NAME

locale − Description of multi-language support

SYNOPSIS

#include <locale.h>

DESCRIPTION

A locale is a set of language and cultural rules. These cover aspects such as language for messages, different character sets, lexicographic conventions, etc. A program needs to be able to determine its locale and act accordingly to be portable to different cultures.

The header <*locale.h*> declares data types, functions and macros which are useful in this task.

The functions it declares are **setlocale**(3) to set the current locale, and **localeconv**(3) to get information about number formatting.

There are different categories for local information a program might need; they are declared as macros. Using them as the first argument to the **setlocale**(3) function, it is possible to set one of these to the desired locale:

LC_COLLATE

This is used to change the behavior of the functions **strcoll**(3) and **strxfrm**(3), which are used to compare strings in the local alphabet. For example, the German sharp s is sorted as "ss".

LC_CTYPE

This changes the behavior of the character handling and classification functions, such as **isupper**(3) and **toupper**(3), and the multi-byte character functions such as **mblen**(3) or **wctomb**(3).

LC_MONETARY

changes the information returned by **localeconv**(3) which describes the way numbers are usually printed, with details such as decimal point versus decimal comma. This information is internally used by the function **strfmon**(3).

LC_MESSAGES

changes the language messages are displayed in and what an affirmative or negative answer looks like. The GNU C-library contains the **gettext**(3), **ngettext**(3), and **rpmatch**(3) functions to ease the use of these information. The GNU gettext family of functions also obey the environment variable **LANGUAGE** (containing a colon-separated list of locales) if the category is set to a valid locale other than **"C"**.

LC_NUMERIC

changes the information used by the **printf**(3) and **scanf**(3) family of functions, when they are advised to use the locale-settings. This information can also be read with the **localeconv**(3) function.

LC_TIME

changes the behavior of the **strftime**(3) function to display the current time in a locally acceptable form; for example, most of Europe uses a 24-hour clock versus the 12-hour clock used in the United States.

LC_ALL

All of the above.

If the second argument to **setlocale**(3) is empty string, **""**, for the default locale, it is determined using the following steps:

1. If there is a non-null environment variable **LC_ALL**, the value of **LC_ALL** is used.
2. If an environment variable with the same name as one of the categories above exists and is non-null, its value is used for that category.
3. If there is a non-null environment variable **LANG**, the value of **LANG** is used.

Values about local numeric formatting is made available in a *struct lconv* returned by the **localeconv**(3) function, which has the following declaration:

```
struct lconv {

  /* Numeric (non-monetary) information */

  char *decimal_point;     /* Radix character */
  char *thousands_sep;     /* Separator for digit groups to left
                              of radix character */
  char *grouping; /* Each element is the number of digits in a
                     group; elements with higher indices are
                     further left.  An element with value CHAR_MAX
                     means that no further grouping is done.  An
                     element with value 0 means that the previous
                     element is used for all groups further left. */

  /* Remaining fields are for monetary information */

  char *int_curr_symbol;   /* First three chars are a currency symbol
                              from ISO 4217.  Fourth char is the
                              separator.  Fifth char is '\0'. */
  char *currency_symbol;   /* Local currency symbol */
  char *mon_decimal_point; /* Radix character */
  char *mon_thousands_sep; /* Like thousands_sep above */
  char *mon_grouping;      /* Like grouping above */
  char *positive_sign;     /* Sign for positive values */
  char *negative_sign;     /* Sign for negative values */
  char  int_frac_digits;   /* International fractional digits */
  char  frac_digits;       /* Local fractional digits */
  char  p_cs_precedes;     /* 1 if currency_symbol precedes a
                              positive value, 0 if succeeds */
  char  p_sep_by_space;    /* 1 if a space separates currency_symbol
                              from a positive value */
  char  n_cs_precedes;     /* 1 if currency_symbol precedes a
                              negative value, 0 if succeeds */
  char  n_sep_by_space;    /* 1 if a space separates currency_symbol
                              from a negative value */
  /* Positive and negative sign positions:
     0 Parentheses surround the quantity and currency_symbol.
     1 The sign string precedes the quantity and currency_symbol.
     2 The sign string succeeds the quantity and currency_symbol.
     3 The sign string immediately precedes the currency_symbol.
     4 The sign string immediately succeeds the currency_symbol. */
  char  p_sign_posn;
  char  n_sign_posn;
};
```

CONFORMING TO

POSIX.1-2001.

The GNU gettext functions are specified in LI18NUX2000.

SEE ALSO

locale(1), **localedef**(1), **gettext**(3), **localeconv**(3), **ngettext**(3), **nl_langinfo**(3), **rpmatch**(3), **setlocale**(3), **strcoll**(3), **strfmon**(3), **strftime**(3), **strxfrm**(3)

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/.