Next Previous Contents

The Literate Programming FAQ

David B. Thompson
<thompson@shelob.ce.ttu.edu>

Version: 1.3.1, Mar 15, 2000

The purpose of this document is two-fold: First, there is a need to present a basic description of literate programming and how application of literate programming principles can improve the resulting code. Second, there is a need to present a list of tools available to iterate programmers. Hopefully, this document will meet both needs.

6. What is Literate Programming?

7. How do I begin literate programming?

8. Important and Actively-Supported Tools

9. Unsupported Tools

10. Are there other tools I should know about?

11. What other resources are available?

12. Are there any code examples?

13. Bibliographies

14. Other Opinions about Literate Programming

15. How to anonymously ftp

16. Acknowledgements

17. End notes

Next Previous Contents

Next Previous Contents

1. Welcome

Information contained in this document is the best available at preparation. The original file was dated October 15, 1993 (just for historical purposes).

1.1 Disclaimer

This FAQ is presented with no warranties or guarantees of ANY KIND including correctness or fitness for any particular purpose. The author of this document has attempted to verify correctness of the data contained herein; however, slip-ups can and do happen. If you use this data, you do so at your own risk.

1.2 Copyright

1.3 What's New?

- Updated dpp entry. See Section dpp
- Added noweb-outline.el entry. See section noweb-outline.el

1.4 What's Needed?

- I've checked some of the links to software. If anyone finds the FAQ useful, please let me know if the links are active or dead when you're surfing.
- Some authors have disappeared. If you know one of them, or are an author (and wish to remain in contact ;), then please provide current contact information.
- I could use some feedback on the state of the FAQ. It's about as complete as I know how to make it.

Next Previous Contents

Next Previous Contents

2. Introduction

Literate programming is a phrase coined by Donald Knuth to describe the approach of developing computer programs from the perspective of a report or prose. The focus, then, is on description (and documentation) of the approach in human-readable form. This is in contrast to the normal approach of focusing on the code.

This document is for new and experienced users of literate programming tools. The purpose is to explain the concept of literate programming and to provide a resource for locating resources of interest to literate programmers and those interested in literate programming.

The Literate Programming (LitProg) Frequently Asked Questions (FAQ) list is maintained by Dave Thompson <thompson@shelob.ce.ttu.edu>.

Comments and constructive criticisms are welcome. Direct flames to `/dev/null` (or `nul` if you're a msdos user! `;-)` If you find an error, please report it. I'm particularly interested in establishing the locations of generally available literate programming tools. If you are the author of such a tool and wish to have it included in this list, please send email.

Please note this is a work-in-progress. It is *not* complete, and probably will never be complete. Nevertheless, the information contained herein may be useful to some. Use it as it is intended.

Next Previous Contents

Next Previous Contents

3. How do I get the FAQ?

3.1 Literate Programming FAQ

You have many ways to get a current copy of this FAQ. One is to use anonymous ftp (if you don't know how, see a later section in this FAQ) to connect to one of the Comprehensive TeX Arvchive Network sites or the Literate Programming Archive and retrieve a copy of the file. Open an ftp connection to one of the CTAN sites and retrieve the file `help/comp.programming.literate_FAQ`.

Cesar Bellardini cballard@santafe.com.ar prepared a translation of the FAQ into Spanish. It's available at

(For more information on CTAN and the literate programming archive, see the section below entitled ``*Internet Nodes of Interest to Literate Programmers*''.)

Next Previous Contents

Next Previous Contents

4. Is there a newsgroup?

One of the most important resources is the literate programming newsgroup, comp.programming.literate. Because of the amount of spamming and unrelated posts, the newsgroup is now moderated. You can read this newsgroup using your standard reader.

Next Previous Contents

5. What internet nodes are of interest to literate programmers?

The principal nodes of interest to literate programmers are the Literate Programming Archive (LPA hereafter) and the CTAN (Comprehensive TeX Archive Network).

5.1 Web Ring

There is a web ring for literate programming. It is at the URL
www.webring.org/cgi-bin/webring?ring=litprog;list

5.2 The Literate Programming Archive (LPA)

The Literate Programming Archive (LPA) is:

- **Node:** ftp.th-darmstadt.de [130.83.55.75]
- **Directory:** /pub/programming/literate-programming
- **Notes:** Fastest response during off-U.S. [yep] business hours.

However, the LPA seems to be defunct in that no files are available in the /pub directory. If anyone knows anything about the status of the LPA, please send email.

5.3 Comprehensive TeX Archive Network (CTAN)

Participating hosts in the Comprehensive TeX Archive Network are (from the file `CTAN.sites`):

- **ftp.dante.de (Mainz, Germany)**
  - anonymous ftp `/tex-archive (/pub/tex /pub/archive)`
  - Gopher: `gopher.dante.de`
  - e-mail ftpmail@dante.de
  - WWW www.tex.ac.uk
  - Administrator: <ftpmaint@dante.de>
- **ftp.tex.ac.uk (Cambridge, UK)**
  - anonymous ftp `/tex-archive (/pub/tex /pub/archive)`
  - Gopher: `gopher.tex.ac.uk`
  - NFS mountable from `nfs.tex.ac.uk:/public/ctan/tex-archive`
  - WWW www.tex.ac.uk

    - Administrator: <ctan-uk@tex.ac.uk>

- **ctan.tug.org (Boston, Massachusetts, USA)**
    - anonymous ftp `/tex-archive (/pub/archive)`
    - WWW ctan.tug.org
    - Administrator: <ctan@tug.org>

The pointer, ftp://ftp.cdrom.com/pub/tex/ctan/CTAN.sites, is directed to the official list of CTAN archive sites and their mirrors.

Next Previous Contents

Next Previous Contents

6. What is Literate Programming?

Literate programming is the combination of documentation and source together in a fashion suited for reading by human beings. In fact, literate programs should be enjoyable reading, even inviting! (Sorry Bob, I couldn't resist!) In general, literate programs combine source and documentation in a single file. Literate programming tools then parse the file to produce either readable documentation or compilable source. The WEB style of literate programming was created by D.E. Knuth during the development of his TeX typsetting software.

All the original work revolves around a particular literate programming tool called WEB. Knuth says:

> The philosophy behind WEB is that an experienced system programmer, who wants to provide the best possible documentation of his or her software products, needs two things simultaneously: a language like TeX for formatting, and a language like C for programming. Neither type of language can provide the best documentation by itself; but when both are appropriately combined, we obtain a system that is much more useful than either language separately.
>
> The structure of a software program may be thought of as a web that is made up of many interconnected pieces. To document such a program we want to explain each individual part of the web and how it relates to its neighbours. The typographic tools provided by TeX give us an opportunity to explain the local structure of each part by making that structure visible, and the programming tools provided by languages such as C or Fortran make it possible for us to specify the algorithms formally and unambigously. By combining the two, we can develop a style of programming that maximizes our ability to perceive the structure of a complex piece of software, and at the same time the documented programs can be mechanically translated into a working software system that matches the documentation.

See Section Other Opinions for some additional thoughts on literate programming.

Next Previous Contents

Next Previous Contents

7. How do I begin literate programming?

I've given considerable thought as to what should be in this section of the FAQ. This is probably the most important section of this document. My suggestion is that you review Section Supported Tools and Section Unsupported Tools to choose a system appropriate for the kind of development you do. Then, use the manual that accompanies the system to determine how it complements your development style.

Both Eric van Ammers, Section van Ammers, and Norman Ramsey, Section Ramsey, wrote some thoughts on literate programming. I've included these thoughts in Section Other Opinions below.

I started with a pretty-printing tool, Section cnoweb, as a test of the utility of interweaving significant documentation with code. My experience is detailed in Section Thompson.

Wayne Sewell's (1989) *Weaving a Program: Literate Programming in WEB*. Van Nostrand Reinhold, ISBN 0-442-31946-0 (pbk). This book focuses on using Knuth's web system.

I've read D. E. Knuth's collection of articles (1992) entitled *Literate Programming*. Center for the Study of Language and Information, Stanford University, ISBN 0-937073-80-6 (pbk). This book gives insight into Knuth's thoughts as he developed the web system of literate programming (and TeX for typesetting). However, it does not document methods for literate programming.

Some talk exists in the newsgroup/mailing list for a Usenet University course in literate programming. I'm sure discussion of this topic will be welcomed. (1Feb2000: Note this thread has been dead for a long, long time. I wish someone would pick it up.)

Next Previous Contents

Next Previous Contents

8. Important and Actively-Supported Tools

I have selected a few of the tools from my list that appear to be most actively supported. Inclusion here does not imply endorsement; exclusion does not imply lack of quality.

8.1 CWEB

Developer:

Silvio Levy and D.E. Knuth

Version:

3.5

Hardware:

Unix systems (dos and amiga ports available)

Languages:

C and C++

Formatter:

Plain TeX and LaTeX.

Availability:

Anonymous ftp from:

- ftp://labrea.stanford.edu:/pub/cweb
- LPA:/c.c++
- CTAN:/web/c_cpp/cweb
- DOS version is no longer available.
- Win32 version www.literateprogramming.com
- Amiga version from Aminet:dev/c.
- Mac port of CTANGLE in LPA:/machines/mac
- LaTeX support in LPA:/c.c++

Readme:

Bundled with above

Description:

No description provided.

Support:

Bugs to <levy@math.berkeley.edu>

8.2 CWEBx3.0

Developer:

Marc van Leeuwen

Version:

3.04

Hardware:

Any system using ASCII code

Languages:

ANSI C

Formatter:

Plain TeX

Availability:

Anonymous ftp from:

- wwwmathlabo.univ-poitiers.fr/~maavl/CWEBx/

Readme:

Bundled with above

Brief description:

A modified implementation of CWEB, with some extensions. Provides a mode for full compatibility with Levy/Knuth CWEB. The most significant extras are:

- Typedef declarations affect formatting througout source file
- Include files are scanned for typedef definitions
- Flexible selection of layout style
- Possibility to refer to sections using symbolic labels
- CTANGLE detects unbalanced braces and parentheses
- CWEAVE can be made to report syntax errors more easily
- Some additional mechanisms to avoid formatting problems
- New and modular set of grammar rules, based on ANSI C syntax
- Possibility to suppress #line directives
- A new manual

Support:

bugs and remarks to maavl@mathlabo.univ-poitiers.fr

8.3 FWEB

Developer:

John A. Krommes

Version:

1.62

Hardware:

Unix, VMS, and DOS platforms (anything with ANSI C)

Languages:

C, C++, Fortran-77, Fortran-90, Ratfor, TeX; also, a anguage-independent mode.

Formatter:

LaTeX. Plain TeX may work, but is no longer supported.

Availability:

Anonymous ftp from:

- ftp.pppl.gov/pub/fweb
- CTAN:/web/fweb
- msdos version on ftp.ppl.gov site

Readme:

In bundle with above.

Description:

It also has a well-developed user's manual and its own FAQ (see above). Beginning with 1.40, documentation is maintained in gnu texinfo format. It runs on most platforms: VMS, PC, UNIX, and pretty much anything that the GNU C compiler (GCC) is supported for.

Features:

- Processes multiple languages during a single run (so one can mix C and Fortran, for example).
- Language-independent mode (v1.40).
- Ability to turn off pretty-printing (v1.40).
- Built-in Ratfor translator.
- Built-in macro preprocessor (closely follows ANSI C, with extensions).
- A style file that allows the user to adjust many parameters and behavior patterns of FWEB.
- Various operator-overloading features that provide additional pretty-printing capabilities to languages such as C++ and Fortran-90.
- Numerous miscellaneous features and command-line options.

Support:

Bug reports and suggestions to krommes@princeton.edu Online documentation is available at w3.pppl.gov/%7ekrommes/fweb_toc.html

8.4 noweb

Developer:

Norman Ramsey <nr@eecs.harvard.edu>

Version:

2.9a

Hardware:

Unix and DOS platforms (DOS binaries available for v2.7).

Languages:

All programming languages, singly or in combination. Automatic indexing for C, Icon, Pascal, Standard ML, TeX, Yacc

Formatter:

Plain TeX, LaTeX, and HTML formatters. Will convert LaTeX to HTML automatically.

Availability:

Anonymous ftp from:

- CTAN:/web/noweb
- LPA:/independent
- Last recourse, use ftp.cs.virginia.edu:pub/nr

Readme:

With bundle above, or see the noweb home page: www.eecs.harvard.edu/~nr/noweb Those without http access can consult ``Literate Programming Simplified," IEEE Software, September 1994, pp97-105, or ``Literate Programming Using Noweb," Linux Journal, October 1997, pp64-69.

Description:

Noweb is designed to meet the needs of literate programmers while retaining the simplest possible input format. Its primary advantages are simplicity, extensibility, and language-independence. Noweb uses 5 control sequences to WEB's 27. The noweb manual is only 3 pages; an additional page explains how to customize its LaTeX output. Noweb works ``out of the box" with any programming language, and supports TeX, latex, and HTML back ends. A back end to support full hypertext or indexing takes about 250 lines; a simpler one can be written in 40 lines of awk. Unlike WEB, Noweb does not have prettyprinting built in, but there are several third-party extensions that provide prettyprinting, includeing dpp, pretzel, and nwpp.

Noweb supports indexing and identifier cross-reference, including hypertext ``hot links.'' noweb includes a simple, efficient LaTeX-to-HTML converter, so you can use hypertext browsers on your legacy documents. Noweb can also process nuweb programs, so you can use noweb to convert a standard nuweb program to HTML with one command.

Support:

email to the author

8.5 nuweb

Developer:

Preston Briggs: <preston@cs.rice.edu>

Version:

0.87

Hardware:

Unix systems: Sparcs, RS/6000s, HPs; (!) MSDOS and Amiga.

Languages:

Any programming language or combination of programming languages.

Formatter:

Latex

Availability:

Anonymous ftp from:

- Unix: CTAN:/web/nuweb
- DOS: CTAN:/web/nuweb-pc
- LPA:/independent
- Amiga: CTAN:/web/nuweb/nuweb_ami
- Amiga: wuarchive.wustl.edu/pub/aminet

Readme:

Send mail to <preston@cs.rice.edu>

Description:

A single program that takes a web file written in a combination of latex and any programming language(s) and produces a latex file that can be pretty printed and a set of files containing code for compilation/interpretation by the appropriate language processors.

Strengths include speed, simplicity, multiple languages, nice indices and cross-references, latex. Doesn't require any special macros or macro files.

Drawbacks: latex-dependent, no code pretty printing, harder to make indices than cweb.

More good stuff: nice support for make, doesn't reformat source files, so they're easy to debug. Lots of control without too much effort. That is, it doesn't do too much!

Future directions... Very little change planned, except perhaps refinements in the indexing software.

Support:

Hack it yourself or send e-mail to <preston@cs.rice.edu>

8.6 ProTeX

Developer:

Eitan Gurari <gurari@cis.ohio-state.edu>

Version:

ProTeX 1.5, AlProTeX 2.3

Hardware:

Any platform with (La)TeX

Languages:

Any language

Formatter:

TeX or LaTeX

Availability:

Anonymous ftp from:

- www.cis.ohio-state.edu/~gurari/systems.html
- LPA:/independent

Readme:

With bundle above

Description:

- Easy to use
- Extensible
- Language independent
- Multiple output files
- Fast (single compilation provides output and dvi files)
- Option for XHTML and pdf files
- No installation is needed besides copying the files (written in TeX) Introduction of main features and examples on web site above. Complete manual in Eitan M. Gurari, "TeX and LaTeX: Drawing and Literate Programming", McGraw-Hill, 1994

Support:

<gurari@cis.ohio-state.edu>

Next Previous Contents

Next Previous Contents

9. Unsupported Tools

9.1 AFTWEB (Almost Free Text WEB)

Developer:

Todd A. Coram

Version:

4.6

Hardware:

Linux, Unix, MSDOS Any system with Perl, and a C++ compiler with STL (such as gcc 2.7.2).

Languages:

Any (C/C++ support supplied)

Formatter:

LaTeX or HTML by way of AFT.

Availability:

www.mindspring.com/~coram/aft.html

Readme:

Bundled with above.

Brief description:

AFTWEB uses a CWEB-like syntax. It uses AFT for documentation markup (AFT is a minimalistic, yet powerful, markup language with very few commands). AFTWEB was written in AFTWEB (using C++) and the weaved document is available online (as HTML) at the URL listed above.

Support for C and C++ is supplied. You can easily support other languages (such as Java and Perl) by writing a new language description file.

The markup language AFT is very easy to learn and is available at the same URL as AFTWEB.

Support:

Bugs to tcoram@pobox.com

9.2 APLWEB

Developer:

Christoph von Basum

Version:

Unknown

Hardware:

MSDOS

Languages:

IBM APL2 and STSC APL

Formatter:

Plain TeX

Availability:

Anonymous ftp from: watserv1.uwaterloo.ca:/languages/apl/aplweb

Readme:

At above ftp location.

Description:

None available.

Support:

Unknown

Note:

The status of this particular package is unknown. It's at the ftp site, but other than that I can't say. Last known email address of developer is CvB@erasmus.hrz.uni-bielefeld.de.

9.3 CLiP

Developer:

E.W. van Ammers and M.R. Kramer

Versions:

2.0 and 2.4b (DOS only)

Platform:

Vax/VMS, Unix, DOS

Languages:

Any programming language

Formatter:

Any formatter (TeX, LaTeX, Troff, Runoff, HTML, etc) or any wordprocessor including WYSIWYG systems (Word Perfect, WinWord, Ami Pro, Word Pro, etc.)

Availability:

Anonymous ftp from:

- ftp://sun01.info.wau.nl:/CLIP/ms_dos
- ftp://sun01.info.wau.nl:/CLIP/ms_dos_24b
- ftp://sun01.info.wau.nl:/CLIP/vax_vms
- ftp://sun01.info.wau.nl:/CLIP/unix
- CTAN:/web/clip
- LPA:/machines/ms-dos
- LPA:/machines/vax

Readme:

With bundle above

Description:

CLiP does not use explicit commands to perform the extraction process. Rather it recognizes pseudostatements written as comments in the programming language in question. CLiP distinguishes pseudostatements from ordinary comments because the former comply with a particular style. This style can be adjusted to suit virtually any programming language. The CLiP approach to LP makes the system extremely versatile. It is independent of programming language and text processing environment. We designed CLiP to be compatible with hypertext systems as well. Some hypertext examples are at:

- ftp://sun01.info.wau.nl/clip/html/queens.htm
- ftp://sun01.info.wau.nl/clip/html/pal1.htm

Features:

- CLiP imposes virtually no limitations on the text-processing system used to produce the documentation. If the text-processor supports these items you can
- structure the documentation according to your own taste.
- include drawings, pictures, tables etc.
- disclose your documentation my means of X-ref tables, Indexes, Table of contents, Table of tables, Table of figures, etc.
- typeset the documented code.
- Extracts any number of modules from a maximum of 64 source files.
- No pretty-printing. Code from the source files is copied "as is" to the module.
- Appearance of code segments in the documentation matches those of the modules to ease

the identification of code segments.

- Supports partially specified data types.
- Comprehensive user manual (preliminary version) and technical description.
- No automatic generation of a X-ref table for program identifiers.

Support:

Bugs, problems and assistance by e-mail to <Eric.vanAmmers@user.info.wau.nl>

9.4 mCWEB

Developer:

Markus Oellinger

Version:

1.0

Hardware:

Unix

Languages:

C/C++

Formatter:

plain TeX

Availability:

anonymous ftp from ist.tu-graz.ac.at:/pub/utils/litprog/mcweb/mcweb.tgz

Readme:

at same location

Description:

This is mCWEB 1.0, a descendant of the CWEB system of structured documentation by Donald E. Knuth and Silvio Levy. It adds some features that are indispensable when working in a team. mCWEB regards a project of a book consisting of several chapter files. By means of import and export commands, it automatically manages all relationships between the chapters of a book and to other books.

Interface documentation is now also part of mCWEB. It is extracted into a second TeX file. This makes it possible to define well known interfaces between the individual parts of a project that will be implemented by different persons.

In addition, mCWEB parses C header files to find out about all the datatypes defined there.

mCWEB comes with a full completely rewritten user manual and is compatible with CWEB.

Support:

Institute of Software Technology, moell@ist.tu-graz.ac.at

9.5 FunnelWeb

Developer:

Ross N. Williams ross@ross.net

Version:

V3.2 (May 1999).

Hardware:

MS-DOS, MacOS, Win32, OpenVMS, Solaris, Red Hat Linux, BSD/OS, FreeBSD, Digital Unix, IRIX.

Status:

Open Source GNU.

Languages:

No restrictions.

Formatter:

Generates TeX and/or HTML

Web:

www.ross.net/funnelweb/

Availability:

ftp.ross.net/clients/ross/funnelweb/

Readme:

With bundle above.

Description:

FunnelWeb is a production-quality literate-programming tool that emphasises simplicity and reliability. Everything about FunnelWeb, from the simplicity of its language to the comprehensive tutorial in the user's manual, has been designed to make this as simple, as practical, and as usable a tool as possible.

Features:

- Provides a simple macro preprocessor facility.
- Generates typeset documentation in TeX and/or HTML formats.
- Runs on a wide range of platforms.
- Portable C source code distributed under GNU licence.

- Comprehensively documented online:
  - www.ross.net/funnelweb/tutorial/
  - www.ross.net/funnelweb/reference/
  - www.ross.net/funnelweb/developer/
- Programming-language independent.
- Mature and essentially bug-free (released 1992).
- Can generate multiple output files.
- Allows complete control over the output text.
- Also useful for generating web sites!

Support:

No formal support available. Mailing list maintained with about 50 subscribers. Informal assistance available from mailing list.

9.6 FunnelWeb 3.0AC

Developer:

Enhanced by A.B.Coates coates@physics.uq.edu.au from FunnelWeb v3.0 by Ross N. Williams ross@guest.adelaide.edu.au

Version:

3.0AC

Hardware:

MSDOS, Mac, VMS, Sun, OSF/1, Linux, Sys.V, OS/2.

Languages:

No restrictions.

Formatter:

Tex, LaTeX, or HTML.

Availability:

Anonymous ftp from ftp.physics.uq.oz.au:/pub/funnelwebAC30.tar.gz

Readme:

With bundle above; for FunnelWeb manual see WWW page www.physics.uq.oz.au:8001/people/coates/funnelweb.html

Description:

FunnelWeb 3.0AC is an enhanced version of FunnelWeb (see the entry for FunnelWeb). FunnelWeb is designed to be typesetter independent, though FunnelWeb v3.0 only supports

(La)TeX as the typesetter. FunnelWeb 3.0AC also supports HTML, and creates appropriate hypertext links within the document among the code sections. FunnelWeb 3.0AC also supports automatic and manual insertion of line directives, so that compiler errors can be flagged back to the original FunnelWeb source file. FunnelWeb 3.0AC is completely compatible with FunnelWeb v3.0 sources (with one minor exception; see the file README.ABC which comes with the FunnelWeb 3.0AC distribution).

Support:

Supported by A.B.Coates coates@physics.uq.edu.au, subject to the time constraints imposed by his thesis.

9.7 LEO

Developer:

Edward K. Ream edream@mailbag.com

Version:

1.0

Hardware:

Windows

Languages:

Unknown

Formatter:

Unknown

Availability:

Contact the author or see www.mailbag.com/users/edream/front.html

Readme:

Unknown

Description:

See web site.

Support:

Unknown.

9.8 Literate Programmer's Workshop (LPW)

Developer:

Norbert Lindenberg

Version:

1.1

Hardware:

Apple Macintosh

Languages:

C++, Object Pascal & others

Formatter:

self-contained WYSIWYG system

Availability:

Anonymous ftp from:

- CTAN:/web/lpw
- ftp.apple.com:/pub/literate.prog

Readme:

With bundle above. Also comes with 38-page manual.

Description:

The Literate Programming Workshop is an environment for the integrated development of program source text and documentation in combined documents. It consists of a WYSIWYG word processor based on a style sheet approach, a mechanism to extract parts of the text in a document, and a project management system that handles multi-document projects. The system is designed to be used in conjunction with the Macintosh Programmer's Workshop: it prepares raw source text for the MPW compilers, accepts MPW error messages, and shows them in the context of the original documents. Automatic indexing and hypertext features allow for easy access to both source text and documentation.

LPW is shareware.

Support:

Bugs, problems, and questions to lpw@aol.com

9.9 MapleWEB

Developer:

Unknown

Version:

Unknown

Hardware:

Unknown

Languages:

Maple

Formatter:

Unknown

Availability:

Anonymous ftp from CTAN/maple/mapleweb

Readme:

Unknown

Description:

None

Support:

Unknown

9.10 Matlabweb

Developer:

Mark Potse

Version:

2.09

Hardware:

any, but only Unix tested & supported

Languages:

Matlab

Formatter:

Plain TeX and LaTeX.

Availability:

Anonymous ftp from the CTAN archives,

Readme:

Bundled with above

Description:

CWEB-like literate programming system for the Matlab language. Created with a modified version of the Spider system. Several more or less language-specific features:

- macros with multiple arguments
- comments and verbatim comments
- strings can be formatted as code, with help for nested strings, e.g. for callbacks in user interface programming.
- string arguments for macros, that get inserted in strings in the replacement text
- "@f foo TeX" works as in recent versions of CWEB

Support:

not guaranteed. Try M.Potse@amc.uva.nl, comments are welcome.

9.11 RWEB

Developer:

Unknown

Version:

Unknown

Hardware:

Unknown

Languages:

Unknown

Formatter:

Unknown

Availability:

Anonymous ftp from CTAN

Readme:

Unknown

Description:

Web generator in AWK.

Support:

Unknown

9.12 SchemeWEB

Developer:

John D. Ramsdell

Version:

2.1

Hardware:

Unix and DOS platforms

Languages:

Any dialect of Lisp.

Formatter:

LaTeX.

Availability:

The Unix version is in the Scheme Repository and it is available via anonymous ftp from:

- cs.indiana.edu:/pub/scheme-repository/utl/schemeweb.sh
- CTAN:/tex-archive/web/schemeweb
- The DOS version is part of the PCS/Geneva Scheme system which is available via anonymous ftp from: cui.unige.ch:/pub/pcs

Readme:

In bundle with above.

Description:

SchemeWEB is a Unix filter that allows you to generate both Lisp and LaTeX code from one source file. The generated LaTeX code formats Lisp programs in typewriter font obeying the spacing in the source file. Comments can include arbitrary LaTeX commands. SchemeWEB was originally developed for the Scheme dialect of Lisp, but it can easily be used with most other dialects.

Support:

Bug reports to ramsdell@mitre.org.

9.13 SpideryWEB

Developer:

Norman Ramsey <nr@eecs.harvard.edu>

Version:

Unknown

Hardware:

Unix and DOS platforms

Languages:

Most Algol-like languages, including C, Ada, Pascal, Awk, and many others.

Formatter:

Plain TeX and latex for text formatters.

Availability:

Anonymous ftp from CTAN.

Readme:

In distribution.

Description:

A system for building language-dependent WEBs. Spider is frozen; no further development is planned.

Support:

Bug reports to spider-bugs@oracorp.com.

9.14 WEB

Developer:

Donald Knuth

Version:

4.4 (apparently)

Hardware:

Any TeX system should have it.

Languages:

Pascal

Formatter:

TeX (of course! ;-)

Availability:

Distributed with TeX systems. Also avaliable in source form from labrea.stanford.edu/tex/web.

Readme:

Unknown

Documentation:

Available from labrea.stanford.edu/tex/web/webman.tex

Description:

This is the original software that started it all. The original TeX processor was written in WEB.

Support:

None known.

9.15 WinWordWEB

Developer:

Lee Wittenberg leew@pilot.njin.net

Version:

Unknown

Hardware:

Needs Microsoft Word for Windows, v.2.x, and, of course, MS-Windows 3.x.

Languages:

Any programming language.

Formatter:

Word for Windows 2.x for text formatting and file maintenance.

Availability:

samson.kean.edu:pub/leew

Readme:

WORDWEB.DOC in the downloadable package describes the system.

Description:

WinWordWEB is a set of a Word for Windows macros (plus a paragraph style) that provide a crude literate programming environment. The ``look and feel'' of the system is based on Norman Ramsey's noweb, but can easily be modified to suit individual tastes.

Support:

None. WinWordWEB was written as a prototype to see if a WYSIWYG literate programming system was possible. It is intended as a jumping off point for future work by others. However, the system is surprisingly usable as it stands, and the author is interested in hearing from users (satisfied and dissatisfied).

Anyone interested in actively supporting (and improving) the product should contact the author via email.

Next Previous Contents

Next Previous Contents

10. Are there other tools I should know about?

Follows is a list of some not-quite-literate-programming tools. Some term these pretty-printers. Others may call them literate programming tools. In any event, they don't seem to be quite in the same category as the tools listed above, so I'll include them here.

10.1 C2LaTeX

Developer:

John D. Ramsdell

Version:

Unknown

Hardware:

Unix

Languages:

C

Formatter:

LaTeX but it's easy to change the formatter.

Availability:

Anonymous ftp from omnigate.clarkson.edu:/pub/tex/tex-programs/c2latex.

Readme:

Absent. Documentation is in the C source for c2latex.

Description:

C2latex provides simple support for literate programming in C. Given a C source file in which the comments have been written in LaTeX, c2latex converts the C source file into a LaTeX source file. It can be used to produce typeset listings of C programs and/or documentation associated with the program.

C2latex produces LaTeX source by implementing a small number of rules. A C comment that starts at the beginning of a line is copied unmodified into the LaTeX source file. Otherwise, non-blank lines are surrounded by a pair of formatting commands (\begin{flushleft} and \end{flushleft}), and the lines are separated by *. Each non-blank line is formatted using LaTeX's \verb command, except comments within the line are formatted in an \mbox.

Support:

Send bug reports to ramsdell@mitre.org.

10.2 c2cweb

Developer:

Werner Lemberg

Version:

1.5

Hardware:

DOS, OS/2, Unix (gcc) - CWEB source included

Languages:

C, C++

Formatter:

TeX

Availability:

Anonymous ftp from CTAN:/web/c_cpp/c2cweb

Readme:

In distribution.

Description:

c2cweb will transform plain C or C++ code into a CWEB file to get a pretty formatted output. A modified CWEAVE (which transforms the CWEB file into a TeX file, see below) is included also.

Support:

Werner Lemberg <a7971428@unet.univie.ac.at>

10.3 c2man

author:

Graham Stoney <greyham@research.canon.oz.au>

language:

C, nroff, texinfo, latex, html

version:

2.0 patchlevel 33

parts:

documentation generator (C -> nroff -man, -> texinfo, ->latex, -> html)

location:

ftp from

- any comp.sources.misc archive, in volume42 (the version in the comp.sources.reviewed archive is obsolete)
- dnpap.et.tudelft.nl/pub/Unix/Util/ c2man-2.0.*.tar.gz
- Australia archie.au/usenet/comp.sources.misc/volume42 c2man-2.0/*
- N.America ftp://ftp.wustl.edu/usenet/comp.sources.misc/volume42/ c2man-2.0/*
- Europe: ftp://ftp.irisa.fr/News/comp.sources.misc/volume42/ c2man-2.0/*
- Japan: ftp://ftp.iij.ad.jp/pub/NetNews/comp.sources.misc/volume42/ c2man-2.0/*

Patches:

lth.se/pub/netnews/sources.bugs/volume93/sep/c2man*

description:

c2man is an automatic documentation tool that extracts comments from C source code to generate functional interface documentation in the same format as sections 2 & 3 of the Unix Programmer's Manual. It requires minimal effort from the programmer by looking for comments in the usual places near the objects they document, rather than imposing a rigid function-comment syntax or requiring that the programmer learn and use a typesetting language. Acceptable documentation can often be generated from existing code with no modifications.

conformance:

supports both K&R and ISO/ANSI C coding styles

features:

- generates output in nroff -man, TeXinfo, LaTeX or HTML format
- handles comments as part of the language grammar
- automagically documents enum parameter & return values
- handles C (/* */) and C++ (//) style comments
- doesn't handle C++ grammar (yet)

requires:

yacc/byacc/bison, lex/flex, and nroff/groff/texinfo/LaTeX.

ports:

Unix, OS/2, MSDOS, VMS.

portability:

very high for unix, via Configure

status:

actively developed; contributions by users are encouraged.

discussion:

via a mailing list: send "subscribe c2man <Your Name>" (in the message body) to listserv@research.canon.oz.au

help:

from the author and other users on the mailing list: c2man@research.canon.oz.au

announcements:

patches appear first in comp.sources.bugs, and then in comp.sources.misc.

updated:

1994/10/07

10.4 cnoweb

Developer:

Jim Fox

Version:

1.4 (January 4, 1991)

Hardware:

Anything with C and TeX.

Languages:

C

Formatter:

Plain TeX.

Availability:

Anonymous ftp from:

- CTAN
- LPA:/c.c++

Readme:

Unknown, cnoweb.tex contains documentation.

Description:

cnoweb is as it's name describes: write C, not web. No tangling or weaving is implemented. Documentation (between standard /* */ delimiteres) is written in TeX. cnoweb provides typesetting of documentation, an table of contents of routines, and pretty-printing of C source.

Support:

None known.

10.5 dpp

Developer:

Dan Schmidt <dfan@alum.mit.edu>

Version:

0.2.1

Hardware:

Any platform with Perl 5

Languages:

C/C++ (Java soon), under noweb

Formatter:

LaTeX

Availability:

www.dfan.org/real/dpp.nw

Readme:

www.dfan.org/real/dpp.html

Support:

email to the author <dfan@alum.mit.edu>

Description:

dpp is a C/C++ prettyprinter for noweb. Its output is extremely similar to that of CWEB, but it respects the indentation and line breaks of the source file.

Features include:

- user-defined keywords
- the ability to turn prettyprinting off for specified output files (e.g., makefiles)
- the option to typeset comments in TeX, or not
- prettyprinting of quoted code, in documentation or chunk names
- the ability to undo whitespace hand-formatting that looks good monospaced but awful in a proportional font

10.6 Fold2Web

Developer:

Bernhard Lang lang@tu-harburg.d400.de

Version:

V0.8

Hardware:

MSDOS

Languages:

All (must allow comment lines)

Formatter:

LaTeX

Availability:

Anonymous ftp from: kirk.ti1.tu-harburg.de (134.28.41.50) /pub/fold2web/readme /pub/fold2web/fold2web.zip

Readme:

In distribution

Description:

The idea behind the Fold2Web tool is the following: A programmer can write his program source with a folding editor and later map the folded source files automatically to WEB-files. The generated WEB-files can then be modified by inserting required documentations.

The advantage by starting program developement with original sources is to get short design cycles during the compile/debug steps. By using a folding editor the global structuring information can be already captured in folds during this developement phase. Fold information is typically stored in comment lines and thus will not affect the efficiency of the compile/debug design cycle.

Some folding editors and a folding mode for the emacs are available (e.g. see our FUE folding editor for MSDOS machines which is a modified micro emacs. Pick it at kirk in directory /pub/fold2web).

After reaching a stable version of a program source its time to convert the source file to a WEB-file and do the program documentation. Fold2Web is written to convert folded source text of any programming language to nuweb files. The folded structure is kept by mapping folds to scraps. Fold markers which differ between languages due to different ways of specifying comments can be configured for each language.

Good results can also achived when given but poor documented program sources have to be modified. Such sources can be folded using a folding editor to extract the global structures. This offers a global view to the program structures and help to understand its functionality. Furthermore

the program code is not affected, only comment lines are inserted. Once folded the program source can be automatically translated to a WEB document using the above tool.

Support:

email to lang@tu-harburg.d400.de

10.7 Funnelweb Mode

Developer:

Daniel Simmons simmdan@kenya.isu.edu

Version:

Unknown

Availability:

www.miscrit.be/~ddw

Description:

The other day I did a quick hack to nuweb.el as included with the nuweb distribution so as to make a funnelweb-mode.el. I've only used it briefly, and I'm sure that it can be improved quite a bit. I've been thinking about adding support for folding on sections, a pull-down menu to select macro definitions (like the recent functions posted to gnu.emacs.sources for a C function definition pull-down menu) and some kind of tags support for funnelweb.

Support:

Unknown

10.8 noweb.el

Developer:

Bruce Stephens (no email contact)

Version:

Unknown.

Availability:

Lost

Description:

This is a very simple mode I just hacked up. There's a lot wrong with it, but I thought others may be interested, even as it stands. It *requires* text properties, and assumes those used in GNU Emacs 19.22; it'll quite likely work with Lucid Emacs, but I haven't tried it.

I use it with auctex8.1 and cc-mode 3.229, both of which are loaded separately (I think my emacs

is dumped with them, in fact).

The idea is to have one mode (which calls itself c-mode, but actually has LaTeX-mode keybindings) generally (this means that the code is hilighted nicely), and have the code chunks use a different keymap.

Support:

Unknown

10.9 noweb-outline.el

Developer:

Dan Schmidt dfan@alum.mit.edu

Version:

0.0.3

Hardware:

Any platform with Emacs

Languages:

noweb

Availability:

www.dfan.org/real/noweb-outline.el

Readme:

www.dfan.org/real/noweb-outline.html

Support:

email to the author, dfan@alum.mit.edu

Description:

noweb-outline.el is a mode for Emacs that allows you to easily navigate the chunk tree of a noweb program.

One of the problems with literate programming is that it's easy to lose track of how your tangled source file (the one that the compiler actually sees) is structured. In noweb-outline-mode, you can interactively explore the tree of chunks you are creating, giving you the big picture as well as the small. Enough description; it would take more time for me to explain it than for you to just go ahead and try it out.

noweb-outline.el is currently in an alpha state (I've worked on it for only a couple of days), but it is already very useful. A nice file to use to try it out is example/wc.nw in the noweb distribution.

10.10 nuweb.el

Developer:

Dominique de Waleffe ddw@acm.org

Version:

1.99

Availability:

Anonymous ftp from CTAN

Description:

Provides a major mode extending Auctex for editing nuweb files. Main features (in 2.0):

- Edit scrap bodies in a separate buffer in a different mode (selected using emacs defaults for files, specific indication -*-mode-*-, or a buffer-local variable)
- Extends Auctex commands so that nuweb is called before LaTeX,
- Easy navigation on scrap definition and use points.
- Now creates an imenu (C-M-mouse1) with user index entries, macro definition positions and file definition positions.

Support:

Email to ddw@acm.org

10.11 Web mode

Developer:

Bart Childs bart@cs.tamu.edu

Version:

Unknown

Tools supported:

web, fweb, cweb, funnelweb

Availability:

Anonymous ftp ftp.cs.tamu.edu:pub/tex-web/web/EMACS.web-mode
thrain.anu.edu.au:pub/web/EMACS.web-mode

Description:

This version works with versions 18 and 19 of Emacs to be best of my knowledge. I have cleaned up a number of documentation items ... In the same directory is wm_refcard.tex which is an edited version of the famous one to include some web-mode commands.

The files limbo* are related to its use and notice that half them have an uppercase L in them for LaTeX. The setup is based upon the fact that we (I am not alone here) primarily use FWEB for C and Fortran programming.

We are using version 1.40 of FWEB although John Krommes warns that it is not mature and the manual is not yet updated. The info files are! We are using LaTeX almost exclusively. That will likely change and we will revert to version 1.30 if the final form of 1.40 cannot return to the simple section numbers and avoid the HORRIBLE LATEX 0.1.7.2.4.6 type section numbers.

Support:

Unknown

Next Previous Contents

Next Previous Contents

11. What other resources are available?

11.1 TeX Resources

Another resource of interest to literate programmers is the comp.text.tex newsgroup. If you're using (La)TeX as your typsetting system and have access to internet, then you should investigate this resource.

Another reason the TeX resources should be important is that so many of the literate programming tools rely on either plain TeX or LaTeX as their text formatter. (La)TeX software systems exist for most computing platforms. These systems can be found on CTAN and other major archive sites. Use archie to find them or simply ftp to one of the CTAN sites and browse.

Next Previous Contents

Next Previous Contents

12. Are there any code examples?

Examples of web programs are included with the FWEB, CWEB, and noweb distributions. nuweb is written in itself.

Cameron Smith converted the K&R calculator program into a literate program. It can be retrieved by anonymous ftp from:

```
niord.shsu.edu [192.92.115.8] directory kr-cweb-sample as
  krcwsamp.zip
or from
  LPA/Documentation
```

Ross Williams has released a funnelweb example. You can retrieve this file from node `ftp.adelaide.edu.au` [129.127.40.3] as

```
/pub/funnelweb/examples/except.*
```

This file should be on CTAN as well.

Lee Wittenberg has posted a few litprog examples. They are available via anonymous ftp from:

```
ftp://samson.kean.edu/pub/leew/samples.LP
```

The Stanford GraphBase is a large collection of programs by Don Knuth for doing all kinds of computations and games with graphs; it is written in (Levy/Knuth) CWEB. More details in the distribution. It is available via anonymous ftp from:

```
labrea.stanford.edu:/pub/sgb
```

Next Previous Contents

Next Previous Contents

13. Bibliographies

Nelson Beebe has collected an extensive bibliography treating literate programming. His work is available for anonymous ftp from ftp://ftp.math.utah.edu/pub/tex/bib/index.html#litprog. Be sure to look around this site; there are many things of interest to user of TeX resources as well as literate programmers.

Although I have not verified this, LPA is an alternate source for these files. Note that they are updated frequently (Nelson says several times each week), so be sure to get a fresh copy before extensive use. Joachim Schrod indicates that these files may be updated daily and can be retrieved via anonymous ftp at LPA/documentation.

Next Previous Contents

Next Previous Contents

14. Other Opinions about Literate Programming

14.1 van Ammers

An author (Eric W. van Ammers) wrote me a short article treating his opinions on literate programming.

First observation on LP

About 90% of the disussion on this list is about problems with applying some WEB-family member to a particular programming language or a special documentation situation. This is ridiculous, I think. Let me explain shortly why.

Lemma 1:

I have proposed for many years that programming has nothing to do with programming langauges, i.e. a good programmer makes good programs in any language (given some time to learn the syntax) and a bad programmer will never make a good program, no matter the language he uses (today many people share this view, fortunately).

Lemma 2:

Literate Programming has (in a certain way not yet completely understood) to do with essential aspects of programming.

Conclusion 1:

A LP-tool should be independent of programming language.

Lemma 3:

It seems likely that the so called BOOK FORMAT PARADIGM [ref. 1] plays an important role in making literate programs work.

Lemma 4:

There are very many documentation systems currently being used to produce documents in the BOOK FORMAT.

Conclusion 2:

A LP-tool should be independent of the documentation system that the program author whishes to use.

My remark some time ago that we should discuss the generic properties of an LP-tool was based on the above observation.

References:

[1] Paul W. Oman and Curtus Cook. ``Typographical style is more than cosmetic." *CACM* 33, 5, 506-520

(May 1990)

Second observation on LP

The idea of a literate program as a text book should be extendend even further. I would like to see a literate program as an (in)formal argument of the correctness of the program.

Thus a literate program should be like a textbook on mathematicics. A mathematical textbook explains a theory in terms of lemma and theorems. But the proofs are never formal in the sense that they are obtaind by symbol manipulation of a proof checker. Rather the proofs are by so called ``informal rigour'', i.e. by very precise and unambiguous sentences in a natural language.

Eric W. van Ammers <ammers@rcl.wau.nl>

14.2 Ramsey

Another author (Norman Ramsey) wrote me and asked that his opinions be included in the FAQ. What follows are Norman's comments verbatim.

I see it's time for the ``how is literate programming different from verbose commenting'' question. Perhaps David Thompson will get this into the FAQ. Alert! What follows are my opinions. In no way do I claim to speak for the (fractious) literate-programming community.

How is literate programming different from verbose commenting?

There are three distinguishing characteristics. In order of importance, they are:

1. flexible order of elaboration
2. automatic support for browsing
3. typeset documentation, especially diagrams and mathematics

Flexible order of elaboration means being able to divide your source program into chunks and write the chunks in any order, independent of the order required by the compiler. In principle, you can choose the order best suited to explaining what you are doing. More subtly, this discipline encourages the author of a literate program to take the time to consider each fragment of the program in its proper sphere, e.g., not to rush past the error checking to get to the ``good parts.'' In its time and season, each part of the program is a good part. (This is the party line; your mileage may vary.)

I find the reordering most useful for encapsulating tasks like input validation, error checking, and printing output fit for humans --- all tasks that tend to obscure ``real work'' when left inline. Reordering is less important when using languages like Modula-3, which has exceptions and permits declarations in any order, than when using languages like C, which has no exceptions and requires declaration before use.

Automatic support for browsing means getting a table of contents, index, and cross-reference of your program. Cross-reference might be printed, so that you could consult an index to look up the definition of an identifier `foo'. With good tools, you might get a printed mini-index on every page if you wanted. Or if you can use a hypertext technology, cross-reference might be as simple as clicking on an identifier to reach its definition.

Indexing is typically done automatically or `semi-automatically', the latter meaning that identifier definitions are marked by hand. Diligently done semi-automatic indexes seem to be best, because the author can mark only the identifiers he or she considers important, but automatic indexing can be almost as good and requires no work. Some tools allow a mix of the two strategies.

Some people have applied literate-programming tools to large batches of legacy code just to get the table of contents, index, and cross-reference.

I don't use diagrams and mathematics very often, but I wouldn't want to have to live without them. I have worked on one or two projects where the ability to use mathematical formulae to document the program was indispensible. I also wouldn't like to explain some of my concurrent programs without diagrams. Actually I write almost all of my literate programs using only sections headers, lists, and the occasional table.

```
>Wouldn't it be easier to do one's literate programming using
>a wysiwyg word processor (e.g. Word for Windows) and
>indicate what is source code by putting it in a different
>font?
```

The data formats used in wysiwyg products are proprietary, and they tend to be documented badly if at all. They are subject to change at the whim of the manufacturer. (I'll go out on a limb and say there are no significant wysiwyg tools in the public domain. I hope the Andrew people will forgive me.) These conditions make it nearly impossible to write tools, especially tools that provide automatic indexing and cross-reference support. The CLiP people have a partial solution that works for tools that can export text --- they plant tags and delimiters throughout the document that enable the reordering transformation (``tangling'').

People use TeX, roff, and HTML because free implementations of these tools are widely available on a variety of platforms. TeX and HTML are well documented, and TeX and roff are stable. TeX is the most portable. I think I have just answered the FAQ ``how come all these tools use TeX, anyway?'' :-)

Norman Ramsey

14.3 My (Dave Thompson's) Experience

In contrast to Eric's and Norman's comments, I'd like to interject from an anecdotal perspective.

I first ran across the idea of literate programming in 1992 while poking around George Greenwade's TeX archive (at niord.shsu.edu) and stumbling on some of the tools. My first experience was tinkering with `cnoweb`, see Section cnoweb. I used `cnoweb` to document a simple Bernoulli equation toy I built (in C) while working on a one-dimensional hydrodynamic model (in Fortran). I was convinced that literate programming had promise (although `cnoweb` really qualifies as a pretty-printing tool).

After reading Sewell's book, I kept hunting through the tools available until I found things that worked for me. (More here as I have time to develop the story.)

14.4 Others

I recently received email from Dave Johnson <scope@faroc.com.au> about his work developing language independent techniques. The web site is www.dscope.com.au.

Next Previous Contents

15. How to anonymously ftp

Pretty much everything mentioned here is available by anonymous FTP. FAQ lists cross-posted to news.answers and rec.answers can be gotten from `rtfm.mit.edu` [18.181.0.24], under `/pub/usenet/news.answers` or under `/pub/usenet/more.specific.group.name`

"anonymous FTP" is just a way for files to be stored where anyone can retrieve them over the Net. For example, to retrieve the latest version of the literate programming FAQ, do the following:

```
> ftp rtfm.mit.edu              /* connect to the site; message follows */
> anonymous                     /* type this when it asks for your name */
> <your email address>          /* type your address as the password    */
> cd /pub/usenet                /* go to the directory you want to be   */
> cd comp.programming.literate  /* one level down (no slash).           */
> dir                           /* look at what's there                 */
> get literate-progamming-faq   /* get the file; case-sensitive         */
> quit                          /* stop this mysterious thing           */
```

If your FTP program complains that it doesn't know where the site you want to use is, type the numerical address instead of the sitename:

```
> ftp 18.181.0.24               /* connect with numerical address */
```

If you don't have ftp access, send e-mail to `mail-server@rtfm.mit.edu` with the single word "help" in the body of the message.

Getting binary files (executables, or any compressed files) is only slightly more difficult. You need to set binary mode inside FTP before you transfer the file.

```
> binary            /* set binary transfer mode  */
> ascii             /* set back to text transfer mode */
```

FAQs and spoiler lists are generally ascii files; everything else is generally binary files.

Some common extensions on binary files in archive sites are:

```
  .Z           Compressed; extract with uncompress
  .tar.Z       Compressed 'tape archive'; uncompress then untar or tar -xvf
  .gz or .z    Gnu gzip; use gunzip (available from prep.gnu.ai.mit.edu)
  .sit         (Mac) StufIt archive
  .zip         Extract with Zip or Unzip
  .zoo         Yet another archive/compress program
  .lhe         (Amiga) ?
  .lzh         Lha archive program.
  .arj         (PC) Arj archive program.
  .exe         (PC) Sometimes self-extracting archives-just execute them.
  .uue or .UUE Transfer as text file; use uudecode to convert to binary
```

```
.hqx          (Mac) BinHex format; transfer in text mode
```

Generic help can be found in the FAQs of comp.binaries. `<your_system_type>` for how to transfer, extract, and virus-check binary files. (At `rtfm.mit.edu`)

If you can't FTP from your site, use one of the following ftp-by-mail servers:

```
ftpmail@decwrl.dec.com
ftpmail@src.doc.ic.ac.uk
ftpmail@cs.uow.edu.au
ftpmail@grasp.insa-lyon.fr
```

For complete instructions, send a message reading "help" to the server.

If you don't know exactly what you're looking for, or exactly where it is, there are programs and servers that can help you. For more info, send e-mail to `mail-server@rtfm.mit`.with with the body of the message reading send usenet/news.answers/finding-sources

Thanks to Aliza R. Panitz (the "buglady") for this text. I copied it verbatim from her post on faq-maintainers with only minor modifications.

Next Previous Contents

Next Previous Contents

16. Acknowledgements

This document would not have happened without the help of many people. George Greenwade was instrumental in establishing the original mailing list way back in the early '90's (when I first became involved). Marcus Speh started one of the early `ftp` sites and was an active participant. Among them are, Rob Beezer, Joachim Schrod, Piet van Oostrum, Ross N. Williams, Nelson H. F. Beebe, and Andrew Johnson. We wouldn't have literate programming if it wasn't for Donald Knuth and TeX. Certainly, we wouldn't be where we are without the tool developers (all credited in their entries above).

Cesar Bellardini cballard@santafe.com.ar deserves thanks for translating the FAQ into Spanish.

A special thanks to Aliza R. Panitz for the text describing how to execute an anonymous ftp for files of interest.

Any omissions from these acknowledgements should be considered an accident on my part. Furthermore, participants in the `comp.programming.literate` newsgroup all contributed in various fashions. Thank all of you.

Next Previous Contents

Next Previous Contents

---

17. End notes

This document remains in a state of evolution (although I'm a strict creationist! <grin>). I'm working on the SGML version to improve formatting of the resulting documents. I'm also reorganizing the FAQ to improve its usability. Comments are solicited for such improvements. Omission of a particular tool should not be considered a snub in any sense--simply an error or oversight on my part.

Next Previous Contents