-- --

IEEE P1003.2 Draft 11.2 – September 1991

4.15 `date` — Write the date and time

4.15.1 Synopsis

`date [−u] [+`*format*` ]`

4.15.2 Description

The `date` utility shall write the date and time to standard output. By default, the current date and time shall be written. If an operand beginning with `+` is specified, the output format of `date` shall be controlled by the field descriptors and other text in the operand.

4.15.3 Options

The `date` utility shall conform to the utility argument syntax guidelines described in 2.10.2.

The following option shall be supported by the implementation:

`−u` Perform operations as if the **TZ** environment variable was set to the string `UTC0`, or its equivalent historical value of `GMT0`. Otherwise, `date` shall use the time zone indicated by the **TZ** environment variable or the system default if that variable is not set. 2

4.15.4 Operands

When the format is specified, each field descriptor shall be replaced in the standard output by its corresponding value. All other characters shall be copied to the output without change. The output shall be always terminated with a `<newline>` character.

Field Descriptors

`%a`	Locale's abbreviated weekday name.
`%A`	Locale's full weekday name.
`%b`	Locale's abbreviated month name.
`%B`	Locale's full month name.
`%c`	Locale's appropriate date and time representation.
`%C`	Century (a year divided by 100 and truncated to an integer) as a decimal number (00-99).
`%d`	Day of the month as a decimal number (01-31).
`%D`	Date in the format *mm/dd/yy* .
`%e`	Day of the month as a decimal number (1-31 in a two-digit field with leading `<space>` fill).
`%h`	A synonym for `%b`.

`%H`	Hour (24-hour clock) as a decimal number (00-23).
`%I`	Hour (12-hour clock) as a decimal number (01-12).
`%j`	Day of the year as a decimal number (001-366).
`%m`	Month as a decimal number (01-12).
`%M`	Minute as a decimal number (00-59).
`%n`	A `<newline>` character.
`%p`	Locale's equivalent of either AM or PM.
`%r`	12-Hour clock time (01-12) using the *AM/PM* notation; in the POSIX Locale, this shall be equivalent to `"%I:%M:%S %p"`.
`%S`	Seconds as a decimal number (00-61).
`%t`	A `<tab>` character.
`%T`	24-Hour clock time (00-23) in the format *HH:MM:SS*.
`%U`	Week number of the year (Sunday as the first day of the week) as a decimal number (00-53).
`%w`	Weekday as a decimal number [0 (Sunday)-6].
`%W`	Week number of the year (Monday as the first day of the week) as a decimal number (00-53).
`%x`	Locale's appropriate date representation.
`%X`	Locale's appropriate time representation.
`%y`	Year (offset from `%C`) as a decimal number (00-99).
`%Y`	Year with century as a decimal number.
`%Z`	Time-zone name, or no characters if no time zone is determinable.
`%%`	A <percent-sign> character.

See the LC_TIME description in 2.5.2.5 for the field descriptor values in the POSIX Locale.

Modified Field Descriptors

Some field descriptors can be modified by the `E` and `O` modifier characters to indicate a different format or specification as specified in the LC_TIME locale description (see 2.5.2.5). If the corresponding keyword (see `era`, `era_year`, `era_d_fmt`, and `alt_digits` in 2.5.2.5) is not specified or not supported for the current locale, the unmodified field descriptor value shall be used.

`%Ec`	Locale's alternate appropriate date and time representation.
`%EC`	The name of the base year (period) in the locale's alternate representation.
`%Ex`	Locale's alternate date representation.
`%Ey`	Offset from `%EC` (year only) in the locale's alternate representation.
`%EY`	Full alternate year representation.
`%Od`	Day of month using the locale's alternate numeric symbols.
`%Oe`	Day of month using the locale's alternate numeric symbols.
`%OH`	Hour (24-hour clock) using the locale's alternate numeric symbols.

`%OI` Hour (12-hour clock) using the locale's alternate numeric symbols.

`%Om` Month using the locale's alternate numeric symbols.

`%OM` Minutes using the locale's alternate numeric symbols.

`%OS` Seconds using the locale's alternate numeric symbols.

`%OU` Week number of the year (Sunday as the first day of the week) using the locale's alternate numeric symbols.

`%Ow` Weekday as number in the locale's alternate representation (Sunday = 0).

`%OW` Week number of the year (Monday as the first day of the week) using the locale's alternate numeric symbols.

`%Oy` Year (offset from `%C`) in alternate representation.

4.15.5 External Influences

4.15.5.1 Standard Input

None.

4.15.5.2 Input Files

None.

4.15.5.3 Environment Variables

The following environment variables shall affect the execution of `date`:

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_CTYPE This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments).

LC_MESSAGES This variable shall determine the language in which messages should be written.

LC_TIME This variable shall determine the format and contents of date and time strings written by `date`.

TZ This variable shall specify the time zone in which the time and date are written, unless the `−u` option is specified. If the **TZ** variable is not set and the `−u` is not specified, an unspecified system default time zone is used.

4.15.5.4 Asynchronous Events

Default.

4.15.6 External Effects

4.15.6.1 Standard Output

When no formatting operand is specified, the output in the POSIX Locale shall be equivalent to specifying

```
date "+%a %b %e %H:%M:%S %Z %Y"
```

4.15.6.2 Standard Error

Used only for diagnostic messages.

4.15.6.3 Output Files

None.

4.15.7 Extended Description

None.

4.15.8 Exit Status

The `date` utility shall exit with one of the following values:

0 The date was written successfully.

>0 An error occurred.

4.15.9 Consequences of Errors

Default.

4.15.10 Rationale. *(This subclause is not a part of P1003.2)*

Examples, Usage

The option for setting the date and time was not included. It is normally a system administration option, which is outside the scope of POSIX.2.

The following are input/output examples of `date` used at arbitrary times in the POSIX Locale:

```
$ date
Tue Jun 26 09:58:10 PDT 1990

$ date "+DATE: %m/%d/%y%nTIME: %H:%M:%S"
DATE: 11/21/87
TIME: 13:36:16

$ date "+TIME: %r"
TIME: 01:36:32 PM
```

Field descriptors are of unspecified format when not in the POSIX Locale. Some of them can contain `<newline>`s in some locales, so it may be difficult to use the format shown in Standard Output for parsing the output of `date` in those locales.

The range of values for `%S` extends from 0 to 61 seconds to accommodate the occasional leap second or double leap second.

Although certain of the field descriptors in the POSIX Locale (such as the name of the month) are shown with initial capital letters, this need not be the case in other locales. Programs using these fields may need to adjust the capitalization if the output is going to be used at the beginning of a sentence.

The date string formatting capabilities are intended for use in Gregorian style calendars, possibly with a different starting year (or years). The `%x` and `%c` field descriptors, however, are intended for ‘‘local representation’’; these may be based on a different, non-Gregorian calendar.

The `%C` field descriptor was introduced to allow a fallback for the `%EC` (alternate year format base year); it can be viewed as the base of the current subdivision in the Gregorian calendar. A century is not calculated as an ordinal number; this standard was approved in century 19, not the twentieth (let’s hope). Both the `%Ey` and `%y` can then be viewed as the offset from `%EC` and `%C`, respectively.

The `E` and `O` modifiers modify the traditional field descriptors, so that they can always be used, even if the implementation (or the current locale) does not support the modifier.

The `E` modifier supports alternate date formats, such as the Japanese Emperor’s Era, as long as these are based on the Gregorian calendar system. Extending the `E` modifiers to other date elements may provide an implementation-specific extension capable of supporting other calendar systems, especially in combination with the `O` modifier.

The `O` modifier supports time and date formats using the locale's alternate numerical symbols, such as Kanji or Hindi digits, or ordinal number representation.

Non-European locales, whether they use Latin digits in computational items or not, often have local forms of the digits for use in date formats. This is not totally unknown even in Europe; a variant of dates uses Roman numerals for the months: the third day of September 1991 would be written as 3.IX.1991. In Japan, Kanji digits are regularly used for dates; in Arabic-speaking countries, Hindi digits are used. The `%d`, `%e`, `%H`, `%I`, `%m`, `%S`, `%U`, `%w`, `%W`, and `%y` field descriptors always return the date/time field in Latin digits (i.e., 0 through 9). The `%O` modifier was introduced to support the use for display purposes of non-Latin digits. In the LC_TIME category in `localedef`, the optional `alt_digits` keyword is intended for this purpose. As an example, assume the following (partial) `localedef` source:

```
alt_digits  "";"I";"II";"III";"IV";"V";"VI";"VII";"VIII" \
            "IX";"X";"XI";"XII"
d_fmt       "%e.%Om.%Y"
```

With the above date, the command

```
date "+x"
```

would yield ''3IX.1991.'' With the same `d_fmt`, but without the `alt_digits`, the command would yield ''3.9.1991.''

History of Decisions Made

Some of the new options for formatting are from the C Standard {7}. The `-u` option was introduced to allow portable access to Coordinated Universal Time (UTC). The string `GMT0` is allowed as an equivalent **TZ** value to be compatible with all of the systems using the BSD implementation, where this option originated.

The `%e` format field descriptor (adopted from System V) was added because the C Standard {7} descriptors did not provide any way to produce the historical default `date` output during the first nine days of any month.

4.16 dd — Convert and copy a file

4.16.1 Synopsis

`dd` [*operand* ...]

4.16.2 Description

The `dd` utility shall copy the specified input file to the specified output file with possible conversions using specific input and output block sizes. It shall read the input one block at a time, using the specified input block size; it then shall process the block of data actually returned, which could be smaller than the requested block size. It shall apply any conversions that have been specified and write the resulting data to the output in blocks of the specified output block size. If the `bs=`*expr* operand is specified and no conversions other than `sync` or `noerror` are requested, the data returned from each input block shall be written as a separate output block; if the read returns less than a full block and the `sync` conversion is not specified, the resulting output block shall be the same size as the input block. If the `bs=`*expr* operand is not specified, or a conversion other than `sync` or `noerror` is requested, the input shall be processed and collected into full-sized output blocks until the end of the input is reached.

The processing order shall be as follows:

(1) An input block is read.

(2) If the input block is shorter than the specified input block size and the `sync` conversion is specified, null bytes shall be appended to the input data up to the specified size. The remaining conversions and output shall include the pad characters as if they had been read from the input.

(3) If the `bs=`*expr* operand is specified and no conversion other than `sync` or `noerror` is requested, the resulting data shall be written to the output as a single block, and the remaining steps are omitted.

(4) If the `swab` conversion is specified, each pair of input data bytes shall be swapped. If there are an odd number of bytes in the input block, the results are unspecified.

(5) Any remaining conversions (`block`, `unblock`, `lcase`, and `ucase`) shall be performed. These conversions shall operate on the input data independently of the input blocking; an input or output fixed-length record may span block boundaries.

(6) The data resulting from input or conversion or both shall be aggregated into output blocks of the specified size. After the end of input is reached, any remaining output shall be written as a block without padding if `conv=sync` is not specified; thus the final output block may be shorter than the output block size.