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. See **locale**(7) for additional details for these categories.

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 in double quotes which should be copied. The exceptions for this rule are **LC_COLLATE** and **LC_CTYPE** where a *copy* statement can be followed by locale-specific rules and selected overrides.

When defining a locale or a category from scratch, an existing system- provided locale definition file should be used as a reference to follow common glibc conventions.

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 glibc 2.2, the GNU C library supports the following nonstandard categories:

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

See **locale**(7) for a more detailed description of each category.

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:

%n   Person's name, possibly constructed with the **LC_NAME** *name_fmt* keyword (since glibc 2.24).

%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.

%l   Local township within town or city (since glibc 2.24).

%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 international license plate country code.

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 **cntrl**, **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.

charclass
> followed by a list of locale-specific character class names which are then to be defined in the locale.

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.

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.

map totitle
> followed by a list of mapping pairs of characters and letters to be used in titles (headings).

class followed by a locale-specific character class definition, starting with the class name followed by the characters belonging to the class.

charconv
>   followed by a list of locale-specific character mapping names which are then to be defined in the locale.

outdigit
>   followed by a list of alternate output digits for the locale.

map to_inpunct
>   followed by a list of mapping pairs of alternate digits and separators for input digits for the locale.

map to_outpunct
>   followed by a list of mapping pairs of alternate separators for output for the locale.

translit_start
>   marks the start of the transliteration rules section. The section can contain the *include* keyword in the beginning followed by locale-specific rules and overrides. Any rule specified in the locale file will override any rule copied or included from other files. In case of duplicate rule definitions in the locale file, only the first rule is used.
>
>   A transliteration rule consist of a character to be transliterated followed by a list of transliteration targets separated by semicolons. The first target which can be presented in the target character set is used, if none of them can be used the *default_missing* character will be used instead.

include in the transliteration rules section includes a transliteration rule file (and optionally a repertoire map file).

default_missing
>   in the transliteration rules section defines the default character to be used for transliteration where none of the targets cannot be presented in the target character set.

translit_end
>   marks the end of the transliteration rules.

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

LC_COLLATE

Note that glibc does not support all POSIX-defined options, only the options described below are supported (as of glibc 2.23).

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

The following keywords are allowed:

coll_weight_max
>   followed by the number representing used collation levels. This keyword is recognized but ignored by glibc.

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.

define followed by **string** to be evaluated in an *ifdef* **string** / *else* / *endif* construct.

reorder−after
>   followed by a redefinition of a collation rule.

reorder−end
>   marks the end of the redefinition of a collation rule.
>   *-sections−after*
reorde
>   followed by a script name to reorder listed scripts after.

reorder−sections−end
> marks the end of the reordering of sections.

script   followed by a declaration of a script.

symbol−equivalence
> followed by a collating-symbol to be equivalent to another defined collating-symbol.

The collation rule 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 collation order and is terminated with the keyword *order_end*.

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

LC_IDENTIFICATION

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. As of glibc 2.24, this keyword is deprecated in favor of other contact methods.

fax   followed by the fax number (in international format) of the organization that maintains this document. As of glibc 2.24, this keyword is deprecated in favor of other contact methods.

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 provider of the source of 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:

(1)   a string that identifies this locale category definition,

(2)   a semicolon, and

(3)   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 single-character string that will be used as the decimal delimiter when formatting monetary quantities.

mon_thousands_sep
followed by the single-character 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 *grouping* 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.

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_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*.

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*.

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

int_p_sep_by_space
> followed by an integer that indicates the separation of *int_curr_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_curr_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*.

int_p_sign_posn
> followed by an integer that indicates where the *positive_sign* should be placed for a nonnegative 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 single-character string that will be used as the decimal delimiter when formatting numeric quantities.

thousands_sep
> followed by the single-character 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 for *tel_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.

  d_t_fmt
     followed by the appropriate date and time format (for syntax, see **strftime**(3)).

  d_fmt followed by the appropriate date format (for syntax, see **strftime**(3)).

  t_fmt followed by the appropriate time format (for syntax, see **strftime**(3)).

  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.

  t_fmt_ampm
     followed by the appropriate time format (for syntax, see **strftime**(3)) when using 12h clock for-
     mat. This should be left empty for locales not using AM/PM convention.

  era   followed by semicolon-separated strings that define how years are counted and displayed for each
     era in the locale. Each string has the following format:

     direction:*offset*:*start_date*:*end_date*:*era_name*:*era_format*

     The fields are to be defined as follows:

     direction
       Either + or −. + means the years closer to *start_date* have lower numbers than years closer to
       end_date. − means the opposite.

     offset
       The number of the year closest to *start_date* in the era, corresponding to the *%Ey* descriptor
       (see **strptime**(3)).

     start_date
       The start of the era in the form of *yyyy/mm/dd*. Years prior AD 1 are represented as negative
       numbers.

     end_date
       The end of the era in the form of *yyyy/mm/dd*, or one of the two special values of −* or +*.
       −* means the ending date is the beginning of time. +* means the ending date is the end of
       time.

     era_name
       The name of the era corresponding to the *%EC* descriptor (see **strptime**(3)).

     era_format
       The format of the year in the era corresponding to the *%EY* descriptor (see **strptime**(3)).

era_d_fmt
> followed by the format of the date in alternative era notation, corresponding to the *%Ex* descriptor (see **strptime**(3)).

era_t_fmt
> followed by the format of the time in alternative era notation, corresponding to the *%EX* descriptor (see **strptime**(3)).

era_d_t_fmt
> followed by the format of the date and time in alternative era notation, corresponding to the *%Ec* descriptor (see **strptime**(3)).

alt_digits
> followed by the alternative digits used for date and time in the locale.

week   followed by a list of three values separated by semicolons: 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 day from the *day* list to be shown as the first day of the week 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 number 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) (for syntax, see **strftime**(3)).

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.

STANDARDS

POSIX.2.

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* and *first_workday* to **1** or **2**, depending on whether the week and work week actually starts on Sunday or Monday for the locale.

SEE ALSO

      iconv(1), **locale**(1), **localedef**(1), **localeconv**(3), **newlocale**(3), **setlocale**(3), **strftime**(3), **strptime**(3), **use-locale**(3), **charmap**(5), **charsets**(7), **locale**(7), **unicode**(7), **utf−8**(7)

      iconv(1), **locale**(1), **localedef**(1), **localeconv**(3), **newlocale**(3), **setlocale**(3), **strftime**(3), **strptime**(3), **use-locale**(3), **charmap**(5), **charsets**(7), **locale**(7), **unicode**(7), **utf−8**(7)