datetime2 v1.1: date and time formats

Nicola L. C. Talbot
http://www.dickimaw-books.com/

2015-09-15

Abstract

The datetime2 package replaces the datetime package. Languages and regional variations are dealt with by the datetime2 language modules which are independently maintained and installed. Make sure that when you install datetime2 you also install the required datetime2 language modules.

Contents

1 Introduction

I wrote the original datetime package back in the 1990s as an alternative to the ukdate package, which had dropped out of some of the TeX distributions, so it was designed specifically for UK date formats.[1] However some users found the time formats useful and the ability to save dates for later use, so when babel came along I had a number of requests to make datetime compatible with babel so that the regional date formats were preserved but the other datetime functions could be used. Then PDFTEX came into existence and its `\pdfcreationdate` now provided a way of obtaining the seconds and time zone, which can't be obtained from TEX's `\time` primitive. Over time, the continual updates to the package has started to put a strain on the original naïve LATEX code of my first package, as it's been stretched well past its intended design.

The other problem with datetime is that some of the commands aren't expandable but some users want to be able to use them in expandable contexts (such as, for example, PDF bookmarks or writing a date stamp to an external file) or they want to be able to upper case the first letter if the date comes at the start of a sentence. This isn't an issue in English, as the weekday and month names are proper nouns and so automatically start with an upper case letter, regardless of where they appear in a sentence. Users who don't know the history and original purpose of the datetime package are puzzled as to why the defaults are all UK English or some styles were hard-coded, and some users are confused as to the ordering of the day, month and year parameters. In addition, some command names were incompatible with other date-related packages, but renaming those commands would break compatibility with older documents.

In order to address all these issues, a replacement package is necessary. Your old documents that use datetime should still be able to compile, but for your new documents, you may prefer the improved datetime2 package instead.

[1] Of course, `ukdatetime` would've been a better name, but the 8 dot 3 filename restriction was a concern back then.

2 Example Usage

```
\documentclass{article}
\usepackage{datetime2}
\begin{document}
This PDF was created on \today.
\end{document}
```

In the above example, the date is displayed in the form:

2015-03-01

This is the `default` style.

```
\documentclass{article}
\usepackage{datetime2}
\begin{document}
This PDF was created on \DTMnow.
\end{document}
```

In the above example, the full date, time and time zone is displayed in the form

2015-03-01 15:35:09Z

or

2015-04-01 08:55:39+01:00

unless you are using XeLaTeX in which case the seconds and time zone are omitted. (XeLaTeX doesn't provide this information.) Alternatively you can hide the seconds and zone using the package options showseconds=false and showzone=false. If you want UTC+0 to be displayed numerically instead of using a Z you can use the showisoZ=false package option.

```
\documentclass[british]{article}
\usepackage{babel}
\usepackage{datetime2}
\begin{document}
This PDF was created on \DTMnow.
\end{document}
```

This has the same default numerical output as the previous example, but there are now two additional styles available: `en-GB` and `en-GB-numeric`. The `\datebritish` command provided by babel is redefined to prevent babel from overriding your preferred date style. The regional style can be enabled with the useregional option.

```
\documentclass[british]{article}
\usepackage{babel}
\usepackage[useregional]{datetime2}
\begin{document}
This PDF was created on \DTMnow.
\end{document}
```

The full date, time and zone are displayed in the form

1st March 2015 3:35pm GMT

or

1st April 2015 8:55am BST

```
\documentclass[british]{article}
\usepackage{babel}
\usepackage[useregional=numeric]{datetime2}
\begin{document}
This PDF was created on \DTMnow.
\end{document}
```

The full date, time and zone are displayed in the form

1/3/2015 15:35:09 GMT

or

1/4/2015 8:55:39 BST

```
\documentclass[english]{article}
\usepackage{babel}
\usepackage[useregional=numeric]{datetime2}
\begin{document}
This PDF was created on \DTMnow.
\end{document}
```

The full date, time and zone are displayed in the form

2015-03-01 15:35:09Z

This is because no *regional dialect* has been specified. The language name `english` is ambiguous, so the `default` style is used.

```
\documentclass[english]{article}
\usepackage{babel}
\usepackage[useregional]{datetime2}
\begin{document}
This PDF was created on \today.
\end{document}
```

The date is now displayed as

March 1, 2015

since that's LaTeX's default behaviour for `\today`. For other languages, you should check the language module documentation to find out what happens when the region can't be determined from the language name.

```
\documentclass[british]{article}
\usepackage{babel}
\usepackage[useregional,showdow]{datetime2}

\DTMlangsetup[en-GB]{abbr}

\begin{document}
This PDF was created on \today.
\end{document}
```

The date is now displayed as

Sun 1st Mar 2015

The options used in `\DTMlangsetup` (such as `abbr`) are provided by the language modules and should be described in the module's documentation. Different languages may have different options so `abbr` may not be available for some of them. The showdow (show day of week) option is a package-wide option. You need to check the documentation to find out which styles check the showdow setting as not all of them do.

3 Displaying the Date and Time

A specific date can be displayed using:

\DTMdisplaydate

```
\DTMdisplaydate{⟨year⟩}{⟨month⟩}{⟨day⟩}{⟨dow⟩}
```

The format used to display the date is governed by the *display style*.

The arguments are all *numerical*: ⟨*year*⟩ is the year, ⟨*month*⟩ is the month number (starting from 1 for January), ⟨*day*⟩ is the day of the month and ⟨*dow*⟩ is the day of the week number starting from 0 for Monday. The day of week number may be `-1`, which indicates that the style should ignore it. (Some styles always ignore the day of week, regardless of its value.) This command is intended for use in expandable contexts (such as writing the date to another file or using the date in the bookmarks) and is used by `\today`. The date styles should ensure that any fragile content is protected. (This is why the ⟨*dow*⟩ isn't an optional argument otherwise the command wouldn't be expandable.)

Some styles may start the date with a word (such as the day of the week name or the month name). In English, proper nouns are capitalised regardless of where they appear in a sentence but some languages use lower case month or day of week names. In this event, if the initial letter needs to be capitalised then you can use:

\DTMDisplaydate

```
\DTMDisplaydate{⟨year⟩}{⟨month⟩}{⟨day⟩}{⟨dow⟩}
```

which is analogous to `\DTMdisplaydate`. Styles that are unaffected by this issue (for example, numerical or English dates) set `\DTMDisplaydate` to just `\DTMdisplaydate`. As with `\DTMdisplaydate` the style needs to ensure that any fragile content is protected in the event that `\DTMDisplaydate` is used in an expandable context. (Note that for this reason, I don't recommend the use of the commands provided by the mfirstuc package as they're not expandable.)

The current date is displayed using

\today

```
\today
```

This uses `\DTMdisplaydate` to format the date so it will match the currently selected date style. There's also a first letter upper case version that uses

\DTMDisplaydate:

\Today

```
\Today
```

If you use babel or polyglossia you must make sure you have the relevant datetime2 language modules installed. (See Section 6.) You also need to make sure that datetime2 is loaded *after* babel/polyglossia otherwise \today will be redefined so that it no longer uses \DTMdisplaydate.

As mentioned above, some styles allow the day of the week to be displayed. This requires the datetime2-calc package which will automatically be loaded if you set showdow in the datetime2 package option list or if you set showdow in \DTMsetup *in the preamble*. The package option calc will also load datetime2-calc or you can load it explicitly using \usepackage after datetime2 has been loaded. (You may use showdow=true in the document environment if the datetime2-calc package has been loaded in the preamble either explicitly or through the calc option.)

When datetime2-calc is loaded, it computes the current day of the week (using commands provided by the pgfcalendar package) which can then be used by \today or \Today. If datetime2-calc isn't loaded then neither \today nor \Today will display the day of the week, regardless of the current style.

If you would like a more convenient syntax and don't care about expansion, there is also a robust *non-expandable* command that can be used to display a particular date:

\DTMdate

```
\DTMdate{⟨date⟩}
```

As before there's also a capitalised version:

\DTMDate

```
\DTMDate{⟨date⟩}
```

In these cases the date should be provided as ⟨*YYYY*⟩-⟨*MM*⟩-⟨*DD*⟩ in the argument ⟨*date*⟩. For example:

```
\DTMdate{2015-03-23}
```

Note that hyphens must always be used, regardless of the separator options. Take care that the category code of the hyphen hasn't changed when you use this syntax.

The year ⟨*YYYY*⟩ can't be negative[1] in \DTMdate or \DTMDate. Use \DTMdisplaydate or \DTMDisplaydate instead.

These commands internally use \DTMdisplaydate and \DTMDisplaydate, respectively. If the datetime2-calc package has been loaded, the day of the week will be computed, otherwise the day of the week will be set to -1. Another benefit of the datetime2-calc package is that it allows additional formats permitted by the pgfcalendar package:

- ⟨*YYYY*⟩-⟨*MM*⟩-last (the last day of the given month).
- ⟨*YYYY*⟩-⟨*MM*⟩-⟨*DD*⟩+-⟨*n*⟩ (⟨*n*⟩ days before the given date).
- ⟨*YYYY*⟩-⟨*MM*⟩-⟨*DD*⟩+⟨*n*⟩ (⟨*n*⟩ days after the given date).
- ⟨*YYYY*⟩-⟨*MM*⟩-last+-⟨*n*⟩ (⟨*n*⟩ days before the last day of the given month).
- ⟨*YYYY*⟩-⟨*MM*⟩-last+⟨*n*⟩ (⟨*n*⟩ days after the last day of the given month).

See the pgfcalendar package for further details.

If you want to be able to use a date in an expandable context that can perform these calculations, consider first saving the date using one of the commands described in Section 4 and then use one of the expandable commands such as \DTMuse to display the date.

An error or unexpected results may occur if you try using one of these extended formats without loading the datetime2-calc package. An example that only works with datetime2-calc:

```
\DTMdate{2015-03-last}
```

An example that works with or without datetime2-calc:

```
\DTMdate{2015-03-31}
```

In this second case, you'll only notice a difference in the output if the style should show the day of the week.

The style of the date is the same as for \DTMdisplaydate and \DTMDisplaydate (which \DTMdate and \DTMDate internally use, as mentioned above).

A time can be displayed using

\DTMdisplaytime

\DTMdisplaytime{⟨*hour*⟩}{⟨*minute*⟩}{⟨*sec*⟩}

[1] Well, actually it can if you put it in braces and don't use datetime2-calc.

where the arguments are all numerical (using 24 hours). The *time style* currently in effect determines how the time is formatted. The command is designed to be used in an expandable context so the styles should take care to protect any fragile commands.

Note that this command doesn't display the time zone. To display the time zone, you need to use

\DTMdisplayzone

```
\DTMdisplayzone{⟨TZh⟩}{⟨TZm⟩}
```

where ⟨*TZh*⟩ is the hour offset and ⟨*TZm*⟩ is the minute offset. The display is governed by the *zone style*. Again, the style should protect any fragile commands in case this is used in an expandable context.

The current time (as set at the start of the document build) can be displayed using

\DTMcurrenttime

```
\DTMcurrenttime
```

This internally just uses `\DTMdisplaytime` and so is designed for use in an expandable context.

The current zone can be displayed using

\DTMcurrentzone

```
\DTMcurrentzone
```

This internally just uses `\DTMdisplayzone` and so is designed for use in an expandable context.

If the PDFTEX primitive `\pdfcreationdate` is defined, the current time information is obtained from that, which includes the seconds and time zone. LuaTEX also defines this command but XETEX doesn't, and in that case the only way to determine the current time is from TEX's `\time` primitive which only contains the number of minutes since midnight, which means that the seconds and time zone are unavailable. Therefore if XETEX is used, the showseconds and showzone options are automatically switched off.

There is also a non-expandable robust command to display the time:

\DTMtime

```
\DTMtime{⟨tm⟩}
```

where ⟨*tm*⟩ must be in the 24 hour format ⟨*hh*⟩:⟨*mm*⟩:⟨*ss*⟩ (colon-separated numerical arguments). Take care if you use babel with a language setting that

makes the colon character active. You will have to switch off the shorthands in order to use this command correctly.

The full date, time and zone (if available) can be displayed using

\DTMdisplay

```
\DTMdisplay{⟨year⟩}{⟨month⟩}{⟨day⟩}{⟨day of
week⟩}{⟨hh⟩}{⟨mm⟩}{⟨ss⟩}{⟨TZh⟩}{⟨TZm⟩}
```

The arguments are all numerical. The way the information is displayed in the document is governed by the *full style* (or *date-time style*). Typically the full style will redefine this command to use \DTMdisplaydate, \DTMdisplaytime and (optionally) \DTMdisplayzone. The showzone setting may govern whether or not to display the time zone (although a style may ignore this setting). The separators between the date and time and between the time and zone are governed by the style.

There is also an analogous version if capitalisation is required:

\DTMDisplay

```
\DTMDisplay{⟨year⟩}{⟨month⟩}{⟨day⟩}{⟨day of
week⟩}{⟨hh⟩}{⟨mm⟩}{⟨ss⟩}{⟨TZh⟩}{⟨TZm⟩}
```

Some styles may simply make this equivalent to \DTMdisplay. Other styles may use a similar format to \DTMdisplay but replace \DTMdisplaydate with \DTMDisplaydate.

The full current date, time and (optionally) zone can be displayed using:

\DTMnow

```
\DTMnow
```

which uses \DTMdisplay or

\DTMNow

```
\DTMNow
```

which uses \DTMDisplay.

4 Storing and Using Dates and Times

Date, time and zone information can be saved for later use. Note that the information is always saved numerically. The style is only applied when the information is later used. The commands that save the information are robust and not expandable. The commands that use the data are typically expandable although there may be some exceptions. Take care that the colon (`:`) and hyphen (`-`) characters haven't had their normal category code changed. (For example, through babel's shortcuts.) In the commands below, the ⟨*name*⟩ (no active characters) is a name that uniquely identifies the information.

Dates are saved using

`\DTMsavedate`

```
\DTMsavedate{⟨name⟩}{⟨date⟩}
```

where ⟨*date*⟩ is in the same format as for `\DTMdate`. As with `\DTMdate` (and `\DTMDate`) the format can be extended with the datetime2-calc package. If you want to access the day of week, you must make sure that datetime2-calc has been loaded *before you save the date.* (Remember that the calc and showdow package options will automatically load datetime2-calc.) If datetime2-calc has been loaded, the day of week number will be calculated and saved. Whether or not it is displayed in the document when the date is later used depends on the settings when the date is displayed not when it's saved.

This command will override any previously defined date saved with this ⟨*name*⟩. If a time or zone hasn't been defined with this ⟨*name*⟩, the time and zone elements will all be set to 0 otherwise they will remain unchanged.

Note that you can't have a negative year in ⟨*date*⟩. There's an alternative command you can use instead that doesn't try parsing ⟨*date*⟩:

`\DTMsavenoparsedate`

```
\DTMsavenoparsedate{⟨name⟩}{⟨YYYY⟩}{⟨MM⟩}{⟨DD⟩}{⟨dow⟩}
```

The day of week ⟨*dow*⟩ may be `-1` if unknown. This command doesn't calculate the day of week, even if datetime2-calc has been loaded.

Times are saved using

`\DTMsavetime`

```
\DTMsavetime{⟨name⟩}{⟨time⟩}
```

where the ⟨*time*⟩ is in the same format as for `\DTMtime`.

This command will override any previously defined time saved with this ⟨*name*⟩. If a date or zone hasn't been defined with this ⟨*name*⟩, the date and zone elements will all be set to 0 (or `-1` for the day of week) otherwise they will remain unchanged.

Times and zone are saved using

`\DTMsavetimezn`

```
\DTMsavetimezn{⟨name⟩}{⟨time and zone⟩}
```

where the ⟨*time and zone*⟩ is in the form

⟨*hh*⟩:⟨*mm*⟩:⟨*ss*⟩ ⟨*TZh*⟩:⟨*TZm*⟩

(Note the space between the seconds and the hour offset.)

This command will override any previously defined time and zone saved with this ⟨*name*⟩. If a date hasn't been defined with this ⟨*name*⟩, the year, month and day will be set to zero and the day of the week to `-1` otherwise they will remain unchanged.

All date, time and zone information can be saved at the same time using:

`\DTMsavetimestamp`

```
\DTMsavetimestamp{⟨name⟩}{⟨data⟩}
```

where ⟨*data*⟩ is in the format:

⟨*YYYY*⟩-⟨*MM*⟩-⟨*DD*⟩T⟨*hh*⟩:⟨*mm*⟩:⟨*ss*⟩⟨*zone*⟩

The ⟨*zone*⟩ may either be `Z` or in the form ⟨*TZh*⟩`:`⟨*TZm*⟩ (for example, `-03:00` or `-3:0`). This will override any date, time or zone data previously saved with this ⟨*name*⟩.

The current date and time can be saved using:

`\DTMsavenow`

```
\DTMsavenow{⟨name⟩}
```

There is also a command that can be used to save the modification date of a file, but it's not available for some TEX engines:

`\DTMsavefilemoddate`

```
\DTMsavefilemoddate{⟨name⟩}{⟨file name⟩}
```

where ⟨*file name*⟩ is the name of the file (remember to use forward slashes / for the directory divider). If you build your document using PDFLaTeX, this command will use the PDFTeX primitive `\pdffilemoddate`. If you use LuaTeX this command will attempt to use `os.date` but it uses `%z` for the time zone, which may not work on some operating systems. If you use XeLaTeX this command will generate a warning and will assume a date of 0000-00-00T00:00:00Z.

The above commands are all localised to the current scope. If the data is required after the end of the scope, you can make the assignments global using:

`\DTMmakeglobal`

```
\DTMmakeglobal{⟨name⟩}
```

For example:

```
\DTMsavenow{mydate}\DTMmakeglobal{mydate}
```

A previously saved date can be displayed using the current style with

`\DTMusedate`

```
\DTMusedate{⟨name⟩}
```

This just uses `\DTMdisplaydate`. An error will occur if ⟨*name*⟩ hasn't been defined. Alternatively for the capitalised version:

`\DTMUsedate`

```
\DTMUsedate{⟨name⟩}
```

which uses `\DTMDisplaydate` instead.

A previously saved time can be displayed using the current style with

`\DTMusetime`

```
\DTMusetime{⟨name⟩}
```

This just uses `\DTMdisplaytime`. An error will occur if ⟨*name*⟩ hasn't been defined.

A previously saved zone can be displayed using the current style with

`\DTMusezone`

```
\DTMusezone{⟨name⟩}
```

This just uses `\DTMdisplayzone`. An error will occur if ⟨*name*⟩ hasn't been defined.

The entire date, time and zone can be displayed in the current style with

`\DTMuse`

```
\DTMuse{⟨name⟩}
```

This uses \DTMdisplay. An error will occur if ⟨*name*⟩ hasn't been defined. Alternatively,

\DTMUse

```
\DTMUse{⟨name⟩}
```

will use \DTMDisplay instead.

You can determine if a given ⟨*name*⟩ has been defined using

\DTMifsaveddate

```
\DTMifsaveddate{⟨name⟩}{⟨true⟩}{⟨false⟩}
```

The individual numerical elements can be fetched using one of the following commands. These don't check if the given data identified by ⟨*name*⟩ has been defined and will expand to \relax if the name isn't recognised.

\DTMfetchyear

```
\DTMfetchyear{⟨name⟩}
```

This expands to the year.

\DTMfetchmonth

```
\DTMfetchmonth{⟨name⟩}
```

This expands to the month number.

\DTMfetchday

```
\DTMfetchday{⟨name⟩}
```

This expands to the day of the month.

\DTMfetchdow

```
\DTMfetchdow{⟨name⟩}
```

This expands to the day of the week number (-1 if unknown).

\DTMfetchhour

```
\DTMfetchhour{⟨name⟩}
```

This expands to the hour.

\DTMfetchminute

```
\DTMfetchminute{⟨name⟩}
```

This expands to the minute.

\DTMfetchsecond

```
\DTMfetchsecond{⟨name⟩}
```

This expands to the second.

\DTMfetchTZhour

```
\DTMfetchTZhour{⟨name⟩}
```

This expands to the hour offset.

\DTMfetchTZminute

```
\DTMfetchTZminute{⟨name⟩}
```

This expands to the minute offset.

5 Styles

If you want to just change the date style use:

\DTMsetdatestyle

```
\DTMsetdatestyle{⟨name⟩}
```

where ⟨*name*⟩ identifies the style. For example:

```
\DTMsetdatestyle{iso}
```

This will just change the date style (\DTMdisplaydate and \DTMDisplaydate), not the time or zone styles. Note that \DTMdisplay typically uses \DTMdisplaydate so this will also change the date element of \DTMdisplay.

If you want to just change the time style use:

\DTMsettimestyle

```
\DTMsettimestyle{⟨name⟩}
```

where ⟨*name*⟩ identifies the style. For example:

```
\DTMsettimestyle{iso}
```

This will just change the time style (\DTMdisplaytime), not the date or zone styles. Note that \DTMdisplay typically uses \DTMdisplaytime so this will also change the time element of \DTMdisplay.

If you want to just change the zone style use:

\DTMsetzonestyle

```
\DTMsetzonestyle{⟨name⟩}
```

where ⟨*name*⟩ identifies the style. For example:

```
\DTMsetzonestyle{iso}
```

This will just change the zone style (\DTMdisplayzone), not the date or time styles. Note that \DTMdisplay typically uses \DTMdisplayzone so this will also change the zone element of \DTMdisplay.

If you want to change the full style use:

\DTMsetstyle

```
\DTMsetstyle{⟨name⟩}
```

where ⟨*name*⟩ identifies the style. For example:

```
\DTMsetstyle{iso}
```

Note that in this case this does more than simply

```
\DTMsetdatestyle{iso}\DTMsettimestyle{iso}\DTMsetzonestyle{iso}
```

as it also changes `\DTMdisplay` and `\DTMDisplay`. If the style ⟨*name*⟩ is only a partial style, a warning will be issued for any partial styles that aren't defined for the given name as well as a warning for the undefined full style. An error will occur if there are neither partial nor full styles with the given ⟨*name*⟩.

The predefined styles listed in Section 5.1.1 are all *full styles*. This means that they change the date, time, zone and full format, so any of them can be used in `\DTMsetdatestyle`, `\DTMsettimestyle`, `\DTMsetzonestyle` or `\DTMsetstyle`. However it's possible for a style to be only a *partial style*, such as those described in Section 5.1.2.

For example, if `foo` is a date style and a time style but isn't a zone style or a full style then you can use

```
\DTMsetdatestyle{foo}
```

and

```
\DTMsettimestyle{foo}
```

but you can't use `\DTMsetzonestyle`. You can use

```
\DTMsetstyle{foo}
```

but this will now only be equivalent to

```
\DTMsetdatestyle{foo}\DTMsettimestyle{foo}
```

and while `\DTMdisplay` and `\DTMDisplay` will typically use these date and time settings, the way that the date, time and zone are arranged will be governed by the full style setting that was already in effect before the date and time style changed.

The style changes are all local and so are affected by the current scope.

5.1 Predefined Styles

The base datetime2 package provides a number of predefined numerical styles. Section 5.1.1 lists the full styles, which can be used with `\DTMsetstyle`, `\DTMsetdatestyle`, `\DTMsettimestyle` and `\DTMsetzonestyle`. Section 5.1.2 lists the predefined (partial) times styles, which can be used with `\DTMsettimestyle` and `\DTMsetstyle`.

5.1.1 Full Styles

The following are predefined full styles that are provided by the base datetime2 package. Additional styles are available through the language modules (see Section 6).

`default` The `default` style displays the date in the form

⟨*YYYY*⟩⟨*YMsep*⟩⟨*MM*⟩⟨*MDsep*⟩⟨*DD*⟩

where the month ⟨*MM*⟩ and day of the month ⟨*DD*⟩ numbers are formatted as two digits. The separators ⟨*YMsep*⟩ and ⟨*MDsep*⟩ default to a hyphen but can be changed using the options yearmonthsep, monthdaysep or datesep (either through the package options or using `\DTMsetup`).

The time is displayed in the form:

⟨*hh*⟩⟨*HMsep*⟩⟨*mm*⟩⟨*MSsep*⟩⟨*ss*⟩

where the hour, month and seconds are formatted as two digits. The final ⟨*MSsep*⟩⟨*ss*⟩ is omitted if the option showseconds has been set to `false`. The separators ⟨*HMsep*⟩ and ⟨*MSsep*⟩ default to a colon (`:`) but these may be changed using the options hourminsep, minsecsep or datetimesep.

The zone is displayed in form

⟨*TZh*⟩⟨*HMsep*⟩⟨*TZm*⟩

or just `Z` if the option showisoZ is set to `true` and both ⟨*TZh*⟩ and ⟨*TZm*⟩ are zero. The separator ⟨*HMsep*⟩ is the same as used for the time format. The final ⟨*HMsep*⟩⟨*TZm*⟩ is omitted if the option showzoneminutes is set to `false`. The hour offset ⟨*TZh*⟩ is formatted as two digits proceeded by either + or - and the minute offset is formatted as two digits. Note that since one of the main purposes of this package is to provide expandable date commands that can be used to write information to external files, no attempt is made to convert the hyphen - (for negative offsets) into a minus sign. If you want it rendered correctly in your document, consider placing the time zone command in math mode and adjust the separators as necessary.

The full style is in the form

⟨*date*⟩⟨*DTsep*⟩⟨*time*⟩⟨*TZsep*⟩⟨*zone*⟩

The ⟨*date*⟩⟨*DTsep*⟩ part is omitted if the option showdate is set to `false`, and the ⟨*TZsep*⟩⟨*zone*⟩ part is omitted if the option showzone is set to `false`. The separator between the date and time ⟨*DTsep*⟩ defaults to `\space` but may be changed using the datetimesep option. The separator between the time and zone ⟨*TZsep*⟩ defaults to nothing but may be changed using the timezonesep option.

`iso` The `iso` style is like the `default` style but the separators can't be changed. The separators used in the date format are fixed as hyphens and the separators used in the time and zone formats are fixed as colons. In the full format, the separator between the date and time is fixed as `T` and there's no separator between the time and zone. The only options that can change the `iso` style are showseconds, showdate, showzone, showzoneminutes and showisoZ.

`yyyymd` This is like the `default` style except that the month and date aren't forced into a two-digit format.

`ddmmyyyy` This is like the `default` style except that the date is formatted in the reverse order

⟨*DD*⟩⟨*MDsep*⟩⟨*MM*⟩⟨*YMsep*⟩⟨*YYYY*⟩

The day and month are displayed as two-digits and the separators are as for the `default` style. The options that modify the `default` style similarly modify this style.

`dmyyyy` This is like the `ddmmyyyy` style except that it doesn't force the day and month into a two-digit format. The options that modify the `default` style similarly modify this style.

`dmyy` This is like the `dmyy` style except that it only displays the final two digits of the year. The options that modify the `default` style similarly modify this style.

`mmddyyyy` This is like the `ddmmyyyy` style except the day and month numbers are reversed. The separator between the month and day is still given by the monthdaysep or datesep options. The separator between the day and year is given by the dayyearsep or datesep options.

`mdyyyy` This is like the `mmddyyyy` style except that it doesn't force the day and month into a two-digit format.

`mdyy` This is like the `mdyyyy` style except that the year only has the final two digits displayed.

`pdf` This formats the date, time and zone so that the full style is in the form required by the date settings in `\pdfinfo`. The date format is

D:⟨*YYYY*⟩⟨*MM*⟩⟨*DD*⟩

where the month and day numbers are displayed as two digits.

The time format is

⟨*hh*⟩⟨*mm*⟩⟨*ss*⟩

where the numbers are displayed as two digits.

The zone format is

⟨*hh*⟩'⟨*mm*⟩'

or Z for zero time offset if the option showisoZ is used. (The showisoZ option is the only option that modifies the `pdf` style.) The hour and minutes are displayed as two digits where the hour has the sign present (either + or -).

The full style is a concatenation of the date, time and zone.

D:⟨*YYYY*⟩⟨*MM*⟩⟨*DD*⟩⟨*hh*⟩⟨*mm*⟩⟨*ss*⟩⟨*hh*⟩'⟨*mm*⟩'

5.1.2 Time Styles

There's only one predefined time (partial) style provided by the base datetime2 package. This style can be used to override the time format part of full styles. For example, to use the `default` full style with the hmmss time style:

```
\DTMsetstyle{default}\DTMsettimestyle{hmmss}
```

`hmmss` The `hmmss` style is like the time style provided by the full `default` style except that the hour isn't forced into two digits.

5.1.3 Zone Styles

The following are predefined zone (partial) styles that are provided by the base datetime2 package. These styles can be used to override the zone format part of full styles. For example, to use the `default` full style with the `map` zone style:

```
\DTMsetstyle{default}\DTMsetzonestyle{map}
```

`map` The `map` style uses `\DTMusezonemapordefault` to display the mapping, if one exists, or use the default style, if a mapping doesn't exist. For example:

```
\DTMNatoZoneMaps
\DTMsetzonestyle{map}
```

This first defines the NATO mappings and then switches to the `map` style.

`hhmm` The `hhmm` style displays the time zone in the form

⟨*TZh*⟩⟨*HMsep*⟩⟨*TZm*⟩

where ⟨*HMsep*⟩ is given by the hourminsep option. This style honours the showzoneminutes option but ignores the showisoZ option. The hour is always prefixed by the sign.

5.2 Defining New Styles

A new date style can be defined using:

`\DTMnewdatestyle`

```
\DTMnewdatestyle{⟨name⟩}{⟨definition⟩}
```

This defines a partial style that should only modify `\DTMdisplaydate` and `\DTMDisplaydate`. The redefinition of these commands should be placed in ⟨*definition*⟩.

A new time style can be defined using:

`\DTMnewtimestyle`

```
\DTMnewtimestyle{⟨name⟩}{⟨definition⟩}
```

This defines a partial style that should only modify `\DTMdisplaytime`. The redefinition should be placed in ⟨*definition*⟩.

A new zone style can be defined using:

`\DTMnewzonestyle`

```
\DTMnewzonestyle{⟨name⟩}{⟨definition⟩}
```

This defines a partial style that should only modify `\DTMdisplayzone`. The redefinition should be placed in ⟨*definition*⟩.

A new full style can be defined using:

`\DTMnewstyle`

\DTMnewstyle{⟨*name*⟩}{⟨*date style definition*⟩}{⟨*time style definition*⟩}{⟨*zone style definition*⟩}{⟨*full style definition*⟩}

This does

\DTMnewdatestyle{⟨*name*⟩}{⟨*date style definition*⟩}

\DTMnewtimestyle{⟨*name*⟩}{⟨*time style definition*⟩}

\DTMnewzonestyle{⟨*name*⟩}{⟨*zone style definition*⟩}

and finally ⟨*full style definition*⟩ should redefine \DTMdisplay and \DTMDisplay.

Remember to use a double-hash to reference the parameters (##1, ##2 etc) within ⟨*definition*⟩ in all the above. In each case ⟨*name*⟩ is the label identifying the style and shouldn't contain active characters.

There are some helper commands provided that you might want to use in the style definitions.

\DTMtwodigits

\DTMtwodigits{⟨*number*⟩}

This displays ⟨*number*⟩ so that it has *only* two digits. Unlike LaTeX's \two@digits this will check for a negative number and will trim a number whose absolute value is greater than 100. This command is expandable.

\DTMcentury

\DTMcentury{⟨*year*⟩}

This converts ⟨*year*⟩ to the century. If ⟨*year*⟩ is negative it does:

-\DTMcentury{-⟨*year*⟩}

Example:

\DTMcentury{1945}

expands to 20 (not 19). Note that

\DTMcentury{1900}

expands to 19.

\DTMdivhundred

\DTMdivhundred{⟨*number*⟩}

This expands to $\lfloor$⟨*number*⟩/100$\rfloor$ (integer division by 100 rounded down). For example:

```
\DTMdivhundred{1945}
```

expands to 19 and

```
\DTMdivhundred{1900}
```

expands to 19.

\DTMtexorpdfstring

\DTMtexorpdfstring{⟨*TEX*⟩}{⟨*PDF*⟩}

If hyperref is loaded, this is equivalent to `\texorpdfstring` otherwise it just does the first argument and ignores the second. (The check for hyperref is deferred until the start of the document environment, so it doesn't matter if hyperref is loaded after datetime2.) This command may be used to provide alternative text to use if the date/time/zone is displayed in the PDF bookmarks.

\DTMsep

\DTMsep{⟨*tag*⟩}

This accesses the value of the ⟨*tag*⟩`sep` base package option. (Not the language module options.) For example

```
\DTMsep{yearmonth}
```

expands to the value supplied by the yearmonthsep package option.

\DTMusezonemap

\DTMusezonemap{⟨*TZh*⟩}{⟨*TZm*⟩}

This expands to the time zone abbreviation or `\relax` if no mapping has been set for the given time zone.

You can define a time zone mapping using

\DTMdefzonemap

\DTMdefzonemap{⟨*TZh*⟩}{⟨*TZm*⟩}{⟨*abbr*⟩}

For example

```
\DTMdefzonemap{00}{00}{GMT}
\DTMdefzonemap{01}{00}{BST}
```

Note that datetime2 doesn't know anything about daylight saving, so this is only really designed for dates and times in a specific location. This overwrites any previous mapping for this time zone.

The base datetime2 package provides

\DTMNatoZoneMaps

```
\DTMNatoZoneMaps
```

This defines the military/NATO mappings from A (Alpha time) to Z (Zulu time). You can use this command if you want these time zones (but remember to set an appropriate time zone style that uses the zone mappings).

The language modules may provide mappings that are enabled when you switch to that style. For example, the `en-GB` language module provides the mapzone option which, if set to `true`, will map +00:00 to GMT and +01:00 to BST. See the documentation for the language module for further details.

\DTMclearmap

```
\DTMclearmap{⟨TZh⟩}{⟨TZm⟩}
```

Clears the time zone mapping. The regional time zone styles should use

\DTMresetzones

```
\DTMresetzones
```

before applying any regional mappings. This defaults to nothing which means that any mappings previously defined by other styles won't be cleared. You can redefine this command if you want to clear any mappings that aren't relevant for other regions.

You can test if a mapping is defined using

\DTMhaszonemap

```
\DTMhaszonemap{⟨TZh⟩}{⟨TZm⟩}{⟨true⟩}{⟨false⟩}
```

This will do ⟨*true*⟩ if there is a mapping defined for that time zone or ⟨*false*⟩ otherwise.

\DTMusezonemapordefault

```
\DTMusezonemapordefault{⟨TZh⟩}{⟨TZm⟩}
```

This will use the mapping if its defined otherwise it will expand to the format ⟨*TZh*⟩⟨*HMsep*⟩⟨*TZm*⟩ where ⟨*HMsep*⟩⟨*TZm*⟩ is omitted if the option showzoneminutes is set to `false`. The separator ⟨*HMsep*⟩ is as given by the hourminsep option. (The showisoZ option isn't used here so UTC+00:00 will be displayed as +00:00 or +00 if there's no mapping.)

Here's an example of a simple date style that just displays the year and month as two digits but uses the yearmonthsep option:

```
\newdatestyle
 {mmyy}% label
 {% definitions
   \renewcommand*{\DTMdisplaydate}[4]{%
     \DTMtwodigits{##2}\DTMsep{yearmonth}\DTMtwodigits{##1}}%
   \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
 }
```

If you want to distribute your new styles, just put the definitions in a package and upload it to CTAN. For example (replace `mystylename` with something more appropriate, and also change the date in the `\ProvidesPackage` line):

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{mystylename}[2014/03/24 v1.0]
\RequirePackage{datetime2}

% style definitions here

\endinput
```

Save the file as `mystylename.sty`, add some documentation about the style (or styles) provided and read the instructions at http://www.ctan.org/upload and http://www.ctan.org/file/help/ctan/CTAN-upload-addendum. The upload location for additions to the datetime2 package (either for packages defining new styles or for language modules) should be `/macros/latex/contrib/datetime2-contrib/mystylename` (remember to replace `mystylename` as appropriate).

6 Multi-Lingual Support

If you want to use datetime2 with babel or polyglossia, make sure you load babel/polyglossia *before* you load datetime2 otherwise their `\date`⟨*language*⟩ will overwrite `\datetime`'s definition of `\today`. *Additionally* you need to make sure you install the relevant datetime2 language modules. These modules are automatically loaded, if required, by datetime2 but only if they are already installed. Remember that if you use XeLaTeX you won't have the seconds or time zone available for the current date and time.

If the required language modules aren't installed or datetime2 is loaded before babel/polyglossia then datetime2's definition of `\today` will be overridden and may no longer match the currently selected date style.

Each language module defines a textual style (where the month is displayed as a word) for that language or region which can be used in the argument of `\DTMsetstyle`, `\DTMsetdatestyle`, `\DTMsettimestyle` or `\DTMsetzonestyle`. The language module may also define a numeric style. In the ambiguous cases where the language name alone doesn't indicate the region (for example, `english` instead of `UKenglish` or `USenglish`) the module should use the `default` numeric style (see Section 5.1.1).

The textual style provided by the module will automatically be set using `\DTMsetstyle` *if the useregional option is set to* `text`. By default useregional is `false`, unless the language/region is passed via the datetime2 package option list. (The useregional option is unaffected if the setting is passed through the document class option list.) The numeric style provided by the module will automatically be set if the useregional option is set to `numeric`. See the descriptions for the useregional and style options in Section 7.

Be careful not to mix the language/region options between the document class option list and the babel/polyglossia interface. For example:

```
\documentclass[en-GB]{article}
\usepackage[canadien,british]{babel}
```

This will prevent the tracklang package from picking up the babel setting and it will only detect the `en-GB` option. Use only the document class options or only the babel package option list or duplicate *all* the babel package options with analogous tracklang options in the document class. For example

```
\documentclass[canadien,british]{article}
\usepackage{babel}
```

or

```
\documentclass{article}
\usepackage[canadien,british]{babel}
```

or

```
\documentclass[fr-CA,en-GB]{article}
\usepackage[canadien,british]{babel}
```

Language modules may be used without babel or polyglossia. For example:

```
\documentclass{article}
\usepackage[en-GB]{datetime2}
\begin{document}
\today
\end{document}
```

If you have more than one language or region you will need to switch styles using `\DTMsetstyle` etc:

```
\documentclass{article}
\usepackage[en-GB,en-CA]{datetime2}
\begin{document}
\DTMsetstyle{en-GB}\today.
\DTMsetstyle{en-CA}\today
\end{document}
```

If you want to change the number separators for the *regional* numeric styles, you need to use `\DTMlangsetup`. If you want to change the number separators for the base datetime2 predefined numeric styles (see Section 5.1) then you need to use `\DTMsetup` or the package options. You therefore need to use `\DTMsetup` for the ambiguous regionless language numeric settings since they just use the `default` style. Check the module documentation to find out if the `default` style is used.

Examples of use:

1. Language option specified through the document class and picked up by tracklang (which is loaded by datetime2). This setting is also picked up by babel which is loaded before datetime2.

```
\documentclass[british]{article}

\usepackage{babel}
\usepackage{datetime2}

\begin{document}
\today
\end{document}
```

   The date is displayed in the default format 2015-03-01.

   In this case, the `en-GB` language module is loaded which defines the text style `en-GB` and the numeric style `en-GB-numeric`. Since useregional hasn't been set, `\today` uses datetime2's `default` numerical format. If babel was loaded after datetime2, the babel's hook management system would overwrite datetime2's definition of `\today` so that it no longer used `\DTMdisplaydate`. A similar result is obtained if in the above example babel is replaced with polyglossia (where the language is set in the document class option).

   You can change the useregional setting either through datetime2's package options or using `\DTMsetup` however it will only have an effect during the module loading (when the value is changed via the package option) and when `\date`⟨*language*⟩ is used.

   For example, in the document below, the date is displayed using the `default` numeric format because useregional has been changed *after* babel uses `\datebritish` to set the language at the start of the document.

```
\documentclass[british]{article}

\usepackage{babel}
\usepackage{datetime2}

\begin{document}
\DTMsetup{useregional}
\today
\end{document}
```

   So here `\today` again displays the date in the form 2015-03-01.

   If the setting is moved to the preamble:

```
\documentclass[british]{article}

\usepackage{babel}
\usepackage{datetime2}

\DTMsetup{useregional}
\begin{document}
\today
\end{document}
```

then the useregional setting is checked at the beginning of the document when babel uses `\datebritish`. So in this case `\today` will display the date in the form 1st March 2015.

2. Language setting specified through babel's package option list:

```
\documentclass{article}

\usepackage[british]{babel}
\usepackage{datetime2}

\begin{document}
\today
\end{document}
```

This has the same result as placing `british` in the document class option list, so the date is again displayed in the default format 2015-03-01 but, as in the previous example, the `en-GB` and `en-GB-numeric` styles are both defined if required.

However a problem occurs if babel is replaced by polyglossia:

```
\documentclass{article}

\usepackage{fontspec}
\usepackage{polyglossia}
\setdefaultlanguage[variant=uk]{english}

\usepackage{datetime2}

\begin{document}
\today
\end{document}
```

In this case tracklang is unable to pick up the variant and can only detect the root language, so it will load the generic `english` module instead of the `en-GB` module. This means that the `en-GB` and `en-GB-numeric` styles are no longer available. However, since useregional is `false` the date is still displayed using the `default` numeric style in the form 2015-03-01.

3. As mentioned above neither babel nor polyglossia are required in order to use the datetime2 language modules. You can simply supply the language setting in the package option list:

```
\documentclass{article}

\usepackage[british]{datetime2}

\begin{document}
\today
\end{document}
```

This additionally sets useregional=true (since the language is in the package option list not the document class option list) so the date produced by `\today` now uses the `en-GB` date style in the form 1st March 2015.

4. The regional numeric format can be used instead if useregional is set to `numeric`:

```
\documentclass{article}

\usepackage[british,useregional=numeric]{datetime2}

\begin{document}
\today
\end{document}
```

This now displays the date in the form 1/3/2015.

Many of the language options have synonyms. In addition to the babel synonyms (such as british or UKenglish) the tracklang package provides options in ISO form, such as en-GB. Note that the style name provided by each language module is independent of the package option used to select that style. So regardless of whether you use british, UKenglish or en-GB, the text style name is `en-GB` and the numeric style name is `en-GB-numeric`. If just english is used, the text style name is `english` but the numeric style is `default`.

Languages where the region is automatically implied, such as scottish, provide a text style with the root language name (`scottish` in this instance) and a numeric style in the form ⟨*language*⟩`-numeric` (such as `scottish-numeric`). Note that `irish` has regionless styles `irish` and `irish-numeric` but also has regional styles `ga-IE` and `ga-IE-numeric` (for the Republic of Ireland) and `ga-GB` and `ga-GB-numeric` (for Northern Ireland). In this case the regionless style has a numeric style instead of using the `default` style since both `ga-IE-numeric` and `ga-GB-numeric` are the same so there's no ambiguity. The only difference in the three modules `datetime2-irish`, `datetime2-ga-IE` and `datetime2-ga-GB` is the time zone mappings.

The language modules may provide additional settings that can be applied using

\DTMlangsetup

\DTMlangsetup[⟨*language list*⟩]{⟨*options*⟩}

where ⟨*language list*⟩ is a comma-separated list of language modules that have been loaded (such as `en-GB,en-US`) and ⟨*options*⟩ is a ⟨*key*⟩=⟨*value*⟩ list of options. If ⟨*language list*⟩ is omitted, then the list of all loaded language modules is assumed. The modules may also provided user commands to further customise the style. These settings should all be described in the module's documentation, which should be accessible via `texdoc datetime2-`⟨*language*⟩ where ⟨*language*⟩ is the root language name in lower case (such as `english`).

Note that although I maintain the datetime2 English language module, I don't maintain the other modules. If you have an issue with one of the other modules, please contact the module maintainer. If there is no maintainer, feel free to volunteer to take over the maintenance (send me a message). If there's no module for your language you can create your own module and upload it to CTAN in the `/macros/latex/contrib/datetime2-contrib/datetime2-`⟨*language*⟩ directory.

You can use the English or Irish modules as a template for a language with multiple regions. Just download the English source files `datetime2-english.dtx` and `datetime2-english.ins` or the Irish source files `datetime2-irish.dtx` and `datetime2-irish.ins` from CTAN and make the appropriate modifications. Alternatively you can use the Scottish module as a template for a single-region language. Just download the Scottish source files `datetime2-scottish.dtx` and `datetime2-scottish.ins` from CTAN and make the appropriate modifications. (Don't forget to provide a README file.)

Each language module should be in a file named `datetime2-`⟨*lang*⟩`.ldf` where ⟨*lang*⟩ is the language name or ⟨*language ISO code*⟩-⟨*country ISO code*⟩. (See the tracklang documentation for further details of the naming scheme.)

If you want to provide a language module don't assume all users want to use the same input encoding or babel shortcuts as you. Use LaTeX commands for non-ASCII characters (and remember to use `\protect` where necessary).

As an addendum to the above warning, LuaTeX and XeTeX support UTF-8 characters without the need to make them active, so I recommend you provide two files: one with the LaTeX commands, such as `\c`, for (PDF)LaTeX users, and one with UTF-8 characters for LuaLaTeX and XeLaTeX users. For example, the `fr-FR` module could start with:

```
\ProvidesDateTimeModule{fr-FR}
```

```
\RequirePackage{ifxetex,ifluatex}

\ifxetex
 \RequireDateTimeModule{french-utf8}
\else
 \ifluatex
   \RequireDateTimeModule{french-utf8}
 \else
   \RequireDateTimeModule{french-ascii}
 \fi
\fi
```

This helps provide fully expandable dates for LuaLaTeX and XeLaTeX users. (See the Scottish or Irish modules.)

7 Package Options

The following package options are provided. Most of these are ⟨*key*⟩=⟨*value*⟩ options, unless stated otherwise.

Settings that govern the predefined numerical styles (not including the fixed styles `iso` and `pdf`):

yearmonthsep This sets the separator between the year and month for the big-endian and little-endian styles. Default: - (hyphen).

monthdaysep This sets the separator between the month and day. Default: - (hyphen).

dayyearsep This sets the separator between the day and year for the middle-endian styles. Default: - (hyphen).

datesep This sets the separators between the day and month, the month and year, and the day and year. Example:

```
\usepackage[datesep=/]{datetime2}
```

This is equivalent to:

```
\usepackage[yearmonthsep=/,monthdaysep=/,dayyearsep=/]{datetime2}
```

hourminsep This sets the separator between the hour and minute. (Both for the time and for the zone.) Default: : (colon).

minsecsep This sets the separator between the minute and seconds. Default: : (colon).

timesep This sets the separators between the hour and minute and between the minute and seconds. Example:

```
\usepackage[timesep=:]{datetime2}
```

This is equivalent to:

```
\usepackage[hourminsep=:,minsecsep=:]{datetime2}
```

The following settings are used by the predefined numerical styles when displaying the full date, time and zone (excluding the fixed styles `iso` and `pdf`) with commands that use `\DTMdisplay` or `\DTMDisplay`.

datetimesep Sets the separator between the date and time. Default: `\space`.

timezonesep Sets the separator between the time and zone. Default: empty.

The following settings are used by the predefined styles and may also be used by the language modules.

showseconds Boolean key to determine whether or not to show the seconds when the time is displayed. The `iso` style honours this setting but the `pdf` style ignores it. Default: `true` unless XeTeX is used.

showdate Boolean key to determine whether or not to show the date with commands that use `\DTMdisplay` or `\DTMDisplay`. (Some styles may ignore this.) The `iso` style honours this setting but the `pdf` style ignores it. Default: `true` unless XeTeX is used.

showzone Boolean key to determine whether or not to show the time zone with commands that use `\DTMdisplay` or `\DTMDisplay`. (Some styles may ignore this.) The `iso` style honours this setting but the `pdf` style ignores it. Default: `true` unless XeTeX is used.

showzoneminutes Boolean key to determine whether or not to show the zone offset minutes. The `iso` style honours this setting but the `pdf` style ignores it. This setting is ignored if showzone is `false`. Default: `true`.

showisoZ Boolean key to determine whether or not to show UTC+00:00 as `Z` instead of numerically. This option may be ignored by zone styles that use the zone mappings. If you want all the time zones in military form, you can use `\DTMNatoZoneMaps` to set up the time zone abbreviations and then use a zone style that uses the mappings. Default: `true`.

General settings:

useregional Allowed values: `false`, `text` or `numeric`. You may also use `num` as an abbreviation for `numeric`. If no value is supplied `text` is assumed.

This key determines whether or not to *use* the loaded regional settings and, if the regional setting should be used, it determines whether the text style (months as words) or numeric style should be used. If you haven't loaded babel or polyglossia, this key only has an effect when used as a package option. If you have loaded one of those packages, the change comes into effect at module load time and whenever `\date`⟨*language*⟩ is used (which includes at the beginning of the document environment). If you want to switch the style at any other time, you need to use `\DTMsetstyle` but unless useregional=false the next instance of `\date`⟨*language*⟩ will change the style.

Note that setting this option to `false` doesn't prevent the modules from being loaded. It just prevents them from automatically setting the style

and prevents \date⟨*language*⟩ from changing the style if you are using babel or polyglossia.

The default value is false unless the language or region is passed to the datetime2 package option list. However, using style will set useregional to false.

Examples:

```
\documentclass[british]{article}
\usepackage{babel}
\usepackage{datetime2}
```

In the above useregional is false.

```
\documentclass{article}
\usepackage[british]{datetime2}
```

In the above useregional is text.

```
\documentclass{article}
\usepackage[british,style=iso]{datetime2}
```

In the above useregional is false. (The british option implements usenumerical=text but the style option then implements usenumerical=false.)

```
\documentclass{article}
\usepackage[style=iso,british]{datetime2}
```

In the above useregional is text. (The style option implements usenumerical=false but the british option then implements usenumerical=text.)

style Sets the current style using \DTMsetstyle when the datetime2 package has finished loading. This also sets useregional=false but that setting can be overridden later in the option list.

Default value: empty (use the default style or the regional style, according to the value of useregional).

This key isn't available in \DTMsetup. Use \DTMsetstyle instead.

calc Load the datetime2-calc package. This will allow the day of week to be computed and allow you to use the pgfcalendar offset style date formats in commands like \DTMdate as well as defining the commands described in Section 8. This option doesn't take a value. It can't be switched off. This option can't be used in \DTMsetkeys. The default is to not load datetime2-calc.

showdow This is a boolean key that determines whether or not to show the day of week in styles that support this. Note that showdow=true will automatically load datetime2-calc so

```
\usepackage[showdow]{datetime2}
```

is equivalent to

```
\usepackage[showdow,calc]{datetime2}
```

This option may be used in `\DTMsetup`, but if you attempt to switch it on in the document environment you'll get an error if the datetime2-calc package hasn't been loaded. Default: `false`.

warn This is a boolean key. If true (default) datetime2 warnings will be displayed. If false, the warnings will be suppressed. Default: `true`.

Any additional option passed to the datetime2 package (not through `\DTMsetup`) will be considered a tracklang option and will be passed to `\TrackPredefinedDialect`. (See the tracklang documentation for further details of that command.)

Apart from calc, style and the regional options, all the above options can also be set using:

`\DTMsetup`

`\DTMsetup{`⟨*option list*⟩`}`

The language modules may additionally provide options which can be set using:

`\DTMlangsetup`

`\DTMlangsetup[`⟨*module list*⟩`]{`⟨*option list*⟩`}`

This will set the ⟨*option list*⟩ for each module listed in ⟨*module list*⟩. Unknown options will generate a warning rather than an error message. The default value of ⟨*module list*⟩ is the list of all loaded modules.

Example:

```
\documentclass{article}
\usepackage[british]{datetime2}
\DTMlangsetup{mapzone}
```

The module list here is `english-base,en-GB` and since the `english-base` doesn't have a `mapzone` option, this will result in a warning:

```
Package datetime2 Warning: Region 'english-base' has ignored
(datetime2)                the following settings:
(datetime2)                mapzone
```

You can either ignore the warning or use the optional argument to exclude the english-base module:

```
\documentclass{article}
\usepackage[british]{datetime2}
\DTMlangsetup[en-GB]{mapzone}
```

Note that some modules may have options with the same name as the above listed package options, but the keys are defined in different families (see xkeyval documentation) so you need to take care to use \DTMsetup for package-wide settings and \DTMlangsetup for the module-specific settings.

For example, the datesep package option described above is used by the predefined numerical styles but regional modules that provide their own numerical styles may use a different date separator that matches their region so they may also provide a datesep option independent of the base datesep option.

Examples:

```
\documentclass[british]{article}
\usepackage[datesep=.]{datetime2}
\begin{document}
\today
\end{document}
```

The above displays the date in the form 2015.09.15 since the default style is in use and datesep is used as a package option.

```
\documentclass[british]{article}
\usepackage{datetime2}
\DTMsetup{datesep=.}
\begin{document}
\today
\end{document}
```

The above displays the date in the form 2015.09.15 since the default style is in use and datesep is used in \DTMsetup.

```
\documentclass[british]{article}
\usepackage{datetime2}
\DTMlangsetup{datesep=.}
\begin{document}
\today
\end{document}
```

The above displays the date in the form 2015-09-15 since the default style is in use but datesep is used in \DTMlangsetup, which only influences the en-GB-numeric style, which isn't the current style.

```
\documentclass[british]{article}
\usepackage[useregional=numeric]{datetime2}
```

```
\DTMlangsetup{datesep=.}
\begin{document}
\today
\end{document}
```

The above displays the date in the form 15.9.2015 since the `en-GB-numeric` style is in use and datesep is used in `\DTMlangsetup`.

```
\documentclass[british]{article}
\usepackage[useregional]{datetime2}
\DTMlangsetup{datesep=.}
\begin{document}
\today
\end{document}
```

The above displays the date in the form 15th September 2015 since the `en-GB` style is in use and datesep is used in `\DTMlangsetup`, which only influences the `en-GB-numeric` style.

```
\documentclass[british]{article}
\usepackage[useregional=numeric]{datetime2}
\DTMsetup{datesep=.}
\begin{document}
\today
\end{document}
```

The above displays the date in the form 15/9/2015 since the `en-GB-numeric` style is in use but datesep is used in `\DTMsetup` which influences the base predefined numeric styles not the regional styles.

8 The datetime2-calc Package

The datetime2-calc package can be loaded after datetime2 in the usual way:

```
\usepackage{datetime2}
\usepackage{datetime2-calc}
```

or using the calc package option to datetime2:

```
\usepackage[calc]{datetime2}
```

or by using showdow=true:

```
\usepackage[showdow]{datetime2}
```

This package loads the pgfcalendar package which provides a way of computing the day of week from a given date. Once datetime2-calc has been loaded, you can enable or disable the weekday in dates where the style supports this, but note that not all styles support this, even if the datetime2-calc package has been loaded.

As with the commands in Section 4, the commands described below that save date/time information will *overwrite* any previously defined date/time data with the same identifying ⟨*name*⟩. However, they may only overwrite specific elements of the data (for example, just the year, month, day and day of week elements) and leave the other elements unchanged. Where the remaining elements are undefined they'll be set to zero, except for the day of week element, which will be set to -1.

In addition to enabling the weekday calculations, the datetime2-calc package also provides the following commands:

`\DTMsavejulianday`

`\DTMsavejulianday{`⟨*name*⟩`}{`⟨*number*⟩`}`

This uses `\pgfcalendarjuliantodate` to obtain the year, month and day from the given Julian day number and uses `\pgfcalendarjuliantoweekday` to obtain the day of week and then saves it. The date can later be used with commands such as `\DTMuse{`⟨*name*⟩`}` described in Section 4. Example:

```
\DTMsavejulianday{mydate}{2457023}
```

`\DTMsaveddatetojuliandate`

`\DTMsaveddatetojulianday{`⟨*name*⟩`}{`⟨*register*⟩`}`

This uses \pgfcalendardatetojulian to convert a previously saved date (identified by ⟨*name*⟩) to a Julian day. The result is stored in ⟨*register*⟩ which should be a count register (not a LaTeX counter name). Example:

```
\newcount\myct
\DTMsaveddatetojulianday{mydate}{\myct}
```

\DTMsaveddateoffsettojuliandate

\DTMsaveddateoffsettojulianday{⟨*name*⟩}{⟨*offset*⟩}{⟨*register*⟩}

This is like the previous command but converts the date obtained by incrementing the saved date with ⟨*offset*⟩. The result is stored in ⟨*register*⟩. This is equivalent to

\pgfcalendardatetojulian{⟨*y*⟩-⟨*m*⟩-⟨*d*⟩+⟨*offset*⟩}{⟨*register*⟩}

where ⟨*y*⟩, ⟨*m*⟩ and ⟨*d*⟩ are the year, month and day fetched from the saved date. A negative ⟨*offset*⟩ indicates an earlier date. Example:

```
\DTMsaveddateoffsettojulianday{mydate}{2}{\myct}
```

or

```
\DTMsaveddateoffsettojulianday{mydate}{-7}{\myct}
```

\DTMifdate

\DTMifdate{⟨*name*⟩}{⟨*test*⟩}{⟨*true*⟩}{⟨*false*⟩}

This is just a convenient interface to \pgfcalendarifdate for a saved date (identified by ⟨*name*⟩). The remaining arguments are the same as the final three arguments of \pgfcalendarifdate. Note that the `equals`, `at least`, `at most` and `between` keywords available in ⟨*test*⟩ need to be in the format specified by the pgf manual, but remember that you can use commands like \DTMfetchyear. Example:

```
Is \texttt{mydate2} (\DTMusedate{mydate2}) before
\texttt{mydate} (\DTMusedate{mydate})?
\DTMifdate
 {mydate2}
 {at most=
   \DTMfetchyear{mydate}-\DTMfetchmonth{mydate}-\DTMfetchday{mydate}}
 {yes}{no}.
```

\DTMsaveddatediff

\DTMsavedatediff{⟨*name1*⟩}{⟨*name2*⟩}{⟨*register*⟩}

Computes the difference (in days) between two saved dates and stores the result in the given count register. The first date is identified by ⟨*name1*⟩ and the second date is identified by ⟨*name2*⟩. The dates are converted to their respective Julian day numbers ⟨*J1*⟩ and ⟨*J2*⟩ and the result is given by ⟨*J1*⟩−⟨*J2*⟩.

Note that the time and zone are not taken into account, even if they were provided when the dates were stored.

Example:

```
\DTMsaveddatediff{mydate}{mydate2}{\myct}

\DTMusedate{mydate} is
\ifnum\myct=0
 the same day as
\else
  \ifnum\myct<0
   \number-\myct\space day\ifnum\myct<-1s\fi\space before
  \else
   \number\myct\space day\ifnum\myct>1s\fi\space after
  \fi
\fi
\DTMusedate{mydate2}.
```

The pgfcalendar package also provides a variety of useful date-related commands. See the documentation (part of the pgf manual) for further details. Note that the language modules don't use pgfcalendar month and weekday names as the pgfcalendar package isn't loaded by default.

The datetime2-calc package also provides commands that convert a datetime instance into Zulu[1] time (UTC+00:00).

\DTMsaveaszulutime

```
\DTMsaveaszulutime{⟨name⟩}{⟨YYYY⟩}{⟨MM⟩}{⟨DD⟩}{⟨hh⟩}{⟨mm⟩}
{⟨ss⟩}{⟨TZh⟩}{⟨TZm⟩}
```

This converts the given datetime instance into UTC+00:00 and saves the result. You can then use the date with commands like \DTMuse described in Section 4. The ⟨*name*⟩ argument is the label identifying the saved data. The other arguments are all numbers. Example:

```
\DTMsaveaszulutime{mydate}{2014}{6}{3}{20}{45}{0}{6}{0}
```

\DTMtozulu

```
\DTMtozulu{⟨name1⟩}{⟨name2⟩}
```

[1] That's Zulu as in the NATO alphabet representation of the letter Z.

Uses `\DTMsaveaszulutime` to convert the datetime stored in ⟨*name1*⟩ and saves it to ⟨*name2*⟩. Example:

```
\DTMsavetimestamp{mydate}{2014-05-01T03:55:00 -06:00}
Original date: \DTMuse{mydate}.

\DTMtozulu{mydate}{mydate2}
UTC+00:00: \DTMuse{mydate2}.
```

The above produces (using the `default` format):

Original date: 2014-05-01 03:55:00-06:00.
UTC+00:00: 2014-05-01 09:55:00Z.

9 The Code

9.1 datetime2.sty code

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{datetime2}[2015/09/15 v1.1 (NLCT) date and time formats]
```

Use tracklang to find out what languages have been loaded.

```
\RequirePackage{tracklang}
```

Also require etoolbox.

```
\RequirePackage{etoolbox}
```

Need xkeyval for ⟨*key*⟩=⟨*value*⟩ interface.

```
\RequirePackage{xkeyval}[2006/11/18]
```

`\dtm@yearmonthsep` Separator between year and month for numeric dates.

```
\newcommand*{\dtm@yearmonthsep}{-}
```

`\dtm@monthdaysep` Separator between month and day for numeric dates.

```
\newcommand*{\dtm@monthdaysep}{-}
```

`\dtm@dayyearsep` Separator between day and year for numeric middle-endian dates.

```
\newcommand*{\dtm@dayyearsep}{-}
```

`\dtm@hourminsep` Separator between the hour and minute for times.

```
\newcommand*{\dtm@hourminsep}{:}
```

`\dtm@minsecsep` Separator between the minute and second for times.

```
\newcommand*{\dtm@minsecsep}{:}
```

`\dtm@timezonesep` Separator between the date and time.

```
\newcommand*{\dtm@datetimesep}{\space}%
```

`\dtm@timezonesep` Separator between the time and time zone.

```
\newcommand*{\dtm@timezonesep}{}
```

`datesep` Set year/month and month/day separator.

```
\define@key{datetime2.sty}{datesep}{%
  \renewcommand*{\dtm@yearmonthsep}{#1}%
  \renewcommand*{\dtm@monthdaysep}{#1}%
  \renewcommand*{\dtm@dayyearsep}{#1}%
}
```

yearmonthsep Set year/month separator.

```
\define@key{datetime2.sty}{yearmonthsep}{%
  \renewcommand*{\dtm@yearmonthsep}{#1}%
}
```

monthdaysep Set month/day separator.

```
\define@key{datetime2.sty}{monthdaysep}{%
  \renewcommand*{\dtm@monthdaysep}{#1}%
}
```

dayyearsep Set day/year separator for middle-endian dates.

```
\define@key{datetime2.sty}{dayyearsep}{%
  \renewcommand*{\dtm@dayyearsep}{#1}%
}
```

timesep Set hour/minute and minute/second separator.

```
\define@key{datetime2.sty}{timesep}{%
  \renewcommand*{\dtm@hourminsep}{#1}%
  \renewcommand*{\dtm@minsecsep}{#1}%
}
```

hourminsep Set hour/minute separator.

```
\define@key{datetime2.sty}{hourminsep}{%
  \renewcommand*{\dtm@hourminsep}{#1}%
}
```

minsecsep Set minute/second separator.

```
\define@key{datetime2.sty}{minsecsep}{%
  \renewcommand*{\dtm@minsecsep}{#1}%
}
```

timezonesep Set separator between the time and the time zone (used in \DTMnow).

```
\define@key{datetime2.sty}{timezonesep}{%
  \renewcommand*{\dtm@timezonesep}{#1}%
}
```

datetimesep Set separator between the date and the time (used in \DTMnow).

```
\define@key{datetime2.sty}{datetimesep}{%
  \renewcommand*{\dtm@datetimesep}{#1}%
}
```

showseconds Boolean key to determine whether or not to show the seconds.

```
\define@boolkey{datetime2.sty}[DTM]{showseconds}[true]{}
```

showdate Boolean key to determine whether or not to show the date in \DTMdisplay and \DTMDisplay.

```
\define@boolkey{datetime2.sty}[DTM]{showdate}[true]{}
\DTMshowdatetrue
```

`showzone` Boolean key to determine whether or not to show the time zone in `\DTMdisplay` and `\DTMDisplay`.

```
\define@boolkey{datetime2.sty}[DTM]{showzone}[true]{}
```

`showisoZ` Boolean key to determine whether or not to use `Z` instead of `+00:00` for UTC in the `default`, `iso` or `pdf` styles. (Other styles may also use this.)

```
\define@boolkey{datetime2.sty}[DTM]{showisoZ}[true]{}
\DTMshowisoZtrue
```

Switch off seconds and time zone if `\pdfcreationdate` isn't defined, otherwise switch on.

```
\ifdef\pdfcreationdate
{%
  \DTMshowsecondstrue
  \DTMshowzonetrue
}%
{%
  \DTMshowsecondsfalse
  \DTMshowzonefalse
}%
```

`showzoneminutes` Boolean key to determine whether or not to show the time zone minutes. (If `\DTMshowzonefalse` then this option is irrelevant.)

```
\define@boolkey{datetime2.sty}[DTM]{showzoneminutes}[true]{}
\DTMshowzoneminutestrue
```

`\DTMifcaseregional` `\DTMifcaseregional{⟨false⟩}{⟨text⟩}{⟨numeric⟩}`

Determines if the user wants the language modules to set the regional format. The first argument ⟨*false*⟩ indicates that they don't want the regional format set, the second argument ⟨*text*⟩ indicates they want the textual format (e.g. 1st March, 2015 or March 1, 2005) and the third argument ⟨*numeric*⟩ indicates they want the numeric format (e.g. 1/3/2015 or 3/1/2015). A change in the setting will only have an affect when the module is loaded and when `\date`⟨*language*⟩ is used to set the style. The default is false.

```
\newcommand*{\DTMifcaseregional}[3]{#1}
```

`useregional` Setting to determine whether or not to use the regional settings (if any are loaded).

```
\define@choicekey{datetime2.sty}{useregional}[\val\nr]%
 {false,text,numeric,num}[text]%
 {%
   \ifcase\nr\relax
     \renewcommand*{\DTMifcaseregional}[3]{##1}%
```

```
    \or
      \renewcommand*{\DTMifcaseregional}[3]{##2}%
    \or
      \renewcommand*{\DTMifcaseregional}[3]{##3}%
    \or
      \renewcommand*{\DTMifcaseregional}[3]{##3}%
    \fi
}
```

\@dtm@setusecalc

```
\newcommand*{\@dtm@setusecalc}{%
  \renewcommand*{\@dtm@usecalc}{\RequirePackage{datetime2-calc}}%
}
```

\@dtm@usecalc

```
\newcommand*{\@dtm@usecalc}{}
```

Disable attempt to load datetime2-calc in the document.

```
\AtBeginDocument{%
  \@ifpackageloaded{datetime2-calc}%
  {%
    \renewcommand*{\@dtm@setusecalc}{}%
  }%
  {%
    \renewcommand*{\@dtm@setusecalc}{%
      \PackageError{datetime2}{You must load 'datetime2-calc'
      package to use option 'showdow'}{Try one of the following:^^J
      pass 'calc' option to 'datetime2' package when you load it^^J
      or move 'showdow' option to 'datetime2' package option list^^J
      or move \string\DTLsetup\space to the preamble.}%
    }%
  }%
}
```

calc This option will load the datetime2-calc which uses the pgfcalendar package to compute the day of week and offsets. The package is loaded at the end of this one.

```
\DeclareOptionX{calc}{\@dtm@setusecalc}
```

showdow Boolean key to determine whether or not to show the day of week for the styles that can show the day of week. If this is switched on, then datetime2-calc is required. If this key is set later in the document with \DTMsetup, then the datetime2-calc package must previously be loaded for it to have an effect.

```
\define@boolkey{datetime2.sty}[DTM]{showdow}[true]{%
  \ifDTMshowdow \@dtm@setusecalc \fi
}
\DTMshowdowfalse
```

\@dtm@warning Warning messages.

```
\newcommand*{\@dtm@warning}[1]{%
  \if@dtm@warn
    \PackageWarning{datetime2}{#1}%
  \fi
}
```

warn Allow user to suppress package warnings.

```
\define@boolkey{datetime2.sty}[@dtm@]{warn}[true]{}
\@dtm@warntrue
```

\@dtm@initialstyle

```
\newcommand*{\@dtm@initialstyle}{}
```

style Set the style. This automatically sets useregional=false.

```
\define@key{datetime2.sty}{style}{%
  \renewcommand*{\@dtm@initialstyle}{#1}%
  \ifstrempty{#1}%
  {}%
  {%
    \renewcommand*{\DTMifcaseregional}[3]{##1}%
  }%
}
```

Pass any unknown options to tracklang. This will automatically switch the useregional setting to text.

```
\DeclareOptionX*{%
  \TrackPredefinedDialect{\CurrentOption}%
  \renewcommand*{\DTMifcaseregional}[3]{#2}%
}
```

Process options passed to this package:

```
\ProcessOptionsX
```

Disable calc option. If it's required, just load datetime2-calc with \usepackage.

```
\disable@keys{datetime2.sty}{calc}
```

Disable style option. If it's required, just use \DTMsetup.

```
\disable@keys{datetime2.sty}{style}
```

Provide a way to set options after package has been loaded.

\DTMsetup

```
\newcommand*{\DTMsetup}[1]{%
 \def\@dtm@usecalc{}%
 \setkeys{datetime2.sty}{#1}%
 \@dtm@usecalc
}
```

9.1.1 Defaults

This section sets up the defaults.

`\@dtm@parsedate` Parse date in the format ⟨*year*⟩-⟨*month*⟩-⟨*day*⟩. The arguments are expanded. (This is redefined by datetime2-calc.)

```
\def\@dtm@parsedate#1-#2-#3\@dtm@endparsedate{%
  \edef\@dtm@year{\number#1}%
  \edef\@dtm@month{\number#2}%
  \edef\@dtm@day{\number#3}%
  \def\@dtm@dow{-1}%
}
```

`\@dtm@parsetime` Define command to parse time in the format ⟨*h*⟩:⟨*m*⟩:⟨*s*⟩. The results are stored in `\@dtm@hour`, `\@dtm@minute` and `\@dtm@second`. The arguments are expanded.

```
\def\@dtm@parsetime#1:#2:#3\@dtm@endparsetime{%
  \edef\@dtm@hour{\number#1}%
  \edef\@dtm@minute{\number#2}%
  \edef\@dtm@second{\number#3}%
}
```

`\@dtm@parsetimezn` Define command to parse time in the format ⟨*h*⟩:⟨*m*⟩:⟨*s*⟩ ⟨*znh*⟩:⟨*znm*⟩. The results are stored in `\@dtm@hour`, `\@dtm@minute`, `\@dtm@second`, `\@dtm@timezonehour` and `\@dtm@timezoneminute`. The arguments are expanded.

```
\def\@dtm@parsetimezn#1:#2:#3 #4\@dtm@endparsetimezn{%
  \@dtm@parsetime#1:#2:#3\@dtm@endparsetime
  \@dtm@parsezone{#4}%
}
```

`\@dtm@parsezone` Define command to parse time zone in the format `Z` or ⟨*znh*⟩:⟨*znm*⟩. The results are stored in `\@dtm@timezonehour` and `\@dtm@timezoneminute`. The arguments are expanded in the event that registers are used.

```
\newcommand*{\@dtm@parsezone}[1]{%
  \ifstrequal{#1}{Z}%
  {%
    \def\@dtm@timezonehour{+00}%
    \def\@dtm@timezoneminute{00}%
  }%
  {%
    \@dtm@parse@zone#1\@dtm@endparse@zone
  }%
}
\def\@dtm@parse@zone#1:#2\@dtm@endparse@zone{%
  \edef\@dtm@timezonehour{\number#1}%
  \edef\@dtm@timezoneminute{\number#2}%
}
```

\@dtm@parsetimestamp Parse date and time in ISO format ⟨*YYYY*⟩-⟨*MM*⟩-⟨*DD*⟩T⟨*hh*⟩:⟨*mm*⟩:⟨*sec*⟩⟨*time zone*⟩ where ⟨*time zone*⟩ may be Z or in the form ⟨*hh*⟩:⟨*mm*⟩ (where ⟨*hh*⟩ includes the sign).

```
\def\@dtm@parsetimestamp#1-#2-#3T#4:#5:#6#7#8\@dtm@endparsetimestamp{%
  \@dtm@parsedate#1-#2-#3\@dtm@endparsedate
  \@dtm@parsetime#4:#5:#6#7\@dtm@endparsetime
  \@dtm@parsezone{#8}%
}
```

\DTMsavefilemoddate Not available for some engines.

```
\newcommand*{\DTMsavefilemoddate}[2]{%
  \@dtm@warning{Your TeX engine doesn't support accessing
  file modification dates}%
  \cslet{@dtm@#1@year}{0}%
  \cslet{@dtm@#1@month}{0}%
  \cslet{@dtm@#1@day}{0}%
  \cslet{@dtm@#1@dow}{-1}%
  \cslet{@dtm@#1@hour}{0}%
  \cslet{@dtm@#1@minute}{0}%
  \cslet{@dtm@#1@second}{0}%
  \cslet{@dtm@#1@TZhour}{0}%
  \cslet{@dtm@#1@TZminute}{0}%
}
```

Find out the current time. If PDFLaTeX is being used, then it can be fetched from \pdfcreationdate

```
\ifdef\pdfcreationdate
{%
```

Define commands to parse \pdfcreationdate

```
  \def\@dtm@parsepdfdatetime#1:#2#3#4#5#6#7#8#9{%
    \def\@dtm@year{#2#3#4#5}%
    \def\@dtm@month{#6#7}%
    \def\@dtm@day{#8#9}%
    \@dtm@parsepdftime
  }
  \def\@dtm@parsepdftime#1#2#3#4#5#6#7\@dtm@endparsepdfdatetime{%
    \def\@dtm@hour{#1#2}%
    \def\@dtm@minute{#3#4}%
    \def\@dtm@second{#5#6}%
    \ifstrequal{#7}{Z}%
    {%
      \def\@dtm@timezonehour{00}%
      \def\@dtm@timezoneminute{00}%
    }%
    {%
      \@dtm@parsepdftimezone#7%
    }%
  }
```

```
\def\@dtm@parsepdftimezone#1'#2'{%
  \def\@dtm@timezonehour{#1}%
  \def\@dtm@timezoneminute{#2}%
}%
```

Now parse \pdfcreationdate

```
\expandafter\@dtm@parsepdfdatetime\pdfcreationdate\@dtm@endparsepdfdatetime
```

Save the values.

```
\let\@dtm@currentyear\@dtm@year
\let\@dtm@currentmonth\@dtm@month
\let\@dtm@currentday\@dtm@day
\let\@dtm@currenthour\@dtm@hour
\let\@dtm@currentminute\@dtm@minute
\let\@dtm@currentsecond\@dtm@second
\let\@dtm@currenttimezonehour\@dtm@timezonehour
\let\@dtm@currenttimezoneminute\@dtm@timezoneminute
%
```

LuaTEX doesn't provide \pdffilemoddate (but it does provide \pdfcreationdate).

```
\ifdef\pdffilemoddate
{%
  \renewcommand*{\DTMsavefilemoddate}[2]{%
    \expandafter\@dtm@parsepdfdatetime\pdffilemoddate{#2}\@dtm@endparsepdfdatetime
    \cslet{@dtm@#1@year}{\@dtm@year}%
    \cslet{@dtm@#1@month}{\@dtm@month}%
    \cslet{@dtm@#1@day}{\@dtm@day}%
    \cslet{@dtm@#1@dow}{\@dtm@dow}%
    \cslet{@dtm@#1@hour}{\@dtm@hour}%
    \cslet{@dtm@#1@minute}{\@dtm@minute}%
    \cslet{@dtm@#1@second}{\@dtm@second}%
    \cslet{@dtm@#1@TZhour}{\@dtm@timezonehour}%
    \cslet{@dtm@#1@TZminute}{\@dtm@timezoneminute}%
  }
}%
{%
```

Lua time zone information provided by %z is OS dependent, so this might not work.

```
  \ifdef\directlua
  {
    \renewcommand*{\DTMsavefilemoddate}[2]{%
      \expandafter\@dtm@parseluadatetime
        \directlua{tex.print(os.date(
           "\expandafter\@gobble\string\%Y-%
            \expandafter\@gobble\string\%m-%
            \expandafter\@gobble\string\%d-%
            \expandafter\@gobble\string\%w
            \expandafter\@gobble\string\%H:%
            \expandafter\@gobble\string\%M:%
            \expandafter\@gobble\string\%S
```

```
            \expandafter\@gobble\string\%z",
          lfs.attributes("#2").modification))}%
      \@dtm@endparseluadatetime
      \cslet{@dtm@#1@year}{\@dtm@year}%
      \cslet{@dtm@#1@month}{\@dtm@month}%
      \cslet{@dtm@#1@day}{\@dtm@day}%
      \cslet{@dtm@#1@dow}{\@dtm@dow}%
      \cslet{@dtm@#1@hour}{\@dtm@hour}%
      \cslet{@dtm@#1@minute}{\@dtm@minute}%
      \cslet{@dtm@#1@second}{\@dtm@second}%
      \cslet{@dtm@#1@TZhour}{\@dtm@TZhour}%
      \cslet{@dtm@#1@TZminute}{\@dtm@TZminute}%
   }
   \def\@dtm@parseluadatetime#1-#2-#3-#4 #5:#6:#7 #8\@dtm@endparseluadatetime{%
      \edef\@dtm@year{\number#1}%
      \edef\@dtm@month{\number#2}%
      \edef\@dtm@day{\number#3}%
      \edef\@dtm@dow{\number#4}%
      \edef\@dtm@hour{\number#5}%
      \edef\@dtm@minute{\number#6}%
      \edef\@dtm@second{\number#7}%
      \@dtm@parseluatimezone#8000000\@dtm@endparseluatimezone
   }
   \def\@dtm@parseluatimezone#1#2#3#4#5#6{%
      \ifstrequal{#1}{+}%
      {%
        \def\@dtm@TZhour{#1#2#3}%
        \ifstrequal{#4}{:}%
        {%
           \def\@dtm@TZminute{#5#6}%
        }%
        {%
           \def\@dtm@TZminute{#4#5}%
        }%
      }%
      {%
        \ifstrequal{#1}{-}%
        {%
           \def\@dtm@TZhour{#1#2#3}%
           \ifstrequal{#4}{:}%
           {%
              \def\@dtm@TZminute{#5#6}%
           }%
           {%
              \def\@dtm@TZminute{#4#5}%
           }%
        }%
        {%
          \ifstrequal{#1}{Z}%
```

```
            {%
              \def\@dtm@TZhour{0}%
              \def\@dtm@TZminute{0}%
            }%
            {%
              \def\@dtm@TZhour{#1#2}%
              \ifstrequal{#3}{:}%
              {%
                 \def\@dtm@TZminute{#4#5}%
              }%
              {%
                 \def\@dtm@TZminute{#3#4}%
              }%
            }%
          }%
        }%
        \@@dtm@parseluatimezone
      }
      \def\@@dtm@parseluatimezone#1\@dtm@endparseluatimezone{%
      }
    }
    {}
  }%
}%
{%
```

`\pdfcreationdate` not defined. By a process of elimination, the TEX engine is either XETEX or it's very old. (LuaTEX recognises `\pdfcreationdate`.) In this case, the seconds and time zone can't be obtained. The hour and minute need to be calculated from TEX's `\time` primitive.

```
    \count@=\time\relax
    \divide\count@ by 60\relax
    \edef\@dtm@currenthour{\number\count@}%
    \multiply\count@ by -60\relax
    \advance\count@ by \time\relax
    \edef\@dtm@currentminute{\number\count@}%
    \newcommand*{\@dtm@currentsecond}{00}%
    \newcommand\@dtm@currenttimezonehour{00}%
    \newcommand\@dtm@currenttimezoneminute{00}%
```

Get the day, month and year from TEX's primitives.

```
    \edef\@dtm@currentyear{\number\year}%
    \edef\@dtm@currentmonth{\number\month}%
    \edef\@dtm@currentday{\number\day}%
}
```

Make `\DTMsavefilemoddate` robust.

```
\robustify\DTMsavefilemoddate
```

Current day of week defaults to `-1` (that is, ignore it).

\@dtm@currentdow

```
\newcommand*{\@dtm@currentdow}{-1}
```

Allow XeLaTeX users a way of manually setting the current time zone.

\DTMsetcurrentzone

```
\newcommand*{\DTMsetcurrentzone}[2]{%
  \renewcommand\@dtm@currenttimezonehour{#1}%
  \renewcommand\@dtm@currenttimezoneminute{#2}%
}
```

\today

```
\renewcommand*{\today}{%
 \DTMdisplaydate
  {\@dtm@currentyear}%
  {\@dtm@currentmonth}%
  {\@dtm@currentday}%
  {\@dtm@currentdow}%
}
```

\Today First letter upper case version.

```
\newcommand*{\Today}{%
 \DTMDisplaydate
  {\@dtm@currentyear}%
  {\@dtm@currentmonth}%
  {\@dtm@currentday}%
  {\@dtm@currentdow}%
}
```

\DTMdisplaydate `\DTMdisplaydate{⟨year⟩}{⟨month⟩}{⟨day⟩}{⟨day of week⟩}`

Display the given date. If the day of week is negative, ignore it. The default style ignores it regardless.

```
\newcommand*\DTMdisplaydate[4]{%
  \number#1\dtm@yearmonthsep\DTMtwodigits{#2}\dtm@monthdaysep\DTMtwodigits{#3}%
}%
```

\DTMDisplaydate First letter upper case version. Defaults to \DTMdisplaydate.

```
\newcommand*{\DTMDisplaydate}{\DTMdisplaydate}
```

\DTMdate Display date where date is specified in the format ⟨*yyyy*⟩-⟨*mm*⟩-⟨*dd*⟩. Use \expandafter in case argument is a control sequence containing the date. This command isn't expandable

```
\newrobustcmd*{\DTMdate}[1]{%
  \expandafter\@dtm@parsedate#1\@dtm@endparsedate
  \DTMdisplaydate{\@dtm@year}{\@dtm@month}{\@dtm@day}{\@dtm@dow}%
}
```

\DTMDate Upper case version.

```
\newrobustcmd*{\DTMDate}[1]{%
  \expandafter\@dtm@parsedate#1\@dtm@endparsedate
  \DTMDisplaydate{\@dtm@year}{\@dtm@month}{\@dtm@day}{\@dtm@dow}%
}
```

\DTMcurrenttime Display the current time.

```
\newcommand*{\DTMcurrenttime}{%
 \DTMdisplaytime
  {\@dtm@currenthour}%
  {\@dtm@currentminute}%
  {\@dtm@currentsecond}%
}
```

\DTMdisplaytime `\DTMdisplaytime{`⟨*hour*⟩`}{`⟨*minute*⟩`}{`⟨*sec*⟩`}`

Display the given time.

```
\newcommand*\DTMdisplaytime[3]{%
  \DTMtwodigits{#1}\dtm@hourminsep\DTMtwodigits{#2}%
  \ifDTMshowseconds\dtm@minsecsep\DTMtwodigits{#3}\fi
}%
```

\DTMtime Display date where time is specified in the format ⟨*hour*⟩:⟨*minute*⟩:⟨*seconds*⟩. This uses `\expandafter` in case argument is a control sequence containing the time. Not expandable.

```
\newrobustcmd*{\DTMtime}[1]{%
  \@dtm@parsetime#1\@dtm@endparsetime
  \DTMdisplaytime{\@dtm@hour}{\@dtm@minute}{\@dtm@second}%
}
```

\DTMcurrentzone Display the current time zone.

```
\newcommand*{\DTMcurrentzone}{%
 \DTMdisplayzone
  {\@dtm@currenttimezonehour}%
  {\@dtm@currenttimezoneminute}%
}
```

\DTMdisplayzone Display time zone.

```
\newcommand*{\DTMdisplayzone}[2]{%
 \ifboolexpe
 { bool{DTMshowisoZ}
   and test{\ifnumequal{#1}{0}}
   and test{\ifnumequal{#2}{0}}
 }%
 {%
```

```
    Z%
  }%
  {%
   \ifnum#1<0\else+\fi\DTMtwodigits{#1}%
   \ifDTMshowzoneminutes\dtm@hourminsep\DTMtwodigits{#2}\fi
  }%
}
```

\DTMnow Current date, time and time zone.

```
\newcommand*{\DTMnow}{%
 \DTMdisplay
 {\@dtm@currentyear}
 {\@dtm@currentmonth}
 {\@dtm@currentday}
 {\@dtm@currentdow}
 {\@dtm@currenthour}%
 {\@dtm@currentminute}%
 {\@dtm@currentsecond}%
 {\@dtm@currenttimezonehour}%
 {\@dtm@currenttimezoneminute}%
}
```

\DTMNow Current date, time and time zone.

```
\newcommand*{\DTMNow}{%
 \DTMDisplay
 {\@dtm@currentyear}
 {\@dtm@currentmonth}
 {\@dtm@currentday}
 {\@dtm@currentdow}
 {\@dtm@currenthour}%
 {\@dtm@currentminute}%
 {\@dtm@currentsecond}%
 {\@dtm@currenttimezonehour}%
 {\@dtm@currenttimezoneminute}%
}
```

\DTMdisplay `\DTMdisplay{⟨YYYY⟩}{⟨MM⟩}{⟨DD⟩}{⟨DOW⟩}{⟨hh⟩}{⟨mm⟩}{⟨ss⟩}{⟨TZh⟩}{⟨TZm⟩}`

Display the date and time.

```
\newcommand*{\DTMdisplay}[9]{%
 \ifDTMshowdate
   \DTMdisplaydate{#1}{#2}{#3}{#4}%
   \dtm@datetimesep
 \fi
 \DTMdisplaytime
```

```
  {#5}%
  {#6}%
  {#7}%
 \ifDTMshowzone
  \dtm@timezonesep
  \DTMdisplayzone
   {#8}%
   {#9}%
 \fi
}
```

\DTMDisplay

\DTMDisplay{⟨*YYYY*⟩}{⟨*MM*⟩}{⟨*DD*⟩}{⟨*DOW*⟩}{⟨*hh*⟩}{⟨*mm*⟩}{⟨*ss*⟩}{⟨*TZh*⟩}{⟨*TZm*⟩}

First letter upper case version. Defaults to `\DTMdisplay`.

```
\newcommand*{\DTMDisplay}{\DTMdisplay}
```

9.1.2 Styles

Provide user level commands for displaying number as two digits. (Truncate if over 99, to allow for converting year to two digits).

\DTMtwodigits

```
\newcommand*{\DTMtwodigits}[1]{%
 \ifnum#1<0
  -\DTMtwodigits{-#1}%
 \else
   \ifnum#1<100
     \ifnum#1<10
       0\number#1
     \else
       \number#1
     \fi
   \else
```

`\numexpr` rounds rather than truncates integer division, which is a little awkward to get around in an expandable context.

```
     \ifnum\numexpr#1-(#1/100)*100<0
       \number\numexpr#1-((#1/100)-1)*100\relax
     \else
       \number\numexpr#1-(#1/100)*100\relax
     \fi
   \fi
 \fi
}
```

\DTMcentury Expands to the given number divided by 100 rounded upwards (in absolute terms). Provided in case the user just wants the century.

```
\newcommand*{\DTMcentury}[1]{%
  \ifnum#1<0
    -\DTMcentury{-#1}%
  \else
    \ifnum\numexpr#1-(#1/100)*100<1
       \number\numexpr#1/100\relax
    \else
       \number\numexpr(#1/100)+1\relax
    \fi
  \fi
}
```

\DTMdivhundred Expands to the given number modulo 100.

```
\newcommand*{\DTMdivhundred}[1]{%
  \ifnum#1<0
    -\DTMdivhundred{-#1}%
  \else
    \ifnum\numexpr#1-(#1/100)*100<0
       \number\numexpr(#1)/100-1\relax
    \else
       \number\numexpr((#1)/100)\relax
    \fi
  \fi
}
```

\DTMtexorpdfstring Provide user with a command that will use hyperref's `\texorpdfstring` if hyperref has been loaded. If hyperref isn't loaded it just does the first argument.

```
\newcommand*{\DTMtexorpdfstring}[2]{#1}
\AtBeginDocument{%
  \@ifpackageloaded{hyperref}%
  {%
    \renewcommand*{\DTMtexorpdfstring}{\texorpdfstring}%
  }%
  {}%
}
```

Access separator:

\DTMsep

```
\newcommand*{\DTMsep}[1]{\csname dtm@#1sep\endcsname}
```

Date-only styles are stored internally as \@dtm@datestyle@⟨*label*⟩, time-only styles are stored internally as \@dtm@timestyle@⟨*label*⟩, zone-only styles are stored internally as \@dtm@zonestyle@⟨*label*⟩.

\DTMnewdatestyle Define a new date-only style. This should only redefine \DTMdisplaydate and \DTMDisplaydate, which may or may not use the separators \dtm@yearmonthsep and \dtm@monthdaysep.

```
\newcommand*{\DTMnewdatestyle}[2]{%
  \ifcsdef{@dtm@datestyle@#1}%
  {%
    \PackageError{datetime2}{Date style `#1' already exists}{}%
  }%
  {%
     \csdef{@dtm@datestyle@#1}{#2}%
  }%
}
```

\DTMnewtimestyle Define a new time-only style. This should only redefine \DTMdisplaytime, which may or may not use the separators \dtm@hourminsep and \dtm@minsecsep.

```
\newcommand*{\DTMnewtimestyle}[2]{%
  \ifcsdef{@dtm@timestyle@#1}%
  {%
    \PackageError{datetime2}{Time style `#1' already exists}{}%
  }%
  {%
     \csdef{@dtm@timestyle@#1}{#2}%
  }%
}
```

\DTMnewtimezone Define a new zone-only style. This should only redefine \DTMdisplayzone, which may or may not use the separator \dtm@hourminsep.

```
\newcommand*{\DTMnewzonestyle}[2]{%
  \ifcsdef{@dtm@zonestyle@#1}%
  {%
    \PackageError{datetime2}{Zone style `#1' already exists}{}%
  }%
  {%
     \csdef{@dtm@zonestyle@#1}{#2}%
  }%
}
```

Zone styles may use mappings to use a regional time zone (such as GMT or BST). It's up to the language modules to define these mappings. A mapping for time zone ⟨*TZh*⟩:⟨*TZm*⟩ is stored in \@dtm@zonemap@⟨*TZh*⟩:⟨*TZm*⟩.

\DTMdefzonemap `\DTMdefzonemap{⟨TZh⟩}{⟨TZm⟩}{⟨map⟩}`

This will override any previous mapping for the given time zone.

```
\newcommand*{\DTMdefzonemap}[3]{%
  \csdef{@dtm@zonemap@\DTMtwodigits{#1}:\DTMtwodigits{#2}}{#3}%
```

```
}
```

usezonemapordefault Expands to the mapping or the default if not defined.

```
\newcommand*{\DTMusezonemapordefault}[2]{%
  \ifcsundef{@dtm@zonemap@\DTMtwodigits{#1}:\DTMtwodigits{#2}}%
  {%
    \ifnum#1<0\else+\fi
    \DTMtwodigits{#1}%
    \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{#2}\fi
  }%
  {\csname @dtm@zonemap@\DTMtwodigits{#1}:\DTMtwodigits{#2}\endcsname}%
}
```

\DTMusezonemap Expands to the mapping. (No check if defined.)

```
\newcommand*{\DTMusezonemap}[2]{%
  \csname @dtm@zonemap@\DTMtwodigits{#1}:\DTMtwodigits{#2}\endcsname
}
```

\DTMhaszonemap

```
\newcommand*{\DTMhaszonemap}[4]{%
  \ifcsundef{@dtm@zonemap@\DTMtwodigits{#1}:\DTMtwodigits{#2}}{#4}{#3}%
}
```

\DTMclearmap Undefines the given zone mapping. No check is made to determine if the map exists.

```
\newcommand*{\DTMclearmap}[2]{%
 \csundef{@dtm@zonemap@\DTMtwodigits{#1}:\DTMtwodigits{#2}}%
}
```

\DTMshowmap Debugging command.

```
\newcommand*{\DTMshowmap}[2]{%
 \csshow{@dtm@zonemap@\DTMtwodigits{#1}:\DTMtwodigits{#2}}%
}
```

\DTMresetzones Regional modules should use this before setting their local zones, so that users can unset previously defined zones that are outside the region if they require. By default this does nothing, so no modifications are made.

```
\newcommand*{\DTMresetzones}{}
```

\DTMNatoZoneMaps Provide a command to set the time zone abbreviations to the military/NATO style.

```
\newcommand*{\DTMNatoZoneMaps}{%
  \defzonemap{01}{00}{A}% Alpha time zone
  \defzonemap{02}{00}{B}% Bravo time zone
  \defzonemap{03}{00}{C}% Charlie time zone
  \defzonemap{04}{00}{D}% Delta time zone
  \defzonemap{05}{00}{E}% Echo time zone
  \defzonemap{06}{00}{F}% Foxtrot time zone
```

```
  \defzonemap{07}{00}{G}% Golf time zone
  \defzonemap{08}{00}{H}% Hotel time zone
  \defzonemap{09}{00}{I}% India time zone
  \defzonemap{10}{00}{K}% Kilo time zone
  \defzonemap{11}{00}{L}% Lima time zone
  \defzonemap{12}{00}{M}% Mike time zone
  \defzonemap{-01}{00}{N}% November time zone
  \defzonemap{-02}{00}{O}% Oscar time zone
  \defzonemap{-03}{00}{P}% Papa time zone
  \defzonemap{-04}{00}{Q}% Quebec time zone
  \defzonemap{-05}{00}{R}% Romeo time zone
  \defzonemap{-06}{00}{S}% Sierra time zone
  \defzonemap{-07}{00}{T}% Tango time zone
  \defzonemap{-08}{00}{U}% Uniform time zone
  \defzonemap{-09}{00}{V}% Victor time zone
  \defzonemap{-10}{00}{W}% Whiskey time zone
  \defzonemap{-11}{00}{X}% X-ray time zone
  \defzonemap{-12}{00}{Y}% Yankee time zone
  \defzonemap{00}{00}{Z}% Zulu time zone
}
```

`\DTMnewstyle` `\DTMnewstyle{⟨label⟩}{⟨date style definition⟩}{⟨time style definition⟩}{⟨zone style definition⟩}{⟨full format definition⟩}`

Define a new style. The full format redefines `\DTMdisplay` and `\DTMDisplay`.

```
\newcommand*{\DTMnewstyle}[5]{%
  \ifcsdef{@dtm@style@#1}%
  {%
    \PackageError{datetime2}{Style `#1' already exists}{}%
  }%
  {%
    \DTMnewdatestyle{#1}{#2}%
    \DTMnewtimestyle{#1}{#3}%
    \DTMnewzonestyle{#1}{#4}%
     \csdef{@dtm@style@#1}{%
        \csuse{@dtm@datestyle@#1}%
        \csuse{@dtm@timestyle@#1}%
        \csuse{@dtm@zonestyle@#1}%
        #5%
     }%
  }%
}
```

`\DTMsetdatestyle`

```
\newrobustcmd*{\DTMsetdatestyle}[1]{%
  \ifcsdef{@dtm@datestyle@#1}%
```

```
  {\csuse{@dtm@datestyle@#1}}%
  {%
    \PackageError{datetime2}{Date style ‘#1’ not defined}{}%
  }%
}
```

\DTMsettimestyle

```
\newrobustcmd*{\DTMsettimestyle}[1]{%
  \ifcsdef{@dtm@timestyle@#1}%
  {\csuse{@dtm@timestyle@#1}}%
  {%
    \PackageError{datetime2}{Time style ‘#1’ not defined}{}%
  }%
}
```

\DTMsetzonestyle

```
\newrobustcmd*{\DTMsetzonestyle}[1]{%
  \ifcsdef{@dtm@zonestyle@#1}%
  {\csuse{@dtm@zonestyle@#1}}%
  {%
    \PackageError{datetime2}{Zone style ‘#1’ not defined}{}%
  }%
}
```

\DTMsetstyle

```
\newrobustcmd*{\DTMsetstyle}[1]{%
  \ifcsdef{@dtm@style@#1}%
  {\csuse{@dtm@style@#1}}%
  {%
     \let\dtm@unknownstyle\@dtm@unknownstyle
     \ifcsdef{@dtm@datestyle#1}%
       {\csuse{@dtm@datestyle@#1}\let\dtm@unknownstyle\@dtm@unknown@style}%
       {\@dtm@warning{No date style ‘#1’ defined}}%
     \ifcsdef{@dtm@timestyle#1}%
       {\csuse{@dtm@timestyle@#1}\let\dtm@unknownstyle\@dtm@unknown@style}%
       {\@dtm@warning{No time style ‘#1’ defined}}%
     \ifcsdef{@dtm@zonestyle#1}%
       {\csuse{@dtm@zonestyle@#1}\let\dtm@unknownstyle\@dtm@unknown@style}%
       {\@dtm@warning{No zone style ‘#1’ defined}}%
     \dtm@unknownstyle{#1}%
  }%
}
```

\@dtm@unknownstyle

```
\newcommand*{\@dtm@unknownstyle}[1]{%
  \PackageError{datetime2}{Unknown style ‘#1’}{}%
}
```

\@dtm@unknown@style

```
\newcommand*{\@dtm@unknown@style}[1]{%
  \@dtm@warning{No full style `#1' defined}{}%
}
```

Define `default` style:

```
\DTMnewstyle
 {default}%label
 {% date style
    \renewcommand*\DTMdisplaydate[4]{%
      \number##1\DTMsep{yearmonth}\DTMtwodigits{##2}%
      \DTMsep{monthday}\DTMtwodigits{##3}%
    }%
    \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
 }%
 {% time style
    \renewcommand*\DTMdisplaytime[3]{%
      \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
      \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
    }%
 }%
 {% zone style
   \renewcommand*{\DTMdisplayzone}[2]{%
     \ifboolexpe
     { bool{DTMshowisoZ}
       and test{\ifnumequal{##1}{0}}
       and test{\ifnumequal{##2}{0}}
     }%
     {%
       Z%
     }%
     {%
      \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
      \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
     }%
   }%
 }%
 {% full style
   \renewcommand*{\DTMdisplay}[9]{%
    \ifDTMshowdate
     \DTMdisplaydate{##1}{##2}{##3}{##4}%
     \DTMsep{datetime}%
    \fi
    \DTMdisplaytime
     {##5}%
     {##6}%
     {##7}%
    \ifDTMshowzone
     \DTMsep{timezone}%
     \DTMdisplayzone
```

```
      {##8}%
      {##9}%
    \fi
  }%
  \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define `iso` style which ignores the separator settings:

```
\DTMnewstyle
{iso}%label
{% date style
   \renewcommand*\DTMdisplaydate[4]{%
     \number##1-\DTMtwodigits{##2}-\DTMtwodigits{##3}%
   }%
   \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
}%
{% time style
   \renewcommand*\DTMdisplaytime[3]{%
     \DTMtwodigits{##1}:\DTMtwodigits{##2}%
     \ifDTMshowseconds:\DTMtwodigits{##3}\fi
   }%
}%
{% zone style
  \renewcommand*{\DTMdisplayzone}[2]{%
    \ifboolexpe
    { bool{DTMshowisoZ}
      and test{\ifnumequal{##1}{0}}
      and test{\ifnumequal{##2}{0}}
    }%
    {%
      Z%
    }%
    {%
     \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
     \ifDTMshowzoneminutes:\DTMtwodigits{##2}\fi
    }%
  }%
}%
{% full style
  \renewcommand*{\DTMdisplay}[9]{%
   \ifDTMshowdate
    \DTMdisplaydate{##1}{##2}{##3}{##4}%
    T%
   \fi
   \DTMdisplaytime
    {##5}%
    {##6}%
    {##7}%
   \ifDTMshowzone
    \DTMdisplayzone
```

```
      {##8}%
      {##9}%
    \fi
  }%
  \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define pdf style which converts into a format that can be used in \pdfinfo:

```
\DTMnewstyle
{pdf}%label
{% date style
   \renewcommand*\DTMdisplaydate[4]{%
     D:\number##1 % space intended
     \DTMtwodigits{##2}\DTMtwodigits{##3}%
   }%
   \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
}%
{% time style
   \renewcommand*\DTMdisplaytime[3]{%
     \DTMtwodigits{##1}\DTMtwodigits{##2}\DTMtwodigits{##3}%
   }%
}%
{% zone style
  \renewcommand*{\DTMdisplayzone}[2]{%
    \ifboolexpe
    { bool{DTMshowisoZ}
      and test{\ifnumequal{##1}{0}}
      and test{\ifnumequal{##2}{0}}
    }%
    {%
      Z%
    }%
    {%
     \ifnum##1<0\else+\fi\DTMtwodigits{##1}'\DTMtwodigits{##2}'%
    }%
  }%
}%
{% full style
  \renewcommand*{\DTMdisplay}[9]{%
   \DTMdisplaydate{##1}{##2}{##3}{##4}%
   \DTMdisplaytime{##5}{##6}{##7}%
   \DTMdisplayzone{##8}{##9}%
  }%
  \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define yyyymd style:

```
\DTMnewstyle
{yyyymd}%label
{% date style
```

```
    \renewcommand*\DTMdisplaydate[4]{%
      \number##1
      \DTMsep{yearmonth}%
      \number##2
      \DTMsep{monthday}%
      \number##3
    }%
    \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
}%
{% time style
    \renewcommand*\DTMdisplaytime[3]{%
      \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
      \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
    }%
}%
{% zone style
   \renewcommand*{\DTMdisplayzone}[2]{%
     \ifboolexpe
     { bool{DTMshowisoZ}
       and test{\ifnumequal{##1}{0}}
       and test{\ifnumequal{##2}{0}}
     }%
     {%
       Z%
     }%
     {%
      \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
      \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
     }%
   }%
}%
{% full style
   \renewcommand*{\DTMdisplay}[9]{%
    \ifDTMshowdate
     \DTMdisplaydate{##1}{##2}{##3}{##4}%
     \DTMsep{datetime}%
    \fi
    \DTMdisplaytime
     {##5}%
     {##6}%
     {##7}%
    \ifDTMshowzone
     \DTMsep{timezone}%
     \DTMdisplayzone
      {##8}%
      {##9}%
    \fi
   }%
   \renewcommand*{\DTMDisplay}{\DTMdisplay}%
```

```
}
```

Define ddmmyyyy style:

```
\DTMnewstyle
 {ddmmyyyy}%label
 {% date style
    \renewcommand*\DTMdisplaydate[4]{%
      \DTMtwodigits{##3}\DTMsep{monthday}%
      \DTMtwodigits{##2}\DTMsep{yearmonth}%
      \number##1
    }%
    \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
 }%
 {% time style
    \renewcommand*\DTMdisplaytime[3]{%
      \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
      \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
    }%
 }%
 {% zone style
   \renewcommand*{\DTMdisplayzone}[2]{%
     \ifboolexpe
     { bool{DTMshowisoZ}
       and test{\ifnumequal{##1}{0}}
       and test{\ifnumequal{##2}{0}}
     }%
     {%
       Z%
     }%
     {%
      \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
      \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
     }%
   }%
 }%
 {% full style
   \renewcommand*{\DTMdisplay}[9]{%
    \ifDTMshowdate
     \DTMdisplaydate{##1}{##2}{##3}{##4}%
     \DTMsep{datetime}%
    \fi
    \DTMdisplaytime
     {##5}%
     {##6}%
     {##7}%
    \ifDTMshowzone
     \DTMsep{timezone}%
     \DTMdisplayzone
      {##8}%
      {##9}%
```

```
    \fi
   }%
   \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define dmyyyy style:

```
\DTMnewstyle
{dmyyyy}%label
{% date style
   \renewcommand*\DTMdisplaydate[4]{%
     \number##3
     \DTMsep{monthday}%
     \number##2
     \DTMsep{yearmonth}%
     \number##1
   }%
   \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
}%
{% time style
   \renewcommand*\DTMdisplaytime[3]{%
     \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
     \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
   }%
}%
{% zone style
  \renewcommand*{\DTMdisplayzone}[2]{%
    \ifboolexpe
    { bool{DTMshowisoZ}
      and test{\ifnumequal{##1}{0}}
      and test{\ifnumequal{##2}{0}}
    }%
    {%
      Z%
    }%
    {%
     \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
     \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
    }%
  }%
}%
{% full style
  \renewcommand*{\DTMdisplay}[9]{%
   \ifDTMshowdate
    \DTMdisplaydate{##1}{##2}{##3}{##4}%
    \DTMsep{datetime}%
   \fi
   \DTMdisplaytime
    {##5}%
    {##6}%
    {##7}%
```

```
    \ifDTMshowzone
     \DTMsep{timezone}%
     \DTMdisplayzone
      {##8}%
      {##9}%
    \fi
   }%
   \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define dmyy style:

```
\DTMnewstyle
 {dmyy}%label
 {% date style
    \renewcommand*\DTMdisplaydate[4]{%
      \number##3 % space intended
      \DTMsep{monthday}%
      \number##2 % space intended
      \DTMsep{yearmonth}%
      \DTMtwodigits{##1}%
    }%
    \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
 }%
 {% time style
    \renewcommand*\DTMdisplaytime[3]{%
      \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
      \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
    }%
 }%
 {% zone style
   \renewcommand*{\DTMdisplayzone}[2]{%
     \ifboolexpe
     { bool{DTMshowisoZ}
       and test{\ifnumequal{##1}{0}}
       and test{\ifnumequal{##2}{0}}
     }%
     {%
       Z%
     }%
     {%
      \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
      \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
     }%
   }%
 }%
 {% full style
   \renewcommand*{\DTMdisplay}[9]{%
    \ifDTMshowdate
     \DTMdisplaydate{##1}{##2}{##3}{##4}%
     \DTMsep{datetime}%
```

```
    \fi
    \DTMdisplaytime
     {##5}%
     {##6}%
     {##7}%
    \ifDTMshowzone
     \DTMsep{timezone}%
     \DTMdisplayzone
      {##8}%
      {##9}%
    \fi
   }%
   \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define mmddyyyy style:

```
\DTMnewstyle
{mmddyyyy}%label
{% date style
   \renewcommand*\DTMdisplaydate[4]{%
     \DTMtwodigits{##2}\DTMsep{monthday}%
     \DTMtwodigits{##3}\DTMsep{dayyear}%
     \number##1
   }%
   \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
}%
{% time style
   \renewcommand*\DTMdisplaytime[3]{%
     \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
     \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
   }%
}%
{% zone style
  \renewcommand*{\DTMdisplayzone}[2]{%
    \ifboolexpe
    { bool{DTMshowisoZ}
      and test{\ifnumequal{##1}{0}}
      and test{\ifnumequal{##2}{0}}
    }%
    {%
      Z%
    }%
    {%
     \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
     \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
    }%
  }%
}%
{% full style
  \renewcommand*{\DTMdisplay}[9]{%
```

```
    \ifDTMshowdate
     \DTMdisplaydate{##1}{##2}{##3}{##4}%
     \DTMsep{datetime}%
    \fi
    \DTMdisplaytime
     {##5}%
     {##6}%
     {##7}%
    \ifDTMshowzone
     \DTMsep{timezone}%
     \DTMdisplayzone
      {##8}%
      {##9}%
    \fi
   }%
   \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define `mdyyyy` style:

```
\DTMnewstyle
 {mdyyyy}%label
 {% date style
    \renewcommand*\DTMdisplaydate[4]{%
      \number##2 % space intended
      \DTMsep{monthday}%
      \number##3 % space intended
      \DTMsep{dayyear}%
      \number##1 % space intended
    }%
    \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
 }%
 {% time style
    \renewcommand*\DTMdisplaytime[3]{%
      \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
      \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
    }%
 }%
 {% zone style
   \renewcommand*{\DTMdisplayzone}[2]{%
     \ifboolexpe
     { bool{DTMshowisoZ}
       and test{\ifnumequal{##1}{0}}
       and test{\ifnumequal{##2}{0}}
     }%
     {%
       Z%
     }%
     {%
      \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
      \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
```

```
    }%
  }%
}%
{% full style
  \renewcommand*{\DTMdisplay}[9]{%
   \ifDTMshowdate
    \DTMdisplaydate{##1}{##2}{##3}{##4}%
    \DTMsep{datetime}%
   \fi
   \DTMdisplaytime
    {##5}%
    {##6}%
    {##7}%
   \ifDTMshowzone
    \DTMsep{timezone}%
    \DTMdisplayzone
     {##8}%
     {##9}%
   \fi
  }%
  \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define `mdyy` style:

```
\DTMnewstyle
{mdyy}%label
{% date style
   \renewcommand*\DTMdisplaydate[4]{%
     \number##2 % space intended
     \DTMsep{monthday}%
     \number##3 % space intended
     \DTMsep{dayyear}%
     \DTMtwodigits{##1}%
   }%
   \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
}%
{% time style
   \renewcommand*\DTMdisplaytime[3]{%
     \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
     \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
   }%
}%
{% zone style
  \renewcommand*{\DTMdisplayzone}[2]{%
    \ifboolexpe
    { bool{DTMshowisoZ}
      and test{\ifnumequal{##1}{0}}
      and test{\ifnumequal{##2}{0}}
    }%
    {%
```

```
      Z%
    }%
    {%
     \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
     \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
    }%
  }%
}%
{% full style
  \renewcommand*{\DTMdisplay}[9]{%
   \ifDTMshowdate
    \DTMdisplaydate{##1}{##2}{##3}{##4}%
    \DTMsep{datetime}%
   \fi
   \DTMdisplaytime
    {##5}%
    {##6}%
    {##7}%
   \ifDTMshowzone
    \DTMsep{timezone}%
    \DTMdisplayzone
     {##8}%
     {##9}%
   \fi
  }%
  \renewcommand*{\DTMDisplay}{\DTMdisplay}%
}
```

Define `hmmss` time style

```
\DTMnewtimestyle
 {hmmss}% label
 {%
    \renewcommand*\DTMdisplaytime[3]{%
      \number##1
      \DTMsep{hourmin}\DTMtwodigits{##2}%
      \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
    }%
 }%
```

Define `map` zone style

```
\DTMnewzonestyle
 {map}% label
 {%
    \renewcommand*\DTMdisplaytime[3]{%
      \DTMusezonemapordefault{##1}{##2}%
    }%
 }%
```

Define `hhmm` zone style

```
\DTMnewzonestyle
 {hhmm}% label
```

```
{%
   \renewcommand*\DTMdisplaytime[3]{%
     \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
     \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
   }%
}
```

9.1.3 Saving and Using Dates

Date and time information is stored in control sequences in the form `\@dtm@`⟨*label*⟩`@`⟨*tag*⟩, where ⟨*label*⟩ is the label uniquely identifying the information and ⟨*tag*⟩ is the element (`year`, `month`, `day`, `dow`, `hour`, `minute`, `second`, `TZhour` and `TZminute`). Missing information is stored as `0` (except for the day of week, which is stored as `-1`).

`\DTMsavedate` Save the date specified in the format ⟨*yyyy*⟩-⟨*mm*⟩-⟨*dd*⟩. `\expandafter` is used in case the argument is a control sequence storing the date. This will redefine an existing saved date with the same label. The first argument is the label.

```
\newrobustcmd*{\DTMsavedate}[2]{%
  \expandafter\@dtm@parsedate#2\@dtm@endparsedate
  \cslet{@dtm@#1@year}{\@dtm@year}%
  \cslet{@dtm@#1@month}{\@dtm@month}%
  \cslet{@dtm@#1@day}{\@dtm@day}%
  \cslet{@dtm@#1@dow}{\@dtm@dow}%
  \ifcsundef{@dtm@#1@hour}{\csdef{@dtm@#1@hour}{0}}{}%
  \ifcsundef{@dtm@#1@minute}{\csdef{@dtm@#1@minute}{0}}{}%
  \ifcsundef{@dtm@#1@second}{\csdef{@dtm@#1@second}{0}}{}%
  \ifcsundef{@dtm@#1@TZhour}{\csdef{@dtm@#1@TZhour}{0}}{}%
  \ifcsundef{@dtm@#1@TZminute}{\csdef{@dtm@#1@TZminute}{0}}{}%
}
```

`\DTMsavenoparsedate` Save the date without parsing the ⟨*YYYY*⟩-⟨*MM*⟩-⟨*DD*⟩ format.

```
\newrobustcmd*{\DTMsavenoparsedate}[5]{%
  \csedef{@dtm@#1@year}{\number#2}%
  \csedef{@dtm@#1@month}{\number#3}%
  \csedef{@dtm@#1@day}{\number#4}%
  \csedef{@dtm@#1@dow}{\number#5}%
  \ifcsundef{@dtm@#1@hour}{\csdef{@dtm@#1@hour}{0}}{}%
  \ifcsundef{@dtm@#1@minute}{\csdef{@dtm@#1@minute}{0}}{}%
  \ifcsundef{@dtm@#1@second}{\csdef{@dtm@#1@second}{0}}{}%
  \ifcsundef{@dtm@#1@TZhour}{\csdef{@dtm@#1@TZhour}{0}}{}%
  \ifcsundef{@dtm@#1@TZminute}{\csdef{@dtm@#1@TZminute}{0}}{}%
}
```

`\DTMsavetime` Save the time specified in the format ⟨*hh*⟩:⟨*mm*⟩:⟨*ss*⟩. `\expandafter` is used in case the argument is a control sequence storing the date. This will redefine an existing saved date with the same label. The first argument is the label.

```
\newrobustcmd*{\DTMsavetime}[2]{%
```

```
  \expandafter\@dtm@parsetime#2\@dtm@endparsetime
  \cslet{@dtm@#1@hour}{\@dtm@hour}%
  \cslet{@dtm@#1@minute}{\@dtm@minute}%
  \cslet{@dtm@#1@second}{\@dtm@second}%
  \ifcsundef{@dtm@#1@year}{\csdef{@dtm@#1@year}{0}}{}%
  \ifcsundef{@dtm@#1@month}{\csdef{@dtm@#1@month}{0}}{}%
  \ifcsundef{@dtm@#1@day}{\csdef{@dtm@#1@day}{0}}{}%
  \ifcsundef{@dtm@#1@dow}{\csdef{@dtm@#1@dow}{-1}}{}%
  \ifcsundef{@dtm@#1@TZhour}{\csdef{@dtm@#1@TZhour}{0}}{}%
  \ifcsundef{@dtm@#1@TZminute}{\csdef{@dtm@#1@TZminute}{0}}{}%
}
```

`\DTMsavetimezn` Save the time (including zone) specified in the format ⟨*hh*⟩:⟨*mm*⟩:⟨*ss*⟩ ⟨*tzh*⟩:⟨*tzm*⟩. `\expandafter` is used in case the argument is a control sequence storing the date. This will redefine an existing saved date with the same label. The first argument is the label.

```
\newrobustcmd*{\DTMsavetimezn}[2]{%
  \expandafter\@dtm@parsetimezn#2\@dtm@endparsetimezn
  \cslet{@dtm@#1@hour}{\@dtm@hour}%
  \cslet{@dtm@#1@minute}{\@dtm@minute}%
  \cslet{@dtm@#1@second}{\@dtm@second}%
  \cslet{@dtm@#1@TZhour}{\@dtm@timezonehour}%
  \cslet{@dtm@#1@TZminute}{\@dtm@timezoneminute}%
  \ifcsundef{@dtm@#1@year}{\csdef{@dtm@#1@year}{0}}{}%
  \ifcsundef{@dtm@#1@month}{\csdef{@dtm@#1@month}{0}}{}%
  \ifcsundef{@dtm@#1@day}{\csdef{@dtm@#1@day}{0}}{}%
  \ifcsundef{@dtm@#1@dow}{\csdef{@dtm@#1@dow}{-1}}{}%
}
```

`\DTMsavetimestamp` Save the time (including zone) specified in the format ⟨*YYYY*⟩-⟨*MM*⟩-⟨*DD*⟩T⟨*hh*⟩:⟨*mm*⟩:⟨*ss*⟩⟨*time zone*⟩

```
\newrobustcmd*{\DTMsavetimestamp}[2]{%
  \expandafter\@dtm@parsetimestamp#2\@dtm@endparsetimestamp
  \cslet{@dtm@#1@year}{\@dtm@year}%
  \cslet{@dtm@#1@month}{\@dtm@month}%
  \cslet{@dtm@#1@day}{\@dtm@day}%
  \cslet{@dtm@#1@dow}{\@dtm@dow}%
  \cslet{@dtm@#1@hour}{\@dtm@hour}%
  \cslet{@dtm@#1@minute}{\@dtm@minute}%
  \cslet{@dtm@#1@second}{\@dtm@second}%
  \cslet{@dtm@#1@TZhour}{\@dtm@timezonehour}%
  \cslet{@dtm@#1@TZminute}{\@dtm@timezoneminute}%
}
```

`\DTMsavenow` Save the current time.

```
\newrobustcmd{\DTMsavenow}[1]{%
  \cslet{@dtm@#1@year}{\@dtm@currentyear}%
  \cslet{@dtm@#1@month}{\@dtm@currentmonth}%
```

```
  \cslet{@dtm@#1@day}{\@dtm@currentday}%
  \cslet{@dtm@#1@dow}{\@dtm@currentdow}%
  \cslet{@dtm@#1@hour}{\@dtm@currenthour}%
  \cslet{@dtm@#1@minute}{\@dtm@currentminute}%
  \cslet{@dtm@#1@second}{\@dtm@currentsecond}%
  \cslet{@dtm@#1@TZhour}{\@dtm@currenttimezonehour}%
  \cslet{@dtm@#1@TZminute}{\@dtm@currenttimezoneminute}%
}
```

\DTMmakeglobal Globally set the stored information.

```
\newrobustcmd{\DTMmakeglobal}[1]{%
  \global\csletcs{@dtm@#1@year}{@dtm@#1@year}%
  \global\csletcs{@dtm@#1@month}{@dtm@#1@month}%
  \global\csletcs{@dtm@#1@day}{@dtm@#1@day}%
  \global\csletcs{@dtm@#1@dow}{@dtm@#1@dow}%
  \global\csletcs{@dtm@#1@hour}{@dtm@#1@hour}%
  \global\csletcs{@dtm@#1@minute}{@dtm@#1@minute}%
  \global\csletcs{@dtm@#1@second}{@dtm@#1@second}%
  \global\csletcs{@dtm@#1@TZhour}{@dtm@#1@TZhour}%
  \global\csletcs{@dtm@#1@TZminute}{@dtm@#1@TZminute}%
}
```

Expandable ways of fetching saved data. (No check for existence performed.) The argument is the label.

\DTMfetchyear

```
\newcommand*{\DTMfetchyear}[1]{\csname @dtm@#1@year\endcsname}
```

\DTMfetchmonth

```
\newcommand*{\DTMfetchmonth}[1]{\csname @dtm@#1@month\endcsname}
```

\DTMfetchday

```
\newcommand*{\DTMfetchday}[1]{\csname @dtm@#1@day\endcsname}
```

\DTMfetchdow

```
\newcommand*{\DTMfetchdow}[1]{\csname @dtm@#1@dow\endcsname}
```

\DTMfetchhour

```
\newcommand*{\DTMfetchhour}[1]{\csname @dtm@#1@hour\endcsname}
```

\DTMfetchminute

```
\newcommand*{\DTMfetchminute}[1]{\csname @dtm@#1@minute\endcsname}
```

\DTMfetchsecond

```
\newcommand*{\DTMfetchsecond}[1]{\csname @dtm@#1@second\endcsname}
```

\DTMfetchTZhour

```
\newcommand*{\DTMfetchTZhour}[1]{\csname @dtm@#1@TZhour\endcsname}
```

\DTMfetchTZminute

```
\newcommand*{\DTMfetchTZminute}[1]{\csname @dtm@#1@TZminute\endcsname}
```

\DTMusedate `\DTMusedate{⟨label⟩}`

Displays the previously saved date using \DTMdisplaydate.

```
\newcommand*\DTMusedate[1]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2}{Undefined date `#1'}{}%
  }%
  {%
     \DTMdisplaydate
      {\csname @dtm@#1@year\endcsname}%
      {\csname @dtm@#1@month\endcsname}%
      {\csname @dtm@#1@day\endcsname}%
      {\csname @dtm@#1@dow\endcsname}%
  }%
}%
```

\DTMUsedate `\DTMUsedate{⟨label⟩}`

Displays the previously saved date using \DTMDisplaydate.

```
\newcommand*\DTMUsedate[1]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2}{Undefined date `#1'}{}%
  }%
  {%
     \DTMDisplaydate
      {\csname @dtm@#1@year\endcsname}%
      {\csname @dtm@#1@month\endcsname}%
      {\csname @dtm@#1@day\endcsname}%
      {\csname @dtm@#1@dow\endcsname}%
  }%
}%
```

\DTMusetime `\DTMusetime{⟨label⟩}`

Displays the previously saved time using \DTMdisplaytime.

```
\newcommand*\DTMusetime[1]{%
```

```
  \ifcsundef{@dtm@#1@hour}%
  {%
     \PackageError{datetime2}{Undefined time `#1'}{}%
  }%
  {%
     \DTMdisplaytime
      {\csname @dtm@#1@hour\endcsname}%
      {\csname @dtm@#1@minute\endcsname}%
      {\csname @dtm@#1@second\endcsname}%
  }%
}%
```

\DTMusezone `\DTMusezone{⟨label⟩}`

Displays the previously saved date using \DTMdisplayzone.

```
\newcommand*\DTMusezone[1]{%
  \ifcsundef{@dtm@#1@TZhour}%
  {%
     \PackageError{datetime2}{Undefined time `#1'}{}%
  }%
  {%
     \DTMdisplayzone
      {\csname @dtm@#1@TZhour\endcsname}%
      {\csname @dtm@#1@TZminute\endcsname}%
  }%
}%
```

\DTMuse `\DTMuse{⟨label⟩}`

Displays the previously saved date and time.

```
\newcommand*\DTMuse[1]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2}{Undefined date-time `#1'}{}%
  }%
  {%
     \DTMdisplay
      {\csname @dtm@#1@year\endcsname}%
      {\csname @dtm@#1@month\endcsname}%
      {\csname @dtm@#1@day\endcsname}%
      {\csname @dtm@#1@dow\endcsname}%
      {\csname @dtm@#1@hour\endcsname}%
      {\csname @dtm@#1@minute\endcsname}%
      {\csname @dtm@#1@second\endcsname}%
```

```
      {\csname @dtm@#1@TZhour\endcsname}%
      {\csname @dtm@#1@TZminute\endcsname}%
  }%
}%
```

\DTMUse `\DTMUse{⟨label⟩}`

Displays the previously saved date and time.

```
\newcommand*\DTMUse[1]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2}{Undefined date-time `#1'}{}%
  }%
  {%
     \DTMDisplay
      {\csname @dtm@#1@year\endcsname}%
      {\csname @dtm@#1@month\endcsname}%
      {\csname @dtm@#1@day\endcsname}%
      {\csname @dtm@#1@dow\endcsname}%
      {\csname @dtm@#1@hour\endcsname}%
      {\csname @dtm@#1@minute\endcsname}%
      {\csname @dtm@#1@second\endcsname}%
      {\csname @dtm@#1@TZhour\endcsname}%
      {\csname @dtm@#1@TZminute\endcsname}%
  }%
}%
```

\DTMifsaveddate Determine if the given label has been assigned to a date, time and zone.

```
\newcommand{\DTMifsaveddate}[3]{%
  \ifcsundef{@dtm@#1@year}{#3}{#2}%
}
```

9.1.4 Language Module Loading

Define commands to load regional settings.

\@dtm@requiremodule Use tracklang interface to find the associated file for the given dialect.

```
\newcommand*{\@dtm@requiremodule}[1]{%
  \IfTrackedLanguageFileExists{#1}%
  {datetime2-}% prefix
  {.ldf}% suffix
  {%
    \RequireDateTimeModule{\CurrentTrackedTag}%
  }%
  {%
    \@dtm@warning{Date-Time Language Module `#1' not installed}%
```

```
  }%
}
```

\@dtm@loadedregions List of loaded datetime2 language modules.

```
\newcommand*{\@dtm@loadedregions}{}
```

quireDateTimeModule Input the language file, if not already loaded. Should only be used with \@dtm@requiremodule which sets commands like \CurrentTrackedDialect. Since the language modules are loaded within \@dtm@requiremodule they may use this command to load dependent modules.

```
\newcommand*{\RequireDateTimeModule}[1]{%
 \ifundef\CurrentTrackedDialect
 {%
   \PackageError{datetime2}%
   {\string\RequireDateTimeModule\space not permitted here}%
   {This command is only permitted inside datetime2 language
    modules.}%
 }%
 {%
   \ifcsundef{ver@datetime2-#1.ldf}%
   {%
     \input{datetime2-#1.ldf}%
     \ifdefempty\@dtm@loadedregions
     {%
       \edef\@dtm@loadedregions{#1}%
     }%
     {%
       \edef\@dtm@loadedregions{\@dtm@loadedregions,#1}%
     }%
   }%
   {}%
 }%
}
```

videsDateTimeModule For use in language module to identify itself.

```
\newcommand*{\ProvidesDateTimeModule}[1]{%
  \ProvidesFile{datetime2-#1.ldf}%
}
```

\DTMdefkey `\DTMdefkey{⟨region⟩}{⟨key⟩}[⟨default⟩]{⟨func⟩}`

Used by language modules to define a key.

```
\newcommand*{\DTMdefkey}[1]{\define@key[dtm]{#1}}
```

\DTMdefchoicekey

\DTMdefchoicekey{⟨*region*⟩}{⟨*key*⟩}[⟨*bin*⟩]{⟨*choice list*⟩}{⟨*default*⟩}{⟨*func*⟩}

Used by language modules to define a choice key.

```
\newcommand*{\DTMdefchoicekey}[1]{\define@choicekey[dtm]{#1}}
```

\DTMdefboolkey

\DTMdefboolkey{⟨*region*⟩}[⟨*mp*⟩]{⟨*key*⟩}[⟨*default*⟩]{⟨*func*⟩}

Used by language modules to define a boolean key.

```
\newcommand*{\DTMdefboolkey}[1]{\define@boolkey[dtm]{#1}}
```

\DTMifbool

\DTMifbool{⟨*region*⟩}{⟨*key*⟩}{⟨*true*⟩}{⟨*false*⟩}

Test boolean key that was defined using \DTMdefboolkey

```
\newcommand*{\DTMifbool}[4]{\ifbool{dtm@#1@#2}{#3}{#4}}
```

\DTMsetbool

\DTMsetbool{⟨*region*⟩}{⟨*key*⟩}{⟨*value*⟩}

Set boolean key that was defined using \DTMdefboolkey

```
\newcommand*{\DTMsetbool}[3]{\setbool{dtm@#1@#2}{#3}}
```

\DTMlangsetup

Set up options for language modules. The optional argument is a list of language/regions. If omitted all loaded regions are iterated over. (I'm not sure why \setkeys doesn't work if the same key is present in multiple families, so this iterates over the families instead.)

```
\newcommand*{\DTMlangsetup}[2][\@dtm@loadedregions]{%
 \@for\@dtm@region:=#1\do{%
   \setkeys*+[dtm]{\@dtm@region}{#2}%
   \ifdefempty\XKV@rm{}%
   {%
     \@dtm@warning{Region `\@dtm@region' has ignored
      \MessageBreak the following settings:\MessageBreak
      \XKV@rm
      ^^J}%
   }%
 }%
}
```

Now load all the required modules (if installed) using the tracklang interface. (Language packages, such as babel or polyglossia must be loaded before this.)

```
\AnyTrackedLanguages
{%
  \ForEachTrackedDialect{\this@dialect}%
  {%
    \@dtm@requiremodule\this@dialect
  }%
}
{%
```

No tracked languages. The default is already set up, so nothing to do here.

```
}
```

Load datetime2-calc if required.

```
\@dtm@usecalc
```

Use the style package option, if set.

```
\ifdefempty\@dtm@initialstyle{}{\DTMsetstyle{\@dtm@initialstyle}}
```

9.2 datetime2-calc.sty code

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{datetime2-calc}[2015/09/15 v1.1 (NLCT)]
```

Load other required packages

```
\RequirePackage{pgfkeys}
\RequirePackage{pgfcalendar}
```

`\@dtm@julianday` Register for storing Julian day number.

```
\newcount\@dtm@julianday
```

`\@dtm@parsedate` Redefine `\@dtm@parsedate` so that it uses pgfcalendar to compute the required information. This allows for offsets, the use of `last` and also determine the day of week.

```
\def\@dtm@parsedate#1-#2-#3\@dtm@endparsedate{%
  \pgfcalendardatetojulian{#1-#2-#3}{\@dtm@julianday}%
  \pgfcalendarjuliantodate{\@dtm@julianday}{\@dtm@year}{\@dtm@month}{\@dtm@day}%
  \pgfcalendarjuliantoweekday{\@dtm@julianday}{\count@}%
  \edef\@dtm@dow{\number\count@}%
}
```

Set the current day of week

`\@dtm@currentdow`

```
\pgfcalendardatetojulian
 {\@dtm@currentyear-\@dtm@currentmonth-\@dtm@currentday}%
 {\@dtm@julianday}%
\pgfcalendarjuliantoweekday{\@dtm@julianday}{\count@}%
\edef\@dtm@currentdow{\number\count@}%
```

\DTMsavejulianday Save the date obtained from the Julian day number.

```
\newrobustcmd*{\DTMsavejulianday}[2]{%
  \pgfcalendarjuliantodate{#2}{\@dtm@year}{\@dtm@month}{\@dtm@day}%
  \pgfcalendarjuliantoweekday{#2}{\count@}%
  \csedef{@dtm@#1@dow}{\number\count@}%
  \cslet{@dtm@#1@year}{\@dtm@year}%
  \cslet{@dtm@#1@month}{\@dtm@month}%
  \cslet{@dtm@#1@day}{\@dtm@day}%
  \ifcsundef{@dtm@#1@hour}{\csdef{@dtm@#1@hour}{0}}{}%
  \ifcsundef{@dtm@#1@minute}{\csdef{@dtm@#1@minute}{0}}{}%
  \ifcsundef{@dtm@#1@second}{\csdef{@dtm@#1@second}{0}}{}%
  \ifcsundef{@dtm@#1@TZhour}{\csdef{@dtm@#1@TZhour}{0}}{}%
  \ifcsundef{@dtm@#1@TZminute}{\csdef{@dtm@#1@TZminute}{0}}{}%
}
```

aveddatetojulianday Converts a saved date to a Julian day number. The first argument is the name referencing the saved date, the second is a count register in which to store the result.

```
\newrobustcmd*{\DTMsaveddatetojulianday}[2]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2-calc}{Unknown date `#1'}{}%
  }%
  {%
    \pgfcalendardatetojulian
     {\csname @dtm@#1@year\endcsname
     -\csname @dtm@#1@month\endcsname
     -\csname @dtm@#1@day\endcsname}
     {#2}%
  }%
}
```

teoffsettojulianday Converts an offset from the saved date to a Julian day number. The first argument is the name referencing the saved date, the second is the offset increment and the third is a count register in which to store the result.

```
\newrobustcmd*{\DTMsaveddateoffsettojulianday}[3]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2-calc}{Unknown date `#1'}{}%
  }%
  {%
    \pgfcalendardatetojulian
     {\csname @dtm@#1@year\endcsname
     -\csname @dtm@#1@month\endcsname
     -\csname @dtm@#1@day\endcsname
     +#2}
     {#3}%
  }%
}
```

\DTMifdate Test a saved date using \pgfcalendarifdate

```
\newrobustcmd*{\DTMifdate}[4]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2-calc}{Unknown date `#1'}{}%
  }%
  {%
    \pgfcalendarifdate
     {\csname @dtm@#1@year\endcsname
     -\csname @dtm@#1@month\endcsname
     -\csname @dtm@#1@day\endcsname}
     {#2}{#3}{#4}%
  }%
}
```

\DTMsaveddatediff Computes the difference between two saved dates. The result is stored in the third argument, which should be a count register.

```
\newrobustcmd*{\DTMsaveddatediff}[3]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2-calc}{Unknown date `#1'}{}%
  }%
  {%
     \ifcsundef{@dtm@#2@year}%
     {%
        \PackageError{datetime2-calc}{Unknown date `#1'}{}%
     }%
     {%
       \pgfcalendardatetojulian
        {\csname @dtm@#1@year\endcsname
        -\csname @dtm@#1@month\endcsname
        -\csname @dtm@#1@day\endcsname}
        {#3}%
       \pgfcalendardatetojulian
        {\csname @dtm@#2@year\endcsname
        -\csname @dtm@#2@month\endcsname
        -\csname @dtm@#2@day\endcsname}
        {\@dtm@julianday}%
       \advance#3 by -\@dtm@julianday\relax
    }
  }%
}
```

\DTMtozulu Converts the datetime data referenced by the first argument into Zulu time and saves it to data referenced by the second argument.

```
\newrobustcmd*{\DTMtozulu}[2]{%
  \ifcsundef{@dtm@#1@year}%
  {%
     \PackageError{datetime2-calc}{Unknown date `#1'}{}%
```

```
  }%
  {%
    \DTMsaveaszulutime{#2}%
    {\DTMfetchyear{#1}}%
    {\DTMfetchmonth{#1}}%
    {\DTMfetchday{#1}}%
    {\DTMfetchhour{#1}}%
    {\DTMfetchminute{#1}}%
    {\DTMfetchsecond{#1}}%
    {\DTMfetchTZhour{#1}}%
    {\DTMfetchTZminute{#1}}%
  }%
}
```

`\DTMsaveaszulutime` Converts the given datetime into Zulu (+00:00) and saves the result.

`\DTMsavetozulutime{`⟨*name*⟩`}{`⟨*year*⟩`}{`⟨*month*⟩`}{`⟨*day*⟩`}{`⟨*hour*⟩`}{`⟨*minute*⟩`}{`⟨*second*⟩`}{`⟨*tzh*⟩`}{`⟨*tzm*⟩`}`

```
\newrobustcmd*{\DTMsaveaszulutime}[9]{%
  \edef\@dtm@year{\number#2}%
  \edef\@dtm@month{\number#3}%
  \edef\@dtm@day{\number#4}%
  \edef\@dtm@hour{\number#5}%
  \edef\@dtm@minute{\number#6}%
  \edef\@dtm@second{\number#7}%
  \edef\@dtm@TZhour{\number#8}%
  \edef\@dtm@TZminute{\number#9}%
  \pgfcalendardatetojulian{\@dtm@year-\@dtm@month-\@dtm@day}{\@dtm@julianday}%
```

First adjust the minute offset if non-zero

```
  \ifnum\@dtm@TZminute=0\relax
  \else
    \count@=\@dtm@minute\relax
```

Add or subtract the offset minute

```
    \ifnum\@dtm@TZhour<0\relax
      \advance\count@ by \@dtm@TZminute\relax
    \else
      \advance\count@ by -\@dtm@TZminute\relax
    \fi
    \edef\@dtm@minute{\number\count@}%
```

Does the hour need adjusting?

```
    \ifnum\count@<0\relax
      \advance\count@ by 60\relax
      \edef\@dtm@minute{\number\count@}%
```

Need to subtract 1 from the hour but does the day need adjusting?

```
      \ifnum\@dtm@hour=0\relax
       \def\@dtm@hour{23}%
```
Day needs adjusting.
```
       \advance\@dtm@julianday by -1\relax
      \else
```
Subtract 1 from the hour
```
        \count@ = \@dtm@hour\relax
        \advance\count@ by -1\relax
        \edef\@dtm@hour{\number\count@}%
      \fi
    \else
```
Minute isn't negative. Is it ≥ 60?
```
      \ifnum\count@>59\relax
        \advance\count@ by -60\relax
        \edef\@dtm@minute{\number\count@}%
```
Add 1 to the hour
```
        \count@ = \@dtm@hour\relax
        \advance\count@ by 1\relax
        \edef\@dtm@hour{\number\count@}%
```
Does the day need adjusting?
```
        \ifnum\@dtm@hour=24\relax
          \def\@dtm@hour{00}%
          \advance\@dtm@julianday by 1\relax
        \fi
      \fi
    \fi
  \fi
```
Now adjust the hour offset if non-zero
```
  \ifnum\@dtm@TZhour=0\relax
  \else
    \count@=\@dtm@hour\relax
    \advance\count@ by -\@dtm@TZhour\relax
```
Does the day need adjusting?
```
    \ifnum\count@<0\relax
      \advance\count@ by 24\relax
      \edef\@dtm@hour{\number\count@}%
      \advance\@dtm@julianday by -1\relax
    \else
      \ifnum\count@>23\relax
      \advance\count@ by -24\relax
      \edef\@dtm@hour{\number\count@}%
      \advance\@dtm@julianday by 1\relax
      \else
        \edef\@dtm@hour{\number\count@}%
      \fi
```

```
    \fi
  \fi
  \pgfcalendarjuliantodate{\@dtm@julianday}{\@dtm@year}{\@dtm@month}{\@dtm@day}%
  \pgfcalendarjuliantoweekday{\@dtm@julianday}{\count@}%
```

Save the results.

```
  \csedef{@dtm@#1@dow}{\number\count@}%
  \cslet{@dtm@#1@year}{\@dtm@year}%
  \cslet{@dtm@#1@month}{\@dtm@month}%
  \cslet{@dtm@#1@day}{\@dtm@day}%
  \cslet{@dtm@#1@hour}{\@dtm@hour}%
  \cslet{@dtm@#1@minute}{\@dtm@minute}%
  \cslet{@dtm@#1@second}{\@dtm@second}%
  \csdef{@dtm@#1@TZhour}{0}%
  \csdef{@dtm@#1@TZminute}{0}%
}
```

Index