iTEXMac, an Integrated TEX Environment for Mac OS X

Jérôme Laurens
Université de Bourgogne
Laboratoire Analyse Appliquée et Optimisation
9, Avenue Alain Savary
BP 47 870
21 078 Dijon CEDEX
jerome.laurens@u-bourgogne.fr
http://www.u-bourgogne.fr/Monge/jlaurens

Abstract

At the end of year 2000, Apple released the first public beta of its new generation operating system named Mac OS X, based on former NeXT technologies. More than just a new release, it opened the world of the Unix tools to the Mac community. Very early, a port of the standard Unix teTEX distribution to the Mac OS X platform was available, made by Gerben Wierda, and the first open source Mac OS X front-end dedicated to TEX, named TEXShop, by Richard Koch. Last year, the first public release of iTEXMac came out when it became clear that TEXShop's developer policy was to keep it basic software. The purpose of this article is to present iTEXMac, an integrated TEX environment, as part of a wider open source project which aims to make all our powerful TEX tools easier to use, with as Mac-like a look and feel as possible.

Résumé

À la fin de l'année 2000, Apple a rendu publique la première version de son système d'exploitation de nouvelle génération appelé Mac OS X, basé sur des technologies NeXT. Plus qu'une simple nouvelle version, cela ouvrit les portes du monde des utilitaires Unix à la communauté Mac. Très tôt furent disponible une version de la distribution Unix standard teTEX adaptée à la plateforme Mac OS X par Gerben Wierda, et la première façade dédiée à TEX sur Mac OS X appelée TEXShop distribuée sous licence open source par Richard Koch. L'année dernière, la première version public de iTEXMac est sortie quand il est clairement apparu que TEXShop allait rester un logiciel d'utilisation basique. Le but de cet article est de présenter iTEXMac, un Environnement TEX Intégré, comme partie d'un projet plus général distribué en open source, dont le but est de rendre plus accessibles tous ces puissants outils TEX en gardant autant que faire ce peu la patte Mac.

Presentation

iTEXMac is an integrated suite of software including a text editor, a PDF viewer, a project manager and a front-end to the tools and engines available in a teTEX-like distribution. Each part of the suite will be discussed in more detail in separate sections of the article; here we point out what makes iTEXMac a unique product.

Regardless of the platform, different kinds of TEX-oriented software are available to compose a document. One can choose a completely WYSIWYG option using TEXmacs, a WYSIWYM option using LyX, or the original way of life with just a code editor, a set of macros and a viewer. It seems that no solution can be fully satisfying: on one hand a really nice user interface is constrained by a proprietary file format, on the other hand using and configuring an emacs-like text editor presents a challenge.

Among all the features one can expect from a good product, conflicts exist between look and feel, power and efficiency. Such problems are partially resolved by using mutability, to adapt the software to the end user. Examples of flexible software are emacs and alpha, each one using its underlying scripting language — Lisp and Tcl, respectively. With them, the user can add new features targeting specific tasks like TEX source editing, but has only limited control over displaying. Moreover, these programs serve the much more general purpose of code editing, so that they are not very easy to use. From a TEX point of view, they are extremely powerful, but some features are useless, and others are lacking. So, the long term aim of iTEXMac is to elaborate an all-in-one modular environment to edit, typeset, preview and print TEX-related files. This software achitecture will also be interesting if the different components are sufficiently tied together to result in a better user experience.

All this is fully implemented in iTEXMac using

Objective-C as an underlying scripting language and Mac OS X Application Kit, because Apple's implementation of this object oriented C variant is particularly suitable. Since this application programming interface named Cocoa evolved from NeXTSTEP and OPENSTEP, iTEXMac might be partly ported to other systems using the free software GNUstep.

iTEXMac is provided with English, French and German localizations, the latter by Keith J. Schultz.

Installing iTEXMac

teTEX-like distribution. As a graphical front-end, iTEXMac needs a teTEX-like distribution to be fully functional. For Mac OS X, three distributions are available. Gerben Wierda maintains both an enhanced port of teTEX, with additional packages and programs from TEX Live:

`http://www.rna.nl/tex.html`

Another teTEX 2.0.2 port is available from fink, through one of its graphical user interfaces, finkcommander.

`http://finkcommander.sourceforge.net/`

Once installed, the standard tools are available from the command line interface of Mac OS X Terminal application, but there is no need to fix any environment settings, which hides the Unix layer in Mac OS X.

With the new operating system, Apple replaced its old graphic format named PICT with PDF, such that iTEXMac is naturally oriented toward pdfTEX instead of TEX. However, DVI and PostScript file formats are supported, provided a Ghostscript package is installed for the latter. One is available from the previous url's.

For those who are not familiar with the Unix side of TEX (and Mac OS X), very important information about Gerben Wierda's teTEX distribution is available in the local folder `/Library/teTeX/`:

- `README.texmf.macosx`
- `README.texmf.howtexfindsfiles`

iTEXMac package and goodies. While iTEXMac may be available on CD, the latest version should be downloaded from the official site hosted by the open source software development website sourceforge at:

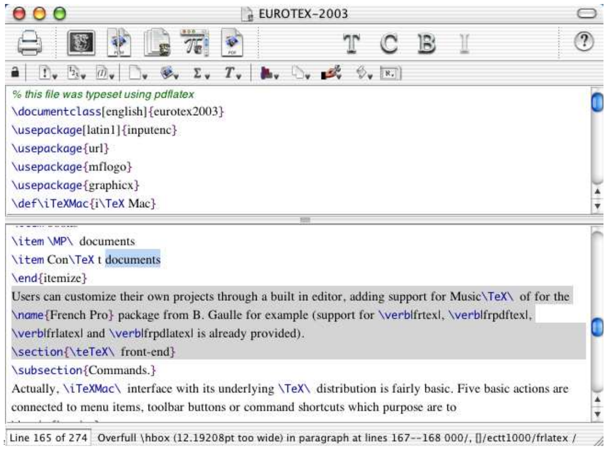
`http://itexmac.sourceforge.net/`

One can find in the download section the disk images for the following products:

- iTEXMac, both stable and developer release
- an external editor for iTEXMac key binding
- *Hypertext Help With LATEX*, wrapped as a searchable Mac OS X help file.
- the TEX Catalog Online, wrapped as a searchable Mac OS X help file.

An updater allows checking easily for new versions. To install iTEXMac, just download the disk image archive, double click then follow the instruction of the readme file.

The built in text editor



```
% this file was typeset using pdflatex
\documentclass[english]{eurotex2003}
\usepackage[latin1]{inputenc}
\usepackage{url}
\usepackage{mflogo}
\usepackage{graphicx}
\def\iTeXMac{i\TeX Mac}
```

```
\item \MP\ documents
\item Con\TeX t documents
\end{itemize}
Users can customize their own projects through a built in editor, adding support for Music\TeX\ of for the
\name{French Pro} package from B. Gaulle for example (support for \verb|frtex|, \verb|frpdftex|,
\verb|frlatex| and \verb|frpdlatex| is already provided).
\section{\teTeX\ front-end}
\subsection{Commands.}
Actually, \iTeXMac\ interface with its underlying \TeX\ distribution is fairly basic. Five basic actions are
connected to menu items, toolbar buttons or command shortcuts which purpose are to
```

```
Line 165 of 274   Overfull \hbox (12.19208pt too wide) in paragraph at lines 167--168 000/, []/ectt1000/frlatex /
```

Editor window with two views of the same text, a main toolbar to switch to the terminal, the output or log documents and launch the typesetting commands, a small toolbar of menus to insert text templates, navigate through the source and launch third party extensions.

The text editor built into iTEXMac heavily relies on the word processing facilities of the Mac OS X Application Kit. However, editing has been enhanced for TEX coding by various means like enhanced text selection schemes, automatic LATEX environment closing and other tools detailed in following subsections. Moreover, iTEXMac allows the use of an external text editor if the built in one does not suit.

Editing. Like many code editors, iTEXMac actually uses text attributes to highlight syntax, changing possibly both the color and shape of the fonts used. Text selection knows about TEX groups, commands and LATEX environments. Balancing parentheses, brackets and braces are highlighted according to user preference.

Character encoding. As an open platform, Mac OS X supports a huge set of character encodings (more than 80, including Unicode) and iTEXMac implements the user interface to manage them. This allows editing files in different languages or from different platforms.

The main problem concerning character encoding is transparency: except for Unicode, there is no rule to automatically recognize the character encoding of a text file. So, iTEXMac has a customizable default encoding used to read and write text files when no external information is given. Moreover, iTEXMac parses the text files for hints about the real character encoding, and when it finds in the file contents an explicit declaration of a character encoding it reads the file again with that encoding. For this purpose, iTeXMac recognizes the following kinds of hints:

1. an inline instruction at the very beginning of a source file:

   `%!iTeXMac(charset):` *character encoding*

   As a TeX comment, it is ignored by the various typesetting engines. The *character encoding* is simply the name of the character encoding with a standard syntax (e.g. ISO Latin 1, code page 1251, Mac OS Roman...)
2. the use of the LaTeX package `inputenc.sty`, parsing the optional argument:

   `\usepackage[`*encoding*`]{inputenc}`
3. the use of the ConTeXt `\enableregime` command, parsing the optional argument:

   `\enableregime[`*encoding*`]`
4. emacs scheme on the first input line:

   `%-*-coding:` *character encoding*`;-*-`
5. Mac OS X hidden internals.

With this management one is able to split a source document into many different files, each using its proper character encoding. This is very similar to the way emacs manages encoding with its `-*-coding:` *name* `-*-` option, with the addition of `inputenc.sty` and `\enableregime` related hints. In a forthcoming version, iTeXMac is planned to also support one more hint: meta data stored in an external project file.

Spell checking is known to be a hard problem when applied to TeX input due to the complex definition of words and word delimiters. For Mac OS X, dedicated software has been ported from previous products or newly developed, namely Rick Zaccone's Excalibur at

`http://www.eg.bucknell.edu/~excalibr/`

and Anton Leuski's cocoAspell, a port of the free and open source spell checker GNU aspell, at

`http://www-ciir.cs.umass.edu/~leouski/cocoaspell`

While both are TeX-aware, the former is based on older Mac word services and will be used as a separate application. The latter also knows about HTML code and is automatically integrated to iTeXMac and virtually any other application when properly installed. Moreover, it is possible with cocoAspell to automatically check spelling as you type, misspelled words being underlined in red. But when used with another text editing application (TextEdit, TeXShop, Mail), this continuous spelling check feature will definitely mark all TeX commands as misspelled, which is unwanted. In order to fix this very annoying problem, iTeXMac code patches some deep Mac OS X Application Kit internals: this is the only TeX aware text editor with working automatic spell checking.

Although spell checkers are generally designed with a default language, we can imagine that each text file could contain informations about the dictionary and language to be used for spell checking. This is planned for a forthcoming version, in an implementation very similar to the one for encoding. The ultimate feature should be to switch the spelling language inside one input file according to the context, but this needs a huge amount of coding ... Splitting the sources according to their natural languages and encodings should be more than sufficient.

Macros menus. The text editor window gives access to five macros menus to help inserting TeX commands for the document, text, math, graphics and everything else. Those menus are configurable and their contents can change on a per document basis. All the standard LaTeX commands are available through these menus, whereas ConTeXt and METAPOST support is not complete (due to a lack of knowledge of the author). However, users can customize the interface according to their needs with the aid of a dedicated graphical menu editor.

Navigation menus Three other menus facilitate navigation through source code. The text is parsed to recognize special tags. One menu is dedicated to LaTeX or ConTeXt sectioning commands and also works for METAPOST figure keywords, another menu is dedicated to labels and references, allowing to easily insert `\ref`'s and `\eqref`'s. Finally, each line beginning with

`%!iTeXMac(mark):` *comment*

gives an entry titled *comment* in a third menu, where one is free to mark any part of the text.

Key binding. To speed up text entry, iTeXMac allows customizing the system key mapping. It is possible to map essentially any key combination to macro actions, including scripting. There is an external editor to manage those mappings in a flexible and friendly way.

Some Apple documentation leads people to believe that any text editor (e.g. TextEdit) written with Mac OS X Application Kit naturally gets the benefit of emacs-like key bindings. This is only the case for a very limited set of shortcuts. iTeXMac expands this feature significantly.

Toward WYSIWYG. The built in text editor does not provide any WYSIWYG support while editing, and is not meant *a priori* to ever become completely WYSIWYG. However, it can show a character palette with 42 menus gathering text and mathematical symbols, as they would appear in the output. It is a graphical front-end to the standard LaTeX packages

- `amsfonts.sty`

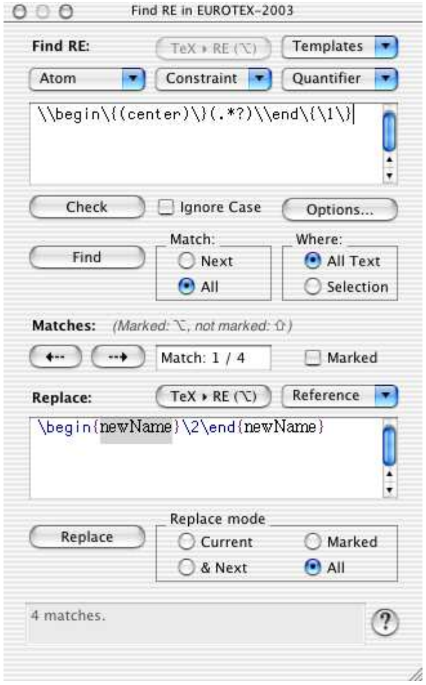
- amssymb.sty
- mathbb.sty
- mathrsfs.sty
- marvosym.sty
- wasysym.sty

which makes thousands of symbols available in one click. Due to a lack of knowledge, there is no counterpart for the ConTEXt world.

Some kind of assistant should also provide the user with a list of standard equation and table templates with both output and input forms. This kind of WYSIWYG support would be a shortcut to the large number of different mathematical environments, very useful for beginners. It is planned for a further release, unless it becomes available as an external plug-in.

Regular expressions. The built in text editor implements Unicode aware regular expressions. A dedicated find/replace panel has an elaborate graphical user interface to help the user construct, check and use regular expressions. A limited set of built in regular expressions is provided (e.g. to change LATEX environment names), and the user can save frequently used regular expressions.

Despite the regular expressions used being advanced ones, they are not Perl compatible. Once a library of Perl compatible and Unicode-aware regular expressions becomes available, code will be added to support them.


Find RE in EUROTEX-2003

Find RE:

```
\\begin\{(center)\}(.*?)\\end\{\1\}
```

☐ Ignore Case

Match: ○ Next ● All

Where: ● All Text ○ Selection

Matches: (Marked: ⌥, not marked: ⇧)

Match: 1 / 4

☐ Marked

Replace:

```
\begin{newName}\2\end{newName}
```

Replace mode: ○ Current ○ & Next ○ Marked ● All

4 matches.

Advanced regular expressions panel with find and replace fields, and various controls to help in elaborating, checking and using the regular expression.

Extensions. An AppleScript implementation and a plug-in architecture are the two means provided to let the user expand iTEXMac's built in text editor.

AppleScript, which is Apple's designated scripting language, is fully supported. The basic features provided by Mac OS X application kit are extended to fully support undo management and string replacement. It is then possible to create high level editing scripts to expand ad infinitum iTEXMac functionality. However, those scripts are of limited use in practice when they deal with very heavy text editing tasks, due to performance reasons.

Plug-in architecture. While using the Mac OS X Objective-C application kit framework, iTEXMac provides a very efficient plug-in architecture. This was originally designed to expand the editor possibilities by using assistants. In fact, an array assistant is available which helps in inserting array code for LATEX. This is a very basic assistant written for demonstration purposes only. An open source development kit is available but there are no developers yet.

Here are some ideas for assistants that would be of great help to new users or for experimentation:

- a layout helper via a geometry.sty front-end
- a layout helper for list environments
- a WYSIWYG helper for fonts
- a helper for customizing the sectioning commands
- ...

The project manager

The notion of project comes naturally when working on a ConTEXt or LATEX document because many files are created as compilation side effects. So, project management is a task assumed by one of the core components of iTEXMac. Each document created should have an associated project file (with extension pTeXMac) where meta data should be stored. Due to historical reasons, the project implementation has been partially hidden, so as to mimic the user interface of former Mac OS TEX editors. This leads to elaborating a small list of default projects that would fit the widest range of situations, including for example

- LATEX documents
- LATEX documents with PDF engines
- books
- METAPOST documents
- ConTEXt documents

Users can customize their own projects through a built in editor, adding support for MusicTEX or for the French Pro package from B. Gaulle for example (frtex, frlatex, frpdftex and frpdflatex support is already provided).

teTeX front-end

Commands. Actually, the iTeXMac interface with its underlying TeX distribution is fairly simple. Five basic actions are connected to menu items, toolbar buttons or command shortcuts which do the following:

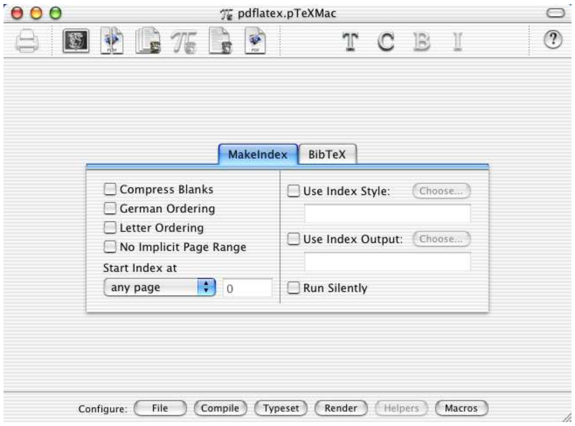
- compile (e.g. running `pdflatex`)
- typeset (e.g. running `pdflatex` twice)
- make the bibliography (e.g. running `bibtex`)
- make the index (e.g. running `makeindex`)
- render graphics (e.g. running `dvipdf`)

The compile action should be used while writing a document, the typeset action should concern the final stage and may include an index or bibliography build phase. By default, all these actions are themselves connected to shell scripts stored on a per project basis. If necessary, the end user can edit project files and customize the shell script launched by iTeXMac. He also can alter the whole action performed in one of two ways:

- customizing the project document
- inserting an in-line instruction at the very beginning of a source file:

  `%!iTeXMac(compile):`*one line shell script*

As shown below, a graphical user interface is available that wraps the command line options of `makeindex`. One is also available for `bibtex`, and others are planned for the standard tools. Apple Events allow third party applications like Alpha X to pilot iTeXMac.

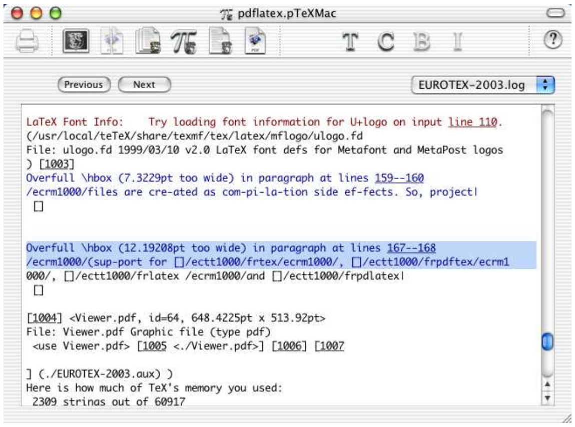


□ Compress Blanks
□ German Ordering
□ Letter Ordering
□ No Implicit Page Range
Start Index at: any page 0
□ Use Index Style:
□ Use Index Output:
□ Run Silently

Project panel wrapping the `makeindex` *command line options.*

Log files are a valuable source of information when a problem arises, not only for beginners but for TeX gurus too. iTeXMac displays and parses both the output of the typesetting action and the corresponding log file. Warnings and errors are displayed with different colors than the rest of the text to point out relevant information, and the corresponding line numbers are displayed as HTML links. They are linked to the line of the input file where the error or warning occurred: clicking on that link will open the file in the text editor, displaying a light gray background at the given line. See below.

This procedure does not work in a very limited number of situations, especially when the TeX source extract explaining the error interacts with the "normal" log contents. At present, the `--file-line-error-style` option of the typesetting engines is not supported, but it is planned for a future version.



```
LaTeX Font Info:    Try loading font information for U+logo on input line 110.
(/usr/local/teTeX/share/texmf/tex/latex/mflogo/ulogo.fd
File: ulogo.fd 1999/03/10 v2.0 LaTeX font defs for Metafont and MetaPost logos
) [1003]
Overfull \hbox (7.3229pt too wide) in paragraph at lines 159--160
/ecrm1000/files are cre-ated as com-pi-la-tion side ef-fects. So, project|
 □


Overfull \hbox (12.19208pt too wide) in paragraph at lines 167--168
/ecrm1000/(sup-port for □/ectt1000/frtex/ecrm1000/, □/ectt1000/frpdftex/ecrm1
000/, □/ectt1000/frlatex /ecrm1000/and □/ectt1000/frpdlatex|
 □

[1004] <Viewer.pdf, id=64, 648.4225pt x 513.92pt>
File: Viewer.pdf Graphic file (type pdf)
 <use Viewer.pdf> [1005 <./Viewer.pdf>] [1006] [1007

] (./EUROTEX-2003.aux) )
Here is how much of TeX's memory you used:
 2309 strinas out of 60917
```

Log file window outlining errors, warnings or page numbers and displaying navigation links.

A navigation facility from log file to output is also provided. When iTeXMac parses a string like `[`*a number...*`]`, *a number* is displayed as an HTML link. When clicked, the window of the output file is brought to the front and switched to the right page. Like the previous feature, it is not completely reliable because of the lack of log syntax rules, however, it works fairly well.

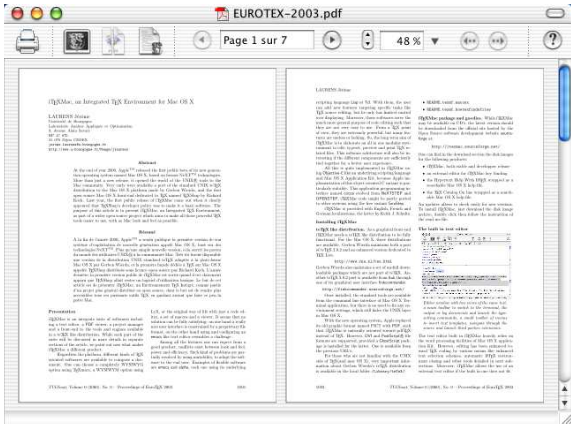
Parsing the log file in the background allows for informing the user when some problem occurs while typesetting, with an audio alert and an informational message at the bottom of the text editing window. Despite this not being absolutely reliable (see above), it can be of great help.

This feature is resource consuming and can be turned off in the preferences.

Documentation and help, provided with teTeX, is available though a menu. Moreover, Graham Williams' TeX Catalog Online and Sheldon Green's HyperText Help With LaTeX are available as Mac OS X searchable help books accessible from the built in Mac OS X Help Viewer. The former lists about 1100 TeX, LaTeX, and related packages and tools.

As a side effect, iTeXMac can open PS, EPS and DVI files with a double click, by converting them to PDF. In a way, it therefore plays the role of a PostScript viewer.

The built in PDF Viewer



Viewer window in the facing pages mode, with a main toolbar to switch to the terminal, the source or log documents, navigate through the pages and manage magnification.

A basic PDF viewer is provided in iTeXMac. Once again, this is a wrapper of built in Mac OS X technologies that implements the basic features of a PDF viewer. Unfortunately, Mac OS X support for this file format is sometimes limited with the output of the `pdftex` command. In such a problematic situation where the PDF can not be properly rendered by iTeXMac, an alternate solution is offered by using an external PDF viewer. This alternate solution can be permanent or temporary.

Using an external PDF viewer is also welcome because Mac OS X technologies do not support any link related or indexing features, thus ignoring the benefits of the `hyperref.sty` package and ConTeXt facilities, for example. Before these are implemented in iTeXMac's own code, or made public by Apple, a standalone previewer can be used.

Navigation between input and output is one of the great features missing in iTeXMac. This "Synchronicity" feature was first available in *Textures*, a Mac product from Blue Sky Research. Then a less efficient implementation was provided for LaTeX users through the use of `srcltx.sty` which added `src:` specials in the DVI files. This task is now assumed by typesetting engines through the command line option `-src-specials`.

But all this is useless since the preferred output file format used by iTeXMac is PDF and all translators from DVI to PDF filter out this information. In fact, there is a real need for `pdftex` support of the `-src-specials` option, not only from the Macintosh community but also from `gsview` and `xpdf` users. It remains evident that the implementation of such a feature must be revisited, because PDF contents is not a stream. Nothing has been done yet in this direction.

However, a poor man "Synchronicity" feature has been implemented to help in switching from source to output. It is based on a tiny LaTeX package written by Pietro D'Ancona and provided with the iTeXMac distribution, which essentially contains

```
\newtoks\@tempoutput
\global\@tempoutput\expandafter
    {\the\output\immediate\write15
    {[\thepage] \the\inputlineno}}
\global\output
    {\expandafter\the\@tempoutput}
\AtBeginDocument
    {\immediate\openout15=%
        \jobname.pageLineIdx
    \immediate\write15{[1]
        \the\inputlineno}}
\AtEndDocument
    {\immediate\write15{[1]
        \the\inputlineno}%
    \immediate\closeout15}
```

This LaTeX code must be inserted in each master LaTeX document. It opens an auxiliary file in which are stored some hints about the correspondence between input file line numbers and output file page numbers. The PDF viewer parses this auxiliary file and uses the hints to display the proper part of the document, either input or output.

Future directions

In addition to the forthcoming improvements previously mentioned, here we detail other features to come. Some of them just reflect the natural development policy of iTeXMac, others come from user requests. This should convince interested people to contribute to the iTeXMac project.

Bibliography management. At present, iTeXMac is missing a built in support for such a feature. The bibliography database management is now provided by Michael McCracken's BibDesk (beta version 0.8), another sourceforge project at

`http://bibdesk.sourceforge.net/`

This application also implements a Mac OS X service which allows any application to automatically insert or complete bibliography references in the currently edited text. There's no way iTeXMac will offer in the near future these features as built in, however, a better integration with a BibDesk-like data base application is planned.

This solution is useful for big bibliography data bases but it does not manage small ones declared explicitly in `thebibliography` LaTeX environments. iTeXMac should provide a feature devoted to this task. In order to improve efficiency, this inline bibliography should be declared in a standalone file called by the master TeX document.

TeX macros are of frequent use as soon as one becomes familiar with TeX. While standard macros are managed by iTeXMac, nothing is done for user defined TeX macros. While parsing the whole source files for `def`'s and other `newcommand`'s would certainly be rather difficult and time-consuming, users should gather their macro declarations in one file called by the master TeX document, just like the inline bibliography above.

Completion is a standard feature of code editors but is not built in to iTeXMac. However, one can install TextExtras from

`http://www.lorax.com/FreeStuff/TextExtras.html`

It is a tiny framework written by Mike Ferris, whose purpose is to give all Cocoa applications useful features, including code completion. It is done by using a dictionary shared by all applications, declaring shortcuts to be completed, and expected output. It also scans the preceding text for words with the same prefix but ignores the backslash introducing TeX macros.

The source code is available; it should be borrowed and extended for iTeXMac's purposes, mainly by considering TeX specifics.

Equation Service and LaTeX Equation Editor The former is a very nice system service to typeset only parts of a document. Its author Doug Rowland makes it available at

`http://www.esm.psu.edu/mac-tex/EquationService`

When some text is highlighted in any Cocoa application, Equation Service can typeset it as LaTeX commands and either put the result on the clipboard or replace the original text by the resulting PDF or TIFF output.

LaTeX Equation Editor was written by J. McKenzie Alexander to add an editor layer to the previous service. This basic editor is downloadable from

`http://evolve.lse.ac.uk/software/EquationEditor`

While iTeXMac's built-in text editor is not graphics oriented, it does not support such a system service, and so should implement a similar feature (from the client side). Moreover, these three packages provide approximately the same functionality, and iTeXMac will implement its own equation service and equation editor functionality.

Technical enhancement. The next major release will benefit from better internal code design. The core components of iTeXMac — built-in editor, PDF viewer, teTeX front end — will become code libraries (frameworks in Mac OS X terminology), such that virtually all the features will be accessible for third party coding. It will be useful for iTeXMac plug-ins but other software will be able to use them too, including the application described above.

Conclusion

After more than one year, iTeXMac has proven to be an interesting product, and the future promises to be even more interesting. (Many significant improvements have been made since this writing.)

Many people have contributed to the project, their credits are found with their contribution in the iTeXMac distribution.

iTeXMac logo: