CS–TN–56

The `cstn` Document Class for LaTeX 2_ε

Martin Reddy

1 Introduction

This document describes (and was written using) the local `cstn` (v2.5) LaTeX 2_ε class. This document class provides a general style for writing technical notes in the department of Computer Science at the University of Edinburgh. The default behaviour of this class is to produce documents which are typeset as two-sided single-spaced 11 point text on A5 sized pages, ready for printing in "book" format. To use this class, all you need do is begin your LaTeX document with a declaration in the form:

`\documentclass[`⟨*options*⟩`]{cstn}`

This document will begin by introducing the various options which the `cstn` class currently supports, followed by the various styles, macros and environments which it exports. A short appendix is provided which describes the current procedures for submitting and publishing technical notes in the department. Finally, a simple example of using the `cstn` class is provided.

2 Class Options

The `cstn` class is built on top of the standard LaTeX 2_ε `article` class and will therefore accept most of the options which can be passed to `article`. However the following class options are explicitly provided by `cstn`:

a5/a4 : specifies the paper size to produce the document on. The default option, `a5`, will produce documents with margins calculated for A5 paper. This is used for printing the technical note with two pages per sheet of paper in a "book" format (see Appendix A for details on printing technical notes). Providing the `a4` option will produce standard A4 sized documents.

`10pt/11pt/12pt` : Specifies the point size for the main body text that you wish to use. The default option is `11pt` for A5 documents, and `12pt` for A4 documents.

3 Title Section

Title sections can be generated in the usual fashion using the `\maketitle` command. However, the `cstn` class extends the functionality (and display) of title sections somewhat. It allows you to specify much of the document's history; including revision number, comments, initial release dates, original authors and current maintainers etc. The following information can be specified before calling `\maketitle`:

- `\title` : the title of the technical note. This will be right justified at the top of the page. Remember that you can use the `\\` command to force a line break within the title.
- `\author` : the original author(s) of the technical note. If there is more than one author, then each name should be separated with a line break command, `\\`.
- `\tnnumber` : the number of the technical note. This will normally be a two digit number (i.e. you should not specify the full technical note designation such as 'CS–TN–xx', only the 'xx' part).
- `\revision[`⟨*info*⟩`]{`⟨*number*⟩`}{`⟨*date*⟩`}` : specifies the current revision number and date of the document. If not specified, the revision will default to 1.0 and the date will default to the current date. If you specify an empty date string, then the date will default to the current date. You can also supply an optional information field to describe the major additions which this revision introduced.
- `\oldrevision[`⟨*info*⟩`]{`⟨*number*⟩`}{`⟨*date*⟩`}` specifies information for a previous document revision (other than the current or the

initial release). The usage is the same as `\revision` above. You may provide any number of `\oldrevision` definitions (whereas `\revision` may only occur once). Previous revisions should be specified in reverse cronological order—most recent first.

N.B. `cstn` will produce a `.tnt` (Tech Note Temporary) file to store all previous revision information. This file is not required once the document has been produced and may be safely deleted. (N.B. The `texpurge` utility will remove this file for you, along with all other LaTeX temporary files.)

- `\initialdate{...}` : the date of the first release of the document. If you specify a revision number greater than 1.0, then you should always define this field too.
- `\maintainer{...}` : the name of the current maintainer of the document. This is used if another person has taken over the development of a technical note since the original author(s) wrote it.

The title section which the `cstn` class produces will contain the title, the full technical note designation, the current revision number and date, and the list of authors. If a maintainer is specified then this name will also appear on the title section, directly below the original authors. For example:

```
\title{Installing \LaTeX2e\ on a Babylon station}
\author{Jeffrey Sinclair\\Susan Ivanova}
\maintainer{John Sheridan}
\tnnumber{88}
\revision[Shadow detection]{2.1}{20 June 2260}
\oldrevision[Minbari language support]{2.0}{1 June 2260}
\oldrevision{1.1}{13 December 2258}
\initialdate{25 September, 2258}
\maketitle
```

4 Logical Styles

A number of logical styles are provided in order to typeset certain common items such as filenames, environment variables, menu items etc. The use of these logical styles is encouraged in order to maintain a consistent formatting style across all technical notes.

Command	Description	Example
`\filename{...}`	a file or path name	`/usr/local/lib/TeX`
`\programname{...}`	a command or program name	`ls -lR`
`\arg{...}`	indicates a required parameter	⟨*filename*⟩
`\oarg{...}`	indicates an optional parameter	[⟨*-help*⟩]
`\envvar{...}`	display a shell environment variable	`$HOME`
`\menu{...}`	an item from a menu strip	**File/Exit**
`\user{...}`	a user name	`nobody`
`\group{...}`	a group name	`sys`
`\ypmap{...}`	a YP map	`ypmap`
`\hostname{...}`	a host name	`papa`
`\domain{...}`	a domain name	`dcs.ed.ac.uk`
`\url{...}`	a URL (Universal Resource Locator)	`http://www....`
`\newsgroup{...}`	a newsgroup name	`eduni.dcs.sys`
`\email{...}`	typesets an e-mail address	<user@dcs.ed.ac.uk>
`\target{...}`	a Makefile target	`clean`
`\action{...}`	a Makefile action	`$(RM) *.o`
`\object{...}`	an LCFG object	`object`
`\method{...}`	an LCFG method	method
`\resource{...}`	an LCFG resource	`resource`
`\value{...}`	an LCFG value	value
`\define{...}`	to define any new terms that you use	*warp core breach*
`\data{...}`	arbitrary data	`foo`

5 Commands

The `cstn` class also provides a number of macros to assist you in writing your technical notes, and to provide standard typesetting formats for various common components. You are encouraged to use these where appropriate.

- `|...|` : This is a short-cut for the `\verb|...|` command which allows you to typeset verbatim text in the typewriter font. This is particularly useful if the text includes LaTeX reserved symbols such as `\`, `{`, `}` etc. For example:

  `|\usepackage{blah}|`

  (N.B. Lamport states that a `\verb` command should not occur in the argument of another command).

  The normal pipe symbol behaviour can be obtained by enclosing the desired section in a `normalpipe` environment (see next section for more details).

- <...> : any text surrounded by angled brackets will be typeset in italics. This is useful to introduce parameters or filenames which include sections that are not to be interpreted literally, but to be filled in with an appropriate value, e.g. |/home/<user>/.benv| produces /home/⟨*user*⟩/.benv. You should use double angled brackets (i.e. << or >>) in order to gain the normal angle bracket behaviour.

- \typedinput[...]{...} , \response{...} , \setprompt{...} and \cmd{...} {...} : often you may wish to display a number of commands which the user should type in to the shell, and the subsequent system responses. You can use the above commands to display these with a shell prompt before each line. The optional (square bracketed) parameter can be used to define the prompt. By default the prompt will be "➜", however you can redefine the default prompt using \setprompt. The \cmd command is just a wrapper for a \typedinput followed by a \response. For example:

  ```
  \cmd{uname -sr}{SunOS 5.5.1}
  ```

  produces,

  ```
  ➜ uname -sr
  SunOS 5.5.1
  ```

- \twiddle : produces the tilde (~) symbol. This may be used for specifing a user's home directory, e.g.

  ```
  ...personal \LaTeXe\ styles live in \filename{\twiddle/tex/latex}.
  ```

- \smallverb and \normalverb : the standard A5 page size used by cstn may prove restricting if trying to use a verbatim environment to paste a section of code into the document. Therefore the above commands can be used to make the verbatim environment use a smaller font and also to return to the normal verbatim font size.

- \key{...} and \keycomb{...} {...} : These two macros can be used to display keyboard combinations which a used should press for certain actions; either a single key (e.g. \key{X}) or a combination (e.g. \key{Ctrl}{A}).

  A number of special keys are already predefined. Some of these can take an optional parameter in square brackets to specify a key combination:

Command	**Description**	**Example(s)**
`\shiftkey[...]`	The Shift key	Shift or Shift + Q
`\altkey[...]`	The Alt key	Alt or Alt + Q
`\ctrlkey[...]`	The Ctrl key	Ctrl or Ctrl + Q
`\metakey[...]`	The Emacs meta key	M-x or M-x + Q
`\tabkey`	The Tab key	Shift or Shift + Q
`\esckey`	The Escape key	Esc
`\enterkey`	The Enter/Return key	Enter
`\deletekey`	The Delete key	Delete
`\backspacekey`	The Backspace key	$\longleftarrow$

6 Environments

The following environments are provided by the `cstn` class:

- **decl** : This environment allows you to typeset declarations (or any other text you wish to highlight). It will render the text in a box, offset from the left margin. For example:

```
\begin{decl}
  Put your declaration here
\end{decl}
```

- **display** : This is an environment for displaying sections of code, or sequences of commands, which are indented from the left margin. For example:

```
\begin{display}
  \cmd{ping calvay}{calvay.dcs.ed.ac.uk is alive}\\
  \cmd{rup calvay}{calvay up 13 days, 21:40, ...}
\end{display}
```

- **normalpipe** : This environment allows you to override the default operation of the pipe symbol in `cstn` (which is used as a shortcut to enter verbatim text). For example,

```
\begin{normalpipe}
  \begin{tabular}{l|l}   % this won't work without
    hi & there           % without "normalpipe"
  \end{tabular}
\end{normalpipe}
```

7 Document Revision History

In addition to the various macros described in Section 5, `cstn` also provides a command called `\revisionhistory`. This will automatically produce a revision history section for your document based upon the information which you have provided for the title section (see Section 3). You should place this command at the end of your document.

The revision history is built up from the information which you provided in the following definitions before the `\maketitle` command: `\tnnumber`, `\revision`, `\oldrevision`, `\initialdate` and `\maintainer`. Below is an example revision history which has been automatically generated for this document.

Revision History

This document is published as technical note number 56 by the Department of Computer Science at the University of Edinburgh. The document's revision history is as follows:

- Revision 1.1 (11 November 1996). Rewritten to use cstn class
- Initial release (6 June 1996).

This document is currently maintained by Martin Reddy.

A Procedures for Printing & Publishing Technical Notes

This appendix provides information on how to print A5 formatted technical notes. It also provides some information on the procedures for publishing a technical note here in the Computer Science department.

A.1 Printing Technical Notes

The default formatting options for the cstn class will produce A5 sized documents, ready for printing as A5 booklets. In order to print the document as such you must use the -Zisa5 filter to lpr.

For example, in order to print an A5 technical note as an A5 booklet, you should use the following command (to a printer which supports duplex printing, such as hp16 or hp24):

```
lpr -Zisa5,book,duplex foo.dvi
```

Note that this provides far superior results than simply producing A4 documents and then printing them with -Zbook,duplex because there is no font scaling involved in the A5 booklet printing. The results should therefore be more visually appealing.

A.2 Publishing Technical Notes

The normal approach is simply to mail support@dcs once you have completed the technical note. They will then assign a number to your report and provide instructions on how to install the document on-line.

The current policy is to make technical notes available in Adobe PDF format. You should therefore endeavour to make your document available in this format. The following is a description on how to produce PDF output from your LaTeX documents:

1. Produce the DVI document as usual, e.g. latex foo.tex.
2. Produce the PostScript with dvips. Using the outline versions of the Computer Modern font should produce better results, e.g. dvips -a -Pcmoutline foo > foo.ps.
3. Produce the PDF document using Distiller 2.0 or later. Currently, this can be done on the Macs by mounting the “PDF Conversion Folder” from arkle, dropping the PostScript file in the “In” folder, and waiting for the equivalent PDF file to appear in the “Out” folder.

All departmental technical notes can be browsed on-line via the department’s World Wide Web server at:

```
http://www.dcs.ed.ac.uk/docs/technote/
```

B Example LaTeX 2_ε Code

The source code to this document can be found on-line as:

```
/usr/local/share/TeX/tex/latex/examples/cstn.tex
```