# Baskerville

**The Annals of the UK TEX Users' Group** **Editor: Editor: Sebastian Rahtz** **Vol. 3 No. 2**

**ISSN 1354–5930** **February 1998**

Articles may be submitted via electronic mail to `baskerville@tex.ac.uk`, or on MSDOS-compatible discs, to Sebastian Rahtz, Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, to whom any correspondence concerning *Baskerville* should also be addressed.

This reprint of *Baskerville* is set in Times Roman, with Computer Modern Typewriter for literal text; the source is archived on CTAN in `usergrps/uktug`.

Back issues from the previous 12 months may be ordered from UKTUG for £2 each; earlier issues are archived on CTAN in `usergrps/uktug`.

Please send UKTUG subscriptions, and book or software orders, to Peter Abbott, 1 Eymore Close, Selly Oak, Birmingham B29 4LB. Fax/telephone: 0121 476 2159. Email enquiries about UKTUG to `uktug-enquiries@tex.ac.uk`.

---

## Contents

# I Editorial

## 1 *Baskerville* new start

At the 1993 UKTUG Annual General Meeting (of which the formal report appears later in this issue) it was agreed that *Baskerville* should enter a new phase of its career, appearing six times a year to a fixed schedule. This last issue of 1993 is the first to appear under the new scheme, and I hope that members will start to 'notice' *Baskerville* a little more and start contributing articles. *Please note the following schedule of copy deadlines*:

| Issue | Submit material for publication | Submit last-minute notices | PostScript file sent to production team |
|---|---|---|---|
| 4.1 | Jan 10 | Jan 17 | Jan 31 |
| 4.2 | Mar 21 | Mar 28 | Apr 4 |
| 4.3 | May 23 | May 30 | Jun 6 |
| 4.4 | Aug 15 | Aug 22 | Aug 29 |
| 4.5 | Oct 17 | Oct 24 | Oct 31 |

*Baskerville* regularly publishes articles answering common TEX questions, and these are available as technical notes to UKTUG members. A first list will appear in the next issue. Members are urged to contribute their own short (one side of A4 may be all that is needed) 'topical tips' which we can add to the library to answer queries from those who do not participate in academic electronic network discussions.

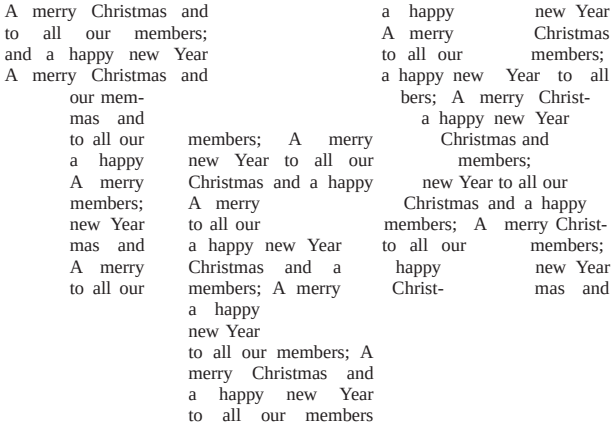
Another innovation in this issue is the first of a regular column by Malcolm Clark — *Malcolm's Gleanings*. Malcolm has an unrivalled knowledge of happenings and publications in the world of computers and typesetting and will share his discoveries with us.

Each issue of *Baskerville* will have a special theme, although articles on any TEX-related subject are always welcome. Contributions on the themes for the first half of 1994 are eagerly solicited:

- ☞ *Baskerville* 4.1 will be a special issue on PostScript (to coincide with the UKTUG meeting on using PostScript fonts);
- ☞ *Baskerville* 4.2 will be a special issue on LATEX2e (which should be widely available by then);
- ☞ *Baskerville* 4.3 will be a 'back to basics' special issue on mathematical and tabular typesetting.

This issue of *Baskerville* is devoted mainly to the issues of interfaces and shells for TEX.

Finally, given the season, and Donald Arseneau's intriguing shape-making macros published in *Baskerville* 3.1, how can I resist trying another shape he recently created?



This issue of the journal was created entirely with LaTeX and printed on a Hewlett Packard LaserJet IV. It was set in ITC New Baskerville Roman, with Computer Modern Typewriter for literal text. Production and distribution was undertaken in Cambridge by Robin Fairbairns and Jonathan Fine.

## II Report of the 1993 UKTUG AGM

R. A. Bailey
Hon. Secretary 1991–

*Official report of the AGM of the UK TEX Users Group*
*held at Aston University (Room 708, Main Building)*
*on Wednesday 20 October 1993 at 1140 hours*

There were at least twenty-five members present. The following is a brief summary of the business transacted; it is categorized by, roughly, the numbered agenda items.

1. **Report of the 1992 AGM** This report had already been published in *Baskerville*, Volume 3, Number 1. Copies were also available at the meeting. The report was received as correct.
2. **Matters Arising not elsewhere on the agenda** The Secretary reported developments on two agenda items.

   6 Of the committee membership in November 1993, four people had resigned because of pressure of other work: S. F. Brooks, I. McNeil-Sinclair, D. V. Murphy and G. Toal.

   12 The committee had considered the suggestions made from the floor and decided that:
   - the Group does not have the resources to organize professional training but should attempt to provide information about training;
   - members should not be able to elect to receive notices from UKTUG primarily by electronic mail or primarily by ordinary mail, because this would effectively double the workload of the officers concerned;
   - there should be a UKTUG Membership List as soon as the committee has sufficient resources to produce one.
3. **Chairman's Report** The Chairman, P. Abbott, read his report; he apologized for the sick leave which had prevented him playing as active a part as he would have liked, and described the success of the international TEX Users' Group meeting held at Aston in the summer.
4. **Approval of Accounts** The Treasurer's report was given orally. Copies of the unaudited accounts for 1992–93 were presented. When audited, the accounts will appear in *Baskerville*.
   There was some discussion of the loss made on the two-day meeting at Royal Holloway and Bedford New College in April 1993, which had been the Group's first residential meeting. The incoming committee was asked to learn from this experience.
5. **Appointment of Auditor(s)** Colin Smith was reappointed auditor for 1994, and thanked for his work to date. It was agreed to pay him an honorarium of £25.
6. **Membership Fees** The Treasurer proposed the following motion, on behalf of the committee:

   *The membership fee for 1994 shall be £18.00 for full membership or £9.00 for full-time student membership. Anyone who joins the UK TEX Users' Group on or after 1 October 1993 will, for the above fee, become a member until 31 December 1994, although they will be ineligible to vote on business of the UK TEX Users' Group until 1 January 1994. There will be no reduced membership fee for those joining after 1 May 1994.*

   It was passed *nem. con.* However, it was pointed out that the proposed increased frequency of *Baskerville* was a substantial new benefit of membership, and that some costs previously absorbed by members' institutions might have to be paid in future. The incoming committee was asked to bear these points in mind when recommending the 1995 membership fee.
7. **Meetings Fees** The Treasurer proposed the following motion, on behalf of the committee:

   *The customary fees for attending ordinary meetings and workshops shall remain at their 1992–1993 levels, that is:*

   | | |
   |---|---|
   | *for meetings—* | *£20 for members* |
   | | *£30 for non-members* |
   | *for attending workshops—* | *£30 for members* |
   | | *£40 for non-members.* |

It was passed *nem. con.* Members were reminded that the differential is set so that members recoup their membership fees by attending only two meetings.

8. **New Chair** The outgoing Chairman announced that only one nomination for Chairman had been received. Therefore, no ballot would be necessary, and the new Chairman would be C. A. Rowley.
9. **Election of Committee** Of the previous committee, R. Fairbairns and S. P. Q. Rahtz continued. Of those retiring, R. A. Bailey stood for re-election. Further nominations for committee membership had been received for M. Clark, J. Fine, A. Jeffrey and A. Nimmo. These five people were all elected to the committee, bringing the total size of the committee to seven (excluding the Chair).
   Subsequently, the committee co-opted P. Abbott and C. Hewlett.
10. **Newsletter Editor** S. P. Q. Rahtz, the current editor the Group's newsletter, *Baskerville*, reported his intention to publish one issue every two months, with content and frequency taking precedence over absolute quality. The meeting supported this approach.
11. **Topics for Meetings** The membership made the following suggestions for meetings.
    - A one-day basic introductory course twice per year, to be advertised widely, and to make sufficient profit to subsidize conferences.
    - A one-day 'New Developments' briefing twice per year, with the opportunity to bring floppy discs and obtain updates of software.
    - LaTeX3.
    - New Typesetting System.
    - Local evening meetings on the model of the Royal Statistical Society, perhaps with the same speaker at several meetings.
    - Joint meetings with other societies.
    - Meetings to be held in academic breaks, so that more members can attend.
    - Meetings to last at most one day, usually.
12. **Aston Archive** P. Abbott reported that `uk.ac.tex` would cease to exist at the end of 1993, but that the Aston Archive would be maintained for the foreseeable future, with `ftp` access. The meeting thanked P. Abbott for managing the Archive for so many years. The question was raised as to whether the Archive should formally belong to UKTUG.
13. **TeXnical services to members** There was some discussion about whether the UKTUG should, or could, provide standard emTeX and ozTeX distributions; about the links between such distributions and the Archive; about maintaining standard sets and documentation jointly with GUTenberg; about the desirability and possibility of providing software to non-members, perhaps at a higher cost; about answering questions relating to the installation and set-up of any software distributed by the group. The committee was asked to consider these matters.
14. **Cathy Booth Memorial Fund** The meeting approved the establishment of a trust to administer the Cathy Booth Memorial Fund.
15. **Deaths** The deaths were reported of UKTUG member Catherine Griffin and Yuri Melnichuk, a founding member of the Ukrainian TeX users' group visiting the U. K., both in summer 1993.
16. **TUG'93 Hardship Fund** P. Taylor reported that the UKTUG had donated £1000 to this fund, which had over £5000 in total. It had been used to pay some travel and accommodation expenses of 16 participants at TUG'93, mostly from Central and Eastern Europe, who would otherwise have been unable to attend. The UKTUG was thanked for its donation.

## III Front-ends—a backward glance

R. Allan Reese
Hull University

Is TEX too ugly to use? Is software a fashion item? Do users know what they are doing, or do they just accept what 'comes out of the computer'?—just some random thoughts brought on by the talks on front-ends to TEX at the October meeting of UKTUG.

The originator of TEX was (probably still is) happy to write `tex poo`, `preview poo`, `print poo` and so on; you are referred to your *Local Guide* for the exact names of 'preview' and 'print'. But the command style of working is associated with mainframes and old times. How can we make writing TEXed documents more palatable?

Sebastian Rahtz described numerous front-ends to TEX on DOS, OS2, Windows, Unix and other platforms. Some TEXies have put a *lot* of effort into integrating TEX itself with previewers, printer drivers and other software. The trouble is that many of the good TEX tools are original and freeware. Each was written by an individual to express his (her) genius, so they don't conform to a 'corporate style'. On top of that, each installation of TEX has to fit on a system that is idiocyncratically configured. Installing some of the free shells may involve great effort—and that's only to read the documentation. Adapting the shell to your choice of TEXware is even more difficult.

The primary need, said Sebastian (and I agree), is to have a good editor. It's even better if the editor is 'TEX-aware', so it will pick up small syntax errors as you go. It may 'helpfully' put in whole skeletons for logical structures. You could distinguish between a 'text editor' in which you type and see raw TEX code, and a 'TEX-processor' in which you type on a WYSIWYG screen but can also see the TEX code which the program is generating. WYSIWYG is fashionable but I have doubts that *most* typists will use a directly visual system to express the logical structure of their text. I'm with Lamport. Write *logical* text and then apply a style to express the logic on an output device. I believe that CGYWYAFNWYW—Computers Give You What You Ask For Not What You Want.

However you type, the TEX input has to be processed for a chosen output device, including the screen for an accurate preview. Those who can't bear to type the commands can buy fully integrated TEX systems. One of these is a good way to get started. Sebastian mentioned the delights of working in Windows, though he admitted in practice to running OS2. This system allowed him to preview a TEXed document and fax the output image just by clicking on a couple of menus (one being his list of fax numbers).

Christopher Mabb was there to promote *Scientific Word* but unfortunately couldn't demonstrate it live. At a price of several hundred pounds per copy, my interest is likely to stay 'academic' but one professor at my site uses a comparable system *Leo*. Both offer a happy-clicky interface which can be used to write maths as visual patterns rather than as readable formulae. Again, it may be old fashioned but I find it difficult to articulate typeset maths from the page. TEX is now a *lingua franca* for 'reading' or 'speaking' maths in the same way that computer languages precisely express algorithms. It was not clear what a non-mathematician would gain from this interface but WYSIWYG typing may be the approach for introducing *design*, leading on to TEX as a specification tool.

For academics and others on shoe-string budgets, there are a number of editors and front-ends in the public domain. Sebastian has looked at several and thought *TEXshell* was the best integrated. I have also fetched TEXshell but failed to install it to use a Novell fileserver—"*tempus fugit*" and "*temper* said *f*** it*". The editor did look attractive. *Redit* is a similar editor but is not *quite* converted from German to English. Both are based on the Borland Turbo Pascal look and feel with a standard menu at the top of screen and pull down boxes for the next level. So 'standard' is this screen now, that I was able to use Redit in German and deduce many of the translations of menu words in the process. In the week of the conference I fetched *ET* (described in *TUGboat*, Vol 14 No 2. July 1993) which is a text editor with *some* facilities for editing *some* mathematical constructs visually. I think I'll stick with ET for now, though many people believe only in *Emacs*.

ET by the way stands for 'Edit TEX' and is quite different from *TE* (Tiny Editor) that has been in the archives for years. I liked TE until I found it hangs the PC if asked to edit a file larger than 60K. You have to reboot the PC to continue and this is not acceptable on a public service. Unfortunately I can't trace either the source or a support person for TE.

The choice is wide and the jury is out, even though most of them (users) haven't looked at the evidence and don't know what they are trying. You can run TEX by calling each component; you can couple the components through a

shell to save some typing; or you can buy a commercial setup that runs everything behind the scenes. Try one. Try them all and choose what suits you. Enjoy!

# IV A brief guide to TEX assistants

Sebastian Rahtz
ArchaeoInformatica
York

This paper offers a 'back to basics' overview of the various types of software which are of assistance to the TEXnical writer on a personal computer (this includes Unix machines, but excludes, for instance, VMS or VM/CMS systems). It is based on a presentation to the October UKTUG meeting, and I am grateful to the other speakers, and our chairman Allan Reese, for insights and news. I would also like to refer readers to the Dutch TEX Group's journal *MAPS*, whose issue 93.2 contains a variety of useful papers on TEX interfaces (two of which are reproduced elsewhere in this issue of *Baskerville*.)

Every TEX user knows that the traditional command-line way of working (the 'edit ; compile; {preview, print}' cycle) is far from ideal, and many people want some help. We may define eight different ways in which the environment we interact with can be improved:

1. Integrated environments (e.g. TEXtures on Mac), where everything is tightly-coupled, and designed from the ground up to work together;
2. Integrating shells (e.g. TEXshell), where a set of different programs are controlled by a single interface, which does its best to protect the user from their idiosyncrasies and errors;
3. Loosely-coupled packages (e.g. Windows, Desqview setups), where a multi-tasking environment is used to combine together various programs to provide a useful working environment;
4. TEX-aware editors (e.g. Gnuemacs AUCTEX package), which simplify the input of complex markup, and check syntax;
5. TEX super-scripts, including Makefile generators (e.g. 'texit'); these take away the need to remember complicated program switches, and remembering which files are up to date, by understanding the rules about the different types of TEX-related files;
6. Super-previewers (e.g. xtex), which allow us to make greater use of the previewer to inspect and interact with our document.
7. Hypertext source convertors (e.g. latex2html), which make the tedium of writing generic markup with LATEX more worthwhile by giving it a life beyond the printed page;
8. Friendly TEX programming; when we *do* interact with TEX's enigmatic '*' prompt, we want macros to help us, not hinder us. This is discussed by Jonathan Fine elsewhere in this *Baskerville*.

Since most computer users seem to belong in distinct 'camps', based on machines and operating systems, it seems appropriate to examine what is on offer in each area.

## 1 Windows, OS/2, and NT

For a fully-fledged Windows TEX, we have to look to the commercial companies: TurboTEX, from Kinch Computer Co, offers a full TEX, previewer, and drivers with a moderately-well integrated Windows interface, although (critically) it lacks a real integrated editor. The 'traditional' PC supplier of TEX, PCTEX, has also recently released a Windows setup, although it (like TurboTEX) fails to take advantage of builtin fonts properly. Y&Y sell a high-quality Windows previewer and PostScript driver, linked with a public domain TEX and commercial programmer's editor, which work together well in a loosely-coupled way, and offer genuine integration with the Windows font systems (although the previewer has trouble using PK fonts). *Scientific Word* (described elsewhere in this *Baskerville*) takes a different direction by offering a WYSIWYG writers interface (particularly for maths) which saves files in LATEX format and does final formatting for printing with TurboTEX. None of these products, however, offer the same degree of integration and instant previewing as Textures on a Mac, and we still await the 'real' Windows TEX.

The more impoverished writer who wants to build her own Windows TEX setup has a not altogether easy time.[1]

[1] Wietse Dol discusses this more fully elsewhere in this *Baskerville*.


**Figure 1.** Microemacs TEX menus


**Figure 2.** Dviwin accessing Windows printer driver

There is an excellent previewer, Hippocrates Sendoukas' *Dviwin* (now at version 2.81, but *still* lacking support for virtual fonts, and nowhere near using the Windows system fonts), and the excellent emTEX ports of the basic TEX programs run in DOS windows (with some shenanigans necessary to change extended memory managers), but integration is left to the user. The powerful editor Microemacs has an adequate Windows port, extensible to allow TEX jobs (compiling, previewing, printing — see Figure 1) and context-sensitive help to be selected from menus, but it does not provide much protection for the user from the essentially unsatisfactory nature of Windows DOS boxes and multi-tasking. The alternative, running a DOS TEX shell in a DOS box under Windows, probably gives a better result (see Wietse Dol's article). But the convenience of access to Windows printer drivers (such as fax boards — see Figure 2) is a pointer to why this possibility should not be ignored.

Those converting to NT will, of course, have access to all the Windows tools, and the Unix 'web2c' TEX has been ported, but there do not yet appear to be any interfaces which make serious use of NT's abilities.

OS/2 users are well served by the basic emTEX kit, which has a proper Presentation Manager previewer, and by various shells (e.g. *PMTEX* which make life easy. Given the good multi-tasking, Windows compatibility, and the availability of compilers to port most Unix tools (such as *Gnuemacs*, *make* etc), the OS/2 TEXie has the best of almost all worlds.


**Figure 3.** TEXshell, after running TEX and finding an error

## 2 DOS

The hard-pressed DOS user (or lover — there are a few) has a plethora of tools to make up for the inadequacies of the command line. The most mature of the 'shell' programs, which run TEX, previewer, printer, editor etc for you sequentially when you select menu items from a character-based screen, is probably Jürgen Schlegelmilch's TEXshell (see Phons Bloemen's article below). A typical screen is shown in Figure 3. Other offerings include Thomas Esken's TEXsurface, Johannes Martin's TEXpert, Ulrich Jahnz's Eddi4TEX, the editors Redit and ET (see Reese's article above) and Petr Olsak's menu builder 'mnu'. All of these have some or all of the following characteristics:

- a model of the original Turbo Pascal — character screens with pull-down menu bars;
- protection from the horrors of DOS environment variables controlling programs from four different authors (*after* you have got the shell properly configured);
- TEX and other programs activated by 'shelling out' to DOS, (an intrinsically 'uncool' procedure), with parsing of error messages;
- heavy use of customizable environments (see Figure 4), which can confuse the novice to screaming point;
- limited editors which can be difficult to use with 'industrial strength' documents;
- access to online TEX context-sensitive help.

This is not to deny the great usefulness of parts of all of the DOS TEX shells, but their parts are greater than their sum. The dedicated DOS TEX user will find them extremely useful, but perhaps they can be summed up by saying that they will not convert many Microsoft Word users.

A recent initiative by Dutch TEX users has resulted in 4TEX (see the article by Wietse Dol), a TEX shell accompanied by a packaging of almost all the good public domain TEX goodies. This is undoubtedly the system which is getting most support and development. The even more powerful AsTEX setup of Michel Lavaud (described in *TUGboat* 14.3, 238–44) unfortunately depends on commercial software for its implementation, but those interested in a wider perspective on *managing* their TEX document library should examine this closely.

Those who want the approach of Scientific Word (WYSIWYG maths composition) in plain DOS can try the public domain ET by John Collins, or the commercial *Leo* from ABK Software.

## 3 Macintosh systems

The Macintosh user does not need convincing about GUIs or integration; for hard cash, she can buy what is probably the fastest (given the processor), slickest TEX around, Textures. A port of the Unix TEX is available (CMacTEX), and a half-way house, the shareware OzTEX, which offers integration between a good compiler, previewer and printer driver, but not quite the smooth transition of Textures, or the use of system fonts. What is lacking on a Mac is a WYSIWYG interface to mathematical writing such as that offered by Scientific Word.

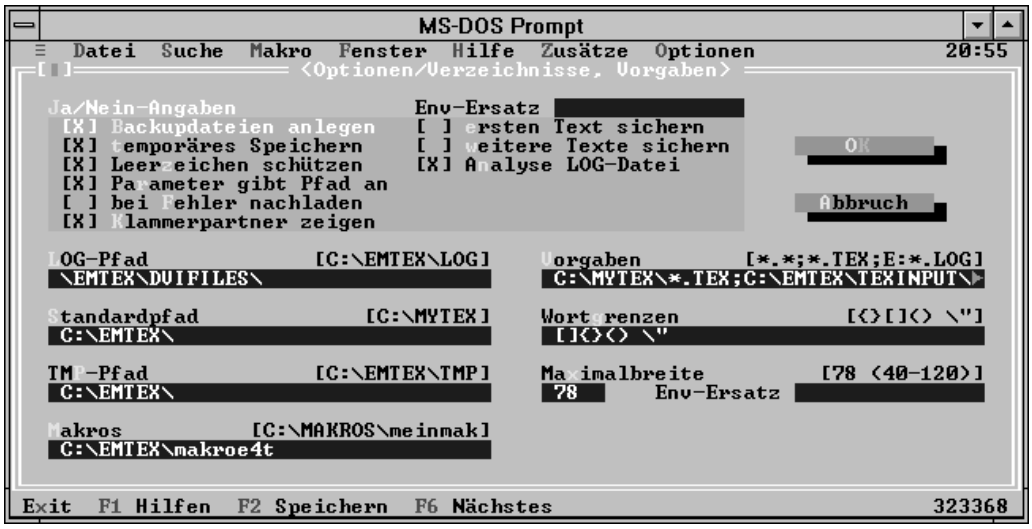



**Figure 4.** Eddi4TEX customization

## 4 Unix, etc

The Unix TEX user has a rich variety of tools to play with; like everything else in Unix, they are dangerously powerful, and require dedication to obtain true mastery. They start with simple things like the *csh* or *bash* history mechanisms, which let you recall previous commands by name:

```
$ history
  171  lutex fine2
  172  dvips fine2
  175  latex interfac
  176  lpq
  180  dvips fine2 -o fine2.ps
  182  cp fine2.ps /user
$ !lat
$ !dv
```

Another universal Unix tool is ‘make’ which manages the relationship between different stages of a document’s ‘compilation’ from various sources, and keeps things up to date. A large book, with dozens of input files, hundreds of graphics inclusions from different programs, a bibliography managed with BIBTEX, and an index, is likely to need such a tool to keep the author sane. There are various tools to help you write ‘make’ files, and to do similar jobs to ‘make’:

**imaketex** Just write an *imake* file and you just type ‘make’; as in ‘just write’…

**texit** A Perl script which makes educated guesses about the state of your TEX file and helps you decide what to do next;

**latexmk** Given a LATEX file this tries to work out what to do, like a ‘makefile’; but does not understand multiple-level dependencies;

**latex_make** A set of rules for real ‘make’, which understand a wide range of common TEX jobs; but tailoring is not for the beginner. Sample ‘make’ rules are shown in Figure 5.

If you are already a Unix programmer, these are well worth investigating.

There do not appear to be any obvious public domain TEX shells for the X Window system; the ‘doc’ editor does provide help in writing LATEX, with limited immediate formatting, but it still leaves you to run TEX. The Berkeley VorTEX program to produce an interactive TEX ended some years ago without a real success (although various results have now been released for general use; they are available in CTAN).

The queen of Unix software is Gnuemacs; as well as being an *extremely* sophisticated editor, it provides hooks to run jobs synchronously or asynchronously from within the editor, and parse the output. The latest X Window version allows for user-defined menu bars, and multiple windows, which allows the Unix user access to as sophisticated a setup as almost everything except Textures. The actual editing can be enhanced with powerful Lisp routines to check syntax

```
# ensure that the bbl file gets regenerated if the bib file is changed
%.bbl : $(BIBFILES)
        @# if there is no aux file, skip this, it will get done later
        -@if [ -r $*.aux ] ;\
        then $(BIBTEX) $* ; \
             fi
# create a dvi file from a tex file
% %.dvi: %.tex
        $(LATEX) $*
        -@egrep -c 'Citation .* undefined.' $*.log && ($(BIBTEX) $*;$(LATEX) $*)
        -@grep 'Rerun to get cross-references right' $*.log && $(LATEX) $*
# create postscript file of a dvi file
%.ps: %.dvi
        echo dvips -D $(PRINT_RES) -p $(PAGE_F) -l $(PAGE_T) -o $@ $*
        dvips -D $(PRINT_RES) -p $(PAGE_F) -l $(PAGE_T) -o $@ $*
```

**Figure 5.** An example of 'make' rules for a TEX file

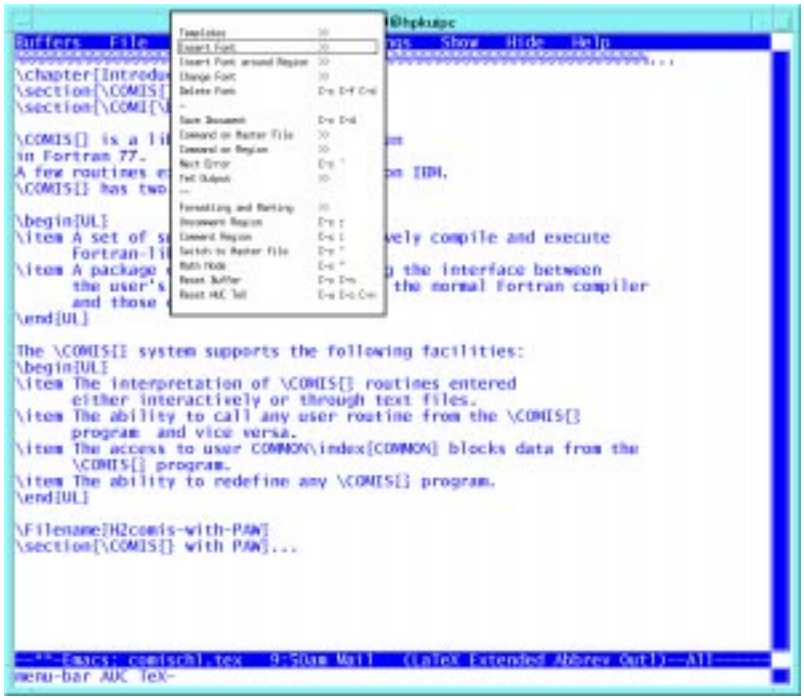



**Figure 6.** Gnuemacs and AUCTEX

provide structure skeletons, edit in outline mode etc; the best-supported and enhanced package is Kresten Krab Thorup and Per Abrahamsen's AUCTEX. If you can learn to be happy with Gnuemacs, this has almost limitless possibilities. An example screen is given in Figure 6.

## 5 Conclusions

If you have to choose a TEX environment, how do you start evaluating the offerings? I would suggest that base your choice on the following criteria:

- The editor; this is where you spend most of the time;
- On-line help; LATEX is unmemorable;
- Integration with windowing systems; doing just TEX is dull;
- *Lack* of special features; compatibility with other TEX setups is important;
- Integration with the 'style' of the environment (ie use of Adobe Type Manager).

For the five possible environments you might consider choosing:

- AUCTEX: if you use the X Window system and *gnuemacs*, there is no other way to live;
- TEXshell: if you like plain DOS, this is well-designed and easily customized;
- Dviwindo: it really is a Windows application, and understands PostScript fonts;
- TEXtures: it works; it has semi-instant previewing; it integrates its fonts with the rest of the Mac;
- OS/2 and emTeX: a very good command-line TEX, a Presentation Manager previewer, and integration with a reliable operating system.

*[Note: all the public domain or shareware software referred to in this article can be found in the CTAN archives. The commercial vendors all advertise in* TUGboat.*]*

# V Shells for TEX

Phons Bloemen
Information & Communication Theory
Dept. of Electrical Engineering
Eindhoven University of Technology
The Netherlands
Phons@EI.ELE.TUE.NL

*[Editor's note: I am grateful to Phons Bloemen and Gerard van Nes (editor) for permission to reprint this article from MAPS 93.2, the journal of the Nederlandstalige TEX Gebruikersgroep.]*

## 1 Introduction

The well-known emTEX package of Eberhard Mattes has brought TEX and its companions into the reach of the PC-compatible computers. Besides a good implementation of TEX and METAFONT (even for various processors like 8088, 80168 and 80386, which make use of additional features like 'protected mode', and special 'big' versions), emTEX offers an excellent previewer, which works on various graphic screens from Hercules to (S)VGA. There are also printer drivers for HP Laserjet and for nearly all dot matrix printers (by making a little configuration `.dot` file). A dozen handy utilities come with the package, like BiBTEX, MakeIndx, TEXcad, and MFJob.

Because this is a collection of various programs, some way of integration must be found, especially for users who want to make TEX their main document preparation system, and do not want to be bothered by MSDOS commands. Furthermore, an integrated environment consisting of an editor with function keys for the various TEX operations and utilities would greatly improve the acceptation of TEX by a large group of users, especially in 'Wordperfect country: The Netherlands'.

I reviewed two environments by installing them on my home emTEX installation, and on a 'clean 1990 emTEX distribution'. The latter is the distribution of 6 high-density disks of emTEX, as it was released in 1990. Since, there have been various updates on several programs, but more about that later. My own emTEX distribution runs on a 386/25Mhz clone, and is up-to-date, with the beta-test releases of TEX, dvidrv and METAFONT installed. I also installed babel and NFSS, and a large collection of fonts, and the amSpell spell checker. The packages were also tested on a 286/8Mhz clone with Hercules display. After installation, I used the two environments to make some documents (including this one).

## 2 TEXShell

TEXShell is an integrated environment for the emTEX family, authored by Jürgen Schlegelmilch. It is based on Borland's Turbo Vision, and it looks a lot like the integrated environment of Borland's programming languages like Turbo C and Turbo Pascal. Version 2.6 comes in a `.ZIP` archive of 619 kilobytes. It can be obtained from various FTP-servers, like `ftp.uni-stuttgart.de`. The archive contains a German and an English version. Version 2.6 was released in September 1993. Most of the evaluation is based on version 2.5.2.

### *2.1 Installation*

The installation is simple: unzip the archive and the files will be placed in the correct directory in the `\emtex` hierarchy. Then the file `\emtex\TEXSHELL.CFG` must be edited to record the drive on which it is installed. It is possible to install TEXShell in a user-defined directory, but this is not recommended. No files of the already installed emTEX need to be changed. Start the program with

```
c:\emtex> texshell
```

Now the TEXShell manual advises you to check the *Options* menu, to examine and adjust the other options to your own needs. The *Options* menu has a consistency check built in: if you enter something impossible (a reference to a program you do not have, or an incorrect path name), the input window will not disappear, and the cursor keeps blinking on the error. The preloaded configuration is sufficient for a base installation of emTEX (i.e. the '1990' distribution, with the german format files).

*2.2 Customization*

If you want to change the external commands, or install new ones, you have to dig into the manual. TEXShell is very powerful in calling external programs, but it takes some study to understand how to make such a call.

There are a few predefined menu options for the 'standard' emTEX utilities such as dvidrv, BiBTEX, TEXcad, MakeIndx, these also have their own predefined 'hotkeys'. With respect to the dvi-drivers: it is possible to install a screen previewer, a draft and a final printer (in most cases: `dviscr, dvidot, dvihplj`). This flaw has been corrected in version 2.6: here you can install up to 32 printing devices, and a screen previewer. There is also a *User* menu, where you can install your own programs: I tested this with amSpell.

For all external programs, the installation procedure is the same: you can choose from a series of *templates* to construct a command-line to call external programs like TEX itself, `dviscr`, `dvidot` and so on. These 'templates' depend on the filename being edited, the format file used (TEX, LATEX) and various directory names you can install in the *Options* menu. Furthermore, you can assign an *environment* to each call. Settings in this environments are provided as DOS environment variables to the program when it is called from TEXShell. If you call a batch file as external command, the COMSPEC variable must be set in the environment (this is not mentioned in the manual)! The process of installing your own tools is tedious, and there are a few caveats.

For the 'bare metal hackers': the configuration file `TEXSHELL.CFG` is a plain ASCII file. It looks a lot like a Windows `.INI` file, and you can edit it to your wishes.

*2.3 Editor*

The editor provided by TEXShell is much like a standard Turbo Pascal/Turbo C editor. The command structure is like old Wordstar with its Ctrl-K and Ctrl-Q commands, it works very quickly. The editor has a limit of 64 kilobytes per file, but it may have several files opened at the same time. It is possible to transfer text from one file to another using a clipboard and cut-n-paste. There are 10 function keys to put macros under. The editor provides wordwrap, but there is no way to make this the default. It also provides full mouse support for pointing the cursor, moving, sizing, closing windows and menu bar selection.

The package looks very colorful. On the slow machine with the Hercules display, the default 'colors' do not look very well. A little playing with the 'color customization' menu improved things a lot, but the program does not recognize monochrome displays.

There is a nice edit file selection scheme, which again much resembles the one used in Turbo Pascal. The editor provides a 'primary' TEX file: when TEX is run, it starts with this file. TEXShell records its configuration together with the primary file, by writing a `.DSK` file.

*2.4 Compile and preview*

There is a menu *TeX* on the menu bar which gives access to these functions. Most of them have a 'hotkey'. A very nice feature is an separate menu to view the log file of the command just executed in a *Log Window*, varying from TEX to previewer and BiBTEX. The 'e' option of TEX (provided when there is an error) is fully implemented, and with Ctrl-Q E / Ctrl-Q W you can go to the next/previous TEX error or warning! You can also use the *Log Window* to find out from which file an error originated, by point-and-shoot on the errors in the log file. Great if you are editing a big document, or are playing around with user-written style files.

The previewer can be operated by a 'hotkey' without any problems (only the installation of MFJob took some time, but everything works fine).

*2.5 Help system*

F1 gives context sensitive help: it provides help for the function you have selected. There is 'help on Help' and with Sft-F1 you can browse through an index and view the whole help file. The help files are in English or German. There is also online TEX help: in an edit window you can put the cursor on a TEX command, and get help on that command with Ctrl-F1.

*2.6 Manuals*

The TEXShell manual seems to be 'not yet ready'. It is more a listing of all the features of every window. It contains a little section on what files you should have, and how to install TEXShell. Then there is a extended explanation of the use of *templates* and *environments*, and how to install new commands. The last section of the manual explains how to use TEXShell: it is a list of command grouped per menu on the menu bar, and a list of the editor functions. Everything is documented, but there is no such thing as a 'guided tour'. There is an appendix containing a revision history.

*2.7 Miscellaneous*

This section contains a few remarks on other features of the system.

- Installation of new format files is simple when you have generated the appropriate `.FMT` files. However, it is not possible to assign emTEX command line options to each format file. Version 2.6 has a new option to do this, but you can also work around it with batch files.
- TEXShell provides good printing support. Before printing there is a menu to enter first and last page, and the number of pages to print. The 'complete' print job is given as a default. You can enter your own dvidrv options (if you want to print two A5 sheets sideways).
- TEXShell provides no spelling support. With some effort, I managed to install amSpell in TEXShell, using the *User* menu and a couple of batch files. amSpell provides all spelling checking you want.
- TEXShell does not support METAFONT when run 'stand-alone' You then have to install it in a batch file in the *User* menu. METAFONT is supported through MFJob for automatic font generation.
- I had to change the 'default' `TEXSHELL.CFG` to customize it to my needs (installed amSpell, METAFONT, MFJob, BiBDB). Contact me for copies of the `TEXSHELL.CFG`) file.

## 3 4TEX

4TEX is authored by Wietse Dol, Erik Frambach and Maarten van der Vlerk, from the University of Groningen. This integrated environment is based on JP Softwares 4DOS and SemWares Qedit. The system is a large collection of 4DOS 'batch' `.BTM` files, which perform the various file-handling actions and call the programs from the emTEX collection. To make the system work nice, the authors have written several little utility programs to perform selection functions.The system was written to operate in networked environments, where the emTEX files are placed on a Novell network server, and the users run the programs from the network disk. The review is based on version 2.15 of 4TEX. I will make some references to version 2.20: this is a 'βtest-version' for the upcoming version 3.0 of 4TEX.

### *3.1 Installation*

4TEX was founded as a 'complete' system, containing the complete emTEX package, amSpell, Ghostscript and so on. The archive `obelix.icce.rug.nl` in Groningen has organized the package in several parts: take a look at the various `.TXT` files. The shell itself hides in `4TEX.ARJ`. This package is intended for use on an existing 'standard' emTEX installation. It is about 3 megabytes, and contains the 4DOS `.BTM` files, updates to various emTEX utilities, and the shareware distributions of PKZIP, Qedit and 4DOS version 4. As a bonus, over a 100 new style files will flood your `\emtex\texinput` directory. The 4TEX shell itself is about 300 'compressed' kilobytes, and the 'isolated' shell can be installed on an existing emTEX installation.

Installation is done as follows:

- Unpack the archive in the root of your emTEX drive with `c:\> arj x -v a:4texupgr.arj`. The use of ARJ is necessary because the archive spans more than one floppy (and ARJ compresses better than `PKZIP 1.1`). The whole package is put into the right directories.
- Some emTEX files must be deleted or moved: the 'compilers' `tex.exe, btex.exe` etc. must be moved to `c:\emtex\compilers`, and the utility programs `bibtex.exe, texchk.exe, makeindx.exe, texcad.exe, texchk.exe` to `c:\emtex\utils`.
- You must put `'LASTDRIVE=T'` in your `CONFIG.SYS`
- You must edit `c:\emtex\btm\system.set` and `c:\emtex\btm\texuser.set` to configure the system to your needs. If you have a 'standard' emTEX, you don't have to touch `system.set`. In `texuser.set` you have to set your personal defaults. Both files provide extended comments on each of the options, but they are quite long at first sight.

The system is started with

```
c:\emtex\btm> tex
```

There are two batch files `tex.bat` and `tex.btm`: if you are already running 4DOS, it will not load it again (However, there are some restrictions to your personal `4DOS.INI` file, and you must have loaded KSTACK). If you are running MSDOS or DRDOS, a new 4DOS command shell will be loaded with the right settings. After a while, the main menu of 4DOS is presented.

### *3.2 Customization*

All customization of 4TEX is done via the configuration files `texuser.set` and `system.set`. There is not much need for the possibility to install 'own' utilities, because there are a lot of utilities predefined, like Ghostscript, various graphics utilities like BM2FONT, bibliography maintenance with BiBTEX and BiBDB, spelling check with amSpell

and much, much more. Even some commercial software like WORD FINDER and EUROGLOT is supported. However, recent versions of 4TEX do offer the opportunity to install own utilities, in user-defined menus.

A system that is made out of 'human-readable' batch files (`.BTM`) is a paradise for 'hackers': if you study the manual of 4DOS (included in the distribution), you can add your own enhancements to the system. Not recommended to keep compatibility with the official release, but hard to resist.

### *3.3 Editor*

The editor of the 4TEX system is the shareware editor Qedit. The authors have defined a set of macros to use with Qedit. These extra macros provide 'hotkeys' to start the emTEX 'compiler' and the previewer, to start the amSpell spell checker, the BiBDB bibliography database program.

An editor help screen is brought up by F1, and it shows the standard Qedit editor commands, as well as the special extensions. The Qedit editor is capable of having more files open at the same time. Spell-checking is done by amSpell on a word basis in the editor, or on file basis from the main menu. Before amSpell is started, you can choose your language.

The mouse is supported in the Qedit editor for cursor pointing, block definition and window switching. In the main menus, the mouse can be used to point to menu choices. 4TEX also provides the 'primary file' system as described in TEXShell. It also records its configuration together with the primary file, using the extension `.OPT`.

### *3.4 Compile and preview*

The emTEX 'compilers' are started from the main menu in most cases. There are also 'hotkeys' to start them from the editor. A nice option is to start the emTEX compiler and previewer from the editor with a block selected: then only that block of text is TEXed and shown. 4TEX is intelligent about choosing the right documentstyles for such a text block: it examines the primary file and puts its preamble together with a surrounding `\begin{document}` and `\end{document}` around the text block. The error processing is not as good as that of TEXShell: the `'e'` option works, and you can view the log files, but that is all. The error checking will be improved in version 3, according to the authors.

The complete 4TEX system comes with LATEX format files with babel and NFSS installed. The format file contains hyphenation patterns for English, German, French and Dutch. In emTEX's memory organization, they set aside 65K words for hyphenation patterns where 36K should be enough according to my information. This leaves little main memory to do the real TEX jobs in (like processing elaborate tables and figures). So the notorious message `TeX capacity exceeded` appears (too) soon.... Version 2.20 has the capability to generate new formats 'on the fly' where you can choose the languages to include from a menu. It then makes a format with just enough space for the hyphenation patterns you choose. A TEX with 7 languages loaded is possible!

### *3.5 Help system*

4TEX provides two different help systems. First, there is help available for each of the menus of the system explaining the various options in short. Second, there is a good 'online' TEXhelp system called TEXHelp. This is a popup command which you can access everywhere with Alt-F2, Alt-F3, Alt-F4. It provides help on the TEX command the cursor is on, and you can navigate through its index. TEXHelp is a TSR program, taking 9 Kb of main memory, and 250 Kb EMS / disk swap space. The help is in English, but it is possible to install a help file in a different language.

### *3.6 Manuals*

The 4TEX manuals are very elaborate. In fact, they provide information on every aspect of the whole emTEX system, and all its utilities. Chapter 2 is fully devoted to using 4TEX itself, and to install it. It contains a section of format-file generation, and installing babel and NFSS.

Then the manual covers the following topics in brief, concise sections: the emTEX 'compilers' themselves, the dvidrv programs, Postscript, bibliography and index support, the amSpell spell-checker, a section on importing graphics in various ways (TEXcad, BM2FONT), some miscellaneous utilities like `detex`, the TSR programs like TEXHelp and commercially available translation programs, and finally something about fonts and METAFONT/MFJob font generation. This is more than a manual, it is an example of how a 'local guide' should look like.

The manual concludes with a list of sources for more information, like TEX users groups (TUG, NTG), and distribution lists (`TEX-NL`). An extended bibliography for further reading is added.

### *3.7 Miscellaneous*

- 4TEX provides support for different types of printers. The printer type can be chosen from a menu in the *Output* menu. There it is also possible to enter the printer port, first and last page, and the number of pages to print. You

can enter your own dvidrv options Postscript printers are supported via `dvips`, and even the use of Ghostscript to get the postscript files on a simpler printer is possible (not tested).
In earlier versions, the printer support was poor, especially for matrix printers: only a 9-pin was supported, and there was no opportunity to install user-defined printers. This was reported to the author, together with a proposal for a better way to do it (4TEX hacking is easy). Version 2.20 contains much better printer support.

- 4TEX provides an extra *Graphics* menu where you can do all types of conversions of pictures to get them into your TEX documents, provided the necessary programs are installed. It also provides extra *BiBTEX* and *MakeIndx* menus to manage bibliography databases and automatic indexing.
- Because 4TEX was initially developed to run on a network, there is an extra menu called *TEXbatch*. Here you can send print jobs to network printers and batch TEX jobs to fast computers on the network.

## 4 Conclusions

To conclude this review I give a short 'pro-and-con' list for both systems. First I want to remark that the authors of both systems did a good job, and that the systems are still in a development phase. The use of a 'shell' around the emTEX system is very useful, and it can speed up things even for experienced TEX users. It is also a step on the way to make TEX more attractable to the 'average Wordperfect user', and to a TEX ' beginners package'. Some attention must be given to the fact that TEX 'beginners' should not be bothered by lengthy installation procedures: it should be 'plug-and-play'.

TEXShell in brief:

+ The use of Turbo Vision gives TEXShell a professional look.
+ There is an excellent system of tracing TEX errors in your file.
+ The package is rather small (about 300K).
+ Powerful mechanism to call external programs.
+ The package is reasonably fast, even on slow machines.
– Hard to configure. A major drawback, if there is a 'universal' `TEXSHELL.CFG` file which covers the complete emTEX distribution and some related things, this would ease the installation in a great way.
– Lacks 'integration' of some utilities like amSpell.
– The manual needs some rewriting (especially if it is going to be used as a beginners guide).

4TEX in brief:

+ The package covers almost everything in the emTEX package, and even more.
+ Excellent manual, discusses also the emTEX utilities.
+ The TEX help is given by a stand-alone TSR program.
+ Spell-checking with amSpell is an integrated part.
– The package is slow (annoying on the 286 clone).
– The package is rather bulky, because it is packed with a lot of other stuff. The shell itself is 300K.
– Installation procedure is elaborate.
– The use of 'human-readable' programs makes the system vulnerable to users who 'customize' it by changing the program itself. But this also may be an advantage…

# VI Text to Hypertext Conversion with LaTeX2HTML

Nikos Drakos,
Computer Based Learning Unit,
University of Leeds, Leeds LS2 9JT, UK.
email: nikos@cbl.leeds.ac.uk
www: http://cbl.leeds.ac.uk/nikos/personal.html

*Summary*
LaTeX2HTML is a conversion tool that allows existing documents written in LaTeX to become part of a global multimedia system. This paper presents some of the reasons for using such a system and describes the basic conversion process.

## 1 World Wide Web — A Global Multimedia System

> Imagine a system that links all the text, data, digital sounds, graphics and video on all the world's computers into a single interlinked hypermedia "web". This is the potential of the Internet-based World Wide Web (WWW or W3) project . . . [2]

The World Wide Web merges hypermedia techniques with networked document retrieval to provide a global information system of linked documents. These are traversed by "clicking" in textual or iconic active areas, or searched via query mechanisms [1]. Hypertext links may point to a different location in the same document or to another document which may be located perhaps in another continent!

Documents are not limited to containing only textual information and may include high resolution images, audio and video samples. WWW also encompasses most of the services currently available on the Internet such as Usenet news, ftp, wais, archie, etc. Access to these services as well as the invocation of arbitrary computer programs (e.g. a database access or a simulation) is completely transparent to the user who sees them all as part of some document and interacts with them in a uniform and intuitive way.

Multimedia documents are written in a language designed specifically for the World Wide Web called HTML (HyperText Markup Language) which is based on SGML (Structured Generalised Markup Language). Documents are written by information providers who just place them on the WWW using a "server" program. Then anyone with access to the Internet can use a "client" or "browser" program to access and view available documents. Clients and servers communicate via the HTTP protocol (HyperText Transfer Protocol). Apart from navigation facilities, browsers also allow full text searches, "cut and paste", text or audio annotations, personal "hotlists", saving and printing in multiple formats and others. Such browser and server programs are freely available for most popular computer configurations.

With the explosive growth of the World Wide Web (500-fold since the first graphical browsers were made available this year [3]), and a potential audience of 15 million in more than 50 countries, providing information via the WWW is becoming an extremely attractive proposition.

## 2 LaTeX to HTML Conversion: Why?

HTML is quite a simple markup language to learn and use. It allows basic formatting commands, bulleted lists, "inlined" images, and hypertext links to other documents, multimedia sources, internet services or computer programs. But despite (and because of) its simplicity it has created a few headaches for information providers:

- there are no intuitive authoring tools (yet);
- yet another hypertext language has to be learned;
- existing documents available in other formats have to be reprocessed;
- hypertext document "webs" are difficult to maintain;
- it is difficult or impossible to create highly formatted documents in HTML.

LaTeX2HTML can be used in order to address to a large degree these problems. The authoring problem simply disappears, existing documents can be reused immediately and a complex web of interlinked documents can be generated from a single source document. The automatic inclusion of formatted information such as tables or mathematical

equations as inlined images also bypasses another serious problem with HTML. An additional benefit is that the paper-based version of a document can also be obtained from the same source.

The utility of a conversion tool like LaTeX2HTML can be seen from the variety of contexts in which it has been applied. Some examples are listed below.

- Electronic books (e.g. that produced by the Computational Science Education Project[2] which is sponsored by the US Department of Energy. This is one of the most complex documents currently available via the WWW.).
- General reports (e.g. the annual report of the Institute of Astronomy at Cambridge[3]).
- User manuals[4].
- System documentation[5].
- Scientific papers such as those on the MIT Transit Project[6].
- Electronic journals (e.g. Complexity International[7] — a new Australian electronic journal).

## 3 LaTeX to HTML conversion: How?

The basic conversion process relies on the ability to distinguish between the *structure*, the *content* and the *formatting* information in a LaTeX document.

On the basis of sectioning information, a document is broken into separate parts and an iconic navigation mechanism is constructed in HTML which reflects this structure and allows a user to "jump" between different parts. The cross-references, citations, footnotes, the table of contents and the lists of figures and tables are also translated into hypertext links. Formatting information which has equivalent "tags" in HTML (lists, quotes, paragraph breaks, type styles, etc.) is also converted appropriately.

Although in most cases the loss of some formatting information (e.g. page margins or line widths) is harmless, there are occasions where the format has meaning e.g. when dealing with tables or user defined environments. Another problem is the replication of the mathematical equations which must retain both their precise format as well as any of the predefined special mathematical symbols.

The innovative solution in such cases relies on the ability of HTML browsers to display inlined images inside the main text. Any part of a LaTeX document for which it is not obvious how it should be translated directly into HTML is extracted from the main document and then placed on a pipeline which converts it into an image. Each image is then placed at the correct position in the final HTML document. Special care is taken to preserve contextual information that may affect the contents of each image (counter values, labels, references, active style files etc). Some examples of converted documents can be seen in Figure 1.

## 4 Hypermedia Extensions to LaTeX

Apart from the obvious hypertext links within a LaTeX document (e.g. navigation between sections, cross-references and citations) it is also possible to take full advantage of the HTML links to arbitrary multimedia sources (e.g. audio or video), electronic forms, and other remote documents or internet services.

This can be done with some new commands defined in a separate style file (`html.sty`) which are processed in a special way by the LaTeX2HTML translator. This style file defines commands for embedding external hypertext links, for extending the basic `\ref-\label` mechanism to operate between remote documents, and specifying that some text should only appear in the paper-based version or only in the HTML document. In most cases these commands have no effect when processed in the conventional way.

Another command allows the inclusion of arbitrary HTML markup directly in a LaTeX document. This can be used to take advantage of new HTML facilities as soon as they become available (HTML is currently evolving towards a new specification called HTML+). A particularly good use of this feature is in the creation of interactive electronic forms from within a LaTeX document.

[2] http://compsci.cas.vanderbilt.edu/csep.html
[3] http://cast0.ast.cam.ac.uk/sub_dir/cambridge/annual_report/annual_report.html
[4] http://cs.indiana.edu:80/elisp/w3/docs.html
[5] http://archie.ac.il:8001/papers/papers.html
[6] http://www.ai.mit.edu/projects/transit/tn-cat.html
[7] http://life.anu.edu.au/ci/ci.html

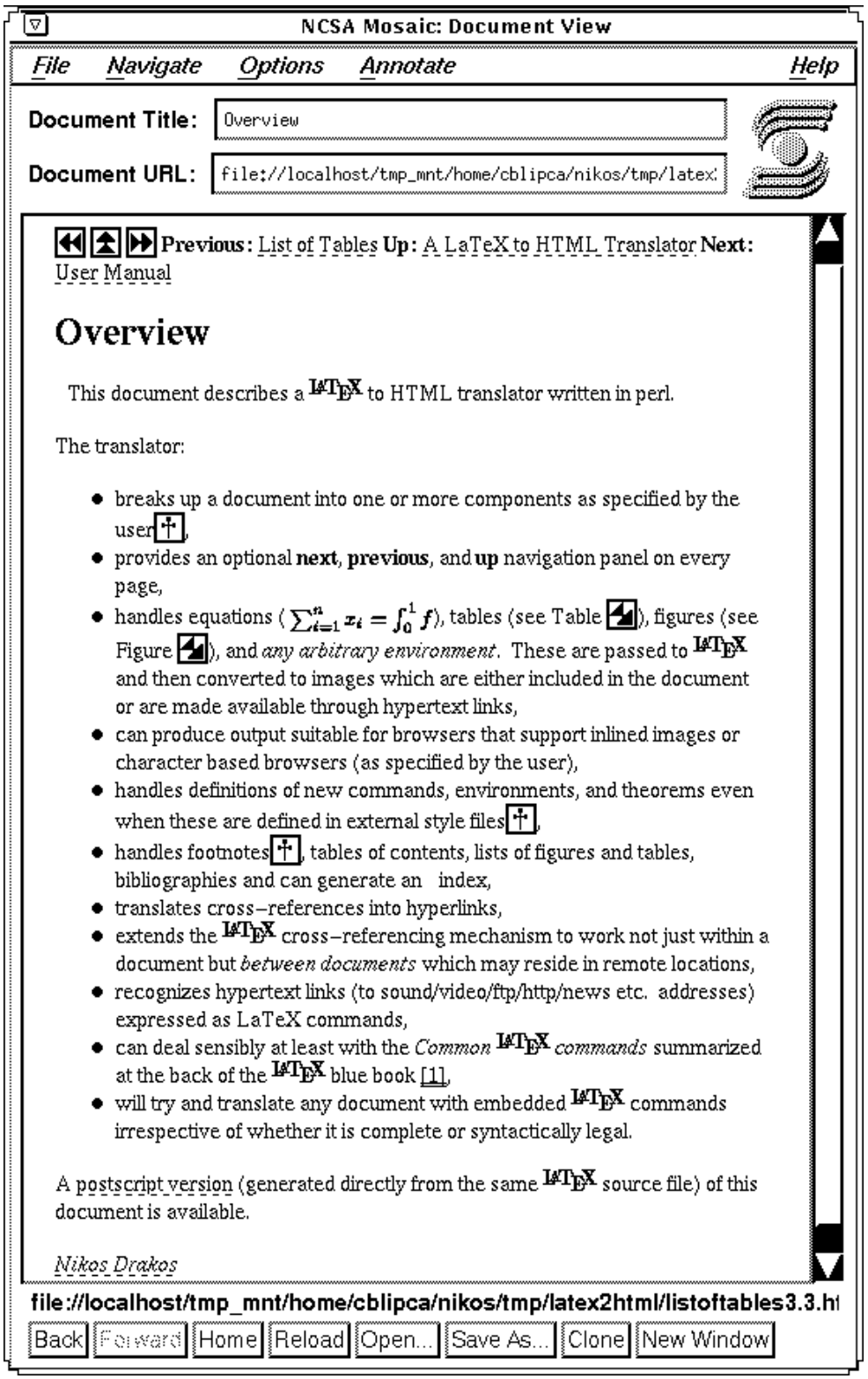



**Figure 1.** A converted document displayed using Mosaic

## 5 Concluding Remarks

Conversion tools like LaTeX2HTML provide an easy migration path from familiar concepts towards authoring complex and format-rich hypermedia documents. In this way, familiarity with a system like LaTeX makes it possible to contribute to and benefit from a rapidly expanding global hypermedia network.

## References

[1] T. Berners-Lee, R. Cailliau, J. Groff, and B. Pollerman. Worldwide web: The information universe. *Electronic Networking: Research, Application and Policy*, (1), 1992.

[2] Joe Levy. The world in a web. *The Guardian*, page 19, November 11 1993.

[3] Vern Paxson. Growth trends in wide-area TCP connections. *IEEE Network*, To Appear 1993. Available at ftp://ftp.ee.lbl.gov/WAN-TCP-growth-trends.revised.ps.Z.

## 6 Further Information

LaTeX2HTML is written in Perl and requires freely available software. More information on how to get, install and use it is available via the WWW[8] or using anonymous ftp from ftp.tex.ac.uk in pub/archive/support/latex2html. A new release is planned for early December 1993.

Several computers on the Internet have public access World Wide Web clients accessible by telnet e.g.

- telnet info.cern.ch (direct connection — no username or password required)
- telnet ukanaix.cc.ukans.edu ("Lynx" requires a vt100 terminal. Log in as www.)

Information on World Wide Web is also available via anonymous ftp from `ftp.germany.eu.net` in `pub/infosystems/www`. The Mosaic clients are in the directory `/pub/infosystems/www/ncsa/Web`.

[8] http://cbl.leeds.ac.uk/nikos/tex2html/doc/latex2html/latex2html.html

# VII Some notes about MS-Windows and TEX

Wietse Dol
Landbouw-Economisch Instituut (LEI-DLO)
P.O.Box 29703
2502 LS Den Haag
The Netherlands
email: W.Dol@LEI.Agro.nl

*[Editor's note: I am grateful to Wietse Dol and Gerard van Nes (editor) for permission to reprint this article from MAPS 93.2, the journal of the Nederlandstalige TEX Gebruikersgroep.]*

## 1 Introduction

I am a real DOS (4DOS) addict[9] and in principle do not like to work under *Windows*. I really detest the many mouse clicks one needs to get simple things done. Things that are done with one or two really simple 4DOS commands (e.g. file moving) need many mouse movements. Perhaps I am an old fashioned guy not realizing that the world is changing…

There are two application that make it worth while using *Windows*. The first one is the multitasking environment of *Windows*. Of course there are ordinary DOS programs (e.g. Quarterdeck Desqview) that do the same but a nice graphical environment with icons makes *Windows* a really user-friendly multitasking environment. The second and most convincing advantage of *Windows* is its graphical interface. There are many excellent graphical packages (e.g. Coreldraw) that allow you the create, manipulate, convert and print all kind of graphics. So when talking graphics one really should work with *Windows*.

People who work with TEX are all people who like to create texts of the highest quality. The greatest disadvantage (others would say advantage) of TEX is that it is not WYSIWYG (what you see is what you get). After the compilation of a TEX document we all want to use some kind of graphical interface to view (see) the results. So we are talking graphics. We also would like some kind of multitasking, even better a straight compilation and viewing of our TEX code while we are typing the text. This all should be possible with *Windows*…

An operating system quite similar to *Windows* for a PC is OS/2 (sorry, but OS/2 is much better than *Windows*). For OS/2 there are several excellent TEX programs and utilities. For example emTEX (absolutely free of charge!) and ASTEX (see MAPS 93.1 page 41). These programs all have the multitasking and excellent graphical display as mentioned above. People who really like to use some kind of TEX for *Windows* will find out that there is no such thing yet. There is a commercial package Scientific word which claims to be a TEX à la WYSIWYG (see MAPS 92.2 page 147). It really looks promising but is not a real and complete TEX system for *Windows*.

After reading this introduction the question arises 'what should we use under *Windows*?'. The next sections will discuss some *Windows* and TEX topics and hopes you to inform how one could set up a *Windows*TEX system.

## 2 emTEX and Windows

When looking at TEX PC packages there is one that is state of the art: emTEX. It is a pity that E. Mattes 'only' developed a MS-DOS and an OS/2 version. emTEX is free of charge and is to my knowledge the best TEX PC package there is. It offers for the novice and advanced TEX user everything one wants. The only drawback to the system is that E. Mattes did not develop a user-friendly TEX shell. This is nowadays no real disadvantage because there are many good shells.

Our first attempt for a *Windows*TEX should be a DOS-window running emTEX. Many people who have tried got really disappointed. When running the 386 version of emTEX under *Windows* we get the error message 'DPMI not supported.' We can use the slower 8088 and 80186 versions of emTEX but we want more… The error message 'DPMI not supported' is the result of E. Mattes own DOS-extender. Running TEX needs a lot of memory or disk swapping.

[9] I am not a *Windows* specialist so do not be offended when I do stupid and clumsy things. This note is intended to inform people and stimulate them to write more about TEX and *Windows*.

When you have a 386 PC or higher with a lot of memory (the manuals suggest 3Mb) the DOS-extender will claim all available memory and use this instead of the slow disk swapping. The DOS-extender will also use the fast 32-bit processor optimally and gain a lot of speed. The second advantage of the 386 version of emTEX is that it uses the memory settings of the BiG-TEX and BiG-LATEX versions. This means that you can compile large documents with many references and labels.

There is nowadays a solution to the DOS-extender problem. Perhaps not the perfect solution but it works and I think in the near future will become idiot proof. Mattes wrote a program (`emxbind -x`) that deletes his own EMX DOS-extender from the `tex386.exe` and adds a small program (`emxl`) to `tex386.exe`. Now `tex386.exe` will by default look for Mattes own DOS-extender (`emx.exe`) on the DOS-path or will look in the environment variable `EMX` what DOS-extender to use. There are two possible memory-extenders you can use

1. The DOS-extender `emx.exe` (version 0.8f), i.e. E. Mattes own DOS-extender. This will only work under DOS and not under *Windows*. To use this DOS-extender set the environment variable `EMX` to
   `SET EMX=C:\EMTEX\COMPILER\EMX.EXE`
2. The DPMI-extender `rsx.exe` (version alpha 0.51) written by R. Schnitker. This will only work under *Windows* and not under DOS. To use this DOS-extender set the environment variable `EMX` to
   `SET EMX=C:\EMTEX\COMPILER\RSX.EXE`

So, simply by updating your `tex386.exe` and adding two memory-extenders to your emTEX system you now can use emTEX under DOS as well as under *Windows*. The DPMI-extender is still in a testing phase (an alpha release), so there could be some bugs. Indeed some bugs are already documented and will be solved in the near future. I have noticed that after running emTEX under *Windows* my log file as well as my dvi file and several others became read-only. No serious problem because with a simple DOS statement these files are no longer read-only (i.e. `attrib -r *.dvi *.log`). A simple batch file solves these inconveniences. All the testing I did under *Windows* did not result in crashes or strange error massages. The error messages that appeared were all the result of the known bug that some files appear to be read-only after `rsx` has opened them. All these errors are easily corrected.

The conclusion of this section is that people who want a TEX for *Windows*, should consider upgrading/installing the emTEX system.

## 3 WinTEXversion 1.0

*WinTEX* is a text editor for *Windows* specialy written for the use with TEX and LATEX. It is written by S. Morin and helps you with all the difficulties of writing documents in LATEX. *WinTEX* is shareware and only costs $25. Not only the ‘normal’ editing facilities are supported but also some ‘tool bars’ and ‘dialog boxes’.

There are three TEX tool bars, i.e. a Text-bar, a Math-bar and a LATEX commands-bar. With the Text-bar you can easily and graphically select fontsize, font type and all kinds of accentuation. The Math-bar graphically displays all LATEX symbols and helps you selecting the correct statements for all math symbols. With the LATEX commands-bar one can select from a list of all LATEX commands. *WinTEX* also has some dialog boxes. There are dialog boxes for the mathematical array, the eqnarray environment, the tabular environment and the LATEX preamble. With dialog boxes the making of the above structures becomes easy.

When you open a new document, *WinTEX* gives you access to dialog boxes that will help you to build the preamble and the style options. With the main dialog box you choose the document style and associated options. Clicking on Page style, Math style or Floating bodies style buttons will open environment specific dialog boxes. Once selected, the options are inserted in the newly opened document.

I could not read many of my own LATEX documents and I really missed the powerful macro possibilities as for instance with Qedit (or TSE). My general conclusion is that the tool-bars and the dialog-boxes makes it really simple to type documents but that there need a lot to be done before *WinTEX* is a real text editor and TEX tool.

## 4 DVIwin version 2.7

The DVIwin driver is written by H. Sendoukas and lets you preview and print DVI files under MS-Windows 3.1. Its main advantages are: speed, compatibility with any raster device with a *Windows* driver, and graphics capability. All screen and printer handling is done through *Windows*, so it should work on any printer supported by the system. You can insert arbitrary graphics files produced by most *Windows* applications, or other standard graphics files (eg. TIFF, PCX, etc.) provided that you have the appropriate graphics filter. The emTEX specials and the PostScript specials to include graphics, however, are not supported.

*DVIwin* is easy to install and also reads fonts from emTEX fontlibrary files (`.fli` files in the directory `c:\emtex\texfonts`). The font-substitution looks much like the one used by emTEX. There only difference is that *DVIwin* does not allow wildcard characters (e.g. `cm*  150 -> cm*  300` is not allowed).

I really like *DVIwin* because every time you switch to *DVIwin*, it checks if the dvi-file that is currently displayed is updated. If updated it will load the new dvi-file and position at the same position (page) as the old dvi-file. This makes it really simple to perform the edit-compile-view cycle.

My general conclusion is that *DVIwin* is an excellent dvi-viewer and printer for *Windows*. I think it will be only a question of time before all emTEX possibilities that are not available yet in *DVIwin* (e.g. automatic font generation, specials) can be used with *DVIwin*.

## 5 A TEX for Windows system

In this section I will explain what I have installed under *Windows* and how I use *Windows* to produce TEX documents.

First I updated the emTEX `tex386.exe` and added the two memory-extenders. These programs will be available on the 27 high density diskettes the NTG will distribute as the complete TEX distribution for the PC. I will also try to get them on all the CTAN servers.

The TEX shell I use is 4TEX. This user friendly menu system can be used to perform all (novice and advance) TEXing. 4TEX version 3.00 (promised to be released in november 1993) will be updated so that it can run every thing in a DOS-window when running *Windows*. The only thing one has to do is to open a new program group, i.e. select in the Program Manger the FILE and then NEW and PROGRAM GROUP and type the new group name (e.g. TeX). After this you can install 4TEX as a new PROGRAM ITEM and use `c:\emtex\btm\4tex.pif` as the COMMAND LINE and use `c:\emtex\btm\4tex.ico` as the icon.

Of course I have installed *WinTEX*, *DVIwin* and some other TEX *Windows* utilities (e.g. `dvips`, `ghostscript` and `gnuplot` for *Windows*) as Program items in the same Program group. Now I have a complete TEX system for *Windows*.

One thing to remember when installing *DVIwin* is that the number of files in the `config.sys` should be at least 50 (due to font loading). Also one needs to copy the `*.dll` files from the directory where one installs *WinTEX* (e.g. `c:\emtex\win\`) to the window system directory (e.g. `c:\windows\system\`). The font substitution file `dviwin.sub` we need to adjust to our own (extra) fonts and then copy it to our font directory ( `c:\emtex\texfonts\`). After installing *DVIwin* one has to start the program and adjust some OPTION settings. First we change the resolution to 300 dpi and then change the OPTION FONT DIRECTORY to `c:\emtex\btm\texfonts \$rdpi;c:\texfiles\fonts\$rdpi`. Now we are ready for action...

*DVIwin* also has two nice utilities. The program `clipmeta.exe` can be used to take a metafile or a bitmap file from the system clipboard and save it to a disk metafile. E.g. you can use `gnuplot` to make nice plots and then paste then to the clipboard and convert this with `clipmeta.exe` to a metafile. The program `wbr.exe` is a text file browser under *Windows*. It is for instance used in combination with *DVIwin* to display the log files.

When I use TEX I first start a 4TEX session and at the same time a *DVIwin* session. I use the menu of 4TEX for all TEXing and switch to *DVIwin* whenever I want to view and print the document. 4TEX has much to offer what is not yet available under windows (e.g. automatic fontgeneration and many other utilities).

The general conclusion is that in the world of *Windows* much is on the move. It will only be a matter of time and there will be a perfect TEX system for *Windows*. Especially the inclusion of all kinds of graphics and the mulitasking (even better real time display while typing) will be possible under *Windows*. Perhaps all this will not be possible under DOS 7.0.

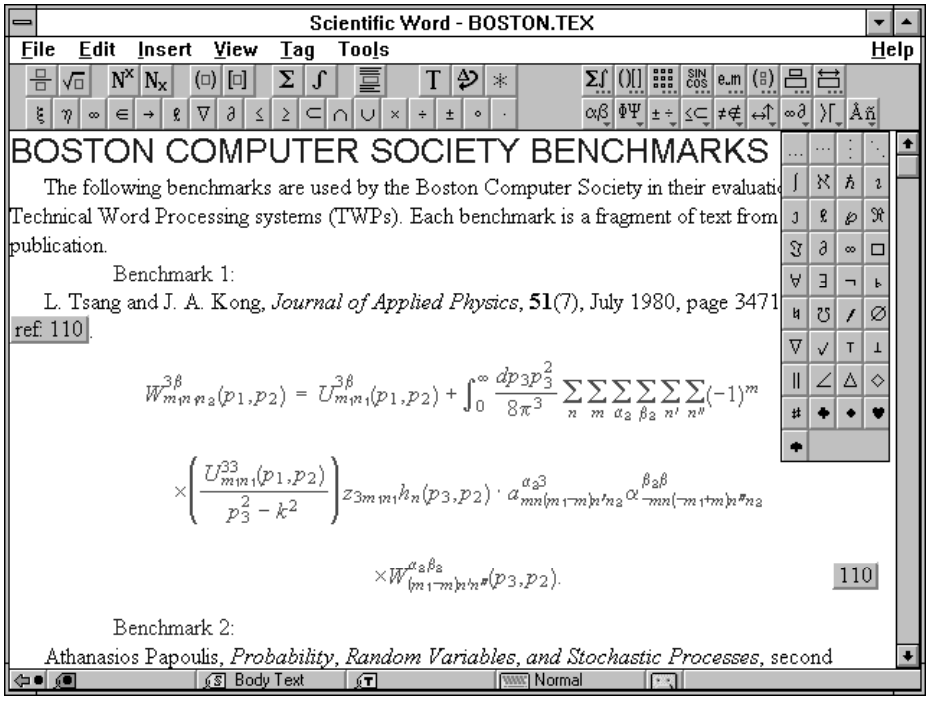



**Figure 1.** Scientific Word, showing user mathematics entry

## VIII Scientific Word

Christopher Mabb

Scientifc Word is 'WYSIWYG' LaTeX for Windows. It was first released in April 1992 (version 1.0) and runs under MS-Windows 3.0 and higher (see Figure 1, showing how mathematical expressions are placed in a document) UNIX and Mac versions are expected in about 12 months. The current version is 1.1, including many small improvements and bug fixes over version 1.0. It comes with its own implementation of LaTeX (TurboTex), but can also be used with PC-TeX, emTeX, (mu)TeX, DVIWINDO and DVIPSONE. A British English spell checker included, and an on-line help system.

Features still to be added include: direct table editing; pasting of graphics from other packages; and a style editor.

The authors/publishers are TCI Software Research, Inc. (USA). The exclusive Distributors in UK & Ireland are:

Scientific Word Ltd.
98 Pont Adam
Ruabon
Wrexham
Clwyd
LL14 6EF
Contact: Dr Christopher J. Mabb (Director)
Tel: (0978) 823088; Fax: (0978) 823066;
Email: `christopher@sciword.demon.co.uk`

Cost: £595 + Carriage + VAT. CHEST Educational discount (33%): £395 + Carriage + VAT. Orders accepted by post, telephone, fax or Email. We do not accept Visa.

## IX New perspectives on TEX macros

Jonathan Fine
J.Fine@pmms.cam.ac.uk

*[Author's note: This article has been prepared from the notes and transparencies I used for the talk I gave at the October UKTUG meeting. As a result, it is somewhat informal and unpolished, and like much conversation, the topic may abruptly change from time to time!]*

Consider the success and failure of TEX, and which triumphs (and disappointments) were expected, and which a surprise. Each person will have their own list. Here are some entries from mine: **Mathematics** and **paragraphs** are expected successes. **Display adverts** is an expected failure; setting of **non-roman** fonts (such as arabic) and the **loyalty of users** are surprise successes. However, **SGML** and **technical documentation**, and **program source** are unexpected failures, as are handling of **floats** (insertions) and (I can expect some correspondence on this item) also **typography**.

Having taken stock of the current state of TEX, let us consider goals: what would we like TEX to be able to do? It is here worth mentioning that, in rough figures, TEX runs 10 times quicker that 10 years ago. This is because more powerful hardware is available. Can we use this additional capacity to make more of the capabilities of TEX?

I have a friend who earns a living using Quark Xpress, and when it was shown to me, I was quite impressed. A certain amount of thought told me that the typesetting was just one part of the capabilities of the program. In any case, when it come to correcting misspelt words, fixing bad page or equation breaks, or bad placement of floats, or even simply adjusting copy to fit space available, it is useful to be able to see and thereby change what is going on. This gives our first goal:

- A visual typsetting system with TEX as its engine.

It is not easy to have TEX process structured documents such as program source files, SGML, etc. We believe (probably correctly) that TEX gives better typesetting than Ventura, for example. But suppose we are given a Ventura document and asked to typeset it. (Ventura stores its documents as ASCII files). There is great difficulty in even having TEX read it, *as a structured document*. The same comments apply to the RTF (Rich Text Format) of Microsoft. This gives the second goal:

- Compatibility with SGML and other file formats.

The concept of a structured document is not built into `tex` the program. Most often, when a typist (one who prepares a document for processing by TEX, who might also be the author) makes an error in tagging the document, it is TEX the program which discovers the error, and TEX and the typist are left to clean up the mess together as best as they can. Typists who use TEX very often need to learn more than they would like of the internal workings of TEX (and the format being used). This is unusual. You don't need to know the *C* programming language to use a program written in *C*. I summarize all these topics in a single phrase:

- Friendly error recovery.

All of the above will require powerful TEX formats, that will, for example, be able to parse a structured document and report on errors. To create these we will require:

- Powerful programming and document design tools.

Having reduced TEX to `tex` the program, I then listed those of its admirable qualities, which I particularly valued. These are:

**Reliable,** it almost always behaves as advertised.
**Stable,** its behaviour does not change from version to version.
**Quality,** its is extremely well-designed and well-written. It produces excellent paragraphs and mathematics (the rest is up to the format designer).
**Widely available,** it runs on an enormous range of machines and operating systems.
**Quick,** it will set text, and expand macros, at a prodigious rate. It really does run very quickly.
**Flexible,** few assumptions were made about how TEX would be used, and so by writing macros it can be used in new and unexpected ways.

The next few paragraphs are somewhat technical, and those who do not know what category codes are, or why they are important, should skip until further notice. What I am proposing to do with TEX may appear a little unorthodox, and so some justification

> … it is best not to play with the category codes very often because … when the arguments to a macro are first scanned … their categories are fixed once and for all at that time. … The author … discourage[s] people from making extensive use of `\catcode` changes …
>
> *The TEXbook*, page 48

from the canonical source is called to support my proposal.

So many problems arise from category codes. The difficulties encountered by verbatim processing are legion. But think of friendly error recovery. When the typist produces an undefined control sequence, a TEX error results. The same applies to a misplaced `$` or `&` character. Even the innocuous (and omnipresent) braces `{` and `}` cause errors. For example, forgetting to turn off emphasised text at the end of a paragraph can result in the rest of the document being mis-set (unlimited propagation of an error) together with the

```
(\end occurred inside a group at level 1)
```

error at the end of the run.

Let us solve all category code problems once and for all by insisting that *the document be read throughout with fixed category codes*. Of course, the format will want 'control sequences' and so forth, so we can let `\`, for instance, be an *active* character, whose meaning will parse the succeeding characters until a non-letter is found, and then turn the parsed string into a control word, and then test the control word for being undefined. This will not be as quick as reading using the usual category codes, but TEX is now so much quicker than when it was first released, that the delay will probably not bother us.

(Those who know not what category codes are, should stop skipping. Something new will start soon). Two of the four goals (SGML etc. and friendly error recovery) are made possible by fixing document category codes to carefully chosen meanings. It is hard to see how else they could be realised.

Now, `tex` the program can be thought of as a typesetting engine. It turns text into paragraphs and pages. Just as a petrol engine could be used to power a car, or an aeroplane, or a lawnmower, so a `tex` could be used for batch typesetting or as the engine for a system similar to Quark Xpress.

By **visual typesetting** I mean interacting with a *graphic* representation of the document being created or processed. The display (or formatting) of the document should be adapted to the device being used to present the document. For example, on a computer screen, colour could be used to indicate emphasis and so forth, rather than shape and weight of font, which are more appropiate to printed representation. And of course the size and resolution of the computer screen (and the visual acuity of the user) are most relevant to making the best of what there is. WYSIWYG is a special case of *visual typesetting*.

The basic idea is that the document is a long galley, set paragraph by paragraph. When a change is made to the underlying text, the affected paragraphs should be reset, and the display refreshed. Please note that TEX will reset a paragraph in a fraction of the time required to update the display, particularly when run as a continuous process. Note also that this approach will put *sensible* restrictions on what the typist can do. For example, it is an error (inadmissable) to make a global change of font within a paragraph, for that would require resetting all subsequent paragraphs.

These ideas are further developed in my article *Editing* `.dvi` *files, or visual TEX*, which will appear in a future issue of *TUGboat*. Since the meeting my proposal for a Special Interest Technical Working Group on Visual TEX was approved by the Technical Council of TUG. If you would like information, or wish to join, please contact me, for I am the chair of this group.

Also since the meeting I found that the same basic underlying concept

> It is sometimes useful to maintain information about a source and a result document simultaneously in the same document, as in "what you see is what you get" (WYSIWYG) word processors. There, the user appears to interact with the formatted output, but the editorial changes are actually made in the source, which is then reformatted for display.

put forward to motivate the CONCUR feature provided by SGML. This quotation comes from Annex C.3.1 of ISO 8879 (the SGML standard) and is also reproduced (as is the whole of ISO 8879) on page 88 of Charles F. Goldfarb, The SGML Handbook, OUP (1990).

Now, the creation of format files to support these new demands presents new problems for the macro writer, not least of which is the very many active characters that will be required. (At least there will be no more than 256 active characters). Notice that macro files contain tokens, while under the new scheme text files contain characters. When

reading macros we wish to have access to special tokens. The solution is to enhance the programming language and compile to a special file format, which can then be loaded.

The basic idea is to provide the power that languages such as *C* take for granted. For example, one would like named parameters, like so,

```
\def \centerline #\text
{
   \line { \hss \text \hss }
}
```

and escape characters, so that

```
\def !  { ... }
```

will define a meaning for the active (!) space character. These ideas are further developed in articles which appear in TUGboat 13(4) **1992**, and Baskerville 3(1) **1993**.

# X Topical Tips — side by side figures in LaTeX

R. A. Bailey
Goldsmiths' College, University of London

**Question 1** How do I set two figures side by side in LaTeX?

**Answer** I assume that you mean how do you set a pair of figures something like Figures 1 and 2 below.

| Source | df |
|---|---|
| blocks | $b-1$ |
| varieties | $k-1$ |
| residual | $(b-1)(k-1)$ |
| total | $bk-1$ |

**Figure 1.** The analysis-of-variance table

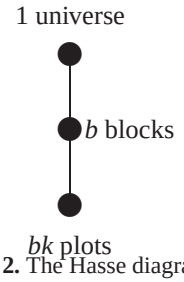



**Figure 2.** The Hasse diagram

That pair of figures consists of two objects in a single `figure` environment (more on that later). Here is how I did them.

```
\begin{figure}
  \noindent
  \makebox[\textwidth]{%
  \begin{minipage}[b]{0.5\textwidth}
  <first object>
  <first caption>
  \end{minipage}%
  \hfill
  \begin{minipage}[b]{0.4\textwidth}
  <second object>
  <second caption>
  \end{minipage}%
  }
  \end{figure}
```

(Well, what I've just said isn't totally true, but I'll come back to that.) The first object is an `array` inside displayed maths, while the second object is a `picture`. The objects can be anything you like, including tables and straight text. Figures 3 and 4 are done similarly.

In both cases I've put each object in a `minipage`: see pages 98–99 of *LaTeX: A Document Preparation System* by Leslie Lamport, hereafter called *The Manual*. The `minipage` takes an argument specifying its width. Rather than defining the widths of the objects absolutely, I prefer to define them in terms of `\textwidth`, which is described on page 94 of *The Manual*. I've chosen two fractions which seem to me in the right proportion and which add up to less than 1: in Figures 3 and 4 the proportions are 0.3 and 0.6.

Normally, `minipage`s are horizontally aligned around a central horizontal axis. I prefer to have the two `caption`s lined up nicely, so I have given both the `minipage`s the optional argument `b`, which aligns them at the bottom.

The two `minipage`s are put inside a box which is just as wide as the text. This is made with `\makebox` (see page 97 of *The Manual*), whose optional argument specifies the width of the box. There is a `\noindent` before it, to make sure that there is no space on the left before the box starts. The `%` sign immediately after the start of the box lets me start a new line in the input file without introducing a space before the first `minipage`: in other words, the first `minipage` is flush left to the edge of the text. In the same way, the `%` sign after each `\end{minipage}` prevents

a little inter-word space creeping in after each `minipage`. This fussiness is not evident in Figures 1 and 2, but does show up in Figures 3 and 4, which have visible outer edges flush with the edges of the surrounding text.

Finally, `\hfill` is put in between the two `minipage`s, forcing them as far apart as they will go: see page 96 of *The Manual*.

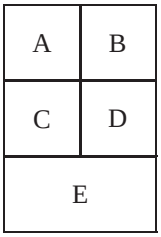



**Figure 3.** Five objects in one figure

Tomorrow, and tomorrow, and tomorrow, creeps in this petty pace from day to day, to the last syllable of recorded time; and all our yesterdays have lighted fools the way to dusty death.

**Figure 4.** Philosophy

I expect that you asked this question because of a misunderstanding about what a LaTeX `figure` is. It is not really a figure at all: it is what *The Manual* calls a *float*. That is, it is a chunk of text—or maths, or pictures, or whatever—that is treated as a whole and is printed in the first convenient place where there is room for it. It may contain any number of `caption`s (even none), each of which produces output beginning "Figure . . . ". I have used the word "object" for what you probably think of as a figure. You have to visually place the objects and captions within the `figure` so that the captions refer to the correct objects.

If you do not want to force the two objects to the edges of the text, you can omit the `makebox` and put explicit space between the two `minipage`s, something like

```
\begin{figure}
\centering
\begin{minipage}[b]{0.5\textwidth}
<first object>
<first caption>
\end{minipage}%
\hspace{1cm}
\begin{minipage}[b]{0.4\textwidth}
<second object>
<second caption>
\end{minipage}%
\end{figure}
```

You can also stack two `minipage`s on top of each other in a single `figure`. If their total width is more than `\textwidth` you do not even have to force them onto two separate lines. In this way you can glue five objects together in one figure in the arrangement shown in Figure 3.

Now to the lies. Since *Baskerville* is set in two columns, all the figures in this article have been done using `\columnwidth` instead of `\textwidth`. If you do the captions as I have suggested, then that for Figure 1 appears to come out a little lower than that for Figure 2. This is because the second `caption` contains the descending letter 'g' whereas 'variance table' contains no descenders. So the bottom of the 'g' is lined up with the baseline of the first caption. To cure this, we can put in the useful command `\strut`, which isn't in *The Manual*. It produces no visible output, and has height but no width. The principle is explained on page 100 of *The Manual*. The command `\strut` produces a strut whose size depends on the current font size. It is big enough to cover the descenders and ascenders on ordinary letters. Two lines of ordinary text that both contain `\strut` both take up the same amount of vertical space. So we finish the first and second captions with

```
table\strut}
```

and

```
diagram\strut}
```

being careful not to introduce an extra inter-word space before the strut.

Any statistician reading this will probably object that Figure 1 should be called a *table*. Here's the rub. I can put any number of `caption`s inside a `figure` and they will all come out "Figure . . . ". Similarly, I can put several

`caption`s inside a `table` and they will all come out "Table …". What I cannot do is mix "Table"s and "Figure"s in the same LaTeX float. In particular, I cannot put a "Table" beside a "Figure". Nor can the average LaTeX user. Let us hope that the LaTeX3 team will give us this flexibility.

# XI Malcolm's Gleanings

Malcolm Clark

## 1 Quotes

'LaTeX, developed at UCLA…' Seybold Report on Publishing Systems, volume 20, number 7, 1990.

'When I first started using TeX, I would have said that if you don't actually need it you should opt for one of the more friendly alternatives. With the benefit of a few hours experience behind me, however, I have to say that I'm becoming hooked. Don't exclude it on the grounds of perceived user unfriendliness, since you'll be throwing away the chance of first-class output." Computer Shopper, number 52, June 1992, pages 128 & 130.

## 2 Frame 0, TeX 1

A recent advertisement from Framemaker claimed that it 'runs on more platforms than any other publishing program'. This is manifestly incorrect and the was referred to the Advertising Standards Authority as an example of an advertisement which was not 'honest and truthful'. The ASA upheld the complaint, as outlined in their Monthly Report 12 for 1992, released on May 13th. A consequence of this decision will be that the offending advertisement will have to be withdrawn or re-written.

Because of TeX's public domain nature, we have no obvious way of combating the misinformation purveyed by commercial vendors of alternative publishing systems, and it depends upon the motivation of individuals to challenge such advertisements. That means you. If you can't be bothered yourself, forward any misleading information of this type to me, and I'll take it up. We've been nice guys for too long.

## 3 Book Review

*Books and Printing, A Treasury for Typophiles*, Paul A Bennett, editor, Frederic C Biel, Savannah, 1991 (reissue of 1951 edition), 417pp, ISBN 0-913720-72-0

This is a collection of short articles (from many sources) collected by Paul Bennett. It covers many aspects of printing and publishing, historical, aesthetic, discursive and bombastic. Some very well-known names are present: I picked the book up originally because it had Beatrice Warde's famous essay 'Printing should be invisible', where she elaborates her metaphor of drawing a parallel between wine in a fine crystal goblet and typography. She also coins the phrase 'stunt typographer' for those who amaze with 'vulgar ostentation'. As a colleague of Stanley Morison, it is natural then to skip to Morison's 'First principles of typography'. It is in this prescription for appropriate typography that Morison expounds his belief that the average line should contain between 10 and 12 words. But there is much more here, often stated rather than demonstrated, but usually with some appeal to a plausible explanation. The underlying theme however is little different to Warde's. Eric Gill's article on 'Typography' contains the recommendation that 'using Italics to emphasize single words should be abandoned in favour of the use of ordinary Lower-case with spaces between the letters (l e t t e r-s p a c e d)'. Gill, perhaps the last flowering of the Arts and Crafts movement, notes that '(it) is not that Industrialism has made things worse, but that it has made them different', and observation that might be applied to the new publishing of the 90s. Goudy too recounts some of the thinking and beliefs which he put into action in the design of his typefaces. Orcutt's 'The anatomy of a book' identifies the physical structure which is partially exemplified in SGML and LaTeX, but fills in some of the details of the whys. This article is used as the basis of a symposium which is given here too, discussing whether there had been any material changes in that structure.

What is missing from this book? It is a very American view of the world: perhaps one might be gracious and extend that to 'english speaking', but in truth the vast majority of writers are American. Virtually the only time that 'compte-morary' European typography makes an appearance is in Updike's round condemnation of the Bauhaus School, and especially its rejection of upper case. This illustrates number of points: typographers, as exemplified in this book, are conservative – this is often stated as a good thing by the writers, and their reasoning seems valid today. They are essentially book people, where the function of the typography is to convey meaning. Advertising typography is something else. But this conservatism spills over into what appears to be a total isolation from the political and social context.

Somehow they manage to realise the social impact of Gutenberg, but not of the Bauhaus' inherent political statement in removing the distinction between upper and lower case. (Yes, Gill is more aware of the social context, but he was English.) This is therefore a rather one-sided or even lop-sided view of the world. Recalling that most of the articles were first published when the USA was pursuing its policy of isolationism, this is perhaps not too surprising. I would still have loved to see but one article by (say) Tschichold.

This quibble aside, there is much to mine in here, from details of Shaw's relationship with his printers and publishers (following his strict instructions on word spacing, they hyphenated a-n and t-he: he relented), to supposed histories of the alphabet. Besides the chart from Dwiggens, a rather remarkable feature of the book is that each article is in a different typeface. Besides giving the rare opportunity to compare typefaces when many other factors are held constant, like page size, paper quality, inking and so on, it has the rather convenient feature of making it easier to spot when articles change, if you are flicking through looking for the start of a particular article.

## 4 A sidelight on 'TEX in Practice'

One of the events of this year was the appearance, finally, of Stephan von Bechtolsheim's monumental *TEX in Practice* volumes. An interesting reaction came in a Usenet posting by Professor David Rogers:

"As some of you have perhaps noted, I am the Editor of the *Monographs in Visual Communication Series* for Springer-Verlag which includes *TEX in Practice* by Stephen von Bechtolsheim. The forward in the volume is **not** what I wrote. It was modified by Stephen without my concurrence. The unmodified version is given below. I think the second paragraph is particularly interesting as I have noticed a significant dichotomy in the way different people approach TEX.

Further, I take **no** responsibility for the quality of the typesetting of the book nor for the quality of the English or the proofreading. I consider the book a prime example of a very poor design and typesetting job. The English is atrocious and the proofreading is nearly non-existent. Both the editorial and production departments at Springer-Verlag and I tried to get these defects corrected but with little success.

Having said that why did we publish the book? Basically because it contains very valuable information about the use of TEX. Information that the TEX community very much needs. After all, the fundamental purpose of a book is to convey information. So the decision was made to ignore the defects and publish it anyway.

I trust that you can ignore the presentation defects in the book and concentrate on the information."

*The original foreword*

> You might well wonder why *TEX in Practice* is a part of the Monographs in Visualization series. However, if you really think about typesetting, especially fine typesetting, you soon realize that, in large part, it is a *visual art* as well as a science. This is especially true for mathematical typesetting. As fine and robust as are the algorithms upon which TEX is based, they do not produce aesthetically perfect results. Visually one frequently wants a little more (or less) space before a subscript or superscript or a little less space above the denominator or below the numerator in a fraction or this page opening would look a little better if the line below the last equation was pushed to the next page, etc. Fine typesetting is a visual art form. Fortunately, Donald Knuth, in his wisdom, recognized this and provided TEX with unsurpassed capabilities for accomplishing these small visual adjustments so critical to fine typesetting.
>
> TEX itself can be considered from at least two significant and quite different viewpoints. The first is as a typesetting *system* in which the typesetter has precise control of the placement of characters and white space, the design and make-up of lines, equations, paragraphs, and pages. The second is as a macro-extensible *programming* language. Fundamentally, TEX in Practice addresses TEX from the latter viewpoint.
>
> The four volumes of Stephan v. Bechtolsheim's long awaited *TEX in Practice* present a comprehensive view of TEX. His thorough discussion of each aspect of TEX is liberally laced with cogent illustrative examples. Many of these examples represent complete, ready to use macros that enhance the capabilities of TEX. These examples are of particular interest to both the typesetter and the TEX programmer. The typesetter can often solve an immediate problem by either using one of the examples directly or by making minor changes to adapt it to the problem at hand. The TEX programmer can use the examples, along with Stephan's detailed discussion, to increase both the depth and breadth of his or her knowledge of TEX. The value of the text is further enhanced by the author's concerted effort to explain the reasoning behind each topic or example. In many cases, he details the inner workings of TEX's processing of the example.
>
> Stephan is to be congratulated on producing a work of fundamental and lasting value to the TEX and publishing community.

# XII Obtaining TEX

Sebastian Rahtz

## 1 From the network

The UK TEX Archive (Internet 'Daughter' archive) on `ftp.tex.ac.uk` is part of a collaborating network of archives organised by the TEX Users Group known as CTAN (Comprehensive TEX Archive Network). The three main archives now follow the same structure and have identical files (`ftp.tex.ac.uk`, `ftp.shsu.edu` and `ftp.uni-stuttgart.de`).

The preferred access method to the UK TEX Archive is using the *gopher* program which has a set of useful indexes to help you locate what you are looking for, but Internet *ftp* access is also very common. JANET users may only access the machine using the `ft-relay` site, as it has no X25 connection.

The CTAN archives all run an enhanced *ftp* server which supports dynamic compression, uncompression, and archive creation options. Fetch the top-level file `README.archive-features` for information. The server also supports site-defined commands to assist you. Please read `README.site-commands` for a brief overview.

The main directories which make up CTAN are listed below; readers are referred to David Jones' *Index of TEX Styles and Macros* for details of macro packages and individual style files. This can be found in CTAN as `info/tex-styles-and-macros.txt`

**biblio** contains bibliography-related files, such as BIBTEX.

**digests** contains back issues of TEX-related periodicals. various aspect of TEX.

**dviware** contains the various dvi-to-whatever filters and drivers.

**fonts** contains a collection of fonts, both sources and pre-compiled.

**graphics** contains utilities and macros related to graphics.

**help** contains files which provide an overview to the archive and the TEX system.

**info** contains files and tutorials which document

**indexing** contains utilities and related files for indexing documents.

**language** contains non-English related implementations of TEX.

**macros** contains macros for TEX and its derivatives in unique subdirectories.

**support** contains files and programs which can be used in support of TEX.

**systems** contains complete system setups, organized by operating system.

**tools** contains the various archiving tools which users may find useful.

**web** contains WEB-related files and utilities.

Details of where to find public domain, or shareware, TEX packages for different machines and operating systems are given in Figure 1. Some readers may prefer to purchase a commercial package, with support. *Neither CTAN not UKTUG are in a position to offer support for software in the archive*.

Please report any problems with CTAN archives via e-mail to `ctan-mgr@shsu.edu`. The entire archive will be available on CDROM from early 1994; details will be given in the next issue of *Baskerville*.

## 2 Unix tapes

For a complete Unix TEX distribution, a 1/4 inch cartridge, QIC-120 or QIC-150 format (DC600A or DC6150) can sent with envelope *and* stamps for return postage to:

David Osborne
Cripps Computing Centre,
University of Nottingham,
Nottingham NG7 2RD

Due to currency exchange, this service is offered only within the UK.

| Name | Environment | CTAN path | Notes |
|---|---|---|---|
| emTEX | DOS, OS/2 | systems/msdos/emtex | the betatest drivers, and 386-specific versions of TEX and MF, are in the `betatest` subdirectory |
| sbTEX | DOS | systems/msdos/sbtex | includes latest TEX and METAFONT |
| gTEX | DOS | systems/msdos/gtex | 386-only TEX and METAFONT, which replace parts of emTEX, and work with Windows memory management |
| TEXas | DOS | systems/msdos/texas | Large TEX, which replaces TEX386 in emTEX, and works with Windows memory management |
| OzTEX | Mac | systems/mac/oztex | This package is now shareware; there is a also multi-lingual version (Euro-OzTEX) |
| CMacTEX | Mac | systems/mac/cmactex | Port of Unix TEX, including Rokicki's dvips. A fuller version (with large memory) can be purchased. |
| DirectTEX | Mac | systems/mac/directtex | |
| pasTEX | Amiga | systems/amiga/pastex | |
| — | Atari | systems/atari/lindner-tex-packed-disks | |
| — | Atari | systems/atari/cs-tex | |
| web2c TEX | Unix | systems/unix/web2c | a complete source kit for TEX and METAFONT, which should compile on most Unix boxes; needs a C compiler. |
| sparctex | Sun Sparc | systems/unix/unixkit | |
| — | DEC Alpha | systems/unix/alpha | for Alpha running Unix |
| — | Xenix | systems/unix/xenix | |
| Decus TEX | VAX/VMS | systems/vms/decus | |
| — | Alpha VMS | systems/vms/alpha | for Alpha running VMS |

**Figure 1.** CTAN location of TEX systems for different machines

## 3 PC and Mac disks

From January 1994 the UKTUG will start distributing an emTEX kit for PCs, and an OzTEX kit for Macintosh, for a small fee to cover copying and postage costs, and the shareware fee for OzTEX. Details will be given in the next issue of *Baskerville*.

Enquiries for TEX for the Atari ST etc. can be directed to: The South West Software Library, P.O. Box 562, Wimborne, Dorset BH21 2YD (JANET e-mail address: `mdryden@uk.co.compulink.cix`)

The international TEX Users Group can also supply many TEX materials on disk. Contact:

TEX Users Group
PO Box 869
Santa Barbara, CA 93102
USA
*Phone:* 805-899-4673 *E-mail*: `tug@tug.org`

## XIII UKTUG Programme of meetings for 1994

Malcolm Clark
Meetings Secretary

- 19th January, 1994: *Choosing and Using PostScript fonts with TEX* A practical meeting addressing the problems of using PostScript fonts in TEX and LATEX documents, to be held at Rewley House in Oxford. Speakers will include Sebastian Rahtz, Alan Jeffrey and Will Shaman. There will be a panel session to address wider issues. *Local organizer*: Ian Hall
- 21–22 March, 1994: *LATEX 2ε* Continuing the tradition established by the successful two day meeting at RHBNC last year, this meeting will concentrate on this recently released new standard for LATEX. Speakers will be from the LATEX3 team (some from across the water – no expense spared!). We plan to cover how to: install it; integrate local styles; use scaleable fonts; integrate graphics; process existing documents; use the new features; write extension packages, and much much more. This meeting will be held at Warwick University, in the heart of England. The cost will be in the region of £80–90 (to include one night's accommodation). *Local organizer*: Malcolm Clark; *Programme organizer*: Chris Rowley
- 2nd week of July, 1994: *A training meeting, on Font Selection Scheme 2 (N/FSS2)* To be held at Cambridge University. Numbers will be limited. *Local organizers*: Robin Fairbairns & Jonathan Fine
- 19 October, 1994: *Annual General Meeting* To be held at Aston University, Birmingham. Your annual chance to make suggestions for future meetings, elect committee members, and to raise anything else you wish to about the workings and activities of the Group.
- November/December, 1994: *TEX, SGML & electronic publishing* Exact date, speakers and location unconfirmed (at least we've got a title!).

All members of the UK TEX Users Group will be provided with more details of these meetings as they become available. Reports of the meetings will appear in *Baskerville*, the organ of the group.