A Brief Guide to LaTeX Tools for Web Publishing

Peter R. Wilson*
peter.r.wilson@boeing.com

11 March 2000

Abstract

This document provides a brief guide to converting LaTeX documents to forms more suitable for dissemination via the Web.

Contents

List of Figures

*With helpful critiques by Eitan Gurari (gurari@cis.ohio-state.edu) and David Wilson (davidw@utopiatype.com.au).

1 Introduction

Publishing on the Web has rapidly achieved significant importance, for example, the International Organization for Standardization (ISO) is moving towards electronic forms of International Standard documents that are suitable for publishing on the Web, and in particular, documents as PDF or HTML files rather than their traditional request for camera-ready paper copy.

Documents written using LaTeX [Lam94] tagging can be easily converted to PostScript, PDF and HTML, all from the single electronic source. This guide briefly notes some of the ways that this can be accomplished. Most of the programs and systems mentioned here are described in more detail in [GR99].

I have made no attempt to design this document for Web publication. The typographical rules for printing on paper are well founded, having been developed over hundreds of years. Display on computer screens is a very different matter and requires a different set of rules, most of which, as yet, are either in a state of flux or unavailable. For LaTeXers who are interested in this topic I suggest a look at D. P. Story's work on AcroTeX (`http://www.math.uakron.edu/~dpstory/acrotex.html`). Further, for the example conversions I have used only the minimal tool options necessary. Many of the tools have extensive capabilities which are well documented in their accompanying user manuals; these should be consulted for further information.

1.1 URLs

I have tried to provide URLs for the programs and systems mentioned here. Most LaTeX-related software is available from the Comprehensive TeX Archive Network (CTAN). There are three sites, `ftp://ctan.tug.org/tex-archive` in the USA, `ftp://ftp.tex.ac.uk/tex-archive` in the UK, and `ftp://ftp.dante.de/tex-archive` in Germany, as well as several mirror sites. Usefully, the CTAN sites (but not necessarily a mirror site) supports on-the-fly zipping of files and entire directories, which makes downloading a group of files less tedious than having to get them one-by-one. Below, I have used `ftp://ctan.tug.org/tex-archive` to stand for any of the three CTAN sites.

1.2 Disclaimer

Nothing that is said in this document is meant to imply any endorsement or recommendation, either positive or negative, concerning any systems or programs mentioned herein.

Many of the systems or programs are 'free' in the sense that they are either public domain or their licences are roughly equivalent to the GNU Public License. Others are either commercial or have more restricitive licenses or may require payment. Where known, programs and systems that are not 'free' are noted.

2 PDF

The traditional output from a LaTeX (e.g., *.tex) file is a 'device independent' *.dvi file. The *.dvi file is then processed further to convert it to a format suitable for printing on a particular printing device. In the vast majority of cases the final printable format has been PostScript, obtained by running the *.dvi file through a program like `dvips`, to generate a *.ps file.

PostScript was developed by Adobe Systems. The Portable Document Format (PDF) has since also been developed by Adobe, and seems to be overtaking PostScript as the format of choice for printing, and especially for display via the Web.

DVI and PDF are somewhat similar in that they both describe where (electronic) ink is to be put on (electronic) paper. PostScript also does this but at the same time it is a complete programming language. This means that it is inherently more difficult, time consuming, and computer intensive, to process PostScript than either DVI or PDF. This is probably the reason behind the popularity of PDF on the Web.

There are now several methods of producing a PDF (e.g., *.pdf) file from *.tex. These include:

- Converting from PostScript to PDF; from *.ps to *.pdf.
- Generate PDF from the device independent file; from *.dvi to *.pdf
- Generate PDF directly from the LaTeX source; from *.tex to *.pdf.

2.1 From PostScript to PDF

There are basically two routes to getting from PostScript to PDF. The first of these is to use Acrobat software from Adobe Systems, which essentially means the commercial `Distiller` program. `Distiller` can read in a PostScript file and output a PDF file where the visual results of printing the two files are identical. This, or any other, PDF file can be viewed and/or printed via the charge-free Acrobat `Reader` program. Note that when using `Reader` the 'fit to paper' option may alter the page layout, for example by changing the height of the text block.

The second route is to use a non-Adobe converter program, like `Ghostscript` which runs on nearly all operating systems and which is obtainable from `http://www.cs.wisc.edu/~ghost`. The `Ghostscript` distribution comes with a script called `ps2pdf` which performs the conversion. The distribution also provides the popular `Ghostview` program, which is a viewer for both PostScript and PDF files.

Another converter program, which does have some licensing conditions that may not be suitable for all users, is `PStill`; it is available from `http://www.this.net/~frank/pstill.html`.

2.2 From DVI to PDF

Mark Wicks' `dvipdfm` program (`http://odo.kettering.edu/dvipdfm`) converts a *.dvi file to a *.pdf file. The program is used in the same manner as `dvips` and provides similar capabilities.

PostScript illustrations are handled in one of two ways. Simple PostScript generated by the MetaPost program [Hob92] is included natively. Any other PostScript file is first converted to PDF by using an external program like `Ghostscript` and then inserted into the output file. Illustrations in PDF, PNG and JPEG formats require no external aids.

`dvipdfm` is written in C but there are some binaries for Linux systems.

2.3 From LaTeX to PDF

The `pdfLaTeX` program being developed by Hàn Thê Thành is a modified version of TeX that generates *.pdf instead of *.dvi output files. `pdfLaTeX` is distributed with many of the free LaTeX distributions, and is also obtainable from `ftp://ftp.cstug.cz/pub/tex/local/cstug/thanh`, although it may be better to try `ftp://ctan.tug.org/tex-archive/systems/pdftex`.

Running `pdfLaTeX` is very similar to running LaTeX, but some minor changes are required to the *.tex file. For example:

```
%  example.tex   example latex file
\documentclass[...]{...}
\newif\ifpdf
\ifx\pdfoutput\undefined
  \pdffalse
\else
  \pdftrue
\fi

\ifpdf
  \pdfoutput=1
%  \usepackage[pdftex]{graphicx}  % uncomment if using graphicx
%  \usepackage[pdftex]{hyperref}  % uncomment if using hyperref
\else
%  \usepackage{graphicx}  % uncomment if using graphicx
%  \usepackage{hyperref}  % uncomment if using hyperref
\fi
....
```

Running

```
latex example
```

will produce example.dvi, while running

```
pdflatex example
```

will produce example.pdf. It is thus very easy to generate both *.dvi and *.pdf from the same LaTeX source file.

pdflatex will handle graphics files in the following formats: PDF, PNG, JPEG and TIFF, but notice that (Encapsulated) PostScript is missing from this list. However, it can handle directly the simple Encapsulated PostScript output by MetaPost [Hob92]. It does, though, expect MetaPost files to have a .mps extension. To include PostScript from other sources it is necessary to convert the PostScript to PDF.

pdftex, and hence pdflatex, has some extra primitive commands that are not available in TeX itself specifically for accessing aspects of the PDF format, for example to create hypertext links, bookmarks or article threads. Consult the manual for details.

Independently of pdflatex the hyperref package (ftp://ctan.tug.org/tex-archive/macros/latex/contrib/supported/hyperref) extends the functionality of the LaTeX cross-referencing commands to include hypertext links, and also ad hoc hypertext links to, for example, external documents and URLs.

2.4 Fonts

The normal fonts used with LaTeX are the Computer Modern family developed by Knuth using METAFONT [Knu86]. All METAFONT fonts are in the form of bitmaps, which is unfortunate when it comes to PDF. Typically, PDF will only use one size of each font for a document, and will scale this if different font sizes are required. This normally works well as fonts used with PDF are typically ‘Type 1’ fonts (e.g., PostScript fonts) which are designed to be scaleable. Bitmap fonts look terrible when scaled or printed at a resolution that they were not designed for.

In other words, expect bad results if you generate a PDF file with the original Computer Modern fonts.

Perhaps the easiest method of dealing with this is to use the most common PostScript fonts, namely Times, Courier and Helvetica. All that is necessary is to add \usepackage{times} to the document’s preamble.

Alternatively, if you need to use the CM fonts, perhaps because a lot of mathematics is involved, many LaTeX distributions include Type 1 versions of the CM fonts. If you don’t have them they can be found at ftp://ctan.tug.org/tex-archive/fonts/cm/ps-type1/bluesky and at ftp://ctan.tug.org/tex-archive/fonts/amsfonts/ps-type1 for the AMS fonts.

Goossens *et al.* provide useful and general information on installing and using different fonts with LaTeX [GMS94], while for the fontophile, Alan Hoenig [Hoe98] delves much more deeply into the installation of PostScript fonts.

TeX doesn’t care about the particular shape of any glyph, nor how it is constructed or represented, it only cares about the space occupied by each character (i.e., the *.tfm files). It is the DVI processor that needs to know in detail about the fonts in a document. So, the DVI processor has to be told to use Type 1 CM PostScript fonts. The following is for the dvips program. For convenience, let $TEXMF stand for the root of the texmf tree (e.g., /usr/teTeX/texmf).

dvips looks in the $TEXMF/dvips/base/psfonts.map to see if it can use any PostScript fonts. This file starts off something like:

```
bchb8r CHarterBT-Bold "TeXBase1Encoding ReEncodeFont" <8r.enc <bchb8a.pfb
...
```

To get dvips to use Type 1 versions of the CM fonts, additional lines must be added to psfonts.map giving similar information about the fonts. The specification for CM fonts is simpler and consists of lines like:

```
cmb10 CMB10 <cmb10.pfb
cmbsy10 CMBSY10 <cmbsy10.pfb
...
```

In the version of teTeX that I use, this information is in files bsr.map, bsr-interpolated.map, cmcyr.map, hoekwater.map, and pl.map, all in directory $TEXMF/dvips/config.

These files can either be copied by hand to the psfonts.map file in $TEXMF/dvips/base or in a modern teTeX distribution (which should also have all the CM Type 1 font data) it is easiest to do the following:

- In directory $TEXMF/dvips/config copy the script file updmap to, say, updmap.orig.
- Edit updmap to comment the line type1_default=false and uncomment the line type1_default=true.
- Run the script via ./updmap.

Another more general method is to edit the file config.ps in directory $TEXMF/dvips/config and at the appropriate place (which should be marked, but in any case after the line p psfonts.map) add lines like:

```
p +bsr.map
p +bsr-interpolated.map
...
```

Another option when using dvips which avoids all of the above, is to call it with options, like:

```
dvips -Pamz -Pcmz -Ppdf -j0 [other options] filename
```

and then use your prefered *.ps to *.pdf conversion process.

2.5 MetaPost

John Hobby's METAPOST [Hob92] is a language based drawing program based on Knuth's METAFONT [Knu86]. METAFONT was principally designed for creating fonts, and generates bitmapped output, while METAPOST is principally for drawing general line illustrations and its output is a particularly simple form of Encapsulated PostScript.

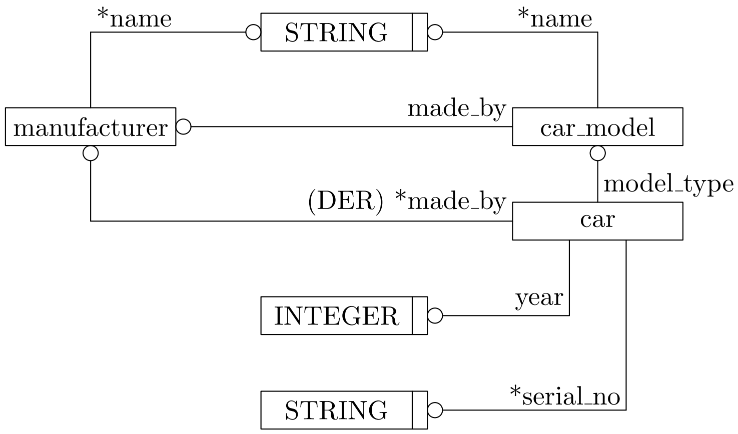

Figure 1: Metapost illustration of an EXPRESS-G diagram

This is not the place to describe METAPOST, but it can generate several output files, one for each drawing, from a single input file called, say, fred.mp. The output files have a numeric extension correspond to the number of the drawing. So, for example, it may generate files fred.1, fred.2 and fred.3. For a document that is to be processed via LaTeX these files can be included as is. However, for processing through pdfLaTeX, the files must have a .mps extension; for example fred1.mps, fred2.mps and fred3.mps.

Figure 1 is a METAPOST illustration that is included in this document by the code:

```
\begin{figure}
\centering
\ifpdf
  \includegraphics{expeg6.mps}
\else
  \includegraphics{expeg.6}
\fi
\caption{Metapost illustration of an \textsc{express-g} diagram}
  \label{fig:mp}
\end{figure}
```

where expeg6.mps is a copy of expeg.6, to cater for processing by either LaTeX or pdfLaTeX. Actually, the following will also work:

```
\begin{figure}
\centering
\includegraphics{expeg6.mps}
\caption{Metapost illustration of an \textsc{express-g} diagram}
  \label{fig:mp}
\end{figure}
```

The figure demonstrates part of the capabilities of the `expressg` METAPOST package (`ftp://ctan.tug.org/tex-archive/graphics/metapost/contrib/macros/expressg`) for drawing diagrams consisting of boxes, lines and annotations, such as flowcharts or ER diagrams.

3 HTML

There are a number of systems that convert a LaTeX tagged document into an HTML tagged document. These can be divided into two classes:

1. Systems that parse the *.tex file themselves.
2. Systems that use TeX as the file parser.

There are several that do their own parsing, but only one that I know of that uses TeX as the parser.

TeX is a macro language and the meaning of existing commands can be changed on the fly, and also new commands can be defined on the fly [Knu84]. As perhaps the most extreme example of this is David Carlisle's xii.tex TeX code, which is obtainable as `ftp://ctan.tug.org/tex-archive/macros/plain/contrib/xii.tex`:

```
\let~\catcode~`76~`A13~`F1~`j00~`P2jdefA71F~`7113jdefPALLF
PA''FwPA;;FPAZZFLaLPA//71F71iPAHHFLPAzzFenPASSFthP;A$$FevP
A@@FfPARR717273F737271P;ADDFRgniPAWW71FPATTFvePA**FstRsamP
AGGFRruoPAqq71.72.F717271PAYY7172F727171PA??Fi*LmPA&&71jfi
Fjfi71PAVVFjbigskipRPWGAUU71727374 75,76Fjpar71727375Djifx
:76jelse&U76jfiPLAKK7172F71l7271PAXX71FVLnOSeL71SLRyadR@oL
RrhC?yLRurtKFeLPFovPgaTLtReRomL;PABB71 72,73:Fjif.73.jelse
B73:jfiXF71PU71 72,73:PWs;AMM71F71diPAJJFRdriPAQQFRsreLPAI
I71Fo71dPA!!FRgiePBt'el@ lTLqdrYmu.Q.,Ke;vz vzLqpip.Q.,tz;
;Lql.IrsZ.eap,qn.i. i.eLlMaesLdRcna,;!;h htLqm.MRasZ.ilk,%
s$;z zLqs'.ansZ.Ymi,/sx ;LYegseZRyal,@i;@ TLRlogdLrDsW,@;G
LcYlaDLbJsW,SWXJW ree @rzchLhzsW,;WERcesInW qt.'oL.Rtrul;e
doTsW,Wk;Rri@stW aHAHHFndZPpqar.tridgeLinZpe.LtYer.W,:jbye
```

If you run this through TeX (not LaTeX) I'm sure you will be surprised at the result.

There is inevitably a problem when converting from LaTeX to HTML for a document that includes figures/illustrations or anything more than the most simple mathematical typesetting as, basically, HTML provides no support. Typically, mathematics and illustrations, are converted to a picture format and then inserted into the HTML document as graphics, usually with a very poor appearance.

However, for mathematics the situation is starting to change because of the advent of MathML (`http://www.w3.org/TR/MathML2`). In particular the Milestone 13 release of Mozilla (`http://www.mozilla.org/binaries.html`) is a MathML-enabled browser. Some examples, generated by TeX4ht, are available at http://www.maths.ox.ac.uk/~gartside/mozSuccess.

All the systems generate HTML tagged documents, with the particular tagging 'style' set by the system. It is advantageous to use a converter which either by default generates your desired style, or which can be modified in some manner to do so.

3.1 Self-parsing systems

The self-parsing systems incorporate their own parsers for the TeX language. In essence, this means that they 'know' the meaning of common TeX commands, but probably not all possible commands. It is advantageous to use a system that can be extended to deal with commands that were not anticipated by the author.

The only system I am familiar with in this class is Peter Wilson's ltx2x program (ftp://ctan.tug.org/tex-archive/support/ltx2x). This program works by replacing known LaTeX commands, and their arguments, by user-specified text strings [Wil96]. It is unable to handle anything more than very simple mathematics and ignores any pictures. The user-specified command texts are kept in a simple command-table file. Within limits, new LaTeX commands and environments may be specified within a command-table file and the command texts modified. The ltx2x program has been used to 'detex' (i.e., remove all LaTeX commands) files, convert to HTML, and convert to SGML. It cannot convert to XML due to a yet to be resolved technical problem in dealing with end of paragraph tags. The program is written in C and so requires a C compiler for installation. The system can be extended via some C programming, in which case the flex and bison programs are also required. There is no chance that ltx2x would ever make any sense whatsoever of xii.tex on page 7.

Perhaps the most venerable system is the LaTeX2HTML system (http://www-texdev.mpce.mq.edu.au/l2h/docs/manual or ftp://ctan.tug.org/tex-archive/support/latex2html), originally by Nikos Drakos and now maintained by Ross Moore and others. This system is written using Perl (ftp://ftp.uu.net/languages/perl). It also requires a database management system such as the Unix DBM or NDBM, or the GNU GDBM system. Further, it requires Ghostscript and the netpbm library of graphics utilities (ftp://ftp.x.org/contrib/utilities). To extend or change the default conversion style requires, I think, some Perl programming. A fuller description and examples are given in [GR99].

Another converter is the TtH program by Ian Hutchinson (http://hutchinson.belmont.ma.us/tth) which is cost-free for non-commercial use; commercial use in this case is roughly by anyone who gets paid while using it (but see http://hutchinson.belmont.ma.us/tth/tth-commercial/email.html for the actual wording). There is also another version, TeX2HTML (http://www.tex2html.com) which is the 'commercial GOLD version of the freeware Tth by Ian Hutchinson'. These programs run on the usual range of operating systems. An example of the output from TtH is given in [GR99].

3.2 TeX-based parsing system

Eitan Gurari's `TeX4ht` system appears to be unique in that it uses TeX as the parser for the LaTeX document and instead effectively takes the *.dvi file as its starting point for conversion to HTML. That is, it does not have to understand TeX code and can, in fact, convert David Carlise's xii.tex (page 7) to HTML. The system is available from `http://www.cis.ohio-state.edu/~gurari/TeX4ht/mn.html`. It consists of two C programs, one package file, and a set of *.4ht configuration files, one for each of the typical LaTeX classes and packages. It also requires ImageMagick (`http://www.wizards.dupont.com/cristy/www/archives.html`) for handling illustrations and non-simple mathematics. Simply speaking, the system is extended by writing new *.4ht file(s) and its output modified by writing simple *.cfg file(s) that override the *.4ht file(s). At the moment, the best and most detailed description is given in [GR99].

By default, `TeX4ht` can generate a non-tagged file or a file tagged with either HTML3.2 or HTML4.0. By writing appropriate *.cfg files it can be made to generate XML tagged files. The system comes with a script called `htlatex` which controls the conversion process from LaTeX; there is also the `httex` script for converting a TeX document. For instance, to convert the earlier example LaTeX file, example.tex (page 3), to an HTML4.0 tagged document, it is enough to run:
`htlatex example`
which will then output example.html. Similarly, to convert xii.tex just run:
`httex xii`.

4 Examples

Hopefully, you should find several versions of this document, all of which have been generated from a single source file. These are:

- webguide.tex — the LaTeX source.
- webguide.ps — A PostScript version from running `latex` and `dvips` on webguide.tex.
- webguide.pdf — A PDF version from runnning `pdflatex` on webguide.tex.
- webguide.html — A HTML4.0 version from running `htlatex` on webguide.tex.

The HTML version uses the GIF file webguide0x.gif for the illustration; the file is automatically generated by `tex4ht` using the ImageMagick `convert` program. The quality of the picture in the viewed document depends on the particular viewer; different versions of Netscape, for example, may display and print the diagram with very different rendering qualities. I asked TeX4ht to generate this particular figure at 180dpi instead of the default 110dpi to improve the quality.

References

[GMS94] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley Publishing Company, 1994.

[GR99] Michel Goossens and Sebastion Rahtz. *The LaTeX Web Companion – Integrating TeX, HTML, and XML.* Addison-Wesley Publishing Company, 1999. (with Eitan Gurari, Ross Moore, and Robert Sutor).

[Hob92] John Hobby. *'A User's Manual for MetaPost'.* Technical Report 162, AT&T Bell Laboratories, Murray Hill, NJ, 1992.

[Hoe98] Alan Hoenig. *TeX Unbound – LaTeX and TeX Strategies for Fonts, Graphics, & More.* Oxford University Press, 1998.

[Knu84] Donald E. Knuth. *The TeXbook.* Addison-Wesley Publishing Company, 1984.

[Knu86] Donald E. Knuth. *The METAFONTbook.* Addison-Wesley Publishing Company, 1986.

[Lam94] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley Publishing Company, second edition, 1994.

[Wil96] Peter R. Wilson. *ltx2x: A LaTeX to X Auto-tagger.* NIST Report NISTIR, June 1996.