The PracTEX Journal, 2007, No. 3
Article revision 2007/07/29

Travels in TEX Land: Fonts, self-publishing and another reason I like TEX

David Walden

Abstract In this column in each issue I muse on my wanderings around the TEX world. Section 1 of this column describes some work I did organizing my experiences of using different fonts within TEX. Section 2 describes use of an external processor in combination with TEX. Section 3 gives an update on my self-publishing efforts using TEX.

1 Organizing my experience with fonts

It occurred to me since the last column that I have used a number of fonts (i.e., font families) in different documents since I began using TEX seriously half a dozen years ago. However, each instance of a font use was pretty much a one-shot effort with me not having much of an overall picture about the spectrum of fonts available for my use. It seems like it is time to try to organize my experiences with fonts a bit.

In the following, I am merely trying to see which fonts are *easily* accessible—not trying to use the best LATEX (or whatever) conventions for specifying the fonts. In particular, I didn't study the compilation logs to see if there were various issues that needed to be resolved; I would do this if I used one of the fonts for a real project.

When I refer to *The LATEX Companion* below, I mean the second edition of this essential reference book by Frank Mittelbach et al.

1.1 My actual experience using different fonts

I decided to list all of the fonts I had used to date along with how I had accessed the font. Most of the fonts were accessed from LATEX; I have noted the couple of instances when another macro package was being used. Also, with a couple of

exceptions I note below, I did nothing special to have access to these fonts; most these are (apparently) fonts that come with my edition ProTEXt.

For simplicity of constructing this column, I have pointed to examples of the fonts in print rather than providing in-line examples of the fonts listed below.

Palatino. My first big project was using LaTeX to draft the book *Four Practical Revolutions in Management* (http://www.walden-family.com/4prim). For this book project I used Palatino by giving the following command.

```
\usepackage{palatino}  %obsolete now according to The LaTeX Companion;
        % \usepackage{mathpazo} is what is now recommended, I believe
```

(This project was described in a paper in *TUGboat* (http://tug.org/TUGboat/Articles/tb24-2/tb77walden.pdf).

Because it was what I already knew, I also the applied same approach to Palatino for my first column in this journal, TPJ 2005-1. (I'll return again to the Palation font in the next section of this column, where there is a pointer to an example.)

Latin Modern. By the second issue of this journal (TPJ 2005-2), a class file had been written and I used this.

```
\documentclass{pracjourn}
 %which in turn uses \RequirePackage{lmodern} and \usepackage[T1]{fontenc}
```

I also used the journal's default style for my column in TPJ 2006-4 and for *this column*.

For more on the Latin Modern fonts, see Will Robertson's exploration in issue 2006-1 http://www.tug.org/pracjourn/2006-1/robertson. (Will did the bulk of the work to develop the LaTeX class file used for this journal.)

Computer Modern. For the pieces I have written for *TUGboat* (see http://www.tug.org/TUGboat/Contents/listauthor.html), I used the `ltugboat` class which in turn uses Computer Modern (the TEX default). I also used this class (slightly modified but not with regard to font use) for my column for TPJ 2006-3; see the PDF at http://www.tug.org/pracjourn/2006-3/walden.

```
\documentclass[final]{ltugboat}
```

Palatino in eplain TEX. My column in TPJ 2005-4 was an experiment using eplain TEX. I set up the fonts to be used with the following code which I put near the top of my eplain file (unlike LATEX, eplain does not have an explicit preamble). The following code is discussed in more detail at http://www.tug.org/pracjourn/2005-4/walden, where the PDF is also an example of the fonts.

```
\input eplain
\def\fmtname{plain} %added per Oleg to make PDFTeX work on my file

\font\smallrm = pplr7t
\font\bigbold = pplb7t at 14pt
\font\bigbig = pplr7t at 18pt
\font\smalltt = pcrr7t
\font\smallit = pplri7t
\font\tensc = pplrc7t at 12pt

\font\sevenrm = pplr7t at 9pt  % or 8pt or whatever looks right
\scriptfont0 = \sevenrm

%The \ten... command names are used in plain TeX, so by redefining the
%fonts this way, the regular commands \rm, \it, \bf, etc. continue to work.
%The final \rm switches to the new roman font by default.
\font\tenrm = pplr7t at 12pt
\font\tenit = pplri7t at 12pt
\font\tensl= pplro7t at 12pt
\font\tenbf = pplb7t at 12pt
\font\tentt = pcrr7t at 12pt  % Courier; maybe smaller would look better
\rm \baselineskip = 16pt

\everyfootnote{\smallrm \baselineskip = 8pt}
```

Lucida. My column in TPJ 2006-1 described purchasing the Lucida fonts from TUG and trying to use them.

```
\usepackage[T1]{fontenc}
\usepackage{textcomp}
\usepackage{lucidabr}
```

Thus, you can see the font in use in the PDF at http://www.tug.org/pracjourn/2006-1/walden.

Minion Pro. For my book *Breakthrough Management* (http://www.walden-family.com/breakthrough), I used Minion Pro (upon the recommendation of Steve Peter). I already had the Minion fonts on my computer because they came with Illustrator and Photoshop; thus, I felt I could use the font legally. Steve Peter then provided me with a small set of files which made the Minion fonts available to LaTeX and the following commands for accessing a limited subset of the font family. I put this code in my class file first discussed at http://www.tug.org/pracjourn/2006-2/walden and elaborated upon at http://www.tug.org/pracjourn/2006-3/walden.

```
\RequirePackage{minion}
\RequirePackage{microtype}
\linespread{1.0325}

\renewcommand\normalsize{%
   %\@setfontsize\normalsize{11.8pt}{15.3pt}
   \abovedisplayskip 11\p@ \@plus2\p@ \@minus6\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6.5\p@ \@plus3.5\p@ \@minus3\p@
   \belowdisplayskip \abovedisplayskip
   \let\@listi\@listI}

\renewcommand\small{%
   \@setfontsize\small{10.7pt}{13.05pt}
   \abovedisplayskip 9.5\p@ \@plus2.5\p@ \@minus5\p@
   \abovedisplayshortskip \z@ \@plus\p@
   \belowdisplayshortskip 5.25\p@ \@plus2.75\p@ \@minus2.5\p@
   \belowdisplayskip \abovedisplayskip}
```

This was my first experience using old style numerals.

Because I was in the habit of using Minion Pro for my book and was describing that experience in my columns for TPJ 2006-2 and TPJ 2006-3, I also used that font for the 2006-2 and 2006-3 columns:

```
\usepackage{textcomp}
\usepackage{minion}
\usepackage{microtype}
```

You can see examples of the font in the PDFs at the above URLs.

Antykwa Toruńska. Quite arbitrarily I used the Antykwa Toruńska font for my column in TPJ 2007-1:

```
\usepackage{anttor}
```

It an interesting font; see the PDF at http://www.tug.org/pracjourn/2007-1/walden. However, I'm not sure when I would use it in a real project (rather than just experimenting in my column); nonetheless, it's nice to know the font is available and maybe I can find a use for it in a future document.

Latin Modern (again). In my column in TPJ 2007-2, I described my first experience using ConTEXt. I didn't make an explicit font specification, so I got ConTEXt's default—Latin Modern I believe (see the PDF at http://www.tug.org/pracjourn/2007-2/walden).

Latin Modern Proportional Typewriter. I have begun work on an oral history book (titled *Recollections*) that was originally typed on a typewriter and printed using a copier a Kinko's. I am going to republish that book and probably have it printed by Lightning Source Inc., and thus it will be available via Amazon. I want to maintain the feel of the original font but I'd like left and right justified pages, so I am planning to use a proportional typewriter font. I found such a font in Latin Modern via Will Robertson's paper in TPJ 2006-1 (http://www.tug.org/pracjourn/2006-1/robertson) which I have enabled as follows:

```
\RequirePackage{lmodern}
\RequirePackage[T1]{fontenc}
\renewcommand{\rmdefault}{lmvtt}
\renewcommand{\sfdefault}{lmvtt}
\renewcommand{\ttdefault}{lmvtt}
```

It's odd, I know, to specify typewriter for the roman and sans fonts, but this was the easiest way I found to tell LATEX to use typewriter for everything. (After I wrote the above, Yuri Robbers directed me to the alltt package.)

Times. For several short, miscellaneous documents I have written, I used Times

```
\usepackage{times}
```

although I understand this is also an obsolete way to specific this font and

```
\usepackage{mathptmx}
```

is a preferred approach.

1.2 The fonts chapter in *The LaTeX Companion*

Having listed all the fonts I had used myself to date, I decided to keep investigating which fonts I could easily access without downloading anything new that did not come with my ProTeXt distribution. I started by thumbing through chapter 7 of *The LaTeX Companion*, on fonts and encodings.

On pages 354 and 372 I found summary charts for Computer Modern and the Postscript base set of 35 fonts. I accessed these using the `\usefont{T1}{XXX}{m}{n}` command where `XXX` specified one of the families in those charts. To simplify my testing, I only used the `m` (medium) series and `n` (normal) shape for each font. I also didn't access the Postscript fonts using the packages listed in the chart on page 371. For instance, I used `\usefont{T1}{cmr}{m}{n}` to specify the Computer Modern Roman font.

My test file for the Computer Modern and Postscript base 35 fonts can be accessed from the link `font-text-cm-ps.tex` on the HTML page for this paper, and the output files has the link `font-text-cm-ps.pdf`.

I also looked at some of the packages described in the book, e.g., `eco` and for the Concrete fonts. My test files for these also have links on the HTML page for this column—with the file names including the text `some-packages`.

Finally, I checked out the yfonts. See the links to the files with `yfonts` in their names.

1.3 Fonts I could use from ConTeXt

In my column in the last issue (TPJ 2007-2, I described a simple approach to using fonts from ConTeXt that comes from Bill McClain.

```
\font\myfirstfont=bchb8r      %CharterBT bold
\myfirstfont
```

Since then, I have tried the following variations on the above commands to get other fonts, e.g., .

```
\font\myfirstfont=pplb8r     %Palatino bold
\myfirstfont

\font\myfirstfont=hlhr8a     %didn't work
\myfirstfont

\font\myfirstfont=rpzcmi     %Chancery italic
\myfirstfont

\font\myfirstfont=rpcrr      %Courier
\myfirstfont
```

As noted, this simple approach didn't work for one of the fonts but worked for the others.

Next I tried the fonts given in the examples in the document "Fonts in ConTEXt—Examples of Using Typescripts" (www.pragma-ade.com/general/manuals/showfont.pdf). The first two lines below were needed to make the rest of the pairs of commands work for the various fonts.

```
\switchtobodyfont[ec-lbr]
\usetypescript[berry][ec]

\definetypeface [times] [rm] [serif] [times] [default] [encoding=ec]
\switchtotypeface [times] [12pt,rm]

\definetypeface [charter] [rm] [serif] [charter] [default] [encoding=ec]
\switchtotypeface [charter] [12pt,rm]

\definetypeface [palatino] [rm] [serif] [palatino] [default] [encoding=ec]
\switchtotypeface [palatino] [12pt,rm]
```

```
\definetypeface [zapf] [cg] [calligraphy] [chancery]
\switchtotypeface [zapf] [12pt,cg]

\definetypeface [helvetica] [ss] [sans] [helvetica] [default] [encoding=ec]
\switchtotypeface [helvetica] [12pt,ss]

\definetypeface [utopia] [rm] [serif] [utopia] [default] [encoding=ec]
\switchtotypeface [utopia] [12pt,rm]

\definetypeface [informal] [rm] [casual] [informal] [default] [encoding=ec]
\switchtotypeface [informal] [12pt,rm]
```

The following combinations of commands didn't work, for reasons I don't yet understand.

```
\definetypeface [antykwa] [rm] [serif] [antykwa] [default] [encoding=ec]
\switchtotypeface [antykwa] [12pt,rm]

\definetypeface [bookman] [rm] [serif] [bookman] [default] [encoding=ec]
\switchtotypeface [bookman] [12pt,rm]

\definetypeface [postscript] [rm] [serif] [times] [default]
\definetypeface [postscript] [ss] [sans] [helvetica] [default] [rscale=.9]
\definetypeface [postscript] [tt] [mono] [courier] [default] [rescale=1.1]
\switchtotypeface [postscript] [11pt]
```

1.4 Summary

All in all, what I described above gives me a lots of font flexibility without buying any more fonts or learning about a lot of other fonts or ways of accessing them, especially since there are *many* variations within some of the font families.

Doing this exercise of summarizing the availability of all these fonts is something I would recommend to others who are also trying to mentally consolidate the font capabilities easily available to them. Maybe I'll expand the experiment some time in the future by investigating which fonts that are not in the tables in *The LaTeX Companion* I can find also come with my TEX distribution.

Still, I am looking forward to the day when I have XeTEX (http://scripts.sil.org) installed and working which I hope will allow me to use more fonts, also without much effort.

2 Another benefit of using TEX

As mentioned in my 2006-4 column (http://www.tug.org/pracjourn/2006-4/walden, one reason for using TEX is that one can choose the text editor best suited to one's needs. Another reason for doing typesetting with a system such as TEX where the editor is separate from the typesetting engine (unlike Word or InDesign) is that the markup is explicit (unlike typical use of Word or InDesign). I keep discovering additional reasons why it is useful to work with explicit markup as TEX uses. A capability I used recently was inserting another processor between my editor and the TEX compiler in my work process for one project on which I am working.

2.1 TUG interview series

For the past two and a half years I have been doing interviews for the TUG Interview series (http://tug.org/interviews). The interviews have been carried out as a sequence of plain text emails with me and the interviewee alternating sending questions and answers. Once an interview has been close to complete, I have edited it a little and then converted it manually to HTML for final review by the interviewee and eventual posting to the interview website.

Recently, however, Karl Berry (TUG president) and I have been contemplating creating a book of interviews with any sales proceeds going to benefit TUG. For this we will have to convert the interviews from HTML to TEX or LATEX.

2.2 Switching between HTML and LATEX using m4

To save an extra conversion step with the remaining interviews before we have enough interviews for a book of interviews, I decided to start doing interviews using m4 macros. For more about m4, see:

http://www.gnu.org/software/m4/

http://en.wikipedia.org/wiki/M4_(computer_language)

http://www.gnu.org/software/m4/manual/

Suffice it to say that among macro processors, m4 is powerful and not embedded in some other system (as Knuth's macro processor is embedded in the TEX engine). (See also the Endnote at the end of this column.)

The m4 macros can be defined to produce HTML for the website or to produce TEX/LATEX for the eventual book. This provides the desired capability of being able to insert another processing system in between the interview editing phase and the interview typesetting phase.

I have continued editing the interview source file using my usual editor, WinEdt, but now use m4 macro calls of macros defined to produce HTML for the interviews rather than inserting HTML markup explicitly; see the "m4 definitions to produce HTML" link and the "Example interview source file" link on the HTML page for this column, and the resulting HTML file at http://tug.org/interviews/interview-files/dick-koch.html. (The .m4 files shown by those links temporarily have .txt extensions so they open in a plain text editor when you click on the link.)

I create the .m4 file in WinEdt. Then I compile it under cygwin (http://en.wikipedia.org/wiki/Cygwin) on Windows XP with the command

```
c:/a-files/m4/bin/m4.exe htmldefs.m4 name.m4 >name.html
```

To generate LATEX, I have a different set of definitions at the beginning of the .m4 file as shown in "m4 definitions to produce LaTeX" on the HTML page for this paper, and I compile it with the command

```
c:/a-files/m4/bin/m4.exe texdefs.m4 name.m4 >name.html
```

and, in turn, I compile the name.tex file with PDFlatex.

The example in the LATEX source file is very crude. We will have to significantly refine and expand these definition to convert the actual collection of interviews for the book into LATEX.

2.3 Simplifying command processing

I actually execute those two different commands by executing two different little files under cygwin; see the files named `m4e.txt` and `m4et.txt` in the links list on the HTML page for this column. There may be a better way to execute this under cygwin, but the way I do it is to give the command

```
. m4e.txt
```

where the period says to execute the following file like the contents of the file had been given on the command line. The file can be reexecuted using the command

```
!!
```

as long as no other command has been given in between.

I suppose it is possible to cascade such commands under cygwin to expand the m4 macros and then compile the resulting LaTeX in one command, but I'm not familiar enough with the command shell within cygwin to do that.

2.4 The m4 approach versus other alternatives

Obviously, if one was using very complicated HTML or LaTeX, then it might take a lot of work to develop the two sets of m4 macros and it might well be easier to use some other capability to be able to generate either HTML or LaTeX for a common source file; an example might be TtH (http://hutchinson.belmont.ma.us/tth/), LaTeX2HTML (http://www.latex2html.org/), or TeX4ht (http://www.cse.ohio-state.edu/~gurari/TeX4ht/mn.html). However, with the small set of definitions we need for the interview series, using m4 to target either HTML or LaTeX seems to be a sensible way to go.

3 More about self-publishing my book

In my columns in issues 2006-2 (http://www.tug.org/pracjourn/2006-2/walden), 2006-3 (http://www.tug.org/pracjourn/2006-3/walden), and 2007-1 (http://www.tug.org/pracjourn/2007-1/walden), I described various TeX techniques I used to self-publish a book. I thought my 2007-2 column included my final notes on this, however I was wrong. I have more to report in the following subsections.

More generally, we often talk about TeX's place in the world of typesetting and publishing, in particular why more publishers don't use it (and why many publishers discourage its use). It occurs to me that the world of self-publishing is a place where we should be trying to teach people about the value of TeX. Self-publishers are often people who are already doing a lot of the work of bringing

their writing into publication, and perhaps doing their own typesetting already, and are perhaps not so subject to the constraints of traditional publishers.

I'm not sure how we should best do such promotion of TEX in the self-publishing world. TEX user John Culleton (http://www.tug.org/interviews/interview-files/john-culleton.html) does a good bit of it in the Yahoo Self-Publishing discussion group (http://groups.yahoo.com/group/Self-Publishing), where I continually see how-to questions about things people are having trouble with that are simple to do in TEX. I do believe that the self-publishing world is getting larger as information technology allows disintermediation in publishing as it is doing in so many other fields.

My own intention to keep reporting on what I have learned about self-publishing to readers of this journal about TEX use, and I am trying to get a local Sandwich, Massachusetts, writers group to let me give them a presentation on self-publishing (and TEX). I also am maintaining a separate document that summarizes what I have learned about self-publishing — http://www.walden-family.com/public/notes-on-self-publishing.pdf).

3.1 Converting the book for printing by Lightning Source

As described in my previous columns, the first two printings of my book, *Breakthrough Management*, were done by sending the PDF file of the interior pages of the book to the printer who took the 6 by 9 inch text blocks from the middle of 8.5 by 11 inch pages in the PDF file and placed them on 6 by 9 inch pages for printing. The printers also made slight adjustments I asked for regarding the margins on the printed pages by sliding my entire 6 by 9 inch text block up or down and in or out on the printed pages. The third (corrected) printing of the book (done recently) was done in the same way.

However, in July I decided also to make the book available via Lightning Source Inc. (LSI). LSI is a very large print-on-demand company that is owned by the Ingram Book Group which says it is the "world's largest wholesale distributor" of books. Consequently, anything printed by LSI is supposed to go into the Ingram catalog which in turn means that Amazon and other on-line or traditional book stores can order copies of books printed by LSI.

LSI's specifications said that I had to send them a PDF file with the pages already set up for printing a 6 by 9 inch book. To do this I changed the parameters

for my calls to the geometry package (http://www.tug.org/teTeX/tetex-texmfdist/doc/latex/geometry/geometry.pdf) as described below.

The original PDF files with 6 x 9 inch text blocks on 8.5 by 11 inch pages were produced with the following commands (I'm not sure why I used metric dimensions):

```
\RequirePackage{geometry}
\geometry{paper=letterpaper,lmargin=4.75cm,
   rmargin=4.75cm,tmargin=4.3cm,bmargin=5.57cm}
```

I thought I should be able to just calculate the new parameters for the geometry package, and started doing that calculation as follows:

```
8.5 - old left margin - old right margin => X
6 - X => Y
Y/2 => new left and right margins
```

That worked perfectly. I then turned on the two-sided geometry option which makes left and right be outer and inner and kept sum of the horizontal margins the same while changing to the minimum outer size, thus leaving more space for at the inner margin for binding.

Next I did the following calculation to obtain new top and bottom margins scaled to put the text block at the same place on the smaller pages:

```
old top margin + old bottom margin => Z
11 - Z => W
11 - W => V
old top margin/Z => a
old bottom margin/Z => b
V*a => new top margin
V*b => new bottom margin
```

Unfortunately, that didn't work exactly right, and I had to adjust this by trial and error to get the pages to be the right length. (There was something else I was not compensating for in the above calculation, e.g., headers may not be part of margin calculation.)

In any case, my trial and error adjustments on the results of the above calculation got TEX to perfectly reproduce the pages of text from the previous printings

of the book on the new page size in the PDF file. I added crop marks, which were hard to see because they are at the corners of the 6 x 9 page. I also did a final adjustment, keeping the sum of the top and bottom margins the same while slightly increasing top and decreasing bottom.

The following are the package options I used for what I ended up sending to LSI (the silliness of using metric dimensions for English system page sizes now becomes obvious):

```
\geometry{twoside,paperwidth=6in,paperheight=9in,lmargin=1.88cm,
   rmargin=1.27cm,tmargin=2.289606716cm,bmargin=2.487143284cm}
\RequirePackage[cross,height=9truein,width=6truein,center]{crop}
\RequirePackage{layouts}
```

I have long been using my own class file which is an augmentation and modification of the normal book class which I call using

```
\documentclass[final]{btbook}
```

For the version of the book going to LSI, I created a different class file with the changed described above and called with with

```
\documentclass[final]{btbookLSI}
```

I'm not sure this is the best way to parameterize which page size is being generated, but it's seems like an OK approach.

I don't recommend these class files as examples of good class file writing, but anyone who wants to look at them can access copies of them via the links on the HTML page of this paper. (The actual class files have `.cls` extensions rather than the `.txt` extensions used with this column to facilitate opening the files by clicking the link.)

3.2 Signing up to use LSI

Signing up to do printing with LSI is done via their website (http://www.lightningsource.com), but you also can talk by email and phone with a customer service person assigned to your account. She was very helpful to me as I navigating their web-based forms.

Once I had the interior pages and cover adjusted to meet LSI's specs for PDF files being submitted, I uploaded cover and interior via their web-based process. The status of my account was constantly noted in my LSI account, e.g., "waiting for uploads," "problem with cover (contact Customer Service Representative)," "proof sent and awaiting approval," and "proof approved."

The problem with my cover was a mistake on my part; I had placed the cover art horizontally on a vertical page, and the sides of the cover were cropped off. So, I had to resubmit the cover.

Costs were $50 for cover, 5 cents a page for the 280 pages of the book, and $40 for any revision, including resubmission of my cover. It also cost $30 to have a proof sent to me but that included overnight express shipping.

The cover art looked fine, except it was a glossy coating rather than the matte coating I prefer (there is no option for a matte coating from LSI). The interior printing was definitely inferior to what was done by Ames On Demand (http://www.amesondemand.com) which did the second printing of my book: the text was not as sharp and the B&W images were darker and muddier. This printing was with LSI's purportedly new presses which are supposed to do better graphics; hopefully the improved software which is still being installed with further improve the image and print quality. Nonetheless, it is good to have the book available via LSI, and I approved the proof.

I also asked Lightning Source Inc. to make my book available through their UK branch so that I can order books printed and shipped from there if that will reduce the cost to get books to European customers. This worked smoothly without me having to pay anything more or upload my files again. Also, the printing from Lightning Source in the UK was slightly better than from LSI in the US. Within a day, the book was listed with http://www.amazon.co.uk (and Amazon in Austria, Germany, France, and Japan), and a few days later the book was also shown on the Amazon UK website as being for sale from two other book stores which were undercutting the Amazon UK pricing for the book. I had told LSI to list my book for US$30, 17 GBP, and 24 Euros, with a 55 percent wholesale discount in all cases, but Amazon UK soon dropped its price to close to the low price from the competing UK book store. No doubt Amazon has a program constantly manipulating its books prices to try to maximize profits. (At this point I am still waiting for the book to be listed at Amazon.com in the US.)

In parallel with setting up to work with LSI, I sent the new cover and interior files to Ames On Demand where I will continue to order books a carton (e.g., 25

or 30 books) at a time and mostly use these for filling orders that come in via my website.

3.3 Another option with self-publishing

In the summer of 2007, I began working with with a group in Hungary to provide them with translation rights to the book, for which I still plan to act as publisher in name but with them paying for the translation, retypesetting, and printing, and doing local selling. One of the advantages of self-publishing is that I completely control the rights to the book and can do anything I want without having to get permission from a publisher or convince a publisher to do something.

3.4 More about receiving payments

Originally I had set thing up so I could receive payments for the books I sold from my book website (`www.walden-family.com/breakthrough`) only via PayPal — in particular only via PayPal for credit cards which can be done without becoming a PayPal member.

However, my experience with people trying to use a credit card via PayPal from some foreign countries, e.g., Thailand, is that it can be *hard* to do. They have to do extra work to get their credit card verified by PayPal which takes talking to their bank, etc. In two cases, we have gone around and around with me trying to tell them how to use PayPal within only a credit card, and not a PayPal account, and eventually I gave up, especially since I couldn't see what was happening with PayPal at their end and thus couldn't give good instructions. I opened a bank account that has no significant money in it for these cases and allow wire transfers into it (which apparently is easy for lots of non-US countries). As soon as I see the money in the bank account, I transfer it to my regular bank account (which I don't tell people in foreign countries how to access), and mail the book to them.

Acknowledgements

William Adams reviewed the accuracy of my statements about Word and InDesign. Elizabeth Dearborn coached me in getting my layout ready for LSI to print.

TPJ issue editor Francisco Reinaldo made many useful comments for improving the clarity of this column. Yuri Robbers spotted many typos and made other suggestion for improvement. Karl Berry and Aditya Mahajan helped me access some of the fonts in ConTEXt.

Biographical note

David Walden is retired after a career as an engineer, engineering manager, and general manager involved with research and development of computer and other high tech systems. He holds an undergraduate math degree and completed a graduate school sequence of courses in computer science. More history is at www.walden-family.com/dave.

Endnote

The history of m4 makes me feel sentimental about using it. It is derived significantly from Christopher Strachey's GPM ("A General Purpose Macro generator", *Computer Journal* 8,3 (1965), pp. 225–241) which I reimplemented for the PDP-1d at Bolt Beranek and Newman in 1967. M4 itself was also used in the implementation of the Ratfor programming language which I and a small team of programmers used in the late 1970s to implement InfoMail, the first multi-platform email system. We used Ratfor because it produced Fortran code that could be compiled on any operating system and thus provide the computer and operating system independence we needed. For more about Ratfor, see one of the following:

http://sepwww.stanford.edu/software/ratfor.html
http://en.wikipedia.org/wiki/Ratfor
http://wolfram.schneider.org/bsd/7thEdManVol2/ratfor/ratfor.html