Clojure in Common Lisp

Timothy Daly

November 15, 2011

Contents

Preface: Why Literate Programming **i**
- Steps to build Clojure i
- Steps to change Clojure ii
- Why Bother? iii

1 From ideas to implementation **1**

2 The clojlisp package **3**

3 The Reader **5**

4 The Evaluator **7**

5 The Printer **9**

6 The Clojlisp REPL **11**

7 Building ClojLisp **13**
- 7.1 The whole thing 14
- 7.2 tangle.c 15
- 7.3 Makefile 19
- 7.4 The Clojure Icon 20
- 7.5 License 26

Foreword

Rich Hickey invented Clojure. This is a fork of the project to build Clojure in Common Lisp.

We would like to have these new lisp ideas available in a standard common lisp environment. There will be some language changes due to the loss of Java and some enhancements due to Common Lisp. This is unavoidable but within the spirit of Clojure.

Every effort is made to give credit for any and all contributions.

We are also experimenting with literate programming as a development and documentation technology.

Clojure is a break with the past traditions of Lisp. This literate fork is a break with the past traditions of code development. As such it is intended as an experiment, not a replacement or competition with the official version of Clojure.

Timothy Daly
November 10, 2011 ((iHy))

Preface: Why Literate Programming

This is a literate program, inspired by Donald Knuth [Knu84]. It is intended to be read like a novel from cover to cover. The ideas are expressed clearly but they are grounded in the actual source code.

The code in this document is the executable source. The chapter [13] building on Building ClojLisp gives the procedure for building a running system from the enclosed sources.

Most programmers are still locked into the idea of making a program out of a large pile of tiny files containing pieces of programs. They do not realize that this organization was forced by the fact that machines like the PDP 11 only had 8k of memory and a limit of 4k buffers in the editor. Thus there was a lot of machinery built up, such as overlay linkers, to try to reconstruct the whole program.

The time has come to move into a more rational means of creating and maintaining programs. Knuth suggested we write programs like we write literature, with the idea that we are trying to communicate the ideas to other people. The fact that the machine can also run the programs is a useful side-effect but not important.

Very few people have seen a literate program so this is intended as a complete working example, published in book form. The intent is that you can sit and read this book like any other novel. At the end of it you will be familiar with the ideas and how those ideas are actually expressed in the code.

If programmers can read it and understand it then they can maintain and modify it. The ideas will have been communicated. The code will be changed to match changes in the idea. We will all benefit.

I've tried to make it as simple as possible. Try it once, you might like it.

Steps to build ClojLisp

Step 1

We will be using SBCL as the initial development environment so you need to get an SBCL version running.

You also need the C program [15] tangle.c which you can clip from this file using a text editor and save it as tangle.c. If you got this file from a git repository the tangle.c program is probably already available. The tangle program is used to extract the source files from this document.

Step 2

Compile tangle.c to create a function called tangle.

```
gcc -o tangle tangle.c
```

Step 3

Run tangle to extract the [19] Makefile from this document.

```
./tangle clojure.pamphlet Makefile >Makefile
```

Step 4

Running make will create a PDF of the documentation and will create the clojlisp.lisp file containing the latest source code.

```
make
```

Step 5

Start lisp, change to the new package, and start the REPL.

```
sbcl
(load "clojlisp.lisp")
(in-package "CLOJLISP")
(clojlisp)
```

The new REPL knows almost nothing at all. In particular, it has no functions and no symbols defined yet so you need to qualify everything. For example,

```
(common-lisp::+ 2 3)
```

To exit the program type:

```
(sb-ext:quit)
```

Steps to change Clojure

Working in literate programming is simple. You need an editor and a command line. Make changes directly to this document using the editor and then type

```
make
```

This will destroy the old source and rebuild the clojlisp.lisp file.

If you change the Makefile section in the document you will have to extract it before the changes take effect:

```
./tangle clojlisp.pamphlet Makefile >Makefile
```

If you are working in Linux it is very effective to use emacs with a split screen. Start a command line in one buffer and a shell in the other buffer. If you start xdvi running in the background you will immediately see the “printed” form of the document every time you switch to the xdvi window.

```
xdvi clojlisp.dvi &
```

The same trick can be used with the PDf file instead. Both xdvi and xpdf will update automatically when the file is changed on disk.

```
xpdf clojlisp.pdf &
```

Resist the urge to edit the clojlisp.lisp file. It is only there for the computer. Edit this file directly. Be sure to write words to communicate your ideas just as you would when writing a book. The machine code is secondary and intended to make the ideas concrete.

Why Bother?

Why bother with such a difficult method of programming? Because worthwhile programs should "live".

Programs "live" because people maintain them. Maintaining and modifying code correctly requires that you understand why the program is written as it is, what the key ideas that make the program work, and why certain, not very obvious, pieces of code exist. Programmers almost never write this information down anywhere. Great ideas are invented, the code is written, a man page of documentation is created, and the job is done.

Well, almost. What does is mean for a program to "live"? How does a program survive once the original developers leave the project? There are many sources of information but almost no source of knowledge. New programmers don't know what the "elders" know. In order to "live" and continue to grow there has to be a way to transmit this knowledge.

Literate programming is Knuth's proposed idea for moving from the world of ideas to the world of information. This is not simply another documentation format. This is meant to be **Literature**. The ideas are presented, the implications are explored, the tradeoffs are discussed, and the code is "motivated", like characters in a novel.

You are encouraged to write or rewrite sections of this document to improve the communication with the readers.

"But I have to learn latex!". Well, for this document you do. But LaTeXis no more than a document markup language like HTML and it is no harder to learn. It gives you the added advantage that you have a real language for publishing real documents. Most books are typeset with this technology and a lot of conferences and Journals require it. If you can learn Clojure, you can learn LaTeX. If you're a programmer you will always need to continue to learn, at least until you retire into management.

Having used literate programming for years I have collected some key quotes that might stimulate your interest.

I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by considering programs to be works of literature. Hence, my title "Literate Programming". Let us change our traditional attitude to the construction of programs. Instead of imagining that our main task is to instruct a computer what to do, let us concentrate on explaining to human beings what we want a computer to do.
–Donald Knuth "Literate Programming (1984)"

Step away from the machine. Literate programming has nothing to do with tools or style. It has very little to do with programming. One of the hard transitions to literate programming is "literate thinking".
–Timothy Daly in Lambda the Ultimate (2010)

The effect of this simple shift of emphasis can be so profound as to change one's whole approach to programming. Under the literate programming paradigm, the central activity of programming becomes that of conveying meaning to other intelligent beings rather than merely convincing the computer to behave in a particular way. It is the difference between performing and exposing a magic trick.
–Ross Williams, FunnelWeb Tutorial Manual

Another thing I've been enjoying lately is literate programming. Amazingly it turns out to be faster to write a literate program than an ordinary program because debugging takes almost no time.
–Bill Hart, SAGE Mailing list, May 3, 2010

The conversation is much more direct if the Design Concept per se, rather than derivative representatives or partial details, is the focus.
–Fred Brooks, "The Design of Design"

We are banning the old notion of literate programming that I used when developing TEX82 because documentation has proven to be too much of a pain.
–Donald Knuth TUG 2010

Once upon a time I took great care to ensure that TEX82 would be truly archival so that results obtainable today would produce the same output 50 years from now but that was manifestly foolish. Let's face it, who is going to care one whit for what I do today after even 5 years have elapsed, let alone 50. Life is too short to re-read anything anymore in the internet age. Nothing over 30 months old is trustworthy or interesting.
–Donald Knuth TUG 2010

Chapter 1

From ideas to implementation

Chapter 2

The clojlisp package

We are defining a lisp-in-lisp so there will be a significant number of symbol collisions. We have to be very careful to only import symbols that do not conflict.

— defpackage clojlisp —

```
#+:AKCL (make-package "COMMON-LISP" :use '("LISP"))
#+:AKCL (in-package "USER")
(make-package "CLOJLISP")
(in-package "CLOJLISP")
```

————-

Chapter 3

The Reader

The reader uses the common lisp reader.

— **defun read** —

```
(common-lisp::defun clojlisp::read () (common-lisp::read))
```

———-

Chapter 4

The Evaluator

The evaluator uses the common lisp evaluator.

— defmacro eval —

```
(common-lisp::defmacro clojlisp::eval (x) `(common-lisp::eval ,x))
```

————-

Chapter 5

The Printer

The printer uses the common lisp printer.

— **defun print** —

```
(common-lisp::defun clojlisp::print (x) (common-lisp::print x))
```

———-

Chapter 6

The Clojlisp REPL

By typing

```
(load "clojlisp.lisp")
(in-package "CLOJLISP")
(clojlisp)
```

you have a read-eval-print loop (REPL). The REPL knows almost nothing about anything so it can be quite painful. You need to qualify symbols that come from any other package at the moment. A working example is:

```
(common-lisp::+ 2 3)
```

— defun repl —

```
(common-lisp::defun clojlisp::clojlisp ()
 (common-lisp::loop
  (clojlisp::print
    (clojlisp::eval
     (clojlisp::read)))
  (common-lisp::terpri)))
```

Chapter 7

Building ClojLisp

This is the whole source listing in one file. When loaded, it leaves you in the clojlisp REPL. If you define a new chunk, add it here.

7.1 The whole thing

— clojlisp —

```
\getchunk{defpackage clojlisp}
\getchunk{defun read}
\getchunk{defmacro eval}
\getchunk{defun print}
\getchunk{defun repl}
```

———-

7.2 tangle.c

The tangle program extract pieces of this document and puts them in a file. Any piece that can be extracted is called a “chunk”. Each chunk is in a LaTeX**chunk** environment. The tangle program expects two arguments, the name of this file and the chunk to extract as in:

```
./tangle clojure.pamphlet clojlisp
```

It outputs the chunk to standard output. To be useful, redirect the output to a file as in:

```
./tangle clojure.pamphlet clojlisp >clojlisp.lisp
```

This trivial function walks the document looking for the named chunk. When it finds it the chunk is printed to standard output. The only exception is if the chunk contains a **getchunk** markup. This is used to embed other chunks inline. So tangle will pause the output of the current chunk, find and print the getchunk reference, and then continue. Of course, if the getchunk reference contains an embedded getchunk we recurse into it. This program could be a lot smarter . . . but why?

— **tangle.c** —

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <sys/types.h>
#include <fcntl.h>

#define DEBUG 0

/* forward reference for the C compiler */
int getchunk(char *chunkname);

/* a memory mapped buffer copy of the file */
char *buffer;
int bufsize;

/* return the length of the next line */
int nextline(int i) {
  int j;
  if (i >= bufsize) return(-1);
  for (j=0; ((i+j < bufsize) && (buffer[i+j] != '\n')); j++);
  return(j);
}
```

```
/* output the line we need */
int printline(int i, int length) {
  int j;
  for (j=0; j<length; j++) { putchar(buffer[i+j]); }
  printf("\n");
}

/* handle begin{chunk}{chunkname}        */
/* is this chunk name we are looking for? */
int foundchunk(int i, char *chunkname) {
  if ((strncmp(&buffer[i+14],chunkname,strlen(chunkname)) == 0) &&
      (buffer[i+13] == '{') &&
      (buffer[i+14+strlen(chunkname)] == '}')) return(1);
  return(0);
}

/* handle end{chunk}   */
/* is it really an end? */
int foundEnd(int i) {
  if ((buffer[i] == '\\') &&
      (strncmp(&buffer[i+1],"end{chunk}",10) == 0)) {
    return(1);
  }
  return(0);
}

/* handle getchunk{chunkname} */
/* is this line a getchunk?    */
int foundGetchunk(int i, int linelen) {
  int len;
  if (strncmp(&buffer[i],"\\getchunk{",10) == 0) {
    for(len=0; ((len < linelen) && (buffer[i+len] != '}')); len++);
    return(len-10);
  }
  return(0);
}

/* Somebody did a getchunk and we need a copy of the name */
/* malloc string storage for a copy of the getchunk name  */
char *getChunkname(int k, int getlen) {
  char *result = (char *)malloc(getlen+1);
  strncpy(result,&buffer[k+10],getlen);
  result[getlen]='\0';
  return(result);
}

/* print lines in this chunk, possibly recursing into getchunk */
int printchunk(int i, int chunklinelen, char *chunkname) {
  int j;
  int k;
  int linelen;
  char *getname;
  int getlen = 0;
  for (k=i+chunklinelen+1; ((linelen=nextline(k)) != -1); ) {
    if (DEBUG==2) {
      printf(">>>>"); printline(k,linelen); printf("<<<<\n");
    }
    if ((getlen=foundGetchunk(k,linelen)) > 0) {
       getname = getChunkname(k,getlen);
       getchunk(getname);
       free(getname);
       k=k+getlen+12l;
    } else {
    if ((linelen >= 11) && (foundEnd(k) == 1)) {
      if (DEBUG) { printf("=================\\end{%s}\n",chunkname); }
      return(k+12);
    } else {
      if (DEBUG==2) {
        printf("======== printchunk else %d %d\n",k,linelen);
      }
      printline(k,linelen);
      k=k+linelen+1;
    }
  }}
  if (DEBUG==2) {
     printf("=================\\out{%s} %d\n",chunkname,k);
  }
  return(k);
}

/* find the named chunk and call printchunk on it */
int getchunk(char *chunkname) {
  int i;
  int j;
  int linelen;
  int chunklen = strlen(chunkname);
  for (i=0; ((linelen=nextline(i)) != -1); ) {
    if (DEBUG==2) {
      printf("----"); printline(i,linelen); printf("----\n");
    }
    if ((linelen >= chunklen+15) && (foundchunk(i,chunkname) == 1)) {
      if (DEBUG) {
         fprintf(stderr,"=================\\getchunk(%s)\n",chunkname);
      }
      i=printchunk(i,linelen,chunkname);
    } else {
      i=i+linelen+1;
    }
  }
  if (DEBUG) {
    fprintf(stderr,"=================getchunk returned=%d\n",i);
  }
  return(i);
}

/* memory map the input file into the global buffer and get the chunk */
int main(int argc, char *argv[]) {
  int fd;
  struct stat filestat;
  if ((argc < 2) || (argc > 3)) {
    perror("Usage: tangle filename chunkname");
    exit(-1);
  }
  fd = open(argv[1], O_RDONLY);
  if (fd == -1) {
    perror("Error opening file for reading");
    exit(-2);
  }
  if (fstat(fd,&filestat) < 0) {
    perror("Error getting input file size");
    exit(-3);
  }
  bufsize = (int)filestat.st_size;
  buffer = mmap(0,filestat.st_size,PROT_READ,MAP_SHARED,fd,0);
  if (buffer == MAP_FAILED) {
    close(fd);
    perror("Error reading the file");
    exit(-4);
  }
  getchunk(argv[2]);
  close(fd);
  return(0);
}
```

7.3 Makefile

This book is actually a literate program[Knu84] and can contain executable source code. In particular, the Makefile for this book is part of the source of the book and is included below.

You can extract this chunk with an editor or use the tangle function:

```
./tangle clojlisp.pamphlet Makefile >Makefile
```

— **Makefile** —

```
PROJECT=clojlisp
TANGLE=tangle
LATEX=/usr/bin/latex
MAKEINDEX=/usr/bin/makeindex
UUDECODE=uudecode
PDF=dvipdf

all:
${TANGLE} ${PROJECT}.pamphlet ${PROJECT} >${PROJECT}.lisp
${TANGLE} ${PROJECT}.pamphlet clojureIcon.eps.uu >clojureIcon.eps.uu
${UUDECODE} clojureIcon.eps.uu
${LATEX} ${PROJECT}.pamphlet 2>/dev/null 1>/dev/null
${MAKEINDEX} ${PROJECT}.idx
${LATEX} ${PROJECT}.pamphlet
${PDF} ${PROJECT}.dvi
```

———-

7.4 The Clojure Icon

This document uses the Clojure Icon on the front page. This is a uuencoded, encapsulated postscript version of the icon. You can recreate the **clojureIcon.eps** file with:

```
./tangle clojlisp.pamphlet clojureIcon.eps.uu >clojureIcon.eps.uu
uudecode clojureIcon.eps.uu
```

— **clojureIcon.eps.uu** —

```
begin 644 clojureIcon.eps
M)2%04RU!9&]B92TS+C`@15!31BTS+C`*)25#<F5A=&]R.B!'24U0(%!O<W13
M8W)I<'0@9FEL92!P;'5G:6X@5B`Q+C$W(&)Y(%!E=&5R($MI<F-H9V5S<VYE
M<@HE)51I=&QE.B!C;&]J=7)E+6EC;VXN97!S"B4E0W)E871I;VY$871E.B!-
M;VX@3F]V(#$T(#(Q.C,U.C0S(#(P,3$*)25$;V-U;65N=$1A=&$Z($-L96%N
M-T)I=`HE)4QA;F=U86=E3&5V96PZ(#(*)25086=E<SH@,0HE)4)O=6YD:6YG
M0F]X.B`Q-"`Q-"`Q,34@,3$U"B4E16YD0V]M;65N=',*)25"96=I;E!R;VQO
M9PHE(%5S92!O=VX@9&EC=&EO;F%R>2!T;R!A=F]I9"!C;VYF;&EC=',*+3`@
M9&EC="!B96=I;@HE)45N9%!R;VQO9PHE)5!A9V4Z(#$@,0HE(%1R86YS;&%T
M92!F;W(@;V9F<V5T"C$T+C$W,S(R.#$T+C$W,S(R.#`@=')A;G-L871E"C$T
M-38V.30@=')A;G-L871E(#`@-V5A9VAT"C,X(&)E9VEN(&]F(&9I<G-T
M('-C86YL:6YE"C`@.3DN.3DY.3DY.3DY.3DY.3@V('1R86YS;&%T90HY.2XY
M.3DY.3DY.3DY.3DY.#8@+3DY+CDY.3DY.3DY.3DY.3DX-B!S8V%L90HE($EM
M86=E(&=E;VUE=')Y"C$P,"`Q,#`@.`HE(%1R86YS9F]R;6%T:6]N(&UA=')I
M>`I;(#$P,"`P(#`@,3`P(#`@,"!=*"B4@4W1R:6YG<R!T;R!H;VQD(%)'0BUS
M86UP;&5S(')!E<B!S8V%N;&EN90HO<G-T<B`Q,#`@<W1R:6YG(&1E9@HO9W-T
M<B`Q,#`@<W1R:6YG(&1E9@HO8G-T<B`Q,#`@<W1R:6YG(&1E9@I[8W5R<F5N
M=&9I;&4@=&4@+T%30TE).#5$96-O9&4@9FEL=&5R(&]2=&6Y,96YG=&A$96-O9&4@
M9FEL=&5R(')S='(@<F5A9'-T<FEN9R!P;W!]"GMC=7)R96YT9FEL92!T9FEL92`O
M24DX-41E8V]D92!F:6QT97(@+U)U;DQE;F=T:$1E8V]D92!F:6QT97(@9W-T
M<B!R96%D<W1R:6YG('!O<'T*>V-U<G)E;G1F:6QE(&=S=')I;F<@-T-)23@U1&5C;V1E
M(&9I;'1E<B`O4G5N3&5N9W1H1&5C;V1E(&9I;'1E<B!B<W1R(')E861S=')I
M;F<@<&]P?0IT<G5E(#,*)25"96=I;D1A=&$Z("`@("`@("`@.3DU,R!!4T-)
M22!">71E<PIC;VQO<FEM86=E"E-C/3-^/@I38STS?CX*4V,],]+,WX^"E-C/3-^
M/@I38STS?CX*4V,],],+,WX^"E-C/3-^/@I38STS?CX*4V,],],+,WX^"E-C/3-^/@I3
M8STS?CX*4V,],],+,WX^"E-C/3-^/@I38STS?CX*4V,],],+,WX^"F<F1#9+95H^*FA$
M/D`Y.T0Q7"M596(N5G%*+'X^"F<F1#9-:31U4FI/;FIO,T]E2R)P:3IK<2E*
M+'X^"F<F1#93<"1#;"1H/BQ<+&@[6U9A<"8S4T1*+'X^"F@C0$@A5&TL;2LA
M9$,_76@C1&U^/@IH(T!(+EP]*6`X(6@D4DYH(T1M?CX*:"-`2$EL+S%M7"%O
M3C@V:"-$;7X^"FAU/&,D45A=*5HA2SXT6G,J='X^"FAU/&,Q65]H3&XA3CU:
M+G,J='X^"FAU/&-,.TTC*$XA5&!K,',J='X^"FER.2E16$11;W4A8B\W<FER
M/4Y^/@II<CDI4UYL9%0L(68]7UUI<CU.?CX*:"H*9'2,.7(Y*$Y*5EM*W!842%O(2,[:7(]
M3GX^"FI3;SM(45A<8%`A3$XJ-'X^"FI3;SM-65]H+F0A3S(E4G,J='X^"FI3
M;SM8.TTB7T0A52<]/W,J='X^"FLU4$U56$-"9V$A350R2W,J='X^"FLU
M4$U77FMP77`A4"5M9',J='X^"FLU4$U=;2M5*T4A53E21G,J='X^"FMO:U,K
MO&(B45U`7V@Q+$HL?CX*:U!K4SA-.DDN6$TY=$=C2BQ^/@IK4&M34V<]/6=6
M9T`T.RI*+'X^"FPR3&573C9T8"Y+)TU6(THL?CX*;#),95E7-G`N<U4D3<3=
M2BQ^/@IL,DQE7VI/*51<:7%H<#Y*+'X^"FQ-9VXNO&%!+5=`82(G.DHL?CX*
M;$UG;CM-.6=?4DTZ:"MN2BQ^/@IL36=N5F<\7$-O9T!/5C!*+'X^"FQI+7-9
M8C\K8C)L:3)*?CX*;&DM<W!BOTM9=&QI,DI^/@IL:2UT3&),8V=8;&DR2GX^
M"FU*9#1;3C8L,"9'4#%!)THL?CX*;4ID-%U7-B=3:U)*-BM+2BQ^/@IM2FOT
M8VI.-B14:'4R<T)*+'X^"FUF*CU<1TLS961'3R)6<DHL?CX*;68J/5Y212=J
M6E))OE-$2BQ^/@IM9BH]9&AO1D!-:'1L9$!*+'X^"FXL149=1TLA66)`8S9?
M5$HL?CX*;BQ%1E]21&I>6$T\1D`M2BQ^/@IN+$5&96AO-#1+9T$H+CI*+'X^
M"FY'8$]>1TID36¡8S9B54HL?CX*;D=@3V!21%A25DT\1D,N2BQ^/@IN1V!/
M9FAO(BA)9T$H,3M*+'X^"FYC)EA?1U!K4$-$/F8E9$!A)7!41#XA2"%*+'X^
M"FYC)EAA4DI?53E/;SIS4DTY3$U/3VY<1$=*+'X^"FYC)EAG:'4I*RQH/E$]
M-6<\03%-:#Y1;41*+'X^"FYC)F,Z/4-25S1I.RI"5&DT=%MR8W)>/&UP)28M
M-7X^"FYC)F-<2E9G5EAK4#XL6VM+<$=Q9"(I-&-P6UP_-WX^"FYC)F1-9EPU
M3EAQ(F%P;'$A9$HF9"M!0E9R551U/7X^"F\IO6-82R,D<%8A5#D]03TY3BEL
M6$]:."MR<F@Y,5A'9U=//3DK/$YS*G1^/@IO*4%C<54A5S\P(54D76%*2%5H
M+EYU-$11<G)H3D->;S,K44I(,ED^<RIT?CX*;RE!9%-I<&-,/"%6:VI49D5"
M<&AM+DPO3G)R:2QI;2Q:;6!F1'0B)W,J/2='X^"F]$7&I,96$\03]0<#AC:54C
M/DI`(6]48FAK=5MQ96]$83U^/@IO1%QJ4VDY9T]*<%14-"I<145$7B%P0"A5
M;;"4F:4UO1&$]?CX*;T1<:F1P)$UB7W)41&AE;#%J9DLA<C(D.6PN/R(F;T1A
M/7X^"F==)3DS;&DN(B]+)TA'/&5B)FQL?CX*9UTE.3IL:2XB/%4D0#$O:3I2
M)B)^/@IG724Y2VQI+B)7:7`L+G)P)3@Y-WX^"E]:)UEE1#QT0"I$/B%4)4HL
M?CX*7UHG67!).D(D-2)C(D%!)5YA*2Q-;G(V2"<D(7!=:C!]<BL_,&DG(V\C
M16$:!",!B+)<%'MDI^/@IE+%0T1G!*+'X^"EY\0S%D/2XS,U1#.C9=/#%&74PT
M1'X^"EY\0S%D0$4Y4UU-,S953"PU8$@K)UA>(D8X23%65BHY5RDQ-$8]4C)2
M-$\T14$M,T554#PR+%HP12XG,U-!*RTF-4(N,BU./T-8*DTY0D4G.S11
M2SHC5$@Z1RDN.R59&#@S,%DQ)%55*%-))B8E*2DK)6!*.C-)1"Q"1CY>+T!)
M12LE,%0P1R$A38,A1SHG+2LI-RE0*31))2XW6%XB1E$M.59-+CQ#/2PU5&!!
M+$54734B)4DO-$$B/#5%4#4F03$Y52E2-31,2#E#,5`_0U@J.R15
M1SI5/5(N-4!%-2(D531Y*%@H-E$F*#8V75$V-R$T.D5%2C1".2(U,C4\,S-)
M7B]`24PS-CU+,5<H5BTF,%0[4D4A,48D035434E,S)CDD/%<D23Q2250_0U@J
M.S8X2C!5-3TN)"TJ*30A1C5V-$P\1RDM.T554RXF2$0O5B4S+T4\4R@S)%,J
M5314+34M8"M42$P_0U@J.S8X2C!5654N)25)*#5%3#-$/2P\1RDM/3)14"\E
M43,L)CTA+D5-)R@S+4L[-"A!+5-)*R]42$P_0U@J.S8X2C!464V*T0Q33E#
M)$<H-B1"7-37#TU*1B%2/$0P5C,V.20]

	
```

```
M,G%99T5P<%!*16LA-SI40DHL?CX*<%QT-390+F<]+7([46(S<&19/3AQ=3\D
M7G([469R9T`\97%N1SPP.'X^"G$C.C\K;S0E9"IR.U%C/'`U;T`Y<CM18G5M
M/G$]87([46)U;E$U9$UQ674G?CX*<2,Z/TU0/$$[2G([46-B<#U06D%R.U%C
M.VU"8VQ$<CM18S1N555</7%9=2=^/@IQ(SH^:F\N9SPV<CM18FQP+B,L,7([
M6BI><CM18V!N7FUJ)G%9=2=^/@IQ/E5+7D0]7S8[<"8T;6IN8B8B151$7%UH
M96$OO'1L35Y?8&]J6T=N<5EU)WX^"G$^54MG6V4D82AQ66=%;W%T.%]%:3M%
M5%1J-E@L46YB<DEG<%!*16MQ674G?CX*<3Y52U<U4"17,6\I.%)G:VLN1D<_
M36IC:')Q=6-S<E,D96]Q674G?CX*<3Y51U!O-"5C;'([46,A;UOY+BER.U%B
M8&T^<3U,<CM18F!N435E07%U.S!^/@IQ/E5(-F\\03M`<CM18UEO7#E(.W([
M46,M;4)C;#9R.U%C)&Y555QU<74[,'X^"G$^54<K;RYG.W%R.U%B/F],06YM
M<CM:*EYR.U%C6VY>;6HW<74[,'X^"G%9<%%*;FU?6U!R5FQL66\X<R1&<CM1
M8RYM(U8U+7)6;&Q5;E$U9&UQ=3LP?CX*<5EP45YO(28R77)6;&QK;T!S/RAR
M.U%C1&TG2&-:<E9L;%QN555<57%U.S!^/@IQ67!1/FYH3#-H<E9L;$9O,29D
M97([6BI><CM18VQN7FUJ+G%U.S!^/@IQ67!046\T)6-%<CM18RYN<E=Q+G)6
M;&Q1;2-6-%)R.U%B-FYL4&U$<74[,'X^"G%9<%$W;SQ!.RUR.U%C76\E6#8[
M<E9L;%UM)TAC.7([46):;G!P93=Q=3LP?CX*<5EP4"QO+F%[/G([46)5;FI@
M72)R.UHJ7G([46-1;R4S<R5Q=3LP?CX*<74V6E]N;5]-=SE)6'%U.S!^/@IR
M.U%B8&TC5C4[<E9L:VMN;%%!M)'%U.S!^/@IQ=39::&\A)C)$<E9L;%0;7STM
M*W([46,M;2=(8V1R5FQL+6YP<&1T<74[,'X^"G%U+5I8.VA,<E9L;%:6Y/
M15,E<CM:*EYR.U%C76\E,W)R5FQL2VX\(5\Y<E9L;%"AM(U8T;7)6;&QG;D0B
M<E9L;%"AM(U8T.W)6;&QG;D0B
M+VTO-FTV?CX*<CM18R5N;5]:=’)6;&Q9;E<\:#1R<CPC,’)6;&PH;5HW1RAR
M5FQL(6YL4&UN<CM6.7X^"G([46-,;R$F,D1R5FQL:VY?/2T\<G(\(U]R5FQL
MO&U>*758<E9L;#5N<’!E5G([5CE^/@IR.U%B8FYH3#,H<E9L;$9N3T54+’)R
M/")7<E9U.6%R.U%C8&\E,W,O<CM6.7X^"G([46,E;FU?6E=R5FQL7VYR5W!9
M<G(S)FE/93M54"$X4B(H(3D])5@A+G-C1"$S2"MU2BQ^/@IR.U%C3&\A)C(V
M<E9L;&UO)5@V+G)R,R9S9W4O+S(A.6EJ6"$Z.5MA(3(X=#<A-5-/-$HL?CX*
M<CM18F)N:$PR5G)6;&Q2;FI@7#)R<C,F8#=P8RQN;68J+F$A.3-3)2$Z.5A@
M2BQ^/@IR.U%C3&YM7UI$<CM18F%O.’,EOW)R,RE!3V-E<BUR<D%$4T]3;RM9
M1T]E:3EB4%]":GX^"G([46-@;R$F,BQR.U%C4F]‘<S]#<G(S*6=G=%]=)W)R
MOE5U6R]"<2A225EN+V9?:V,B?CX*<CM18T!N:$PR/7([46%Q;S$F9D5R<C,H
M<3=N.TOV<S9F;6%R<D-S269$=$]‘<RIT?CX*<CM18TQN4D12;’)6;&PH;UOY
M+6)R5FQT94]C9’5G<G)";BE/.’5"6W,W)S%K:3M%5BI^/@IR.U%C8&Y:8"EJ
M<E9L;%QO7#E(,G)6;’569W1?1’1R<D-5/5II2B]%<S<H6#]K4%E‘,7X^"G([
M46-‘;DTQ*SAR5FQK2V],O6X_<E9L<W4W;CEC77,V9FUA<S<K7T%Q(R@OOGX^
M"G%U/S%R(3<H44,A-S$[(B$Z9R=G(3=Q+C,A,F9?<"$Z9E$_(3<H44,A-V=8
MOB$L7U<U2BQ^/@IQ=3\R:R$Y83U<(3I=6#DA.VQC<2$Z<"U%(3D])5@A.SYO
M:"$X9%Q3(3DA1B$A,%(P64HL?CX*<74_,4$A-4I,-"$S8R-B(3EJ1EXA-’(O
M(B$L1$@S;D=@O&,A.S5P."$X6U!O2BQ^/@IQ=399/VYM7UIT<E9L;&UO;UOV
M8W)6;&MP<DLN*E1R5FQL*&Y7,V)‘<E9L:VMN;%!M8W%U.S!^/@IQ=39:+F\A
M)C)$<E9L;’)P(E11,W)6;&Q8<E,N1$]R5FQLO&Y;)CMI<E9L;"UN<’!E3G%U
M.S!^/@IQ=398:6YH3#,H<E9L;&EO9UTBO’)6;&LT<D,V:UMR5G5"9’([46-=
M;R4S<RQQ=3LP?CX*<74V65QN;5]:1’([46,G<#5OO$9R<C)U8’(O:"$Q<E9L
M;%AN<DYJ47([46)‘;FQO;E=Q=3LP?CX*<74V6CQO(28R+’([46-;<#UO6D9R
M<C)U;G(W:#M#<E9L;&%O(4%$.W([46)B;G!P9C!Q=3LP?CX*<74V63IN:$PR
M/7([46)*<"XC+$AR<C)U4W(G<&%U<CM:/&1R.U%C4V\E,W,,]<74[,’X^"G%U
M-EHN;E)$4EER5FQL67!1-4AE<E9L;"]R+V=U8W([46)U;SAI=#1R5FQL56YL
M4&U$<5EU)WX^"G%U-EIO;EI@*6!R5FQL:W!9-6,U<E9L;%YR-V@[.’([46,[
M;;SQ<36%R5FQL7&YP<&4W<5EU)WX^"G%U-EEM;DTQ*G1R5FQL1G!)/C1"<E9L
M:U9R)W!A/’([6C]E<CM18VQO)3-S)7%9=2=^/@IQ=39:56Y21%%J<CM18R%P
M;;%!22’)6;&MB<6E,;5)R5FQL6&]4,"=3<CM18FIN;%!N0G%9=2=^/@IQ=39:
M8VY:8"D^<CM18UEP=%!L2’)6;&Q3<7%-,DUR5FQL86]8(E8]<CM18RQN<’!F
M(7%9=2=^/@IQ=39:3&Y-,2EO<CM18CYP9%D^2G)6;&%IR<6%565ER.UI"9G([
M46-=;R4S<SAQ674G?CX*<5EP4#1N4D12;’)6;&Q?<3)K6F=R5FQL2W%.,600
M<CM18G5O;TLQ-G)6;&]KO&4S7"55)B)?/7X^"G%9<%$$H;EI@*6IR5FQL;7$Z
M:W4W<E9L;&=Q5C(I07([46,[;W,]7V-R5FQO;4T]6CAU7$<_+U1^/@IQ67!/
M6VY-,2LX<E9L;%)Q*G1&1’)6;&PO<48Z3W-R.UI%9W([469R9T!.<7-L,71‘
M,’X^"G%9<%!E;E)$4BR.U%C/’%.,61*<E9L;"AQ3C%C87([46-7<#5F.6IR
M.U%B=6\R;"0Q<"5E5SQ^/@IQ67!10&Y:8"E(<CM18V)Q5C(I2G)6;&Q;<EMR
M*39R.U%C8’‘Y6&A-<CM18S1O-S9Q*7!<1FD^?CX*<5EP4$9N33$J-’([46)L
M<48Z4$QR5FQK2W%&.D\Z<74_/V=R.U%C8&]‘3RHB<E8_2D1^/@IQ67!1-VY2
M1%0Z;2\V:V!!666Q05EAO.#@B<2)4<T]N8FE#95TH:#]F5CXQ(V@A5DMP;STY
M,FUH<2,^:GX^"G%9<%%4;EI@+"=P)BMG:6HX0%LT:F\L,EIR5C4N2G%U)$AO
M9"YJ<TM?65@V,B%67TA#2D@Z5FIQ(SYJ?CX*<5EP421N33$L,$IO(RQ90T%>
M4B5&;TO]/V]?.F]6:VMT2E-R<75C<EW)3$9S*G1^/@IQ/E5’,VY21%0Z<,$Y2
M1%%@<CM19F92+TXM26@^4C]#2$A4=C]‘,7A4V8]=4V88H1V1F#XK<DXQ)&$:6TY
M,,FUH<%TC87X^"G$^5@G;EI@*3ER+U%F/<&A9;#0P;U]N9&ET‘R06UE;2AC
M<5)123‘A4"YA=7)R0CIQ2D@Z5FIP72-A?CX*<3Y51EIN33$$I8G([469=.W5+
M,F1@<C5O.V!Q5R4H55QK*61R<6Q=<6M/9$,M9T$H1D)*+’X^"G$^54=D;C<I
M241R.U%O8E(D(4UU<CM18S5P;%!12W%U-EI6<6E#:49L35597F5A=7)%1#MB
M+69*+’X^"G$^54@_;C]$=59R.U%O<&A6/V]#<CM18U]P=%!L+7%U-EI?<6TV
```

```
M0S=N8FE#96DZ35(D3VQU2#Q*+'X^"G$^54=%;C%K(5ER.U%053MB)SY6<CM1
M8F%P9%D\:G%:)$5K<74V6FA06VHS(W$B1F`\?CX*<3Y52%-N-RE+.7`F*V=L
M;$)M7D9R.U%C(7!1-4DL<74V6DAR2R4F2&Q-55E?;VMJ02@A2S]>17,J='X^
M"G$^54AA;C]%(R9Q65X_<7!T:F9J<CM18UEP635C/G%U-EI6<DYL53EN8FE#
M9G!1/BTB(4X^8F-S*G1^/@IQ/E5(2FXQ:R,O;RD03&EH+#]H)G([46(^<$D^
M-'!Q6B1+;7%U-EUQ:#U=1"1J4V5C3THL?CX*<2,Z/D]N-RE(2W%U-F!?5$%"
M1T9P-6\_27%9<%U24W)F/"AQ=3992G`O:#])96%R9FM^/@IQ(SH_-6X_1'4O
M<74V8&UI.E%S27`];UHK<5EP75Y=<U`_/W%U-EEI<#0S-SQI.DAU(7X^"G$C
M.CXJ;C%J=4=Q=39@4C](,G!#<"XC*FAQ/EY(;G%9<%%4<#U+12IP)2\S-GX^
M"G$C.C\U;7!C/VAQ=3]=:W([6D=0(39'(3DA4VU*27)R041</3D\/S5I.C9H
M='X^"G$C.C]2;B0I;#QQ=3]=<'([6DA&(3I+6U\A53!#5W)R0CIU2DA#=$1K
M3TI3)GX^"G$C.C\B;6M/;&UQ=3]=9W([6D9;(3)+*6$A.6$N,"%O/"PY;F,K
M*WX^"G!<=#4[;7!C/VAP)D9=22%,5S!$<G)":S-*8V=T-6).94YQ2BQ^/@IP
M7'0V*FXD*6P\<"9&7C\A5"I60W)R0T9#5&!?-R]F7DHX+DHL?CX*<%QT-&5M
M:T]L;7`F1EQ4(44W:$=R<D118VE72CDN;T0W;E1*+'X^"G!<=#8];55(-F=P
M06%C22%.+%E#<RIT?CX*<%QT-E9M76-C.W!!860_(518+C!S*G1^/@IP7'0V
M+FU0-&-L<$%A8E0A1U\X6',J='X^"G!!62Q-;55(.6AP)5Q/9599(W5/5EE*
M9$-*+'X^"G!!62TS;5UC9CQQ63HG:FHW32LM:CA)53!*+'X^"G!!62PH;5`T
M9FU0*&`T8D-`:R%S0T!P+5A*+'X^"G!!62U0;3HM,$EI<5=.5FY>8D]94B].
M+4U2)R(S56Y>9%]J?CX*<$%9+5YM0DA=+6Y'*B)D<7-S*#=H66PT-&A7/2A0
M<7!T9'1^/@IP05DM1VTT;EU%9R5B4DUK9&TN*#MU2S)H.V<B2CUK9V]C87X^
M"G`F/B-I;3HM,#9:+"1":7X^"G`F/B1`;4)(72-F64E..WX^"G`F/B-.;31N
M72Q21$%I47X^"G`F/B=C1T\O9C]*0W%A/4HL?CX*<"8^)VA=0FQD)U\A2G(\
M2BQ^/@IP)CXG8#E>1DTX/4YN65Y*+'X^"F]@(G!.;#TP:E!C2VM+-6-"4#90
M<"1?;EYM*CXZ<G,J='X^"F]@(G!<;$5,0B]K3FDM3FM+,C,Y<5@]1F-P*%PR
M1',J='X^"F]@(G!%;#=R0D]>6RAN)EY.+#I*;R=C4UMJ:$Q"67,J='X^"F]$
M7&<F;"%J9#U6<DDW2"%O9RPE<BIO;EQO9"].9&TO-FMH;2M"<W%6:2,^)%HQ
M2B%'?CX*;T1<9TAL*C$\)64J=6\M(7$]6TYR,S9&0&%K-65#<"8K9W%P)$0E
M=&4F06,C9EYO+&Y^/@IO1%QF96MQ5SPV3C=M8B0A;FE.*W(E7$9816=A7&1J
M;R,L86II:W(Z3BM'-"]226=(+WX^"F\I05U=:T`T6#U4(D5907%U-F9C9EAM
M8"%K0#1,5F\I1C1^/@IO*4%>.&M(4#`E8TI`:EMQ=39F;&TK;W0A:TAO)"]O
M*48T?CX*;RE!73YK.R$P-DHC,4).<74V9EQB8EUE7VL[(218;RE&-'X^"F\I
M06%@2D$U4G!*1R9,-4HL?CX*;RE!865>/5LL5%\C(5@E2BQ^/@IO*4%A73U-
M2%1L/5)U)5]*+'X^"FYC)EA?2D%'7G)$/B%((4HL?CX*;F,F6&1><6TX5EME
M9D]O2BQ^/@IN8R987#U-6F!N-5`B0$5*+'X^"FY'8$]+1#A4:6%$.W0G8DHL
M?CX*;D=@3UE;7V\_3EMD8&5D2BQ^/@IN1V!/0C5*;S57-4T^42M*+'X^"FXL
M149*1#AF=6-$.W0D84HL?CX*;BQ%1EA;8"Q+4%MD8&)C2BQ^/@IN+$5&035+
M+$%9-4T^3BI*+'X^"FUF*CU<2D(I+B-*120B<THL?CX*;68J/6%><DY<7%\A
M<&1L2BQ^/@IM9BH]63U./"]T/5`\+4)*+'X^"FU*9#1;351+/R]R72I1R8],$HL
M?CX*;4ID-&!@43Y`8U\C(4AU2BQ^/@IM2F0T6$%=6EPN/5)T:UI*+'X^"FQI
M+7-?8D`Z3T1L:3)*?CX*;&DM=#9B2%8F;6QI,DI^/@IL:2US1&([)R=);&DR
M2GX^"FQ-9VXQ1#EL7&U$.&MF/THL?CX*;$UG;DI;83(R6EMC-E102BQ^/@IL
M36=N(C5,,BAC-4DP4U-*+'X^"FPR3&5.354^;S=-6#E5+4HL?CX*;#),95=@
M4C%P:V!6)DMQ2BQ^/@IL,DQE1T%>3C<V06!@,59*+'X^"FM0:U,E1#I.*W-$
M-V8A,DHL?CX*:U!K4T);86A68%MB7BU(2BQ^/@IK4&M29S5,:$QI-4=D44-*
M+'X^"FLU4$I45G$U1%I6=$LS4$HL?CX*:S502EEE*#II*F4L)C4L2BQ^/@IK
M-5!*44XW5&9C3CI1+3)*+'X^"FI3;SA)5"5D8%54)UE0/$HL?CX*:E-O.%)C
M2BQ4*6-,<#AU2BQ^/@IJ4V\X0DHH;%Y:2BHC8FM*+'X^"FER.2E(6B,O2#`A
M8SUK)6ER/4Y^/@II<CDI46980F)$(6LM4TYI<CU.?CX*:7(Y*4%2-E188R%>
M1%DJ:7(]3GX^"FAU/&`F4&E:0E509CM06THL?CX*:'4\8#]A;%1C+F%L.S18
M2BQ^/@IH=3Q?;$5O9B5716I&529*+'X^"F@^6U1-8T-4-TE`9G!Q2&-,<"-N
M2BQ^/@IH/EM44FM+7EXL66QH5F!K4$8D-THL?CX*:#Y;5$I>3TIR13$G.COB
M7EMG-%Q*+'X^"F<F1#9+9E=G6G5$/D!:1DOR3U)?8TMS-F%*+'X^"F<F1#90
M;2M"1R%;95LP,UM@+4%`:T]M3RY*+'X^"F<F1#9(8F%%2U\U4%LF/#5`;FPG
M7EI&+TM*+'X^"FOO6"UU9"]357X^"FOO6"XT9"]357X^"FOO6"UI9"]357X^
M"E-C/3-^/@I38STS?CX*4V,],WX^"E-C/3-^/@I38STS?CX*4V,],WX^"E-C
M/3-^/@I38STS?CX*4V,],WX^"E-C/3-^/@I38STS?CX*4V,],WX^"E-C/3-^
M/@I38STS?CX*4V,],WX^"B4E16YD1&%T8OIS:&]W<&%G9OHE)51R86EL97(*
*96YD"B4E14]&"@``
`
end
```

7.5 License

Copyright (c) 2011 Timothy Daly All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- Neither Timothy Daly nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Bibliography

[Knu84] Knuth, Donald E., “Literate Programming” Center for the Study of Language and Information ISBN 0-937073-81-4 Stanford CA (1992)

[Lam94] Lamport, Leslie, “Latex – A Document Preparation System”, Addison-Wesley, New York ISBN 0-201-52983-1

Index

- building, i
- Makefile, ii
- tangle.c, i