Editorial

The Guest Editor of the last issue of *Baskerville*, James Foster, explained in that issue how members of the UK-\TeX{} Committee have assumed editorial responsibility for the preparation and formatting of individual numbers of the newsletter. Like James, I am deeply grateful for, and awed by, the amount of work and expertise which Sebastian Rahtz has put into past issues of *Baskerville*. Thanks, Sebastian!

James also mentioned the hard work which Robin Fairbairns has done over the years in producing and distributing *Baskerville*. Although Robin is now liberated from these particular tasks, he is still heavily involved in supporting the \TeX{} community in various ways. In particular, he is the current custodian and mastermind behind the invaluable \TeX{} FAQ. This he maintains as a multi-format document which includes an incarnation on the world-wide web (see question 20 below).

The \TeX{} FAQ is a very special document, since it tackles head-on many of the most urgent and practical issues about which people need to know when they come into contact with \TeX{}. The actual questions which are answered in this FAQ are those which have appeared most often in the `comp.text.tex' discussion group, plus others which surface repeatedly in various fora. Although Robin is the current maintainer of this FAQ, many people have contributed to it, as is explained in the introduction below.

The \TeX{} FAQ has been published in *Baskerville* twice before, in 1994 and 1995. These are the issues of *Baskerville* which I have most often lent or recommended to other \TeX{} users. In fact, I currently do not have the 1995 FAQ issue because I gave it away to someone who needed it as a matter of urgency! I am confident that this newly updated \TeX{} FAQ, now expanded to cover 126 questions, will be every bit as popular and useful as its predecessors, and will save \TeX{} users many hours of valuable time.

Don’t overlook Gareth Suggett’s review on p. 37. If you write lots of equations, this software could be for you! And finally, I wonder how many *Baskerville* readers remember the bits and pieces recalled in Historical Snippets (p. 38)? If you have never heard of any of this stuff before, then you are just the kind of vibrant, new, upcoming UK-\TeX{} member that we want, infinitely more interested in publishing your work than in fiddling with \TeX{}. But \TeX{} wasn’t built in a day, and perhaps these Snippets give a flavour of some of the evolutionary forces which have made \TeX{} and friends what they are today.

Stop Press: UK \TeX{} Users’ Group Tenth Anniversary Meeting: Knuth to host Q&A session

The Chairman and Committee of UK-\TeX{}G request the pleasure of your company at the Tenth Anniversary Meeting of the UK \TeX{} Users’ Group, to be held at St Anne’s College, Oxford, on Sunday 12th and Monday 13th September 1999.

The meeting will commence at 11:30 on Sunday 12th September with the first of a series of talks on the theme “\TeX{} and its relationship to the \{SG/HT/X\}ML family of languages”. After lunch on Sunday there will be further talks on this topic, following which there will be the 1999 Annual General Meeting of the UK \TeX{} Users’ Group.

On Sunday evening, there will be a special Anniversary Banquet, at which Professor Donald Knuth and his wife Jill will be our guests of honour. After the meal, Don has kindly agreed to give a spontaneous question-and-answer session, and members of UK-\TeX{}G will be most welcome to ask Don questions on any topic which is of mutual interest.

Monday 13th will continue the theme of “\TeX{} and its relationship to the \{SG/HT/X\}ML family of languages” and may include a tutorial on XSL.

Accommodation in St Anne’s will be available for those not wishing to return home between sessions.

A more detailed announcement and booking form accompany this issue of *Baskerville*.

---

1The continuation to Monday is dependent on a sufficient number of knowledgeable speakers being found; final details should be known by the time this issue of *Baskerville* is distributed.
## Contents

### The New \TeX\ FAQ

<table>
<thead>
<tr>
<th>Section</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>10</td>
</tr>
<tr>
<td>E</td>
<td>11</td>
</tr>
<tr>
<td>F</td>
<td>12</td>
</tr>
<tr>
<td>G</td>
<td>16</td>
</tr>
</tbody>
</table>

### A Introduction

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

### B The background

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

#### 1. What is \TeX? | 4
#### 2. How should I pronounce "\TeX"? | 4
#### 3. What is \METAFONT? | 4
#### 4. What is MetaPost? | 4
#### 5. How can I be sure it's really \TeX? | 4
#### 6. Are \TeX\ and friends Y2K compliant? | 5
#### 7. What is \LATEX? | 5
#### 8. What is \TEXinfo? | 5
#### 9. What is Loliplop? | 6
#### 10. What is \ams? | 6
#### 11. What is Explain? | 6
#### 12. What is \molliplop\? | 6
#### 13. What is \molliplop\? | 6
#### 14. If \TeX\ is so good, how come it's free? | 6

### C Documentation and help

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
</tr>
</tbody>
</table>

#### 19. Books on \TeX\ and its relations | 7
#### 20. Where to find this article | 8
#### 21. Mailing lists about \TeX\ and its friends | 8
#### 22. \iftex\ Tutorials | 9
#### 23. \iftex\ Documentation | 9
#### 24. The \iftex\ manual | 9
#### 25. Finding \iftex\ macro packages | 9
#### 26. Finding files in the CTAN archives | 9

### D Bits and pieces of \TeX\

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
</tr>
</tbody>
</table>

#### 27. What is a DVI file? | 10
#### 28. What is a driver? | 10
#### 29. What are PK files? | 10
#### 30. What are TFM files? | 10
#### 31. Virtual fonts | 10
#### 32. \special\ commands | 11
#### 33. Documented \iftex\ sources (.dtx files) | 11
#### 34. What are the EC fonts? | 11

### E Acquiring the software

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
</tr>
</tbody>
</table>

#### 35. Repositories of \TeX\ material | 11
#### 36. Contributing a file to the archives | 12
#### 37. Finding new fonts | 12
#### 38. \TeX\ CD-ROMs | 12

### F \TeX\ systems

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
</tr>
</tbody>
</table>

#### 39. \iftex\ for different machines | 12
#### 40. \iftex\-friendly editors and shells | 13
#### 41. Commercial \TeX\ implementations | 14

### G DVI drivers and previewers

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

#### 42. DVI to PostScript conversion programs | 16
#### 43. DVI drivers for HP LaserJet | 16
#### 44. DVI previewers | 16

### H Support packages for \TeX\

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

#### 45. Fig, a \iftex\-friendly drawing package | 16
#### 46. \iftex\, a drawing package for \iftex\ | 16
#### 47. Spelling checkers for work with \iftex\ | 16
#### 48. The \iftex\ package | 17

### I Literate programming

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
</tr>
</tbody>
</table>

#### 50. Web for C, FORTRAN, and other languages | 17

### J Format conversions

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
</tr>
</tbody>
</table>

#### 51. Conversion between \iftex\ and others | 17
#### 52. Conversion from \iftex\ to plain ASCII | 18
#### 53. Conversion from SGML or HTML to \iftex\ | 18
#### 54. \iftex\ conversion to HTML | 18
#### 55. Making hypertext documents from \iftex\ | 19
#### 56. Making Acrobat documents from \iftex\ | 19

### K \METAFONT

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
</tr>
</tbody>
</table>

#### 57. Getting \METAFONT to do what you want | 19
#### 58. Which font files should be kept | 20
#### 59. Getting bitmaps from the archives | 20

### L PostScript and \TeX\

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
</tr>
</tbody>
</table>

#### 60. Using PostScript fonts with \iftex\ | 21
#### 61. Previewing files using PostScript fonts | 21
#### 62. \iftex\ font metric files for PostScript fonts | 21
#### 63. Problems using PostScript fonts | 22
#### 64. Choice of scalable outline fonts | 22
#### 65. Including a PostScript figure in \iftex\ | 23

### M Special sorts of typesetting

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
</tr>
</tbody>
</table>

#### 66. Drawing with \iftex\ | 23
#### 67. Double-spaced documents in \iftex\ | 24
#### 68. Formatting a thesis in \iftex\ | 24
#### 69. Flowing text around figures in \iftex\ | 24
#### 70. Alternative head- and footlines in \iftex\ | 25
#### 71. Including a file in \iftex\ | 25
#### 72. Including line numbers in \iftex\ typeset output | 25
#### 73. Generating an index in \iftex\ | 25
#### 74. Typesetting URLs | 25
#### 75. Citing URLs with \iftex\ | 26
#### 76. Using \iftex\ with plain \iftex\ | 26
#### 77. Typesetting music in \iftex\ | 26
#### 78. Drawing Feynman diagrams in \iftex\ | 26

### N How do I do X in \iftex\ or \iftex?\?

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
</tr>
</tbody>
</table>

#### 79. Proof environment | 26
#### 80. Symbols for the number sets | 27
#### 81. Roman theorems | 27
#### 82. Fancy enumeration lists | 27
#### 83. Unnumbered sections in the Table of Contents | 27
#### 84. Footnotes in tables | 27
#### 85. Style of section headings | 28
#### 86. Indent after section headings | 28
#### 87. Footnotes in \iftex\ section headings | 28
#### 88. Changing the margins in \iftex\ | 28
#### 89. Finding the width of a letter, word, or phrase | 29
The New \TeX FAQ: Your 126 Questions Answered

A Introduction

This article was prepared by Robin Fairbairns, assisted in very small part by the Committee of the UK \TeX Users’ Group (UK-T\TeX G)\(^2\) as a development of a regular posting to the Usenet newsgroup comp.text.tex that was maintained for some time by Bobby Bodenheimer (bobby@hot. caltech.edu).

Usenet is a mechanism for exchanging articles between people who share interests or needs\(^3\); a newsgroup is an area within Usenet carrying a particular class of articles. Since a common sort of article asks for help, advice or information, and since certain of these questions are regularly repeated (often with monotonous regularity), some public-spirited souls took to writing articles which listed “Frequently Asked Questions” and answers to them. Some members of UK-T\TeX G may not have access to Usenet, but could be expected to value the article freely available (usergrps/uktug/faq), and it can be compiled by anyone who runs reasonably current \LaTeX\Xe (see question 121), and has the required fonts. It was the committee’s original intention that it would also be possible for the content of this article to feed back to the world-wide \TeX community via regular posting to Usenet, but since Bobby was forced to abandon his work in the area, no-one has been able to take his place.

Therefore, a translation of the article has been made available on the World-Wide Web, via URL http://www.tex.ac.uk/cgi-bin/txfaq2html?introduction=yes

A hypertext version is also to be found on the \TeX Live CD-ROM (see question 38).

When we started, we rearranged Bobby’s original, and we have since added new questions and answers on the basis of our experience of answering questions about \TeX, writing documents in \TeX, and developing macros for \TeX, over the years.

The committee is grateful for help and advice, from the following outside its number: Donald Arseneau, Barbara Beeton, Karl Berry, Damian Cugley, Michael Downes, John Hobby, Berthold Horn, Werner Icking, David Kastrup, Ted Nieland, Pat Rau, Piet van Oostrum, Oren Patashnik, Joachim Schrod, Philip Taylor, Ulrik Vieth, Rick Zacccone and Reinhard Zierek.

Further, former UK-T\TeX G committee members Rosemary Bailey, Chris Rowley, Alan Jeffrey, Carol Hewlett, David Carlisle, Robin Fairbairns, Jonathan Fine, and Sebastian Rahtz all made significant contributions to the conception,


\(^3\)Usenet, as its name implies, is a means of using some sort of network; in the earliest days the network was made by stringing together a series of telephone lines, but nowadays Usenet is most often carried over the Internet.
development and subsequent revision of this FAQ during their terms of office, and we are grateful to them for their contributions.

Finding the Files

Unless otherwise specified, all files mentioned in this article are available from a CTAN archive, or from one of their mirrors. Question 35 gives details of the CTAN archives, and how to retrieve files from them. If you don’t have access to the Internet, question 38 tells you of sources of CD-ROMs that offer snapshots of the archives.

The reader should also note that the first directory name of the path name of every file on CTAN has been elided from what follows, for the simple reason that it’s always the same (tex-archive/).

To avoid confusion, we’ve also elided the full stop1 from the end of any sentence whose last item is a path name (note that such sentences occur only at the end of paragraphs). Though the path names are set in a different font from running text, it’s not easy to distinguish the font of a single dot!

B The background

1 What is \TeX?

\TeX{} is a typesetting system written by Donald E. Knuth, who says in the Preface to his book on \TeX{} (see question 19) that it is “intended for the creation of beautiful books — and especially for books that contain a lot of mathematics”.

Knuth developed a system of ‘literate programming’ to write \TeX{}, and he provides the literate (WEB) source of \TeX{} free of charge, together with tools for processing the web source into something that can be compiled and something that can be printed; there’s never any mystery about what \TeX{} does. Furthermore, the WEB system provides mechanisms to port \TeX{} to new operating systems and computers; in order that one may have some confidence in the ports, Knuth supplied a test by means of which one may judge the fidelity of a \TeX{} system. \TeX{} and its documents are therefore highly portable.

\TeX{} is a macro processor, and offers its users a powerful programming capability. For this reason, \TeX{} on its own is a pretty difficult beast to deal with, so Knuth provided a package of macros for use with \TeX{} called \texttt{plain \TeX{}}. \texttt{Plain \TeX{}} is effectively the minimum set of macros one can usefully employ with \TeX{}, together with some demonstration versions of higher-level commands (the latter are better regarded as models than used as-is). When people say they’re “programming in \TeX{}”, they usually mean they’re programming in \texttt{plain \TeX{}}.

2 How should I pronounce “\TeX{}”? The ‘X’ stands for the Greek letter Chi (\(\chi\)), and is pronounced by English-speakers either a bit like the ‘ch’ in ‘loch’ ([\(\chi\)] in the IPA) or like ‘k’. It definitely is not pronounced ‘ks’.

3 What is METAFONT?

METAFONT was written by Knuth as a companion to \TeX{}; whereas \TeX{} defines the layout of glyphs on a page, METAFONT defines the shapes of the glyphs and the relations between them. METAFONT details the sizes of glyphs, for \TeX{}’s benefit, and details the rasters used to represent the glyphs, for the benefit of programs that will produce printed output as post processes after a run of \TeX{}.

METAFONT’s language for defining fonts permits the expression of several classes of things: first (of course), the simple geometry of the glyphs; second, the properties of the print engine for which the output is intended; and third, ‘meta’-information which can distinguish different design sizes of the same font, or the difference between two fonts that belong to the same (or related) families.

Knuth (and others) have designed a fair range of fonts using METAFONT, but font design using METAFONT is much more of a minority skill than is \TeX{} macro-writing. The complete \TeX{}-user nevertheless needs to be aware of METAFONT, and to be able to run METAFONT to generate personal copies of new fonts.

4 What is MetaPost?

The MetaPost system (by John Hobby) implements a picture-drawing language very much like that of METAFONT except that it outputs PostScript commands instead of run-length-encoded bitmaps. MetaPost is a powerful language for producing figures for documents to be printed on PostScript printers. It provides access to all the features of PostScript and it includes facilities for integrating text and graphics. (Knuth tells us that he uses nothing else for diagrams in text that he is writing.)

Much of MetaPost’s source code was copied from METAFONT’s sources with Knuth’s permission.

5 How can I be sure it’s really \TeX{}?

\TeX{} (and METAFONT and MetaPost) are written in a ‘literate’ programming language called Web (see question 49) which is designed to be portable across a wide range of computer systems. How, then, is a new version of \TeX{} checked?

Of course, any sensible software implementor will have his own suite of tests to check that his software runs: those who port \TeX{} and its friends to other platforms do indeed perform such tests.

Knuth, however, provides a ‘conformance test’ for both \TeX{} (\texttt{trip}) and METAFONT (\texttt{trap}). He characterises these as ‘torture tests’: they are designed not to check the obvious things that ordinary typeset documents, or font designs, will exercise, but rather to explore small alleyways off the main path through the code of \TeX{}.

They are, to the casual reader, pretty incomprehensible!

Once an implementation of \TeX{} has passed its \texttt{trip} test, or an implementation of METAFONT has passed its \texttt{trap} test, it may reasonably be distributed as a working version.

---

1 Full stop (British English) = ‘period’ (American English)
6 Are \TeX{} and friends Y2K compliant?

**Crashing:** None of \TeX{}, \MetaFont{} or MetaPost can themselves crash due to any change whatever in the date of any sort.

**Timestamps:** As Knuth delivers the sources, a 2-digit year is stored as the creation time for format files and that value is printed in logfiles. These items should not be of general concern, since the only use of the date format file is to produce the log output, and the log file is designed for human readers only.

Knuth’s distributed source does not designate the code which generates this 2-digit date as a valid area where implementations may differ. However, he announced in 1998 that implementors can alter this code without fear of being accused of non-compliance. Nearly all implementations that are being actively maintained had been modified to generate 4-digit years in the format file and the log, by the end of 1998.

The \texttt{\year} primitive: Certification of a \TeX{} implementation (see question 5) does not require that \texttt{\year} return a meaningful value (which means that \TeX{} can, in principle, be implemented on platforms that don’t make the value of the clock available to user programs). The \TeX{}book (see question 19) defines \texttt{\year} as “the current year of our Lord”, which is the only correct meaning for \texttt{\year} for those implementations which can supply a meaningful value, which is to say nearly all of them.

In short, \TeX{} implementations should provide a value in \texttt{\year} giving the 4-digit year Anno Domini, or the value 1776 if the platform does not support a date function.

Note that if the system itself fails to deliver a correct date to \TeX{}, then \texttt{\year} will of course return an incorrect value. \TeX{} cannot be considered Y2K compliant, in this sense, on a system that is not itself Y2K compliant.

**Macros:** \TeX{} macros can in principle perform calculations on the basis of the value of \texttt{\year}. The \texttt{\LaTeX{}} suite (see question 7) performs such calculations in a small number of places; the calculations performed in the current (supported) version of \texttt{\LaTeX{}} are known to be Y2K compliant.

Other macros and macro packages should be individually checked.

**External software:** Software such as DVI translators needs to be individually checked.

7 What is \LaTeX{}?

\LaTeX{} is a \TeX{} macro package, originally written by Leslie Lamport, that provides a document processing system. \LaTeX{} allows markup to describe the structure of a document, so that the user need not think about presentation. By using document classes and add-on packages, the same document can be produced in a variety of different layouts.

Lamport says that \LaTeX{} “represents a balance between functionality and ease of use”. This shows itself as a continual conflict that leads to the need for such as the present article: \LaTeX{} can meet most user requirements, but finding out how is often tricky.

8 How should I pronounce “\LaTeX{}(2\epsilon)”?

Lamport never recommended how one should pronounce \LaTeX{}, but a lot of people pronounce it ‘Lay \TeX{}’ or perhaps ‘Lah \TeX{}’ (with \TeX{} pronounced as the program itself; see question 2).

The ‘epsilon’ in ‘\LaTeX{} 2\epsilon’ is supposed to be suggestive of a small improvement over the old \LaTeX{} 2.09. Nevertheless, most people pronounce the name as ‘\LaTeX{}-two-ee’.

9 Should I use plain \TeX{} or \LaTeX{}?

There’s no straightforward answer to this question. Many people swear by plain \TeX{}, and produce highly respectable documents using it (Knuth is an example of this, of course). But equally, many people are happy to let someone else take the design decisions for them, accepting a small loss of flexibility in exchange for a saving of brain power.

The arguments around this topic can provoke huge amounts of noise and heat, without offering much by way of light; your best bet is to find out what those around you are using, and to go with the crowd. Later on, you can always switch your allegiance; don’t bother about it.

If you are preparing a manuscript for a publisher or journal, ask them what markup they want before you develop your own; many big publishers have developed their own \LaTeX{} styles for journals and books, and insist that authors stick closely to their markup.

10 What are the AMS packages (\AMSTeX{}, etc.)?

\AMSTeX{} is a \TeX{} macro package, originally written by Michael Spivak for the American Mathematical Society (AMS) during 1983–1985. It is described in “The Joy of \TeX{}” by Michael D. Spivak (second edition, AMS, 1990, ISBN 0-8218-097-1). It is based on plain \TeX{}, but provides many features for producing more professional-looking maths formulas with less burden on authors. It pays attention to the finer details of sizing and positioning that mathematical publishers care about. The aspects covered include multi-line displayed equations, equation numbering, ellipsis dots, matrices, double accents, multi-line subscripts, and syntax checking (faster processing on initial error-checking \TeX{} runs), and other things.

As \LaTeX{} increased in popularity, authors asked to submit papers to the AMS in \LaTeX{}, and so the AMS developed \AMSTeX{}, which is a collection of \LaTeX{} packages and classes that offer authors most of the functionality of \AMSTeX{}.
11 What is Eplain?
The Eplain macro package expands on and extends the definitions in plain \TeX. Eplain is not intended to provide “generic typesetting capabilities”, as do \EPlain or Texinfo (see question 13). Instead, it provides definitions that are intended to be useful regardless of the high-level commands that you use when you actually prepare your manuscript.

For example, Eplain does not have a command \section, which would format section headings in an “appropriate” way, as \EPlain’s \section. The philosophy of Eplain is that some people will always need or want to go beyond the macro designer’s idea of “appropriate”. Such canned macros are fine — as long as you are willing to accept the resulting output. If you don’t like the results, or if you are trying to match a different format, you are out of luck.

On the other hand, almost everyone would like capabilities such as cross-referencing by labels, so that you don’t have to put actual page numbers in the manuscript. Karl Berry, the author of Eplain, says he is not aware of any generally available macro packages that do not force their typographic style on an author, and yet provide such capabilities.

12 What is Lollipop?
Lollipop is a macro package written by Victor Eijkhout; it was used in the production of his book “\TeX by Topic” (see question 19). The manual says of it:

Lollipop is ‘\TeX made easy’. Lollipop is a macro package that functions as a toolbox for writing \TeX macros. It was my intention to make macro writing so easy that implementing a fully new layout in \TeX would become a matter of less than an hour for an average document, and that it would be a task that could be accomplished by someone with only a very basic training in \TeX programming.

Lollipop is an attempt to make structured text formatting available for environments where previously only \TeX macros could be used because adapting the layout is so much more easy with them than with traditional \TeX macro packages.

The manual goes on to talk of ambitions to “capture some of the \EPlain market share”; it’s a very witty package, but little sign of it taking over from \EPlain is detectable… An article about Lollipop appeared in \TUGboat 13(3).

13 What is Texinfo?
Texinfo is a documentation system that uses one source file to produce both on-line information and printed output. So instead of writing two different documents, one for the on-line help and the other for a typeset manual, you need write only one document source file. When the work is revised, you need only revise one document. You can read the on-line information, known as an “Info file”, with an Info documentation-reading program. By convention, Texinfo source file names end with a .\texti or .\texinfo extension. You can write and format Texinfo files into Info files within GNU emacs, and read them using the emacs Info reader. If you do not have emacs, you can format Texinfo files into Info files using makeinfo and read them using info.

The Texinfo distribution, including a set of \TeX macros for formatting Texinfo files, is available as macros/texinfo.tar.gz

14 If \TeX is so good, how come it’s free?
It’s free because Knuth chose to make it so. He is nevertheless apparently happy that others should earn money by selling \TeX-based services and products. While several valuable \TeX-related tools and packages are offered subject to restrictions imposed by the GNU General Public Licence (‘Copyleft’), \TeX itself is not subject to Copyleft.

There are commercial versions of \TeX available; for some users, it’s reassuring to have paid support. What is more, some of the commercial implementations have features that are not available in free versions. (The reverse is also true: some free implementations have features not available commercially.)

Usually, this article does not describe commercial versions; Question 41 lists the major vendors.

15 What is the future of \TeX?
Knuth has declared that he will do no further development of \TeX; he will continue to fix any bugs that are reported to him (though bugs are rare). This decision was made soon after \TeX version 3.0 was released; at each bug-fix release the version number acquires one more digit, so that it tends to the limit $\pi$ (at the time of writing, Knuth’s latest release is version 3.14159). Knuth wants \TeX to be frozen at version $\pi$ when he dies; thereafter, no further changes may be made to Knuth’s source. (A similar rule is applied to METAFONT; its version number tends to the limit $e$, and currently stands at 2.718.)

There are projects (some of them long-term projects: see, for example, question 122) to build substantial new macro packages based on \TeX. For the even longer term, there are various projects to build a successor to \TeX; see questions 123 and 124.

16 What are \TUG and \TUGboat?
\TUG is the \TeX Users Group. \TUGboat is \TUG’s main journal, containing useful articles about \TeX and METAFONT. \TUG also organises a yearly conference, runs training courses, sells almost all \TeX-related books, and distributes \TeX-related microcomputer software on disk. \TUG has a Technical Council to coordinate \TeX-related developments (see question 18). Enquiries should be directed to:

\TeX Users Group
PO Box 2311
Portland, OR 97208-2311
USA
fax: +1-503-223-3960
voice: +1-503-223-9994
Email: tug@mail.tug.org
17 Are there nationally-based user groups, too?

The following groups publish their membership (etc.) information electronically on CTAN archives:

DANTE, Deutschsprachige Anwendervereinigung
\TeX\ e.V.
Postfach 101840
D-69008 Heidelberg
Germany
Tel: (+49) 06221 297 66
Fax: (+49) 06221 167906
Email: dante@dante.de
Web: http://www.dante.de/
CTAN details: usergrps/dante

Association GUTenberg,
BP 10,
93220 Gagny principal,
France
Email: gut@irisa.fr
Web: http://www.ens.fr/gut/
CTAN details: usergrps/gut

NTG
Postbus 394, 1740AJ Schagen,
The Netherlands
Email: ntg@nic.surfnet.nl
Web: http://www.ntg.nl/
CTAN details: usergrps/ntg

UK \TeX\ Users’ Group,
\%Peter Abbott,
1 Eymore Close,
Selly Oak,
Birmingham B29 4LB
UK
Tel: (+44) 0121 476 2159
Email: UKTuG-Enquiries@tex.ac.uk
Web: http://uk.tug.org/
CTAN details: usergrps/uktug

A listing of all known groups is available as usergrps/info/usergrps.tex

18 TUG Technical Working Groups

TUG (see question 16) has an autonomous Technical Council which can appoint Technical Working Groups related to the specific areas of interest within the \TeX\ community. Each group has a chair/contact person and establishes its own working methods and membership; anyone interested in taking part should contact the group’s chair person. Suggestions for new groups should be addressed to tech-council@tug.org; a list of members of the Technical Council can be found at http://tug.org/committees.html

Below is a brief list of currently active groups:

WG-92-01 \TeX\ Extended Mathematics Font Encoding.
To create font encoding standards for Mathematical fonts used in \TeX\ systems.
Contact: Barbara Beeton (bnb@math.ams.org)

WG-92-05 \TeX\ Archive Guidelines.
To develop guidelines for the effective management and utilisation of major \TeX\ archives, and to initiate communication among the maintainers of the existing archives for the purpose of coordination and synchronisation.
Contact: Sebastian Rahtz (s.rahtz@elsevier.co.uk)

WG-94-07 \TeX\ Directory Structure.
To identify a universal directory structure for macros, fonts and other related \TeX\ software so that recommendations can be made to all suppliers of \TeX\ software.

The group’s current set of proposals are to be found on CTAN at tds/draft-standard
Contact: Karl Berry (kb@cs.umb.edu)

WG-94-08 DVI Driver Implementation and Standardisation Issues.
To study the issues in the requirements of DVI Drivers imposed by changing needs and technologies, and to make recommendations for implementation and standardisation of such drivers to enhance the uniformity of their use. Work will include, but not be limited to, the examination of the use, syntax, and semantics of \special{} commands.

Contact: Michael Sofka (sofkam@rpi.edu)

WG-94-10 \TeX\ and Linguistics.
To study and discuss the requirements for typesetting linguistics in \TeX\ and as a means of identifying, examining, testing, and comparing macros, fonts, style files and other aids for typesetting linguistics.

Contact: Christina Thiele (cthiele@ccs.carleton.ca)

C Documentation and help

19 Books on \TeX\ and its relations

While Knuth’s book is the definitive reference for \TeX, there are other books covering \TeX:\


Introduction to \TeX\ by Norbert Schwarz (Addison-Wesley, 1989, ISBN 0-201-51141-X)

A Plain \TeX\ Primer by Malcolm Clark (Oxford University Press, 1993, ISBNs 0-198-53724-7 (hardback) and 0-198-53784-0 (paperback))
**The New \TeX\ FAQ: Your 126 Questions Answered**

---

**\TeX\ by Topic** by Victor Eijkhout (Addison-Wesley, 1992, ISBN 0-201-56882-9)

**\TeX\ for the Beginner** by Wynter Snow (Addison-Wesley, 1992, ISBN 0-201-54799-6)


**\TeX\: Starting from \textit{Square One}** by Michael Doob (Springer Verlag, 1993, ISBN 3-540-56441-1)


A collection of Knuth’s publications about typography has recently been published:


For \LaTeX\, see:


**The \LaTeX\ Companion** by Michel Goossens, Frank Mittelbach, and Alexander Samarin (Addison-Wesley, 1993, ISBN 0-201-54199-8)

**The \LaTeX\ Graphics Companion: Illustrating documents with \TeX\ and \PostScript** by Michel Goossens, Sebastian Rahtz and Frank Mittelbach (Addison-Wesley, 1997, ISBN 0-201-35469-4)

**The \LaTeX\ Web Companion Integrating \TeX, HTML and XML** by Michel Goossens and Sebastian Rahtz (Addison-Wesley, 1999, ISBN 0-201-43311-7)


**Math into \LaTeX\: An Introduction to \LaTeX\ and \AMSTeX** by George Grätzer (Birkhäuser, 1996, ISBN 0-817-63805-9)

---

**First steps in \ET\X\ by George Grätzer (Birkhäuser, 1999 [to appear in July 1999], ISBN 0-8176-4132-7**


A sample of George Grätzer’s book, in Adobe Acrobat format, is also available ([info/mil/mil.pdf](http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes)).

Example files for the \ET\Xh\ Graphics and Web Companions are available in [info/lgc](http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes) (Graphics) and [info/lwc](http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes) (Web). Example files for George Grätzer’s ‘First Steps’ are available in [info/FirstSteps](http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes).

The list for META\FONT is rather short:


This list only covers books in English: UK-TUG cannot hope to maintain a list of books in languages other than our own.

---

### 20 Where to find this article

Bobby Bodenheimer’s article, from which the present one was developed, used to be posted (nominally monthly) to newsgroup comp.text.tex and cross-posted to newsgroups news.answers and comp.answers. The most recently posted copy of that article is kept on CTAN in directory obsolete/help; it is no longer kept in the news.answers archives.

A version of the present article may be browsed via the World-Wide Web, at URL [http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes](http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes); the sources of the article are available from usergrps/uktug/faq.

Both the Francophone \TeX\ usergroup Gutenberg and the Czech/Slovak usergroup CS-TUG have published translations of this FAQ, with extensions appropriate to their languages.

In addition, the German usergroup Dante e.V. posts a FAQ in German to de.comp.tex, which is archived as usergrps/dante/de-tex-faq, and Marie-Paule Kluth posts a FAQ in French to fr.comp.text.tex, which is archived as help/LaTeX-FAQ-francaise.

### 21 Mailing lists about \TeX\ and its friends

There are (still) people who can use networks but can’t read Usenet news; for them, not all is lost if they can send and receive email.

The \TeX\hax digest is operated as a mailing list. Send a message ‘subscribe texhax’ to texhax-request@tex.ac.uk to join it. Its turn-around is not rapid, but questions submitted to it do eventually get answered.

Announcements of \TeX\-related installations on the CTAN archives are sent to the mailing list ctan-ann. Subscribe to

---

\[^{5}\text{That’s ‘Starting from Square One’}\]
the list by sending a message ‘subscribe ctan-ann <your name>’ to listserv@urz.uni-heidelberg.de

Issues related to METAPOST (and, increasingly, MetaPost) are discussed on the metamath mailing list; subscribe by sending a message ‘subscribe metamath <your name>’ to listserv@ens.fr

Several other \TeX-related lists may be accessed via listserv@urz.uni-heidelberg.de. Send a message containing the line ‘help’ to this address.

22 \TeX\ Tutorials

Some very fine tutorials have been written, over the years. Michael Doob’s splendid ‘Gentle Introduction’ to plain \TeX\ has been stable for a very long time. See info/gentle.pdf

More dynamic is Tobias Oetiker’s ‘(Not so) Short Introduction to \LaTeX\’, which is regularly updated, as people suggest better ways of explaining things, etc. See info/lshort/english/lshort.pdf

23 \BibTeX Documentation

\BibTeX, a program originally designed to produce bibliographies in conjunction with \LaTeX, is explained in Section 4.3 and Appendix B of Leslie Lamport’s \LaTeX manual (see question 19). The document “Designing \BibTeX Styles”, contained in the file bttxdoc.tex, gives a more complete description. The \LaTeX Companion (see question 19) also has information on \BibTeX and writing \BibTeX\ style files.

The document “Designing \BibTeX Styles”, contained in the file bttxak.tex, explains the postfix stack-based language used to write \BibTeX styles (\bst files). The file bttxbst.doc is the template for the four standard styles (plain, abbrv, alpha, unsvrt). It also contains their documentation. The complete \BibTeX documentation set (including the files above) is in biblio/bibtex/distrib/doc

There is a Unix \BibTeX man page in the web2c package (see question 39). Any copy you may find of a man page written in 1985 (before “Designing \BibTeX Styles” appeared) is obsolete, and should be thrown away.

24 The \PCTeX Manual

\PCTeX is a set of macros by Michael Wichura for drawing diagrams and pictures. The macros are freely available in graphics/pictex; however, the \PCTeX manual itself is not free. Unfortunately, TUG is no longer able to supply copies of the manual (as it once did), and it is now available only through Personal \TeX Inc, the vendors of \PCTeX (http://www.pctex.com/). The manual is not available electronically.

25 Finding (\BibTeX) \TeX\ macro packages

Before you ask for a \TeX\ macro or \LaTeX\ class or package file to do something, try searching Graham Williams’ catalogue, available as help/Catalogue/catalogue.html, or for efficient interactive searching via http://www.tex.ac.uk/tex-archive/help/Catalogue/catalogue.html; this lists many macro packages together with brief descriptive texts.

Having learnt of a file that seems interesting, search a CTAN archive for it (see question 26). For packages listed in The \BibTeX Companion (see question 19), the file info/companion.ctan may be consulted as an alternative to searching the archive’s index. It lists the current location in the archive of such files.

An alternative procedure is to use http://ctan.tug.org/ctan/, which permits limited ‘keyword’ searching for files on the CTAN sites.

26 Finding files in the CTAN archives

To find software at a CTAN site, you can use anonymous ftp to the host with the command ‘quote site index <term>’, or the searching script at http://www.dante.de/cgi-bin/ctan-index

To get the best use out of the ftp facility you should remember that <term> is a Regular Expression and not a fixed string, and also that many files are distributed in source form with an extension different to the final file. (For example \LaTeX\ packages are often distributed sources with extension \dtx rather than as package files with extension \sty.)

One should make the regular expression general enough to find the file you are looking for, but not too general, as the ftp interface will return only the first 20 lines that match your request.

The following examples illustrate these points. To search for the \LaTeX package ‘caption’, you might use the command:

quote site index caption.sty

but it will fail to find the desired package (which is distributed as caption.dtx) and does return unwanted ‘hits’ (such as hangcaption.sty). Also, although this example does not show it the ‘.’ in ‘caption.sty’ is used as the regular expression that matches any character. So

quote site index doc.sty

matches such unwanted files as language/swedish/slatex/doc2sty/makefile

Of course if you know the package is stored as .dtx you can search for that name, but in general you may not know the extension used on the archive. The solution is to add ‘/’ to the front of the package name and ‘\’, to the end. This will then search for a file name that consists solely of the package name between the directory separator and the extension. The two commands:

quote site index /caption/\.
quote site index /doc/\.

do narrow the search down sufficiently. (In the case of doc, a few extra files are found, but the list returned is sufficiently small to be easily inspected.)

If the search string is too wide and too many files would match, the list will be truncated to the first 20 items found. Using some knowledge of the CTAN directory tree you can usually narrow the search sufficiently. As an example suppose
you wanted to find a copy of the dvips driver for MS-DOS. You might use the command:

```
quote site index dvips
```

but the result would be a truncated list, not including the file you want. (If this list were not truncated 412 items would be returned!) However we can restrict the search to MS-DOS related drivers as follows.

```
quote site index msdos.*dvips
```

Which just returns relevant lines such as systems/msdos/dviware/dvips/dvips5528.zip

A basic introduction to searching with regular expressions is:

- Most characters match themselves, so "a" matches "a" etc.;
- " . " matches any character;
- " [abcD-F] " matches any single character from the set \{"a","b","c","D","E","F"\};
- " * " placed after an expression matches zero or more occurrences so "a*" matches "a" and "aaaaa", and "[a-zA-Z]*" matches a 'word';
- "\" 'quotes' a special character such as "." so ".\." just matches ".";
- "\-" matches the beginning of a line;
- "\$" matches the end of a line.

For technical reasons in the quote site index command, you need to 'double' any \ whence the string /caption\. in the above example. The quote site command ignores the case of letters. Searching for caption or CAPTION would produce the same result.

## D Bits and pieces of \TeX\n
### 27 What is a DVI file?

A DVI file (that is, a file with the type or extension .dvi) is \TeX\’s main output file, using \TeX\ in its broadest sense to include \LaTeX~, etc. ‘DVI’ is supposed to be an acronym for Device-Independent, meaning that the file can be printed on almost any kind of typographic output device. The DVI file is designed to be read by a driver (see question 28) to produce further output designed specifically for a particular printer (e.g., a LaserJet) or to be used as input to a previewer for display on a computer screen. DVI files use \TeX\’s internal coding; a \TeX\ input file should produce the same DVI file regardless of which implementation of \TeX\ is used to produce it.

A DVI file contains all the information that is needed for printing or previewing except for the actual bitmaps or outlines of fonts, and possibly material to be introduced by means of \special\ commands (see question 32).

The canonical reference for the structure of a DVI file is the source of dvitype (systems/knuth/ctexware/dvitype.web).

### 28 What is a driver?

A driver is a program that takes as input a dvi file (see question 27) and (usually) produces a file that can be sent to a typographic output device, called a printer for short.

A driver will usually be specific to a particular printer, although any PostScript printer ought to be able to print the output from a PostScript driver.

As well as the DVI file, the driver needs font information. Font information may be held as bitmaps or as outlines, or simply as a set of pointers into the fonts that the printer itself 'has'. Each driver will expect the font information in a particular form. For more information on the forms of fonts, see questions 29, 30, 31 and 60.

### 29 What are PK files?

PK files (packed raster) contain font bitmaps. The output from METAFONT (see question 57) includes a generic font (GF) file and the utility gftopk produces the PK file from that. There are a lot of PK files, as one is needed for each font, that is each magnification (size) of each design (point) size for each weight for each shape for each family. Further, since the PK files for one printer do not necessarily work well for another, the whole set needs to be duplicated for each printer type at a site. As a result, they are often held in an elaborate directory structure, or in ‘font library files’, to regularise access.

### 30 What are TFM files?

TFM stands for \TeX\ font metrics, and TFM files hold information about the sizes of the characters of the font in question, and about ligatures and kerns within that font. One TFM file is needed for each font used by \TeX, that is for each design (point) size for each weight for each shape for each family; one TFM file serves for all magnifications, so that there are (typically) fewer TFM files than there are PK files. The important point is that TFM files are used by \TeX\ (\LaTeX, etc.), but are not, in general, needed by the printer driver.

### 31 Virtual fonts

Virtual fonts for \TeX\ were first implemented by David Fuchs in the early days of \TeX, but for most people they started when Knuth redefined the format, and wrote some support software, in 1989. Virtual fonts provide a way of telling \TeX\ about something more complicated than just a one-to-one character mapping. The entities you define in a virtual font look like characters to \TeX\ (they appear with their sizes in a font metric file), but the DVI processor may expand them to something quite different. You can use this facility just to remap characters, to make a composite font with glyphs drawn from several sources, or to build up an effect in arbitrarily complicated ways — a virtual font may contain anything which is legal in a DVI file. In practice, the most common use of virtual fonts is to remap PostScript fonts (see question 62) or to build ‘fake’ maths fonts.

It is important to realise that \TeX\ itself does not see virtual fonts; for every virtual font read by the DVI driver there is a corresponding TFM file read by \TeX. Virtual fonts are normally created in a single ASCII vpl (Virtual Property List)
file, which includes both sets of information. The \texttt{vpl} program is then used to create the binary TM and VF files. The commonest way (nowadays) of generating \texttt{vpl} files is to use the \texttt{fontinst} package, which is described in detail in question 62. \texttt{qdtexvpl} is another utility for creating ad-hoc virtual fonts.

### 32 \special commands

\TeX{} provides the means to express things that device drivers can do, but about which \TeX{} itself knows nothing. For example, \TeX{} itself knows nothing about how to include PostScript figures into documents, or how to set the colour of printed text; but some device drivers do.

Such things are introduced to your document by means of \special commands; all that \TeX{} does with these commands is to expand their arguments and then pass the command to the DVI file. In most cases, there are macro packages provided (often with the driver) that provide a comprehensible interface to the \special; for example, there’s little point including a figure if you leave no gap for it in your text, and changing colour proves to be a particularly fraught operation that requires real wizardry. \texttt{vptovf} has standard graphics and colour packages that make file inclusion, rotation, scaling and colour via \specials all easy.

The allowable arguments of \special depend on the device driver you’re using. Apart from the examples above, there are \special commands in the \texttt{emTeX} drivers (e.g., \texttt{diwhi}, \texttt{diwhp}, \texttt{diwse}, etc.) that will draw lines at arbitrary orientations, and commands in \texttt{dscat} that permit the page to be set in landscape orientation.

### 33 Documented \LaTeX{} sources (.dtx files)

\LaTeX{} \texttt{2\epsilon}, and many support macro packages, are now written in a literate programming style (see question 49), with source and documentation in the same file. This format, known as ‘doc’, was originated by Frank Mittelbach. The documented sources conventionally have the suffix \texttt{.dtx}, and should normally be stripped of documentation before use with \LaTeX{}.

Alternatively you can run \LaTeX{} on a \texttt{.dtx} file to produce a nicely formatted version of the documented code. An installation script (with suffix \texttt{.ins}) is usually provided, which needs the standard \texttt{2\epsilon} \texttt{docstrip} package (among other things, the installation process strips all the comments that make up the documentation for speed when loading the file into a running \LaTeX{} system). Several packages can be included in one \texttt{.dtx} file, with conditional sections, and there facilities for indices of macros etc. Anyone can write \texttt{.dtx} files; the format is explained in \textit{The \LaTeX{} Companion} (see question 19). There are no programs yet to assist in composition.

\texttt{.dtx} files are not used by \LaTeX{} after they have been processed to produce \texttt{.sty} or \texttt{.cls} (or whatever) files. They need not be kept with the working system; however, for many packages the \texttt{.dtx} file is the primary source of documentation, so you may want to keep \texttt{.dtx} files elsewhere.

### 34 What are the EC fonts?

A font consists of a number of \textit{glyphs}. In order that the glyphs may be printed, there has to be some way of accessing them; in \TeX{} they’re arranged in a numerical order called an \textit{encoding}, and their number in the encoding is used. For various reasons, Knuth chose rather eccentric encodings; in particular, he chose different encodings for different fonts.

When \TeX{} version 3 arrived, some at least of the reasons for the eccentricity of Knuth’s encodings went away, and at TUG’s Cork meeting, an encoding for a set of 256 glyphs, for use in \TeX{} text, was defined. The intention was that these glyphs should cover ‘most’ European languages, in the sense of including all accented letters needed. (Knuth’s CMR fonts missed things necessary for Icelandic, Polish and Sami, for example, but the Cork fonts have them.) \texttt{2\epsilon} (see question 121) refers to the Cork encoding as ‘T1’, and provides the means to use fonts thus encoded to avoid problems with the interaction of accents and hyphenation (see question 100).

The only \texttt{METAFONT}-fonts that conform to the Cork encoding are the EC fonts (available as \texttt{fonts/ec}). They look CM-like, and are now regarded as ‘stable’ (in the same sense that the CM fonts are stable: their metrics are unlikely ever to change). Their serious disadvantage for the casual user is that they are large — each EC font is roughly twice the size of the corresponding CM font; what’s more, until corresponding fonts for mathematics are produced, the CM fonts must be retained.

The EC fonts supersede the experimental DC fonts, which have now been removed from the archives. They are distributed with a set of ‘Text Companion’ (TC) fonts that provide glyphs for symbols commonly used in text. The TC fonts are encoded according to the \texttt{TS1} encoding, and are not viewed as ‘stable’ in the same way as are the EC fonts are.

The Cork encoding is also implemented by the PSNFSS system (see question 60), for PostScript fonts.

### E Acquiring the software

#### 35 Repositories of \TeX{} material

To aid the archiving and retrieval of \TeX{}-related files, a TUG working group developed the Comprehensive \TeX{} Archive Network (CTAN). Each CTAN site has identical material, and maintains authoritative versions of its material. These collections are extensive; in particular, almost everything mentioned in this article is archived at the CTAN sites, even if its location isn’t explicitly stated.

The CTAN sites are currently \texttt{ftp.dante.de} (129.206.100.192), \texttt{ftp.tex.ac.uk} (128.232.1.87) and \texttt{ctan.tug.org} (158.121.106.16). The organisation of \TeX{} files on all these sites is identical and starts at \texttt{tex-archive/}. To reduce network load, please use the CTAN site or mirror closest to you. A complete and current list of CTAN sites and known mirrors can be obtained by using the \texttt{finger} utility on ‘user’ \texttt{ctan@ftp.tex.ac.uk}, \texttt{ctan@ftp.dante.de} or \texttt{ctanstug.cs.umb.edu}; it is also available as file \texttt{CTAN.sites}.

To find software at a CTAN site, use anonymous \texttt{ftp} to
the host, and then execute the command ‘quote site index <term>’ (see question 26 for details).

The email server ftpmail@ftp.dante.de provides an ftp-like interface through mail. Send a message containing just the line ‘help’ to your nearest server for details of use.

There is also the DECUS \TeX\ collection of material for VMS, Unix, MS-DOS, and the Macintosh. It is available via anonymous ftp from wuarchive.wustl.edu (128.252.135.4) in decus/tex/. It can also be obtained from the DECUS Library (reference number VS0058) in the US, or through your DECUS office outside of the US. To contact the DECUS Library, send mail or telephone:

DECUS
LIBRARY ORDER PROCESSING
334 South Street, SHR3-1/T25
Shrewsbury, MA 01545-4195
USA
Tel: 800-DECUS55 (within the USA, for information)
Fax: (+1) 508-841-3373 (for inquiries)

or send electronic mail for information to the DECUS \TeX\ Collection Editor, Ted Nieland (nieland@ted.hcst.com).

Finally, of course, the \TeX\ user who has no access to any sort of network may buy a copy of the archive on CD-ROM (see question 38).

36 Contributing a file to the archives

Use anonymous ftp to any CTAN archive (see question 35) and retrieve the file README.uploads in the root directory. It contains instructions for uploading files and notifying the appropriate people for that site.

If you cannot use ftp, mail your contribution to ctan@ctan.org and it will be passed along. You will make everyone’s life easier if you choose a descriptive and unique name for your submission, so it’s probably a good idea to check that your style file’s name is not already in use by means of the ‘site index’ command (see question 26).

37 Finding new fonts

A comprehensive list of METAFONT fonts is posted to comp. fonts and to comp.text.tex, roughly every six weeks, by Lee Quin (lee@sq.sq.com); it is available as info/metafont-list

The list contains details both of commercial fonts and of fonts available via anonymous ftp. Most of the fonts are available via anonymous ftp from the CTAN archives (see question 35).

38 \TeX\ CD-ROMs

If you don’t have access to the Internet, there are obvious attractions to \TeX\ collections on a CD-ROM. Even those with net access may find it a great convenience to have large quantities of \TeX\-related files to hand.

Ready-to-run \TeX\ systems on CD-ROM are available:

- A consortium of User Groups (notably TUG, UK-\TeX\, and GUTenberg) distribute the \TeX\ Live CD-ROM, now in its fourth edition. Members of many User Groups are receiving copies, so if you want a copy you would be well advised to join your local user group.

- The Dutch \TeX\ Users Group (NTG) publish the whole 4All\TeX\ workbench on a 2-CD-ROM set packed with all the Win-32 \TeX\ software, macros and fonts you can want. It is available from NTG direct (see question 17).

An alternative to the ready-to-run system is the CTAN archive snapshot; in general one would expect that such systems would be harder to use, but that the volume of resources offered would balance this extra inconvenience.

Walnut Creek CDROM provide a two-disc CD-ROM set, holding 1000Mb of \TeX\-related material. Information about the CD-ROM is available at http://www.cdr0m.com/titles/prog/tex.html, which also has a link to an ordering page. Walnut Creek’s address, etc., are:

Walnut Creek CDROM
4041 Pike Lane, Ste D-www
Concord, CA 94520
USA
Tel: (+1) 510 674-0783 or
800 786-9907 (within the USA and Canada)
Fax: (+1) 510 674-0821
Email: info@cdrom.com (for questions) and
orders@cdrom.com (for orders)

Prime Time Freeware produced \TeXcetera\ 1.1 in July 1994, which was a snapshot of CTAN taken in June 1994. Regular updates were planned (but have not apparently been forthcoming). The material is all compressed in ZIP format to fit it all on one CD, and to avoid the limitations of the ISO 9660 file system directory. You could buy the CD from:

Prime Time Freeware
370 Altair Way, Suite 150
Sunnyvale CA 94086
USA
Tel: (+1) 408 433 9662
Fax: (+1) 408 433 0727
Email: ptf@cfcl.com

or from many CD-ROM resellers, or the TUG office (see question 16). Price was around $60. Please note that PTF is not a big commercial firm, and is a good friend of the \TeX\ community.

F \TeX\ systems

39 (B)\TeX\ for different machines

We list here the free or shareware packages; see question 41 for commercial packages.

Unix Instructions for retrieving the Unix \TeX\ distribution via anonymous ftp are available in the document
Linux There are at least two respectable implementations of \TeX\ to run on Linux, NT\TeX\ (available as systems/unix/ntext) and te\TeX\ (available as systems/unix/texlive-1.0/distrib/sources). Beware the Slackware '96 CD-ROM distribution of NT\TeX\: it includes a version of the CM fonts that has deeply offended Don Knuth (since it contravenes his distribution conditions). The Slackware updates now offer te\TeX, as do most Linux distributions.

PC The em\TeX\ package for PCs running OS/2, MS-DOS or Windows includes \TeX\, Bu\TeX, previewers, and drivers, and is available in systems/msdos/extend as a series of zip archives. The package was written by Eberhard Mattes, and documentation is available in both German and English. Appropriate memory managers for using em\TeX\ with 386 (and better) processors and under Windows, are included in the distribution. A second package, g\TeX, runs under MS-DOS or Windows (and its users speak well of it). It is available from systems/msdos/gtex.

The most recent offering is an MS-DOS port of the Web2C 7.0 implementation, using the GNU \dj\pp compiler. It is available from systems/msdos/djgpp.

PC: Win32 fp\TeX, by Fabrice Popineau, is a version of tc\TeX\ for Windows systems. As such, it is particularly attractive to those who need to switch back and forth between Windows and Unix environments, and to administrators who need to maintain both (fp\TeX\ can use the same \textsf{texmf} tree as a tc\TeX installation). fp\TeX\’s previewer (\textsf{Windvi}) is based on \textsf{xdvi}, and takes advantage of extra facilities in the Win32 environment. Printing is available via \textsf{dvips} only. fp\TeX\ is available from systems/win32/fptex.

Mik\TeX, by Christian Schenk, is also a comprehensive distribution, developed separately from the tc\TeX work. It has its own previewer, Yap, which is itself capable of printing, though the distribution also includes a port of dvips. The current version is in systems/win32/miktecp.

Windows NT, other platforms A Power PC port of Mik\TeX\ is available from systems/win32/miktecp, and an AXP port is available from systems/win32/miktex-AXP.

Mac Oz\TeX\ is a shareware version of \TeX\ for the Macintosh. A DVI previewer and PostScript driver are also included. It should run on any Macintosh Plus, SE, II, or newer model, but will not work on a 128K or 512K Mac. It was written by Andrew Trevorrow, and is available in nonfree/systems/mac/oztex.

UK-T\TeX\G pre pays the shareware fee, so that its members may acquire the software without further payment. Questions about Oz\TeX\ may be directed to oztex@midway.uchicago.edu.

Another partly shareware program is CMac\TeX\ (available as systems/mac/cmactex), put together by Tom Kiffe. This is much closer to the Unix \TeX\ setup (it uses dvips, for instance). CMac\TeX\ includes a port of the latest version of Omega (see question 123).

OpenVMS \TeX\ for OpenVMS is available as systems/openvms/TeX97_CTAN.ZIP.

Standard tape distribution is through DECUS (see question 35).

Atari \TeX\ is available for the Atari ST in systems/atari.

If anonymous ftp is not available to you, send a message containing the line ‘\texttt{to atari@archive.umich.edu}’.

Amiga Full implementations of \TeX\ 3.1 (Pas\TeX) and METAFONT 2.7 are available in systems/amiga.

You can also order a CD-ROM containing this and other Amiga software from Walnut Creek CDROM, telephone (+1) 510-947-5997.

TOPS-20 \TeX\ was originally written on a DEC-10 under WAITS, and so was easily ported to TOPS-20. A distribution that runs on TOPS-20 is available via anonymous ftp from ftp.math.utah.edu (128.110.198.34) in pub/tex/pub/web.

40 \TeX\-friendly editors and shells

There are good \TeX\-writing environments and editors for most operating systems; some are described below, but this is only a personal selection.

Unix Try GNU \texttt{emacs}, and the AUCTeX mode (\texttt{support/auctex}). This provides menu items and control sequences for common constructs, checks syntax, lays out markup nicely, lets you call \TeX\ and drivers from within the editor, and everything else like this that you can think of. Complex, but very powerful.

VMS An \texttt{isle}t mode for editing \TeX\ source is available from TUG (see question 16) as \texttt{Xniques 1}, VAX Language-Sensitive Editor, by Kent MacPherson (1985).
MS-DOS There are several choices:

- The earlier (shareware) 4AllTEX workbench (systems/msdos/4allex) provided a very comprehensive environment written in 4DOS which lets you access most TeX-related software in a friendly way. You can choose your own editor; something such as QEdit or Brief is suitable. This whole package was available in easy-to-use form on CD-ROM from various TeX user groups, but the most recent version is now Win-32 specific rather than running under MS/DOS.

- TeXshell (systems/msdos/texshell) is a simpler, easily-customisable environment, which can be used with the editor of your choice.

- EddiTEX (systems/msdos/et4: also shareware) is a specially-written TeX editor which features intelligent colouring, bracket matching, syntax checking, online help and the ability to call TeX programs from within the editor. It is highly configured, and features a powerful macro language.

You can also use GNU emacs and AUCTeX under MS-DOS.

Windows 3.1 Your best public domain bet is probably to use MicroEmacs as an editor and control centre for TeX programs. The gTeX package (systems/msdos/gtex) comes with MicroEmacs ready to go, integrated with TeX, previewer, dvips and GhostScript.

TeXelmExtel (systems/msdos/entex-contrib/TeXelmExtel) is a Shell for emTeX or WinTeX and related tools under Windows. It includes a simple multiple-document editor, a built-in spelling checker, automatic OEM/ANSI character conversion, user-definable point-and-click Templates, support for the forward and inverse search mechanism of the DV1 driver for Windows and for automatic font generation. Besides the predefined tools, up to 10 user-defined tools can be set up.

On a PC with large enough memory, a version of GNU emacs, that will run under Windows, is available; thus you can also use AUCTeX under Windows.

Y&Y’s commercial (and high-quality) Windows previewer, docview, can be used as a good TeX shell, calling programs such as TeX, drivers, and editors (Y&Y supply the public domain PE, and recommend the commercial Epsilon) from customisable menus (see question 41 for details of Y&Y).

Scientific Word is a wysiwyg editing program, strong on maths, which uses Epsilon for output (see question 41 for contact address).

Windows ’9x, NT Winedt, a shareware package (systems/win32/winedt/winedt32.exe), is highly spoken of. It provides a shell for the use of TeX and related programs, as well as a powerful and well-configured editor.

OS/2 EddiTEX works under OS/2; look also at systems/os2/epmtex for a specific OS/2 shell.

Macintosh The commercial Textures provides an excellent integrated Macintosh environment with its own editor. More powerful still (as an editor) is the shareware Alpha (systems/mac/support/alpha) which is extensible enough to let you perform almost any TeX-related job. It works well with OzTEX.

Atari, Amiga and NeXT users also have nice environments. Epsilon users who like make should try support/latexmk

There is another set of shell programs to help you manipulate BibTeX databases.

41 Commercial TeX implementations

There are many commercial implementations of TeX. The first appeared not long after TeX itself appeared. Of the vendors, ArborText (formerly Textset) and Personal TeX are those who have survived longest (since the mid or early ’80s).

What follows is probably an incomplete list. Naturally, no warranty or fitness for purpose is implied by the inclusion of any vendor in this list. The source of the information is given to provide some clues to its currency.

In general, a commercial implementation will come ‘complete’, that is, with suitable previewers and printer drivers. They normally also have extensive documentation (ie, not just the TeXbook!) and some sort of support service. In some cases this is a toll free number (probably applicable only within the USA and/or Canada), but others also have email, and normal telephone and fax support.

Unix; TeX Silicon Graphics Iris/Indigo, Solaris 2.1, IBM RS/6000, DEC/RISC-Ultrix, HP 9000. “Complete TeX packages. Ready to use, fully documented and supported.”

ArborText Inc
1000 Victors Way
Suite 400
Ann Arbor MI 48108
USA
Tel: (+1) 313-996-3566
Fax: (+1) 313-996-3573

Source: TUGboat 15(1) (1994)

VAX/VMS; Convergent TeX Complete system for VAX/VMS machines (a version for Alphas was in preparation); includes Epsilon, multinational typesetting support, METAFONT and Web.

Northlake Software, Inc.
812 SW Washington, Ste 1100
Portland, OR 97201
USA
Tel: (+1) 503-228-3383
Fax: (+1) 503-228-5662
Email: rau@nls.com

Source: Email from Pat Rau, November 1994
PC; **True\TeX** Runs on Windows 3.1, Window NT and Windows 95.

Richard J. Kinch  
6994 Pebble Beach Court  
Lake Worth FL 33467  
USA  
Tel: (+1) 561-966-8400 Fax: (+1) 561-966-0962 Email: kinch@holonet.net  
Web: http://idt.net/~true\tex

Source: News posting from Richard Kinch, October 1997

**PC; \TeX** “Bitmap free \TeX for Windows.”

Y\&Y, Inc.  
45 Walden Street  
Concord MA 01742  
USA  
Tel: 800-742-4059 (within the USA)  
Tel: (+1) 508-371-3286  
Fax: (+1) 508-371-2004  
Email: sales-help@YandY.com and tech-help@YandY.com  
Web: http://www.YandY.com/

Source: Y\&Y announcement, February 1995

**pc\TeX** Long-established: pc\TeX32 is a Windows implementation.

Personal \TeX Inc  
12 Madrona Street  
Mill Valley, CA 94941  
USA  
Tel: 800-808-7906 (within the USA)  
Fax: (+1) 415-388-8856  
Email: texsales@pctex.com and texsupp@pctex.com  
Web: http://www.pctex.com/

Source: Mail from Personal \TeX Inc, September 1997

**PC; \VT\TeX** Also “Bitmap-free”.

MicroPress Inc  
68-30 Harrow Street  
Forest Hills, NY 11375  
USA  
Tel: (+1) 718-575-1816  
Fax: (+1) 718-575-8038  
Email: support@micopress-inc.com  
Web: http://www.micopress-inc.com/

Source: MicroPress home page, April 1996

**PC; micro\TeX** Micro\TeX and \TeX tools.

Micro Programs, Inc.  
251 Jackson Ave.  
Syosset, NY 11791

Source: AMS listing, November 1994

**PC; Scientific Word** Scientific Word and Scientific Workplace offer a mechanism for near-wysiwyg input of \LaTeX documents; they ship with True\TeX from Kinch (see above). Queries within the UK and Ireland should be addressed to Scientific Word Ltd, others should be addressed directly to the publisher, MacKichan Software Inc.

Dr Christopher Mabb  
Scientific Word Ltd.  
49 Queen Street  
Peterhead  
Aberdeenshire, AB42 1TU  
UK  
Tel: 0845 766 0340 (within the UK)  
Tel: +44 1779 490500  
Fax: 01779 490600 (within the UK)  
Email: christopher@sciword.demon.co.uk Web: http://www.sciword.demon.co.uk

MacKichan Software Inc.  
600 Ericksen Ave. NE, Suite 300  
Bainbridge Island WA 98110  
USA  
Tel: (+1) 206 780 2799  
Fax: (+1) 206 780 2857  
Email: info@mackichan.com  
Web: http://www.mackichan.com

Source: Mail from Christopher Mabb, May 1999

**Macintosh; Textures** “A \TeX system ‘for the rest of us’”; also gives away a \METAFONT implementation and some font manipulation tools.

Blue Sky Research  
534 SW Third Avenue  
Portland, OR 97204  
USA  
Tel: 800-622-8398 (within the USA)  
Tel: (+1) 503-222-9571  
Fax: (+1) 503-222-1643  
Email: sales@bluesky.com  
Web: http://www.bluesky.com/

Source: *TUGboat* 15(1)(1994)

**Amiga\TeX** A full implementation for the Commodore Amiga, including full, on-screen and printing support for all PostScript graphics and fonts, IFF raster graphics, automatic font generation, and all of the standard macros and utilities.
G  DVI drivers and previewers

42 DVI to PostScript conversion programs

The best public domain DVI to PostScript conversion program which runs under many operating systems is Tom Rokicki’s \textit{dvips}. \textit{dvips} is written in C and ports easily to other operating systems; it is available as \texttt{dviware/dvips}.

VMS versions are available through the DECUS library (see question 35), and also as part of the distribution of \texttt{TEX} for VMS (\texttt{systems/OpenVMS/TEX97\_CTAN.ZIP}).

A precompiled version for MS-DOS is available from \texttt{systems/mdos/dviware/dvips}

Karl Berry’s version of \textit{dvips} (called \textit{dvipsk}) has a configure script and path searching code similar to that in his other programs (e.g., \textit{web2c}); it is available from \texttt{dviware/dvipsk}.

Another good portable program is \textit{dviljk} by James Clark, which is also written in C and will compile under Unix, MS-DOS, VMS, and Primos; however, it does not support virtual fonts. It is available from \texttt{dviware/dviljk}

Macintosh users can use either the excellent drivers built into \texttt{OzTEX} or Textures, or a port of \textit{dvips} in the CMac\TeX{} package.

43 DVI drivers for HP LaserJet

The \texttt{em\TeX} package (see question 39) contains a driver for the LaserJet, \texttt{dviljkt}. Version 2.10 of the Beebe drivers supports the LaserJet. These drivers will compile under Unix, VMS, and on the Atari ST and DEC-20’s, and are available from \texttt{dviware/beebe}

Karl Berry’s \textit{dviljk}, which has the same path-searching code as his \textit{dvips} (see question 42), is available in \texttt{dviware/dviljk}

44 DVI previewers

\texttt{Em\TeX} and \texttt{g\TeX} for the PC, and \texttt{Oz\TeX} for the Macintosh, all come with previewers that can be used on those platforms. There is a good OS/2 Presentation Manager previewer in \texttt{em\TeX}, and a public domain Windows previewer (\texttt{dviware/dvwin}). Commercial PC \TeX{} packages (see question 41) have good MS-DOS and Windows previewers.

Previewers available for other operating systems include:

\texttt{xdi} The most widely usedpreviewer for the X Window System (and hence almost any Unix or modern VMS workstation); available in \texttt{dviware/xdi}

Karl Berry’s version of \texttt{xdi}, called \texttt{xdisk}, has features analogous to his \texttt{dvips} (see question 42); it is available in \texttt{dviware/xdisk}

\texttt{dvipage} For SunView on (old enough) Sun workstations. This was published in volume 13 of \texttt{comp.sources.unix} and is archived in \texttt{dviware/dvipage}

\texttt{xtex} An older previewer for the X Window System; available in \texttt{dviware/xtex}

\texttt{dviall} For Apollo Domain workstations; available in \texttt{dviware/dviall}

\texttt{dviljk} Karl Berry’s \textit{dviljk}, which has the same path-searching code as his \texttt{dvips} (see question 42), is available in \texttt{dviware/dviljk}

Macintosh users can use either the excellent drivers built into \texttt{OzTEX} or Textures, or a port of \textit{dvips} in the CMac\TeX{} package.

H  Support packages for \TeX{}

45 Fig, a \TeX{}-friendly drawing package

\texttt{(X)fig} is a menu driven tool that allows you to draw objects on the screen of an X workstation. \texttt{transfig} is a set of tools which translate the code \texttt{fig} produces to other graphics languages including PostScript and the \texttt{L\TeX} picture environment. They are available in \texttt{graphics/xfigg}\texttt{raphic/\texttt{g\TeXxx}fig}

\texttt{fig} is supported by Micah Beck (beck@cs.cornell.edu) and \texttt{transfig} is maintained by Brian Smith (bsmith@lbl.gov). Another tool for \texttt{fig} conversion is \texttt{fig2mf} which generates METAFONT code from \texttt{fig} input. It is available in \texttt{graphics/fig2mf}

46 T\TeX{}CAD, a drawing package for \l\TeX{}

\texttt{T\TeX{}CAD} is a program for the PC which enables the user to draw diagrams on screen using a mouse or arrow keys, with an on-screen menu of available picture-elements. Its output is code for the \texttt{L\TeX} picture environment. Optionally, it can be set to include lines at all angles using the \texttt{em\TeX} driver-family \texttt{\special{}} (see question 32). \texttt{T\TeX{}CAD} is part of the \texttt{em\TeX} distribution.

A Unix port of the program exists, \texttt{graphics/xtexcad/xtexcad-2.4.tar.gz}

47 Spelling checkers for work with \TeX{}

For Unix, \texttt{ispell} is probably the program of choice. It is available in \texttt{support/ispell}; beware of any version with a number 4.x — such versions represent a divergent version of the source which lacks many useful facilities of the 3.x series.

For MS-DOS, there are several programs. \texttt{amspell} can be called from within an editor (available as \texttt{support/amspell}). \texttt{jspell} is an extended version of \texttt{ispell} (available as \texttt{support/jspell}).

For the Macintosh, \texttt{Excaliber} is the program of choice.
It will run in native mode on both sorts of Macintosh, and is available as systems/mac/support/excalibur/Excalibur-2.5-sit.hqx (there are other dictionaries in the same directory).

For VMS, a spell checker can be found in support/vmspell

48 The VORTEX package

VORTEX (available in support/vortex) is a package of programs written at the University of California at Berkeley, and was described by Michael A. Harrison in “News from the VORTEX project” in TUGboat 10(1), pp. 11–14, 1989. It includes several nice previewers and some emacs modes for \TeX and Bu\TeX. The VORTEX distribution is not maintained, and now looks distinctly long in the tooth (it was never upgraded to \TeX version 3).

VORTEX needed a separate workstation to run \TeX in the background; modern PCs for the home can provide more processor power (than was available to VORTEX) in a single box. This fact has been recognised by Blue Sky Research in their ‘Lightning Textures’ (which runs on a Macintosh in a somewhat similar way) and by TCI Software Research in ‘Scientific Word’ (see question 41), and is also the basis of many of the other environments mentioned in question 40.

I Literate programming

49 What is Literate Programming?

Literate programming is the combination of documentation and source together in a fashion suited for reading by human beings. In general, literate programs combine source and documentation in a single file. Literate programming tools then parse the file to produce either readable documentation or compilable source. The \TeX style of literate programming was created by D. E. Knuth during the development of his \TeX typesetting software.

Discussion of literate programming is conducted in the newsgroup comp.programming.literate. The literate programming FAQ is stored as help/LitProg-FAQ

50 WEB for C, FORTRAN, and other languages

\TeX is written in the programming language WEB; WEB is a tool to implement the concept of “literate programming”.

CW\TeX, a WEB for C programs, written by Silvio Levy, is available as web/c_c++/cw\TeX

Spidery WEB supports many languages including Ada, awk, and C. It was written by Norman Ramsey and, while not in the public domain, is usable free. It is available in web/spiderweb

FW\TeX is a version for Fortran, Ratfor, and C written by John Krommes. It is available in web/f\TeX

SchemeWEB is a Unix filter that translates SchemeWEB into \TeX source or Scheme source. It was written by John Ramsdell and is available in web/schemeweb

APLWEB is a version of WEB for APL and is available in web/apl/aplweb

FunnelWEB is a version of WEB that is language independent. It is available in web/funnelweb

Other language independent versions of WEB are nuweb (which is written in ANSI C), available in web/nuweb, and noweb, available in web/noweb

A WEB for plain \TeX macro files, using noweb, has recently been made available in web/tweb

J Format conversions

51 Conversion between (Be)\TeX and others

troff to-\LaTeX (available as support/troff-to-\LaTeX), written by Kamal Al-Yahya at Stanford University (California, USA), assists in the translation of a troff document into \LaTeX format. It recognises most \_ms and \_man macros, plus most \_egn and some \_hlp processor commands. Anything fancier needs to be done by hand. Two style files are provided. There is also a man page (which converts very well to \LaTeX...). The program is copyright but free. An enhanced version of this program, \_tr\_to-\LaTeX, is available in support/tr2latex

The DECUS \TeX distribution (see question 35) also contains a program which converts troff to \TeX.

WordPerfect \_wp\_to-\LaTeX (available as support/wp2\LaTeX) has recently been much improved, and is now available both for MS-DOS and for Unix systems, thanks to its current maintainer Jaroslav Fojtik.

PC-Write pcwritex.arc, from support/pcwritex, is a print driver for PC-Write that “prints” a PC-Write V2.71 document to a \TeX-compatible disk file. It was written by Peter Flynn at University College, Cork, Republic of Ireland.

runoff Peter Vanroose’s (vanroose@esat.kuleuven.ac.be) conversion program is written in VMS Pascal. The sources and a VAX executable are available from support/ rtnotex

refer/tib There are a few programs for converting bibliographic data between Bu\TeX and refer/tib formats. They are in biblio/bibtex/\_util\_\_\_\_\_tools

In spite of the directory name, it also contains a shell script to convert Bu\TeX to refer as well. The collection is not maintained.

RTF A program for converting Microsoft’s Rich Text Format to \TeX is available in support/rtf2\LaTeX, which was written and is maintained by Robert Lupton (rhl@astro.princeton.edu). There is also a converter to \TeX by Erwin Wechtl, in support/rtf2latex

Translation to RTF may be done (for a somewhat constrained set of \TeX documents) by \_x2rtf, which can produce ordinary RTF, Windows Help RTF (as well as HTML, see question 54). \_x2rtf is supported on various Unix platforms and under Windows 3.1; it is available from support/tex2rtf
Microsoft Word A rudimentary program for converting MS-Word to \TeX is wd2latex, for MS-DOS (\texttt{dviware/wd2latex}); a better idea, however, is to convert the document to RTF format and use one of the RTF converters mentioned above.

A FAQ that deals specifically with conversions between \TeX-based formats and word processor formats is regularly posted to \texttt{comp.text.tex}, is available via \url{http://www.kfa-juelich.de/isr/1/texivconv/texivconv.html} and is archived as \texttt{help/wp-conv/wp-conv.zip}.

A group at Ohio State University (USA) is working on a common document format based on SGML, with the ambition that any format that could be translated to or from this one. \texttt{FrameMaker} provides “import filters” to aid translation from alien formats (possibly including \TeX) to \texttt{FrameMaker’s} own.

52 Conversion from (B)\TeX{} to plain ASCII

The aim here is to emulate the Unix \texttt{nroff}, which formats text as best it can for the screen, from the same input as the Unix typesetting program \texttt{nroff}.

Ralph Droms (droms@bucknell.edu) has a style file and a program that provide the \texttt{IStex} equivalent of \texttt{nroff}, though it doesn’t do a good job with tables and mathematics. The software is available in \texttt{support/txt}; the original \texttt{dviiatty} often does an acceptable job and is available in \texttt{dviware/dviiatty}.

Another possibility is to use \texttt{screen.sty} (available as \texttt{macros/latex209/contrib/misc/screen.sty}). Use a \texttt{dviiatty} program of some kind; you might try \texttt{dviware/crude-type} as well. Another possibility is to use the \texttt{IStex} to ASCI conversion program, \texttt{IStex} (\texttt{support/12a}), although this is really more of a \texttt{dc-TeX}ing program.

The canonical dc-\TeX{}Xing program is \texttt{detex} (\texttt{support/detex}), which removes all comments and control sequences from its input before writing it to its output. Its original purpose was to prepare input for a dumb spelling checker.

53 Conversion from SGML or HTML to \TeX{}

SGML is a very important system for document storage and interchange, but it has no formatting features; its companion ISO standard DSSSL (\url{http://www.jclark.com/dsssl/}) is designed for writing transformations and formatting, but this has not yet been widely implemented. Some SGML authoring systems (e.g., SoftQuad \texttt{Author/Editor}) have formatting abilities, and there are high-end specialist SGML typesetting systems (e.g., Miles33’s \texttt{Genera}). However, the majority of SGML users probably transform the source to an existing typesetting system when they want to print. \TeX{} is a good candidate for this. There are three approaches to writing a translator:

1. Write a free-standing translator in the traditional way, with tools like \texttt{yacc} and \texttt{lex}; this is hard, in practice, because of the complexity of SGML.

2. Use a specialist language designed for SGML transformations; the best known are probably \texttt{Omninack} and \texttt{Balise}. They are expensive, but powerful, incorporating SGML query and transformation abilities as well as simple translation.

3. Build a translator on top of an existing SGML parser. By far the best-known (and free!) parser is James Clark’s \texttt{sgmlls}, and this produces a much simpler output format, called ESIS, which can be parsed quite straightforwardly (one also has the benefit of an SGML parse against the DTD). Two good public domain packages use this method:

   - David Megginson’s \texttt{sgml2pm}, written in Perl 5. Available from \url{http://www.perl.com/CPAN/modules/by-module/SGMLS}

Both of these allow the user to write ‘handlers’ for every SGML element, with plenty of access to attributes, entities, and information about the context within the document tree.

If these packages don’t meet your needs for an average SGML typesetting job, you need the big commercial stuff.

Since HTML is simply an example of SGML, we do not need a specific system for HTML. However, Nathan Torkington (\texttt{nathan.torkington@vuw.ac.nz}) developed \texttt{html2latex} from the HTML parser in NCSA’s Xmosaic package. The program takes an HTML file and generates a \texttt{LaTeX} file from it. The conversion code is subject to NCSA restrictions, but the whole source is available as \texttt{support/html2latex}.

Michel Goossens and Janne Saarela published a very useful summary of SGML, and of public domain tools for writing and manipulating it, in \textit{TUGboat} \textbf{16}(2).

54 (B)\TeX{} conversion to HTML

\TeX{} is a typesetting language, not a markup system. With properly-used \texttt{LaTeX}, you may be luckier, but don’t expect a free lunch. Remember that a) if you want a really good Web document, you had better redesign it from scratch, and b) HTML (even HTML3) has pretty poor ‘typesetting’ facilities, and anything beyond the trivial will probably need to end up a graphic. HTML 4 plus CSS is much better from this perspective.

\texttt{IStex2HTML} (\texttt{support/latex2html}) is a package by Nikos Drakos (mostly of perl scripts) that breaks up a \texttt{IStex} document into one or more components, and links them together so that they can be read over the World-Wide Web as an hypertext document. It defines a mapping between \texttt{IStex} intra-document references and hyperlinks, and extends the mechanisms to permit reference to other (possibly remote) documents and other Internet resources. It translates \texttt{IStex} accented and other characters (as best it can) to things that World-Wide Web browsers can display, and translates mathematics (and other things that browsers can’t deal with) to...
images that can be loaded in-line into the hypertext document.

\texttt{LATEX2HTML} needs \texttt{Perl}, the \texttt{PBM} utilities, \texttt{dvips}, \texttt{GhostScript}, and other sundries; it assumes it is running on a Unix system. Michel Goossens and Janne Saarela published a detailed discussion of \texttt{LATEX2HTML}, and how to tailor it, in \textit{TUGboat} 16(2).

There are two possible strategies:

1. Free-standing \texttt{LATEX} to HTML translations. Hard, but not impossible. Julian Smart’s \texttt{latex2html} (available from \texttt{support/latex2html}) does a plausible job on a subset of \texttt{LATEX};

2. Writing an HTML-output backend in \texttt{LATEX} itself. See Sebastian Rahtz’ paper in \textit{TUGboat} 16(3) for a discussion of how to go about this for the general case of \texttt{SGML}.

55 Making hypertext documents from \texttt{LATEX}

If you want on-line hypertext with a \texttt{LATEX} source, probably on the World Wide Web, consider four technologies (which overlap):

1. Try direct \texttt{LATEX} conversion to HTML; see question 54;

2. Rewrite your document using Texinfo (see question 13), and convert that to HTML;

3. Look at Adobe Acrobat, an electronic delivery system guaranteed to preserve your typesetting perfectly. See question 56;

4. Invest in the \texttt{hyperLATEX} conventions (standardised \texttt{\special} commands); there are supporting macro packages for plain \texttt{LATEX} and \texttt{LATEX2}.

The \texttt{hyperLATEX} project aims to extend the functionality of all the \texttt{LATEX} cross-referencing commands (including the table of contents) to produce \texttt{\special} commands which are parsed by DVI processors conforming to the \texttt{hyperLATEX} guidelines; it provides general hypertext links, including those to external documents.

The \texttt{hyperLATEX} specification says that conformant viewers/translators must recognize the following set of \texttt{\special} commands:

\begin{verbatim}
href: html:<a href = "href_string">
name: html:<a name = "name_string">
end: html:<a/>
image: html:<img src = "href_string">
base_name: html:<b name = "href_string">
\end{verbatim}

The \texttt{href}, \texttt{name} and \texttt{end} commands are used to do the basic hypertext operations of establishing links between sections of documents.

Further details are available on http://xxx.lanl.gov/hypertext/; there are two commonly-used implementations of the specification, a modified \texttt{ads} and (recent releases of) \texttt{dvips}. Output from the latter may be used in recent releases of \texttt{GhostScript} or Acrobat Distiller.

56 Making Acrobat documents from \texttt{LATEX}

There are now two general routes to Acrobat output: Adobe’s original ‘distillation’ route, and the use of \texttt{PdfLATEX} (see question 125).

For simple documents (with no hyper-references), you can either

- distill ‘ordinary’ PostScript output of your document, or (on a Windows or Macintosh machine with \texttt{AcrobatExchange} installed) pass it through the PDFWriter or Distiller Assistant in place of a printer, or
- process with \texttt{PdfLATEX}, enabling Acrobat output.

(Note that the PDFWriter route is a dead end: it can only be used in this simple mode, as it cannot create hyperlinks.)

To translate all the \texttt{LATEX} cross-referencing into Acrobat links, you need a \texttt{LATEX} package to suitably redefine the internal commands. There are two of these for \texttt{LATEX2}, both based on the \texttt{hyperLATEX} specification (see question 55): Sebastian Rahtz’s \texttt{hyperref} (available from \texttt{macros/latex/contrib/supported/hyperref}), and Michael Mehlich’s \texttt{hyper} (available from \texttt{macros/latex/contrib/supported/hyper}). \texttt{Hyperref} can operate using \texttt{PdfLATEX} primitives rather than the \texttt{hyperLATEX} conventions. You can use \texttt{dvips} or Y&Y’s \texttt{DVIPSONE} to translate the DVI into PostScript acceptable to Distiller.

Sadly, there is no free implementation of all of Distiller’s functionality, but \texttt{GhostScript} (version 4.00 onwards) provides some restricted distilling capability, and \texttt{Distiller} itself is now remarkably cheap (for academics at least).

Adobe’s \texttt{AcrobatReader} is available for a very wide range of platforms. For those still omitted, \texttt{GhostScript} (versions 3.51 onwards) can display and print PDF files.

Work on a DVI to PDF translator is in progress, but shows no sign of immediate release (Sergey Lesenko spoke about the work at TUG’96 and again at TUG’97).

K METAFONT

57 Getting METAFONT to do what you want

\texttt{METAFONT} allows you to create your own fonts, yet most \texttt{LATEX} users will never need to use it explicitly. \texttt{METAFONT}, unlike \texttt{LATEX}, requires some customisation: each output device for which you will be generating fonts needs a mode associated with it. Modes are defined using the \texttt{mode_def} convention described on page 94 of The \texttt{METAFONT}book (see question 19). You will need a file, conventionally called \texttt{local.mf}, containing all the \texttt{mode_defs} you will be using. If \texttt{local.mf} doesn’t already exist, Karl Berry’s collection of modes, available as \texttt{fonts/modes/modes.mf}, is a good starting point (it can be used as a ‘\texttt{local.mf}’ without modification in a ‘big enough’ implementation of \texttt{METAFONT}). Lists of settings for various output devices are also published periodically in \textit{TUGboat} (see question 16). Now create a plain base file using \texttt{initmf}, \texttt{plain.mf}, and \texttt{local.mf}:
%% inimf
This is METAFONT...

*plain     you type ‘plain’
(output)
*input local you type this
(output)
*dump      you type this
(output)

Beginning to dump on file plain ...

This will create a base file named plain.base (or something similar; for example, it will be PLAIN.BAS on MS-DOS systems) which should be moved to the directory containing the base files on your system (note that some systems have two or more such directories, one for each ‘size’ of METAFONT used).

Now you need to make sure METAFONT loads this new base when it starts up. If METAFONT loads the plain base by default on your system, then you’re go. Under Unix (using the default web2c distribution) this does indeed happen, but we could for instance define a command mf which executes virmf &plain loading the plain base file.

The usual way to create a font with plain METAFONT is to start it with the line

\mode=<mode name>; mag=<magnification>; input <font file name>

in response to the ‘**’ prompt or on the METAFONT command line. If <mode name> is unknown or omitted, the mode defaults to ‘proof’ and METAFONT will produce an output file called <font file name>.2602gf) The <magnification> is a floating point number or ‘magstep’ (magsteps are defined in The METAFONT book and The TeXbook). If mag=<magnification> is omitted, then the default is 1 (magstep 0). For example, to generate cmr10 at 12pt for an epson printer you would type

mf \mode=epson; mag=magstep 1; input cmr10

Note that under Unix the \ and ; characters must usually be quoted or escaped, so this would typically look something like

mf ‘\mode=epson; mag=magstep 1; input cmr10’

If you don’t have inimf or need a special mode that isn’t in the base, you can put its commands in a file (e.g., ln03.mf) and invoke it on the fly with the \mode command. For example, to create cmr10.300gf for an LN03 printer, using the file

% This is ln03.mf as of 1990/02/27
% mode_def courtesy of John Sauter
proofing:=0;
fontmaking:=1;
tracingtitles:=0;
pixels_per_inch:=300;
blacker:=0.65;
fillin:=-0.1;
o_correction:=.5;

(note the absence of the mode_def and enddef commands, you would type

mf \smode="ln03"; input cmr10

This technique isn’t one you should regularly use, but it may prove useful if you acquire a new printer and want to experiment with parameters, or for some other reason are regularly editing the parameters you’re using. Once you’ve settled on an appropriate set of parameters, you should use them to rebuild the base file that you use.

A summary of the above written by Geoffrey Tobin, and tips about common pitfalls in using METAFONT, is available as info/metafont-for-beginners.txt

58 Which font files should be kept

METAFONT produces from its run three files, a metrics (TFM) file, a generic font (GF) file, and a log file; all of these files have the same base name as does the input (e.g., if the input file was cmr10.mf, the outputs will be cmr10.tfm, cmr10.nnngf7 and cmr10.log).

For TeX to use the font, you need a TFM file, so you need to keep that. However, you are likely to generate the same font at more than one magnification, and each time you do so you’ll (incidentally) generate another TFM file; these files are all the same, so you need only keep one of them.

To preview or to produce printed output, the DVI processor will need a font raster file; this is what the GF file provides. However, while there used (once upon a time) to be DVI processors that could use GF files, modern processors use packed raster (PK) files. Therefore, you need to generate a PK file from the GF file; the program gftopk does this for you, and once you’ve done that you may throw the GF file away.

The log file should never need to be used, unless there was some sort of problem in the METAFONT run, and need not be ordinarily kept.

59 Getting bitmaps from the archives

Most people these days start using TeX with a 300 dots-per-inch (dpi) laser printer, and Computer Modern bitmap fonts for this resolution are supplied with most TeX packages. There are also two such sets available on CTAN: fonts/cm/pk/pk300.zip (for write-black printer engines) and fonts/cm/pk/pk300w.zip (for write-white engines). However, some users want to send their work to high quality typesetting machines (typically with a resolution of 1270 dpi or greater); it is also becoming more common to use a 600 dpi laser printer. Why don’t the archives or suppliers provide bitmap fonts at these sizes? There are two reasons:

1. When a bitmap font is created with METAFONT, it needs to know the characteristics of the device; who knows what 600 or 1270 dpi device you have? (Of course, this objection applies equally well to 300 dpi}

---

6The command name is symbolically linked to virmf, and virmf loads command_name.base
7Note that the file name may be transmitted by such operating systems as MS-DOS, which don’t permit long file names
printers, but at this resolution the greatest difference lies between the write-black and write-white devices; these differences tend to mask all lesser components such as overshoot or fill-in.)

2. Bitmap fonts get big at high resolutions. Who knows what fonts at what sizes you need?

It would be possible to provide some set of 1270 dpi bitmap fonts in the archives, but it would take a lot of space, and might not be right for you.

So what to do? You can build the fonts you need yourself with METAFONT: this isn’t at all hard, and some drivers help you (dvips, and the emTeX drivers construct the METAFONT commands). You might need to look at Karl Berry’s collection of METAFONT modes (fonts/modes/modes.mf). Alternatively, if it is a PostScript device you have, consider using the fonts in Type 1 font format. You can buy all the Computer Modern fonts in outline form from Blue Sky Research, Kincl or Y&Y (see question 41 for addresses), or you can use Basil Malyshiev’s public domain versions in fonts/cm/ps-type1 (the Paradissa collection is complete, but has largely been replaced by the better BaKoMa collection). The Blue Sky fonts, too, may now be found on CTAN.

L PostScript and \TeX

60 Using PostScript fonts with \TeX

In order to use PostScript fonts, \TeX needs metric (called TFM) files. Several sets of metrics are available from the archives; for mechanisms for generating new ones, see question 62. You also need the fonts themselves; PostScript printers come with a set of fonts built in, but to extend your repertoire you might not be right for you.

2. Bitmap fonts get big at high resolutions. Who knows what fonts at what sizes you need?

It would be possible to provide some set of 1270 dpi bitmap fonts in the archives, but it would take a lot of space, and might not be right for you.

So what to do? You can build the fonts you need yourself with METAFONT: this isn’t at all hard, and some drivers help you (dvips, and the emTeX drivers construct the METAFONT commands). You might need to look at Karl Berry’s collection of METAFONT modes (fonts/modes/modes.mf). Alternatively, if it is a PostScript device you have, consider using the fonts in Type 1 font format. You can buy all the Computer Modern fonts in outline form from Blue Sky Research, Kincl or Y&Y (see question 41 for addresses), or you can use Basil Malyshiev’s public domain versions in fonts/cm/ps-type1 (the Paradissa collection is complete, but has largely been replaced by the better BaKoMa collection). The Blue Sky fonts, too, may now be found on CTAN.

61 Previewing files using PostScript fonts

Most \TeX previews display only bitmap PK fonts. If you want to preview documents using PostScript fonts, you have three choices:

1. Convert the DVI file to PostScript and use a PostScript previewer. Some modern Unix X implementations have this built in (as does NeXT-step); (X11) Unix, Windows, OS/2, and MS-DOS users can use the free Ghostscript (support/ghostscript), a complete level 2 implementation.

2. Under Windows on a PC, or on a Macintosh, let Adobe Type Manager display the fonts. Textures (Macintosh) works like this, and under Windows you can use Y&Y’s drxwinda for bitmap-free previewing. (See question 41 for details of these suppliers.)

3. If you have the PostScript fonts in Type 1 format, use ps2pk (fonts/utilities/ps2pk) or gsftopk (designed for use with the Ghostscript fonts; fonts/utilities/ gsftopk) to make PK bitmap fonts which your previewer will understand. This can produce excellent results, also suitable for printing with non-PostScript devices. Check the legalities of this if you have purchased the fonts. The very commonest PostScript fonts such as Times and Courier come in Type 1 format on disk with Adobe Type Manager (often bundled with Windows, and part of OS/2).

62 \TeX font metric files for PostScript fonts

Font vendors such as Adobe supply metric files for each font, in AFM (Adobe Font Metric) form; these can be converted to TFM (\TeX Font Metric) form. The CTAN archives have prebuilt metrics which will be more than enough for many people (fonts/psfonts; beware — this directory is at the root of a huge tree), but you may need to do the conversion yourself if you have special needs or acquire a new font. One important question is the encoding of (Latin character) fonts; while we all more or less agree about the position of about 96 characters in fonts (the basic ASCII set), the rest of the (typically) 256 vary. The most obvious problems are with floating accents and special characters such as the ‘pounds sterling’ sign. There are three ways of dealing with this: either you change the \TeX macros which reference the characters (not much fun, and error-prone); or you change the encoding of the font (easier than you might think); or you use virtual fonts (see question 31) to pretend to \TeX that the encoding is the same as it expects. If you use \Euler, it allows for changing the encoding in \Euler; read the \Euler Companion (see question 19) for more details. In practice, if you do much
non-English (but Latin script) typesetting, you are strongly recommended to use the `fontenc` package with option ‘T1’ to select T1 (`Cork`: see question 34) encoding.

Alan Jeffrey’s `fontinst` package (`fonts/utilities/fontinst`) is an AFM to TFM converter written in `tex`; it is used to generate the files used by `lmodern`’s PSNFSS package to support use of PostScript fonts. It is a sophisticated package, not for the faint-hearted, but is powerful enough to cope with most needs. Much of its power relies on the use of AFM files.

For slightly simpler problems, Rokicki’s `afm2tfm`, distributed with `dvips`, is fast and efficient; note that the metrics and styles that come with `dvips` are not currently `lmodern` compatible, but Karl Berry plans to distribute metrics directly compatible with PSNFSS in his `dvips` package.

For the Macintosh, there is a program called `EdMetrics` which does the job (and more). It comes with the Textures distribution, but is in fact free software, available as `systems/mac/textures/utilities/EdMetrics.sea.hqx`.

MS-DOS users can buy (`see question 41`) Y&Y’s Font Manipulation Tools package which includes a powerful `afmtotfm` program among many other goodies.

### 63 Problems using PostScript fonts

For the typical `tex` user trying to use the PSNFSS (see question 60) package, three questions often arise. First, you have to declare to the DVI1 driver that you are using PostScript fonts; in the case of `dvips`, this means adding lines to the `psfonts.map` file. Otherwise, `dvips` will try to find PK files. If the font isn’t built into the printer, you have to acquire it (in many cases this means buying it from a commercial supplier!). You then have to instruct the driver to download it with each job (the mechanism depends on your driver). So it’s no good just installing the metrics for Optima and expecting it to work. You have to pay hard cash for the font itself, which will come (for Unix and MS-DOS users) in `pfb` (Printer Font Binary) form.

Second, you cannot expect your previewer to suddenly start displaying PostScript fonts; most of them know only about PK bitmap fonts such as Computer Modern. `ps2pk` (`fonts/utilities/ps2pk`) can create these from the `pfb` file you have bought; this would also let you use the fonts with non-PostScript device drivers such as the emTeX ones. You are responsible for making sure you are not breaking the licence restrictions on fonts that you have bought.

Third, the stretch and shrink between words is a function of the font metric; it is not specified in AFM files, so different converters choose different values. The PostScript metrics that come with PSNFSS used to produce quite tight setting, but they were revised in mid 1995 to produce a compromise between American and European practice. Really sophisticated users may not find even the new the values to their taste, and want to override them. Even the casual user may find more hyphenation or overfull boxes than CMR produces; but CMR is extremely generous.

### 64 Choice of scalable outline fonts

If you are interested in text alone, you can use any of over 20,000 fonts! in Adobe Type 1 format (called ‘PostScript fonts’ in the `tex` world and ‘ATM fonts’ in the DTP world), or any of several hundred fonts in TrueType format. That is, provided of course, that your previewer and printer driver support scalable outline fonts.

`tex` itself cares only about metrics, not the actual character programs. You just need to create a `tex` metric file TFM using some tool such as `afm2tfm`, `afmtotfm` (from Y&Y, `see question 41`) or `fontinst`. For the previewer or printer driver you need the actual outline font files themselves (pfa for Display PostScript, `pfb` for ATM on IBM PC, Mac outline font files on Macintosh).

If you also need mathematics, then you are severely limited by the demands that `tex` makes of maths fonts (for details, see the paper by B.K.P. Horn in TUGboat 14(3)). For maths, then, there are relatively few choices:

**Computer Modern** (75 fonts — optical scaling) Donald E. Knuth

Note that CM is available in scalable outline form. There are commercial as well as public domain versions, and there are both Adobe Type 1 and TrueType versions. Some of these are ‘commercial grade’, with full hand-tuned hinting, some render very poorly, while others are merely incompatible with Adobe Type Manager (ATM).

**Lucida Bright** with **Lucida New Math** (25 fonts) Chuck Bigelow and Kris Holmes

Lucida is a family of related fonts including serif, sans serif, sans serif fixed width, calligraphic, blackletter, fax, Kris Holmes’ connected handwriting font, etc; they’re not as ‘spindly’ as Computer Modern, with a large x-height, and include a larger set of maths symbols, operators, relations and delimiters than CM (over 800 instead of 384: among others, it also includes the AMS `msam` and `msbm` symbol sets). The planned ‘Lucida Bright Expert’, (14 fonts) adds serifed fixed width, another handwriting font, smallcaps, bold maths, upright ‘maths italic’, etc., to the set. The distribution includes support for use with `plain` `tex` and `lmodern` 2.09. Support under `lmodern` 2.09 is provided in PSNFSS (see question 60) thanks to Sebastian Rahtz.

**MathTime 1.1** (3 fonts) Texplorators (Michael Spivak)

The set contains maths italic, symbol, and extension fonts, designed to work well with Times-Roman. These are typically used with Times, Helvetica and Courier (which are resident on many printers, and which are supplied with some PC versions). In addition you may want to complement this basic set with Adobe’s Times Smallcap, and perhaps the set of Adobe ‘Math Pi’ fonts, which include blackboard bold, blackletter, and script faces. The distribution includes support for use with `plain` `tex` and `lmodern` 2.09 (including code to link in Adobe Math Pi 2 and Math Pi 6). Support under `lmodern` 2.09 is provided in PSNFSS (see question 60) thanks to Sebastian Rahtz.
Adobe Lucida, LucidaSans and LucidaMath (12 fonts)
Lucida and LucidaMath are generally considered to be a bit heavy. The three maths fonts contain only the glyphs in the CM maths italic, symbol, and extension fonts. Support for using LucidaMath with \TeX{} is not very good; you will need to do some work reencoding fonts etc. (In some sense this is the ancestor of the LucidaBright plus LucidaNewMath font set.)

Concrete, the AMS maths fonts etc. Donald E. Knuth and the AMS.
These are sometimes mentioned as alternatives to CM, but they are really adjuncts, in that you need to use at least the basic CM maths fonts with them.

Proprietary fonts Various sources.
Since having a high quality font set in scalable outline form that works with \TeX{} can give a publisher a real competitive advantage, there are some publishers that have paid (a lot) to have such font sets made for them. Unfortunately, these sets are not available on the open market, despite the likelihood that they're more complete than those that are.

Mathpmtm (4 fonts) Alan Jeffrey.
This set contains maths italic, symbol, extension, and roman virtual fonts, built from Adobe Times, Symbol, Zapf Chancery, and the Computer Modern fonts. The Mathpmtm fonts are free, and the resulting PostScript files can be freely exchanged. Contains most of the CM math symbols. Support under \EPTeX{} 2e in PSNFSS (see question 60) thanks to Alan Jeffrey and Sebastian Rahtz.

(A similar development by Thomas Esser, using the Adobe Palatino fonts, is available from systems/unix/teTeX/updates/texmf/mathpmt.sh)

All of the first three font sets are available in formats suitable for IBM PC/Windows, Macintosh and Unix/NeXT from Y\&Y and from Blue Sky Research (see question 41 for details). The MathTime fonts are also available from:

\TeX{}Xplorators
1572 West Gray #377
Houston TX 77019
USA

The very limited selection of maths font sets is a direct result of the fact that a maths font has to be explicitly designed for use with \TeX{} and as a result it is likely to lose some of its appeal in other markets. Furthermore, the \TeX{} market for commercial fonts is minute (in comparison, for example, to Microsoft TrueType font pack #1, which sold something like 10 million copies in a few weeks after release of Windows 3.1!).

Text fonts in Type 1 format are available from many vendors including Adobe, Monotype, Bitstream. Avoid cheap rip-offs: not only are you rewarding unethical behaviour, destroying the cottage industry of innovative type design, but you are also very likely to get junk. The fonts may not render well (or at all under ATM), may not have the ‘standard’ complement of 228 glyphs, or may not include metric files (needed to make TFM files). Also, avoid TrueType fonts from all but the major vendors. TrueType fonts are an order of magnitude harder to ‘hint’ properly than Type 1 fonts and hence TrueType fonts from places other than Microsoft and Apple may be suspect. In any case you may find other problems with TrueType fonts such as service bureaux not accepting jobs calling for them.

65 Including a PostScript figure in \EPTeX{}
\EPTeX{} 2e (see question 121) has a standard package for graphics inclusion, rotation, colour, and other driver-related features. The package is documented in the second edition of Leslie Lamport's \EPTeX{} book (see question 19), and is available in macros/latex/required/graphics

If you don't use \EPTeX{} 2e, perhaps the best method is to use the psfig macros written by Trevor Darrell, available in graphics/psfig

You will also need a DVI to PostScript conversion program that supports the \specials. The drivers mentioned in question 42 do, and come with a version of psfig ready to use with them. The psfig macros work best with Encapsulated PostScript Files (EPS). In particular, psfig will need the file to have a BoundingBox (see Appendix H of the PostScript Language Reference Manual). If you don't have an EPS file, life can be difficult.

One point to note about including PostScript figures is that they are not part of the DVI file, but are included only when you use a DVI to PostScript conversion program. As a result, most DVI previewers will simply show the blank space \TeX{}X has reserved for your figure, not the figure itself.

There are two rather good documents on CTAN addressing figure production with rather different emphasis. Anil K. Goel's info/figsinltx.ps covers the different ways in which you might generate figures, and also the old (\EPTeX{} 2.09) ways of including them into documents. Keith Reckdahl's info/epslatex.pdf covers the standard \EPTeX{} 2e ways, as well as some of the supporting packages, notably subfigure (macros/latex/contrib/subfigure) and psfrag (macros/latex/contrib/supported/psfrag).

M Special sorts of typesetting

66 Drawing with \TeX{}

There are many packages to do pictures in \EPTeX{} itself (rather than importing graphics created externally), ranging from simple use of the \EPTeX{} picture environment, through enhancements such as epic, to sophisticated (but slow) drawing with PCTeX. Depending on your type of drawing, and setup, four systems should be at the top of your list to look at:

1. graphics/psfrag; this gives you access to all the power of PostScript from \TeX{} itself, by sophisticated use of \specials. You need a decent DVI to PostScript driver (like dvips), but the results are worth it. The well-documented package gives you not only low-level drawing commands (and full colour) such as lines, circles, shapes at arbitrary coordinates, but also
high-level macros for framing text, drawing trees and matrices, 3D effects, and more.

2. MetaPost; you liked METAFONT, but never got to grips with font files? Try MetaPost (see question 4) — all the power of METAFONT, but it generates PostScript figures. Knuth uses it for all his work...

3. Mfpic; you liked METAFONT, but can't understand the language? The package [graphics/mfpic] makes up METAFONT code for you while using familiar-looking \TeX macros. Not quite the full power of METAFONT, but a friendlier interface.

4. You liked \TeX but don't have enough memory or time? Look at Eitan Gurari's macros/generic/dratex, which is as powerful as most other \TeX drawing packages, but is an entirely new implementation, which is not as hard on memory, is much more readable (and is fully documented).

67 Double-spaced documents in \TeX

Are you producing a thesis, and trying to obey regulations that were drafted in the typewriter era? Or are you producing copy for a journal that insists on double spacing for the submitted articles?

\TeX is a typesetting system, so the appropriate design conventions are for “real books”. If your requirement is from thesis regulations, find whoever is responsible for the regulations, and try to get the wording changed to cater for typeset theses (e.g., to say “if using a typesetting system, aim to make your thesis look like a well-designed book”). If your requirement is from a journal, you’re probably even less likely to be able to get the rules changed, of course.) Bear in mind that thesis regulations are drafted to allow the examiner to add inter-linear and marginal notes; you may be able to compromise with 1.25 line spacing...

If you fail to convince your officials, or want some inter-line space for copy-editing:

- Try changing \baselinestretch:\renewcommand{\baselinestretch}{1.2} may be enough to give officials the impression you’ve kept to their regulations. Don’t try changing \baselineskip: its value is reset at any size-changing command.
- Alternatively, use a line-spacing package. The best for current is macros/latex/contrib/supported/setspace/setspace.sty

68 Formatting a thesis in \TeX

Thesis styles are usually very specific to your University, so it’s usually not profitable to ask around for a package outside your own University. Since many Universities (in their eccentric way) still require double-spacing, you may care to refer to question 67. If you want to write your own, a good place to start is the University of California style (available as macros/latex/contrib/supported/ucthesis), but it’s not worth going to a lot of trouble. (If officials won’t allow standard typographic conventions, you won’t be able to produce an aesthetically pleasing document anyway!)

69 Flowing text around figures in \TeX

There are several \TeX packages that purport to do this, but they all have their limitations because the \TeX machine isn't really designed to solve this sort of problem. Piet van Oostrum has conducted a survey of the available packages; he recommends:

\texttt{pincs\_pincs\_sty} is part of a large package [systems/msdos/picins/picins.zip] that allows inclusion of pictures (e.g., with shadow boxes, various MS-DOS formats, etc.). The command is:

\texttt{\begin{wrapfigure}\[\text{figure contents}\]\end{wrapfigure}}

There is a (more or less similar) \texttt{floatingtable} environment.

The tables or figures can be set left or right, or alternating on verso/recto pages in a double-sided document.

The package works with the \texttt{multicol} package, but doesn’t work well in the neighbourhood of list environments (unless you change your \TeX document).

\texttt{\begin{wrapfigure}\[\text{figure, caption, etc.}\]\end{wrapfigure}}

The syntax of the \texttt{wraptable} environment is similar.

Height can be omitted, in which case it will be calculated by the package; the package will use the greater of the specified and the actual width. The \texttt{[\text{line}]} parameter can also be specified as \texttt{[\text{inside}]} or \texttt{[\text{outside}]} for two-sided documents, and uppercase can be used to indicate that the picture should float. The overhang allows the figure to be moved into the margin. The figure or table will entered into the list of figures or tables if you use the \texttt{\caption} command.

The environments do not work within list environments that end before the figure or table has finished, but can be used in a parbox or minipage, and in two-column format.
70 Alternative head- and footlines in LATEX

The standard LATEX document classes define a small set of 'page styles' which (in effect) specify head- and footlines for your document. The set defined is very restricted, but LATEX is capable of much more; people occasionally set about employing LATEX facilities to do the job, but that's quite unnecessary — Piet van Oostrum has already done the work.

The package is found in directory macros/latex/contrib/supported/fancyhdr and provides simple mechanisms for defining pretty much every head- or footline variant you could want; the directory also contains some (rather good) documentation and one or two smaller packages. Fancyhdr also deals with the tedious behaviour of the standard styles with initial pages (see question 103), by enabling you to define different page styles for initial and for body pages.

71 Including a file in verbatim in LATEX

A good way is to use Rainer Schöpf's verbatim.sty, which provides a command \verbatiminput{} that takes a file name as argument. This package is available as part of macros/latex/required/tools

Another way is to use the alltt environment, which requires alltt.sty (which is now part of \verbatim). alltt interprets its contents 'mostly' verbatim, but executes any TEX commands it finds: so one can say:

\begin{alltt}
\input{verb.txt}
\end{alltt}

of course, this is little use for inputting (\verbatim) source code...

The moreverb package (macros/latex/contrib/supported/moreverb) extends the facilities of verbatim package, providing a listing environment and a \verbatiminput{} command, which line-number the text of the file.

The fancyvrb package (macros/latex/contrib/supported/fancyvrb) offers configurable implementations of everything verbatim and moreverb have, and more besides. It is nowadays the package of choice for the discerning typesetter of verbatim text, but its wealth of facilities makes it a complex beast and study of the documentation is strongly advised.

72 Including line numbers in typeset output

For general numbering of lines, there are two packages for use with LATEX, macros/latex/contrib/supported/lineno (which permits labels attached to individual lines of typeset output) and macros/latex/contrib/supported/numline/numline.sty

Both of these packages play fast and loose with the LATEX output routine, which can cause problems: the user should beware...

If the requirement is for numbering verbatim text, the macros/latex/contrib/supported/moreverb or macros/latex/contrib/supported/fancyvrb packages (see question 71) may be used.

One common use of line numbers is in critical editions of texts, and for this the edmac package (macros/plain/contrib/edmac) offers comprehensive support.

73 Generating an index in (B)\TeX

Making an index is not trivial; what to index, and how to index it, is difficult to decide, and uniform implementation is difficult to achieve. You will need to mark all items to be indexed in your text (typically with \index commands).

It is not practical to sort a large index within TeX, so a post-processing program is used to sort the output of one \TeX run, to be included into the document at the next run.

The following programs are available:

makeindex for LATEX under Unix (but runs under other OSs without changes). Available in indexing/makeindex; a version for the Macintosh is available as systems/mac/macmakeindex.sit, and ones for MS-DOS are part of the cm\TeX and g\TeX distributions (the \TeX version also runs under OS/2).

The Makeindex documentation is a good source of information on how to create your own index. Makeindex can be used with \TeX macro packages other than LATEX, such as Eplain (see question 11), and \TeXs, nonfree/macros/texsis/index/index.tex, can be used independently with plain).

idxtex for LATEX under VMS. Available (together with a glossary-maker called glotex in indexing/glotex/idxtex

texindex A witty little shell/script-based utility for LATEX under Unix. Available from support/texindex

There are other programs called texindex, notably one that comes with the Texinfo distribution (see question 13).

xindy a recent development, designed for wide-ranging flexibility (including support for multilingual indexes), based on Common Lisp. The system is available on CTAN (support/xindy), but is more easily accessed from a web browser via http://www.iti.informatik.th-darmstadt.de/xindy/ since the distribution contains several different implementations.

74 Typesetting URLs

URLs tend to be very long, and contain characters that would naturally prevent them being hyphenated even if they weren't typically set in \ttfamily, verbatim. Therefore, without special treatment, they often produce wildly overfull \hboxes, and their typeset representation is awful.

There are two approaches to this problem:

- macros/latex/contrib/other/misc/path.sty, which defines a \path{} command. The command...
defines each potential break character as a \discretionary, and offers the user the opportunity of specifying a personal list of potential break characters. Its chief disadvantage is fragility in the \End{T\LaTeX} context.

* macros/latex/contrib/other/misc/url.sty, which defines an \url command (among others, including its own \path command). The command gives each potential break character a math-mode ‘personality’, and then sets the URL itself in the user’s choice of font, in math mode. It can produce \End{T\LaTeX}-style ‘robust’ commands (see question 115) for use within moving arguments. Note that, because the operation is conducted in math mode, spaces within the URL argument are ignored unless special steps are taken.

The author of this answer prefers the (rather newer) url.sty; both packages work equally well with plain \T\LaTeX (though of course, the fancy \End{T\LaTeX} facilities of url.sty don’t have much place there).

75 Citing URLs with \Bib{T\LaTeX}

There is no citation type for URLs, per se, in the standard \Bib{T\LaTeX} styles, though Oren Patashnik (the author of \Bib{T\LaTeX}) is considering developing one such for use with the long-awaited \Bib{T\LaTeX} version 1.0.

The actual information that need be available in a citation of an URL is discussed at some length in the publicly available on-line extracts of ISO 690–2, available via http://www.nlc-bnc.ca/iso/tc46sc9/standard/690-2e.htm; the techniques below do not satisfy all the requirements of ISO 690–2, but they offer a solution that is at least available to users of today’s tools.

Until the new version arrives, the simplest technique is to use the howpublished field of the standard styles’ \@misc function. Of course, the structures about typesetting URLs (see question 74) still apply, so the entry will look like:

\begin{verbatim}
@misc{..., 
    howpublished = "\url{http://...}" 
}
\end{verbatim}

Another possibility is that some conventionally-published paper, technical report (or even book) is also available on the Web. In such cases, a useful technique is something like:

\begin{verbatim}
@techreport{..., 
    note = "Also available as \url{http://...}" 
}
\end{verbatim}

There is good reason to use macros/latex/contrib/other/misc/url.sty in this context, since (by default) it ignores spaces in its argument. \Bib{T\LaTeX} has a habit of splitting lines it considers excessively long, and if there are no space characters for it to use as ‘natural’ breakpoints, \Bib{T\LaTeX} will insert a comment (%) character ... which is an acceptable character in an URL, so that \url will typeset it. The way around the problem is to insert odd spaces inside the URL itself in the \.bib file, to enable \Bib{T\LaTeX} to make reasonable decisions about breaking the line.

76 Using \Bib{T\LaTeX} with plain \T\LaTeX

The file macros/plain/biblatex/biblatex.tex contains macros and documentation for using \Bib{T\LaTeX} with \plain{T\LaTeX}, either directly or with Eplain (see question 11). See question 23 for more information about \Bib{T\LaTeX} itself.

77 Typesetting music in \T\LaTeX

A powerful package which allows the typesetting of polyphonic and other multiple-stave music is Music\LaTeX, written by Daniel Taupin (taupin@rosvox.lps.upsud.fr). It is available in macros/musicxtex.

In the recent past, Daniel (as well as with various other people, notably Ross Mitchell and Andreas Egler) have been working on a development of Music\LaTeX, known as MusiX\LaTeX. MusiX\LaTeX is a three-pass system (with a processor program that computes values for the element spacing in the music), and achieves finer control than is possible in the unmodified \LaTeX-based mechanism that Music\LaTeX uses. Daniel Taupin and Andreas Egler are pursuing distinct versions of MusiX\LaTeX; they are available, respectively, from macros/musicxtex/taupin and macros/musicxtex/egler.

Digital music fans can typeset notation for their efforts by using \texttt{midi2tex}, which translates MIDI data files into Music\LaTeX source code. It is available from support/midi2tex.

A rather simpler notation than Music\LaTeX is supported by \texttt{abc2mtex}; this is a package designed to notate tunes stored in an ASCII format (\abc notation). It was designed primarily for folk and traditional tunes of Western European origin (such as Irish, English and Scottish) which can be written on one stave in standard classical notation. However, it should be extensible to many other types of music. It is available from support/abc2mtex.

There is a mailing list for discussion of typesetting music in \LaTeX. To subscribe, send mail to mutex-request@stolaf.edu containing the word ‘subscribe’ in the body.

78 Drawing Feynman diagrams in \IsiT\LaTeX

Michael Levine’s macro package for drawing Feynman diagrams in \IsiT\LaTeX is available in macros/latex209/contrib/feynman.

Another possibility is Thorsten Ohl’s macros/latex/contrib/supported/feynmf, that works in combination with \texttt{METAFONT} (or MetaPost). The \texttt{feynmf} or \texttt{feynmf} package reads a description of the diagram written in \LaTeX, and writes out code. \texttt{METAFONT} (or MetaPost) can then produce a font (or PostScript file) for use in a subsequent \IsiT\LaTeX run.

For new users who have access to MetaPost, the PostScript version is probably the better route, for document portability and other reasons.

N How do I do X in \LaTeX or \IsiT\LaTeX?

79 Proof environment

It is not possible to make a \texttt{proof} environment which automatically includes an ‘end-of-proof’ symbol. Some proofs
end in displayed maths; others do not. If the input file contains \end{proof} then \LaTeX finishes off the displayed maths and gets ready for a new line before it reads any instructions connected with ending the proof. But traditionally the end-of-proof sign goes in the display, not on a new line. So you just have to put it in by hand in every proof.

### 80 Symbols for the number sets

It is a good idea to have commands such as \texttt{\textbackslash \textit{R}} for the real numbers and other standard number sets. Traditionally these were typeset in bold. Because mathematicians usually do not have access to bold chalk, they invented the special symbols that are now often used for \texttt{\textbackslash \textit{R}}, \texttt{\textbackslash \textit{C}}, \texttt{\textbackslash \textit{Q}}, \texttt{\textbackslash \textit{N}}, etc. These symbols are known as “blackboard bold”. Before insisting on using them, consider whether going back to the old system of ordinary bold might not be acceptable (it is certainly simpler).

A set of blackboard bold capitals is available in the AMS fonts \texttt{\textit{msam}} (e.g., \texttt{\textit{msam10}} for 10pt) and \texttt{\textit{msbm}}. The fonts have a large number of mathematical symbols to supplement the ones in the standard \LaTeX distribution. The fonts are available in \texttt{fonts/ams/amsfonts/sources/symbols}.

Two files which load the fonts and define the symbols are provided, and both work with either \LaTeX or \LaTeXe. Questions or suggestions regarding these fonts should be directed to tech-support@math.ams.org.

Another complete set of blackboard bold fonts, the \texttt{bbold} family, is available in METAFONT (in \texttt{fonts/bbold}). This set has the interesting property of offering blackboard bold forms of lower-case letters, something rather rarely seen on actual blackboards.

The “lazy person’s” blackboard bold macros:

\begin{verbatim}
\newcommand{\R}{\{\sf R\}}
\newcommand{\N}{\{\sf N\}}
\newcommand{\Q}{\{\sf Q\}}
\newcommand{\Z}{\{\sf Z\}}
\newcommand{\C}{\{\sf C\}}
\end{verbatim}

work well at normal size if the surrounding text is \texttt{cmr10}. However, they are not part of a proper maths font, and so do not work in sub- and superscripts. Moreover, the size and position of the vertical bar is affected by the font of the surrounding text.

### 81 Roman theorems

If you want to take advantage of the powerful \newtheorem command without the constraint that the contents of the theorem is in a sloped font (for example, to use it to create remarks, examples, proofs, ...) then you can use the style file \texttt{theoremm.sty} (part of \texttt{macros/latex/required/tools}). Alternatively, the following creates an environment \texttt{remark} whose content is in roman.

```
\newtheorem{preremark}{Remark}
\newenvironment{remark}{}
{\begin{preremark}\rm}{\end{preremark}}
```

This will not work if you are using NFSS (see question 120) outside of \LaTeXe (see question 121), because the command \texttt{\textbackslash \textit{rm}} behaves differently there.

### 82 Fancy enumeration lists

Suppose you want your top-level \texttt{enum}erates to be labelled ‘I/’, ‘II/’, ...; you give these commands:

```
\renewcommand{\theenumi}{\roman{enumi}}
\renewcommand{\labelenumi}{\theenumi}
```

The possible styles of numbering are given in Section 6.3 of Lamport’s book (see question 19). Both \texttt{\theenumi} and \texttt{\labelenumi} must be changed, since \texttt{\theenumi} is used in cross-references to the list.

For lower level \texttt{enum}erates, replace \texttt{enumi} by \texttt{enumii}, \texttt{enumiii} or \texttt{enumiv}, according to the level. If your label is much larger than the default, you should also change \texttt{\leftmargini}, \texttt{\leftmarginii}, etc.

If you’re running \LaTeXe, the package \texttt{enumerate.sty} (part of \texttt{macros/latex/required/tools}) offers similar facilities. With \texttt{enumerate.sty}, the example above would be achieved simply by starting the enumeration \texttt{\begin{enumerate}[I/]}.

### 83 Unnumbered sections in the Table of Contents

The easiest way to get headings of funny ‘sections’ such as prefaces in the table of contents is to use the counter \texttt{secnumdepth} described in Appendix C of the \LaTeX manual. For example:

```
\setcounter{secnumdepth}{-1}
\chapter{Preface}
```

Of course, you have to set \texttt{secnumdepth} back to its usual value (which is 2 in the standard styles) before you do any ‘section’ which you want to be numbered.

Similar settings are made automatically in the \LaTeX book class by the \texttt{\frontmatter} and \texttt{\backmatter} commands. This is why it works. \texttt{\chapter} without the star does

1. put something in the \texttt{.toc} file;
2. if \texttt{secnumdepth} \texttt{\geq} 0, increase the counter for the chapter and write it out.
3. write the chapter title.

Other sectioning commands are similar, but with other values used in the text.

### 84 Footnotes in tables

The standard \LaTeX \texttt{\footnote} command doesn’t work in tables; the table traps the footnotes and they can’t escape to the bottom of the page.

If your table is floating, your best bet is (unfortunately) to put the table in a \texttt{minipage} environment and to put the notes underneath the table, or to use Donald Arseneau’s package \texttt{macros/latex/contrib/other/misc/threeparttable.sty}.
Otherwise, if your table is not floating (it’s just a ‘tabular’ in the middle of some text), there are several things you can do to fix the problem.

1. Use \footnotemark to position the little marker appropriately, and then put in \footnotetext commands to fill in the text once you’ve closed the tabular environment. This is described in Lamport’s book, but it gets messy if there’s more than one footnote.

2. Stick the table in a minipage anyway. This provides all the ugliness of footnotes in a minipage with no extra effort.

3. Use \textbf{macros/latex/contrib/other/misc/threeparttable.sty} anyway; the package is intended for floating tables, and the result might look odd if the table is not floating, but it will be reasonable.

4. Use \texttt{tabularx} or \texttt{longtable} from the \texttt{LATEX} tools distribution (\texttt{macros/latex/required/tools}); they’re noticeably less efficient than the standard \texttt{tabular} environment, but they do allow footnotes.

5. Grab hold of \texttt{footnote.sty} from CTAN, lurking in \texttt{macros/latex/contrib/supported/mdwtools}.

Then put your tabular environment inside a \texttt{savenotes} environment. Alternatively, say \texttt{\makesavenoteenv{tabular}} in the preamble of your document, and tables will handle footnotes correctly.

6. Use \texttt{mdvtab.sty} from the same directory (\texttt{macros/latex/contrib/supported/mdwtools}).

This will handle footnotes properly, and has other facilities to increase the beauty of your tables. It may also cause other table-related packages (not the standard ‘tools’ ones, though) to become very unhappy and stop working.

### 85 Style of section headings

Suppose that the editor of your favourite journal has specified that section headings must be centred, in small capitals, and subsection headings ragged right in italic, but that you don’t want to get involved in the sort of programming described in The LATEX Companion (see question 19; the programming itself is discussed in question 109). The following hack will probably satisfy your editor. Define yourself new commands

\begin{verbatim}
\newcommand\section[1]{\centering\sc #1}
\newcommand\subsection[1]{\raggedright\it #1}
\end{verbatim}

and then use \texttt{\section} and \texttt{\subsection} in place of \texttt{section} and \texttt{subsection}. This isn’t perfect: section numbers remain in bold, and starred forms need a separate redifinition. Also, this will not work if you are using NFFS (see question 120) outside of LATEX2e (see question 121), because the font-changing commands behave differently there.

### 86 Indent after section headings

LATEX implements a style that doesn’t indent the first paragraph after a section heading. There are coherent reasons for this, but not everyone likes it. The package \texttt{indentfirst.sty} (part of \texttt{macros/latex/required/tools}) suppresses the mechanism, so that the first paragraph is indented.

### 87 Footnotes in LATEX section headings

The \texttt{footnote} command is fragile, so that simply placing the command in \texttt{section}’s arguments isn’t satisfactory. Using \texttt{\protect\footnote} isn’t a good idea either: the arguments of a section command are used in the table of contents and (more dangerously) potentially also in page headings. Unfortunately, there’s no mechanism to suppress the footnote in the heading while allowing it in the table of contents, though having footnotes in the table of contents is probably unsatisfactory anyway.

To suppress the footnote in headings and table of contents:

- Take advantage of the fact that the mandatory argument doesn’t ‘move’ if the optional argument is present: \texttt{\section{title}{title\footnote{title footnote}}}
- Use the \texttt{\footnote.sty} package, with package option \texttt{stable} — this modifies footnotes so that they softly and silently vanish away if used in a moving argument.

### 88 Changing the margins in LATEX

Don’t do it. Learn some LATEX, produce some documents, and then ask again.

You can never change the \texttt{margins} of a document by software, because they depend on the actual size of the paper. What you can change are the distances from the apparent top and left edges of the paper, and the width and height of the text. Changing the last two requires more skill than you might expect. The height should bear a certain relationship to \texttt{baselineskip}. And the width should not be more than 75 characters. Lamport’s warning in his section on ‘Customizing the Style’ really must be taken seriously. One-inch margins on A4 paper are fine for 10- or 12-pitch typewriters, but not for 10pt type (or even 11pt or 12pt) because so many characters per line will irritate the reader. However...

Perhaps the easiest way to get more out of a page in LATEX is to get \texttt{macros/latex209/contrib/misc/fullpage.sty}, which sets the margins of the page identical to those of \texttt{plain \te}, i.e., 1-inch margins at all four sides of the paper. It also contains an adjustment for A4 paper.

Somewhat more flexible is \texttt{macros/latex/contrib/other/misc/vmargin.sty}, which has a canned set of paper sizes (a superset of that provided in \texttt{LATEX2e}), provision for custom paper, margin adjustments and provision for two-sided printing.

For details of LATEX’s page parameters, see section \texttt{C.5.3} of the LATEX manual (pp. 181–182). The origin in DVI coordinates is one inch from the top of the paper and one inch from the left side; positive horizontal measurements extend
right down the page, and positive vertical measurements extend down the page. Thus, for margins closer to the left and top edges of the page than 1 inch, the corresponding parameters, i.e., \texttt{\vsize\hspace{\oddsidemargin} \hoffset\hspace{\evensidemargin}, \texttt{\topmargin}}, can be set to negative values.

You cannot simply change the margins of part of a document within the document by modifying the parameters shown in Lamport’s figure C.3. They should be changed only in the preamble of the document, i.e., before the \begin{document} statement. To adjust the margins within a document we define an environment:

\begin{verbatim}
\newenvironment{changemargin}[2]{%  \begin{list}{}{\setlength{\topsep}{0pt} \setlength{\parsep}{\parskip}\setlength{\listparindent}{\parindent}\setlength{\leftmargin}{#1}\setlength{\rightmargin}{#2}\setlength{\itemindent}{\parindent}\setlength{\parskip}{\parskip}\setlength{\topsep}{0pt} \setlength{\partopsep}{0pt}} \item[]\{\end{list}}
\end{verbatim}

This environment takes two arguments, and will indent the left and right margins by their values, respectively. Negative values will cause the margins to be narrowed, so \begin{changemargin}{-1cm}{-1cm} narrows the left and right margins by 1cm.

89 Finding the width of a letter, word, or phrase

Put the word in a box, and measure the width of the box. For example,

\begin{verbatim}
\settowidth{\gnat}{hi} \setlength{\stringwidth}{\wd0}
\end{verbatim}

Note that if the quantity in the \hbox is a phrase, the actual measurement only approximates the width that the phrase will occupy in running text, since the inter-word glue can be adjusted in paragraph mode.

The same sort of thing is expressed in \LaTeX by:

\begin{verbatim}
\settowidth{\gnat}{\textbf{small}} \setlength{\stringwidth}{\textwidth}
\end{verbatim}

This sets the value of the length command \texttt{\gnat} to the width of “small” in bold-face text.

90 Changing the space between letters

A common technique in advertising copy (and other text whose actual content need not actually be read) is to alter the space between the letters (otherwise known as the tracking). As a general rule, this is a Very Bad Idea: it detracts from legibility, which is contrary to the principles of typesetting (any respectable font you might be using should already have optimum tracking built into it).

The great type designer, Eric Gill, is (possibly apocryphally) credited with saying “he who would letterspace lower-case text, would steal sheep” (stealing sheep was, before Gill’s time, a capital offence in Britain). As the remark suggests, though, letterspacing of upper-case text is less awful a crime; the technique used also to be used for emphasis of text set in gothic (or similar) fonts and is still used in German where it is referred to as \texttt{speratsch}.

Straightforward macros (usable, in principle, with any \LaTeX macro package) may be found in macros/generic/letterspacing.tex

A more comprehensive package is macros/latex/contrib/supported/soul (which is optimised for use with \LaTeX, but also works with Plain \TeX). Soul also permits hyphenation of letterspaced text; Gill’s view of such an activity is not (even apocryphally) recorded.

91 Excluding blocks of text from the DVI file

Rainer Schöpf’s \verb|verbatim|.sty provides a comment environment which excludes everything between \begin{verbatim} and \end{verbatim}. This package is available as part of macros/latex/required/tools.

A more general environment for doing the job is Victor Eijkhout’s \verb|comment|.sty, which lets you define environments for inclusion or exclusion in a document, thus offering a primitive configuration structure. It is available from the CTAN sites in macros/latex209/contrib/misc/comment.sty.

92 Setting bold Greek letters in \LaTeX

The simple solution \verb|\mathbf\theta| doesn't always work, because lower-case Greek letters behave differently from upper-case Greek letters (due to Knuth’s esoteric font encoding decisions). However, \verb|\mathbf\theta| can be used for upper-case Greek letters in ordinary circumstances, but the \verb|\AMS-L\TeX| package \verb|amsmath.sty| disables this font-switching and you must use one of the techniques outlined below.

The plain \LaTeX solution does work, in a limited way:

\begin{verbatim}
{\boldmath\theta}$
\end{verbatim}

but \verb|\boldmath| may not be used in maths mode, so this ‘solution’ requires arcania such as:

\begin{verbatim}
$\ldots \boxed{{\boldmath\theta}} \ldots$
\end{verbatim}

which then causes problems in superscripts, etc.

These problems may be addressed by using a bold mathematics package.

- The package \verb|bm.sty|, which is part of the \LaTeX tools distribution \verb|macros/latex/required/tools|, defines a command \verb|\bm| which may be used anywhere in maths mode.

- The package \verb|amsbsy.sty|, which is part of \verb|\AMS-L\TeX| defines a command \verb|\boldsymbol|, which (though slightly less comprehensive than \verb|\bm|) covers almost all common cases.

All these solutions cover all mathematical symbols, not merely Greek letters.
93 Defining a new log-like function in \LaTeX

Use the \texttt{\mathop} command, as in:

\begin{verbatim}
newcommand\{\diag\}{\mathop\{\rm diag\}}
\end{verbatim}

Subscripts and superscripts on \texttt{\diag} will be placed exactly as they are on \texttt{\im}. If you want your subscripts and superscripts always placed to the right, do:

\begin{verbatim}
newcommand\{\diag\}{\mathop\{\rm diag\}\nolimits}
\end{verbatim}

This works in \LaTeX{} 2.09 and in \LaTeX{} 2\epsilon{}, but not under \TeX{} 3\epsilon{} alone (see question 105). However, the canonical method for doing this in \LaTeX{} 2\epsilon{} is to use the \texttt{\DeclareMathOperator} command of \texttt{amsopn.sty} (which is part of the \texttt{AMS-L}\LaTeX{} package: \texttt{macros/latex/required/amslatex}).

(\textit{It should be noted that “log-like” was reportedly a \textit{joke} on Lamport’s part; it is of course clear what was meant.})

94 Typesetting all those \TeX{}-related logos

Knuth was making a particular point about the capabilities of \TeX{} when he defined the logo. Unfortunately, many believe, he thereby opened floodgates to give the world a whole range of rather silly ‘bumpy road’ logos such as \texttt{\AMS-\TeX{}}, \texttt{\Pic\TeX{}}, \texttt{\Bib\TeX{}} and so on, produced in a flurry of different fonts, sizes, and baselines — indeed, everything one might believe would cause them to obstruct the reading process. In particular, Lamport invented \LaTeX{} (silly enough in itself) and marketing input from Addison-Wesley led to the even stranger current logo \LaTeX{} 2\epsilon{}.

Sensible users don’t have to follow this stuff wherever it goes, but, for those who insist, a large collection of logos is defined in \texttt{macros/eplain/texnames.sty} (but note that this set of macros isn’t entirely reliable in \LaTeX{} 2\epsilon{}).

The \texttt{METAFONT} and \texttt{MetaPost} logos can be set in fonts that \LaTeX{} 2\epsilon{} knows about (so that they scale with the surrounding text) using the package \texttt{macros/latex/contrib/supported/mflogo}; but be aware that booby-traps surround the use of the Knuthian font for MetaPost (you might get \texttt{METR O T}). You needn’t despair, however — the author himself uses just ‘MetaPost’.

For those who don’t wish to acquire the ‘proper’ logos, the canonical thing to do is to say \texttt{AMS-\TeX{}} (\texttt{AMS-\TeX{}}) for \texttt{\AMS-\TeX{}}, \texttt{\Pic\TeX{}} (\texttt{\Pic\TeX{}}) for \texttt{\Pic\TeX{}}, \texttt{\Bib\TeX{}} (\texttt{\Bib\TeX{}}) for \texttt{\Bib\TeX{}} and so on.

While the author of this FAQ list can’t quite bring himself to do away with the bumpy-road logos herein, he regularly advises everyone else to...

O Macros for particular types of documents

95 Setting papers for journals

Publishers of journals have a wide range of requirements for the presentation of papers, and while many publishers do accept electronic submissions in \LaTeX{}, they often don’t submit their recommended macros to public archives.

Nevertheless, there are considerable numbers of macros of one sort or another available on CTAN; searching for your journal name in the CTAN catalogue (http://www.tex.ac.uk/tex-archive/help/Catalogue/catalogue.html) may well turn up what you’re seeking.

Failing that, you may be well advised to contact the prospective publisher of your paper; many publishers have macros on their own web sites, or otherwise available only upon application.

Check that the publisher is offering you macros suitable for an environment you can use: a few still have no macros for current \LaTeX{}, for example, claiming that \LaTeX{} 2.09 is good enough...

Some publishers re-key anything sent to them anyway, so that it doesn’t really matter which macros you use. Others merely encourage you to use as few extensions of a standard package as possible, so that they will find it easy to transform your paper to their own internal form.

96 A ‘report’ from lots of ‘article’s

This is a requirement, for example, if one is preparing the proceedings of a conference whose papers were submitted in \LaTeX{}.

There’s no canned solution, but Matt Swift’s \texttt{include} and \texttt{moredefs} packages (both part of the \texttt{macros/latex/contrib/supported/frankenstein} bundle) offer a possible way forward.

\texttt{include} enables you to ‘\texttt{\includedoc}’ complete articles (in the way that you ‘\texttt{\include}’ chapter files in an ordinary report). It doesn’t do the whole job for you, though. You need to analyse the package use of the individual papers, and ensure that a consistent set is loaded in the preamble of the main report.

Another tedious problem to address is that \LaTeX{} suppresses all title-related commands after \texttt{\maketitle}. There’s no canned solution, but Matt Swift’s \texttt{\maketitle} has been used once in a document: thus you need to define replacements for all of them, for use in the papers.

More work is plainly needed in this area, but at least a start has been made.

97 Curriculum Vitae (Resumé)

A framework class for \textit{Curricula Vitae}, provided by Andrej Brodnik, is available from \texttt{macros/latex/contrib/other/vita}.

The class can be customised both for subject (example class option files are offered for both computer scientists and singers), and for language (both options provided are available for both English and Slovene). Extensions may be written by creating new class option files, or by using macros defined in the class to define new entry types, etc.

P Things are going wrong...

98 Weird hyphenation of words

You may have a version mismatch problem. \TeX{}’s hyphenation system changed between version 2.9 and 3.0. If you are using \texttt{\plain} \TeX{} version 3.0 or later (you \textit{should} be us-
If for some reason you can't, the last version of LATEX 2.09, December 1995 release of LATEX2\newline latex209.tar and ought to solve this problem. At least) from released on 25 March 1992, is available (for the time being in the font release specified use of the alternative hyphen character, thus removing this source of problems. A multi-lingual version.

For the curious, here's what happened: before T\TeX 3.0, the hyphenation algorithm would not break a word if the part before the break was not at least two characters long and the part after the break at least three characters long. Starting with version 3.0, the parameters `leftthyphenmin` and `righththyphenmin` control the length of these fragments. These are set to 2 and 3, respectively, in the new `plain` and `1plain` formats. They can be set to any value, of course, but if `leftthyphenmin`>`righththyphenmin` is greater than 62, all hyphenation is suppressed.

A further source of oddity can derive from the 1995 release of Cork-encoded fonts (see question 34), which introduced an alternative hyphen character. The \LaTeX2e configuration files in the font release specified use of the alternative hyphen, and this could produce odd effects with words containing an explicit hyphen. The font configuration files in the December 1995 release of \LaTeX2e do not use the alternative hyphen character, thus removing this source of problems.

99 (Merely) peculiar hyphenation

You may have found that T\TeX's famed automatic word-division does not produce the break-points recommended by your dictionary. This may be because \LaTeX is set up for American English, whose rules for word division (as specified, for example, in Webster's Dictionary) are completely different from the British ones (as specified, for example, in the Oxford Dictionaries). This problem is being addressed by the UK \LaTeX User community (see Bekvemelle, issue 4.4) but an entirely satisfactory solution will take time. An interim hyphenation file is available in `language/hyphenation/ukhyphen.tex`.

100 Accented words aren't hyphenated

\LaTeX's algorithm for hyphenation gives up when it encounters an `accent` command; there are good reasons for this, but it means that quality typesetting in non-English languages can be difficult.

For \LaTeX itself, avoiding this effect means using Cork-encoded fonts (see question 34) which contain accented letters as single glyphs. In the future, perhaps, Omega (see question 123) will provide a rather different solution.

101 Enlarging \LaTeX

People sometimes get messages saying 'memory capacity exceeded'. Most of the time this error can be fixed without enlarging \LaTeX. The most common causes are unmatched braces, extra-long lines, and poorly-written macros. Extra-long lines are often introduced when files are transferred in-

\begin{itemize}
  \item Are the placement parameters on your figures right? The default (`tbp`) is reasonable; you should never simply say `h`, for example, since that says "if it can't go here, it can't go anywhere", and as a result all subsequent floats pile up behind it.
  \item Can you perhaps prevent your figures from floating by adjusting \LaTeX's placement parameters? Again, the defaults are reasonable, but can be overridden in case of problems. The parameters are described on pages 199–200, section C.9 of the \LaTeX manual.
  \item Are there places in your document where you could 'naturally' put a `\clearpage` command? If so, do: the backlog of floats is cleared after a `\clearpage`. (Note that the `\chapter` command implicitly executes `\clearpage`, so you can't float past the end of a chapter.)
  \item Have a look at the `\LaTeX2e afterpage` package (part of macros/latex/required/tools). Its documentation gives as an example the idea of putting `\clearpage` after the current page (where it will clear the backlog, but not cause an ugly gap in your text), but also admits that the package is somewhat fragile (though it's improving).
  \item As a last resort, try the package macros/latex209/contrib/misc/morefloats.sty; this 'simply' increases the number of floating inserts that \LaTeX can handle at one time (from 18 to 36), but that may suit your needs.
  \item If you actually wanted all your figures to float to the end of the chapter,\texttt{\textbackslash clearpage}\textbf{\textbackslash chapter} is closer to what you might need.
\end{itemize}
end (e.g., for submitting a draft copy of a paper), don't rely on \texttt{LATEX}'s mechanism: get the package \texttt{macros/latex/contrib/support/endedfloat} to do the job for you.

103 \texttt{\textbackslash pagestyle{empty}} on first page in \texttt{LATEX}

If you use \texttt{\pagestyle{empty}}, but the first page is numbered anyway, you are probably using the \texttt{\textbackslash maketitle} command too. This is not a bug but a feature! The standard \texttt{LATEX} styles are written so that initial pages (pages containing a \texttt{\maketitle}, \texttt{\part}, or \texttt{\chapter}) have a different page style from the rest of the document; Hence, the commands internally issue \texttt{\thispagestyle{plain}}. This is usually not acceptable behaviour if the surrounding page style is 'empty'.

Possible workarounds include:

- Put \texttt{\thispagestyle{empty}} immediately after the \texttt{\maketitle} command, with no blank line between them.
- Use \texttt{fancyhdr.sty}, which allows you to customise the style for initial pages independently of that for body pages. It is available in \texttt{macros/latex/contrib/support/fancyhdr}
- Use \texttt{nopageno.sty}, which suppresses this behaviour. It is available in \texttt{macros/latex/contrib/support/carlisle/nopageno.sty}

104 Underlined text won't break

Knuth made no provision for underlining text: he took the view that underlining is not a typesetting operation, but rather one that provides emphasis on typewriters, which typically offer but one typeface. The corresponding technique in typeset text is to switch from upright to italic text (or vice-versa); the \texttt{LATEX} command \texttt{\textbf{emph}} does just that to its argument.

Nevertheless, typographically illiterate people (such as those that specify double-spaced thesis styles — see question 67) or those that choose to \textit{deliberately} flout convention (after very careful consideration, one hopes) continue to require underlining of us, so \texttt{LATEX} as distributed defines an \texttt{\underline{underline}} command that applies the mathematical 'underbar' operation to text. This technique is not entirely satisfactory, however: the text gets stuck into a box, and won't break at line end.

The solution is the package \texttt{macros/latex/contrib/other/misc/ulem.sty}, which redefines the \texttt{\textbf{emph}} command to underline its argument; the underlined text thus produced behaves as ordinary emphasised text, and will break over the end of a line. (The package is capable of other peculiar effects, too: read its documentation, contained within the file itself.)

105 Odd behaviour of \texttt{\textbackslash rm}, \texttt{\textbackslash bf}, etc.

If commands such as \texttt{\textbackslash rm} and \texttt{\textbackslash bf} have suddenly stopped working in \texttt{LATEX} in the way that you expect, it is likely that your system administrator has installed a version of \texttt{LATEX 2.09} with NFSS (see question 120). Complain loudly, ask your system administrator to replace this version with \texttt{LATEX 2.09}, in which commands such as \texttt{\textbackslash rm} and \texttt{\textbackslash bf} work just as before if you are using one of the standard classes — \texttt{article}, \texttt{report} and \texttt{book} (among others). In the meantime, use the option \texttt{oldlfont.sty}, which should have been installed at the same time as NFSS.

106 Old \texttt{LATEX} font references such as \texttt{\textbackslash tenrm}

\texttt{LATEX 2.09} defined a large set of commands for access to the fonts that it had built in to itself. For example, various flavours of \texttt{cmr} could be found as \texttt{\fiverm}, \texttt{\sixrm}, \texttt{\sevrm}, \texttt{\egtrm}, \texttt{\frtnrm}, \texttt{\svtnrm}, \texttt{\twlrm}, \texttt{\twtlyrm} and \texttt{\twsvrm}. These commands were never documented, but certain packages nevertheless used them to achieve effects they needed.

Since the commands weren't public, they weren't included in \texttt{LATEX 2.09}; to use the unconverted \texttt{LATEX 2.09} packages under \texttt{LATEX 2\epsilon}, you need also to include the package \texttt{rawfonts.sty} (which is part of the \texttt{LATEX 2\epsilon} distribution).

107 Missing symbols

If some symbols, such as \texttt{\Box} and \texttt{\H}, no longer appear to exist, then your system administrator has probably upgraded your version of \texttt{LATEX} to either NFSS (see question 120) or \texttt{LATEX 2\epsilon} (see question 121). In the former case, use \texttt{oldlfont.sty}, as in the question 105. In the latter, use the package \texttt{latexsym}, which is part of the \texttt{LATEX 2\epsilon} distribution, or the package \texttt{amsfonts}, if it is available.

108 \texttt{LATEX} gets cross-references wrong

Sometimes, however many times you run \texttt{LATEX}, the cross-references are just wrong. Remember that the \texttt{\label} command must come \textit{after} the \texttt{\caption} command, or be part of it. For example,

\begin{verbatim}
\begin{figure} \begin{figure} \caption{A Figure} \caption{A Figure}\% \\
\label{fig} \label{fig} \end{figure} \end{figure}
\end{verbatim}

109 \texttt{\@} and \texttt{\@} in macro names

A common source of problems in a \texttt{LATEX} document is the diagnostic about the appearance of the command \texttt{\@}, or about other commands containing the character \texttt{\@}. The most common complaint is "You can't use "\texttt{\@spacefactor} in vertical mode", but others occur.

Such problems are usually caused by including a \texttt{LATEX 2\epsilon} class or package file into a \texttt{LATEX} document by some means other than \texttt{\documentclass} or \texttt{\usepackage}. \texttt{LATEX} defines internal commands whose names contain the character \texttt{\@}; this enables it to avoid clashes between its internal names and names that we would normally use in our documents. In order that these commands may work at all, \texttt{\documentclass} and \texttt{\usepackage} play around with the meaning of \texttt{\@}. Solve this problem by using the correct command to include the file.

But, you will say, "The \texttt{LATEX} Companion tells me to use commands containing \texttt{\@}!"
Indeed; for example, there’s a lengthy section about \@startsection and how to use it to control the appearance of section titles. Page 15 of The Companion explains this; and suggests that you make such changes in the document preamble, between \makeatletter and \makeatother. So the definition of \subsection on page 26 could be:

\makeatletter
\renewcommand{\subsection}{% \@startsection
{subsection}{% name
...
{\normalfont\normalsize\itshape}}% style
\makeatother

110 Where are the msx and msy fonts?
The msx and msy fonts were designed by the American Mathematical Society in the very early days of \TeX, for use in typesetting papers for mathematical journals. They were designed using the ‘old’ METAFONT, which wasn’t portable and which is no longer available; for a long time they were available only in 300dpi versions which imperfectly matched many modern printers. The AMS has now redesigned the fonts, using the current version of METAFONT, and the new versions are called the msa and msb families; they are available from fonts/ams/amsfonts/sources/symbols

Nevertheless, msx and msy continue to turn up to plague us. There are, of course, still sites that haven’t got around to upgrading; but, even if everyone upgraded, there would still be the problem of old documents that specify them.

If you have a .tex source that requests msx and msy, the best technique is to edit it so that it requests msa and msb (you need only change the single letter in the font names).

If you have a DVI file that requests the fonts, there is a package of virtual fonts (see question 31) to map the old to the new series; it’s available in fonts/vf-files/msx2msa

111 Where are the am fonts?
One still occasionally comes across a request for the am series of fonts. The initials stood for ‘Almost [Computer] Modern’, and they were the predecessors of the Computer Modern fonts that we all know and love (or hate). There’s not a lot one can do with these fonts; they are (as their name implies) almost (but not quite) the same as the cm series; if you’re faced with a document that requests them, all you can reasonably do is to edit the document. The appearance of DVI files that request them is sufficiently rare that no-one has undertaken the mammoth task of creating a translation of them by means of virtual fonts; however, most drivers let you have a configuration file in which you can specify font substitutions. If you specify that every am font should be replaced by its corresponding cm font, the output should be almost correct.

112 ‘String too long’ in \TeX
The \TeX diagnostic “Warning: you’ve exceeded 1000, the global-string-size, for entry foo” usually arises from a very large abstract or annotation included in the database. The diagnostic usually arises because of an infelicity in the coding of abstract.bst, or styles derived from it. (One doesn’t ordinarily output annotations in other styles.)

The solution is to make a copy of the style file (or get a clean copy from CTAN — biblio/bibtex/contrib/abstract.bst), and rename it (e.g., on a long file-name system to abstract-long.bst). Now edit it: find function outputnonnull and

- change its first line (line 60 in the version on CTAN) from “s := ” to “s := swap$”, and
- delete its last line, which just says "s" (line 84 in the version on CTAN).

Finally, change your \bibliographystyle command to refer to the name of the new file.

This technique applies equally to any bibliography style: the same change can be made to any similar outputnonnull function.

If you’re reluctant to make this sort of change, the only way forward is to take the entry out of the database, so that you don’t encounter \TeX’s limit, but you may need to retain the entry because it will be included in the typeset document. In such cases, put the body of the entry in a separate file:

@article{long.boring, 
author = "Fred Verbose",
abstract = "\{\input{abstracts/long.tex}\}"
}

In this way, you arrange that all \TeX has to do with is the file name, though it will tell \TeX (when appropriate) to include all the long text.

Q Why does it do that?
113 What’s going on in my \include commands?
The original \TeX provided the \include command to address the problem of long documents: with the relatively slow computers of the time, the companion \includeonly facility was a boon. With the vast increase in computer speed, \includeonly is less valuable (though it still has its place in some very large projects). Nevertheless, the facility is retained in current \TeX, and causes some confusion to those who misunderstand it.

In order for \includeonly to work, \include makes a separate .aux file for each included file, and makes a ‘checkpoint’ of important parameters (such as page, figure, table and footnote numbers); as a direct result, it must clear the current page both before and after the \include command. What’s more, this mechanism doesn’t work if a \include command appears in a file that was \included itself: \TeX diagnoses this as an error.

\footnote{The fonts acquired their label ‘Almost’ following the realisation that their first implementation in METAFONT 79 still wasn’t quite right; Knuth’s original intention had been that they were the final answer}
So, we can now answer the two commonest questions about \texttt{\include}:

- Why does \TeX{} throw a page before and after \texttt{\include} commands?
  
  **Answer:** because it has to. If you don’t like it, replace the \texttt{\include} command with \texttt{\input} — you won’t be able to use \texttt{\includeonly} any more, but you probably don’t need it anyway, so don’t worry.

- Why can’t I nest \texttt{\include}ed files? — I always used to be able to under \TeX{} 2.09.
  
  **Answer:** in fact, you couldn’t, even under \TeX{} 2.09, but the failure wasn’t diagnosed. However, since you were happy with the behaviour under \TeX{} 2.09, replace the \texttt{\include} commands with \texttt{\clearempty} as appropriate.

### 114 Why does it ignore paragraph parameters?

When \TeX{} is laying out text, it doesn’t work from word to word, or from line to line; the smallest complete unit it formats is the paragraph. The paragraph is laid down in a buffer, as it appears, and isn’t touched further until the end-paragraph marker is processed. It’s at this point that the paragraph parameters have effect; and it’s because of this sequence that one often makes mistakes that lead to the paragraph parameters not doing what one would have hoped (or expected).

Consider the following sequence of \TeX:  

\begin{verbatim}
\raggedright \% declaration for ragged text
Here's text to be ranged left in our output,
but it's the only such paragraph, so we now end the group.}
\end{verbatim}

Here's more that needn't be ragged...

\TeX{} will open a group, and set the ragged-setting parameters within that group; it will then save a couple of sentences of text and close the group (thus restoring the previous value of the ragged-setting parameters). Then it encounters a blank line, which it knows to treat as a \verb+\par+ token, so it typesets the two sentences; but because the enclosing group has now been closed, the parameter settings have been lost, and the paragraph will be typeset normally.

The solution is simple: close the paragraph inside the group, so that the setting parameters remain in place. An appropriate way of doing that is to replace the last three lines above with:

\begin{verbatim}
\end{the group.\par}
Here's more that needn't be ragged...
\end{verbatim}

In this way, the paragraph is completed while the setting parameters are still in force within the enclosing group.

Another alternative is to define an environment that does the appropriate job for you. For the above example, \TeX{} already defines an appropriate one:

\begin{verbatim}
\begin{flushleft}
Here's text to be ranged left...
\end{flushleft}
\end{verbatim}

Other issues, too, can come into play here (\TeX{} resets many of its paragraphing parameters at end-of-paragraph), but \TeX{} does its best to defend the naïve user against such problems; for \textit{real} \TeX{} users, however, the only defense is the usual one: read the \TeX{}book, over and over and over again...

### 115 What’s the reason for ‘protection’?

Sometimes \TeX{} saves data it will re-read later. These data are often the argument of some command; they are the so-called moving arguments. (‘Moving’ because data are moved around.) Places to look for are all arguments that may go into the table of contents, list of figures, etc.; namely, data that are written to an auxiliary file and read in later. Other places are those data that might appear in head- or footlines. Section headers and figure captions are the most prominent examples; there’s a complete list in Lamport’s book (see question 19).

What’s going on really, behind the scenes? The commands in the moving arguments are already expanded to their internal structure during the process of saving. Sometimes this expansion results in invalid TEX code when processed again. \verb+\protect+ tells TEX to save \verb+\cmd+ as \verb+\cmd+, without expansion.

What is a ‘fragile command’? It’s a command that expands into illegal TEX code during the save process.

What is a ‘robust command’? It’s a command that expands into legal TEX code during the save process.

No-one (of course) likes this situation; the \TeX{}3 team have removed the need for protection of some things in the production of \TeX{}2\textit{e}, but the techniques available to them within current \TeX{} mean that this is an expensive exercise. It remains a long-term aim of the team to remove all need for these things.

### 116 Why doesn’t \texttt{\verb} work within...?

The \TeX{} verbatim commands work by changing category codes. Knuth says of this sort of thing “Some care is needed to get the timing right...”, since once the category code has been assigned to a character, it doesn’t change. So \verb+\cmd+ has to assume that it is getting the first look at its parameter text; if it isn’t, \TeX{} has already assigned category codes so that \verb+\cmd+ doesn’t have a chance. For example:

\begin{verbatim}
\verb+\error+
\end{verbatim}

will work (typesetting \verb+\error+), but

\begin{verbatim}
\newcommand{\unbrace}{\verb+\error+}
\end{verbatim}

will not (it will attempt to execute \verb+\error+). Other errors one may encounter are \verb+\error ended by end of line+, or even \verb+\error illegal in command argument+.

This is why the \TeX{} book insists that verbatim commands must not appear in the argument of any other command; they aren’t just fragile, they’re quite unusable in any command parameter, regardless of \verb+\protect+ (see question 115).
117 Case-changing oddities

\TeX provides two primitive commands \texttt{\textbackslash upper case} and \texttt{\textbackslash lowe r case} to change the case of text; they’re not much used for this purpose (although they are used for an incredible variety of apparently unrelated tasks!), but are capable of creating considerable confusion.

The two commands do not expand the text that is their parameter — the result of \texttt{\textbackslash upper case(abc)} is ’ABC’, but \texttt{\textbackslash upper case(\texttt{\textbackslash abc})} is always ’\texttt{\textbackslash abc}’, whatever the meaning of ’\texttt{\textbackslash abc}’. The commands are simply interpreting a table of equivalences between upper- and lowercase characters. They have (for example) no mathematical sense, and

\begin{verbatim}
\texttt{\textbackslash upper case\{About \$y=f(x)\$\}}
\end{verbatim}

will produce

\begin{verbatim}
ABOUT $y=\textit{f}(X)$
\end{verbatim}

which is probably not what is wanted.

In addition, \texttt{\textbackslash upper case} and \texttt{\textbackslash lowe r case} do not deal very well with non-American characters, for example \texttt{\textbackslash upper case(a\texttt{\textbackslash e})} is the same as \texttt{\textbackslash a e}. \LaTeX provides commands \texttt{\textbackslash MakeUpper case} and \texttt{\textbackslash MakeLower case} which fixes the latter problem. These commands are used in the standard classes to produce upper case running heads for chapters and sections.

Unfortunately \texttt{\textbackslash MakeUpper case} and \texttt{\textbackslash MakeLower case} do not solve the other problems with \texttt{\textbackslash upper case}, so — for example — a section title containing \texttt{\textbackslash begin\{tabular\}} \texttt{\ldots \textbackslash end\{tabular\}} will produce a running head containing \texttt{\begin{tabular}\textbf{Table}\textbackslash end\{\textbf{Table}\}}. The simplest solution to this problem to use a user-defined command, for example:

\begin{verbatim}
newcommand\{mytable\}\{\begin{tabular}\end{tabular}\}
\section\{A section title \textprotect\{mytable\}\{ with a table\}
\end{verbatim}

Note that \texttt{\textbackslash mytable} has to be protected, otherwise it will be expanded and made upper case.

118 Why are \texttt{#} signs doubled in macros?

The way to think of this is that \texttt{\textbackslash #} gets replaced by \texttt{#} in just the same way that \texttt{\textbackslash 1} gets replaced by ‘whatever is the first argument’.

So if you define a macro and use it as:

\begin{verbatim}
def\a\#1{+++++++\def\x \#1\{xxx\#1\}} \a\{b\}
\end{verbatim}

the macro expansion produces ‘+++++++\texttt{\textbackslash d ef\x \#1\{xxx\#1\}’’, which people find normal. However, if we now replace part of the macro:

\begin{verbatim}
def\a\#1{+++++++\def\x \#1\{xxx\#1\}}
\end{verbatim}

\texttt{\textbackslash a\{b\}} will expand to ‘+++++++\texttt{\textbackslash def\x b\{xxxb\}}’. This defines \texttt{\x} to be a macro delimited by \texttt{b}, and taking no arguments, which people may find strange, even though it is just a specialisation of the example above. If you want \texttt{\a} to define \texttt{\x} to be a macro with one argument, you need to write:

\begin{verbatim}
def\a\#1{+++++++\def\x \#1\{xxx\#1\}}
\end{verbatim}

and \texttt{\textbackslash a\{b\}} will expand to ‘+++++++\texttt{\textbackslash def\x \#1\{xxx\#1\}’’, because \texttt{\#1} gets replaced by ‘\texttt{\textbackslash b}’ and \texttt{\#\#} gets replaced by \texttt{\textbackslash #}.

To nest a definition inside a definition inside a definition... which fixes the latter problem. These commands are used in the standard classes to produce upper case running heads for chapters and sections.

Unfortunately \texttt{\textbackslash MakeUpper case} and \texttt{\textbackslash MakeLower case} do not solve the other problems with \texttt{\textbackslash upper case}, so — for example — a section title containing \texttt{\begin{tabular}\end{tabular}} will produce a running head containing \texttt{\begin{tabular}\textbf{Table}\textbackslash end\{\textbf{Table}\}}. The simplest solution to this problem to use a user-defined command, for example:

\begin{verbatim}
newcommand\{mytable\}\{\begin{tabular}\end{tabular}\}
\section\{A section title \textprotect\{mytable\}\{ with a table\}
\end{verbatim}

Note that \texttt{\textbackslash mytable} has to be protected, otherwise it will be expanded and made upper case.

119 Why does \LaTeX split footnotes across pages?

\LaTeX splits footnotes when it can think of nothing better to do. Typically, when this happens, the footnote mark is at the bottom of the page, and the complete footnote would overfill the page. \LaTeX could try to salvage this problem by making the page short of both the footnote and the line with the footnote mark, but its priorities told it that splitting the footnote would be preferable.

As always, the best solution is to change your text so that the problem doesn’t occur in the first place. Consider whether the text that bears the footnote could move earlier in the current page, or on to the next page.

If this isn’t possible, you might want to change \LaTeX’s perception of its priorities: they’re controlled by \texttt{\textbackslash \textit{interfootnotelinepenalty}} — the larger it is, the less willing \LaTeX is to split footnotes.

Setting

\begin{verbatim}
\texttt{\textbackslash \textit{interfootnotelinepenalty}=10000}
\end{verbatim}

inhibits split footnotes altogether, which will cause ‘Underfull \texttt{\textbackslash vbox}’ messages unless you also specify \texttt{\textbackslash \textit{raggedbottom}}. The default value of the penalty is 100, which is rather mild.

An alternative technique is to juggle with the actual size of the pages. \texttt{\enlargethispage} changes the size of the current page by its argument (for example, you might say \texttt{\enlargethispage\{15mm\}} to add a single line to the page, but you can use any ordinary \TeX length such as \texttt{15mm} or \texttt{-20pt} as argument). Reducing the size of the current page could force the offending text to the next page; increasing the size of the page may allow the footnote to be included in its entirety. It may be necessary to change the size of more than one page.

R “Recent” developments

120 The New Font Selection Scheme (NFSS)

NFSS was an extension to \LaTeX written by Frank Mittelbach and Rainer Schöpf. It is described in \textit{TUGboat} 10(2). In traditional typesetting, fonts are described by four parameters: the \texttt{family} (e.g., computer modern), the \texttt{series} (i.e., the weight and width of the font, such as light or bold), the \texttt{shape} (e.g., italic), and the \texttt{size}. NFSS is a mechanism allowing the user to change any of these independently. NFSS makes it relatively easy to use nonstandard fonts such as the PostScript ones with \LaTeX, and makes it easy to change maths fonts as well. It also allows dynamic loading of fonts at runtime (i.e., not when the format file is created).

With the demise of \LaTeX 2.09 as supported software, the label ‘NFSS’ has become somewhat misleading, as there’s no
The base distribution of $\Omega$ was released in November 1996; it is available via \texttt{system/omega}.

Implementations of Omega are available as part of the \texttt{tcT\TeX}, \texttt{mikT\TeX}, \texttt{fpT\TeX} and \texttt{CMacT\TeX} distributions (\texttt{QrefT\TeX-systems}).

124 The $\mathcal{N}\mathcal{T}\mathcal{S}$ project

The $\mathcal{N}\mathcal{T}\mathcal{S}$ project first saw the light of day at the Hamburg meeting of DANTE during 1992, as a response to an aspiration to produce something even better than \T\EX. The project is not simply enhancing \T\EX, for two reasons: first, that \T\EX itself has been frozen by Knuth (see question 15), and second, even if they were allowed to develop the program, some members of the $\mathcal{N}\mathcal{T}\mathcal{S}$ team feel that \T\EX in its present form is simply unsuitable for significant further development. While all those involved in the project are involved with, and committed to, \T\EX, they recognise that the end product may very well have little in common with \T\EX other than its philosophy; although it is a major design goal that $\mathcal{N}\mathcal{T}\mathcal{S}$ will for at least the first five years of its life be able to process all extant \T\EX documents to produce typographically identical output.

Initially, and despite the reservations expressed at the inaugural meeting, the group concentrated on extending \T\EX per se; members implemented extensions and enhancements to \T\EX through the standard medium of a change-file. These extensions and enhancements, together with \T\EX proper, form a system called $\varepsilon$-\T\EX, which is 100\% compatible with \T\EX; furthermore, it is possible during format creation to construct a format that is \T\EX: no extensions or enhancements are present.

The final aim of the project will be to produce an entirely new typesetting system, building on the experience gained in the earlier phases. This system is intended to provide a stable basis for typesetting in the future, in the way that \T\EX has since it was first offered to the world.

A distribution of $\varepsilon$-\T\EX was made available in November 1996. It is available via \texttt{systems/e-tex}; $\varepsilon$-\T\EX is also distributed on the \T\EX Live CD-ROM (see question 38).

In terms of $\mathcal{N}\mathcal{T}\mathcal{S}$ proper, work started in Q1 1998 by Karel Skoupý, a gifted Czech programmer working under the direct supervision of Prof. Jiří Zlatuska (currently Rector of Masaryk University, Brno, CZ). During mid-1999 Prof. Zlatuska stood down as project leader and the group elected Hans Hagen of Pragma (NL) to replace him; Jiří continues as a member of the team and is now Managing Director, replacing Joachim Lammarsh (URZ, Heidelberg, DE) who had been M.D. since the project’s inception and who was (and is) its \textit{fons et origo}. Other members of the team include Philip Taylor (RHBNC, UK) who is Technical Director, Peter Breitenlohner (Max Planck, München, DE) who is the group’s resident “Wizard of \T\EX”, Taco Hoekwater (Kluwer, NL), Jerzy Ludwikowski (MKU, Toruń, PL) and Volker Schaar (GSI, DE) who acts as Dante e.V. liaison. Former members of the team, who have contributed greatly to its success, include Friedhelm Sowa, Bernd Raichle, Rainer Schöpf and Joachim Schrod. Marion Neubauer, too, has helped greatly by acting as liaison with the Dante e.V. Präsidium. Karel will demonstrate a pre-release of $\mathcal{N}\mathcal{T}\mathcal{S}$ at EuroT\EX’99, and

123 The Omega project

Omega ($\Omega$) is a program built as an extension of the \T\EX sources which works internally with 16-bit characters (Unicode); this allows it to work with most scripts in the world with much reduced coding scheme complications. Omega also has a powerful concept of input and output filters to allow the user to work with existing transliteration schemes, etc. Omega is an ongoing project by John Plaice (\texttt{plaice@cse.unsw.edu.au}) and Yannis Haralambous (\texttt{Yannis.Haralambous@univ-lille1.fr}). An email discussion list is available: subscribe by sending a message ‘\texttt{subscribe omega <your name>\textquotesingle}’ to \texttt{listserv@ens.fr}.

121 \LaTeX\textsc{2e} (the ‘new’ standard \LaTeX\textsc{2e})

\LaTeX\textsc{2e} is the version of \LaTeX\textsc{} prepared and supported by the \LaTeX\textsc{3} project team (see question 122). With the advent of \LaTeX\textsc{2e}, users gained a formal support structure, and the knowledge that bug fixes would be incorporated into the distribution on a fairly regular basis (new releases appear at approximately 6-month intervals).

\LaTeX\textsc{2e} is so structured that significant enhancements may be added using macro add-ons, rather than by the different \TeX formats that were the bane of the \LaTeX\textsc{2.09} world.

\LaTeX\textsc{2e} is upwardly compatible with \LaTeX\textsc{2.09}, but has new features. In the December 1998 release, these include:

- NFSS (see question 120) is part of the distribution.
- \texttt{SiN\LaTeX} is now merely a different document class, so that there is no longer a need for a separate format.
- Better control of floating environments, such as figures.
- There is a documented interface for package and class writers (though not yet for designers).
- The box commands have been enhanced, \textit{e.g.}, options to specify the height of a minipage.
- Several standard commands are no longer fragile (see question 115); they can therefore be included in the argument of commands such as \texttt{\caption} without being protected.
- \texttt{newcommand} can define commands with one optional argument; such commands are automatically robust.
- There is now a standard package for colour and graphics inclusion.
- There is standard support for typesetting in cyrillic.

122 The \LaTeX\textsc{3} project

The \LaTeX\textsc{3} project team (see \url{http://www.latex-project.org/latex3.html}) is a small group of volunteers whose aim is to produce a major new document processing system based on the principles pioneered by Leslie Lamport in the current \LaTeX. It will remain freely available to the world.

\LaTeX\textsc{3} was made available in November 1996. It is available via \texttt{systems/latex3}; \LaTeX\textsc{3} is also distributed on the \LaTeX Live CD-ROM (see question 38).

\LaTeX\textsc{3} is upwardly compatible with \LaTeX\textsc{2.09}, but has new features. In the December 1998 release, these include:

- NFSS (see question 120) is part of the distribution.
- \texttt{SiN\LaTeX} is now merely a different document class, so that there is no longer a need for a separate format.
- Better control of floating environments, such as figures.
- There is a documented interface for package and class writers (though not yet for designers).
- The box commands have been enhanced, \textit{e.g.}, options to specify the height of a minipage.
- Several standard commands are no longer fragile (see question 115); they can therefore be included in the argument of commands such as \texttt{\caption} without being protected.
- \texttt{newcommand} can define commands with one optional argument; such commands are automatically robust.
- There is now a standard package for colour and graphics inclusion.
- There is standard support for typesetting in cyrillic.

123 The Omega project

Omega ($\Omega$) is a program built as an extension of the \T\EX sources which works internally with 16-bit characters (Unicode); this allows it to work with most scripts in the world with much reduced coding scheme complications. Omega also has a powerful concept of input and output filters to allow the user to work with existing transliteration schemes, etc. Omega is an ongoing project by John Plaice (\texttt{plaice@cse.unsw.edu.au}) and Yannis Haralambous (\texttt{Yannis.Haralambous@univ-lille1.fr}). An email discussion list is available: subscribe by sending a message ‘\texttt{subscribe omega <your name>\textquotesingle}’ to listserv@ens.fr.
the final version will be demonstrated and released at TUG 2000.

125 The PdfTEX project

PdfTEX (formerly known as TEx2PDF) arose from Hàn Thế Thành’s post-graduate research at Masaryk University, Brno, Czech Republic. The basic idea is very simple: to provide a version of TEx that can output PDF as an alternative format to DVI. PdfTEX implements a number of new primitives, to switch to PDF output, and to control various PDF features. Hàn Thế Thành is continuing work on the project; the latest version is available from systems/pdftex, and a version was distributed on the TEX Live CD-ROM (see question 38).

S Perhaps there isn’t an answer?

126 What to do if you find a bug

For a start, make entirely sure you have found a bug. Double-check with books about TEx, E\textsc{e}X, or whatever you’re using; compare what you’re seeing against the other answers above; ask every possible person you know who has any TEx-related expertise. The reasons for all this caution are various.

If you’ve found a bug in TEx itself, you’re a rare animal indeed. Don Knuth is so confident of the quality of his code that he offers real money prizes to finders of bugs; the cheques he writes are such rare items that they are seldom cashed. If you think you have found a genuine fault in TEx itself (or METAFONT, or the CM fonts, or the TExbook), don’t immediately write to Knuth, however. He looks at bugs less than once per year now, and even then only after they are agreed as bugs by a small vetting team. In the first instance, contact Barbara Beeton at the AMS (bmb@math.ams.org), or contact TUG (see question 16).

If you’ve found a bug in E\textsc{e}X, look in the bugs database to see if it’s already been reported. If not, you should submit details of the bug to the E\textsc{e}X3 team. To do this, you should process the file latexbug.tex with E\textsc{e}X (the file is part of the E\textsc{e}X distribution). The process will give you instructions about what to do with your bug report (it can, for example, be sent to the team by email). Please be sparing of the team’s time; they’re doing work for the good of the whole E\textsc{e}X community, and any time they spend tracking down non-bugs is time not available to write or debug new code. Details of the whole process, and an interface to the database, are available via http://www.latex-project.org/help.html.

If you’ve found a bug in LATEX 2.09, or some other such unsupported software, there’s not a lot you can do about it. You may find help or de facto support on a newsgroup such as comp.tex.tex or on a mailing list such as texhax@tex.ac.uk, but posting non-bugs to any of these fora can lay you open to ridicule! Otherwise you need to go out and find yourself a willing TEx-consultant — TUG maintains a register of TEx consultants (see http://www.tug.org/consultants.html).

The UK TEx Users Group Committee
after the original maintained by
Bobby Bodenheimer

Software Review: Scientific WorkPlace, Version 3.0 for Windows

Scientific WorkPlace, Version 3.0. MacKichan Software (UK Distributors: Scientific Word Ltd., 49 Queen Street, Peterhead, Aberdeenshire, AB42 1TU. Tel: 0845 766 0340. Fax: 01779 490600. Email: christopher@sciword.demon.co.uk Web: http://www.sciword.demon.co.uk/) Price: £520+VAT.

Once when lecturing to a class Lord Kelvin wrote on the blackboard the formula

$$\int_{-\infty}^{\infty} e^{-x^2} \, dx = \sqrt{\pi}$$

Then putting his finger on what he had written he turned to the class and said: “A mathematician is one to whom that is as obvious as that twice two makes four is to you.”

What would Kelvin have made of Scientific WorkPlace, a program in which the left-hand side of this equation can be ‘written’ as easily as on a blackboard, and which knows the answer just as well as it knows the answer to $2 \times 2$ (in either case one just needs to select Evaluate from the Maple menu)? It may not be a mathematician, but it is a mathematician’s computer program.

MacKichan Software’s graphical equation editor is the easiest to use of any that I have encountered. This was my reaction when I first used it in Version 1.1 of Scientific Word (Scientific Word is the document preparation part of Scientific WorkPlace, i.e. without the ability to calculate) five years ago, and I was pleased to find on installing this latest version of the software that the same easiness of use is still there. One minor gripe is that the default set of toolbars generated on installation did not include all the buttons familiar from the earlier version, so there was a slight culture shock while I wondered where to find the Greek letters. If you have been lucky enough to have used the Lotus Ami Pro word-processor, you may already have encountered this equation editor, and I understand it is still used by Word Pro.

Another impressive feature is that the on-screen display produces equations that are pleasurable to view. Such aesthetic considerations may not weigh highly with some, but for me mathematics is a beautiful subject and deserves to be presented in a way that respects its beauty. It is a welcome feature in version 3.0 that as an alternative to typesetting documents via E\textsc{e}X one can print them using the same rendering of the mathematics that appears on screen.

When I first used Scientific Word five years ago, colleagues more experienced in E\textsc{e}X than I then was wondered why I should need software costing several hundred pounds to do
what they could achieve with a simple text editor and a \texttt{LATEX} previewer. In terms of the abilities of Scientific \texttt{Word}, I have no answer to this. If you are comfortable composing \texttt{LATEX} with a text editor, then I would not try to persuade you otherwise. Scientific \texttt{Word} is software for the occasional \texttt{LATEX} user rather than the expert. Scientific \texttt{WorkPlace}, however, is a different matter. This is a working environment for anyone involved in the mathematical sciences, in which problems can be written in as near to standard mathematical language as one is going to get, and solutions to those problems derived using the numerical and symbolic calculation facilities of Maple. In this respect its nearest competitor is MathCad, as one is going to get, and solutions to those problems derived using \texttt{LATEX} as its native file format. Also, as well as including Maple, Scientific \texttt{WorkPlace} offers the ability to link to Mathematica, if this is already installed on your system. If, like me, you admire the Mathematica kernel, but find its notebook front-end awkward and cumbersome to use then Scientific \texttt{WorkPlace} will provide a welcome alternative.

Do not expect to be able to view and edit absolutely any \texttt{LATEX} file with Scientific \texttt{WorkPlace}. There are occasions when documents cannot be loaded because “SW cannot handle certain \texttt{newcommand} or \texttt{def} statements in the \texttt{LATEX} preamble”. However, having said this, after choosing a couple of files at random from the Los Alamos preprint archive, I had no trouble loading them into Scientific \texttt{WorkPlace}.

In the current version of the software, a distinction has been drawn between the \texttt{style} of the document, which controls the on-screen appearance and the non-typeset printing, and the \texttt{LATEX} typesetting specifications for the document. Earlier versions of the product used the terminology ‘style’ to refer to the typesetting specifications, a usage which has confusingly remained in the title of the program that allows new typesetting specifications to be created: the \texttt{style editor}. If like me, you have always been daunted by the prospect of creating your own .\texttt{sty} files, this \texttt{style editor} could be a useful tool. The time available for completing this review was too short to be able to offer a fully considered opinion on its abilities, but in a relatively short time I was able to create a reasonable .\texttt{sty} file, something I have never before attempted in five years of \texttt{LATEX} use. For many purposes, however, there is no need to even attempt to create your own typesetting specifications. The built-in document shells include specific-
Modern—by their smaller counters and heavier, cramped appearance. Some releases of Almost Modern had a tell-tale signature: a small horizontal stroke protruding from the right notch of lower case ‘y’, a bit like this: ‘y’.

The changes made to Almost Modern to produce Computer Modern were largely the result of interaction between Knuth and the type designers Hermann Zapf, Matthew Carter, Richard Southall (who did the Computer Modern sans serif), Neenie Billawalla (who did the CALLIGRAPHIC CAPS), Chuck Bigelow and Chris Holmes.

There was another wrinkle, to do with PK files, and to understand it we need to look back a little further into \TeX’s history. The very first versions of \TeX and METAFONT were rather more device-specific than today’s versions. For example, \TeX sometimes produced output files ready for direct ingestion by particular printers, rather as PdfTEX today rather more device-specific than today’s versions. For example, \TeX sometimes produced output files ready for direct ingestion by particular printers, rather as PdfTEX today produces an alternative output format to DVI. Early METAFONT too produced pixel files in the format of particular devices. Some of these early font formats had intriguing features. For example the VNT format fonts contained each character of the font twice: once in the normal orientation, and once rotated by 90°. This was very useful for jobs such as labelling the axes of graphs. (Remember that PostScript was not yet invented, so there was no obvious way of rotating text.) Later, the ‘pixel’ or PXL format became METAFONT’s standard output. Rotated characters were abandoned, with the proviso that ‘there is a separate utility that can read a

DVI

format, it allows for fonts of up to 256 characters, and it is also easier for DVI translators to process efficiently. The \TeX community settled on the PK format as its standard, and use of the PXL format faded away. Programs were made available that converted to and from each of the different font formats: PXtoPK, PKtoPX, PKtoGF, and so on.

Why hasn’t METAFONT been altered to write PK files instead of GF? At least two reasons: GF files can contain METAFONT special s and numspecial s, which may not appear in PK files. Such special s are used by GftoDVI, for example, to print labels on proof sheets of fonts.

For a time, \TeX users could still come across PXL format fonts. For example, there was a useful program by David Strip and Dimitri Vulis, called HP2PXL, which read Hewlett-Packard soft fonts and generated a TFM that could be read. It also converted the soft font bitmaps into PXL format, so that DVI drivers could set type using them.

An ingenious use of HP2PXL was made by an anonymous person in early 1989. S/he used PC Paintbrush to write the letters of the alphabet, in handwriting, creating the file HAND.PCX. S/he then created an HP softfont, HAND.SFP, from these handwritten letters using a program called PCX2FTN1. Finally, HP2PXL was used to convert HAND.SFP into HAND.PXL and HAND.TFM, which could be used directly in \TeX and on a 300 dpi laser printer. However, the wide availability of PXtoPK meant that even these PXL files were usually converted into PK format before use.

It may come as rather a shock to see \TeX setting type in handwriting! But the possibilities are intriguing.

The quick brown fox jumped over the lazy sheep. 0123456789

Dominik Wajastyk
The 1998–99 UK-TUG committee

Chairman
Philip Taylor: P.Taylor@mail.rhbnc.ac.uk

Treasurer
Peter Abbott: peter.abbott@tex.ac.uk

Secretary
John Palmer: johnp@bcs.org.uk

Membership Secretary
Phil Molyneux: molyneux@kingston.ac.uk

Kaveh Bazargan: kaveh@focal.demon.co.uk
Malcolm Clark: malcolm.clark@kcl.ac.uk
Roy Everett: rpe@sanglianet.co.uk
James Foster: j.foster@sussex.ac.uk
David Hardy: dhardy@sunrise-setting.co.uk
Hong Ji: hong@focal.demon.co.uk
Kim Roberts: robertsk@oup.co.uk
Dominik Wujastyk: ucgadkw@ucl.ac.uk

Contacting UK-TUG

Orders for books or software should be addressed to the Treasurer: Peter Abbott, 1 Eymore Close, Selly Oak, Birmingham B29 4LB. Fax/telephone: 0121-476 2159.

General enquiries should be sent to the Secretary: John Palmer, 69 St Cross Road, Winchester SO23 9RE. Telephone: 01962 865261.

Please get in touch with the Membership Secretary [Phil Molyneux, Kingston Business School, Kingston University, Kingston Hill, Kingston-upon-Thames KT2 7LB; e-mail: molyneux@kingston.ac.uk; tel (work) 0181-547 2000 ext 5233, (home) 0181-549 0045] regarding payment of UK-TUG subscriptions, or changes to personal details held on file.

UK-TUG maintains pages on the World Wide Web at http://uk.tug.org/. E-mail enquiries about UK-TUG may be sent touktug-enquiries@tex.ac.uk.

Baskerville

Articles should be submitted using electronic mail to baskerville@tex.ac.uk; contributors who cannot submit using e-mail may seek advice from the Secretary. Any correspondence concerning Baskerville may also be addressed in this way. Earlier issues of Baskerville can be found on CTAN inusergrps/uktug/baskervi.

Book Discounts for UK-TUG members

We have arrangements with Addison-Wesley for their well-known \TeX-related publications, with International Thomson Publishing for the O’Reilly & Associates Inc. series, and with Oxford University Press, to supply books to members at discounted prices.

Please send details (including ISBN if possible) of books required to Peter Abbott. Peter will supply a quotation.

We are allowed to offer this service only to current members of UK-TUG and/or members of TUG. Please send your order and cheque (in UK £) to Peter Abbott. Cheques should be made payable to ‘UKTUG’. All books will be routed through UK-TUG. In all cases, please notify Peter Abbott by e-mail, ‘phone, fax or letter when books are delivered.

Provided that the book(s) are in stock, it will normally take about a week from receipt of order to delivery of the book(s).

Obtaining \TeX from CTAN

The UK \TeX Archive at ftp://ftp.tex.ac.uk/ is part of the CTAN (Comprehensive \TeX Archive Network) collaborating network of archives on the Internet organised under the aegis of the \TeX Users Group.

The CTAN archives 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.

For interfaces to, and catalogues of, CTAN, you are referred to Graham Williams’ \TeX and \LaTeX Catalogue which is available from CTAN as help/Catalogue/catalogue.html; other interfaces are listed in http://www.tug.org/interest.html.

Please report any problems with the CTAN archives to ctan@ctan.org.

Obtaining \TeX on CD-ROM

UK-TUG, in collaboration with GUTenberg, TUG, NTG and DANTE e.V., have produced \TeX Live, a “plug-and-play” CD-ROM for Linux, Unix, and Windows 32, based on the Web2c \TeX setup. As it is formatted according to ISO 9660, with both Joliet and Rock Ridge extensions, the platform-independent files can, in principle, be read on all operating systems which can deal with that format. The CD is supplied free of charge to all members of UK-TUG.

Complete DOS, Windows 32 and Macintosh implementations are available as packages on the CD; floppy disk sets will now be supplied only on special request.

Future meetings of UK-TUG

The committee planned to hold two meetings in 1999. One has already taken place on May 28th, and was on “\TeX for Windows 32”. The second meeting, which will include the AGM, will commence on Sunday 12th September, at St Anne’s College Oxford (http://www.stannes.ox.ac.uk/). It will be on the relationships between \TeX and \LaTeX on the one hand, and SGML, HTML, XML and related notations on the other.

For further information please contact one of the Committee, or visit the UK-TUG Web site at http://uk.tug.org/.

The Committee of the UK \TeX Users’ Group