FORMAT OF PREFERENCE FILE.

SUBJ: * (R)
FROM: JamesP100 11/11/89 S#: 808495

Here is the format that I got from the MouseUp3.0 source. (which can be found in the libraries)

Byte#	Description
0	prefMaxSpeed- Max speed of mouse-cursor.
1	prefMinSpeed- Min speed of mouse-cursor.
2	prefAcceleration- Desired speed change.
3	prefForeColor- Upper nibble.
4	prefBackColor- Lower nibble.
5	prefMouseColor- Mouse-cursor color.
6-68	prefMouseShape- Mouse-cursor picture data. (63 bytes)
69	prefBorderColor- Border color.

Hope this information helps.

(: JamesP100 :)

AUTOEXEC FILES.

SUBJ: An easy way to do it would (R)
FROM: WillJ14 12/04/89 S#: 95911

take a disk sector editor and change the filetype designation byte from application to autoexec. I'd tell you exactly how to do it but I don't have my GEOS P.R.G. handy.

One thing to be aware of when doing this is that your input device (unless using a joystick) and your printer won't work. This is because the application would be running before the desktop, which means that the input and printer driver isn't loaded. One way to get the input driver loaded is to use the MOUSEUP program located somewhere in the GEOS libraries (use search to find it!).

How this helps (somewhat).

Unless Jim or someone else responds to this, I'll leave all the info tomorrow.

-Bill

*****Unless you know what you're doing with a disk sector editor (or have explicit directions), practice on a disk with NOTHING on it BUT the application that you want to change. One mistake could render the disk USELESS!!!!*****

@@@@@NEVER MODIFY A FILE ON A MASTER DISK...ALWAYS USE A COPY...A MISTAKE COULD BE FATAL@@@@@

-::-

SUBJ: Major problem... (R)
FROM: GEOREP JIM 12/04/89 S#: 472416

at $5000 in memory during bootup and auto-exec management is the section of code some people call the 'BootTop' - it's the section of the Kernal that starts up GEOS.

Big applications like geoWrite and geoPaint will overwrite this area, and even though they will return properly to the BootTop, the problem is that the BootTop will no longer be there, because it was overwritten by your auto-exec program!!!

If the program you want to auto-exec in any way modifies the code from $5000-$5FFF, you must first dump that area to disk (use a swap file) and load it back in again before quitting.

Basically what I'm getting at is that you need a separate program that will in turn load your application. If you just change the filebyte of geoWrite or geoPaint (or any other big file), even though it will auto-exec and run ok, when you Quit the program, you'll crash the computer.

Plus, you have the problem of the Printer and Input drivers not being selected.

-Jim :)

SUBJ: AUTO-EXECs question (R4)
FROM: ILLINI70 11/25/90 S#: 19377

Is there a size limit on AUTO-EXEC files? I have one that works fine when called from the desktop but locks things up when it finishes as an A-E.

The program currently uses space from $400 up to $6000 (the program itself only goes up to $1400 or so but various buffers extend - and are used - up to $6000).

Any other A-E pitfals I'll need to know about? My first A-E program (QTAuto) works with no problems. Thanx
John

-::-

SUBJ: Your problem is that... (R)
FROM: Michael530 11/25/90 S#: 523156

... your data section is trashing the GEOS Boot routines which reside from $5000-$6000. For an auto-exec to work that area must be preserved.

-::-

SUBJ: Also, you asked for ... (R)
FROM: Michael530 11/25/90 S#: 495054

...other pitfalls-

1) Auto-Execs are run in the order they are found, so if you need full disk support (more than DRIVE B) then your A-E must be after CONFIGURE in the directory (I WOULD ASSUME),
BUT
2) If you patch the kernal than your A-E should go before Configure. This is so RBOOT will reflect the changes.

3) If you need the MOUSE than you'll have to load InputDriver yourself.

-::-

SUBJ: And (R)
FROM: Crycket 11/25/90 S#: 461781

it's better if you load the preferences too, if your autoexec will write to the screen at all. Use the firstBoot flag at $88c5 to see if driver/prefs need to be

loaded. That way you won't waste time
when your program is run as an app.

USING PSECT.

SUBJ: no. (R)
FROM: Crycket 11/06/89 S#: 831650

my code really isn't broken up. I'm just trying to find/set/follow some kind of convention.

1. Each file requires a zero page declaration before first pass through the code. I therefore .include the same set of zsect in each file, only I .noeqin and .noglbl every file after the first. This is similar to the sample vlir application. All the files use the same zpage variables, so it makes sense to me.

2. For the assembler to code, this zsect gets followed by a .psect in each file as well. For assembly purposes psect starts at relative 0 in every file, no matter how many .psects there are. In fact, the book says psect is the default. In my case, though, .psect is required to shut off the .zsect processing, so it follows the .zsect in each file that has one. opcodes,.bytes and .words following the included zsect would generate a context error without a psect.

3. The linker should then chain seq files in the order given in the .lnk file. If no .psect is specified the chain starts at $400. Each subsequent file is appended where the previous file terminated. The psects in the source should have no influence on this.

4. With the exception of the main loop, all my files hold a separate subroutine. When I examine the symbol table I see ALL my routines starting at $400, which is where only the main loop should be starting. If I'd written the linker (out of my league to do so, but not to design one) it would be impossible for the program counter to reset while linking a seq file. Obviously in this case it is possible. How is it happening?

I'm not inclined to believe it's because I switch sects in my source. I assume this is intentionally permitted, or it would defeat the purpose of relative module code. Go back to what HAL said

about assembling, debugging, documenting and then reusing modules.

Two other things: counting pixels, not by using the ruler, i measure my icon 24*21.

Also, ramsect is an effective way for any programmer to access more memory without increasing file length. I think it's a good idea. Memory's not cheap in a 64. SUBJ: zsects
FROM: WildBill00 11/12/89 S#: 404729
Just to clarify a bit...
ZSECTS are NOT required in an assembler file. If have written several and have yet to use a zsect. ZSECT forces page 0 assembly, RAMSECT identifies variables WITHOUT reserving space INSIDE your application (although you can, if you want) and PSECT simply says "here's my code". It is the LINKER that worries about where stuff goes and should link the routines in the order given. The assembler really doesn't care about where the code will end up as it leaves references open to be filled in by the linker.

You are quite right regarding the manual's "how to" information. A little more would go a long way. I got things running by trial and error (& lots of ERROR!)... keep plugging!

SUBJ: .PSECTs.... (R6)
FROM: GEOREP JIM 11/06/89 S#: 440395

HAL, you should ALWAYS begin each of your source code files with a .psect. (Unless you have a .zsect, in which case you'd put the .psect after it)

I've made the mistake before of leaving the .psect out of a few of the source files, and it causes bugs to happen that are VERY hard to find, and don't seem at all related to the .psect - until you add it in and the bugs go away!

But remember, just a .psect - no argument.

-Jim :)

SUBJ: sex (R)
FROM: Crycket 11/07/89 S#: 438562

uh, I mean sects. I think I've learned more about them in the last couple of days than alot of people ever do.

From what the book says, HAL's probably right about not needing .psects. The assembler will default to psect. The only genuine requirement would be to terminate a ramsect or zsect.

Inside the file you can switch freely from psect and ramsect. The assembler figures it all out, sorts the symbols and passes only the total number of psect and ramsect bytes to the linker.

Zsect is used ONLY by the assembler, which needs to identify zero page addressing on pass1. Of course, the linker still sees the symbols from zsect, but the associated object is fixed by the assembler. In other words, if you redefine a zsect label to another address, you will need to reassemble, since the linker never relocates zpage addresses. This is not the case with ordinary ramsect symbols.

I'm surprised more programmers don't use the zsect and ramsect. Equates require more planning than zsect, and .block takes no disk space in ramsect. SUBJ: Well, (R) FROM: GEOREP JIM 11/08/89 S#: 851259

you may be right about leaving out the .psect, but I know for a fact that it has caused problems with me in the past, so I figure, why take a chance?

About using .zsect/.ramsect, here's my opinion. I DO use .ramsect, to hold all my variables. And anyone who writes a desk accessory should DEFINITELY use .ramsect, for setting up the screen buffers.

About .zsect, it's too much of a bother to me. I just use a0-a9. Most (if not all) of the free zpage space GEOS gives you already has a pseudo-reg assigned to it.

-Jim :)

SUBJ: Your both right, (R)
FROM: HAL 9069 11/10/89 S#: 445620

Jim, the zsects isn't really needed because of the zero page registors such as the a and r regs being predefined for you in the sym files. And yes, the .ramsect does come in handy for saving space and keeping the application small. Myself, I'm a creature of habit. I'v programmed on systems in the past that if you've used .ramsects or it's equiv, you allways ran the risk of 'clobbering' some important system variables at run time, especially on small memory systems. For you young guys, there was a time when 8 or 16K was a LARGE system!! (I'm talking PDP and Nova machines) Your right Jim, why take the chance, just use the .psect in every module. And yes, the linker does default to .psect.

HAL 9069

SAVING dlgBoxRamBuf

SUBJ: desk accessory (R1)
FROM: JohnW53 12/13/89 S#: 401780

I am having trouble recovering to my application from a desk accessory. Sometimes I am able to recover the screen, but the mouse and keyboard freeze up.
What sort of things MUST I do for a full recovery, say, from inside a GEOwrite file?
Ive tried using the RstrAppl, but alas, no luck.
Any suggestions... Ive been messin around with geoprogramming for about a year now.

-::-

SUBJ: Well,
FROM: GEOREP JIM 12/14/89 S#: 404862

I need to know more about what your DA does. One big thing that a lot of people don't know about (and get really tripped up on) is that you CANNOT have dialog boxes in a desk accessory, unless you first save the buffer area. The area used to stash the GEOS system variables is the same for DB's and DA's. If you use a DB from within a DA, you'll trash the original data that was stored in those buffers.

So are you using a dialog box? If not, what does your DA basically do, and what kind of routines are you using (and memory locations playing around with)?

Basically, anything that doesn't say is saved when DB's or DA's are loaded (fro the GEOS Programmer's Reference Guide) is messy stuff to tamper with from a DA/DB. If you change those locations, you have to restore them before you exit.

-Jim :)

SUBJ: dlgBoxRamBuf (R1)
FROM: Hellio 05/06/90 S#: 467151

I noticed in another message it said something about saving dlgBoxRamBuf before calling a DB. I looked up page 428 in the manual and found the information. I have never saved any of this info and my DBs have never given me any trouble. Please explain.

-::-

SUBJ: The only time you
FROM: GEOREP JIM 05/06/90 S#: 850538

have to save that buffer is when you are:

(1) Calling a DB from within a DB - very messy and not recommended at all

(2) Calling a DB from a DA. (Done a fair amount)

The reason is that Desk Accessories also use the dlgBoxRamBuf area to save the code they need. If you called a DB within the DA without first saving it, you'd wipe out the old info, and the system would possibly crash when the DA terminated.

-Jim :)

SUBJ: Thanks Peter!
FROM: GEOREP JIM 03/06/90 S#: 800090

Also, remember that AutoLoader loads PROGRAMS only, not data files. If you specify AUTO COPY in AutoLoader it will not work.

Here's how to do it:

(1) Create your list file in Batch Copier
(2) Save it as "AUTO COPY" on disk.
(3) Copy AutoLoader and Batch Copier to your boot disk
(4) Open the Info box on AutoLoader
(5) Change the first line to read "Batch Copier" or whatever the filename of Batch Copier is on disk. NOT AUTO COPY.

When it boots, AutoLoader will run Batch Copier. Batch Copier will then look (all by itself) for AUTO COPY and copy it if it's found. All AutoLoader does is get Batch Copier running, it's Batch Copier itself that takes care of the auto-exec copying and searching for AUTO COPY. Note that like Configure, the screen will not change if Batch Copier is run during bootup, so you will not see anything different on the screen.

Good luck! Post back here if you can't get it to work.

-Jim :)

SUBJ: symbol table overflow (R3)
FROM: SIDist 08/23/90 S#: 837221

I know this has been discussed before, but would someone please review it? I divided a .seq application into 5 modules and now when I invoke the linkage editor I get the message "Overlay module symbol table overflow" in the second mod. What overlay, does he think I'm trying to do a VLIR? My .lnk file matches the example in the geoProgrammer ref. guide for .seq files. The error message explanation suggests adding some .noglbl and .noeqin's. Each mod starts with:

```
.if Pass1
    .include  geosSym
    .include  geosMac
.endif
```

Can someone please tell me where the .noeqin and .noglbl's go and why? Thanx 10E6.

-::-

SUBJ: No problem, (R)
FROM: MichaelS30 08/24/90 S#: 800222

Consider that there are hundreds of symbols in geosSym. No consider that each of your 5 modules has a copy of each of those symbols. It adds up quick.

What needs to be done is to include the symbols in the Assembly process but to exclude them from the linking preocess. If you use geoDebugger than keep one of your 5 link modules the same as it currently is and change the other 4 to-

```
.if Pass1
.noeqin
.noglbl
.include geosSym
.include geosMac
 .
 .
 .
.eqin
.glbl
.endif  ;Pass1
```

The above will allow geosSym's symbols to be used by geoAssembler, but they won't be passed to geoLinker. If

your using geoDebugger, however, you probably want one copy of the symbols to be sent thru geoLinker so they would be available to geoDebugger.

-::-

SUBJ: If you don't use (R)
FROM: GEOREP JIM 08/24/90 S#: 792682

geoDebugger then you can change ALL the files to read

```
.if Pass1
 .noeqin
 .noglbl
 .include geosSym
 .include geosMac
 .eqin
 .glbl
.endif
```

And none of the symbols will be passed. It will speed up assembly and linking a bit and save some disk space... the drawback is that if you use geoDebugger you won't be able to use the default GEOS symbols.

But for us 128 programmers that's no big deal, since we can't really use geoDebugger anyways.

-Jim :)

-::-

SUBJ: Or...
FROM: ILLINI70 08/24/90 S#: 826364

Delete the Symbols and Macros that your program isn't using forming a tailored copy of Sym and Mac for that program. I had to do this fir QwikTop 3.1 'cause it was getting kinda big

John

SUBJ: geoLinker (R2)
FROM: SIDist 08/27/90 S#: 438246

I'm having a bizarre problem and am wondering if anybody else out there has experienced it. I have a large .seq program divided into five source files, individually assembled. I got that to link O.K. with the .noeqin/.eqin mentioned in response to my previous post. But then I made a few changes, re-assembled everything, and the linkage editor went insane! As far as I can see, all my labels are now being flagged as both "expression cannot be resolved" and "symbol defined more than once". In fact, the first line in the error file is "symbol defined more than once" referring to the name of the first .rel filename in the .lnk file! And the line number he gave in the .lnk file doesn't even exist (10, there are only 8 lines).
It's hard to even tell what's going on, actually, because he seems to stop counting at 98 errors (and actually reporting the errors long before that!).
I lopped off the fifth file, and then I only got the error messages one would expect, so I added it back in a page at a time, re-assembling and linking as I went. All fine till the fifth page. So O.K., I think I have it narrowed down, but I put the sixth, etc. pages back in and it bombs again. Is this something to do with the size of either a source file or relocatable module? Just sign me Tearing My Hair Out In Milwaukee.

-::-

SUBJ: I had the same... (R)
FROM: ILLINI70 08/28/90 S#: 398737

problem with QwikTopV3.0! There are 5 files to be assembled and I got the same series of error messages you describe. After MANY hours of tearing my hair (only the grey ones!) here's the solution I came up with. TOO MANY SYMBOLS for the linker to deal with! (My 8th grade English teacher would have a fit - 2 sentences in a row ending in with) Anyway instead of going thru ALL that code to see where I could safely put .glbl/.noglbl and .equin/.noequin I just went thru a copy of geoSym and

deleted all the extraneous symbols (ie
the ones I wasn't using in QTV3.0).
Needless to say that solved the problem
and QwikTop V3.1 is on the street :)
    Hope this helps...
                John
                 -::-

SUBJ: I've had similar
FROM: GEOREP JIM 08/29/90   S#: 91072

probs before where it will list one of
the .rel files as an unresolved symbol
too. Usually it's just a quirk and if I
modify the program a bit I can get it to
go away.

You probably do have too many symbols,
there is a limit but I can't remember
what it is... :(

                -Jim   :)

SUBJ: Phantom errors.... (R1)
FROM: Dibief 08/29/90 S#: 411103

Occasionally I run into a batch of phantom errors generated during assembly. Maybe a dozen or so "branch out of range" errors will be generated for a particular page. Sometime the page numbers are out of order, duplicated or just wierd (page # [3).
I can isolate the problem down to one line of code, in one case just a label that had nothing branching to, from or over it. In other words, the errors are false.
This almost always happens after I've added a few lines of code, sometimes not even on the page where the false errors occur.
Once I was able to solve this problem by throwing in a bunch of "nops", but this last time that trick didn't work.
I don't run into this all the time.
Thanks,
Dave Ferguson
DiBieF

-::-

SUBJ: Same here.
FROM: GEOREP JIM 08/29/90 S#: 90879

It's weird and it doesn't happen much but it does hit me every so often and it's a REAL pain. I usually get stuff like Hidden error (when there is no error in the first place, and the only thing hidden from pass 2 is geosSym and geosMac which never changes), but I get the .rel files showing up as symbols, etc.... there is something flaky in there but I really don't know what.

-Jim :)

-::-

SUBJ: Phantom solved (kinda) (R1)
FROM: Dibief 08/29/90 S#: 818112

Well, I think my phantom errors must have been caused by some kind of assembly or reference over memory-page boundaries. I solve the problem by taking some code from BEFORE the error page and putting it BEHIND the error

page. No problem. Vewy Wied!
Dave

-::-

SUBJ: Yep,
FROM: GEOREP JIM 08/30/90 S#: 521725

I was able to make my errors like that
go away once by moving code around.

I've really had some weird ones before.

-Jim :)

REU LAYOUT

SUBJ: georam map? (R4)
FROM: Mariner4 09/04/90 S#: 5990

I would like to be able to use parts of the georam that are not currently used. Is there a map of georam avalable that shows what pages are used for the various disk drive configurations and for some of the popular applications so I can stay compatable?

-::-

SUBJ: It's all... (R)
FROM: GeoBasic 09/05/90 S#: 789620

used. Actually only the first bank is standard, the rest are drive dependent. I've got a map of bank 0 around here somewhere, I'll look for it.

-::-

SUBJ: Bank 0... (R)
FROM: GEOREP JIM 09/05/90 S#: 395328

is reserved for the system.

*$0000-$78FF - MoveData space
$8300-$B8FF - disk drivers A-D
$7900-$7DFF - holds GEOS global RAM
area $8400-$88FF by ToBasic
$7E00-$82FF - loaded with reboot code
by Configure (runs at $6000)
*$B900-$FC3F - GEOS KERNAL

* on GEOS 128 the Kernal is bigger and extends down into the DMA area, I don't remember the exact limit but on GEOS 128 only $0000-$3800 or so is free, it's slightly under 16K. $3800-$78FF portion of DMA/MoveData holds the back RAM portion of the Kernal.

As for other free areas - you will need to write a routine to see how many banks are present (via ramExpSize) and then examine each RAM/Shadowed disk to see where it is (ramBase) and how big it is (use a lookup table) and then allocate the used banks, then look to see what is free. This is what I did in geoWizard. Note that geoWizard will steal one whole 64K bank for itself.

-Jim :)

-::-

SUBJ: There's got to be (R)
FROM: Crycket 09/05/90 S#: 482887

some space in there somewhere. The SuperDebugger hides in it (no small task) and obviously geoWizard lives somewhere in there, too.

The real question is: how do you access these non-dos banks with standard system calls? For example, if the SuperDebugger is hiding from the dos, how does MoveData find it? It's GOT to be through system calls, or it wouldn't be compatible with both the REU and the geoRam.

What's the secret, Jimbo?

-::-

SUBJ: DMA/MoveData
FROM: GEOREP JIM 09/05/90 S#: 395429

is where SuperDebugger sits. As well as overwriting the 128 Kernal, because it extends into the DMA area.

The standard StashRAM/FetchRAM/SwapRAM/VerifyRAM calls can access any of the REU, including Bank 0. You can use any of the REU you like to, the point is you're not supposed to.

I access the geoWizard bank like any other, and I can't protect it from being stolen by something else as there's no way to allocate individual banks, luckily I think geoWizard is currently the only thing residing in an individual bank by itself. If anyone does anything else let me know, I'll give you info on where to look to see if geoWizard is running and what bank it is using.

-Jim :)