

# EDITmOR

Version 1.0

by Robert S. T. Gibson

## **General Use:**

### **WHAT IS EDITMOR?**

EDITmOR is a desk accessory (DA) which allows you to extend your editor (or even your word processor). Being an extendable DA, if you are a programmer, you can write your own external commands (ECMDs) which EDITmOR will accept and run for you (see Programming below).

### **USING EDITMOR**

To install EDITmOR, follow the steps you would normally use when installing a DA. If you have Font/DA Juggler or Suitcase, you can simply load the DA suitcase with that utility. If you do not, or if you wish to group your DAs together, copy EDITmOR into your System file or another suitcase using Font/DA Mover.

When you see EDITmOR in your Apple (DA) menu, you will know that things worked (at least partially). If the following steps work, then you will know that things went well.

To use EDITmOR in an editor, launch the program. Then, choose EDITmOR from the Apple menu. If you are using MultiFinder, you should hold down the option key while making the choice. This will open the DA in the editor only, which will allow it to get the text from your editor. Otherwise, since MultiFinder DAs are treated like separate applications, things will probably not work properly.

When EDITmOR opens, you will see a menu appear at the end of the menu bar (or just before the MacroMaker menu). Selecting the menu reveals several different possible choices: Help, Close, and hopefully three other items (Lower case, Reverse case, and Upper case). If you do not see these items, then either loading was not proper, or these have been removed from your file by someone. If you see more, then someone has added ECMDs to your file.

The Help item displays a dialog with a popup menu and scrollable text. The current help text title is displayed in the shaded box (the popup menu hot spot) and the actual text is displayed in the scrolling area.

The Close item simply removes the EDITmOR menu from the menu bar and releases the memory it occupied.

The three other items are the external commands which edit the text. To use them, simply select the text you wish to edit, then select the item from the menu. Before you do this, you should save your document. Aside from being a good practice anyway, you cannot always be sure what will happen when you make a menu selection from such a DA as this one. The ECMDs are not always reliable. Don't get scared by this — it's just a warning. When the selection is made, your window will be disabled for a second while the copy is made, and the command will be applied. If things were successful, you should see the new desired text instead of the old selected text. Note that only selected text is edited. The commands I have bundled with this program (Lower case, Reverse case, and Upper case) are fairly self-explanatory. See the help file for a line or two of information about them. Remember that when you select any item, the selected text will be placed into the clipboard. This version of EDITmOR does not save or replace its contents when a command is executed. Make sure there's

nothing important in the clipboard before you select a command. Another important point here is that styled text is converted to plain text by my three external commands.

## **Programming:**

### **MAKING ECMDS**

ECMDs (Editmor CoMmanDs) are very similar to XCMDs, MDEFs, LDEFs, and other code resources. They consist simply of the compiled code in an ECMD resource. Any language capable of compiling definition resources should be capable of creating an ECMD resource.

You are not passed any parameters by EDITmOR in this version. So far, I have seen no need for them, so I have left them out. Your functions should not require any parameters.

Your code is responsible for its own scrap management. That is, you should get your own text from the scrap, and put it back when you're done. If everything was successful, return TRUE from your function, and the new scrap will be pasted back in. If you are going to be converting text in the scrap, also remember to make sure that what you want is indeed there. You do not always have to edit text; you can display alerts, invert the screen, or do whatever you like. You don't even have to edit TEXT formats; PICTs and other types can also be edited. If you want to write a function that disassembles a PICT from the scrap and pastes it into the current text window, then you can — or you can assemble a PICT from the scrap and paste it into a graphics window. As you can see, you aren't limited to editors or changing text.

It is possible to convert styled text as well as normal text. All you must do is check the scrap for different styled text types (eg MacWrite's MWRT or styl resources, Word 4.0's RTF, etc.). If you know their formats, then it is fairly easy to do the conversion. Remember that if you delete any bits of text from the handle, you must change the style record accordingly.

When you write an ECMD, you must give it a name so it is displayed in the menu. It should not begin with a period (.) or a percent (%) sign, lest it be omitted from the menu. Usually, any ID is fine. At the moment, they appear in the menu according to alphabetical order. I intend to fix this in a future version, allowing similar functions to be grouped together or placed in a special order, so keep it mind when assigning IDs.

To write an ECMD in Pascal, write a UNIT of any name, bearing the function name of your main code in the INTERFACE. The implementation can contain any number of routines, but the main code will be the only one called directly by EDITmOR. You can, of course, call these routines within your command with no problems. The C example (Reverse case.c) can be easily translated to Pascal. Remember that your main code must be declared as a function with no parameters, returning a Boolean (eg. FUNCTION ReverseCase : BOOLEAN;).

ECMD writing in C is also fairly simple. Your main code should be titled main(), and must be declared as returning a Boolean. Also, keep in mind that it must be declared as pascal. That's what it is called as, so that's what it should be. See the source code example for more details.

### **EHLP RESOURCES**

The EHLP (Editmor HeLP) resource stores the help or about text you wish to assign to your external command. Simply write your information in any text editor or word processor (without any styles — yet), copy the text, create a new EHLP resource in ResEdit, paste your text in the window (the text part rather than the hex area), assign the resource the proper ID (just in case I do fix that menu organization thing), name it, and save it. You should watch it here, as some versions of ResEdit are prone to crashes around this area. Save your help text before doing this, as the clipboard is lost if you crash badly enough to force a restart of the machine.

The EHLP resource's name will be loaded into the menu accordingly, and the text will be displayed when it is selected.

**Copyrights, Disclaimers, Requirements, and Credits:**

EDITmOR, Copyright © 1989 by Robert S. T. Gibson.  
Portions copyright © 1986 THINK Technologies, Inc.

I cannot be held responsible for any damage to software, hardware, text files, brains, etc. that might (or even definitely did) result because of use or misuse of EDITmOR. Use it at your own risk. Hopefully, it's won't turn out to be very risky at all. I have done quite a bit of testing and found no real bugs. See below if you do.

Please pass it around as long as it's not being sold. Commercial distribution prohibited without proper licensing from the author.

No monetary compensation is necessary for use of this program. I have only two requirements to ask of you:

- If you write any new and useful features to add to this, please pass them around. You can put all your about stuff in the Help dialog, so you will be getting credit for them. I would also appreciate it if you would let me know of your commands and how I could locate them. Source code would be great too, but not necessary.
- If you come up with any good ideas, bad bugs, or constructive criticisms, please let me know. I'm always glad to hear them. I want to improve this program as much as possible, as I use it too!

Special thanks must go to Joel McNamara, whose article in MacTutor's July, 1989 (Vol. 5 No. 7) issue entitled "Extend your Favorite Editor" inspired me, and basically showed me how to do it. I improved the code and techniques on how to get the selection into the scrap, and took it from there. Thanks very much.

Please do not change the about box or the according help resources.

Please send your comments to:

Robert S. T. Gibson  
RR#1 Carrying Place,  
Ontario, Canada.  
KØK 1LØ

Compuserve (CIS): 71261,2236  
GEnie: J.GIBSON4

