# The GnuDOS library

for version 1.8

**Mohammed Isam** (mohammed_isam1984@yahoo.com)

This manual is for the GnuDOS library (version 1.8).

# Table of Contents

# 1 Overview of the GnuDOS library

## 1.1 About the GnuDOS library

The GnuDOS package is a GNU software. It is a library designed to help new users of the GNU system, who are coming from a DOS background, fit into the picture and start using the GNU system with ease. It also addresses the console programmers of such programs that have the look and feel of the old DOS system. The library is composed of core utilities and software applications:

- The core library (corelib) contains five utilities: Kbd (for keyboard handling), UKbd (includes unicode support), Screen (for screen drawing), Dialogs (for dialog boxes/window drawing), and Strings (for extended strings functions).
- The software applications are three: Prime (console file manager), Mino (console text editor), and Fog (console form designer).

## 1.2 The rationale behind the GnuDOS corelib library

So, you like programming under the GNU/Linux console, right?. And you came from the DOS land where every thing was white/blue or yellow/black. You want to make users coming from the DOS land feel home when switching to the powerful GNU system. Okay, That's good. But there are some catches when programming under the console.
First of all, you can't format your output exactly the way you want in terms of color, positioning, and so on. You can go deep and use terminal escape sequences (as most GNU/Linux consoles emulate the VT100 terminal), but who can remember these?.

Next comes the problem of the terminal driver interfering with the keyboard input. You don't get the real key scancodes sent by the keyboard. The driver gets in the way and performs a lot of steps to map the right key to the right keycode, process some special key combinations (like CTRL+ALT+DEL) and so on, before passing the result to the terminal. And in the case of XTerminal, the X terminal does more processing before sending the final result to your program. You say what difference does it make? you are taking all the pain off my head, why should I bother? Here is why:

If you want your program to be REALLY interactive, like waiting the user to press a key ('press', not 'press and release' and then 'press ENTER'!) you can't rely on the good old getc() or getchar() functions, as they will return an input char alright, but only after the user presses ENTER!. That's no good for us, you know.
Another thing is reading special keys, like SHIFT, ALT and CTRL. You don't get scancodes for these keys (not all times, at least).

So how to make your program get over these problems? Well, you can implement your own keyboard driver, which will be very painful: to construct your keymap tables and do all the calculations; or you can interfere with the input sent from the console driver before it does any further processing on it. The console-utils See Chapter 4 [Kbd], page 6. utility does this. It tells the console driver to send it raw data (with no processing), and it then looks into its own table to see what key (or key combinations) does this scancode means, and then gives you the result.

Right now, the See Chapter 4 [Kbd], page 6. utility doesn't recognize ALL the possible keys that can be entered through a keyboard. It recognizes all the alphanumeric charset, the TAB, CAPS, ENTER, SPACE, CTRL, ALT, SHIFT, DEL, INS, HOME, ESC, and END. More keys (like function keys F1-F12) will be added with future releases.

The other thing the GnuDOS library provides is a utility for controlling the screen See Chapter 6 [Screen], page 12. It provides functions for getting the screen size (height and width), setting the screen colors, changing cursor position, and clearing the screen.

The third utility is the See Chapter 5 [Dialogs], page 9. utility, which (as its name says) provides a ready-to-use classes of dialog boxes under the console. It provides two types of boxes: simple dialog box (to provide the user with a messeage, or asking for confirmation, ...) and an input box (to ask the user to enter some input).

The fourth utility is the See Chapter 7 [Strings], page 14. utility. It provides some handy functions to make working with strings under C much easier for the programmer.

There are two sample programs: the See Chapter 2 [hello_gnudos], page 3, demonstrates how to use the various elements and utilities of the GnuDOS corelib library (except for the strings utility). The other example is See Chapter 3 [hello_strings], page 5, which demonstrates how to use the strings utility.

# 2 An example of using the GnuDOS library

This is a sample program that demonstrates how to use the GnuDOS library utilities:

```
#include "console/dialogs.h"
#include "console/screen.h"
#include "console/kbd.h"

void sighandler(int signo)
{
    //do what ever needs to be done here. The following line is just an example.
    fprintf(stderr, "SIGNAL %d received\n", signo);
}

int main(int argc, char *argv[])
{
  if(!catchSignals())
  {
    fprintf(stderr, "Error catching signals. Exiting.\n");
    exit(1);
  }
  if(!initTerminal())
  {
    fprintf(stderr, "Error initializing keyboard. Aborting.\n");
    exit(1);
  }

  getScreenSize(); //gets screen size
  clearScreenC(WHITE, BGBLACK); //clear the screen
  //loads color arrays with default values
  loadDefaultColors();
  setScreenColors(FG_COLOR[COLOR_WINDOW], BG_COLOR[COLOR_WINDOW]);

  msgBox("This was an example", OK, INFO);
  drawBox(2, 2, SCREEN_H-2, SCREEN_W-2, " Example ", YES);
  locate(3, 3); printf("Hello GnuDOS!");
  locate(4, 3); printf("This is an example Window.");
  locate(5, 3); printf("Press ENTER to exit...");
  while(1)
  {
    if(getKey() == ENTER_KEY) break;
  }

  clearScreen();
  //very important to restore keyboard state to its
  //previous state before exiting
  restoreTerminal();
  exit(0);
```

```
}
```

Note that including the header file "dialogs.h" automatically includes both "screen.h" and "kbd.h", as the dialogs utility uses both of the other two.

And now, REMEMBER two things:

1. a call to initTerminal() must be invoked before using the library
2. a call restoreTerminal() must be done before exiting the program

For deatils about these functions please see See Chapter 4 [Kbd], page 6.

If you forget point (2), you will leave the user's terminal in raw mode, which (under console) means he/she will not be able to do virtually anything (not even switching terminal by CTRL+ALT+F key!). The only way out is a reboot!. And I am talking about hard reboot by pressing the power button or restart key. Under X it is less worse, usually the user will need to close the xterm or kill the process. Still though, it is IMPERATIVE to call restoreTerminal() before exiting your program!.
To make sure no funny things happen (like your progrm crashing for whatever reason, or your admin killing it, to name a few) before you call restoreTerminal(), you better use the catchSignals() function of the See Chapter 5 [Dialogs], page 9, utility. Remember though that there are some signals that can't be caught by your program, like the SIGSTOP and SIGKILL signals. This is why we used the catchSignals() function instead of the catchAllSignals() function.

# 3 An example of using the strings utility

This is a sample program that demonstrates how to use the strings utility:

```
#include <stdio.h>
#include "console/strings.h"

int main(int argc, char **argv)
{
    printf("Hello World");
    str s;
    s = "Hello world";
    printf("\n%s", s);
    printf("\n%d", indexof(s, 'H'));
    printf("\n%d", nindexof('H'));
    printf("\n%d", lindexof(s, 'H'));
    printf("\n%s", substr(s, 4));
    printf("\n%s", nsubstr(s, 4, 5));
    return 0;
}
```

# 4 Using the Kbd utility

The Kbd utility of the GnuDOS library provides functions for getting input from the keyboard, initializing and restoring the terminal state to enable the utility to grasp proper keyboard input, and some global variables.

The global variables defined in kbd.h are:

```
bool ALT;
bool CTRL;
bool SHIFT;
bool CAPS;
bool INSERT;
bool X_IS_RUNNING;
```

This is their explanation:

- ALT: Boolean variable that indicates the state of the ALT key
  (1=pressed, 0=released)

- CTRL: Boolean variable that indicates the state of the CTRL key
  (1=pressed, 0=released)

- SHIFT: Boolean variable that indicates the state of the SHIFT key
  (1=pressed, 0=released)

- CAPS: Boolean variable that indicates the state of CAPSLOCK
  (1=pressed/ON, 0=released/OFF)

- INSERT: Boolean variable that indicates the state of the INSERT key
  (1=pressed/ON, 0=released/OFF)

- X_IS_RUNNING: Boolean variable that indicates whether X is running
  (1=running under X, 0=running under console)

Three functions are defined:

```
int initTerminal();
void restoreTerminal();
int getKey();
```

The initTerminal() function must be called before any other library function is used. It initializes the terminal for library use. What this means in simple English is that the console will be messed up for other programs during your program execution. This is why it is MANDATORY to call restoreTerminal() just before your program exits to ensure that the terminal is restored to its previous state. Failing to do so, the terminal is left in an intermediate state that the user will have only one option: to reboot (under console) or to kill (or close) the terminal (under X).

The function getKey() is called to get the next key press from the keyboard. It actually relies on two functions internally: one to get the key under X, the other to get it under console mode. The difference between the two is of no relevance to the user. Just call getKey() to get the next keypress whether under X or the console.

The getKey() function returns its result as an integer. For alphanumeric keys this will mean the ASCII value of that key (ASCII 65-90 for Latin capitals, 97-122 for Latin smalls, 32 for Space, 33-64 for numbers and punctuation, 96 for backtick, 123-126 for braces, vertical bar and tilde). Other keys like arrows and ESC and ENTER are defined as macros in the kbd.h file:

```
#define ESC_KEY  27
#define BACKSPACE_KEY 8
#define TAB_KEY  9
#define ENTER_KEY 13
#define CAPS_KEY 1
#define SHIFT_KEY 2
#define CTRL_KEY 3
#define ALT_KEY  4
#define SPACE_KEY 32
#define UP_KEY   5
#define DOWN_KEY 6
#define LEFT_KEY 7
#define RIGHT_KEY 10
#define DEL_KEY  11
#define HOME_KEY 12
#define END_KEY  14
#define INS_KEY  15
#define SHIFT_DOWN 17
#define SHIFT_UP 18
#define PGUP_KEY 19
#define PGDOWN_KEY 20
```

What you need to do is to match the return value of getKey() against the desired key. For example:

```
if(getKey() == ESC_KEY)
{
   exit(0);
}
```

Or, more elegantly, in a switch loop:

```
int c = getKey();
switch(c)
{
  case(ESC_KEY):
     //do-something
     break;
```

```
    case(UP_KEY):
        //do-other-stuff
        break;
    default:
        if(c >= 32 && c <= 126)
            print("%c", c);
        break;
}
```

To test for special key combinations (e.g. CTRL+S):

```
c = getKey()
if(c == 's' && CTRL)
{
    //do something
    ...
}
```

Another utility has been added, which is called UKbd ("U" stands for Unicode). As such, this utility is the exact same replica of the Kbd utility, with the exception that it handles unicode characters. The functions defined are almost the same as Kbd's functions, with an added "u" in front of each, i.e.:

```
char *ugetKey();
char *ugetKeyUnderConsole();
char *ugetKeyUnderX();
```

The results are returned as a character pointer in each.

One additional piece of information is the mask that is used to determine the length of a given unicode char, as unicode chars have variable lengths:

```
static unsigned short mask[] = {192, 224, 240};
```

# 5 Using the Dialogs utility

The Dialogs utility provides three types of dialog boxes: simple dialog boxes, input boxes, and empty boxes.

## 5.1 Simple Dialog Box

The function to draw a simple dialog box is defined in "dialogs.h" as:

```
int msgBox(char *msg, int buttons, msgtype tmsg);
```

Where:

- msg: is a pointer to the string that will be the output message of the dialog box
- buttons: an integer value defining the number and type of buttons to be displayed (see below)
- tmsg: a value of type "msgtype" (see below) defining the type of dialog box. This will be the title of the dialog

The value of `buttons` can be: OK, OK|CANCEL, YES|NO, or OK|CANCEL|ALL. Note when using two or more buttons they need to be ORed with the vertical bar. The macros defining those buttons are declared in "dialogs.h" as:

```
//buttons used in message boxes//
#define OK 1       //00000001
#define YES 2       //00000010
#define CANCEL 4       //00000100
#define NO 8       //00001000
#define ALL 16      //00010000
#define ABORT 32      //00100000
```

The value of `tsmg` can be:

- INFO: This is an information box. The title will be "INFORMATION"
- ERROR: This is an error message box. The title will be "ERROR"
- CONFIRM: This is a confirmation dialog box. The title will be "CONFIRMATION"

## 5.2 Input boxes

The function to draw a simple dialog box is defined in "dialogs.h" as:

```
char* inputBox(char *msg, char *title);
```

Where:

- msg: is a pointer to the string that will be the output message of the dialog box
- title: is a pointer to the string that will be the title of the input box

The function returns the user input as a char pointer. If the user entered nothing, or pressed CANCEL button or ESC, the function returns NULL. You can also access the return value in the globally accessed variable 'input', which is defined:

```
char input[MAX_INPUT_MSG_LEN+1]; //input string returned by inputBox() function
```

Another function for drawing input boxes is defined:

```
char* inputBoxI(char *msg, char *inputValue, char *title);
```

The only difference is that it takes as the second parameter a string that will be displayed in the input box as an initial input value for the user. This is helpful if you want to give the user a default value for whatever input is required from the user. The user can change the input or just press ENTER and accept the default value.

## 5.3 Empty boxes

Drawing empty boxes or windows is done via one of two functions:

```
void drawBox(int x1, int y1, int x2, int y2, char *title, int clearArea);
void drawBoxP(point p1, point p2, char *title, int clearArea);
```

They basically do the same thing, except that drawBoxP() accepts the window coordinates as two 'point' structures which are defined as:

```
typedef struct { int row; int col; } point;
```

Whereas the drawBox() function accepts coordinates as four integer values. The explanation of the parameters to the two functions is as follows:

- x1: The x-coordinate (row) of the upper left corner
- y1: The y-coordinate (column) of the upper left corner
- x2: The x-coordinate (row) of the lower right corner
- y2: The y-coordinate (column) of the lower right corner
- char *title: A string pointer to the title of the dialog box
- int clearArea: A boolean value indicating whether to clear the box area (YES=clear, NO=don't clear). Not clearing the box area can be handy when, for example, you need to redraw the window frame but leave the window contents intact.

Other things of concern are:

```
int MAX_MSG_BOX_W;
int MAX_MSG_BOX_H;
#define MAX_INPUT_MSG_LEN 100
```

The first two are global variables used to determine the maximum size of a dialog box. MAX_MSG_BOX_W defines the maximum width (columns) and MAX_MSG_BOX_H the maxium height (rows). Their values are calculated in the msgBox() and inputBox() functions as:

```
MAX_MSG_BOX_W = SCREEN_W-2;
MAX_MSG_BOX_H = SCREEN_H-2;
```

The last one, MAX_INPUT_MSG_LEN is a macro defining the maximum length of the input string returned by an input box. Currently it is restricted to 100 chars.

## 5.4 The catchSignals() function

The last two functions of "dialogs.h" are:

```
int catchSignals();
int catchAllSignals();
```

Which are handy and so important. Remember that after a call to initTerminal() the terminal will be in an intermediate state, which is not of much use to the user. Calling restoreTerminal() is an important step to do before leaving your program. But what if your program crashed for whatever reason? (bad things happen all the time), or if a system administrator decided to kill your process?. Here is what catchSignals() does: it catches all the important signals (namely: SIGINT, SIGQUIT, SIGABRT, and SIGTERM) and passes them to a signal handler, which you will define as:

```
void sighandler(int signo)
{
    //do what ever needs to be done here. The following line is just an example.
    fprintf(stderr, "SIGNAL %d received\n", signo);
}
```

The catchAllSignals() does the same, except it tries to catch also SIGSTP, SIGKILL, and SIGSTOP. It is a futile effort of course, as these signals can't be caught, it is just included for convenience.
If either function succeeds in catching the signals, it will return 1. Otherwise, 0. Expect catchAllSignals() to return 0 at all times because of the reason above.
Note that you will need to define the signal handler even if you will not use the catchSignals() function (which is, by the way, not recommended at all! We explained the reasons several times above). It can be defined as an empty function as:

```
void sighandler(int signo)
{

}
```

Again, please define the signal handler in a proper way whenever possible.

# 6 Using the Screen utility

The screen utility provides functions to manipulate the screen colors, clearing the screen, and positioning of the cursor. It also defines values for the screen size. The member variables of the screen utility (defined in "screen.h") are:

```
int SCREEN_W;
int SCREEN_H;
```

Both these variables are filled with their proper values after a call to getScreenSize().

```
int FG_COLOR[color_components];
int BG_COLOR[color_components];
```

The color_components is a macro defined with a value of 6. The possible values for color_components which is an index into arrays of colors determining what color is assigned to which component (i.e., dialogs, buttons, ...) are:

```
COLOR_WINDOW    0
COLOR_HIGHLIGHT_TEXT 1
COLOR_MENU_BAR  2
COLOR_STATUS_BAR 3
COLOR_BUTTONS   4
COLOR_HBUTTONS  5
```

You can define the colors in the color arrays by using integer values, although using macro names (as discussed below) is recommended. Initializing the arrays can be done with code like:

```
 FG_COLOR[COLOR_WINDOW] = 37;
 FG_COLOR[COLOR_HIGHLIGHT_TEXT] = 34;
 FG_COLOR[COLOR_MENU_BAR] = 34;
 FG_COLOR[COLOR_STATUS_BAR] = 34;
 FG_COLOR[COLOR_BUTTONS] = 37;
 FG_COLOR[COLOR_HBUTTONS] = 32;
 BG_COLOR[COLOR_WINDOW] = 44;
 BG_COLOR[COLOR_HIGHLIGHT_TEXT] = 47;
 BG_COLOR[COLOR_MENU_BAR] = 47;
 BG_COLOR[COLOR_STATUS_BAR] = 47;
 BG_COLOR[COLOR_BUTTONS] = 41;
 BG_COLOR[COLOR_HBUTTONS] = 41;
```

For convenience, the names of colors used in screen utility functions can be retrieved from the array screen_colors[] after a call to getScreenColors():

```
getScreenColors();
for(int i = 0; i < 16; i++)
  printf("%s\n", screen_colors[i]);
```

To set the screen colors (e.g. before clearing the screen,), use the function:

```
void setScreenColors(int FG, int BG);
```

where FG is the foreground color, BG is the background color. Color values are defined as macros in the (screen.h) file:

```
#define BLACK      30      //set black foreground
#define RED        31      //set red foreground
#define GREEN      32      //set green foreground
#define BROWN      33      //set brown foreground
#define BLUE       34      //set blue foreground
#define MAGENTA    35      //set magenta foreground
#define CYAN       36      //set cyan foreground
#define WHITE      37      //set white foreground
#define BGBLACK    40      //set black background
#define BGRED      41      //set red background
#define BGGREEN    42      //set green background
#define BGBROWN    43      //set brown background
#define BGBLUE     44      //set blue background
#define BGMAGENTA  45      //set magenta background
#define BGCYAN     46      //set cyan background
#define BGWHITE    47      //set white background
#define BGDEFAULT  49      //set default background color
```

To get the size of screen coordinates, use function:

```
void getScreenSize();
```

which will fill the values into SCREEN_W and SCREEN_H global variables.
The functions

```
void clearScreen();
void clearScreenC(int FG, int BG);
```

basically do the same thing, except clearScreen() uses whatever colors where passed into previous call of setScreenColors(), and clearScreenC() takes the values of colors to use when clearing the screen. Last color function is

```
void loadDefaultColors();
```

which resets the color arrays into default values.
To reposition the cursor, use:

```
void locate(int row, int col);
```

giving the row and column as int values. Remember the screen has top-left based coordinates, meaning position 1-1 is at the top-left corner, position 25-80 is at the bottom-right (for a 25x80 screen size).

To show/hide the cursor:

```
/* Turn on the cursor */
void showCursor();
/* Turn off the cursor */
void hideCursor();
```

# 7 Using the Strings utility

The strings utility defines some handy functions for dealing with strings. Strings in C are problematic: they involve a lot of pointer manipulation which is often complicated, error-prone and a source of bugs. The strings utility defines a wrapper type for strings (only for convenience), which is defined as:

```
typedef char *str;
```

The functions of the strings utility, as defined in "strings.h", are:

```
int indexof(str string, char chr);
int nindexof(char chr);
int lindexof(str string, char chr);

str substr(str string, int start);
str nsubstr(str string, int start, int length);
str ltrim(str string);
str rtrim(str string);
str trim(str string);

str toupper(str string);
str tolower(str string);
```

What the functions do is as following:

- The indexof() function returns the zero-based index of the first occurence of 'chr' in 'string'.
- The nindexof() function returns the zero-based index of the next occurence of 'chr' in 'string'. It should be called after a previous call the indexof().
- The lindexof() function returns the zero-based index of the last occurence of 'chr' in 'string'. If there is only one occurence of 'chr' in 'string', the return value is essentially the same as that of indexof().
- The substr() function returns a substring of 'string' starting from position 'start'. Note start is zero-based.
- The nsubstr() function returns a substring of 'string' starting from position 'start' and spanning 'length' characters. Note start is zero-based.
- The ltrim() function trims (removes) all the whitespace characters from the strings' left side. Whitespace characters removed are: space, tab, and newline. If there are no whitespace characters in the lefthand side of the string, the original string is returned.
- The rtrim() function trims (removes) all the whitespace characters from the strings' right side. Whitespace characters removed are: space, tab, and newline. If there are no whitespace characters in the lefthand side of the string, the original string is returned.
- The trim() function trims (removes) all the whitespace characters from both strings' ends. Whitespace characters removed are: space, tab, and newline. If there are no whitespace characters in either side of the string, the original string is returned.
- The toupper() function returns the string in upper case letters.

- The tolower() function returns the string in lower case letters.

# 8 The ASCII character table

ASCII stands for 'American Standard Code for Information Interchange'. It's a 7-bit character code used to be the standard of text representation. Although there are a number of other standards in use today, especially those that support wide characters and multi-language interfaces, the ASCII character set is the base for most of the character encodings used today.

The first 32 (0-31) characters in the ASCII-table are unprintable control codes that are classicaly used to control peripheral devices such as printers.

Codes 32-127 are the printable characters, which represent letters, digits, punctuation marks, and a few other symbols. Almost every character is available on standard keyboards. Character 127 represents the command DEL.

| **Dec** | **Oct** | **Hex** | **Name** | **Description** |
|---|---|---|---|---|
| 0 | 000 | 00 | NUL | Null char |
| 1 | 001 | 01 | SOH | Start of heading |
| 2 | 002 | 02 | STX | Start of text |
| 3 | 003 | 03 | ETX | End of text |
| 4 | 004 | 04 | EOT | End of transmission |
| 5 | 005 | 05 | ENQ | Enquiry |
| 6 | 006 | 06 | ACK | Acknowledgment |
| 7 | 007 | 07 | BEL | Bell |
| 8 | 010 | 08 | BS | Back space |
| 9 | 011 | 09 | HT | Horizontal tab |
| 10 | 012 | 0A | LF | Line feed |
| 11 | 013 | 0B | VT | Vertical tab |
| 12 | 014 | 0C | FF | Form feed |
| 13 | 015 | 0D | CR | Carriage return |
| 14 | 016 | 0E | SO | Shift out/XOn |
| 15 | 017 | 0F | SI | Shift in/XOff |
| 16 | 020 | 10 | DLE | Data line escape |
| 17 | 021 | 11 | DC1 | Device control 1 |
| 18 | 022 | 12 | DC2 | Device control 2 |
| 19 | 023 | 13 | DC3 | Device control 3 |
| 20 | 024 | 14 | DC4 | Device control 4 |
| 21 | 025 | 15 | NAK | Negative acknowledgment |
| 22 | 026 | 16 | SYN | Synchronous idle |
| 23 | 027 | 17 | ETB | End of transmit block |
| 24 | 030 | 18 | CAN | Cancel |
| 25 | 031 | 19 | EM | End of medium |
| 26 | 032 | 1A | SUB | Substitute |
| 27 | 033 | 1B | ESC | Escape |
| 28 | 034 | 1C | FS | File separator |
| 29 | 035 | 1D | GS | Group separator |

| | | | | |
|---|---|---|---|---|
| 30 | 036 | 1E | RS | Record separator |
| 31 | 037 | 1F | US | Unit separator |
| 32 | 040 | 20 | SP | Space |
| 33 | 041 | 21 | ! | Exclamation mark |
| 34 | 042 | 22 | " | Double quotes |
| 35 | 043 | 23 | # | Number |
| 36 | 044 | 24 | $ | Dollar |
| 37 | 045 | 25 | % | Procenttecken |
| 38 | 046 | 26 | & | Ampersand |
| 39 | 047 | 27 | ’ | Single quote |
| 40 | 050 | 28 | ( | Open parenthesis |
| 41 | 051 | 29 | ) | Close parenthesis |
| 42 | 052 | 2A | * | Asterisk |
| 43 | 053 | 2B | + | Plus |
| 44 | 054 | 2C | , | Comma |
| 45 | 055 | 2D | - | Hyphen |
| 46 | 056 | 2E | . | Full stop |
| 47 | 057 | 2F | / | Slash or divide |
| 48 | 060 | 30 | 0 | Zero |
| 49 | 061 | 31 | 1 | One |
| 50 | 062 | 32 | 2 | Two |
| 51 | 063 | 33 | 3 | Three |
| 52 | 064 | 34 | 4 | Four |
| 53 | 065 | 35 | 5 | Five |
| 54 | 066 | 36 | 6 | Six |
| 55 | 067 | 37 | 7 | Seven |
| 56 | 070 | 38 | 8 | Eight |
| 57 | 071 | 39 | 9 | Nine |
| 58 | 072 | 3A | : | Colon |
| 59 | 073 | 3B | ; | Semicolon |
| 60 | 074 | 3C | < | Less than |
| 61 | 075 | 3D | = | Equals |
| 62 | 076 | 3E | > | Grater than |
| 63 | 077 | 3F | ? | Question mark |
| 64 | 100 | 40 | @ | At symbol |
| 65 | 101 | 41 | A | Capital A |
| 66 | 102 | 42 | B | Capital B |
| 67 | 103 | 43 | C | Capital C |
| 68 | 104 | 44 | D | Capital D |
| 69 | 105 | 45 | E | Capital E |
| 70 | 106 | 46 | F | Capital F |
| 71 | 107 | 47 | G | Capital G |
| 72 | 110 | 48 | H | Capital H |
| 73 | 111 | 49 | I | Capital I |
| 74 | 112 | 4A | J | Capital J |
| 75 | 113 | 4B | K | Capital K |
| 76 | 114 | 4C | L | Capital L |

| | | | | |
|---|---|---|---|---|
| 77 | 115 | 4D | M | Capital M |
| 78 | 116 | 4E | N | Capital N |
| 79 | 117 | 4F | O | Capital O |
| 80 | 120 | 50 | P | Capital P |
| 81 | 121 | 51 | Q | Capital Q |
| 82 | 122 | 52 | R | Capital R |
| 83 | 123 | 53 | S | Capital S |
| 84 | 124 | 54 | T | Capital T |
| 85 | 125 | 55 | U | Capital U |
| 86 | 126 | 56 | V | Capital V |
| 87 | 127 | 57 | W | Capital W |
| 88 | 130 | 58 | X | Capital X |
| 89 | 131 | 59 | Y | Capital Y |
| 90 | 132 | 5A | Z | Capital Z |
| 91 | 133 | 5B | [ | Opening bracket |
| 92 | 134 | 5C | \ | Backslash |
| 93 | 135 | 5D | ] | Closing bracket |
| 94 | 136 | 5E | ^ | Caret |
| 95 | 137 | 5F | _ | Underscore |
| 96 | 140 | 60 | ‘ | Grave accent |
| 97 | 141 | 61 | a | Small a |
| 98 | 142 | 62 | b | Small b |
| 99 | 143 | 63 | c | Small c |
| 100 | 144 | 64 | d | Small d |
| 101 | 145 | 65 | e | Small e |
| 102 | 146 | 66 | f | Small f |
| 103 | 147 | 67 | g | Small g |
| 104 | 150 | 68 | h | Small h |
| 105 | 151 | 69 | i | Small i |
| 106 | 152 | 6A | j | Small j |
| 107 | 153 | 6B | k | Small k |
| 108 | 154 | 6C | l | Small l |
| 109 | 155 | 6D | m | Small m |
| 110 | 156 | 6E | n | Small n |
| 111 | 158 | 6F | o | Small o |
| 112 | 160 | 70 | p | Small p |
| 113 | 161 | 71 | q | Small q |
| 114 | 162 | 72 | r | Small r |
| 115 | 163 | 73 | s | Small s |
| 116 | 164 | 74 | t | Small t |
| 117 | 165 | 75 | u | Small u |
| 118 | 166 | 76 | v | Small v |
| 119 | 167 | 77 | w | Small w |
| 120 | 170 | 78 | x | Small x |
| 121 | 171 | 79 | y | Small y |
| 122 | 172 | 7A | z | Small z |
| 123 | 173 | 7B | { | Opening brace |

| | | | | |
|---|---|---|---|---|
| 124 | 174 | 7C | \| | Vertical bar |
| 125 | 175 | 7D | } | Closing brace |
| 126 | 176 | 7E | ~ | Tilde |
| 127 | 177 | 7F | DEL | Delete |

# Appendix A GNU Free Documentation License

## A.1 GNU Free Documentation License



0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

   This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

   We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

   This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

   A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

   A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

   The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

   In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

## ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

# Index