## NAME

**localeconv** - natural language formatting for C

## LIBRARY

Standard C Library (libc, -lc)

## SYNOPSIS

**#include <locale.h>**

*struct lconv **
**localeconv**(*void*);

**#include <xlocale.h>**

*struct lconv **
**localeconv_l**(*locale_t locale*);

## DESCRIPTION

The **localeconv**() function returns a pointer to a structure which provides parameters for formatting numbers, especially currency values:

```
struct lconv {
        char    *decimal_point;
        char    *thousands_sep;
        char    *grouping;
        char    *int_curr_symbol;
        char    *currency_symbol;
        char    *mon_decimal_point;
        char    *mon_thousands_sep;
        char    *mon_grouping;
        char    *positive_sign;
        char    *negative_sign;
        char    int_frac_digits;
        char    frac_digits;
        char    p_cs_precedes;
        char    p_sep_by_space;
        char    n_cs_precedes;
        char    n_sep_by_space;
        char    p_sign_posn;
        char    n_sign_posn;
```

```
        char    int_p_cs_precedes;
        char    int_n_cs_precedes;
        char    int_p_sep_by_space;
        char    int_n_sep_by_space;
        char    int_p_sign_posn;
        char    int_n_sign_posn;
};
```

The individual fields have the following meanings:

*decimal_point* The decimal point character, except for currency values, cannot be an empty string.

*thousands_sep* The separator between groups of digits before the decimal point, except for currency values.

*grouping* The sizes of the groups of digits, except for currency values. This is a pointer to a vector of integers, each of size *char*, representing group size from low order digit groups to high order (right to left). The list may be terminated with 0 or CHAR_MAX. If the list is terminated with 0, the last group size before the 0 is repeated to account for all the digits. If the list is terminated with CHAR_MAX, no more grouping is performed.

*int_curr_symbol* The standardized international currency symbol.

*currency_symbol* The local currency symbol.

*mon_decimal_point* The decimal point character for currency values.

*mon_thousands_sep*
The separator for digit groups in currency values.

*mon_grouping* Like *grouping* but for currency values.

*positive_sign* The character used to denote nonnegative currency values, usually the empty string.

*negative_sign* The character used to denote negative currency values, usually a minus sign.

*int_frac_digits* The number of digits after the decimal point in an international-style currency value.

*frac_digits* The number of digits after the decimal point in the local style for currency values.

*p_cs_precedes* 1 if the currency symbol precedes the currency value for nonnegative values, 0 if it follows.

*p_sep_by_space* 1 if a space is inserted between the currency symbol and the currency value for nonnegative values, 0 otherwise.

*n_cs_precedes* Like *p_cs_precedes* but for negative values.

*n_sep_by_space* Like *p_sep_by_space* but for negative values.

*p_sign_posn* The location of the *positive_sign* with respect to a nonnegative quantity and the *currency_symbol*, coded as follows:

0 Parentheses around the entire string.
1 Before the string.
2 After the string.
3 Just before *currency_symbol*.
4 Just after *currency_symbol*.

*n_sign_posn* Like *p_sign_posn* but for negative currency values.

*int_p_cs_precedes* Same as *p_cs_precedes*, but for internationally formatted monetary quantities.

*int_n_cs_precedes* Same as *n_cs_precedes*, but for internationally formatted monetary quantities.

*int_p_sep_by_space*
Same as *p_sep_by_space*, but for internationally formatted monetary quantities.

*int_n_sep_by_space*
Same as *n_sep_by_space*, but for internationally formatted monetary quantities.

*int_p_sign_posn* Same as *p_sign_posn*, but for internationally formatted monetary quantities.

*int_n_sign_posn* Same as *n_sign_posn*, but for internationally formatted monetary quantities.

Unless mentioned above, an empty string as a value for a field indicates a zero length result or a value that is not in the current locale. A CHAR_MAX result similarly denotes an unavailable value.

The **localeconv_l**() function takes an explicit locale parameter. For more information, see xlocale(3).

## RETURN VALUES

The **localeconv**() function returns a pointer to a static object which may be altered by later calls to setlocale(3) or **localeconv**(). The return value for **localeconv_l**() is stored with the locale. It will remain valid until a subsequent call to freelocale(3). If a thread-local locale is in effect then the return value from **localeconv**() will remain valid until the locale is destroyed.

## ERRORS

No errors are defined.

## SEE ALSO

setlocale(3), strfmon(3)

## STANDARDS

The **localeconv**() function conforms to ISO/IEC 9899:1999 ("ISO C99").

## HISTORY

The **localeconv**() function first appeared in 4.4BSD.