TUGboat online

Karl Berry and David Walden

1 Introduction

The TEXbook has traditionally been, and remains, a print journal. This was been described to some extent in Barbara Beeton's 2006 paper ("How to Create a TEX Journal: A Personal Journey", *TUGboat* 28:1, 2007, pp. 29–49, `http://tug.org/TUGboat/tb28-1/tb88beet-keynote.pdf`).

Nevertheless, in more recent years *TUGboat* has also existed online. This article sketches the evolution of the online version of *TUGboat*.

2 First steps

TUGboat has been part of the TUG web site since the web site's beginning. To a large extent, the original pages remain, with the basic description of the journal, editor's wish list, information for authors, and so on.

It was apparent from the outset that it would be useful to publish back issues on the web. Mimi Burbank began the process of scanning paper copies. In a few cases articles were reprocessed, or DVI converted to PDF, but in the main, sources, figures, and/or fonts were lacking, so no systematic reprocessing could be attempted. Scanning to PDF was the only viable approach.

As the work proceeded, Mimi created table of contents files in HTML by hand. The scanning took so much more time than writing the HTML, and was proceeding so slowly, there was no need to think about any tools.

As the years passed, additional volunteers came forward. In particular, Brooks Moses (with the support of his department at Stanford) ended up doing the bulk of the scanning. Robin Laakso in the TUG office also made significant contributions. Most of the issues were scanned by 2006, though the process was not completely finished until 2009, so the scanning overlapped with the other work we describe below. (Even now, additional work remains — some of the early issues were scanned at 150 dpi, some pages are crooked, a few pages are missing, etc.)

Once enough back issues had been scanned, the question arose of how to make the material accessible on the web. Our attempts to answer that question are described in the rest of this article.

3 Two steps toward automating generation of *TUGboat* tables of contents

3.1 *The PracTEX Journal*

In 2004, TUG, at the suggestion of Lance Carnes, sponsored a TEX conference explicitly focused on practical use of TEX. Near the end of the conference, Tim Null and Robin Laakso discussed developing a section of the TUG web site that would contain introductory material on TEX organized by category. In an exchange of emails between Tim Null and Karl Berry after the conference, Tim came up with the alternative idea of an online journal. Lance Carnes became the editor of that the journal, and other people volunteered or were recruited to join the new journal's editorial board.

The Editorial Board's planning discussions naturally were centered on issues such as the target audience, look and feel, and frequency and schedule. Meanwhile, Karl Berry hand-crafted HTML for a prototype web site for the new journal, and other members of the editorial board made many comments on it. We then embarked on a collaboration to build a production program to generate a web site similar to the prototype. The initial issue of *The PracTEX Journal* was published (posted at `http://tug.org/pracjourn`, which remains the journal's home) on January 15, 2005. This web site includes a table of contents for each issue with links to the PDFs of the papers, and author, title, and BIBTEX lists covering all issues.

The relevance of the *The PracTEX Journal* effort to *TUGboat* online is that we had to think extensively about how a program would generate the web site for an online journal, Dave wrote a lot of code, and we noted some things to be done better or differently if another journal web site generation program was ever written.

3.2 Contents, ordered by difficulty

In early 2005 Dave and Karl began discussing the possibility of creating an additional table of contents for each issue of *TUGboat* that would be organized by level of difficulty. Karl took the issue to Barbara Beeton and the other members of the *TUGboat* production group, who supported the idea. *TUGboat* Volume 25 (2005), No. 2 (*TUGboat* issue 81) was the first issue that included this additional table of contents: see `http://tug.org/TUGboat/tb25-2/cover3.pdf`. The design was discussed extensively in advance, and has had only minor refinements since that first appearance.

From the beginning, we knew we would want this information to be available online (in HTML) as well as in print (via PDF). So, we devised a format for the source file that could both be compiled by TEX to produce PDF, and would also be plausible to parse with a Perl program to generate HTML for the online contents page. We informally refer to these

dual-purpose source files as "capsule files".

The capsule files begin with some prologue definitions relevant only to TEX, then followed by the main material: a sequence of `\capsule` commands, each one being a capsule summary of an item in the table of contents. The `\capsule` entries are in the order they would appear in the printed level-of-difficulty table of contents (since Perl can do reordering more easily than TEX). The `\capsule` command evolved over time, eventually having nine arguments:

1. The difficulty rating — follows `\capsule` on first line (example given below).
2. The *TUGboat* category, possibly with a "sub-argument" `replace` or `add`, as described below. Arguments 2–7 are always alone on their lines.
3. The author(s).
4. The title.
5. A one-line description for the item.
6. The page number, possibly with an `offset` sub-argument.
7. The url of the item's PDF.
8. Optional subtitles. These can be multiple lines.
9. Optional additional text for the HTML. This can also be multiple lines.

(Some of the subtleties of the `\capsule` command's interpretation will be described in the next section.)

For ease of parsing by the Perl program, the arguments to `\capsule` are mostly on a single line (the exceptions are the last two, which can be multiple lines, as noted above). Here's an example from the present issue:

```
\capsule{Introductory}
  {Resources}
  {Jim Hef{}feron}
  {Which way to the forum?}
  {review of the major online help forums}
  {\getfirstpage{heff}}
  {/TUGboat/!TBIDENT!heff.pdf}
  {}
  {}
```

4 Writing a program to generate *TUGboat* contents pages

Even given the prior experience with *The PracTEX Journal* (described in section 3.1), the program to generate the *TUGboat* contents pages and lists of authors, titles, and categories/keywords evolved over a fairly long period. As we handled more of *TUGboat*'s 100 issues, we continually learned of more variations with which we had to cope.

4.1 The series of steps

Our memory of the evolution is as follows.

a. Dave first cobbled together a program to convert the capsule files (which only existed from issue 81 on) into tables of contents for new issues as they came out. In this way, we began to understand the task at hand and had real examples to review as we decided what to do next.

For this program, Dave reworked parts of the *PracTEX Journal* program. In particular, for this and all additional evolutions of the project he used a pair of template routines he developed (originally to generate HTML for the *PracTEX Journal* effort), in order to avoid learning about Perl's template capabilities; see `http://walden-family.com/public/texland/perl-common-code.pdf`.

b. Then we started to work on issues prior to #81. The material we had to work with was (a) the online contents pages, which existed as hand-crafted HTML (as mentioned in section 2); and (b) the so-called "`.cnt` files" created by Barbara Beeton; these are TEX files, covering (at the time) the tables of contents for all but the last few years prior to issue 81.

(Aside: The `.cnt` files are also used by Nelson Beebe to produce his *TUGboat* bibliography files. Both are available on CTAN at `http://mirror.ctan.org/info/digests/tugboat`.)

c. Where only HTML files existed, Dave converted the HTML into capsule files using a combination of Perl programming, editor macros, and manual editing. By this time, the urls for the PDFs of the individual papers already existed in the HTML for transfer into the capsule files. (When initially putting the results of the scanning online, Karl had updated the HTML contents files with these links to PDFs for individual items.)

d. For the years which had a `.cnt` file available, Dave wrote another Perl program for conversion into capsule files. In this case, Dave then looked at the HTML files for those issues and transfer the urls for the PDFs into the capsule files. Dave did this manually, looking at each PDF file as he went along, spotting instances of multiple short or related papers in a single PDF file such that the same url would be the argument to more than one `\capsule` command. (This happened because sometimes Karl had combined several items into one PDF, according to his subjective idea of when splitting would be more help than hindrance to readers.) In a few instances, we split PDF files that had previously been combined.

e. At this point, we had capsule files for all issues, and some insight into the special cases that needed to be handled. Dave then did a major renovation and expansion of the program mentioned in paragraph a. In addition to generating the tables of contents for

each issue, the new version of the program created author, title and keyword lists across all issues.

f. Dave ran the new version of the program on the latest issues (see paragraph a) to make sure those capsule files still converted correctly.

g. Starting with the first issue, Dave then ran the new version of the program on successive issues, fairly often discovering new situations the program had to be augmented to handle and discussing possible solutions with Karl. Some of these situations and solutions are described in the next sections.

h. Karl and Barbara reviewed the result and suggested additional refinements. Some of these just required fixes to the capsule files to the program. Others required changes to the program.

i. Finally, we felt ready to make the full set of computer-generated web pages publicly available. They are all linked from `http://tug.org/TUGboat/contents.html`.

Over time, as new *TUGboat* issues have been produced a little additional program maintenance and improvement has been required. Altogether there have been about 50 iterations of the program.

4.2 Program structure

The program is driven by several plain text files:

- A file showing translations of words with diacritical marks, etc., both from TEX into HTML for web site display and from TEX into plain text for alphabetic sorting.
- A file for unifying both (a) different versions of the same author's name, defining a single version of the name which is used for all when sorting items for the author list), and (b) different versions of the same *TUGboat* article category (e.g., 'Fonts', 'Font Forum', 'Font Design and New Fonts', and 'Fonts and Tools'), again defining a single version which is used for the category/keyword list.
- A file listing the *TUGboat* issues to be processed.
- The capsule file for each issue.

Examples of all the files discussed here are at `http://tug.org/TUGboat/tb32-1/tubonline`.

The program reads in the first two files to prime its data structures and then begins processing the third file, one issue number at a time, which in turn involves processing the capsule file for that issue. As each capsule file is processed, the necessary information is saved for the title, author, and keyword/category lists. The HTML contents page for each individual issue is also output in turn. After all the issues have been processed, the saved information for the three types of lists is sorted appropriately, and the web pages for these lists are created.

Thus, the program is not unusual: parsing, processing, output. It is inherently a bit messy because of different combinations of situations that must be handled in different ways, for example, the different combinations of the title, author, *TUGboat* category, and PDF url that may be present, resulting in a different output formats.

The web site generation program skips over the TEX at the beginning of the file until it reaches the first of the few TEX commands it understands, for instance, `\issue{25}{2}{81}{2004}{}{}` which indicates the year, volume number and issue number within the year, and issue sequence number, starting at 1 for the first issue of *TUGboat*.

4.3 Program capabilities

Some of the capabilities of the program have already been mentioned, such as its conversion of the TEX with diacritical marks for a person's name into HTML with its equivalent diacritical marks for display on the webpages, and in turn into the strictly English A–Z and a–z alphabet for sorting. Unifying different versions of an author's name and of *TUGboat* categories has also been previously mentioned.

The program must also display slightly different table of contents pages for proceedings than for non-proceedings issues. Additionally, twice (to date) something other than a normal *TUGboat* issue was distributed by TUG such that we nevertheless want to handle it like a *TUGboat* issue: the preprints for the 2004 proceedings (which filled the role of *TUGboat* issue #79), and the EuroTEX 2005 proceedings (which filled the role of *TUGboat* issue #85). These instances require different formatting.

Sometimes a capsule file has text to be included in the online table of contents that doesn't appear in the printed edition (argument #9 to `\capsule`, see section 3.2). This is typically used for links to videos of conference presentations and supplementary information for an article. Our program handles these commands by simply passing along the HTML to the output, while the TEX macros for generating the printed cover ignore these commands.

We want the items in the online contents to be ordered according to their appearance in the issue. However, from issue 81 on, the items in the capsule file are in the correct order for generating the level-of-difficulty table of contents, so the program has to resort them by page number. The items in capsule files before issue 81 are generally in page number order, but even then, sometimes the start-

of-article page numbers are insufficient. Sometimes multiple items appear on the same page, some page numbers have a non-numeric format, e.g., c3 for cover page 3 (the inside back cover), and there are other occasional special cases. Thus, an optional `\offset{...}` parameter may follow a page number in the page-number argument to a `\capsule` command, and these values are used to adjust the order of things in the page-number sort. `\offset` is ignored by TEX.

In the example at the end of section 3.2, the page number is not given directly: it's specified as `\getfirstpage{heff}`. Seeing that, the program looks for an auxiliary file `heff/firstpage.tex`, as our production directory is arranged. TEX itself creates that file automatically when an issue is run, thus avoiding manually entering page numbers.

The example has one more small feature: the url is given as `/TUGboat/!TBIDENT!heff.pdf` instead of `/TUGboat/tb32-1/tb100heff.pdf`. The curious directive `!TBIDENT!` came about after we had been creating capsule files for new issues for some time. We realized that (naturally enough) new capsule entries were usually created by copying old ones, and it was easy to neglect updating the issue number (`32-1`), the sequence number (`100`), or both. `!TBIDENT!` simply expands to that sequence; the actual values needed are given in the `\issue` command mentioned earlier.

Finally, we added some consistency checking to the program to help discover typos, etc.:

- the PDF files referred to in the urls must actually exist (this assumes the program is being run on `tug.org` itself with full access to the archive of PDFs of individual articles);
- everything in the capsule file must end up in the output, i.e., no two items have identical page numbers (we manually differentiate with `\offset` when that would otherwise be the case).

5 Opportunities

Having the archives of *TUGboat* available online made it possible for people to access articles in old issues without having to find a physical copy of the issue. It also brought home the question of the extent to which non-members of TUG would have access. The attendant full lists of authors, titles, and keywords are particularly useful to researchers (ourselves included) trying to find things in the *TUGboat* archive. For example, we have used those lists constantly in doing background research for the TUG interview series (`http://tug.org/interviews`), and in creating the book *TEX's* 2^5 *Anniversary*, which involved creating lists of papers from *TUGboat*. (A fuller description of how that book came about is at `http://tug.org/store/tug10/10paper.pdf`.)

From the outset of *TUGboat* online, issues more than a year old were made publicly available. A year after the publication of each new issue, it was posted to the *TUGboat* web site. Following this precedent, we immediately put all older issues online in full as the material became available.

Eventually, the TUG board of directors decided to put each issue online for members to access as soon as the paper copy was published, but to wait a year before making it available to non-members. At about the same time, members were offered a membership option (with a discount from the regular cost of membership) where they were not sent paper copies of *TUGboat* and only accessed it online. About 15 percent of members choose this option.

In general it seems that having *TUGboat* online is good for both TUG members and other people interested in TEX, typography, and fonts. Having *TUGboat* online is consistent with TUG's mission:

> TUG is a not-for-profit organization by, for, and of its members, representing the interests of TeX users worldwide and providing an organization for people who are interested in typography and font design.

However, having *TUGboat* available online has the downside concern of being one less (substantial) reason for joining TUG — to subscribe to the journal.

More generally, having the machine readable capsule files for all issues of *TUGboat* allows the possible reuse of *TUGboat* data in other, perhaps as yet unforeseen, contexts.

Acknowledgments

Barbara Beeton, Mimi Burbank, and Christina Thiele all searched their memories and archives to help us. As described in the beginning of the article, Mimi initiated the process of getting *TUGboat* issues online (among many other *TUGboat* and TUG and TEX efforts). We'd like to dedicate this article to her memory; sadly, she passed away late last year (a full memorial is printed elsewhere in this issue).

◇ Karl Berry and David Walden
`http://tug.org/TUGboat/Contents`