NAME

locale − Describes a locale definition file

DESCRIPTION

The **locale** definition files 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.

LC_CTYPE

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

There are the following keywords 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.

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_CYTPE*.

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.

There are the following keywords allowed:

collating-element

collating-symbol

The order-definition starts with a line:

order_start

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

order_end.

For more details see the sources in */usr/lib/nls/src* notably the examples **POSIX**, **Example** and **Example2**

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

LC_MONETARY

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

There are the following keywords 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 string that describes the formatting of numeric quantities.

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 set to **1** if the *currency_symbol* or *int_curr_symbol* should precede the formatted monetary quantity or set to **0** if the symbol succeeds the value.

p_sep_by_space
followed by an integer.

0 means that no space should be printed between the symbol and the value.

1 means that a space should be printed between the symbol and the value.

2 means that a space should be printed between the symbol and the sign string, if adjacent.

n_cs_precedes

0 - the symbol succeeds the value.

1 - the symbol precedes the value.

n_sep_by_space
An integer set to **0** if no space separates the *currency_symbol* or *int_curr_symbol* from the value for a negative monetary quantity, set to **1** if a space separates the symbol from the value and set to **2** if a space separates the symbol and the sign string, if adjacent.

p_sign_posn

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

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*.

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

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 string that describes the formatting of numeric quantities.

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

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 weekday names. The list starts with the first day of the week as specified by *week* (Sunday by default).

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

abmon followed by a list of abbreviated month names.

mon followed by a list of month names.

am_pm The appropriate representation of the **am** and **pm** strings.

d_t_fmt
The appropriate date and time format.

d_fmt The appropriate date format.

t_fmt The appropriate time format.

t_fmt_ampm
The appropriate time format when using 12h clock format.

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. Thus, countries using **19971130** should have local Sunday name as the first day in the *day* list, while countries using **19971201** should have Monday translation as the first item in the *day* list.

first_weekday (since glibc 2.2)
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.

first_workday (since glibc 2.2)
Number of the first working day from the *day* list.

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

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.

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

See the POSIX.2 standard for details.

FILES

/usr/lib/locale/ — database for the current locale setting of that category
/usr/lib/nls/charmap/* — charmap-files

CONFORMING TO

POSIX.2, ISO/IEC 14652.

BUGS

This manual page isn’t complete.

SEE ALSO

locale(1), **localedef**(1), **localeconv**(3), **setlocale**(3), **charmap**(5)

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.