Tools for Publishing LaTeX Documents on the Web

Julius O. Smith III

Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University, Stanford, California 94305 USA

Abstract

This paper documents the author's configuration for generating online publications (such as this one) in HTML and PDF formats from LaTeX source. A self-contained directory tree is provided which includes an example conference paper, journal article, and book. Typing "make install" in the top-level directory conveniently generates a website containing the three example documents in both formats, with links to the "hardcopy" formats appearing at the bottom of each HTML page. Additionally, a slightly updated version of `latex2html` is provided that fixes some bugs affecting the examples of this paper.

Contents

1 Introduction

This note describes software tools for publishing on the Web in HTML and PDF formats, from LaTeX source. As an example, the website `http://ccrma.stanford.edu/~jos/` (including this document you are reading) was generated from source using a single "`make`" command in the source root directory. These tools evolved on Red Hat Linux distributions since 1999, and they should work equally well on any Linux or UNIX system including LaTeX,[1] BibTeX,[2] and `latex2html`.[3] They should also work, with minor adaptations, under Windows, e.g., using the CygWin port of the GNU tools.

Tools used here include

1. LaTeX, TeX, and BibTeX— for processing the document source, including

   - `dvips` — for converting LaTeX output to PostScript, and
   - `ps2pdf` — for converting PostScript to PDF format.

   On a Red Hat Linux system, in particular, all of the above and more are contained within the following software packages:

   - `tetex-latex`
   - `tetex`
   - `tetex-dvips`
   - `tetex-fonts`
   - `tetex-xdvi`

   For example, the command "`yum install tetex-latex`" will install the first four of the above packages, if they are not already installed.

2. `latex2html` — for generating HTML versions. Installation of `latex2html` is described in §3 below.

3. `~/.latex2html-init` — a custom initialization file provided herein.

4. A `Makefile` collection for compiling and installing documents in HTML and PDF formats (also provided herein).

Regarding PDF format generation, note that it is possible to compile LaTeX directly into PDF using `pdflatex`, which is included in the `tetex-latex` distribution. However, at the time of this writing, PostScript figures are not supported by `pdflatex`. Since the author uses PostScript figures extensively, this option will not be discussed further (although it is possible to convert one's PostScript figures to PDF figures and proceed with `pdflatex`).[4]

[1] `http://www.latex-project.org/`

[2] `http://www.ecst.csuchico.edu/~jacobsd/bib/formats/bibtex.html`

[3] `www.latex2html.org`

[4] I have often made use of the editability of PostScript files to repair, port, and otherwise upgrade PostScript figure files. I have not had much luck trying to edit PDF figure files, other than what can be accomplished using Adobe Acrobat on a Windows system. I should note that, in part for this reason, I have never actually tried converting my figures to PDF and using `pdflatex`. Another reason is that I am quite satisfied with the results of generating PostScript and converting to PDF.

I will henceforth assume the reader is familiar with the use of LaTeX, BibTeX (optionally), and `latex2html`, since those are documented elsewhere [4, 1, 2, 3]. Therefore, this article will focus mainly on simple working examples, the Makefiles, and the custom `.latex2html-init` file.

2 Features

Here are the main features of the `make` files and `latex2html` init-file described in this article:

1. All documents are unified under a single set of LaTeX format settings (which are easily overridden in individual documents).
2. Several documents may be compiled and/or installed at once.
3. HTML and PDF versions of each document are generated automatically from one LaTeX source.
4. Every generated HTML page (optionally) has a link inviting the reader to download the current document in PDF format.
5. If the document is part of a larger work, such as a book, a configuration variable can be set in the document's `.l2h` file which causes a link to the larger work to be added at the bottom of each HTML page. (The document's `.l2h` file supplements and overrides the more global `.latex2html-init` file.
6. A full bibliographic citation is included in the footer of each page in all formats.
7. A copyright notice, name, address, etc., are generated at the bottom of each HTML page.

3 Installing LaTeX2HTML

You should already have `latex` and `bibtex` installed on your system. You may also have `latex2html` installed (normally in `/usr/bin` on Unix machines). However, there are at least five bugs in the latest release of LaTeX2HTML that I have found it necessary to fix. My own "forklet" of LaTeX2HTML that either fixes or gets around these bugs is available at GitHub:

`https://github.com/josmithiii/l2hmj`

This variant consists of simplest fixes and workarounds to known bugs in version 2002.2.1-6 of LaTeX2HTML, where the '-6' refers to the version containing patches for the Red Hat Fedora Core 6 release. (Hopefully the Red Hat patches are beneficial for all platforms.)

If you can't install your own version of l2hsp for any reason, continue on to the following subsections, where some alternatives are suggested (and the bugs described in more detail). Otherwise, install my "l2h forklet" and skip to the next section, "Installing and Compiling the `webpubdemo` Directory".

3.1 Overriding Bugs in Your Init File

Three of the bugs are in the `latex2html` perl script itself, and these are easily fixed by overriding the affected subroutines in your init file `~/.latex2html-init`. A simplified version of my own init file containing these fixes may be found here[5].

3.2 Empty Figures

Somewhere around ghostscript-8.63 (and presumably somewhat earlier), all images generated by the latex2html's pstoimg utility became empty. Some debugging revealed that the problem could be worked around by disabling the `$have_geometry` variable in pstoimg. This results in no translation or cropping of the image when converting from PostScript to `.pnm` using `gs`. The cropping is then done later by pnmcrop. Fixing this bug will speed up image translation. I first noticed the problem in early 2009 (on Fedora 10 and Mac OS X).

Here is the relevant diff for pstoimg:

```
diff -cb pstoimg-prv pstoimg
*** pstoimg-prv Sat Dec 27 04:54:49 2008
--- pstoimg Fri Mar 13 03:39:48 2009
***************
*** 961,967 ****
             $bbw = int($bbw + 0.99);
             $bbh = int($bbh + 0.99);
             $GEOMETRY = "${bbw}x${bbh}";
!            $have_geometry = 1;
             last;
             }
           }
--- 968,974 ----
             $bbw = int($bbw + 0.99);
             $bbh = int($bbh + 0.99);
             $GEOMETRY = "${bbw}x${bbh}";
! #jos:           $have_geometry = 1;
             last;
             }
           }
***************
*** 1001,1007 ****
      $bbw += 10;  # add a 5pt margin for safety
      $bbh += 40;  # add a 20pt margin for safety
      $GEOMETRY = "${bbw}x$bbh";
!     $have_geometry = 1;
      }
```

[5] `http://ccrma.stanford.edu/~jos/webpub/dot-latex2html-init`

```
    if($have_geometry) {
      $gs_size = "-g$GEOMETRY ";
--- 1008,1014 ----
      $bbw += 10;  # add a 5pt margin for safety
      $bbh += 40;  # add a 20pt margin for safety
      $GEOMETRY = "${bbw}x$bbh";
! #jos:    $have_geometry = 1;
      }
    if($have_geometry) {
      $gs_size = "-g$GEOMETRY ";
```

Needless to say, this is just a workaround for the problem, and really the arguments to ghostscript should be properly debugged.

3.3 Fixing Grey Backgrounds in Equation Images

Another important bugfix is needed in the perl script `pstoimg` (normally located in `/usr/bin/`). It's a one-character change described by the following "diff":

```
diff -cb pstoimg /usr/bin/pstoimg
*** pstoimg Sat Dec 16 15:55:09 2006
--- /usr/bin/pstoimg Sun Dec 11 03:14:00 2005
***************
*** 1021,1027 ****
        }
      my $had_nonwhite;
      if($opt{white}) {
!       $had_nonwhite = ($ps =~ s/(\n\d+ \d+ bop gsave) \d*\.\d+ \
        (TeXcolorgray clippath fill grestore)/$1 1 $2/s);
        }
      $ps_changed = $had_papersize || $had_nonwhite;
      if($ps_changed) {
--- 1021,1027 ----
        }
      my $had_nonwhite;
      if($opt{white}) {
!       $had_nonwhite = ($ps =~ s/(\d+ \d+ bop gsave) \d*\.\d+ \
        (TeXcolorgray clippath fill grestore)/$1 1 $2/s);
        }
      $ps_changed = $had_papersize || $had_nonwhite;
      if($ps_changed) {
```

The change is to remove the '`\n`' from the search string that detects whether or not the PostScript image-file being processed has a gray background or not. Without this fix, equation images all have a grey background, even when the option `$WHITE_BACKGROUND` is set nonzero in

~/.latex2html-init. Some years ago, the TeXcolorgray fill command always started on its own line. Since around the time of Fedora Core 2, this is no longer the case, so expecting it at the beginning of the line fails. The fix is simply to allow a match anywhere in the line.

3.4 Eliminating Black Rules Under Equation Images

The fifth problem is random underbars and sometimes left-bars on the boundaries of equation images. This only happens if the option '-Ppdf' is passed to dvips for image creation, but that is unfortunately wired in as a default in the standard distribution (at least Red Hat's).

The problem here is due to the relatively new file

```
/usr/share/texmf/dvips/misc/alt-rule.pro
```

which defeats LaTex2HTML's algorithm for stripping off these temporary black bars during image processing.

One way to avoid this problem is to eliminate the '-Ppdf' option for dvips in the latex2html configuration file

```
/usr/share/latex2html/l2hconf.pm
```

My line reads as follows:

```
$DVIPSOPT = ''; # -Ppdf pulls in alt-rule.pro which causes black eqn underlines
```

Another work-around is to modify /usr/share/texmf/dvips/misc/alt-rule.pro to round the dimensions of the marker rules to an integer number of pixels before drawing them (which is how it worked before alt-rule.pro appeared). A copy of alt-rule.pro with the one-line fix installed may be found here.[6] (The line beginning with transform is the added line.)

A *true fix* (as opposed to the above workarounds) would be to modify LaTex2HTML so that it generates rules having dimensions which are always an integer number of pixels in device coordinates. For example, search for "\\def\\centerinlinemath" in the file /usr/bin/latex2html (which generates code into the file images.tex when latex2html is run), and observe the \\vrule code and other rules (the code is easier to read in images.tex). I believe zero-width rules are ok, since that's supposed to map to a width of 1 pixel in print-device coordinates (although nowadays, one pixel can be quite invisible). The problem, if I recall correctly, occurs when the length of the rule is not an integer number of pixels. In that case, alt-rule.pro uses gray-levels to achieve a more accurate image, but then pnmcrop, which is looking for solid black rules, fails to recognize them completely and leaves one or both of them in the image. Another proper fix would be to recognize the gray-edged rules using more robust image detection methods, such as cross-correlation, but that would add a lot of computation. For further discussion, see the mailing list entry on this topic. Another true fix would be to modify the method by which images are forced to be embedded in a rectangle of a certain size by placing black rules on the left and bottom of each image and then trying to crop them off later using pnmcrop. To see all this in action, run with debugging turned on via "latex2html -d".

[6] http://ccrma.stanford.edu/~jos/webpub/alt-rule.pro

3.5 Why Fork the LaTeX2HTML Distribution?

I have reported all of the above-cited bugs to the official mailing list, to some individuals, and to various bugzilla bug reports, but they were never incorporated into an official "upstream" release. Also, there didn't appear to be any new releases since 2002, and even that one seemed to be an unofficial personal version. The original CVS repository trunk appears to have disappeared. Since I had a funded research project that depended on all team members having a working version of `latex2html`, I decided to create the above version and post it on my website in 2006. There is now a 2008 version which I have not tried because I have no problems that warrant trying to upgrade. More interestingly, Paulo Ney de Souza has released a new version in 2017 that should contain all my bugfixes (as we corresponded and he knew about my fork):

`https://www.ctan.org/pkg/latex2html`

Installing and trying out this one is high on my list of priorities!

4 Installing and Compiling the `webpubdemo` Directory

In the Web version[7] of this paper, there is a demo directory contained in the file `webpubdemo.tar.gz`. Copy this file to your computer, and unpack it using the command

```
tar -xvzf webpubdemo.tar.gz
```

in the directory containing the file. This should create a subdirectory named `webpubdemo`.

4.1 Installing `.latex2html-init`

If you don't already have a `.latex2html-init` file, simply type

```
cd webpubdemo
cp dot-latex2html-init ~/.latex2html-init
```

If you do have a `.latex2html-init` file, you can probably just *append* the new one to your existing one. (Insert it just before the last line if it is simply "1;". Such a line provides a "true" return value for Perl.)

Since `latex2html` init files are executed as Perl scripts, appending the new one may override some preferences you had set. However, you should be able to try out the new preferences and then delete the new lines you don't want.

No changes to `dot-latex2html-init` are required for simply trying out a build of the `webpubdemo` examples. However, for future use, the following edits are desirable:

1. edit the $MYNAMEHOME variable to point to your Web home page,

[7] `http://ccrma.stanford.edu/~jos/webpub/`

2. edit $MYNAMEEMAIL to contain your email address,

3. change "My Full Name" to your full name, and

4. optionally change all occurrences of "MYNAME" to your initials, or whatever. This is just a variable-name prefix to avoid collisions with `latex2html` symbols. If you make this change, you need to make it also in the `.l2h` file in each of the three example subdirectories if you want them to still compile properly.

4.2 Installing LaTeX and BibTeX Input Files

Since you are presumed to have LaTeX, BibTeX, and LaTeX2HTML already installed, you should already have the environment variables $TEXINPUTS, $BSTINPUTS, $BIBINPUTS defined. In my `.tcshrc` file, for example, I have the lines

```
setenv TEXINPUTS ".:$HOME/texinputs::"
setenv BSTINPUTS ".:$HOME/texinputs::"
setenv BIBINPUTS ".:$HOME/texinputs::"
```

Thus, in all cases, the current directory is searched, followed by my `~/texinputs` directory, followed by whatever the "system directory" is in each case. (The null entry '::' denotes "default system directories" — try finding *that* in the documentation.)

Let <texinputs>, <bstinputs>, and <bibinputs> denote any valid directory you are using for each case. Then, to be able to compile the `webpubdemo` examples, you need to type, in the `webpubdemo` directory,

cp cmj.bst <bstinputs>

cp jos.bib <bibinputs>

cp index.ist <texinputs>

Alternatively, you could easily edit the examples to not need these files. However, then you would miss out on some of my suggestions!

For example, I generally prefer `cmj.bst`, created with `natbib`,[8] to any of the default bibliography formats in LaTeX. (Here, "cmj" stands for "Computer Music Journal.") For example, references are cited using up to two *author-last-names* (using "et al." for three or more), followed by the *year* (using suffixes when there is more than one paper by the same author(s) in the same year, etc.). It also handles *multiple citations* better than similar styles I've seen. I find it very helpful to see papers cited in the form "[Morse and Ingard 1967, Kinsler and Frey 1982]" rather than "[1,2]" or "(Mor1967,Kin1982)", for example, since, if I've studied the paper at all, I'll usually recognize the citation immediately and save myself a trip to the bibliography to see what it is. Finally, the bibliography generated by `cmj.bst` is *sorted alphabetically* by the first-author's last name, so that multiple papers by the same author are conveniently grouped together.

[8] `http://www.tug.org/teTeX/`

Perhaps the most unconventional feature in my bibliography file, `jos.bib`, is the use of *live hyperlinks* to publications which are available online. For example, if you click on [5] to visit the full bibliographic citation, you will see the hyperlink

> Available online at `http://ccrma.stanford.edu/~jos/kna/`

and clicking on that will take you directly to the paper. If online papers ever adopt the convention of placing a *full bibliographic citation at the bottom of each page* (which these tools do), then it is easily seen that there is *no longer any need for the bibliographic citation at all* (except in the printed version, of course).

If you are an active researcher in signal processing applied to music and audio, then `jos.bib` might give you some useful references you didn't already have (☺).

The index style file, textttindex.ist, is nothing special, but something like it is needed to create a reasonable looking index. Tagging words for the index using LaTeX's `\index{}` macro as you write is a very good habit to get into.

4.3 Building the Examples

Having installed the four "personal system input files" (and perhaps adjusting some environment variables), you are now ready to `cd` into the `webpubdemo` directory and type

```
make install
```

This should create and populate the directory `/tmp/TEST-INSTALLDIR`, which you can visit in your Web browser, e.g.,

```
netscape /tmp/TEST-INSTALLDIR/index.html &
```

You can install elsewhere by specifying the `DESTROOT` `make` variable, e.g.,

```
make install DESTROOT=/tmp/NEWTEST
```

4.4 Browsing Precompiled Examples on the Web

The result of a successful `make` may be *browsed* at

`http://ccrma.stanford.edu/~jos/webpub/TEST-INSTALLDIR/`

and *downloaded* from

`http://ccrma.stanford.edu/~jos/webpub/TEST-INSTALLDIR.tar.gz.`

The example documents lie in the following subdirectories of the `webpub` directory:

```
myconferencepaper  - example conference paper
myarticle          - example journal article
mybook             - example book
```

If the `make` succeeded, you should find HTML and PDF versions of each of three example documents, all reachable via hyperlinks from the main "home page" in the automatically generated website (in `TEST-INSTALLDIR`). Browse the HTML versions and test the "download PDF" link at the bottom of any HTML page. Note also how the automatically generated "navigation links" allow easy travel in either direction through the document, as well as the ability to go "up" all the way to the central home page at the top. When one of your HTML pages is reached via a hyperlink somewhere on the Web, the navigation links make it easy for the reader to determine the *context* of your Web page by zooming out until he or she feels oriented.

4.5 Troubleshooting

If the `make` fails, you might need to type in some explicit paths to executables such as `latex` or `latex2html`. These go at the top of `webpubdemo/Makefile.tex`.

5 Makefiles

As seen above, the primary `make` target is "install", e.g., (to publish a website on most Linux[9] systems running the Apache[10] web server):

```
make install DESTROOT=/home/httpd/html
```

5.1 Commonly Used Make Targets

Other `make` targets I regularly use include

```
make dvio       ;; make and view (‘open’) LaTeX’s output DVI file
make html       ;; make HTML version
make htmlo      ;; open the HTML version in a Web browser
make thtml      ;; ‘‘touch’’ and remake the HTML version
make hclean     ;; delete the HTML version completely
make tclean     ;; ‘‘totally clean’’ deletes all generated files
```

Note that `make html` will reuse figures from a previous `make`, if they exist. This is usually a good thing, but if they seem to be out of synch in some way, try "`make hclean html`" to regenerate them from scratch. I use `make thtml` when I've changed something outside of the `make` dependencies, such as `.latex2html-init`, and I want to force a remake of the HTML version while reusing figures for speed.

[9] `http://www.linux.org`
[10] `http://www.apache.org`

5.2 All Make Targets

Here are (almost) all `make` targets (see `webpubdemo/Makefile.tex` for exactly what they do):

```
make            ;; make DVI file only
make dvi        ;; make DVI file only
make dvio       ;; open DVI file (making it if necessary)

make ps         ;; make PostScript file
make pso        ;; open PostScript file (making it if necessary)

make gz         ;; make compressed PostScript file (.ps.gz)
make pdf        ;; make PDF file

make html       ;; make HTML version
make htmlo      ;; open and view HTML file (making it if necessary)

make all        ;; make all installed products (.pdf, .ps.gz, HTML)

make clean      ;; clean up, leaving products and aux files
make rclean     ;; make "really clean" = everything but products
make tclean     ;; make totally clean = rclean + products
make fresh      ;; = make fresh

make index      ;; creat LaTeX index file ('.ind' file)

make t          ;; touch main .tex file and then remake
make test       ;; just run latex once, forgetting about dependencies

make htgz       ;; Gzipped tarfile of HTML version, breaking links
make htgzl      ;; Same as htgz, preserving links

make help       ;; print this documentation
```

5.3 A Word About `.TIMESTAMP` Files

You may notice that when a LaTeX source file is compiled, *four* "hidden" time-stamp files are created, e.g., `.LATEX-PASS3-TIMESTAMP`. These files are necessary to keep track of all the recompiling needed by LaTeX, especially when used with BiBTeX. The standard drill is "`latex, bibtex, latex, latex`." If only the `.bib` file is touched, we only need "`bibtex, latex, latex`," and so on. For years, I tried to write `webpubdemo/Makefile.tex` using *no* time-stamp files, trying to use generated files such as the `.aux` and `.bbl` files in all `make` dependencies. However, I have recently finally given up on this quest, since it never worked completely right, and the use of time-stamp files has greatly simplified my life. If you can figure out how to get rid of all time-stamp files and get all the dependencies right, please send me a copy of your `Makefile`!

6 More About `.latex2html-init`

As summarized above in §4 (and in the file `webpubdemo/INSTALL`), the file `webpubdemo/dot-latex2html-init` should be copied into your login directory, renamed to `.latex2html-init`, and edited to install your name, user name, address, and the like.

This init file adds the following features to `latex2html`:

- A complete bibliographic citation of the current document at the bottom of each HTML page, fully hyperlinked.
- Links at the bottom of each HTML page for downloading a PDF version of the current document.
- Copyright and other notices at the bottom of each HTML page.

In addition, various `latex2html` defaults are overridden. All overrides are merely my preferences except for `$TRANSPARENT_FIGURES=0`, which is explained further below.

When editing `.latex2html-init`, keep in mind that it is a Perl[11] script executed by `latex2html` after its own initialization. As a result, since `latex2html` is just a Perl script itself, your `.latex2html-init` can be used to override any variables or functions in `latex2html`. Also note that the `.l2h` file in each document directory can further override settings on a document by document basis. See, for example, `webpubdemo/mybook/mybook.l2h` for the sorts of things expected in the `.l2h` file. For your reading convenience, that file is listed below:

```
# Perl script webpubdemo/mybook/mybook.l2h
$MYNAMEDOCDIR = "mybook";
$MYNAMEDOCTITLE = "My Book Title";
$MYNAMEDOCAUTHORS = $MYNAME;
$MYNAMEDOCPUBINFO = "Bibliographic citation goes here";
$NOHARDCOPIES = 1; # Reasonable choice for books in print
$ADDRESS = &make_myname_address;
1;      # This must be the last line
```

Note the `$NOHARDCOPIES` variable which is used to suppress the PDF link in the HTML. Read the `.l2h` files in the other examples for more suggestions like this.

As enumerated in §4, customization of `.latex2html-init` usually goes as follows:

1. Edit `$MYNAMEHOME` to point to your Web home page, under which all your publications will reside. For example, mine is

   ```
   $JOSHOME = "http://ccrma.stanford.edu/~jos";
   ```

2. Edit `$MYNAMEEMAIL` to hold your email address, if desired. I have a URL to a web form for this.

[11] `http://www.cpan.org`

3. Change "My Full Name" to your name. This affects one line. Mine looks like

```
$JOS = "&lt;A href=\"" . \$JOSHOME . "/\"&gt;Julius O. Smith III&lt;/A&gt;";
```

   Note that double-quote must be quoted with a backslash (\") in a Perl string.

4. Change all occurrences of "MYNAME" and "myname" to your initials, or whatever. (This is the variable-name and subroutine prefix string.) I use "JOS", so my `.l2h` files look as follows:

```
# Perl script webpubdemo/mybook/mybook.l2h
$JOSDOCDIR = "mybookdirectory";
$JOSDOCTITLE = "My Book Title";
$JOSDOCAUTHORS = $JOS;
$JOSDOCPUBINFO = "Bibliographic citation goes here";
$ADDRESS = &make_jos_address; # Call the subroutine in the init file
1;      # This must be the last line (return true for success)
```

6.1 Figure Backgrounds

Ideally, we would like to use `$TRANSPARENT_FIGURES=1` in `.latex2html-init` and not set the page background to white. However, I have not been able to make this work. The failure is that I get various filled blotches in the figures. By setting both the figure and document backgrounds to white (`$TRANSPARENT_FIGURES=0`, `$WHITE_BACKGROUND=1;`, `BGCOLOR="#FFFFFF"`), this problem is made invisible. You may not have this problem. I think I have it because many of my PostScript figures use white-filled boxes to effectively "erase" lines (in the old NeXT Draw program); the transparent background conversion is supposed to be able to convert all occurrences of white to transparent, but I haven't been able to get this to work throughout my PostScript figures.

To try out transparent figure backgrounds, set `$TRANSPARENT_FIGURES=1`, and delete the string `""BGCOLOR="#FFFFFF""` in the `$BODYTEXT` variable. You may also need to set `$WHITE_BACKGROUND=1;`, I don't remember. Nothing works for me except white backgrounds for all.

6.2 Color Overrides

The `$BODYTEXT` variable in `.latex2html-init` is set to

```
$BODYTEXT = "BGCOLOR=\"#FFFFFF\"
        TEXT=\"BLACK\"
        LINK=\"BLACK\"
        ALINK=\"BLUE\"
        VLINK=\"GRAY\"";
```

Thus, in addition to overriding the background color, the color scheme for text and links is prescribed. Both text and non-visited hyperlinks are black. I prefer this because I make extensive use of hyperlinks, and setting them to black along with the text makes them less obtrusive. Feel free to delete or modify this as desired.

Note that the use of HTML `<BODY>` tag parameters is "deprecated", and that we are all supposed to set things like this in Cascading Style Sheets (.css files) or XML Style Sheets instead. I personally hope the above `BODY` tag settings will continue to override any style sheet settings.

6.3 Miscellaneous Settings

I set the `$PAPERSIZE` variable to `b0` because that is the largest setting. HTML does not care how large the page is, and I actually had a "clipping" problem in the past with some large pages, until I hit upon this solution. This problem may be gone now, (if it was an early `latex2html` bug), but I don't see how the setting can hurt anything.

The `$ICONSERVER` variable is to a local directory containing a small collection of my own icons (names having the form `a_*.gif`) along with all the icons that came with `latex2html`, version 99.2-beta6. If you encounter missing icons using a future release of `latex2html`, you should be able to comment-out the setting of `$ICONSERVER` and copy the local icons into the `icons` subdirectory of your `latex2html` installation, `$LATEX2HTMLDIR/icons/`.

To take out my icon overrides, delete the `$ICONSERVER` line and also delete the six lines beginning with `$icons` and `$iconsizes`.

Similarly, deleting the subroutine `common_navigation_panel` (which overrides the way navigation panels are laid out) will restore the default `latex2html` version of this function. However, doing this will remove the top-level "up" link in each document which takes the user to your master home page.

6.4 Global Website Index

In the `common_navigation_panel` subroutine override in the supplied `.latex2html-init` file, there is a commented-out section which provides a link to your "global index file". These lines are commented out because you probably don't have such a file. See `http://ccrma.stanford.edu/~jos/GlobalJOSIndex` for an example of such a file. I generate this file using a Perl script which spiders my website and creates a hyperlink to every .html file it finds, using the `TITLE` tag as the anchor text. If you title your web pages well, the "global index" can be a useful resource, and a fast way to "get around" when you know what you're looking for. (Using the supplied `.latex2html-init` defaults, HTML page titles will consist of paper *titles*, *section titles*, and *subsection titles*. Choose your section and subsection titles carefully!)

7 Using your Own Home Page

In the demonstration example source, there is a placeholder "home page" located in

`webpubdemo/other/index.html`.

If you don't have a home page yet, you can modify this one to make it your own. It should be fairly self-explanatory, even if you don't already know HTML yet.

If you do have a home page (or even an entire website) already, you can just delete `webpubdemo/other/index.ht` and point installation to your website. Then you have to add explicit links to each document subdirectory somewhere within your Web pages. In `webpubdemo/other/index.html`, the links to the examples are handled as follows:

```
<H1>Online Publications</H1>
<H4>(generated from LaTeX source using the tools described in this document)
</H4>
<P>
<UL>
<LI><A HREF="TEST-INSTALLDIR/myconferencepaper/index.html">Example Conference Paper</A>
<LI><A HREF="TEST-INSTALLDIR/myarticle/myarticle.html">Example Journal Article</A>
<LI><A HREF="TEST-INSTALLDIR/mybook/mybook.html">Example Book</A>
</UL>
```

Thus, each automatically installed document appears in its own subdirectory at the same directory level.

Any files in the `webpubdemo/other` directory will be copied (recursively, in the case of subdirectories) into the installation directory. Use this directory for images and other resources that are used by multiple documents. Document-specific extras can be handled by the document's `Makefile`, as the `myarticle` example illustrates:

```
# File webpubdemo/myarticle/Makefile
NAME = myarticle
# Software (usually in Matlab) for generating all figures in the paper:
EXTRAS_NAME = $(NAME)_extras
OTHER_GOODIES = $(EXTRAS_NAME).tgz original.ps.gz
include ../Makefile.tex

rclean:: clean
- /bin/rm $(EXTRAS_NAME).tgz

$(EXTRAS_NAME).tgz:
- /bin/rm $(EXTRAS_NAME).tgz
tar -hcvzf $(EXTRAS_NAME).tgz $(EXTRAS_NAME)
```

We see that there are two "extras" installed "as is" for this paper: (1) a compressed tarfile containing software source for computing the figures in the paper, and (2) a compressed PostScript file containing the original conference version of the paper.

Other goodies which can go here include sound examples, talk overheads, raw HTML files created outside of LaTeX, and so on. Whatever you put here, *remember to place explicit links somewhere in the body of the paper.* In the `myarticle` example, the extra goodies are linked in as follows:

```
The following related items are available from the
```

```
\htmladdnormallinkfoot{Web version}{http://www.myplace.edu/\~{}mylogin/myarticle}
of this paper:
\begin{itemize}
\item Software used to generate all figures in this paper:
\htmladdnormallink{\texttt{myarticle\_extras.tgz}}{./myarticle\_extras.tgz}
\item Original conference paper (Proc.\ IXXX-95):
\htmladdnormallink{\texttt{myarticle\_original.ps.gz}}{./myarticle\_original.ps.gz}
\end{itemize}
```

Note the use of the construct '`\~{}`' which is necessary to get a printed tilde (~) in the output.

7.1 Using the HTML BASE Element

Sometimes it is useful to have one web-page pose as another. That is, a web page at one URL is a link to a web page in a different directory, and the links in that page use relative path specifications. The `BASE element`[12] of HTML 4[13] is useful for this purpose. However, it must appear within the <HEAD>! tag before any relative path-names are used, and not merely in the <BODY>. There is a variable in `latex2html` called `BASE` for this purpose. I set it in my standard procedure for setting the $ADDRESS variable:

```
$BASE = &make_josdocdir_url . "/index.html";
```

8 Cascading Style Sheets

In my own website, I use a custom style sheet `jos.css` that affects every document in my website (generated as described above). To activate use of the CSS file, I added the following line to my `~/.latex2html-init` file:

```
$STYLESHEET = "../jos.css";
```

The latest version of this file (which is quite simple), may be viewed at `http://ccrma.stanford.edu/~jos/jos.cs`
.

9 Conclusions

I hope you find these tools useful for online publishing.

References

[1] M. Goossens, F. Mittelbach, and A. Samarin, *The LaTeX Companion*, Reading MA: Addison-Wesley, 1994.

[12] `http://www.w3.org/TR/html401/struct/links.html#edef-BASE`
[13] `http://www.w3.org/TR/REC-html40/`

[2] M. Goossens, S. Rahtz, and F. Mittelbach, *The LaTeX Graphics Companion*, Reading MA: Addison-Wesley, 1997.

[3] M. Goossens, S. Rahtz, E. Gurari, R. Moore, and R. Sutor, *The LaTeX Web Companion*, Reading MA: Addison-Wesley, 1999.

[4] L. Lamport, *LaTeX: A Document Preparation System. User's Guide and Reference Manual*, Reading MA: Addison-Wesley, 1994.

[5] J. O. Smith, "Viewpoints on the history of digital synthesis," in *Proceedings of the 1991 International Computer Music Conference, Montréal*, pp. 1–10, Computer Music Association, searchable at `http://quod.lib.umich.edu/i/icmc/`, 1991, `https://-ccrma.stanford.edu/~jos/kna/`.