Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'locale.1p' command

$ man locale.1p

LOCALE(1P) POSIX Programmer's Manual LOCALE(1P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

locale ? get locale-specific information

SYNOPSIS

locale [-a|-m]

locale [-ck] name...

DESCRIPTION

The locale utility shall write information about the current locale en? vironment, or all public locales, to the standard output. For the pur? poses of this section, a public locale is one provided by the implemen? tation that is accessible to the application.

When locale is invoked without any arguments, it shall summarize the current locale environment for each locale category as determined by the settings of the environment variables defined in the Base Defini? tions volume of POSIX.1?2017, Chapter 7, Locale.

When invoked with operands, it shall write values that have been as? signed to the keywords in the locale categories, as follows:

* Specifying a keyword name shall select the named keyword and the

category containing that keyword.

* Specifying a category name shall select the named category and all keywords in that category.

OPTIONS

The locale utility shall conform to the Base Definitions volume of POSIX.1?2017, Section 12.2, Utility Syntax Guidelines.

The following options shall be supported:

-a Write information about all available public locales. The available locales shall include POSIX, representing the POSIX locale. The manner in which the implementation determines what other locales are available is implementation-defined.

-c Write the names of selected locale categories; see the STDOUT section. The -c option increases readability when more than one category is selected (for example, via more than one key? word name or via a category name). It is valid both with and without the -k option.

-k Write the names and values of selected keywords. The imple? mentation may omit values for some keywords; see the OPERANDS section.

-m Write names of available charmaps; see the Base Definitions volume of POSIX.1?2017, Section 6.1, Portable Character Set.

OPERANDS

The following operand shall be supported:

name The name of a locale category as defined in the Base Defini? tions volume of POSIX.1?2017, Chapter 7, Locale, the name of a keyword in a locale category, or the reserved name charmap. The named category or keyword shall be selected for output. If a single name represents both a locale category name and a keyword name in the current locale, the results are unspeci? fied. Otherwise, both category and keyword names can be spec? ified as name operands, in any sequence. It is implementa? tion-defined whether any keyword values are written for the categories LC_CTYPE and LC_COLLATE.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of lo?
cale:

LANG Provide a default value for the internationalization vari?
ables that are unset or null. (See the Base Definitions vol?
ume of POSIX.1?2017, Section 8.2, Internationalization Vari?
ables for the precedence of internationalization variables
used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of
all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of
bytes of text data as characters (for example, single-byte as
opposed to multi-byte characters in arguments and input
files).

LC_MESSAGES
Determine the locale that should be used to affect the format
and contents of diagnostic messages written to standard er?
ror.

NLSPATH Determine the location of message catalogs for the processing
of LC_MESSAGES.

The application shall ensure that the LANG, LC_*, and NLSPATH environ?
ment variables specify the current locale environment to be written
out; they shall be used if the -a option is not specified.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The LANG variable shall be written first using the format:

"LANG=%s\n", <value>

If LANG is not set or is an empty string, the value is the empty

string.

If locale is invoked without any options or operands, the names and values of the LC_* environment variables described in this volume of POSIX.1?2017 shall be written to the standard output, one variable per line, and each line using the following format. Only those variables set in the environment and not overridden by LC_ALL shall be written using this format:

"%s=%s\n", <variable_name>, <value>

The names of those LC_* variables associated with locale categories de? fined in this volume of POSIX.1?2017 that are not set in the environ? ment or are overridden by LC_ALL shall be written in the following for? mat:

"%s=\"%s\"\n", <variable_name>, <implied value>

The <implied value> shall be the name of the locale that has been se? lected for that category by the implementation, based on the values in LANG and LC_ALL, as described in the Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables.

The <value> and <implied value> shown above shall be properly quoted for possible later reentry to the shell. The <value> shall not be quoted using double-quotes (so that it can be distinguished by the user from the <implied value> case, which always requires double-quotes).

The LC_ALL variable shall be written last, using the first format shown above. If it is not set, it shall be written as:

"LC_ALL=\n"

If any arguments are specified:

1. If the -a option is specified, the names of all the public locales shall be written, each in the following format:

   "%s\n", <locale name>

2. If the -c option is specified, the names of all selected categories shall be written, each in the following format:

   "%s\n", <category name>

   If keywords are also selected for writing (see following items), the category name output shall precede the keyword output for that

category.

If the -c option is not specified, the names of the categories shall not be written; only the keywords, as selected by the <name> operand, shall be written.

3. If the -k option is specified, the names and values of selected keywords shall be written. If a value is non-numeric and is not a compound keyword value, it shall be written in the following for? mat:

"%s=\"%s\"\n", <keyword name>, <keyword value>

If a value is a non-numeric compound keyword value, it shall either be written in the format:

"%s=\"%s\"\n", <keyword name>, <keyword value>

where the <keyword value> is a single string of values separated by <semicolon> characters, or it shall be written in the format:

"%s=%s\n", <keyword name>, <keyword value>

where the <keyword value> is encoded as a set of strings, each en? closed in double-quotation-marks, separated by <semicolon> charac? ters.

If the keyword was charmap, the name of the charmap (if any) that was specified via the localedef -f option when the locale was cre? ated shall be written, with the word charmap as <keyword name>.

If a value is numeric, it shall be written in one of the following formats:

"%s=%d\n", <keyword name>, <keyword value>

"%s=%c%o\n", <keyword name>, <escape character>, <keyword value>

"%s=%cx%x\n", <keyword name>, <escape character>, <keyword value>

where the <escape character> is that identified by the escape_char keyword in the current locale; see the Base Definitions volume of POSIX.1?2017, Section 7.3, Locale Definition.

Compound keyword values (list entries) shall be separated in the output by <semicolon> characters. When included in keyword values, the <semicolon>, <backslash>, double-quote, and any control charac? ter shall be preceded (escaped) with the escape character.

4. If the -k option is not specified, selected keyword values shall be written, each in the following format:

"%s\n", <keyword value>

If the keyword was charmap, the name of the charmap (if any) that was specified via the localedef -f option when the locale was cre? ated shall be written.

5. If the -m option is specified, then a list of all available charmaps shall be written, each in the format:

"%s\n", <charmap>

where <charmap> is in a format suitable for use as the option-argu? ment to the localedef -f option.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 All the requested information was found and output successfully.

>0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

If the LANG environment variable is not set or set to an empty value, or one of the LC_* environment variables is set to an unrecognized value, the actual locales assumed (if any) are implementation-defined as described in the Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables.

Implementations are not required to write out the actual values for keywords in the categories LC_CTYPE and LC_COLLATE; however, they must write out the categories (allowing an application to determine, for ex?

ample, which character classes are available).

EXAMPLES

In the following examples, the assumption is that locale environment variables are set as follows:

```
LANG=locale_x
LC_COLLATE=locale_y
```

The command locale would result in the following output:

```
LANG=locale_x
LC_CTYPE="locale_x"
LC_COLLATE=locale_y
LC_TIME="locale_x"
LC_NUMERIC="locale_x"
LC_MONETARY="locale_x"
LC_MESSAGES="locale_x"
LC_ALL=
```

The order of presentation of the categories is not specified by this volume of POSIX.1?2017.

The command:

```
LC_ALL=POSIX locale -ck decimal_point
```

would produce:

```
LC_NUMERIC
decimal_point="."
```

The following command shows an application of locale to determine whether a user-supplied response is affirmative:

```
printf 'Prompt for response: '
read response
if printf "%s\n$response" | grep -- -Eq "$(locale yesexpr)"
then
    affirmative processing goes here
else
    non-affirmative processing goes here
fi
```

RATIONALE

The output for categories LC_CTYPE and LC_COLLATE has been made imple?
mentation-defined because there is a questionable value in having a
shell script receive an entire array of characters. It is also diffi?
cult to return a logical collation description, short of returning a
complete localedef source.

The -m option was included to allow applications to query for the exis?
tence of charmaps. The output is a list of the charmaps (implementa?
tion-supplied and user-supplied, if any) on the system.

The -c option was included for readability when more than one category
is selected (for example, via more than one keyword name or via a cate?
gory name). It is valid both with and without the -k option.

The charmap keyword, which returns the name of the charmap (if any)
that was used when the current locale was created, was included to al?
low applications needing the information to retrieve it.

According to the Base Definitions volume of POSIX.1?2017, Section 6.1,
Portable Character Set, the standard requires that all supported lo?
cales must have the same encoding for <period> and <slash>, because
these two characters are used within the locale-independent pathname
resolution sequence. Therefore, it would be an error if locale -a
listed both ASCII and EBCDIC-based locales, since those two encodings
do not share the same representation for either <period> or <slash>.

Any system that supports both environments would be expected to provide
two POSIX locales, one in either codeset, where only the locales appro?
priate to the current environment can be visible at a time. In an XSI-
compliant implementation, the dd utility is the only portable means for
performing conversions between the two character sets.

FUTURE DIRECTIONS

None.

SEE ALSO

localedef

The Base Definitions volume of POSIX.1?2017, Section 6.1, Portable
Character Set, Chapter 7, Locale, Chapter 8, Environment Variables,
Section 12.2, Utility Syntax Guidelines

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Por? table Operating System Interface (POSIX), The Open Group Base Specifi? cations Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at http://www.opengroup.org/unix/online.html .

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.ker? nel.org/doc/man-pages/reporting_bugs.html .