GNU Libidn

for version 0.1.9, 20 February 2003

Simon Josefsson (bug-libidn@gnu.org)

This manual is for GNU Libidn (version 0.1.9, 20 February 2003), which is a library for internationalized string processing.

Table of Contents

1 Introduction **1**
- 1.1 Getting Started 1
- 1.2 Features 1
- 1.3 Supported Platforms 2
- 1.4 Bug Reports 3

2 Preparation **4**
- 2.1 Header 4
- 2.2 Initialization 5
- 2.3 Version Check 5
- 2.4 Building the source 6

3 Stringprep Functions **7**

4 Punycode Functions **11**

5 IDNA Functions **12**

6 Examples **16**
- 6.1 Example 1 16
- 6.2 Example 2 18
- 6.3 Example 3 23
- 6.4 Example 4 24

7 Acknowledgements **27**

Appendix A GNU LESSER GENERAL PUBLIC LICENSE **28**
- A.0.1 Preamble 28
- A.0.2 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION 29
- A.0.3 How to Apply These Terms to Your New Libraries 36

Appendix B Copying This Manual **37**
- B.1 GNU Free Documentation License 37
  - B.1.1 ADDENDUM: How to use this License for your documents 43

Index **44**

1 Introduction

GNU Libidn is an implementation of the Stringprep, Punycode and IDNA specifications defined by the IETF Internationalized Domain Names (IDN) working group, used for internationalized domain names. It is available under the GNU Lesser General Public License. Currently the Nameprep, Kerberos 5, SASL and XMPP Stringprep profiles are supported.

The library contains a generic Stringprep implementation (including Unicode 3.2 NFKC normalization, table mapping of characters, and Bidirectional Character handling), a few Stringprep profiles, and an implementation of the functionality defined by Punycode and IDNA.

The Stringprep API consists of two main functions, one for converting data from the system's native representation into UTF-8, and one function to perform the Stringprep processing. Each stringprep profile has a corresponding CPP macro. Adding a new Stringprep profile for your application within the API is straightforward. The Punycode API consists of one encoding function and one decoding function. The IDNA API consists of the ToASCII and ToUnicode functions, as well as an high-level interface for converting entire domain names to and from the ACE encoded form.

The library is used by forthcoming network applications to process user names and passwords before they are input to cryptographic operations. Libidn can be built into GNU Libc to enable a new getaddrinfo() flag for system-wide IDN processing.

GNU Libidn is developed for the GNU/Linux system, but runs on over 20 Unix platforms (including Solaris, IRIX, AIX, and Tru64) and Windows.

1.1 Getting Started

This manual documents the library programming interface. All functions and data types provided by the library are explained.

The reader is assumed to possess basic familiarity with internationalization concepts and network programming in C or C++.

This manual can be used in several ways. If read from the beginning to the end, it gives a good introduction into the library and how it can be used in an application. Forward references are included where necessary. Later on, the manual can be used as a reference manual to get just the information needed about any particular interface of the library. Experienced programmers might want to start looking at the examples at the end of the manual (see Chapter 6 [Examples], page 16), and then only read up those parts of the interface which are unclear.

1.2 Features

This library might have a couple of advantages over other libraries doing a similar job.

It's Free Software
: Anybody can use, modify, and redistribute it under the terms of the GNU Lesser General Public License.

It's thread-safe
    No global state is kept in the library.

It's portable
    It should work on all Unix like operating systems, including Windows.

1.3 Supported Platforms

Libidn has at some point in time been tested on the following platforms.

1. Debian GNU/Linux 3.0 (Woody)

   GCC 2.95.4 and GNU Make. This is the main development platform. `alphaev67-unknown-linux-gnu`, `alphaev6-unknown-linux-gnu`, `hppa64-unknown-linux-gnu`, `i686-pc-linux-gnu`, `ia64-unknown-linux-gnu`.
2. Tru64 UNIX

   Tru64 UNIX C compiler and Tru64 Make. `alphaev67-dec-osf5.1`, `alphaev68-dec-osf5.1`.
3. SuSE Linux 7.1

   GCC 2.96 and GNU Make. `alphaev6-unknown-linux-gnu`, `alphaev67-unknown-linux-gnu`.
4. SuSE Linux 7.2a

   GCC 3.0 and GNU Make. `ia64-unknown-linux-gnu`.
5. RedHat Linux 7.2

   GCC 2.96 and GNU Make. `alphaev6-unknown-linux-gnu`, `alphaev67-unknown-linux-gnu`, `ia64-unknown-linux-gnu`.
6. RedHat Linux 8.0

   GCC 3.2 and GNU Make. `i686-pc-linux-gnu`.
7. RedHat Advanced Server 2.1

   GCC 2.96 and GNU Make. `i686-pc-linux-gnu`.
8. Slackware Linux 8.0.01

   GCC 2.95.3 and GNU Make. `i686-pc-linux-gnu`.
9. Mandrake Linux 9.0

   GCC 3.2 and GNU Make. `i686-pc-linux-gnu`.
10. IRIX 6.5

    MIPS C compiler, IRIX Make. `mips-sgi-irix6.5`.
11. AIX 4.3.2

    IBM C for AIX compiler, AIX Make. `rs6000-ibm-aix4.3.2.0`.
12. Microsoft Windows 2000 (Cygwin)

    GCC 3.2, GNU make. `i686-pc-cygwin`.
13. HP-UX 11

    HP-UX C compiler and HP Make. `ia64-hp-hpux11.22`, `hppa2.0w-hp-hpux11.11`.

14. SUN Solaris 2.8

    Sun WorkShop Compiler C 6.0 and SUN Make. `sparc-sun-solaris2.8`.
15. NetBSD 1.6

    GCC 2.95.3 and GNU Make. `alpha-unknown-netbsd1.6`, `i386-unknown-netbsdelf1.6`.
16. OpenBSD 3.1 and 3.2

    GCC 2.95.3 and GNU Make. `alpha-unknown-openbsd3.1`, `i386-unknown-openbsd3.1`.
17. FreeBSD 4.7

    GCC 2.95.4 and GNU Make. `alpha-unknown-freebsd4.7`, `i386-unknown-freebsd4.7`.

If you use Libidn on, or port Libidn to, a new platform please report it to the author.

1.4 Bug Reports

If you think you have found a bug in Libidn, please investigate it and report it.

- Please make sure that the bug is really in Libidn, and preferably also check that it hasn't already been fixed in the latest version.
- You have to send us a test case that makes it possible for us to reproduce the bug.
- You also have to explain what is wrong; if you get a crash, or if the results printed are not good and in that case, in what way. Make sure that the bug report includes all information you would need to fix this kind of bug for someone else.

Please make an effort to produce a self-contained report, with something definite that can be tested or debugged. Vague queries or piecemeal messages are difficult to act on and don't help the development effort.

If your bug report is good, we will do our best to help you to get a corrected version of the software; if the bug report is poor, we won't do anything about it (apart from asking you to send better bug reports).

If you think something in this manual is unclear, or downright incorrect, or if the language needs to be improved, please also send a note.

Send your bug report to:

‘`bug-libidn@gnu.org`’

2 Preparation

To use 'Libidn', you have to perform some changes to your sources and the build system. The necessary changes are small and explained in the following sections. At the end of this chapter, it is described how the library is initialized, and how the requirements of the library are verified.

A faster way to find out how to adapt your application for use with 'Libidn' may be to look at the examples at the end of this manual (see Chapter 6 [Examples], page 16).

2.1 Header

The library contains a few independent parts, and each part export the interfaces (data types and functions) in a header file. You must include the appropriate header files in all programs using the library, either directly or through some other header file, like this:

```
#include <stringprep.h>
```

The header files and the functions they define are categorized as follows:

stringprep.h
: The low-level stringprep API entry point. Normal applications uses one specific stringprep profile, and should rather include the corresponding profile header file (see below). If you are writing an application that only makes use of the utility functions, including this header file may be more appropriate however.

  The name space of the stringprep part of Libidn is `stringprep*` for function names, `Stringprep*` for data types and `STRINGPREP_*` for other symbols. In addition the same name prefixes with one prepended underscore are reserved for internal use and should never be used by an application.

stringprep_generic.h
: The entry point to the generic tables specified in the stringprep specification. It is normally only needed by applications that want to define its own stringprep profile, based on the generic tables.

  This header file uses the same namespace as the main stringprep.h header file.

stringprep_kerberos5.h
: The entry point to the experimental Kerberos 5 profile of stringprep.

  This header file uses the same namespace as the main stringprep.h header file.

stringprep_nameprep.h
: The entry point to the IDN ("nameprep") profile of stringprep. This is the entry point used by applications needing low-level access to the stringprep profile used in IDN. Most applications requesting IDN functionality will want idna.h instead though.

  This header file uses the same namespace as the main stringprep.h header file.

stringprep_plain.h
: The entry point to the SASL ANONYMOUS ("plain") profile of stringprep.

  This header file uses the same namespace as the main stringprep.h header file.

stringprep_xmpp.h
: The entry point to the experimental XMPP node ("nodeprep") and resource identifier ("resourceprep") profiles of stringprep.

  This header file uses the same namespace as the main stringprep.h header file.

punycode.h
: The entry point to Punycode encoding and decoding functions. Normally punycode is used via the idna.h interface, but some application may want to perform raw punycode operations.

  The name space of the punycode part of Libidn is `punycode_*` for function names, `Punycode*` for data types and `PUNYCODE_*` for other symbols. In addition the same name prefixes with one prepended underscore are reserved for internal use and should never be used by an application.

idna.h
: The entry point to the IDNA functions. This is the normal entry point for applications that need IDN functionality.

  The name space of the IDNA part of Libidn is `idna_*` for function names, `Idna*` for data types and `IDNA_*` for other symbols. In addition the same name prefixes with one prepended underscore are reserved for internal use and should never be used by an application.

2.2 Initialization

Libidn is stateless and does not need any initialization.

2.3 Version Check

It is often desirable to check that the version of 'Libidn' used is indeed one which fits all requirements. Even with binary compatibility new features may have been introduced but due to problem with the dynamic linker an old version is actually used. So you may want to check that the version is okay right after program startup.

`const char *` **stringprep_check_version** (`const char * req_version`) [Function]

req_version: Required version number, or NULL.

Check that the the version of the library is at minimum the requested one and return the version string; return NULL if the condition is not satisfied. If a NULL is passed to this function, no check is done, but the version string is simply returned.

See *STRINGPREP_VERSION* for a suitable `req_version` string.

Version string of run-time library, or NULL if the run-time library does not meet the required version number.

The normal way to use the function is to put something similar to the following first in your `main()`:

```
if (!stringprep_check_version (STRINGPREP_VERSION))
  {
    printf ("stringprep_check_version() failed:\n"
            "Header file incompatible with shared library.\n");
    exit(1);
  }
```

2.4 Building the source

If you want to compile a source file including e.g. the 'idna.h' header file, you must make sure that the compiler can find it in the directory hierarchy. This is accomplished by adding the path to the directory in which the header file is located to the compilers include file search path (via the '-I' option).

However, the path to the include file is determined at the time the source is configured. To solve this problem, 'Libidn' uses the external package `pkg-config` that knows the path to the include file and other configuration options. The options that need to be added to the compiler invocation at compile time are output by the '--cflags' option to `pkg-config libidn`. The following example shows how it can be used at the command line:

```
gcc -c foo.c `pkg-config libidn --cflags`
```

Adding the output of '`pkg-config libidn --cflags`' to the compilers command line will ensure that the compiler can find e.g. the idna.h header file.

A similar problem occurs when linking the program with the library. Again, the compiler has to find the library files. For this to work, the path to the library files has to be added to the library search path (via the '-L' option). For this, the option '--libs' to `pkg-config libidn` can be used. For convenience, this option also outputs all other options that are required to link the program with the 'libidn' libarary. The example shows how to link 'foo.o' with the 'libidn' library to a program `foo`.

```
gcc -o foo foo.o `pkg-config libidn --libs`
```

Of course you can also combine both examples to a single command by specifying both options to `pkg-config`:

```
gcc -o foo foo.c `pkg-config libidn --cflags --libs`
```

3 Stringprep Functions

Stringprep describes a framework for preparing Unicode text strings in order to increase the likelihood that string input and string comparison work in ways that make sense for typical users throughout the world. The stringprep protocol is useful for protocol identifier values, company and personal names, internationalized domain names, and other text strings.

STRINGPREP_NO_NFKC [Enumerated type of Stringprep_profile_flags]

STRINGPREP_NO_NFKC disables the NFKC normalization, as well as selecting the non-NFKC case folding tables. Usually the profile specifies BIDI and NFKC settings.

STRINGPREP_NO_BIDI [Enumerated type of Stringprep_profile_flags]

STRINGPREP_NO_BIDI disables the BIDI step. Usually the profile specifies BIDI and NFKC settings.

STRINGPREP_NO_UNASSIGNED Enumerated type of Stringprep_profile_flags]

STRINGPREP_NO_UNASSIGNED causes stringprep() abort with an error if string contains unassigned characters according to profile.

`int` **stringprep** (`char *` *`in`*, `size_t` *`maxlen`*, `int` *`flags`*, `Stringprep_profile *` *`profile`*) [Function]

in: input/ouput array with string to prepare.

maxlen: maximum length of input/output array.

flags: optional stringprep profile flags.

profile: pointer to stringprep profile to use.

Prepare the input UTF-8 string according to the stringprep profile. Normally application programmers use stringprep profile macros such as `stringprep_nameprep()`, `stringprep_kerberos5()` etc instead of calling this function directly.

Since the stringprep operation can expand the string, `maxlen` indicate how large the buffer holding the string is. The `flags` are one of Stringprep_profile_flags, or 0. The profile indicates processing details, see the profile header files, such as stringprep_generic.h and stringprep_nameprep.h for two examples. Your application can define new profiles, possibly re-using the generic stringprep tables that always will be part of the library. Note that you must convert strings entered in the systems locale into UTF-8 before using this function.

Returns 0 iff successful, or an error code.

`unsigned long` **stringprep_utf8_to_unichar** (`const char *` *`p`*) [Function]

p: a pointer to Unicode character encoded as UTF-8

Converts a sequence of bytes encoded as UTF-8 to a Unicode character. If `p` does not point to a valid UTF-8 encoded character, results are undefined.

the resulting character

int **stringprep_unichar_to_utf8** (unsigned long *c*, char * *outbuf*) [Function]

c: a ISO10646 character code

outbuf: output buffer, must have at least 6 bytes of space. If *NULL*, the length will be computed and returned and nothing will be written to `outbuf`.

Converts a single character to UTF-8.

number of bytes written

unsigned long * **stringprep_utf8_to_ucs4** (const char * *str*, ssize_t *len*, size_t * *items_written*) [Function]

str: a UTF-8 encoded string

len: the maximum length of `str` to use. If `len` < 0, then the string is nul-terminated.

items_written: location to store the number of characters in the result, or *NULL*.

Convert a string from UTF-8 to a 32-bit fixed width representation as UCS-4, assuming valid UTF-8 input. This function does no error checking on the input.

a pointer to a newly allocated UCS-4 string. This value must be freed with `free()`.

char * **stringprep_ucs4_to_utf8** (const unsigned long * *str*, ssize_t *len*, size_t * *items_read*, size_t * *items_written*) [Function]

str: a UCS-4 encoded string

len: the maximum length of `str` to use. If `len` < 0, then the string is terminated with a 0 character.

items_read: location to store number of characters read read, or *NULL*.

items_written: location to store number of bytes written or *NULL*. The value here stored does not include the trailing 0 byte.

Convert a string from a 32-bit fixed width representation as UCS-4. to UTF-8. The result will be terminated with a 0 byte.

a pointer to a newly allocated UTF-8 string. This value must be freed with `free()`. If an error occurs, *NULL* will be returned and `error` set.

char * **stringprep_utf8_nfkc_normalize** (const char * *str*, ssize_t *len*) [Function]

str: a UTF-8 encoded string.

len: length of `str`, in bytes, or -1 if `str` is nul-terminated.

Converts a string into canonical form, standardizing such issues as whether a character with an accent is represented as a base character and combining accent or as a single precomposed character. You should generally call `g_utf8_normalize()` before comparing two Unicode strings.

The normalization mode is NFKC (ALL COMPOSE). It standardizes differences that do not affect the text content, such as the above-mentioned accent representation. It standardizes the "compatibility" characters in Unicode, such as SUPERSCRIPT THREE to the standard forms (in this case DIGIT THREE). Formatting information may be lost but for most text operations such characters should be considered the same. It returns a result with composed forms rather than a maximally decomposed form.

a newly allocated string, that is the NFKC normalized form of `str`.

unsigned long * **stringprep_ucs4_nfkc_normalize** (unsigned long * *str*, ssize_t *len*) [Function]

str: a Unicode string.

len: length of `str` array, or -1 if `str` is nul-terminated.

Converts UCS4 string into UTF-8 and runs `stringprep_utf8_nfkc_normalize()`.

a newly allocated Unicode string, that is the NFKC normalized form of `str`.

const char * **stringprep_locale_charset** (*void*) [Function]

Return the character set used by the system locale. It will never return NULL, but use "ASCII" as a fallback.

char * **stringprep_convert** (const char * *str*, const char * *to_codeset*, const char * *from_codeset*) [Function]

str: input zero-terminated string.

to_codeset: name of destination character set.

from_codeset: name of origin character set, as used by `str`.

Convert the string from one character set to another using the system's `iconv()` function.

Returns newly allocated zero-terminated string which is `str` transcoded into to_codeset.

char * **stringprep_locale_to_utf8** (const char * *str*) [Function]

str: input zero terminated string.

Convert string encoded in the locale's character set into UTF-8 by using `stringprep_convert()`.

Returns newly allocated zero-terminated string which is `str` transcoded into UTF-8.

char * **stringprep_utf8_to_locale** (const char * *str*) [Function]

str: input zero terminated string.

Convert string encoded in UTF-8 into the locale's character set by using `stringprep_convert()`.

Returns newly allocated zero-terminated string which is `str` transcoded into the locale's character set.

int **stringprep_nameprep_no_unassigned** (char * *in*, int *maxlen*) [Function]

in: input/ouput array with string to prepare.

maxlen: maximum length of input/output array.

Prepare the input UTF-8 string according to the nameprep profile. The AllowUnassigned flag is false, use `stringprep_nameprep()` for true AllowUnassigned. Returns 0 iff successful, or an error code.

`int` **stringprep_kerberos5** (`char *` *in*, `int` *maxlen*) [Function]

in: input/ouput array with string to prepare.

maxlen: maximum length of input/output array.

Prepare the input UTF-8 string according to the draft Kerberos5 stringprep profile. Returns 0 iff successful, or an error code.

`int` **stringprep_plain** (`char *` *in*, `int` *maxlen*) [Function]

in: input/ouput array with string to prepare.

maxlen: maximum length of input/output array.

Prepare the input UTF-8 string according to the draft SASL ANONYMOUS profile. Returns 0 iff successful, or an error code.

`int` **stringprep_xmpp_nodeprep** (`char *` *in*, `int` *maxlen*) [Function]

in: input/ouput array with string to prepare.

maxlen: maximum length of input/output array.

Prepare the input UTF-8 string according to the draft XMPP node identifier profile. Returns 0 iff successful, or an error code.

`int` **stringprep_xmpp_resourceprep** (`char *` *in*, `int` *maxlen*) [Function]

in: input/ouput array with string to prepare.

maxlen: maximum length of input/output array.

Prepare the input UTF-8 string according to the draft XMPP resource identifier profile. Returns 0 iff successful, or an error code.

`int` **stringprep_generic** (`char *` *in*, `int` *maxlen*) [Function]

in: input/ouput array with string to prepare.

maxlen: maximum length of input/output array.

Prepare the input UTF-8 string according to a hypotetical "generic" stringprep profile. This is mostly used for debugging or when constructing new stringprep profiles. Returns 0 iff successful, or an error code.

4 Punycode Functions

Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It uniquely and reversibly transforms a Unicode string into an ASCII string. ASCII characters in the Unicode string are represented literally, and non-ASCII characters are represented by ASCII characters that are allowed in host name labels (letters, digits, and hyphens). This document defines a general algorithm called Bootstring that allows a string of basic code points to uniquely represent any string of code points drawn from a larger set. Punycode is an instance of Bootstring that uses particular parameter values specified by this document, appropriate for IDNA.

int **punycode_encode** (size_t *input_length*, const unsigned long *input*[], const unsigned char *case_flags*[], size_t * *output_length*, char *output*[]) [Function]

input_length: The input_length is the number of code points in the input.

output_length: The output_length is an in/out argument: the caller passes in the maximum number of code points that it can receive, and on successful return it will contain the number of code points actually output.

Converts Unicode to Punycode.

The return value can be any of the punycode_status values defined above except punycode_bad_input; if not punycode_success, then output_size and output might contain garbage.

int **punycode_decode** (size_t *input_length*, const char *input*[], size_t * *output_length*, unsigned long *output*[], unsigned char *case_flags*[]) [Function]

input_length: The input_length is the number of code points in the input.

output_length: The output_length is an in/out argument: the caller passes in the maximum number of code points that it can receive, and on successful return it will contain the actual number of code points output.

Converts Punycode to Unicode.

The return value can be any of the punycode_status values defined above; if not punycode_success, then output_length, output, and case_flags might contain garbage. On success, the decoder will never need to write an output_length greater than input_length, because of how the encoding is defined.

5 IDNA Functions

Until now, there has been no standard method for domain names to use characters outside the ASCII repertoire. The IDNA document defines internationalized domain names (IDNs) and a mechanism called IDNA for handling them in a standard fashion. IDNs use characters drawn from a large repertoire (Unicode), but IDNA allows the non-ASCII characters to be represented using only the ASCII characters already allowed in so-called host names today. This backward-compatible representation is required in existing protocols like DNS, so that IDNs can be introduced with no changes to the existing infrastructure. IDNA is only meant for processing domain names, not free text.

`int` **idna_to_ascii** (`const unsigned long *` *in*, `size_t` *inlen*, `char *` *out*, `int` *allowunassigned*, `int` *usestd3asciirules*) [Function]

in: input array with unicode code points.

inlen: length of input array with unicode code points.

out: output zero terminated string that must have room for at least 63 characters plus the terminating zero.

allowunassigned: boolean value as per IDNA specification.

usestd3asciirules: boolean value as per IDNA specification.

The ToASCII operation takes a sequence of Unicode code points that make up one label and transforms it into a sequence of code points in the ASCII range (0..7F). If ToASCII succeeds, the original sequence and the resulting sequence are equivalent labels.

It is important to note that the ToASCII operation can fail. ToASCII fails if any step of it fails. If any step of the ToASCII operation fails on any label in a domain name, that domain name MUST NOT be used as an internationalized domain name. The method for deadling with this failure is application-specific.

The inputs to ToASCII are a sequence of code points, the AllowUnassigned flag, and the UseSTD3ASCIIRules flag. The output of ToASCII is either a sequence of ASCII code points or a failure condition.

ToASCII never alters a sequence of code points that are all in the ASCII range to begin with (although it could fail). Applying the ToASCII operation multiple times has exactly the same effect as applying it just once.

Returns 0 on success, or an error code.

`int` **idna_to_unicode** (`const unsigned long *` *in*, `size_t` *inlen*, `unsigned long *` *out*, `size_t *` *outlen*, `int` *allowunassigned*, `int` *usestd3asciirules*) [Function]

in: input array with unicode code points.

inlen: length of input array with unicode code points.

out: output array with unicode code points.

outlen: on input, maximum size of output array with unicode code points, on exit, actual size of output array with unicode code points.

allowunassigned: boolean value as per IDNA specification.

usestd3asciirules: boolean value as per IDNA specification.

The ToUnicode operation takes a sequence of Unicode code points that make up one label and returns a sequence of Unicode code points. If the input sequence is a label in ACE form, then the result is an equivalent internationalized label that is not in ACE form, otherwise the original sequence is returned unaltered.

ToUnicode never fails. If any step fails, then the original input sequence is returned immediately in that step.

The ToUnicode output never contains more code points than its input. Note that the number of octets needed to represent a sequence of code points depends on the particular character encoding used.

The inputs to ToUnicode are a sequence of code points, the AllowUnassigned flag, and the UseSTD3ASCIIRules flag. The output of ToUnicode is always a sequence of Unicode code points.

Returns error condition, but it must only be used for debugging purposes. The output buffer is always guaranteed to contain the correct data according to the specification (sans malloc induced errors). NB! This means that you normally ignore the return code from this function, as checking it means breaking the standard.

`int` **idna_ucs4_to_ace** (`const unsigned long *` *input*, `char **` *output*) [Function]

input: zero terminated input Unicode string.

output: pointer to newly allocated output string.

Convert UCS-4 domain name to ASCII string. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

`int` **idna_utf8_to_ace** (`const char *` *input*, `char **` *output*) [Function]

input: zero terminated input UTF-8 string.

output: pointer to newly allocated output string.

Convert UTF-8 domain name to ASCII string. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

`int` **idna_locale_to_ace** (`const char *` *input*, `char **` *output*) [Function]

input: zero terminated input UTF-8 string.

output: pointer to newly allocated output string.

Convert domain name in the locale's encoding to ASCII string. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

int **idna_ucs4ace_to_ucs4** (const unsigned long * *input*, unsigned long ** *output*) [Function]

input: zero-terminated Unicode string.

output: pointer to newly allocated output Unicode string.

Convert possibly ACE encoded domain name in UCS-4 format into a UCS-4 string. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

int **idna_utf8ace_to_ucs4** (const char * *input*, unsigned long ** *output*) [Function]

input: zero-terminated UTF-8 string.

output: pointer to newly allocated output Unicode string.

Convert possibly ACE encoded domain name in UTF-8 format into a UCS-4 string. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

int **idna_utf8ace_to_utf8** (const char * *input*, char ** *output*) [Function]

input: zero-terminated UTF-8 string.

output: pointer to newly allocated output UTF-8 string.

Convert possibly ACE encoded domain name in UTF-8 format into a UTF-8 string. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

int **idna_utf8ace_to_locale** (const char * *input*, char ** *output*) [Function]

input: zero-terminated UTF-8 string.

output: pointer to newly allocated output string encoded in the current locale's character set.

Convert possibly ACE encoded domain name in UTF-8 format into a string encoded in the current locale's character set. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

int **idna_localeace_to_locale** (const char * *input*, char ** *output*) [Function]

input: zero-terminated string encoded in the current locale's character set.

output: pointer to newly allocated output string encoded in the current locale's character set.

Convert possibly ACE encoded domain name in the locale's character set into a string encoded in the current locale's character set. The AllowUnassigned flag is false and std3asciirules flag is false. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

Returns IDNA_SUCCESS on success, or error code.

6 Examples

This chapter contains example code which illustrate how 'Libidn' can be used when writing your own application.

6.1 Example 1

This example demonstrates how the stringprep functions are used.

```
/* example.c Example code showing how to use stringprep().
 * Copyright (C) 2002, 2003  Simon Josefsson
 *
 * This file is part of GNU Libidn.
 *
 * GNU Libidn is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * GNU Libidn is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with GNU Libidn; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 *
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stringprep_nameprep.h>

/*
 * Compiling using libtool and pkg-config is recommended:
 *
 * $ libtool cc -o example example.c `pkg-config --cflags --libs libidn`
 * $ ./example
 * Input string encoded as `ISO-8859-1':
 * Before locale2utf8 (length 2): aa 0a
 * Before stringprep (length 3): c2 aa 0a
 * After stringprep (length 2): 61 0a
 * $
 *
 */
```

```
int
main (int argc, char *argv[])
{
  char buf[BUFSIZ];
  char *p;
  int rc;
  size_t i;

  printf ("Input string encoded as ‘%s’: ", stringprep_locale_charset ());
  fflush (stdout);
  fgets (buf, BUFSIZ, stdin);

  printf ("Before locale2utf8 (length %d): ", strlen (buf));
  for (i = 0; i < strlen (buf); i++)
    printf ("%02x ", buf[i] & 0xFF);
  printf ("\n");

  p = stringprep_locale_to_utf8 (buf);
  if (p)
    {
      strcpy (buf, p);
      free (p);
    }
  else
    printf ("Could not convert string to UTF-8, continuing anyway...\n");

  printf ("Before stringprep (length %d): ", strlen (buf));
  for (i = 0; i < strlen (buf); i++)
    printf ("%02x ", buf[i] & 0xFF);
  printf ("\n");

  rc = stringprep (buf, BUFSIZ, 0, stringprep_nameprep);
  if (rc != STRINGPREP_OK)
    printf ("Stringprep failed with rc %d...\n", rc);
  else
    {
      printf ("After stringprep (length %d): ", strlen (buf));
      for (i = 0; i < strlen (buf); i++)
printf ("%02x ", buf[i] & 0xFF);
      printf ("\n");
    }

  return 0;
}
```

6.2 Example 2

This example demonstrates how the punycode functions are used.

```
/* example2.c Example code showing how to use punycode.
 * Copyright (C) 2002, 2003  Simon Josefsson
 * Copyright (C) 2002  Adam M. Costello
 *
 * This file is part of GNU Libidn.
 *
 * GNU Libidn is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * GNU Libidn is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with GNU Libidn; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 *
 */

/*
 * This file is derived from from draft-ietf-idn-punycode-03.txt by
 * Adam M. Costello.
 *
 * Disclaimer and license: Regarding this entire document or any
 * portion of it (including the pseudocode and C code), the author
 * makes no guarantees and is not responsible for any damage resulting
 * from its use.  The author grants irrevocable permission to anyone
 * to use, modify, and distribute it in any way that does not diminish
 * the rights of anyone else to use, modify, and distribute it,
 * provided that redistributed derivative works do not contain
 * misleading author or version information.  Derivative works need
 * not be licensed under similar terms.
 *
 */

#include <assert.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#include "punycode.h"

/* For testing, we'll just set some compile-time limits rather than */
/* use malloc(), and set a compile-time option rather than using a  */
/* command-line option.                                              */

enum
{
  unicode_max_length = 256,
  ace_max_length = 256
};

static void
usage (char **argv)
{
  fprintf (stderr,
   "\n"
   "%s -e reads code points and writes a Punycode string.\n"
   "%s -d reads a Punycode string and writes code points.\n"
   "\n"
   "Input and output are plain text in the native character set.\n"
   "Code points are in the form u+hex separated by whitespace.\n"
   "Although the specification allows Punycode strings to contain\n"
   "any characters from the ASCII repertoire, this test code\n"
   "supports only the printable characters, and needs the Punycode\n"
   "string to be followed by a newline.\n"
   "The case of the u in u+hex is the force-to-uppercase flag.\n",
   argv[0], argv[0]);
  exit (EXIT_FAILURE);
}

static void
fail (const char *msg)
{
  fputs (msg, stderr);
  exit (EXIT_FAILURE);
}

static const char too_big[] =
  "input or output is too large, recompile with larger limits\n";
static const char invalid_input[] = "invalid input\n";
static const char overflow[] = "arithmetic overflow\n";
static const char io_error[] = "I/O error\n";


/* The following string is used to convert printable */
/* characters between ASCII and the native charset:  */
```

```
static const char print_ascii[] =
  "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n"
  "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n"
  " !\"#$%&’()*+,-./"
  "0123456789:;<=>?"
  "\x40""ABCDEFGHIJKLMNO"
  "PQRSTUVWXYZ[\\]^_" "‘abcdefghijklmno" "pqrstuvwxyz{|}~\n";


int
main (int argc, char **argv)
{
  int status;
  int r;
  size_t input_length, output_length, j;
  unsigned char case_flags[unicode_max_length];

  if (argc != 2)
    usage (argv);
  if (argv[1][0] != ’-’)
    usage (argv);
  if (argv[1][2] != 0)
    usage (argv);

  if (argv[1][1] == ’e’)
    {
      unsigned long input[unicode_max_length];
      unsigned long codept;
      char output[ace_max_length + 1], uplus[3];
      int c;

      /* Read the input code points: */

      input_length = 0;

      for (;;)
{
  r = scanf ("%2s%lx", uplus, &codept);
  if (ferror (stdin))
    fail (io_error);
  if (r == EOF || r == 0)
    break;

  if (r != 2 || uplus[1] != ’+’ || codept > (unsigned long) -1)
    {
      fail (invalid_input);
```

```
    }

  if (input_length == unicode_max_length)
    fail (too_big);

  if (uplus[0] == 'u')
    case_flags[input_length] = 0;
  else if (uplus[0] == 'U')
    case_flags[input_length] = 1;
  else
    fail (invalid_input);

  input[input_length++] = codept;
}

      /* Encode: */

      output_length = ace_max_length;
      status = punycode_encode (input_length, input, case_flags,
&output_length, output);
      if (status == PUNYCODE_BAD_INPUT)
fail (invalid_input);
      if (status == PUNYCODE_BIG_OUTPUT)
fail (too_big);
      if (status == PUNYCODE_OVERFLOW)
fail (overflow);
      assert (status == PUNYCODE_SUCCESS);

      /* Convert to native charset and output: */

      for (j = 0; j < output_length; ++j)
{
  c = output[j];
  assert (c >= 0 && c <= 127);
  if (print_ascii[c] == 0)
    fail (invalid_input);
  output[j] = print_ascii[c];
}

      output[j] = 0;
      r = puts (output);
      if (r == EOF)
fail (io_error);
      return EXIT_SUCCESS;
    }

  if (argv[1][1] == 'd')
```

```
    {
      char input[ace_max_length + 2], *p, *pp;
      unsigned long output[unicode_max_length];

      /* Read the Punycode input string and convert to ASCII: */

      fgets (input, ace_max_length + 2, stdin);
      if (ferror (stdin))
fail (io_error);
      if (feof (stdin))
fail (invalid_input);
      input_length = strlen (input) - 1;
      if (input[input_length] != '\n')
fail (too_big);
      input[input_length] = 0;

      for (p = input; *p != 0; ++p)
{
  pp = strchr (print_ascii, *p);
  if (pp == 0)
    fail (invalid_input);
  *p = pp - print_ascii;
}

      /* Decode: */

      output_length = unicode_max_length;
      status = punycode_decode (input_length, input, &output_length,
output, case_flags);
      if (status == PUNYCODE_BAD_INPUT)
fail (invalid_input);
      if (status == PUNYCODE_BIG_OUTPUT)
fail (too_big);
      if (status == PUNYCODE_OVERFLOW)
fail (overflow);
      assert (status == PUNYCODE_SUCCESS);

      /* Output the result: */

      for (j = 0; j < output_length; ++j)
{
  r = printf ("%s+%04lX\n",
      case_flags[j] ? "U" : "u", (unsigned long) output[j]);
  if (r < 0)
    fail (io_error);
}
```

```
        return EXIT_SUCCESS;
      }

    usage (argv);
    return EXIT_SUCCESS; /* not reached, but quiets compiler warning */

  }
```

6.3 Example 3

This example demonstrates how the library is used to convert internationalized domain names into ASCII compatible names.

```
/* example3.c Example code showing how to use Libidn.
 * Copyright (C) 2002, 2003  Simon Josefsson
 *
 * This file is part of GNU Libidn.
 *
 * GNU Libidn is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * GNU Libidn is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with GNU Libidn; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 *
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stringprep.h> /* stringprep_locale_charset() */
#include <idna.h> /* idna_locale_to_ace() */

/*
 * Compiling using libtool and pkg-config is recommended:
 *
 * $ libtool cc -o example3 example3.c ‘pkg-config --cflags --libs libidn‘
 * $ ./example3
 * Input domain encoded as ‘ISO-8859-1’: www.rksmrgs.example
 * Read string (length 23): 77 77 77 2e 72 e4 6b 73 6d f6 72 67 e5 73 aa 2e 65 78 61 6
```

```
 * ACE label (length 33): 'www.xn--rksmrgsa-0zap8p.example'
 * 77 77 77 2e 78 6e 2d 2d 72 6b 73 6d 72 67 73 61 2d 30 7a 61 70 38 70 2e 65 78 61 6d
 * $
 *
 */

int
main (int argc, char *argv[])
{
  char buf[BUFSIZ];
  char *p;
  int rc;
  size_t i;

  printf ("Input domain encoded as `%s': ", stringprep_locale_charset ());
  fflush (stdout);
  fgets (buf, BUFSIZ, stdin);
  buf[strlen (buf) - 1] = '\0';

  printf ("Read string (length %d): ", strlen (buf));
  for (i = 0; i < strlen (buf); i++)
    printf ("%02x ", buf[i] & 0xFF);
  printf ("\n");

  rc = idna_locale_to_ace (buf, &p);
  if (rc != IDNA_SUCCESS)
    {
      printf ("ToASCII() failed... %d\n", rc);
      exit (1);
    }

  printf ("ACE label (length %d): '%s'\n", strlen (p), p);
  for (i = 0; i < strlen (p); i++)
    printf ("%02x ", p[i] & 0xFF);
  printf ("\n");

  free (p);

  return 0;
}
```

6.4 Example 4

This example demonstrates how the library is used to convert ASCII compatible names to internationalized domain names.

```
/* example4.c Example code showing how to use Libidn.
```

```
 * Copyright (C) 2002, 2003  Simon Josefsson
 *
 * This file is part of GNU Libidn.
 *
 * GNU Libidn is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * GNU Libidn is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with GNU Libidn; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 *
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stringprep.h> /* stringprep_locale_charset() */
#include <idna.h> /* idna_localeace_to_locale() */

/*
 * Compiling using libtool and pkg-config is recommended:
 *
 * $ libtool cc -o example4 example4.c ‘pkg-config --cflags --libs libidn‘
 * $ ./example4
 * Input domain encoded as ‘ISO-8859-1’: www.xn--rksmrgsa-0zap8p.example
 * Read string (length 33): 77 77 77 2e 78 6e 2d 2d 72 6b 73 6d 72 67 73 61 2d 30 7a 6
 * ACE label (length 23): ’www.rksmrgsa.example’
 * 77 77 77 2e 72 e4 6b 73 6d f6 72 67 e5 73 61 2e 65 78 61 6d 70 6c 65
 * $
 *
 */

int
main (int argc, char *argv[])
{
  char buf[BUFSIZ];
  char *p;
  int rc;
  size_t i;
```

```
  printf ("Input domain encoded as `%s': ", stringprep_locale_charset ());
  fflush (stdout);
  fgets (buf, BUFSIZ, stdin);
  buf[strlen (buf) - 1] = '\0';

  printf ("Read string (length %d): ", strlen (buf));
  for (i = 0; i < strlen (buf); i++)
    printf ("%02x ", buf[i] & 0xFF);
  printf ("\n");

  rc = idna_localeace_to_locale (buf, &p);
  if (rc != IDNA_SUCCESS)
    {
      printf ("ToUnicode() failed... %d\n", rc);
      exit (1);
    }

  printf ("ACE label (length %d): '%s'\n", strlen (p), p);
  for (i = 0; i < strlen (p); i++)
    printf ("%02x ", p[i] & 0xFF);
  printf ("\n");

  free (p);

  return 0;
}
```

7 Acknowledgements

Simon Josefsson created the library autumn 2002 when he really should have been studying mathematics.

The punycode code was taken from the IETF IDN Punycode specification, by Adam M. Costello.

Some functions (see nfkc.c and toutf8.c) has been borrowed from GLib downloaded from www.gtk.org.

Several people reported bugs, sent patches or suggested improvements, see the file THANKS.

Appendix A GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence the
version number 2.1.]

A.0.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software—typically libraries—of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the *Lesser* General Public License because it does *Less* to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

A.0.2 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A “library” means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The “Library”, below, refers to any such software library or work which has been distributed under these terms. A “work based on the Library” means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term “modification”.)

“Source code” for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library’s complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. The modified work must itself be a software library.

   b. You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

   c. You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

   d. If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

      (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

   Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

   This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

   If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

   However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

   When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

   You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

   a. Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
   b. Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
   c. Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
   d. If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
   e. Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

   For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

   a. Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

   b. Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library 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.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way

you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public 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.

    Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.
14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE

RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

A.0.3 How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the library's name and an idea of what it does.
Copyright (C) year  name of author

This library is free software; you can redistribute it and/or modify it
under the terms of the GNU Lesser General Public License as published by
the Free Software Foundation; either version 2.1 of the License, or (at
your option) any later version.

This library is distributed in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307,
USA.
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the library
'Frob' (a library for tweaking knobs) written by James Random Hacker.

signature of Ty Coon, 1 April 1990
Ty Coon, President of Vice
```

That's all there is to it!

Appendix B Copying This Manual

B.1 GNU Free Documentation License

0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other written 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 that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

   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. (For example, 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.

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 "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and 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 has been designed to thwart or discourage subsequent modification by readers is not Transparent. 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 designed for human modification. Opaque formats include PostScript, PDF, 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 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.

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 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 publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. 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 less than five).

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", and 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. In any section entitled "Acknowledgments" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgments 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 as "Endorsements" or to conflict in title with any Invariant Section.

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.

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 "Acknowledgments", 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, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they 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 quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around 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 provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

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.

B.1.1 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 1.1
or any later version published by the Free Software Foundation;
with the Invariant Sections being list their titles, with the
Front-Cover Texts being list, and with the Back-Cover Texts
being list.  A copy of the license is included in the section
entitled ``GNU Free Documentation License''.
```

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being *list*"; likewise for Back-Cover Texts.

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

A

AIX 2

C

Compiling your application 6

D

Debian 2

E

Examples 16

F

FDL, GNU Free Documentation License 37
FreeBSD 3

H

HP-UX 2

I

IDNA Functions 12
IRIX 2

L

LGPL, Lesser General Public License 28

M

Mandrake 2

N

NetBSD 3

O

OpenBSD 3

P

Punycode Functions 11

R

RedHat 2
RedHat Advanced Server 2
Reporting Bugs 3

S

Solaris 3
Stringprep Functions 7
SuSE 2
SuSE Linux 2

T

Tru64 2

W

Windows 2