

2 *Release Notes and some History*

This chapter contains the release notes to all the ClassEditor versions. It documents new features, the development steps taken, major bugs that have been fixed and bugs that are known.

Version 0.4

This version is mainly a maintainance release. It fixes some nasty bugs and introduces code formating which is very nice to have while working with ASCII sources^{1/4}somehow it makes them more useful then RTF sources.

There are still many nasty rough edges but I wanted to get those major improvements out to you as quickly as possible.

New Features

Even more features out here. But many more are still missing.

- **Better Emacs Support.**
A new Emacs Text object adds better support for code editing, brace checking etc. pp. (originally by Glen Diener)
- **ASCII Colorization.**
The same EmacsText is responsible for the colorization of the sources. See the *Known Problems* on how to change the fonts and colors.
Colorization seems to work with RTF text too but is not applied to it to prevent corruption of enrichments done by hand.
You can trigger reformatting by using the "Format code¼" menu item. This has not been tested very heavy ± so be careful. It *always* reformats the *whole* text, not only the selection!
- **Check Documentation.**
This has been done automatically in old releases but it slowed down the handling too much. So whenever you feel like you are missing some piece of docu¼check the documentation by using the menu action.
- **Drag&Drop "See Also".**
Just drag a certain method into the documentation view and its selector will be added to the current text. This makes it quite simple to create "See Also:" references.
- **Adjusted Docustyle.**
The current release comes with a slightly adjusted docu style which reflects the changes NeXT has made for NeXTSTEP 3.3. The new style is simpler to parse

and looks slightly better. ClassEditor will handle the old style as it did in earlier versions.

- **Nicer Icons.**

Some new artwork has been done and now it looks prettier all around the screen. None of the images has been tuned for 2bit gray displays. Sorry about that, but DisplayPostscript convertes them quite nicely so most icons can still be recognized.

- **Interface "Redesign".**

I changed some parts of the editor window to reflect the new ideas I have for the browser (thanx to Ralph Zazula for his RZBrowserCell).

- **Cheat Window Tab works.**

The Tabs inside the cheat window will now resize the split view for you. It is not a clean solution but it does the job.

- **Stupid Search Pathes.**

The CEClassEditor.m uses some search pathes which can be adjusted only from a direct dwrite (single entries are separated by a tab).

Bugs Fixed in Release 0.4

Most of the fixed bugs belong into the category^{1/4}nasty little beast.

- **Wrong Savepath.**

ClassEditor did not really remember where the single files came from. This caused files to endup in a different place then it was loaded from.

- **Crashing without finding a Header.**

We crashed if we failed to find a header file Ð oops.
This was due to the following bug^{1/4}

- **Crashing an empty Header.**
The app crashed if there were no methods defined inside the header. After all the fix is stupid. I really should parse the methods inside the implementation too.
- **Fails to open at Lines.**
The app now can open files at a certain line. So e.g. clicking on the ProjectBuilders error lines will bring up the class and marks the requested line. The selection will only be done inside the cheat window!
- **Finding Methods inside the Docu.**
Having methods like **init** and **initData** could have caused trouble if they appeared in the wrong order. It caused the parser to corrupt parts of the wrong method. That is fixed now.

Known Problems

Some of the known problems won't hurt anybody at this moment ± I hope. But I know that they are in there. If you find a way how to reproduce some of the serious bugs or run into problems which are not mentioned here, please tell me.

- **Crashing during Drag.**
In some strange situations starting a drag operation will crash the application due to an wrong "source" object ref inside the Pasteboard. I still have no idea why it mostly works^{1/4}but sometimes crashes ??
- **Adding Methods.**
The ideas for adding methods are already build in but do not work. The only way to

add methods is as you would do it in Edit. Open the cheat window and add a definition to the header plus an empty method body to the source. Be careful that both definitions are exactly the same.

- **No reverting.**

You can't revert to the saved version without closing and reopening a class. ClassEditor will not recognize that a certain file has been changed by another application. Saved last¹/₄ you'll win.

- **No Category loading.**

ClassEditor does not load category source files automatically. This is left to the user. Either merge all the sources (you *can* have multiple categories in there ~~Ð~~ but it is *not* a good habit IMHO) or separate the headerfiles on a per category basis.

- **Missing methods.**

I definitely should parse the implementation file in the first order. Otherwise we will miss all the methods that implement a protocol etc. pp.

The current decision was made because RTF source might be nasty to parse correctly. But with colorizing as we have it in 0.4 I guess RTF source is less important than seeing all categories and methods. I really should look at this.

- **Working with Protocols.**

Based on the previous problem you have to do the following to be able to work with protocols. Add a copy of the protocols methods to the class definition. To not run into trouble with the compiler the whole block has to be commented out by using a pair of `'/*1/4*/'` enclosures.

- **No multiline Methods.**

Multiline method name definitions are not supported. So please remember that. However, inside the documentation you *can* reformat a method to be multiline, but the selector still has to remain inside a single line !

- **Missing Source.**
If your source and header definitions for a method are not the same¹/₄ClassEditor will not be able to find the source. You have to edit the source from inside the cheat window to match the definiton. But no saving is required.
- **Ugly Docu creation.**
The docus our auto-create mechanism spits out are not perfect. They *do* conform to NeXTs style but they do *not*:
 - add method docus in a sorted order after the *first* creation.
 - add instance variables.
 - add methods to the overview
 - add any other dependance info
- **No RTFD Docus.**
At the moment we can't deal with RTFD documentation. Support is build in partially¹/₄but not enabled yet.
- **Corrupted Method Docu.**
Sometimes (in very rare cases) the ClassEditor will replace parts of an existing documentation with a method template. This only happens when performing a **checkDocumentation:** action.
- **Won't remove Method Docus.**
The **checkDocumentation:** method does not remove method documentations which don't have a matching definition inside the headers. This is not a bug but a feature. Otherwise you would always lose delegate method descriptions and such. But we could add a panel that would allow the user to specify the methods that should be removed.
- **Windows out of sync.**
Sometimes you might think that the windows are out of sync. Mostly they are

not¹/₄they only look like they would (somehow they haven't been redrawn).

- **Confusing Selections.**

When making multiple selections inside the cheat window it can happen that old selections won't be cleared correctly. The view failed to redraw and seems to have multiple selections inside Ð wired.

- **Jumping Selections.**

Heavy switching between the windows might cause that you won't end up at the same place where you left a certain window. Please check twice if your cursor is really at the position where you expect it to be.

- **Undo not possible.**

Sometimes the app won't be able to undo your changes. Remember that Undo works only for the currently edited method inside the real class window (not the cheat window).

- **No Find Panel.**

Yes. We still have no Find panel. I recommend using the ProjectBuilders Finder. This is not the best way¹/₄but it helps me for the while.

- **Preferences won't save.**

No code for remembering the prefs has been written yet. In general Pref don't work.

- **Font styles are fixed.**

You can change the color of certain styles¹/₄but you won't be able to change the font. The fonts are hardcoded right inside the NIB.

For the source colorization you can change the defaults by hand. You have to change some undocumented dwrite values. But be careful. Wrong settings might crash you app! (actually they will crash the AppKit's Text object)

- **GUI Fakes.**

Not everything you see does work. Aggressive use of the split views might corrupt the layout..etc..pp.

- **Loding a single Class twice.**

When working with the ProjectBuilder it might happen that you will end up with two open versions of a single class.

PB does not take the path of the file (e.g PB.project) you opened but it always makes the "real" path out of it. So if there is some symbolic link along the way it will remap the path to a real path (starting at the right physical mount point etc.) So if you open files from the Workspace or PB you might have two different pathes. Edit somehow takes care of these circumstances too. ClassEditor does not.

- **Closing Windows.**

When quitting the app windows are not closed in the same order as they appear on the screen. This might cause some confusing window swaps^{1/4}but it works. With the current design the app can not simply close the topmost windows^{1/4}one by one. Future releases should fix that due to a different internal design.

- **Code ugliness.**

Now this is really a project that is just intended to work in the first place. Maybe someday I might decide to take the time and make it a nice project Ð code wise.

Development

Still a lot of work left^{1/4}but I have done something after all.

Jun. 95: Mainly added drag&drop, code formating and a nicer outfit. Also worked on fixing bugs. (5 days of coding and painting)

Version 0.3

The first solid release escaped on 15.02.1995. Now it should really work as advertised. It still has many shortcomings but as long as you stick to the rules it won't corrupt your files^{1/4}as far as I can tell. All the classes of this release have been edited using the ClassEditor. Seems like that worked correctly.

New Features

Even more features out here. But many more are missing.

- **Nicer Icons.**
Some new artwork has been done and now it looks prettier all around the screen.
- **Auto parsing.**
You don't have to save the files anymore to see methods that have been added via the 'cheat window'. This is not really a new feature^{1/4}it's more a fixed bug :-)
Switching from the cheat window to the normal window will cause method reparsing if any changes have been done.
- **FAT.**

We are quad- fad now (NeXT, Intel, HP and Sparc).

Bugs Fixed in Release 0.3

The last release should have been stable, but it was not. So I only focused on fixing the stupid mistakes that were still inside.

- **Pasteboard corruption.**

ClassEditor should never corrupt the pasteboard anymore. We use our own for internal work now.

- **Crashing with Miniwindows.**

Garbage collection should work properly now and you won't crash with a open miniwindow under any circumstances.

- **Parsing the Docus.**

Fixed the parsing bug that caused ClassEditor to corrupt the documentation files when the 'See also' references where used as they should be.

Development

Still a lot of work left^{1/4}but I have done something after all.

Feb. 95: Wrote some categories to the Text object to speed up the parsing nightmare. Some MiscKit classes had to go that way too. (4 days of coding)

Version 0.2

The second public release was on 25.01.1995. I tried to make a useful and save app out of the ugly first release. It doesn't corrupt my sources anymore.

New Features

What's new this time?

- **Emacs Keybindings.**
With the help of the MiscShell palette I added the Emacs text object (by Julie Zelenski). Thanks to the IB that comes with EOF I was able to fix the palettes resize bug.
- **Auto Fileformat detection.**
ClassEditor now does handle RTF or ASCII versions of the source files.
- **Proper exit and Window closing.**
Closing the browser window frees the class document. You are get an alert if the contents hasn't been save yet. The app deals with quitting as it should according to NeXTs guide lines.
- **More Styles.**
Added a "Bug Notes" and math symbol font style.

- **Documentation creation.**
If there was no documentation available we will now create one from scratch.
- **Docu-Template.**
To speed up the process of docu writing you can now access a docu-template that includes the most important text styles so you can cut/copy the layout (rulers and fonts).
- **Preferences.**
Font settings work¹/₄but you can't save them. Sorry.
- **Mutlimethod Documentation.**
You can have one documentation for many methods. Like: **setValue:**,
setValue:after:.
All you have to do is to stick to the layout templates. Take a look at the `ASCIExampleClass`. You will find `multimethod` and `multiline methodname` documentation in there.
- **Undo.**
You can Undo all the changes made inside the source and documentation. But you can only Undo *both* at once.
- **App.info.**
The app comes with a useful *ClassEditor.info* file (similar to .info files of the `Installer.app`). My *NewAppInspector* bundle for the `Workspace` will like this little extra description. Once I'll have the time to polish and finish it I will release this project too.

Bugs Fixed in Release 0.2

There were so many stupid bugs inside v0.1 that I won't list all the small fixes. These are the most serious bugs fixed.

- **Saving.**

Changes should now be saved correctly. v0.1 lost changes to the visible method when edited in both windows.

- **Memory Leaks.**

Now we really free our main editors. You need to switch between applications to trigger the private garbage collection system. But when developing this should happen quite often anyway.

- **Pasteboard Bug.**

The app uses a private pasteboard hack now so in most case you should notice an internal pasteboard work.

- **Syncing.**

Both windows should be in sync most all of the time. If they are not^{1/4}they should be after switching the windows.

Development

Ok. Here is what I did to kick out this version.

Jan. 95: Spent most of the time fixing ugly code sections and making the programm useful. (3 days of coding)

Version 0.1

The first public release of 17.01.95. Therefore everything is new. Nothing will work as expected. This is a fast hack to trigger of some discussion and get some feedback in the early stage.

New Features

Here is a short summary of all the parts that are working now. See the introduction chapter for a list of ideas on future development efforts.

- **Browsing by Methods.**
The app allows you to manipulate the source code on a per-method basis.
- **Loading Files**
Open multiple sets of *.m*, *.h* and *.rtf* files.
- **View the Documentation.**
You can view and edit the a method description side by side with its implementation.
- **Change Fonts.**
Use the "Style" menu for quick access to the fonts you need for a nice docu sheet. The most important fonts are accessible through cmd-key shortcuts for rapid typing.

Development

Here I'll try to show you on which parts I did focus my work in this release.

Jan. 95: Mainly hacked up the NIB file and spent some time learning about the powerful search methods of the Text object :-). This one could really stand many cool categories or subclasses. (2 days of coding)