

NAME

locale - describes a locale definition file

DESCRIPTION

The **locale** definition file contains all the information that the `localedef(1)` command needs to convert it into the binary locale database.

The definition files consist of sections which each describe a locale category in detail.

Syntax

The locale definition file starts with a header that may consist of the following keywords:

<escape_char>

is followed by a character that should be used as the escape-character for the rest of the file to mark characters that should be interpreted in a special way. It defaults to the backslash (`\`).

<comment_char>

is followed by a character that will be used as the comment-character for the rest of the file. It defaults to the number sign (`#`).

The locale definition has one part for each locale category. Each part can be copied from another existing locale or can be defined from scratch. If the category should be copied, the only valid keyword in the definition is **copy** followed by the name of the locale which should be copied.

Locale category sections

The following category sections are defined by POSIX:

- * **LC_CTYPE**
- * **LC_COLLATE**
- * **LC_MESSAGES**
- * **LC_MONETARY**
- * **LC_NUMERIC**
- * **LC_TIME**

In addition, since version 2.2, the GNU C library supports the following nonstandard categories:

- * **LC_ADDRESS**
- * **LC_IDENTIFICATION**
- * **LC_MEASUREMENT**
- * **LC_NAME**
- * **LC_PAPER**
- * **LC_TELEPHONE**

LC_ADDRESS

The definition starts with the string **LC_ADDRESS** in the first column.

The following keywords are allowed:

postal_fmt

followed by a string containing field descriptors that define the format used for postal addresses in the locale. The following field descriptors are recognized:

- %a Care of person, or organization.
- %f Firm name.
- %d Department name.

%b	Building name.
%s	Street or block (e.g., Japanese) name.
%h	House number or designation.
%N	Insert an end-of-line if the previous descriptor's value was not an empty string; otherwise ignore.
%t	Insert a space if the previous descriptor's value was not an empty string; otherwise ignore.
%r	Room number, door designation.
%e	Floor number.
%C	Country designation, from the <country_post> keyword.
%z	Zip number, postal code.
%T	Town, city.
%S	State, province, or prefecture.
%c	Country, as taken from data record.

Each field descriptor may have an R after the % to specify that the information is taken from a Romanized version string of the entity.

country_name

followed by the country name in the language of the current document (e.g., Deutschland for the *de_DE* locale).

country_post

followed by the abbreviation of the country (see CERT_MAILCODES).

country_ab2

followed by the two-letter abbreviation of the country (ISO 3166).

country_ab3

followed by the three-letter abbreviation of the country (ISO 3166).

country_num

followed by the numeric country code (ISO 3166).

country_car

followed by the code for the country car number.

country_isbn

followed by the ISBN code (for books).

lang_name

followed by the language name in the language of the current document.

lang_ab

followed by the two-letter abbreviation of the language (ISO 639).

lang_term

followed by the three-letter abbreviation of the language (ISO 639-2/T).

lang_lib

followed by the three-letter abbreviation of the language for library use (ISO 639-2/B). Applications should in general prefer *lang_term* over *lang_lib*.

The **LC_ADDRESS** definition ends with the string *END LC_ADDRESS*.

LC_CTYPE

The definition starts with the string **LC_CTYPE** in the first column.

The following keywords are allowed:

- upper* followed by a list of uppercase letters. The letters **A** through **Z** are included automatically. Characters also specified as **ascntrl**, **digit**, **punct**, or **space** are not allowed.
- lower* followed by a list of lowercase letters. The letters **a** through **z** are included automatically. Characters also specified as **cntrl**, **digit**, **punct**, or **space** are not allowed.
- alpha* followed by a list of letters. All character specified as either **upper** or **lower** are automatically included. Characters also specified as **cntrl**, **digit**, **punct**, or **space** are not allowed.
- digit* followed by the characters classified as numeric digits. Only the digits **0** through **9** are allowed. They are included by default in this class.
- space* followed by a list of characters defined as white-space characters. Characters also specified as **upper**, **lower**, **alpha**, **digit**, **graph**, or **xdigit** are not allowed. The characters **<space>**, **<form-feed>**, **<newline>**, **<carriage-return>**, **<tab>**, and **<vertical-tab>** are automatically included.
- cntrl* followed by a list of control characters. Characters also specified as **upper**, **lower**, **alpha**, **digit**, **punct**, **graph**, **print**, or **xdigit** are not allowed.
- punct* followed by a list of punctuation characters. Characters also specified as **upper**, **lower**, **alpha**, **digit**, **cntrl**, **xdigit**, or the **<space>** character are not allowed.
- graph* followed by a list of printable characters, not including the **<space>** character. The characters defined as **upper**, **lower**, **alpha**, **digit**, **xdigit**, and **punct** are automatically included. Characters also specified as **cntrl** are not allowed.
- print* followed by a list of printable characters, including the **<space>** character. The characters defined as **upper**, **lower**, **alpha**, **digit**, **xdigit**, **punct**, and the **<space>** character are automatically included. Characters also specified as **cntrl** are not allowed.
- xdigit* followed by a list of characters classified as hexadecimal digits. The decimal digits must be included followed by one or more set of six characters in ascending order. The following characters are included by default: **0** through **9**, **a** through **f**, **A** through **F**.
- blank* followed by a list of characters classified as **blank**. The characters **<space>** and **<tab>** are automatically included.
- toupper* followed by a list of mappings from lowercase to uppercase letters. Each mapping is a pair of a lowercase and an uppercase letter separated with a **,** and enclosed in parentheses. The members of the list are separated with semicolons.
- tolower* followed by a list of mappings from uppercase to lowercase letters. If the keyword *tolower* is not present, the reverse of the *toupper* list is used.

The **LC_CTYPE** definition ends with the string **END LC_CTYPE**.

LC_COLLATE

The **LC_COLLATE** category defines the rules for collating characters. Due to limitations of libc not all POSIX-options are implemented.

The definition starts with the string **LC_COLLATE** in the first column.

The following keywords are allowed:

collating-element

followed by the definition of a collating-element symbol representing a multicharacter collating element.

collating-symbol

followed by the definition of a collating symbol that can be used in collation order statements.

The order-definition starts with a line:

order_start

followed by a list of keywords chosen from **forward**, **backward**, or **position**. The order definition consists of lines that describe the order and is terminated with the keyword *order_end*.

The **LC_COLLATE** definition ends with the string *END LC_COLLATE*.

LC_IDENTIFICATION

This category contains meta-information about the locale definition.

The definition starts with the string **LC_IDENTIFICATION** in the first column.

The following keywords are allowed:

title followed by the title of the locale document (e.g., Maori language locale for New Zealand).

source followed by the name of the organization that maintains this document.

address

followed by the address of the organization that maintains this document.

contact

followed by the name of the contact person at the organization that maintains this document.

email followed by the email address of the person or organization that maintains this document.

tel

followed by the telephone number (in international format) of the organization that maintains this document.

fax

followed by the fax number (in international format) of the organization that maintains this document.

language

followed by the name of the language to which this document applies.

territory

followed by the name of the country/geographic extent to which this document applies.

audience

followed by a description of the audience for which this document is intended.

application

followed by a description of any special application for which this document is intended.

abbreviation

followed by the short name for this document.

revision

followed by the revision number of this document.

date

followed by the revision date of this document.

In addition, for each of the categories defined by the document, there should be a line starting with the keyword *category*, followed by:

- * a string that identifies this locale category definition,
- * a semicolon, and
- * one of the **LC_*** identifiers.

The **LC_IDENTIFICATION** definition ends with the string *END LC_IDENTIFICATION*.

LC_MESSAGES

The definition starts with the string **LC_MESSAGES** in the first column.

The following keywords are allowed:

yesexpr

followed by a regular expression that describes possible yes-responses.

noexpr followed by a regular expression that describes possible no-responses.

yesstr followed by the output string corresponding to yes.

nostr followed by the output string corresponding to no.

The **LC_MESSAGES** definition ends with the string *END LC_MESSAGES*.

LC_MEASUREMENT

The definition starts with the string **LC_MEASUREMENT** in the first column.

The following keywords are allowed:

measurement

followed by number identifying the standard used for measurement. The following values are recognized:

1 Metric.

2 US customary measurements.

The **LC_MEASUREMENT** definition ends with the string *END LC_MEASUREMENT*.

LC_MONETARY

The definition starts with the string **LC_MONETARY** in the first column.

The following keywords are allowed:

int_curr_symbol

followed by the international currency symbol. This must be a 4-character string containing the international currency symbol as defined by the ISO 4217 standard (three characters) followed by a separator.

currency_symbol

followed by the local currency symbol.

mon_decimal_point

followed by the string that will be used as the decimal delimiter when formatting monetary quantities.

mon_thousands_sep

followed by the string that will be used as a group separator when formatting monetary quantities.

mon_grouping

followed by a sequence of integers separated by semicolons that describe the formatting of monetary quantities. See *rouping* below for details.

positive_sign

followed by a string that is used to indicate a positive sign for monetary quantities.

negative_sign

followed by a string that is used to indicate a negative sign for monetary quantities.

int_frac_digits

followed by the number of fractional digits that should be used when formatting with the *int_curr_symbol*.

frac_digits

followed by the number of fractional digits that should be used when formatting with the *currency_symbol*.

p_cs_precedes

followed by an integer that indicates the placement of *currency_symbol* for a nonnegative formatted monetary quantity:

- 0** the symbol succeeds the value.
- 1** the symbol precedes the value.

n_cs_precedes

followed by an integer that indicates the placement of *currency_symbol* for a negative formatted monetary quantity. The same values are recognized as for *p_cs_precedes*.

int_p_cs_precedes

followed by an integer that indicates the placement of *int_currency_symbol* for a nonnegative internationally formatted monetary quantity. The same values are recognized as for *p_cs_precedes*.

int_n_cs_precedes

followed by an integer that indicates the placement of *int_currency_symbol* for a negative internationally formatted monetary quantity. The same values are recognized as for *p_cs_precedes*.

p_sep_by_space

followed by an integer that indicates the separation of *currency_symbol*, the sign string, and the value for a nonnegative formatted monetary quantity. The following values are recognized:

- 0** No space separates the currency symbol and the value.
- 1** If the currency symbol and the sign string are adjacent, a space separates them from the value; otherwise a space separates the currency symbol and the value.
- 2** If the currency symbol and the sign string are adjacent, a space separates them from the value; otherwise a space separates the sign string and the value.

n_sep_by_space

followed by an integer that indicates the separation of *currency_symbol*, the sign string, and the value for a negative formatted monetary quantity. The same values are recognized as for *p_sep_by_space*.

int_p_sep_by_space

followed by an integer that indicates the separation of *int_currency_symbol*, the sign string, and the value for a nonnegative internationally formatted monetary quantity. The same values are recognized as for *p_sep_by_space*.

int_n_sep_by_space

followed by an integer that indicates the separation of *int_currency_symbol*, the sign string, and the value for a negative internationally formatted monetary quantity. The same values are recognized as for *p_sep_by_space*.

p_sign_posn

followed by an integer that indicates where the *positive_sign* should be placed for a nonnegative monetary quantity:

- 0** Parentheses enclose the quantity and the *currency_symbol* or *int_curr_symbol*.
- 1** The sign string precedes the quantity and the *currency_symbol* or the *int_curr_symbol*.
- 2** The sign string succeeds the quantity and the *currency_symbol* or the *int_curr_symbol*.

3 The sign string precedes the *currency_symbol* or the *int_curr_symbol*.

4 The sign string succeeds the *currency_symbol* or the *int_curr_symbol*.

n_sign_posn

followed by an integer that indicates where the *negative_sign* should be placed for a negative monetary quantity. The same values are recognized as for *p_sign_posn*.

int_p_sign_posn

followed by an integer that indicates where the *negative_sign* should be placed for a non-negative internationally formatted monetary quantity. The same values are recognized as for *p_sign_posn*.

int_n_sign_posn

followed by an integer that indicates where the *negative_sign* should be placed for a negative internationally formatted monetary quantity. The same values are recognized as for *p_sign_posn*.

The **LC_MONETARY** definition ends with the string *END LC_MONETARY*.

LC_NAME

The definition starts with the string **LC_NAME** in the first column.

Various keywords are allowed, but only *name_fmt* is mandatory. Other keywords are needed only if there is common convention to use the corresponding salutation in this locale. The allowed keywords are as follows:

name_fmt

followed by a string containing field descriptors that define the format used for names in the locale. The following field descriptors are recognized:

%f Family name(s).

%F Family names in uppercase.

%g First given name.

%G First given initial.

%l First given name with Latin letters.

%o Other shorter name.

%m Additional given name(s).

%M Initials for additional given name(s).

%p Profession.

%s Salutation, such as Doctor.

%S Abbreviated salutation, such as Mr. or Dr..

%d Salutation, using the FDCC-sets conventions.

%t If the preceding field descriptor resulted in an empty string, then the empty string, otherwise a space character.

name_gen

followed by the general salutation for any gender.

name_mr

followed by the salutation for men.

name_mrs

followed by the salutation for married women.

name_miss

followed by the salutation for unmarried women.

name_ms

followed by the salutation valid for all women.

The **LC_NAME** definition ends with the string *END LC_NAME*.

LC_NUMERIC

The definition starts with the string **LC_NUMERIC** in the first column.

The following keywords are allowed:

decimal_point

followed by the string that will be used as the decimal delimiter when formatting numeric quantities.

thousands_sep

followed by the string that will be used as a group separator when formatting numeric quantities.

grouping

followed by a sequence of integers separated by semicolons that describe the formatting of numeric quantities.

Each integer specifies the number of digits in a group. The first integer defines the size of the group immediately to the left of the decimal delimiter. Subsequent integers define succeeding groups to the left of the previous group. If the last integer is not -1, then the size of the previous group (if any) is repeatedly used for the remainder of the digits. If the last integer is -1, then no further grouping is performed.

The **LC_NUMERIC** definition ends with the string *END LC_NUMERIC*.

LC_PAPER

The definition starts with the string **LC_PAPER** in the first column.

The following keywords are allowed:

height followed by the height, in millimeters, of the standard paper format.

width followed by the width, in millimeters, of the standard paper format.

The **LC_PAPER** definition ends with the string *END LC_PAPER*.

LC_TELEPHONE

The definition starts with the string **LC_TELEPHONE** in the first column.

The following keywords are allowed:

tel_int_fmt

followed by a string that contains field descriptors that identify the format used to dial international numbers. The following field descriptors are recognized:

%a Area code without nationwide prefix (the prefix is often 00).

%A Area code including nationwide prefix.

%l Local number (within area code).

%e Extension (to local number).

%c Country code.

%C Alternate carrier service code used for dialing abroad.

%t If the preceding field descriptor resulted in an empty string, then the empty string, otherwise a space character.

tel_dom_fmt

followed by a string that contains field descriptors that identify the format used to dial domestic numbers. The recognized field descriptors are the same as *fortel_int_fmt*.

int_select

followed by the prefix used to call international phone numbers.

int_prefix

followed by the prefix used from other countries to dial this country.

The **LC_TELEPHONE** definition ends with the string *END LC_TELEPHONE*.

LC_TIME

The definition starts with the string **LC_TIME** in the first column.

The following keywords are allowed:

abday followed by a list of abbreviated names of the days of the week. The list starts with the first day of the week as specified by *week* (Sunday by default). See NOTES.

day followed by a list of names of the days of the week. The list starts with the first day of the week as specified by *week* (Sunday by default). See NOTES.

abmon followed by a list of abbreviated month names.

mon followed by a list of month names.

am_pm

followed by the appropriate representation of the **am** and **pm** strings. This should be left empty for locales not using AM/PM convention.

d_t_fmt

followed by the appropriate date and time format.

d_fmt followed by the appropriate date format.

t_fmt followed by the appropriate time format.

t_fmt_ampm

followed by the appropriate time format when using 12h clock format. This should be left empty for locales not using AM/PM convention.

week followed by a list of three values: The number of days in a week (by default 7), a date of beginning of the week (by default corresponds to Sunday), and the minimal length of the first week in year (by default 4). Regarding the start of the week, **19971130** shall be used for Sunday and **19971201** shall be used for Monday. See NOTES.

first_weekday (since glibc 2.2)

followed by the number of the first day from the *day* list to be shown in calendar applications. The default value of **1** corresponds to either Sunday or Monday depending on the value of the second *week* list item. See NOTES.

first_workday (since glibc 2.2)

followed by the number of the first working day from the *day* list. The default value is **2**. See NOTES.

cal_direction

followed by a value that indicates the direction for the display of calendar dates, as follows:

- 1** Left-right from top.
- 2** Top-down from left.
- 3** Right-left from top.

date_fmt

followed by the appropriate date representation for [date\(1\)](#).

The **LC_TIME** definition ends with the string *END LC_TIME*.

FILES

/usr/lib/locale/locale-archive

Usual default locale archive location.

/usr/share/i18n/locales

Usual default path for locale definition files.

CONFORMING TO

POSIX.2, ISO/IEC TR 14652.

NOTES

The collective GNU C library community wisdom regarding *abday*, *day*, *week*, *first_weekday*, and *first_workday* states at <https://sourceware.org/glibc/wiki/Locales> the following:

- * The value of the second *week* list item specifies the base of the *abday* and *day* lists.
- * *first_weekday* specifies the offset of the first day-of-week in the *abday* and *day* lists.
- * For compatibility reasons, all glibc locales should set the value of the second *week* list item to **19971130** (Sunday) and base the *abday* and *day* lists appropriately, and set *first_weekday* to **1** or **2**, depending on whether the week actually starts on Sunday or Monday for the locale.

BUGS

This manual page isn't complete.

SEE ALSO

[locale\(1\)](#), [localedef\(1\)](#), [localeconv\(3\)](#), [newlocale\(3\)](#), [setlocale\(3\)](#), [uselocale\(3\)](#), [charmap\(5\)](#), [charsets\(7\)](#), [locale\(7\)](#), [unicode\(7\)](#), [utf-8\(7\)](#)

COLOPHON

This page is part of release 3.74 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at <http://www.kernel.org/doc/man-pages/>.