun

• **cs.orst.edu & j.cc.purdue.edu** - *the most popular archive sites for NeXT related materials currently used. There is a mail server at purdue:* ***archive-server@c-c.purdue.edu*** *- send mail with the subject line of "HELP".*

• **uvaarpa.acc.virginia.edu** - *has a lot of dsp related materials*

• **umd5.umd.edu** - *an official site, but not used much*

• **prep.ai.mit.edu** - *sacred GNU burial ground - none of it NeXT specific*

• **sutru.sfsu.edu** - *some activity lately*

• **linc.cis.upenn.edu** - *nethack sources -- patchlevel 7 compiles right off if you follow the instructions*

• **athena-dist.mit.edu** - *where to get X sources including NeXT-X*

• **cs.uiuc.edu** - *where to get Epoch (an emacs tool)*

• **ps-file-server@adobe.com or adobe!ps-file server** - *not really an FTP site - these are mail servers of files. Send a mail letter to one of these addresses with the subject line of "HELP"*

• **eesun1.arl.utexas.edu/129.107.2.51** - *new arrival, some nice stuff*

• **sumex.stanford.edu** - *some interesting stuff*

•**winnie.princeton.edu** - *ditto, but supposedly some relating to MUSIC in particular*

• **dagon.acc.stolaf.edu** - *ditto again*

• **cica.cica.indiana.edu** - *some old NeXT update stuff*

• **csus.edu** - *some stuff*

• **nic.stolaf.edu** in the pub/ps -*postscript related archive.*

• **ftp.ncsa.uiuc.edu** - *Mathematica Archive*

***Also** (care of Conrad Geiger): aeneas.mit.edu, csus.edu.;cs.ubc.ca , sachiko.acc.stolaf.edu (/usr/src/local, / usr/spool/ftp, /home/sachiko/cdr),princeton.edu (/pub/music), unidata.ucar.edu, ftphost.cac.washington.edu,-watsun.cc.columbia.edu (version of C-kermit for NeXT), 129.18.18.3 - campus consultant archive server*

• Introduction toEmacs LISP programming book that Robert Chassell (bob@ai.mit.edu) is writing... on the Internet available for anonymous ftp. *I'd guess this would be on prep.ai.mit.edu - expect a size of about 1/2 a meg.*

PLANNED FTP ARCHIVES :

People with BCS, Washinton Apple Pi and BaNG have all expressed interest in starting their own sites.

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

••Scott Hess writes of the 1.0 version of the Stuart Terminal Emulator:

"Stuart, version 1.0- is alive and well. It is on the cs.orst.edu archive site, now in submissions, eventually in the binaries directory. I am very happy with this, and I am very tired. Go get it. Enjoy. I'm going home for a week. It has copy, paste, scrollbars, speed, everything an up and coming NeXT user needs.

This is the reward those of you who registered get - you now have a much better emulator. For those of you who haven't, yet, here's your excuse"

Stuart is shareware and is *not* free. If you use it, you must pay for it. For further information, contact Scott at scott@gacvax1.bitnet

••Meridith McGlon of Inforum writes saying that "**Networking/Unix '90** Sept 24-25 at Inforum, Atlanta, Ga. Co-sponsored by Inforum and Datapro...for more information contact Sue Frericks [sic] conference and exhibit manager at 404-220-2771 or 1-800-343-5046"

•• Media Station (version 1.11) is now shipping with a special developer price of $1495 (vs a normal business-land price of $2500). If you want to contact Imagine, Inc directly, call David Gregory at 313-487-1323.

•• Ariel is now shipping a *QuintProcessor*, a "new hardware product for the NeXT Computer designed to enhance the speed and versatility of the NeXT music and array processing kits. The QuintProcessor features **five** 27-MHz DSP56001 Digital Signal Processors -- a faster version of the **same** DSP that comes built in with every NeXT!" The announcement goes on to talk about its use for "array processing, music synthesis, speech recognition, and digital audio recording and mixing. Its design uses **four** of the processors to perform computation while the **fifth** manages **DRAM** memory, **SCSI** mass storage and interprocessor communication. Communication throughput is enhanced via hardware-supported interrupt-driven channel I/O between the main processors and the memory manager." Sounds like a Hardware-Junkie's dream. Contact Ariel at 201-249-2900 or ariel-west!jeff@ucsd.edu. [Ariel Corp, 433River Road/Highland Park, NJ 08904]. Oh yes, it comes with a special version of Bug56 just for the board. Sounds cool.

•• Ariel is also shipping their Digital Microphone at a developer price of $470 (vs $595).

••With a brief explanation for my benefit from Paul Morin (he explains : "A SIMM is a Single Inline Memeory module. When you buy a NeXT it comes with 8 megabytes in the form of 8, 1 megabyte SIMMs. With present technology a Next can be upgraded to 16 mega-

bytes with 8 additional 1 meg SIMMs. "), he sends me the following press release :

"Computer Care is excited to announces a new line of 100% NeXT compatible memory upgrades. The first, available immediately, is a $99.00, 1 megabyte SIMM (Single Inline Memory Module) which comes complete with installation manual and anti-static wrist strap. Each upgrade can be purchased in easily installed 4 megabyte groups. With eight of these SIMMs a user can upgrade a NeXT to a total of 16 megabytes of RAM.

Computer Care stands behind this memory upgrade with full technical support and a 5 year warantee.Computer care is located at Ford Centre, Suite 1180, 420 N Fifth St, Minneapolis, MN 55401. Telephone: (800) 950 - 2273 and (612) 371 - 0061."

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.

•Objective DB Toolket - 30 classes to links NextStep to Sybase. Professional Software, Inc./Lakeside Office Park,/599 North Avenue - Door 7/Wakefield, MA 01880/ (617)246-2425

•uni-REXX and uni-XEDIT - Unix version of IBM's mainframe/procedural language and general purpose editor./The Workstation Group, wrk/grp/6300 N. River Road/Rosemont, IL 60018/(708) 696-4800

•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.

•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" *SHIPPING*

•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

compiled by
Erica J. Liebman
erica@kong.gatech.edu

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

NEW THIS MONTH : San Diego , Minneapolis, Ohio State, Houston

---

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 We **desperately** need a local activities organizer. Please contact me if you are interested in pitching in for things Buzz

*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 NewsLetter aimed at the Business Community. There are member fees. BCS puts out the newsletter called *WhAT's NeXT.* **Boy** would the Buzzies Staff love a complimentary subscription*!*

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. ***See earlier article about the first meeting.***

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 for information on the **UC-Riverside NeXT Users Group**.

SNUG (San Diego) founded by **Nicholas MacConnell** of MindLinks, 619-481-7535 meets regularly at the UCSD Supercomputer Center : New users 6:30, general meeting begins at 7:00. SNUG puts out a great looking newsletter. I talked to Nicholas over the phone and he sounded like a great guy, so if you're in the SanDiego area, stop by or give a call. I think they meet 2nd Wed's of the month.

Texas (yee-hah!)

Dallas, Texas: **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. "

In **Houston**, contact the NeXT User Group coordinator **Dr. John Glover** at 713-749-1820 or glover@tree.egr.uh.edu for information about meeting times, etc.

Washington (the state)

University of Washington (with over 110 cubes) has a NeXT User group dating back to March 1989. Contact **Corey Satten** (corey@cac.washington.edu) at 206-543-5611.

Oregon

To get information on the **Oregon NeXT Users' Group**, contact **Bryce Jasmer** (of ftp fame) at jasmerb@cs.orst.edu or 503-758-5743.

Minnesota

The Minnesota NeXT Users Group (in the Minneapolis area) has the contact name of Mike Tie (507-663-4067 or mtie@carleton.edu). I know that "Great" Scott Hess, rapidly becoming a NeXT god, is up there, so this group probably has some neat NeXT stuff going.

Canada (O!)

Vancouver NeXT User's Group (in BC, CA) is headed up by **Lionel Tolan** (lionel_tolan@cc.sfu.ca). **Tom Poiker** of Simon Fraser Universit 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

The NeXT users in Colorado have just formed the **Rocky Mountain NeXT Users Group** (**rmNUG**).

Our meetings will be held in various places and feature the different members as they give presentations on what

they are currently developing.

David Hieb (davehieb@boulder.Colorado.EDU, (303)492-5720) is the leader of rmNUG and is a System Administrator for the Computer Science Department at the University of Colorado, Boulder.

Brad Green (green_bk@cubldr.Colorado.EDU, (303)786-0081) is the NeXT Campus Consultant for the University of Colorado, Boulder, and provides support and leadership for rmNUG as well.

We plan on providing an environment for the NeXT users in Colorado to share, learn and profit from the collective expertise of the group.

p.s. To all the NeXT users out there in Colorado that haven't been contacted by one of us : CALL ME at (303)492-5720 or email me at davehieb@boulder.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."

Buzz's Hint Corner

••Anybody ever look at the Makefile.depenancies file generated by a make depend? And get _very_ confused? I think its sort of nice to be able to look at the file, and understand what each section of your program depends on. But, alas, the Makefile.common depend does not do too well - there's too much just in there which never changes (/usr/include/* comes to mind).

Well, you are freed from your drudgery! I got tired enough of it that I went ahead and wrote a Makefile.postamble which just puts the dependancies that count, those of your files on your files. Included there is a Makefile.postamble which you just include in your directory, and then type make dep, and a Makefile.dependancies is created. Fun, fun, fun.

```
------  Makefile.postamble  ------

dep Makefile.dependencies: $(SOURCEFILES)

        -/bin/rm -f Makefile.dependencies

        $(CC) -M $(CFLAGS) $(CLASSES) $(MFILES) $(CFILES) | \

         awk '{ if( $$2 !~ /\/usr\/include\/.*/) if( $$1!=prev)
\

               { if( rec!="") print rec; rec=$$0; prev=$$1; } \

                else { if( length(rec $$2)>78) { print rec; \

                rec = $$0; } else rec=rec " " $$2 } } END { \

                print rec }' > Makefile.dependencies

------ chop, chop, chop ------
```

scott hess,GAC NeXT CC, scott@gacvax1.bitnet

••In the Berkeley Unix environment, there is an option to find(1) called "fast find." You run a program (/usr/lib/find/updatedb) and it creates a list of all of the files on your system in /etc/find.codes. At that point you can say:

```
find tif
```

and it will print out all files on your system that have "tif" anywhere in its name. The advantage of this scheme is that its extremely(!) fast, where

```
find / -name \*tif\* -print
```

(same command using traditional find syntax) will take several minutes to run. The disadvantage of using it is that any files created after the last run of updatedb won't be found. Since most of the files on a typical system are system files that don't change that much, this option can be used as a starting point to help find files.

I suggest that you put an entry in /etc/crontab to run the updatedb (which is a shell script, by the way) program in the wee hours of the night. On a networked system, updatedb is smart enough to recognize a directory that is NFS-mounted and will ignore it. That is, the contents of /etc/find.codes will be a list of files on hard disk only. --*JoEL McClung*

••To patch Emacs to use screen font widths, add these lines:

```
/* NXTextFontInfo appears to be broken for screen
fonts, so wait until now to get it */

{       Font *screenfont = [displayFont screenFont];

        if (screenfont) displayFont = screenfont;

}
```

between the call to NXTextFontInfo & the assignment to 'leading' in EtermView.m - *Mike Dixon, Xerox PARC*

To remove menu cells from submenus, try this :

```
- removeMenuCell:theMenu :(int)cellNum

{

        id matrix;

      [theMenu disableFlushWindow];   /* this gets rid of nasty
flicker */

        matrix = [theMenu itemList];

        [matrix removeRowAt:cellNum andFree:YES];

        /* ^ deletes the unwanted cell... */

        [theMenu sizeToFit];    /* adjust menu to new size */

        [[theMenu reenableFlushWindow] flushWindow];

       /* don't forget the above line, if you do you menu will */

       /* be hozed forever, and it will never hilight again!   */
```

```
        return self;
}
```

This should rip out any unwanted menu Items, providing you know it's number. If you only have the cell's title, you can do this:

```
- removeMenuCell:theMenu name:(const char*)cellTitle
{

        int             i, count;
        id              matrix, cells;
        id              cell;
        const char*     title;



       /* once again we will disable window flushing so that */
       /* we don't have screen flicker during the operation  */
      /* the reason this works, remember a menu is a subclass */
       /* of window                                          */

        [theMenu disableFlushWindow];
        matrix = [theMenu itemList];

        /* next we will need the list of cells... */

        cells = [matrix itemList];

      /* and the number of cells in the matrix (you'll see why)
*/

        count = [cells count];

        /* now we will walk the list comparing titles */

        for(i = 0; i < count; i++)
         {
           cell = [cells objectAt:i];
           title = [cell title];
           if(s && !strcmp(s, cellTitle)
            {
              [remove RowAt:i andFree:YES];
              break;
            }
         }

        /* now resize the menu */
        [theMenu sizeToFit];

        /* re-enable the window flushing */

        [[theMenu reenableFlushWindow] flushWindow];
        return self;
}
```

Bruce Henderson, User Interface KGB, Ashton - Tate NeXTeam

••Use Edit's "zoom" option on *.h files to get a listing of all the methods, with the commentary filtered out.- *Dave Joerg*

NeXT Support Answers,
Care of Doug Kieslar

Q: How do I change the master list of targets in Digital Librarian for everyone on my machine/server?(QA415)

A: Use Librarian to set up your targets in the way that you want to share with everyone on the system. Quit Librarian. This will write out your targets file. Copy your new version (~/.NeXT/targets/Targets1.0) to the master (/NextApps/Librarian.app/targets/targetFile).

Now, everyone on the system who wants this new target file needs to quit Librarian (to write out their old target file), and then remove it (~/.NeXT/targets/Targets1.0). They will lose any of their own specific targets if they do this.

Q: How can a program determine which CPU board revision is in the computer being used? (QA422)

A: Use the table() system call (not documented). Sample code:

```
#include <sys/table.h>/* Not really necessary, but nice for com-
pleteness */
#include <machine/table.h>
#include <machine/cpu.h>
#include <stdio.h>


extern int table(int id, int index, caddr_t address, int n_ele-
ments, int element_size);


main(argc, argv)
int       argc;
char      *argv[];
{
          intres;
          unsignedcpu_rev;
          interror;


          res = table(TBL_NeXT_CPU_REV, 0, (char *)&cpu_rev, 1,
sizeof(cpu_rev));
          if (res == -1) {
          perror(argv[0]);
```

```
          (void)fprintf(stderr, "%s: unable to get table valu-
e.\n", argv[0]);
          exit(1);
          } else {
          (void)printf("%s: table returned; res=%d,
cpu_rev=%d.\n",
          argv[0], res, cpu_rev);
          }
}
```

Q: How do non-NeXT machines deal with e-mail with long lines from our Mail application? (QA335)

A: When a message is sent from our Mail application the text is filtered. A space newline pair is added to the end of each line (not paragraph) if there is none already. Note that in this context the end of the line means the physical end of the line and depends on the window and font size. When Mail displays the message on the receiving end, it removes these space newlines pairs.

This feature was added to help users who use mailers that do not deal well with long paragraphs. So if you make your send window narrow enough, your lines will be short enough so that the newlines are inserted at the right place and receivers of mail with non-NeXT mailers will never know that you only broke your lines at paragraphs.

A bad side effect of this is that if you send your mail out with a space and a newline at the end of a line, the pair will be removed when displayed by the NeXT Mail application. Also, if you are mailing to sites that care about this feature, you need to make sure that your send window is the right size.

As an example, with a default font of Ohlfs 10, the proper size for a send window is what is just large enough to hold all the buttons. This will break lines at 80 characters.

Q: How do I communicate with Mathematica using pipes?(QA358)

A: This example is self-contained.

```
#include     <stdio.h>

     int     rdchnr, rdchnw, wrtchnr, wrtchnw;
     char    buf[80];

main()
{
     int     pid;
     char    *mmapath, *args[5];

/*
**   Open pipes for Mathematica to read and write.
**   The parent process writes to the pipe that Mathematica
reads,
**   and vice versa.
*/
     if (makepipe(&rdchnr,&rdchnw)) exit();
     if (makepipe(&wrtchnr,&wrtchnw)) exit();

/*
**   Split off the child process.
*/
     pid = fork();

     if (pid == 0)
     {
/*
**       ************************************************
**       This is the code for the child process,
**       which sets up for, and then becomes, Mathematica.
**       ************************************************
*/
          /* close the unneeded ends of the pipes */
          close(rdchnw);
          close(wrtchnr);

          /* set up the arguments for, and the full path name
of, Mathematica */
          // mmapath = "/NextApps/Mathematica.app/Kernel/math";
          mmapath = "/usr/bin/math";
          args[0] = mmapath;
          args[1] = "-noprompt";
          args[2] = "-run";
          args[3] = "ResetMedium[\"stdout\", PageWidth-
>80];$PrePrint=FullForm";
          args[4] = (char *) 0;

          /* install the pipes as Mathematica's standard input
and output */
          if ( dup2(rdchnr,0) == -1 )
               { perror("Error establishing read channel: ");
exit(); }
          if ( dup2(wrtchnw,1) == -1 )
               { perror("Error establishing write channel: ");
exit(); }

          /* start Mathematica */
          execvp(mmapath,args);
          perror("Error attempting to run Mathematica: ");
     }
     else
     {
/*
**       ************************************
```

```
**       This is the code for the parent process.
**       ***************************************
*/

          printf("hi there i got here\n");
            /* close the unneeded ends of the pipes */
          close(rdchnr);
          close(wrtchnw);

          /* absorb initialization messages harmlessly */
          fetchresult();

          givecmd("Expand[(1+x+y)^5]\n");
          fetchresult();

          givecmd("N[Pi,200]\n");
          fetchresult();

          /* tell Mathematica to shut down */
          givecmd("Quit[]\\n");
     }
}

int makepipe(rd,wrt)
     int     *rd,*wrt;
{
     int     piperesult, fildes[2];

     piperesult = pipe(fildes);
     if (piperesult) perror("Pipe creation failed:");

     *rd = fildes[0];
     *wrt = fildes[1];

     return(piperesult);
}

givecmd(cmd)
     char    *cmd;
{
     strcpy(buf,cmd);
     write(rdchnw,buf,strlen(buf));
}

fetchresult()
{
     int     count, i, ch;

/*
**   Read buffer after buffer of output from Mathematica,
**   until a NEWLINE is seen.  Because Mathematica is printing
**   everything in FullForm, all output is linear, and because
**   PageWidth was set to Infinity, no line breaking happens, so
**   a NEWLINE means the end of a complete output expression.
*/
     do {
          count = read(wrtchnr,buf,80);
          for ( i=0 ; i<count ; printf("%c", ch = buf[i++]) ) ;
     } while (ch != '\n');
}
```

Q: The Digital Librarian cannot find any lisp terms that contain asterisks. (QA 319)

A: This revised script allows one to search for keys containing asterisks, as long as they are escaped (a backslash is inserted before each asterisk in the query). In addition to the slashes that get inserted before *'s, this new sed script also removes leading package names of symbols before indexing them. Thus COMMAND-LINE-ARGUMENT-COUNT is indexed as COMMAND-LINE-ARGUMENT-COUNT, not SYS:COMMAND-LINE-ARGUMENT-COUNT.

Replace the file /usr/lib/indexing/lispdoc-keys with these contents:

----file starts here

#!/bin/sh

sed -n -f /usr/lib/indexing/lispdoc-sed $1 | sed -e 's/*/*/g' \

| sed -e '/..*:/s/..*:\(.*\)/\1/'

---- file ends here

Now, you need to delete the old index in CommonLisp and rebuild it. (The easiest way to do this is whether / NextLibrary/Documentation/CommonLisp is a target of Librarian. See the documentation on Digital Librarian for more information.)

Q: How do I ensure that a class is really linked in to an app? (Before release 1.0, we used NXLinkUnreferencedClasses.) (QA 394)

A: You have to add "-u .objc_class_name_MyClass" on the link line (where MyClass stands for the name of the class). You only need to do this if it doesn't get pulled in automatically for you. If you are using Interface Builder you need to add a Makefile.preamble file with the following line in it:

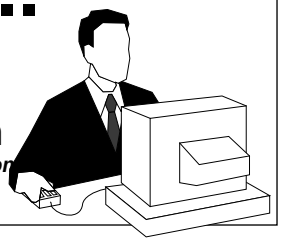
OTHER_LIBS = -u .objc_class_name_MyClass

Note that this class is probably archived in a library. In this case, the line in Makefile.preamble should read:

OTHER_LIBS = -u .objc_class_name_MyClass -lMyClasslib

The way I see it...

by
Bruce Henderson
atncpc!jupiter!bruce@NeXT.com



Bashersand Whiners. In the **NeXT** world, I find that both types seem to exist for the sole purpose to make "Real NeXTies" upset. I talk to other computer types a lot. I also read netnews avidly. No matter how far you run, no matter where you try to hide, one of **them** is there waiting for you.

For those of you who don't have usenet access, netnews is a global information sharing system, a forum for discussion on wide ranges of topics. A place for everyone and anyone who has access to air their opinions and share information. "**Comp-sys-next**" (usually spoken as a single word) is one of my favorites. Yes, its the place where all of the bellowing and chanting and good information circulates about NeXT Computers.

The partitioning of message traffic continuously amazes me. On the one hand, there is a fair amount of messages consisting of questions by people attempting to understand the NeXT or seek answers to their problems. People submit questions from time to time that I had previously wondered about or ask about situations I've found myself in. The fact that people from NeXT sometimes read this group, and even post to it speaks for its *validity*.

On the other hand, it seems that the rest of this space is devoted to two things. NeXT *bashing* and NeXT *whining* -- see comments above about NeXT user partitioning.

NeXT bashing comes in a variety of *smelly, unpleasant* packages. I secretly think that a lot of the academic types, a majority of the posters, find this machine threatens their ivory tower status. They take every chance they can to put it down. To them the NeXT is an *under-powered half-baked* attempt at a workstation. Clearly, **Steve Jobs** and Co. had nothing better to do with their lives than to try (unsuccessfully) to out do **Sun Microsystems**. They complained until recently that the machine was a closed and proprietary architecture because it did not *bow and worship* at the **Sun/AT&T X-Windows** altar. People who go on and on about the fact that "If they are ever going to succeed, they need to drop NeXT Step and start running X-Windows!" alarm me -- and there are *lots* of these people around the net.

X-Windows???? Are you kidding? Don't get me wrong, X has it's place. But that place is *not* as the primary windowing system of *my* NeXT. I doubt these people have spent *any* time using the NeXT, and are posting just to *bash* all of the people who (some of them) are staking a good portion of their careers on the success of this machine.

Most NeXT bashing comes from the **Macintosh** community. This is really sad. I would have thought that anyone *open minded* enough to use a Macintosh would be receptive to new and creative ideas. I developed on the Macintosh for nearly five years before joining a NeXT development team. I found most of the people I spoke to informed me that I am doomed to failure because the NeXT "just wasn't going to make it". I realized then that these *buffoons* who were discharging this expert advice were the **same boot licking toadies** who had told me an identical story some 5 years earlier. Except at that time I was venturing out on Macintosh software. Don't be fooled by their self-installed snobbery. These are the *same people* who crouched behind their **IBMs**, afraid that a "new way" would somehow erode their place in the corporate food chain.

Nowadays, every time I show my NeXT machine to

my industry friends, they always like to bash me because they think that the will never sell well, it's too slow, or the obligatory "It Sucks". Yeah.

Perhaps it's just a simple case of **Freudian penis envy**. The NeXT is the *bold step* that the industry could have, and should have made on it's own. But because of myopic and uninspired corporate leadership, no one had the guts to make a machine like this. It can be assumed that many companies have machines in the works that surpass the NeXT in every way. But the fact still remains that any such CPU can be seen as a sophomoric attempt to say "Me Too!".

Then there are the NeXT *whiners*. These people really drive me up the wall. They have casually used a NeXT machine at some point in their lives. But for one reason or another, they feel *compelled* to complain about the design of the machine. I can't tell you how many times I have heard "Well, it doesn't use C++, how come they didn't use C++? Were they stupid, or is this just Steve Job's Arrogance showing through again...." Another popular discordant eruption is "How come they didn't use the SPARC chip or some other RISC CPU?" or "They had better transition to RISC right away if they ever hope to be competitive!". These people suffer from the classic "Mr Know-it-all" syndrome. If they would ask themselves a few questions about the machine, and it's design, this idiocy would never come up.

For those of you who are curious, the NeXT machine was begun in ernest *long* before a useable (repeat useable) version of C++ was available. We should thank our lucky stars that the people at NeXT decided to create an exclusively object oriented user interface. But at the time there were very few choices for a language. Originally **Smalltalk** was heavily considered, but proved to be unsuitable. There was **object Pascal**.... But **Apple** owned that, and I imagine Steve Jobs wanted to avoid letting Apple know details of the NeXT. The knew that as a unix based machine that C would be a primary factor for building the system itself. And NeXT eventually decided to license the **StepStone's Objective-C** language.

On the front of **RISC CPU**s. At the time they decided what kind of machine to first build, the **SPARC** was not a viable choice (*still* isn't in my opinion), and the entire "RISC Revolution" was not even starting to happen. So NeXT made a very wise choice. If they wanted to be able to develop the system software and programming environment in parallel, they needed to go with a proven base CPU that was easily obtainable, off the shelf, and primarily, purchasable in a machine that would look a lot like the target CPU. It just so happened that the **Sun 3** series fit the bill. This allowed the software team to bring up NeXT Step and the entire NeXT system software before the hardware was even out of alpha.

Otherwise, don't expect Mr. Jobs to bring out a RISC based CPU too soon. For a small company selling a very complex product (software wise), it would be very ill advised to try and support two platforms that are not binary compatible.

In summery, let the crying babies cry, and let the bashers bash. Because in the end, only time will tell who is right and who is wrong. The NeXT is what it is, it isn't perfect, but it is a step in the right direction.

And that's the way I see it.