Mops 2.3 release notes

Here is the usual "release notes" describing the changes from the previous Mops version, so that those already using it don't have to wade through all the documentation (or have to read right through the huge manual!!!).

As with version 2.2, 2.3 has very few changes—both because it is relatively stable, and because I've needed to spend whatever time I could on the manual.

This is the big news with this release—the full manual is now done! I hope you're as relieved as I am.

The size of the release has meant that as well as keeping the manual separate, I'm now releasing the source and binaries separately. But if you get the source, you don't need the binaries since you do get the nucleus, and the README.1ST file contains instructions for compiling up the rest of the system, which should only take a few minutes. The binary release is aimed at people who want to give Mops a try without having to do a gigantic download first.

A few bugs turned up in 2.2, and I posted the fixes at the time on CompuServe and internet (comp.lang.forth). These fixes are now, of course, incorporated in this new release.

ANSI Standard

This is now almost with us, after many long years of waiting. There is now a file ANSI which, if loaded, is intended to make Mops an ANSI-compliant system. Whether it really does is a matter for some conjecture, but it certainly will come close. Please let me know if you have any problems. See the comments at the beginning of the ANSI file for the ANSI features supported. But note, if you want full ANSI compliance, you won't be able to use Mops' OOP features, since under the ANSI standard words ending with colon are ordinary Forth words, whereas Mops uses this syntax for selectors. You can disable this feature, even with ANSI loaded, by setting the Value SLCTRS? true, which will give you an "almost-ANSI" system which still lets you use selectors.

We have made a few other changes to Mops in line with the coming ANSI standard—these changes won't affect many programs, I hope. EXPECT has now been replaced by the almost-equivalent ACCEPT. ASCII has been replaced by the state-dumb CHAR and [CHAR]. (You can still use & as before.)

We have now added CATCH and THROW, and our error handling is now based on these words. This change shouldn't affect existing code, however, since when an error occurs the error number is THROWn, but without a CATCH in place the default action is taken, which to do the same as previously.

We have also brought <#, # and #> back into line with regular Forth practice in regard

to their stack effects—they now operate on a "double number"— except that the double number really isn't. Since we don't have 64-bit arithmetic in the basic Mops system, we ignore the top cell. If you load the LongMath file, you get 64-bit arithmetic, and then perhaps we should implement a version of <# etc. which really will operate on a double number—but we haven't done so yet. I doubt this is very significant. Anyway, I hope this change doesn't bother anybody too much. It was better to have standard stack effects, I decided.

Another change we have made in line with the standard is the meaning of ALIGN. The standard defines this as doing what our word ALIGN-DP does, and defines a word ALIGNED which does what our ALIGN did. We have made these changes, and retained ALIGN-DP as a synonym for the new ALIGN. If your code makes extensive use of ALIGN, you could preface it with : ALIGN aligned ; to restore the old meaning.

EventLoop

This is a new word which will give you an event loop which will be quite adequate for most uses. If fWind is frontmost, keys are interpreted. If some other window is frontmost, keys are sent to the window via its KEY: method. This word will let you test a window which does something special with keys, and still get back to normal Mops interpretation by clicking on the Mops window fWind. Thanks to Doug Hoffman for this word.

Quick Edit and the Mops Glossary

Many thanks too to Doug for this, which is now bundled with this Mops release. It gives you a full on-line glossary while you are in the editor. Just highlight a word, and choose Reference under the Special menu, or hit command-3. Also, in the Glossary window, start typing the name of the word you want to look up, and you will be taken there. I've now started to use Quick Edit for my own Mops work, and recommend it highly.

Coming attractions

These are ideas that I may incorporate into future versions of Mops. However I make no promises as to when. I am only too well aware that this list has remained almost unchanged for the last 12 months (but see the last item!)—however now the manual is finished, we may be able to make some progress here.

*C++ provides a "class instance variable" idea under which an ivar can belong to a class itself, rather than to a particular object. This idea is fairly useful, I gather. It would be easy to add to Mops, and so will probably turn up in the next release.

*We will probably provide an "AppleEvent Object" class to support the protocol

defined for the operands of AppleEvents (System 7). This protocol is object-oriented, and so should fit into Mops quite nicely. Another user has indicated he may be able to do this, so we'll incorporate it when it becomes available.

*We still haven't implemented the menu items "List objects" or "list Classes". We may implement these as they were in Neon, or go to a more general class/object browser. Already you can type SEE xxx and get a structured display of whatever xxx is, including its superclasses if it's a class or an object. Maybe we'll enhance this. Anyway stay tuned.

*I hope soon to give some thought to the "command" class in MacApp which makes "Undo" so straightforward, and also to the idea of a command hierarchy as implemented by both MacApp and TCL, in which various classes in turn can have a shot at an incoming command. I expect something in this area will turn up in Mops soon—well OK, OK, eventually.

*Apple has hinted that in future it may "encourage" a separation between an application's code and data. I'm glad I didn't rush into doing anything about this, because now I don't believe it will ever happen on the 680x0 processors. In fact System 7.1 itself alters code segments on the fly (the new SANE package patches the point of call to speed subsequent calls by bypassing the trap dispatcher). When the new POWER machines appear, things may be different (yes, a lot of things **will** be different!!). Hopefully I'll still be around to update Mops to output native POWER code, and then I'll tackle this problem.

By the way, I'm now working at Mac programming, for a registered developer, in my day job (does this now make Mops a Professional Product™??—so this does give me access to information under non-disclosure which of course I can't disclose, but which ought to help me not bark up any wrong trees with future Mops development. I hope so, anyway.

E-mail

If you need to contact me, the quickest way is via e-mail:

Internet (best): mikeh@kralizec.zeta.org.au
CompuServe: 100033,3164

My mail address is

Michael Hore
54 Frederick St
Sydenham NSW 2044,
AUSTRALIA.

I do hope you enjoy Mops.