

datetime2 v1.2: date and time formats

Nicola L. C. Talbot

<http://www.dickimaw-books.com/>

2015-11-10

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	4
2	Example Usage	5
3	Displaying the Date and Time	8
4	Storing and Using Dates and Times	13
5	Styles	18
5.1	Predefined Styles	19
5.1.1	Full Styles	20
5.1.2	Time Styles	22
5.1.3	Zone Styles	22
5.2	Defining New Styles	23
6	Multi-Lingual Support	29
7	Package Options	36
8	The datetime2-calc Package	42
9	The Code	46
9.1	datetime2.sty code	46
9.1.1	Defaults	51
9.1.2	Styles	59
9.1.3	Saving and Using Dates	78
9.1.4	Language Module Loading	84
9.2	datetime2-calc.sty code	86
	Index	92

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 PDF_{TEX} 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 _{TEX} 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.

¹Of 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 Xe_{La}TeX in which case the seconds and time zone are omitted. (Xe_{La}TeX 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 even though it's a language setting. (This is because the `datetime2-calc` package is also needed if the day of the week should be displayed. As a package option, `showdow` can automatically load the required package.)

You need to check the documentation for the relevant language module or package 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 $\langle YYYY \rangle$ can't be negative¹ in $\backslash DTMdate$ or $\backslash DTMDate$. Use $\backslash DTMdisplaydate$ or $\backslash DTMDisplaydate$ instead.

These commands internally use $\backslash DTMdisplaydate$ and $\backslash DTMDisplaydate$, 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:

- $\langle YYYY \rangle - \langle MM \rangle - \text{last}$ (the last day of the given month).
- $\langle YYYY \rangle - \langle MM \rangle - \langle DD \rangle + - \langle n \rangle$ ($\langle n \rangle$ days before the given date).
- $\langle YYYY \rangle - \langle MM \rangle - \langle DD \rangle + \langle n \rangle$ ($\langle n \rangle$ days after the given date).
- $\langle YYYY \rangle - \langle MM \rangle - \text{last} + - \langle n \rangle$ ($\langle n \rangle$ days before the last day of the given month).
- $\langle YYYY \rangle - \langle MM \rangle - \text{last} + \langle n \rangle$ ($\langle n \rangle$ 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 $\backslash DTMuse$ 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 $\backslash DTMdisplaydate$ and $\backslash DTMDisplaydate$ (which $\backslash DTMdate$ and $\backslash DTMDate$ internally use, as mentioned above).

A time can be displayed using

$\backslash DTMdisplaytime$

```
\DTMdisplaytime{\langle hour \rangle}{\langle minute \rangle}{\langle sec \rangle}
```

¹Well, 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 PDF_T_E_X primitive `\pdfcreationdate` is defined, the current time information is obtained from that, which includes the seconds and time zone. Lua_T_E_X also defines this command but X_Y_T_E_X doesn't, and in that case the only way to determine the current time is from T_E_X's `\time` primitive which only contains the number of minutes since midnight, which means that the seconds and time zone are unavailable. Therefore if X_Y_T_E_X 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 `showdoc` 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 PDF_WTeX, this command will use the PDF_WTeX primitive `\pdffilemoddate`. If you use Lua_WTeX 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 Xe_WTeX 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 $\langle name \rangle$ hasn't been defined. Alternatively,

`\DTMUse`

```
\DTMUse{\langle name \rangle}
```

will use `\DTMdisplay` instead.

You can determine if a given $\langle name \rangle$ has been defined using

`\DTMifsaveddate`

```
\DTMifsaveddate{\langle name \rangle}{\langle true \rangle}{\langle false \rangle}
```

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

`\DTMfetchyear`

```
\DTMfetchyear{\langle name \rangle}
```

This expands to the year.

`\DTMfetchmonth`

```
\DTMfetchmonth{\langle name \rangle}
```

This expands to the month number.

`\DTMfetchday`

```
\DTMfetchday{\langle name \rangle}
```

This expands to the day of the month.

`\DTMfetchdow`

```
\DTMfetchdow{\langle name \rangle}
```

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

`\DTMfetchhour`

```
\DTMfetchhour{\langle name \rangle}
```

This expands to the hour.

`\DTMfetchminute`

```
\DTMfetchminute{\langle name \rangle}
```

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 $\langle name \rangle$ 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 $\langle name \rangle$ 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 $\langle name \rangle$.

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

$$\langle YYYY \rangle \langle YMsep \rangle \langle MM \rangle \langle MDsep \rangle \langle DD \rangle$$

where the month $\langle MM \rangle$ and day of the month $\langle DD \rangle$ numbers are formatted as two digits. The separators $\langle YMsep \rangle$ and $\langle MDsep \rangle$ 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:

$$\langle hh \rangle \langle HMsep \rangle \langle mm \rangle \langle MSsep \rangle \langle ss \rangle$$

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

The zone is displayed in form

$$\langle TZh \rangle \langle HMsep \rangle \langle TZm \rangle$$

or just `Z` if the option `showisoZ` is set to `true` and both $\langle TZh \rangle$ and $\langle TZm \rangle$ are zero. The separator $\langle HMsep \rangle$ is the same as used for the time format. The final $\langle HMsep \rangle \langle TZm \rangle$ is omitted if the option `showzoneminutes` is set to `false`. The hour offset $\langle TZh \rangle$ is formatted as two digits preceded 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

$$\langle date \rangle \langle DTsep \rangle \langle time \rangle \langle TZsep \rangle \langle zone \rangle$$

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 `dmyyyy` 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

$\langle TZh \rangle \langle HMsep \rangle \langle TZm \rangle$

where $\langle HMsep \rangle$ 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{\langle name \rangle}{\langle definition \rangle}
```

This defines a partial style that should only modify `\DTMdisplaydate` and `\DTMdisplaydate`. The redefinition of these commands should be placed in $\langle definition \rangle$.

A new time style can be defined using:

`\DTMnewtimestyle`

```
\DTMnewtimestyle{\langle name \rangle}{\langle definition \rangle}
```

This defines a partial style that should only modify `\DTMdisplaytime`. The redefinition should be placed in $\langle definition \rangle$.

A new zone style can be defined using:

`\DTMnewzonestyle`

```
\DTMnewzonestyle{\langle name \rangle}{\langle definition \rangle}
```

This defines a partial style that should only modify `\DTMdisplayzone`. The redefinition should be placed in $\langle definition \rangle$.

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.

As from version 1.2, you can redefine existing styles with the following commands.

`\DTMrenewdatestyle`

```
\DTMrenewdatestyle{<name>}{<definition>}
```

This redefines the named date style. The original may be either a partial or a full style.

`\DTMrenewtimestyle`

```
\DTMrenewtimestyle{<name>}{<definition>}
```

This redefines the named time style. The original may be either a partial or a full style.

`\DTMrenewzonestyle`

```
\DTMrenewzonestyle{<name>}{<definition>}
```

This redefines the named time zone style. The original may be either a partial or a full style.

`\DTMrenewstyle`

```
\DTMrenewstyle{<name>}{<definition>}
```

This redefines the named full style. The original style must also be a full style.

There are also commands analogous to `\providecommand` that will define styles that don't already exist.

`\DTMprovidedatestyle`

```
\DTMprovidedatestyle{<name>}{<definition>}
```

This defines the named date style. This won't do anything if either a partial date style or a full style with the given name already exists.

`\DTMprovidetimestyle`

```
\DTMprovidetimestyle{<name>}{<definition>}
```

This defines the named time style. This won't do anything if either a partial time style or a full style with the given name already exists.

`\DTMprovidezonestyle`

```
\DTMprovidezonestyle{<name>}{<definition>}
```

This defines the named zone style. This won't do anything if either a partial zone style or a full style with the given name already exists.

`\DTMprovidestyle`

```
\DTMprovidestyle{<name>}{<definition>}
```

This defines the named full style if the named full style doesn't already exist. This internally uses the previous three commands for the partial elements of the full style, so if a partial style with this name already exists, it won't be changed.

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 *exactly* two digits. Unlike `\two@digits` this will check for a negative number and will trim a number whose absolute value is greater than or equal to 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 $\lfloor \langle number \rangle / 100 \rfloor$ (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 $\langle TZh \rangle \langle HMsep \rangle \langle TZm \rangle$ where $\langle HMsep \rangle \langle TZm \rangle$ is omitted if the option `showzone-minutes` is set to `false`. The separator $\langle HMsep \rangle$ 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 `\datetime2`'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 `XYLaTeX` 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 provide 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 `\TeX` commands for non-ASCII characters (and remember to use `\protect` where necessary).

As an addendum to the above warning, `LuaTeX` and `XYTeX` support UTF-8 characters without the need to make them active, so I recommend you provide two files: one with the `\TeX` commands, such as `\c`, for (PDF)`\TeX` users, and one with UTF-8 characters for `LuaTeX` and `XYTeX` 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 Lua_{TeX} and Xe_{TeX} users. (See the Scottish or Irish modules.)

7 Package Options

The following package options are provided. Most of these are $\langle key \rangle = \langle value \rangle$ 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 $X_{\text{L}}\text{T}_{\text{E}}\text{X}$ 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 $X_{\text{L}}\text{T}_{\text{E}}\text{X}$ 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 $X_{\text{L}}\text{T}_{\text{E}}\text{X}$ 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 `usnumerical=text` but the `style` option then implements `usnumerical=false`.)

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

In the above `useregional` is `text`. (The `style` option implements `usnumerical=false` but the `british` option then implements `usnumerical=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. **Not all styles support this setting.** Default: false.

This option is actually a language-dependent option and isn't used by the base package, but it's implemented as a package option as the datetime2-calc package is also needed if the day of the week should be displayed. As a package option, showdow can automatically load the required package.

You need to check the documentation to find out which styles check the showdow setting as not all of them do.

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 pre-defined 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 `2015.11.10` 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 `2015.11.10` 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 2015-11-10 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 10.11.2015 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 10th November 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 10/11/2015 since the en-GB-numeric style is in use but datesep is used in \DTMsetup which influences the base pre-defined 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 `\pgfcalendarstartdate` to convert a previously saved date (identified by $\langle name \rangle$) to a Julian day. The result is stored in $\langle register \rangle$ 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 $\langle offset \rangle$. The result is stored in $\langle register \rangle$. This is equivalent to

```
\pgfcalendarstartdate{<y>-<m>-<d>+<offset>}{<register>}
```

where $\langle y \rangle$, $\langle m \rangle$ and $\langle d \rangle$ are the year, month and day fetched from the saved date. A negative $\langle offset \rangle$ 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 $\langle name \rangle$). 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 $\langle test \rangle$ 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`

```
\DTMsaveddatediff{<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 $\langle name1 \rangle$ and the second date is identified by $\langle name2 \rangle$. The dates are converted to their respective Julian day numbers $\langle J1 \rangle$ and $\langle J2 \rangle$ and the result is given by $\langle J1 \rangle - \langle J2 \rangle$.

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 date-time instance into Zulu¹ time (UTC+00:00).

\DTMsaveaszulutime

```
\DTMsaveaszulutime{\langle name \rangle}{\langle YYYY \rangle}{\langle MM \rangle}{\langle DD \rangle}{\langle hh \rangle}{\langle mm \rangle}
{\langle ss \rangle}{\langle TZh \rangle}{\langle TZm \rangle}
```

This converts the given datetime instance into UTC+00:00 and saves the result. You can then use the date with commands like \DTMusedate described in Section 4. The $\langle name \rangle$ 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{\langle name1 \rangle}{\langle name2 \rangle}
```

¹That'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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{datetime2}[2015/11/10 v1.2 (NLCT) date and time formats]
```

Use tracklang to find out what languages have been loaded.

```
3 \RequirePackage{tracklang}
```

Also require etoolbox.

```
4 \RequirePackage{etoolbox}
```

Need xkeyval for $\langle key \rangle = \langle value \rangle$ interface.

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

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

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

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

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

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

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

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

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

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

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

$\backslash dtm@timezonesep$ Separator between the date and time.

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

$\backslash dtm@timezonesep$ Separator between the time and time zone.

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

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

```
13 \define@key{datetime2.sty}{datesep}{%
14   \renewcommand*{\dtm@yearmonthsep}{#1}%
15   \renewcommand*{\dtm@monthdaysep}{#1}%
16   \renewcommand*{\dtm@dayyearsep}{#1}%
17 }
```

yearmonthsep Set year/month separator.

```

18 \define@key{datetime2.sty}{yearmonthsep}{%
19   \renewcommand*{\dtm@yearmonthsep}{#1}%
20 }

```

monthdaysep Set month/day separator.

```

21 \define@key{datetime2.sty}{monthdaysep}{%
22   \renewcommand*{\dtm@monthdaysep}{#1}%
23 }

```

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

```

24 \define@key{datetime2.sty}{dayyearsep}{%
25   \renewcommand*{\dtm@dayyearsep}{#1}%
26 }

```

timesep Set hour/minute and minute/second separator.

```

27 \define@key{datetime2.sty}{timesep}{%
28   \renewcommand*{\dtm@hourminsep}{#1}%
29   \renewcommand*{\dtm@minsecsep}{#1}%
30 }

```

hourminsep Set hour/minute separator.

```

31 \define@key{datetime2.sty}{hourminsep}{%
32   \renewcommand*{\dtm@hourminsep}{#1}%
33 }

```

minsecsep Set minute/second separator.

```

34 \define@key{datetime2.sty}{minsecsep}{%
35   \renewcommand*{\dtm@minsecsep}{#1}%
36 }

```

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

```

37 \define@key{datetime2.sty}{timezonesep}{%
38   \renewcommand*{\dtm@timezonesep}{#1}%
39 }

```

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

```

40 \define@key{datetime2.sty}{datetimesep}{%
41   \renewcommand*{\dtm@datetimesep}{#1}%
42 }

```

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

```

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

```

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

```

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

```

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

```
46 \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.)

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

```

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

```
49 \ifdef\pdfcreationdate
50 {%
51   \DTMshowsecondstrue
52   \DTMshowzonetrue
53 }%
54 {%
55   \DTMshowsecondsfalse
56   \DTMshowzonefalse
57 }%

```

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

```
58 \define@boolkey{datetime2.sty}[DTM]{showzoneminutes}[true]{}
59 \DTMshowzoneminutetrue

```

\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.

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

```

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

```
61 \define@choicekey{datetime2.sty}{useregional}[\val\nr]%
62 {false,text,numeric,num}[text]%
63 {%
64   \ifcase\nr\relax
65     \renewcommand*{\DTMifcaseregional}[3]{##1}%

```



```

66 \or
67 \renewcommand*\DTMifcaseregional}[3]{##2}%
68 \or
69 \renewcommand*\DTMifcaseregional}[3]{##3}%
70 \or
71 \renewcommand*\DTMifcaseregional}[3]{##3}%
72 \fi
73 }

```

\@dtm@setusecalc

```

74 \newcommand*\@dtm@setusecalc{%
75 \renewcommand*\@dtm@usecalc{\RequirePackage{datetime2-calc}}%
76 }

```

\@dtm@usecalc

```

77 \newcommand*\@dtm@usecalc{}

```

Disable attempt to load datetime2-calc in the document.

```

78 \AtBeginDocument{%
79 \ifpackageloaded{datetime2-calc}%
80 {%
81 \renewcommand*\@dtm@setusecalc{}%
82 }%
83 {%
84 \renewcommand*\@dtm@setusecalc{%
85 \PackageError{datetime2}{You must load 'datetime2-calc'
86 package to use option 'showdow'}{Try one of the following:^^J
87 pass 'calc' option to 'datetime2' package when you load it^^J
88 or move 'showdow' option to 'datetime2' package option list^^J
89 or move \string\DTLsetup\space to the preamble.}%
90 }%
91 }%
92 }

```

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.

```

93 \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.

```

94 \define@boolkey{datetime2.sty}[DTM]{showdow}[true]{%
95 \ifDTMshowdow \@dtm@setusecalc \fi
96 }
97 \DTMshowdowfalse

```

```

\@dtm@warning Warning messages.
98 \newcommand*{\@dtm@warning}[1]{%
99   \if@dtm@warn
100     \PackageWarning{datetime2}{#1}%
101   \fi
102 }

warn Allow user to suppress package warnings.
103 \define@boolkey{datetime2.sty}{@dtm@}{warn}[true]{}
104 \@dtm@warntrue

```

\@dtm@initialstyle

```

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

style Set the style. This automatically sets useregional=false.
106 \define@key{datetime2.sty}{style}{%
107   \renewcommand*{\@dtm@initialstyle}{#1}%
108   \ifstrempy{#1}%
109   {%
110     {%
111       \renewcommand*{\DTMifcaseregional}[3]{##1}%
112     }%
113 }

```

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

```

114 \DeclareOptionX*{%
115   \TrackPredefinedDialect{\CurrentOption}%
116   \renewcommand*{\DTMifcaseregional}[3]{#2}%
117 }

```

Process options passed to this package:

```
118 \ProcessOptionsX
```

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

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

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

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

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

\DTMsetup

```

121 \newcommand*{\DTMsetup}[1]{%
122   \def\@dtm@usecalc{}%
123   \setkeys{datetime2.sty}{#1}%
124   \@dtm@usecalc
125 }

```

9.1.1 Defaults

This section sets up the defaults.

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

```
126 \def\@dtm@parsedate#1-#2-#3\@dtm@endparsedate{%
127   \edef\@dtm@year{\number#1}%
128   \edef\@dtm@month{\number#2}%
129   \edef\@dtm@day{\number#3}%
130   \def\@dtm@dow{-1}%
131 }
```

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

```
132 \def\@dtm@parsetime#1:#2:#3\@dtm@endparsetime{%
133   \edef\@dtm@hour{\number#1}%
134   \edef\@dtm@minute{\number#2}%
135   \edef\@dtm@second{\number#3}%
136 }
```

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

```
137 \def\@dtm@parsetimezn#1:#2:#3 #4\@dtm@endparsetimezn{%
138   \@dtm@parsetime#1:#2:#3\@dtm@endparsetime
139   \@dtm@parsezone{#4}%
140 }
```

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

```
141 \newcommand*{\@dtm@parsezone}[1]{%
142   \ifstrequal{#1}{Z}%
143   {%
144     \def\@dtm@timezonehour{+00}%
145     \def\@dtm@timezoneminute{00}%
146   }%
147   {%
148     \@dtm@parse@zone#1\@dtm@endparse@zone
149   }%
150 }
151 \def\@dtm@parse@zone#1:#2\@dtm@endparse@zone{%
152   \edef\@dtm@timezonehour{\number#1}%
153   \edef\@dtm@timezoneminute{\number#2}%
154 }
```

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

```

155 \def\@dtm@parsetimestamp#1-#2-#3T#4:#5:#6#7#8\@dtm@endparsetimestamp{%
156   \@dtm@parsedate#1-#2-#3\@dtm@endparsedate
157   \@dtm@parsetime#4:#5:#6#7\@dtm@endparsetime
158   \@dtm@parsezone{#8}%
159 }

```

`\DTMsavefilemoddate` Not available for some engines.

```

160 \newcommand*{\DTMsavefilemoddate}[2]{%
161   \@dtm@warning{Your TeX engine doesn't support accessing
162     file modification dates}%
163   \cslet{\@dtm@#1@year}{0}%
164   \cslet{\@dtm@#1@month}{0}%
165   \cslet{\@dtm@#1@day}{0}%
166   \cslet{\@dtm@#1@dow}{-1}%
167   \cslet{\@dtm@#1@hour}{0}%
168   \cslet{\@dtm@#1@minute}{0}%
169   \cslet{\@dtm@#1@second}{0}%
170   \cslet{\@dtm@#1@TZhour}{0}%
171   \cslet{\@dtm@#1@TZminute}{0}%
172 }

```

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

```

173 \ifdef\pdfcreationdate
174 {%

```

Define commands to parse `\pdfcreationdate`

```

175   \def\@dtm@parsepdfdatetime#1:#2#3#4#5#6#7#8#9{%
176     \def\@dtm@year{#2#3#4#5}%
177     \def\@dtm@month{#6#7}%
178     \def\@dtm@day{#8#9}%
179     \@dtm@parsepdftime
180   }
181   \def\@dtm@parsepdftime#1#2#3#4#5#6#7\@dtm@endparsepdfdatetime{%
182     \def\@dtm@hour{#1#2}%
183     \def\@dtm@minute{#3#4}%
184     \def\@dtm@second{#5#6}%
185     \ifstrequal{#7}{Z}%
186     {%
187       \def\@dtm@timezonehour{00}%
188       \def\@dtm@timezoneminute{00}%
189     }%
190     {%
191       \@dtm@parsepdftimezone#7%
192     }%
193   }

```

```

194 \def\@dtm@parsepdfdatetimezone#1'#2' {%
195     \def\@dtm@timezonemonth{#1}%
196     \def\@dtm@timezoneminute{#2}%
197 }%

```

Now parse \pdfcreationdate

```

198 \expandafter\@dtm@parsepdfdatetime\pdfcreationdate\@dtm@endparsepdfdatetime

```

Save the values.

```

199 \let\@dtm@currentyear\@dtm@year
200 \let\@dtm@currentmonth\@dtm@month
201 \let\@dtm@currentday\@dtm@day
202 \let\@dtm@currenthour\@dtm@hour
203 \let\@dtm@currentminute\@dtm@minute
204 \let\@dtm@currentsecond\@dtm@second
205 \let\@dtm@currenttimezonemonth\@dtm@timezonemonth
206 \let\@dtm@currenttimezoneminute\@dtm@timezoneminute
207 %

```

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

```

208 \ifdef\pdffilemoddate
209 {%
210     \renewcommand*\DTMsavefilemoddate}[2] {%
211         \expandafter\@dtm@parsepdfdatetime\pdffilemoddate{#2}\@dtm@endparsepdfdatetime
212         \cslet{\@dtm@#1@year}{\@dtm@year}%
213         \cslet{\@dtm@#1@month}{\@dtm@month}%
214         \cslet{\@dtm@#1@day}{\@dtm@day}%
215         \cslet{\@dtm@#1@dow}{\@dtm@dow}%
216         \cslet{\@dtm@#1@hour}{\@dtm@hour}%
217         \cslet{\@dtm@#1@minute}{\@dtm@minute}%
218         \cslet{\@dtm@#1@second}{\@dtm@second}%
219         \cslet{\@dtm@#1@TZhour}{\@dtm@timezonemonth}%
220         \cslet{\@dtm@#1@TZminute}{\@dtm@timezoneminute}%
221     }
222 }%
223 {%

```

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

```

224 \ifdef\directlua
225 {
226     \renewcommand*\DTMsavefilemoddate}[2] {%
227         \expandafter\@dtm@parseluatdatetime
228         \directlua{tex.print(os.date(
229             "\expandafter\@gobble\string\%Y-%
230             \expandafter\@gobble\string\%m-%
231             \expandafter\@gobble\string\%d-%
232             \expandafter\@gobble\string\%w
233             \expandafter\@gobble\string\%H:%
234             \expandafter\@gobble\string\%M:%
235             \expandafter\@gobble\string\%S

```

```

236         \expandafter\@gobble\string\%z",
237         lfs.attributes("#2").modification))}%
238     \@dtm@endparseluadatetetime
239     \cslet{\@dtm@#1@year}{\@dtm@year}%
240     \cslet{\@dtm@#1@month}{\@dtm@month}%
241     \cslet{\@dtm@#1@day}{\@dtm@day}%
242     \cslet{\@dtm@#1@dow}{\@dtm@dow}%
243     \cslet{\@dtm@#1@hour}{\@dtm@hour}%
244     \cslet{\@dtm@#1@minute}{\@dtm@minute}%
245     \cslet{\@dtm@#1@second}{\@dtm@second}%
246     \cslet{\@dtm@#1@TZhour}{\@dtm@TZhour}%
247     \cslet{\@dtm@#1@TZminute}{\@dtm@TZminute}%
248 }
249 \def\@dtm@parseluadatetetime#1-#2-#3-#4 #5:#6:#7 #8\@dtm@endparseluadatetetime{%
250     \edef\@dtm@year{\number#1}%
251     \edef\@dtm@month{\number#2}%
252     \edef\@dtm@day{\number#3}%
253     \edef\@dtm@dow{\number#4}%
254     \edef\@dtm@hour{\number#5}%
255     \edef\@dtm@minute{\number#6}%
256     \edef\@dtm@second{\number#7}%
257     \@dtm@parseluatetimezone#8000000\@dtm@endparseluatetimezone
258 }
259 \def\@dtm@parseluatetimezone#1#2#3#4#5#6{%
260     \ifstrequal{#1}{+}%
261     {%
262         \def\@dtm@TZhour{#1#2#3}%
263         \ifstrequal{#4}{:}%
264         {%
265             \def\@dtm@TZminute{#5#6}%
266             }%
267         {%
268             \def\@dtm@TZminute{#4#5}%
269             }%
270     }%
271     {%
272         \ifstrequal{#1}{-}%
273         {%
274             \def\@dtm@TZhour{#1#2#3}%
275             \ifstrequal{#4}{:}%
276             {%
277                 \def\@dtm@TZminute{#5#6}%
278                 }%
279             {%
280                 \def\@dtm@TZminute{#4#5}%
281                 }%
282             }%
283         {%
284             \ifstrequal{#1}{Z}%

```

```

285         {%
286         \def\@dtm@TZhour{0}%
287         \def\@dtm@TZminute{0}%
288         }%
289         {%
290         \def\@dtm@TZhour{#1#2}%
291         \ifstrequal{#3}{:}%
292         {%
293         \def\@dtm@TZminute{#4#5}%
294         }%
295         {%
296         \def\@dtm@TZminute{#3#4}%
297         }%
298         }%
299         }%
300     }%
301     \@dtm@parseluatimezone
302 }
303 \def\@dtm@parseluatimezone#1\@dtm@endparseluatimezone{%
304 }
305 }
306 {}
307 }%
308 }%
309 {%

```

\pdfcreationdate not defined. By a process of elimination, the \TeX engine is either $X_{\mathcal{L}}\TeX$ or it's very old. (Lua \TeX 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.

```

310 \count@=\time\relax
311 \divide\count@ by 60\relax
312 \edef\@dtm@currenthour{\number\count@}%
313 \multiply\count@ by -60\relax
314 \advance\count@ by \time\relax
315 \edef\@dtm@currentminute{\number\count@}%
316 \newcommand*{\@dtm@currentsecond}{00}%
317 \newcommand\@dtm@currenttimezonehour{00}%
318 \newcommand\@dtm@currenttimezoneminute{00}%

```

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

```

319 \edef\@dtm@currentyear{\number\year}%
320 \edef\@dtm@currentmonth{\number\month}%
321 \edef\@dtm@currentday{\number\day}%
322 }

```

Make \DTMsavefilemoddate robust.

```

323 \robustify\DTMsavefilemoddate

```

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

\@dtm@currentdow

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

Allow Xe_{La}TeX users a way of manually setting the current time zone.

\DTMsetcurrentzone

```
325 \newcommand*{\DTMsetcurrentzone}[2]{%
326   \renewcommand\@dtm@currenttimezonehour{#1}%
327   \renewcommand\@dtm@currenttimezoneminute{#2}%
328 }
```

\today

```
329 \renewcommand*{\today}{%
330   \DTMdisplaydate
331   {\@dtm@currentyear}%
332   {\@dtm@currentmonth}%
333   {\@dtm@currentday}%
334   {\@dtm@currentdow}%
335 }
```

\Today First letter upper case version.

```
336 \newcommand*{\Today}{%
337   \DTMDisplaydate
338   {\@dtm@currentyear}%
339   {\@dtm@currentmonth}%
340   {\@dtm@currentday}%
341   {\@dtm@currentdow}%
342 }
```

\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.

```
343 \newcommand*{\DTMdisplaydate}[4]{%
344   \number#1\dtm@yearmonthsep\DTMtwodigits{#2}\dtm@monthdaysep\DTMtwodigits{#3}%
345 }
```

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

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

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

```
347 \newrobustcmd*{\DTMdate}[1]{%
348   \expandafter\@dtm@parsedate#1\@dtm@endparsedate
349   \DTMdisplaydate{\@dtm@year}{\@dtm@month}{\@dtm@day}{\@dtm@dow}%
350 }
```


\DTMDate Upper case version.

```
351 \newrobustcmd*{\DTMDate}[1]{%
352   \expandafter\@dtm@parsedate#1\@dtm@endparsedate
353   \DTMDisplaydate{\@dtm@year}{\@dtm@month}{\@dtm@day}{\@dtm@dow}%
354 }
```

\DTMcurrenttime Display the current time.

```
355 \newcommand*{\DTMcurrenttime}{%
356   \DTMdisplaytime
357   {\@dtm@currenthour}%
358   {\@dtm@currentminute}%
359   {\@dtm@currentsecond}%
360 }
```

\DTMdisplaytime `\DTMdisplaytime{<hour>}{<minute>}{<sec>}`

Display the given time.

```
361 \newcommand*{\DTMdisplaytime}[3]{%
362   \DTMtwodigits{#1}\dtm@hourminsep\DTMtwodigits{#2}%
363   \ifDTMshowseconds\dtm@minsecsep\DTMtwodigits{#3}\fi
364 }%
```

\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.

```
365 \newrobustcmd*{\DTMtime}[1]{%
366   \@dtm@parsetime#1\@dtm@endparsetime
367   \DTMdisplaytime{\@dtm@hour}{\@dtm@minute}{\@dtm@second}%
368 }
```

\DTMcurrentzone Display the current time zone.

```
369 \newcommand*{\DTMcurrentzone}{%
370   \DTMdisplayzone
371   {\@dtm@currenttimezonehour}%
372   {\@dtm@currenttimezoneminute}%
373 }
```

\DTMdisplayzone Display time zone.

```
374 \newcommand*{\DTMdisplayzone}[2]{%
375   \ifboolexpe
376   { bool{DTMshowisoZ}
377     and test{\ifnumequal{#1}{0}}
378     and test{\ifnumequal{#2}{0}}
379   }%
380   {%
```

```

381     Z%
382 }%
383 {%
384   \ifnum#1<0\else+\fi\DTMtwdigits{#1}%
385   \ifDTMshowzoneminutes\dtm@hourminsep\DTMtwdigits{#2}\fi
386 }%
387 }

```

\DTMnow Current date, time and time zone.

```

388 \newcommand*\DTMnow{%
389   \DTMdisplay
390   {\@dtm@currentyear}
391   {\@dtm@currentmonth}
392   {\@dtm@currentday}
393   {\@dtm@currentdow}
394   {\@dtm@currenthour}%
395   {\@dtm@currentminute}%
396   {\@dtm@currentsecond}%
397   {\@dtm@currenttimezonehour}%
398   {\@dtm@currenttimezoneminute}%
399 }

```

\DTMNow Current date, time and time zone.

```

400 \newcommand*\DTMNow{%
401   \DTMDisplay
402   {\@dtm@currentyear}
403   {\@dtm@currentmonth}
404   {\@dtm@currentday}
405   {\@dtm@currentdow}
406   {\@dtm@currenthour}%
407   {\@dtm@currentminute}%
408   {\@dtm@currentsecond}%
409   {\@dtm@currenttimezonehour}%
410   {\@dtm@currenttimezoneminute}%
411 }

```

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

Display the date and time.

```

412 \newcommand*\DTMdisplay[9]{%
413   \ifDTMshowdate
414     \DTMdisplaydate{#1}{#2}{#3}{#4}%
415     \dtm@datetimesep
416   \fi
417   \DTMdisplaytime

```

```

418   {#5}%
419   {#6}%
420   {#7}%
421   \ifDTMshowzone
422   \dtm@timezonesep
423   \DTMdisplayzone
424   {#8}%
425   {#9}%
426 \fi
427 }

```

`\DTMDisplay` `\DTMDisplay{<YYYY>}{<MM>}{<DD>}{<DOW>}{<hh>}{<mm>}{<ss>}{<TZh>}{<TZm>}`

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

```

428 \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`

```

429 \newcommand*{\DTMtwodigits}[1]{%
430   \ifnum#1<0
431     -\DTMtwodigits{-#1}%
432   \else
433     \ifnum#1<100
434       \ifnum#1<10
435         0\number#1
436       \else
437         \number#1
438       \fi
439     \else

```

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

```

440     \ifnum\numexpr#1-(#1/100)*100<0
441       \number\numexpr#1-((#1/100)-1)*100\relax
442     \else
443       \number\numexpr#1-(#1/100)*100\relax
444     \fi
445   \fi
446 \fi
447 }

```

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

```

448 \newcommand*\DTMcentury}[1]{%
449   \ifnum#1<0
450     -\DTMcentury{-#1}%
451   \else
452     \ifnum\numexpr#1-(#1/100)*100<1
453       \number\numexpr#1/100\relax
454     \else
455       \number\numexpr(#1/100)+1\relax
456     \fi
457   \fi
458 }
```

`\DTMdivhundred` Expands to the given number divided by 100.

```

459 \newcommand*\DTMdivhundred}[1]{%
460   \ifnum#1<0
461     -\DTMdivhundred{-#1}%
462   \else
463     \ifnum\numexpr#1-(#1/100)*100<0
464       \number\numexpr(#1)/100-1\relax
465     \else
466       \number\numexpr((#1)/100)\relax
467     \fi
468   \fi
469 }
```

`\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.

```

470 \newcommand*\DTMtexorpdfstring}[2]{#1}
471 \AtBeginDocument{%
472   \ifpackageloaded{hyperref}%
473   {%
474     \renewcommand*\DTMtexorpdfstring{\texorpdfstring}%
475   }%
476   {}%
477 }
```

Access separator:

`\DTMsep`

```

478 \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`.

```

479 \newcommand*{\DTMnewdatestyle}[2]{%
480   \ifcsdef{@dtm@datestyle@#1}%
481   {%
482     \PackageError{datetime2}{Date style ‘#1’ already exists}{}%
483   }%
484   {%
485     \csdef{@dtm@datestyle@#1}{#2}%
486   }%
487 }

```

`\DTMrenewdatestyle` Redefine a date style. This should only redefine `\DTMdisplaydate` and `\DTMDisplaydate`, which may or may not use the separators `\dtm@yearmonthsep` and `\dtm@monthdaysep`. This may also be used to modify the date part of a full style.

```

488 \newcommand*{\DTMrenewdatestyle}[2]{%
489   \ifcsundef{@dtm@datestyle@#1}%
490   {%
491     \PackageError{datetime2}{Date style ‘#1’ doesn’t exist}{}%
492   }%
493   {%
494     \csdef{@dtm@datestyle@#1}{#2}%
495   }%
496 }

```

`\DTMprovidedatestyle` Define a date style if it doesn’t already exist.

```

497 \newcommand*{\DTMprovidedatestyle}[2]{%
498   \ifcsdef{@dtm@datestyle@#1}%
499   {%
500   }%
501   {%
502     \csdef{@dtm@datestyle@#1}{#2}%
503   }%
504 }

```

`\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`.

```

505 \newcommand*{\DTMnewtimestyle}[2]{%
506   \ifcsdef{@dtm@timestyle@#1}%
507   {%
508     \PackageError{datetime2}{Time style ‘#1’ already exists}{}%
509   }%
510   {%
511     \csdef{@dtm@timestyle@#1}{#2}%
512   }%
513 }

```

`\DTMrenewtimestyle` Redefine a time style. This should only redefine `\DTMdisplaytime`, which may or may not use the separators `\dtm@hourminsep` and `\dtm@minsecsep`. This may also be used to modify the time part of a full style.

```

514 \newcommand*{\DTMrenewtimestyle}[2]{%
515   \ifcsundef{@dtm@timestyle@#1}%
516   {%
517     \PackageError{datetime2}{Time style ‘#1’ doesn’t exist}{}%
518   }%
519   {%
520     \csdef{@dtm@timestyle@#1}{#2}%
521   }%
522 }
```

`\DTMprovidetimestyle` Define a time style if it doesn’t already exist.

```

523 \newcommand*{\DTMprovidetimestyle}[2]{%
524   \ifcsdef{@dtm@timestyle@#1}%
525   {%
526   }%
527   {%
528     \csdef{@dtm@timestyle@#1}{#2}%
529   }%
530 }
```

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

```

531 \newcommand*{\DTMnewzonestyle}[2]{%
532   \ifcsdef{@dtm@zonestyle@#1}%
533   {%
534     \PackageError{datetime2}{Zone style ‘#1’ already exists}{}%
535   }%
536   {%
537     \csdef{@dtm@zonestyle@#1}{#2}%
538   }%
539 }
```

`\DTMrenewzonestyle` Redefine a new zone style. This should only redefine `\DTMdisplayzone`, which may or may not use the separator `\dtm@hourminsep`. This may also be used to modify the zone part of a full style.

```

540 \newcommand*{\DTMrenewzonestyle}[2]{%
541   \ifcsundef{@dtm@zonestyle@#1}%
542   {%
543     \PackageError{datetime2}{Zone style ‘#1’ doesn’t exist}{}%
544   }%
545   {%
546     \csdef{@dtm@zonestyle@#1}{#2}%
547   }%
548 }
```

DTMprovidezonestyle Defines a new zone style if it doesn't already exist.

```

549 \newcommand*{\DTMprovidezonestyle}[2]{%
550   \ifcsdef{@dtm@zonestyle@#1}%
551   {%
552   }%
553   {%
554     \csdef{@dtm@zonestyle@#1}{#2}%
555   }%
556 }
```

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 $\langle TZh \rangle : \langle TZm \rangle$ is stored in $\backslash @dtm@zonemap@ \langle TZh \rangle : \langle TZm \rangle$.

\DTMdefzonemap $\backslash \text{DTMdefzonemap} \{ \langle TZh \rangle \} \{ \langle TZm \rangle \} \{ \langle map \rangle \}$

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

```

557 \newcommand*{\DTMdefzonemap}[3]{%
558   \csdef{@dtm@zonemap@DTMtwdigits{#1}:DTMtwdigits{#2}}{#3}%
559 }
```

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

```

560 \newcommand*{\DTMusezonemapordefault}[2]{%
561   \ifcsundef{@dtm@zonemap@DTMtwdigits{#1}:DTMtwdigits{#2}}%
562   {%
563     \ifnum#1<0\else+ \fi
564     \DTMtwdigits{#1}%
565     \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwdigits{#2}\fi
566   }%
567   {\csname @dtm@zonemap@DTMtwdigits{#1}:DTMtwdigits{#2}\endcsname}%
568 }
```

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

```

569 \newcommand*{\DTMusezonemap}[2]{%
570   \csname @dtm@zonemap@DTMtwdigits{#1}:DTMtwdigits{#2}\endcsname
571 }
```

\DTMhaszonemap

```

572 \newcommand*{\DTMhaszonemap}[4]{%
573   \ifcsundef{@dtm@zonemap@DTMtwdigits{#1}:DTMtwdigits{#2}}{#4}{#3}%
574 }
```

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

```

575 \newcommand*{\DTMclearmap}[2]{%
576   \csundef{@dtm@zonemap@DTMtwdigits{#1}:DTMtwdigits{#2}}%
577 }
```

\DTMshowmap Debugging command.

```
578 \newcommand*{\DTMshowmap}[2]{%
579   \csshow{@dtm@zonemap\DTMtwodigits{#1}:\DTMtwodigits{#2}}%
580 }
```

\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.

```
581 \newcommand*{\DTMresetzones}{}%
```

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

```
582 \newcommand*{\DTMNatoZoneMaps}{%
583   \defzonemap{01}{00}{A}% Alpha time zone
584   \defzonemap{02}{00}{B}% Bravo time zone
585   \defzonemap{03}{00}{C}% Charlie time zone
586   \defzonemap{04}{00}{D}% Delta time zone
587   \defzonemap{05}{00}{E}% Echo time zone
588   \defzonemap{06}{00}{F}% Foxtrot time zone
589   \defzonemap{07}{00}{G}% Golf time zone
590   \defzonemap{08}{00}{H}% Hotel time zone
591   \defzonemap{09}{00}{I}% India time zone
592   \defzonemap{10}{00}{K}% Kilo time zone
593   \defzonemap{11}{00}{L}% Lima time zone
594   \defzonemap{12}{00}{M}% Mike time zone
595   \defzonemap{-01}{00}{N}% November time zone
596   \defzonemap{-02}{00}{O}% Oscar time zone
597   \defzonemap{-03}{00}{P}% Papa time zone
598   \defzonemap{-04}{00}{Q}% Quebec time zone
599   \defzonemap{-05}{00}{R}% Romeo time zone
600   \defzonemap{-06}{00}{S}% Sierra time zone
601   \defzonemap{-07}{00}{T}% Tango time zone
602   \defzonemap{-08}{00}{U}% Uniform time zone
603   \defzonemap{-09}{00}{V}% Victor time zone
604   \defzonemap{-10}{00}{W}% Whiskey time zone
605   \defzonemap{-11}{00}{X}% X-ray time zone
606   \defzonemap{-12}{00}{Y}% Yankee time zone
607   \defzonemap{00}{00}{Z}% Zulu time zone
608 }
```

\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.

```
609 \newcommand*{\DTMnewstyle}[5]{%
610   \ifcsdef{@dtm@style@#1}%
```



```

611  {%
612    \PackageError{datetime2}{Style ‘#1’ already exists}{}%
613  }%
614  {%
615    \DTMnewdatestyle{#1}{#2}%
616    \DTMnewtimestyle{#1}{#3}%
617    \DTMnewzonestyle{#1}{#4}%
618    \csdef{@dtm@style@#1}{%
619      \csuse{@dtm@datestyle@#1}%
620      \csuse{@dtm@timestyle@#1}%
621      \csuse{@dtm@zonestyle@#1}%
622      #5%
623    }%
624  }%
625 }

```

`\DTMrenewstyle` `\DTMrenewstyle{<label>}{<date style definition>}{<time style definition>}{<zone style definition>}{<full format definition>}`

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

```

626 \newcommand*{\DTMrenewstyle}[5]{%
627   \ifcsundef{@dtm@style@#1}%
628   {%
629     \PackageError{datetime2}{Style ‘#1’ doesn’t exist}{}%
630   }%
631   {%
632     \DTMrenewdatestyle{#1}{#2}%
633     \DTMrenewtimestyle{#1}{#3}%
634     \DTMrenewzonestyle{#1}{#4}%
635     \csdef{@dtm@style@#1}{%
636       \csuse{@dtm@datestyle@#1}%
637       \csuse{@dtm@timestyle@#1}%
638       \csuse{@dtm@zonestyle@#1}%
639       #5%
640     }%
641   }%
642 }

```

`\DTMprovidestyle` `\DTMprovidestyle{<label>}{<date style definition>}{<time style definition>}{<zone style definition>}{<full format definition>}`

Defines a full style if it doesn’t already exist.

```

643 \newcommand*{\DTMprovidestyle}[5]{%
644   \ifcsdef{@dtm@style@#1}%

```

```

645  {%
646  }%
647  {%
648      \DTMprovideddatestyle{#1}{#2}%
649      \DTMprovidetimestyle{#1}{#3}%
650      \DTMprovidezonestyle{#1}{#4}%
651      \csdef{@dtm@style@#1}{%
652          \csuse{@dtm@datestyle@#1}%
653          \csuse{@dtm@timestyle@#1}%
654          \csuse{@dtm@zonestyle@#1}%
655          #5%
656      }%
657  }%
658 }

```

\DTMsetdatestyle

```

659 \newrobustcmd*{\DTMsetdatestyle}[1]{%
660     \ifcsdef{@dtm@datestyle@#1}%
661     {\csuse{@dtm@datestyle@#1}}%
662     {%
663         \PackageError{datetime2}{Date style ‘#1’ not defined}{}%
664     }%
665 }

```

\DTMsettimestyle

```

666 \newrobustcmd*{\DTMsettimestyle}[1]{%
667     \ifcsdef{@dtm@timestyle@#1}%
668     {\csuse{@dtm@timestyle@#1}}%
669     {%
670         \PackageError{datetime2}{Time style ‘#1’ not defined}{}%
671     }%
672 }

```

\DTMsetzonestyle

```

673 \newrobustcmd*{\DTMsetzonestyle}[1]{%
674     \ifcsdef{@dtm@zonestyle@#1}%
675     {\csuse{@dtm@zonestyle@#1}}%
676     {%
677         \PackageError{datetime2}{Zone style ‘#1’ not defined}{}%
678     }%
679 }

```

\DTMsetstyle

```

680 \newrobustcmd*{\DTMsetstyle}[1]{%
681     \ifcsdef{@dtm@style@#1}%
682     {\csuse{@dtm@style@#1}}%
683     {%
684         \let\dtm@unknownstyle\@dtm@unknownstyle
685         \ifcsdef{@dtm@datestyle@#1}%

```

```

686      {\csuse{@dtm@datestyle@#1}\let\dtm@unknownstyle\@dtm@unknown@style}%
687      {\@dtm@warning{No date style ‘#1’ defined}}}%
688      \ifcsdef{@dtm@timestyle#1}%
689      {\csuse{@dtm@timestyle@#1}\let\dtm@unknownstyle\@dtm@unknown@style}%
690      {\@dtm@warning{No time style ‘#1’ defined}}}%
691      \ifcsdef{@dtm@zonestyle#1}%
692      {\csuse{@dtm@zonestyle@#1}\let\dtm@unknownstyle\@dtm@unknown@style}%
693      {\@dtm@warning{No zone style ‘#1’ defined}}}%
694      \dtm@unknownstyle{#1}%
695    }%
696 }

```

\@dtm@unknownstyle

```

697 \newcommand*{\@dtm@unknownstyle}[1]{%
698   \PackageError{datetime2}{Unknown style ‘#1’}{}%
699 }

```

\@dtm@unknown@style

```

700 \newcommand*{\@dtm@unknown@style}[1]{%
701   \@dtm@warning{No full style ‘#1’ defined}{}%
702 }

```

Define default style:

```

703 \DTMnewstyle
704 {default}%label
705 {% date style
706   \renewcommand*{\DTMdisplaydate}[4]{%
707     \number##1\DTMsep{yearmonth}\DTMtwdigits{##2}%
708     \DTMsep{monthday}\DTMtwdigits{##3}%
709   }%
710   \renewcommand*{\DTMdisplaydate}{\DTMdisplaydate}%
711 }%
712 {% time style
713   \renewcommand*{\DTMdisplaytime}[3]{%
714     \DTMtwdigits{##1}\DTMsep{hourmin}\DTMtwdigits{##2}%
715     \ifDTMshowseconds\DTMsep{minsec}\DTMtwdigits{##3}\fi
716   }%
717 }%
718 {% zone style
719   \renewcommand*{\DTMdisplayzone}[2]{%
720     \ifboolexpe
721     { bool{DTMshowisoZ}
722       and test{\ifnumequal{##1}{0}}
723       and test{\ifnumequal{##2}{0}}
724     }%
725     {%
726       Z%
727     }%
728     {%

```

```

729     \ifnum##1<0\else+\fi\DTMtwdigits{##1}%
730     \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwdigits{##2}\fi
731 }%
732 }%
733 }%
734 {% full style
735   \renewcommand*\DTMdisplay}[9]{%
736     \ifDTMshowdate
737       \DTMdisplaydate{##1}-{##2}-{##3}-{##4}%
738       \DTMsep{datetime}%
739     \fi
740     \DTMdisplaytime
741     {##5}%
742     {##6}%
743     {##7}%
744     \ifDTMshowzone
745       \DTMsep{timezone}%
746       \DTMdisplayzone
747       {##8}%
748       {##9}%
749     \fi
750   }%
751   \renewcommand*\DTMDisplay{\DTMdisplay}%
752 }

```

Define iso style which ignores the separator settings:

```

753 \DTMnewstyle
754 {iso}%label
755 {% date style
756   \renewcommand*\DTMdisplaydate[4]{%
757     \number##1-\DTMtwdigits{##2}-\DTMtwdigits{##3}%
758   }%
759   \renewcommand*\DTMDisplaydate{\DTMdisplaydate}%
760 }%
761 {% time style
762   \renewcommand*\DTMdisplaytime[3]{%
763     \DTMtwdigits{##1}:\DTMtwdigits{##2}%
764     \ifDTMshowseconds:\DTMtwdigits{##3}\fi
765   }%
766 }%
767 {% zone style
768   \renewcommand*\DTMdisplayzone[2]{%
769     \ifboolexpe
770     { bool{DTMshowisoZ}
771       and test{\ifnumequal{##1}{0}}
772       and test{\ifnumequal{##2}{0}}
773     }%
774     {%
775       Z%
776     }%

```

```

777     {%
778       \ifnum##1<0\else+\fi\DTMtwdigits{##1}%
779       \ifDTMshowzoneminutes:\DTMtwdigits{##2}\fi
780     }%
781   }%
782 }%
783 {% full style
784   \renewcommand*\DTMdisplay}[9]{%
785     \ifDTMshowdate
786       \DTMdisplaydate{##1}-{##2}-{##3}-{##4}%
787       T%
788     \fi
789     \DTMdisplaytime
790     {##5}%
791     {##6}%
792     {##7}%
793     \ifDTMshowzone
794       \DTMdisplayzone
795       {##8}%
796       {##9}%
797     \fi
798   }%
799   \renewcommand*\DTMDisplay{\DTMdisplay}%
800 }

```

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

```

801 \DTMnewstyle
802 {pdf}%label
803 {% date style
804   \renewcommand*\DTMdisplaydate[4]{%
805     D:\number##1 % space intended
806     \DTMtwdigits{##2}\DTMtwdigits{##3}%
807   }%
808   \renewcommand*\DTMDisplaydate{\DTMdisplaydate}%
809 }%
810 {% time style
811   \renewcommand*\DTMdisplaytime[3]{%
812     \DTMtwdigits{##1}\DTMtwdigits{##2}\DTMtwdigits{##3}%
813   }%
814 }%
815 {% zone style
816   \renewcommand*\DTMdisplayzone[2]{%
817     \ifboolexpe
818     { bool{DTMshowisoZ}
819       and test{\ifnumequal{##1}{0}}
820       and test{\ifnumequal{##2}{0}}
821     }%
822     {%
823       Z%
824     }%

```

```

825     {%
826     \ifnum##1<0\else+\fi\DTMtwdigits{##1}'\DTMtwdigits{##2}'%
827     }%
828 }%
829 }%
830 {% full style
831 \renewcommand*\DTMdisplay}[9]{%
832 \DTMdisplaydate{##1}{##2}{##3}{##4}%
833 \DTMdisplaytime{##5}{##6}{##7}%
834 \DTMdisplayzone{##8}{##9}%
835 }%
836 \renewcommand*\DTMDisplay{\DTMdisplay}%
837 }

```

Define yyyyymd style:

```

838 \DTMnewstyle
839 {yyyyymd}%label
840 {% date style
841 \renewcommand*\DTMdisplaydate[4]{%
842 \number##1
843 \DTMsep{yearmonth}%
844 \number##2
845 \DTMsep{monthday}%
846 \number##3
847 }%
848 \renewcommand*\DTMDisplaydate{\DTMdisplaydate}%
849 }%
850 {% time style
851 \renewcommand*\DTMdisplaytime[3]{%
852 \DTMtwdigits{##1}\DTMsep{hourmin}\DTMtwdigits{##2}%
853 \ifDTMshowseconds\DTMsep{minsec}\DTMtwdigits{##3}\fi
854 }%
855 }%
856 {% zone style
857 \renewcommand*\DTMdisplayzone[2]{%
858 \ifboolexpe
859 { bool{DTMshowisoZ}
860 and test{\ifnumequal{##1}{0}}
861 and test{\ifnumequal{##2}{0}}
862 }%
863 {%
864 Z%
865 }%
866 {%
867 \ifnum##1<0\else+\fi\DTMtwdigits{##1}%
868 \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwdigits{##2}\fi
869 }%
870 }%
871 }%
872 {% full style

```

```

873 \renewcommand*\DTMdisplay}[9]{%
874 \ifDTMshowdate
875 \DTMdisplaydate{##1}{##2}{##3}{##4}%
876 \DTMsep{datetime}%
877 \fi
878 \DTMdisplaytime
879 {##5}%
880 {##6}%
881 {##7}%
882 \ifDTMshowzone
883 \DTMsep{timezone}%
884 \DTMdisplayzone
885 {##8}%
886 {##9}%
887 \fi
888 }%
889 \renewcommand*\DTMDisplay{\DTMdisplay}%
890 }

```

Define ddmmyyyy style:

```

891 \DTMnewstyle
892 {ddmmyyyy}%label
893 {% date style
894 \renewcommand*\DTMdisplaydate[4]{%
895 \DTMtwodigits{##3}\DTMsep{monthday}%
896 \DTMtwodigits{##2}\DTMsep{yearmonth}%
897 \number##1
898 }%
899 \renewcommand*\DTMDisplaydate{\DTMdisplaydate}%
900 }%
901 {% time style
902 \renewcommand*\DTMdisplaytime[3]{%
903 \DTMtwodigits{##1}\DTMsep{hourmin}\DTMtwodigits{##2}%
904 \ifDTMshowseconds\DTMsep{minsec}\DTMtwodigits{##3}\fi
905 }%
906 }%
907 {% zone style
908 \renewcommand*\DTMdisplayzone[2]{%
909 \ifboolexpe
910 { bool{DTMshowisoZ}
911 and test{\ifnumequal{##1}{0}}
912 and test{\ifnumequal{##2}{0}}
913 }%
914 {%
915 Z%
916 }%
917 {%
918 \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
919 \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
920 }%

```

```

921 }%
922 }%
923 {% full style
924 \renewcommand*{\DTMdisplay}[9]{%
925 \ifDTMshowdate
926 \DTMdisplaydate{##1}{##2}{##3}{##4}%
927 \DTMsep{datetime}%
928 \fi
929 \DTMdisplaytime
930 {##5}%
931 {##6}%
932 {##7}%
933 \ifDTMshowzone
934 \DTMsep{timezone}%
935 \DTMdisplayzone
936 {##8}%
937 {##9}%
938 \fi
939 }%
940 \renewcommand*{\DTMDisplay}{\DTMdisplay}%
941 }

Define dmyyyy style:
942 \DTMnewstyle
943 {dmyyyy}%label
944 {% date style
945 \renewcommand*{\DTMdisplaydate}[4]{%
946 \number##3
947 \DTMsep{monthday}%
948 \number##2
949 \DTMsep{yearmonth}%
950 \number##1
951 }%
952 \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
953 }%
954 {% time style
955 \renewcommand*{\DTMdisplaytime}[3]{%
956 \DTMtwdigits{##1}\DTMsep{hourmin}\DTMtwdigits{##2}%
957 \ifDTMshowseconds\DTMsep{minsec}\DTMtwdigits{##3}\fi
958 }%
959 }%
960 {% zone style
961 \renewcommand*{\DTMdisplayzone}[2]{%
962 \ifboolexpe
963 { bool{DTMshowisoZ}
964 and test{\ifnumequal{##1}{0}}
965 and test{\ifnumequal{##2}{0}}
966 }%
967 {%
968 Z%

```



```

969     }%
970     {%
971         \ifnum##1<0\else+\fi\DTMt看odigits{##1}%
972         \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMt看odigits{##2}\fi
973     }%
974 }%
975 }%
976 {% full style
977     \renewcommand*\DTMdisplay}[9]{%
978         \ifDTMshowdate
979             \DTMdisplaydate{##1}-{##2}-{##3}-{##4}%
980             \DTMsep{datetime}%
981         \fi
982         \DTMdisplaytime
983         {##5}%
984         {##6}%
985         {##7}%
986         \ifDTMshowzone
987             \DTMsep{timezone}%
988             \DTMdisplayzone
989             {##8}%
990             {##9}%
991         \fi
992     }%
993     \renewcommand*\DTMDisplay}{\DTMdisplay}%
994 }

```

Define dmyy style:

```

995 \DTMnewstyle
996 {dmyy}%label
997 {% date style
998     \renewcommand*\DTMdisplaydate[4]{%
999         \number##3 % space intended
1000         \DTMsep{monthday}%
1001         \number##2 % space intended
1002         \DTMsep{yearmonth}%
1003         \DTMt看odigits{##1}%
1004     }%
1005     \renewcommand*\DTMDisplaydate}{\DTMdisplaydate}%
1006 }%
1007 {% time style
1008     \renewcommand*\DTMdisplaytime[3]{%
1009         \DTMt看odigits{##1}\DTMsep{hourmin}\DTMt看odigits{##2}%
1010         \ifDTMshowseconds\DTMsep{minsec}\DTMt看odigits{##3}\fi
1011     }%
1012 }%
1013 {% zone style
1014     \renewcommand*\DTMdisplayzone}[2]{%
1015         \ifbool{expe
1016         { bool{DTMshowisoZ}

```

```

1017         and test{\ifnumequal{##1}{0}}
1018         and test{\ifnumequal{##2}{0}}
1019     }%
1020     {%
1021         Z%
1022     }%
1023     {%
1024         \ifnum##1<0\else+\fi\DTMtwdigits{##1}%
1025         \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwdigits{##2}\fi
1026     }%
1027 }%
1028 }%
1029 {% full style
1030     \renewcommand*\DTMdisplay}[9]{%
1031         \ifDTMshowdate
1032             \DTMdisplaydate{##1}{##2}{##3}{##4}%
1033             \DTMsep{datetime}%
1034         \fi
1035         \DTMdisplaytime
1036         {##5}%
1037         {##6}%
1038         {##7}%
1039         \ifDTMshowzone
1040             \DTMsep{timezone}%
1041             \DTMdisplayzone
1042             {##8}%
1043             {##9}%
1044         \fi
1045     }%
1046     \renewcommand*\DTMDisplay}{\DTMdisplay}%
1047 }

```

Define mmddyyyy style:

```

1048 \DTMnewstyle
1049 {mmddyyyy}%label
1050 {% date style
1051     \renewcommand*\DTMdisplaydate[4]{%
1052         \DTMtwdigits{##2}\DTMsep{monthday}%
1053         \DTMtwdigits{##3}\DTMsep{dayyear}%
1054         \number##1
1055     }%
1056     \renewcommand*\DTMDisplaydate}{\DTMdisplaydate}%
1057 }%
1058 {% time style
1059     \renewcommand*\DTMdisplaytime[3]{%
1060         \DTMtwdigits{##1}\DTMsep{hourmin}\DTMtwdigits{##2}%
1061         \ifDTMshowseconds\DTMsep{minsec}\DTMtwdigits{##3}\fi
1062     }%
1063 }%
1064 {% zone style

```

```

1065 \renewcommand*{\DTMdisplayzone}[2]{%
1066 \ifbool{DTMshowisoZ}
1067 { bool{DTMshowisoZ}
1068 and test{\ifnumequal{##1}{0}}
1069 and test{\ifnumequal{##2}{0}}
1070 }%
1071 {%
1072 Z%
1073 }%
1074 {%
1075 \ifnum##1<0\else+\fi\DTMtwodigits{##1}%
1076 \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwodigits{##2}\fi
1077 }%
1078 }%
1079 }%
1080 {% full style
1081 \renewcommand*{\DTMdisplay}[9]{%
1082 \ifDTMshowdate
1083 \DTMdisplaydate{##1}{##2}{##3}{##4}%
1084 \DTMsep{datetime}%
1085 \fi
1086 \DTMdisplaytime
1087 {##5}%
1088 {##6}%
1089 {##7}%
1090 \ifDTMshowzone
1091 \DTMsep{timezone}%
1092 \DTMdisplayzone
1093 {##8}%
1094 {##9}%
1095 \fi
1096 }%
1097 \renewcommand*{\DTMDisplay}{\DTMdisplay}%
1098 }

```

Define mdyyyy style:

```

1099 \DTMnewstyle
1100 {mdyyyy}%label
1101 {% date style
1102 \renewcommand*{\DTMdisplaydate}[4]{%
1103 \number##2 % space intended
1104 \DTMsep{monthday}%
1105 \number##3 % space intended
1106 \DTMsep{dayyear}%
1107 \number##1 % space intended
1108 }%
1109 \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
1110 }%
1111 {% time style
1112 \renewcommand*{\DTMdisplaytime}[3]{%

```

```

1113     \DTMtwdigits{##1}\DTMsep{hourmin}\DTMtwdigits{##2}%
1114     \ifDTMshowseconds\DTMsep{minsec}\DTMtwdigits{##3}\fi
1115 }%
1116 }%
1117 {% zone style
1118   \renewcommand*\DTMdisplayzone}[2]{%
1119     \ifboolxpe
1120     { bool{DTMshowisoZ}
1121       and test{\ifnumequal{##1}{0}}
1122       and test{\ifnumequal{##2}{0}}
1123     }%
1124     {%
1125       Z%
1126     }%
1127     {%
1128       \ifnum##1<0\else+\fi\DTMtwdigits{##1}%
1129       \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwdigits{##2}\fi
1130     }%
1131   }%
1132 }%
1133 {% full style
1134   \renewcommand*\DTMdisplay}[9]{%
1135     \ifDTMshowdate
1136     \DTMdisplaydate{##1}{##2}{##3}{##4}%
1137     \DTMsep{datetime}%
1138     \fi
1139     \DTMdisplaytime
1140     {##5}%
1141     {##6}%
1142     {##7}%
1143     \ifDTMshowzone
1144     \DTMsep{timezone}%
1145     \DTMdisplayzone
1146     {##8}%
1147     {##9}%
1148     \fi
1149   }%
1150   \renewcommand*\DTMDisplay}{\DTMdisplay}%
1151 }

```

Define mdy style:

```

1152 \DTMnewstyle
1153 {mdy}%label
1154 {% date style
1155   \renewcommand*\DTMdisplaydate[4]{%
1156     \number##2 % space intended
1157     \DTMsep{monthday}%
1158     \number##3 % space intended
1159     \DTMsep{dayyear}%
1160     \DTMtwdigits{##1}%

```

```

1161 }%
1162 \renewcommand*{\DTMDisplaydate}{\DTMdisplaydate}%
1163 }%
1164 {% time style
1165 \renewcommand*{\DTMdisplaytime}[3]{%
1166 \DTMtwdigits{##1}\DTMsep{hourmin}\DTMtwdigits{##2}%
1167 \ifDTMshowseconds\DTMsep{minsec}\DTMtwdigits{##3}\fi
1168 }%
1169 }%
1170 {% zone style
1171 \renewcommand*{\DTMdisplayzone}[2]{%
1172 \ifboolexpe
1173 { bool{DTMshowisoZ}
1174 and test{\ifnumequal{##1}{0}}
1175 and test{\ifnumequal{##2}{0}}
1176 }%
1177 {%
1178 Z%
1179 }%
1180 {%
1181 \ifnum##1<0\else+\fi\DTMtwdigits{##1}%
1182 \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwdigits{##2}\fi
1183 }%
1184 }%
1185 }%
1186 {% full style
1187 \renewcommand*{\DTMdisplay}[9]{%
1188 \ifDTMshowdate
1189 \DTMdisplaydate{##1}{##2}{##3}{##4}%
1190 \DTMsep{datetime}%
1191 \fi
1192 \DTMdisplaytime
1193 {##5}%
1194 {##6}%
1195 {##7}%
1196 \ifDTMshowzone
1197 \DTMsep{timezone}%
1198 \DTMdisplayzone
1199 {##8}%
1200 {##9}%
1201 \fi
1202 }%
1203 \renewcommand*{\DTMDisplay}{\DTMdisplay}%
1204 }

```

Define hmmss time style

```

1205 \DTMnewtimestyle
1206 {hmmss}% label
1207 {%
1208 \renewcommand*{\DTMdisplaytime}[3]{%

```

```

1209     \number##1
1210     \DTMsep{hourmin}\DTMtwdigits{##2}%
1211     \ifDTMshowseconds\DTMsep{minsec}\DTMtwdigits{##3}\fi
1212 }%
1213 }%

    Define map zone style
1214 \DTMnewzonestyle
1215 {map}% label
1216 {%
1217     \renewcommand*\DTMdisplaytime[3]{%
1218         \DTMusezonemapordefault{##1}{##2}%
1219     }%
1220 }%

    Define hhmm zone style
1221 \DTMnewzonestyle
1222 {hhmm}% label
1223 {%
1224     \renewcommand*\DTMdisplaytime[3]{%
1225         \ifnum##1<0\else+\fi\DTMtwdigits{##1}%
1226         \ifDTMshowzoneminutes\DTMsep{hourmin}\DTMtwdigits{##2}\fi
1227     }%
1228 }

```

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.

```

1229 \newrobustcmd*{\DTMsavedate}[2]{%
1230     \expandafter\@dtm@parsedate#2\@dtm@endparsedate
1231     \cslet{\@dtm@#1@year}{\@dtm@year}%
1232     \cslet{\@dtm@#1@month}{\@dtm@month}%
1233     \cslet{\@dtm@#1@day}{\@dtm@day}%
1234     \cslet{\@dtm@#1@dow}{\@dtm@dow}%
1235     \ifcsundef{\@dtm@#1@hour}{\csdef{\@dtm@#1@hour}{0}}{}%
1236     \ifcsundef{\@dtm@#1@minute}{\csdef{\@dtm@#1@minute}{0}}{}%
1237     \ifcsundef{\@dtm@#1@second}{\csdef{\@dtm@#1@second}{0}}{}%
1238     \ifcsundef{\@dtm@#1@TZhour}{\csdef{\@dtm@#1@TZhour}{0}}{}%
1239     \ifcsundef{\@dtm@#1@TZminute}{\csdef{\@dtm@#1@TZminute}{0}}{}%
1240 }

```

`\DTMsavenoparsedate` Save the date without parsing the $\langle YYYY \rangle$ - $\langle MM \rangle$ - $\langle DD \rangle$ format.

```

1241 \newrobustcmd*{\DTMsavenoparsedate}[5]{%
1242   \csdef{@dtm@#1@year}{\number#2}%
1243   \csdef{@dtm@#1@month}{\number#3}%
1244   \csdef{@dtm@#1@day}{\number#4}%
1245   \csdef{@dtm@#1@dow}{\number#5}%
1246   \ifcsundef{@dtm@#1@hour}{\csdef{@dtm@#1@hour}{0}}{}%
1247   \ifcsundef{@dtm@#1@minute}{\csdef{@dtm@#1@minute}{0}}{}%
1248   \ifcsundef{@dtm@#1@second}{\csdef{@dtm@#1@second}{0}}{}%
1249   \ifcsundef{@dtm@#1@TZhour}{\csdef{@dtm@#1@TZhour}{0}}{}%
1250   \ifcsundef{@dtm@#1@TZminute}{\csdef{@dtm@#1@TZminute}{0}}{}%
1251 }
```

`\DTMsavetime` Save the time specified in the format $\langle hh \rangle$: $\langle mm \rangle$: $\langle ss \rangle$. `\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.

```

1252 \newrobustcmd*{\DTMsavetime}[2]{%
1253   \expandafter\@dtm@parsetime#2\@dtm@endparsetime
1254   \cslet{@dtm@#1@hour}{\@dtm@hour}%
1255   \cslet{@dtm@#1@minute}{\@dtm@minute}%
1256   \cslet{@dtm@#1@second}{\@dtm@second}%
1257   \ifcsundef{@dtm@#1@year}{\csdef{@dtm@#1@year}{0}}{}%
1258   \ifcsundef{@dtm@#1@month}{\csdef{@dtm@#1@month}{0}}{}%
1259   \ifcsundef{@dtm@#1@day}{\csdef{@dtm@#1@day}{0}}{}%
1260   \ifcsundef{@dtm@#1@dow}{\csdef{@dtm@#1@dow}{-1}}{}%
1261   \ifcsundef{@dtm@#1@TZhour}{\csdef{@dtm@#1@TZhour}{0}}{}%
1262   \ifcsundef{@dtm@#1@TZminute}{\csdef{@dtm@#1@TZminute}{0}}{}%
1263 }
```

`\DTMsavetimezn` Save the time (including zone) specified in the format $\langle hh \rangle$: $\langle mm \rangle$: $\langle ss \rangle$ $\langle tzh \rangle$: $\langle tzm \rangle$. `\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.

```

1264 \newrobustcmd*{\DTMsavetimezn}[2]{%
1265   \expandafter\@dtm@parsetimezn#2\@dtm@endparsetimezn
1266   \cslet{@dtm@#1@hour}{\@dtm@hour}%
1267   \cslet{@dtm@#1@minute}{\@dtm@minute}%
1268   \cslet{@dtm@#1@second}{\@dtm@second}%
1269   \cslet{@dtm@#1@TZhour}{\@dtm@timezonehour}%
1270   \cslet{@dtm@#1@TZminute}{\@dtm@timezoneminute}%
1271   \ifcsundef{@dtm@#1@year}{\csdef{@dtm@#1@year}{0}}{}%
1272   \ifcsundef{@dtm@#1@month}{\csdef{@dtm@#1@month}{0}}{}%
1273   \ifcsundef{@dtm@#1@day}{\csdef{@dtm@#1@day}{0}}{}%
1274   \ifcsundef{@dtm@#1@dow}{\csdef{@dtm@#1@dow}{-1}}{}%
1275 }
```

`\DTMsavetimestamp` Save the time (including zone) specified in the format $\langle YYYY \rangle$ - $\langle MM \rangle$ - $\langle DD \rangle$ T $\langle hh \rangle$: $\langle mm \rangle$: $\langle ss \rangle$ $\langle time zone \rangle$

The first argument is the label.

```

1276 \newrobustcmd*{\DTMsavetimestamp}[2]{%
1277   \expandafter\@dtm@parsetimestamp#2\@dtm@endparsetimestamp
1278   \cslet{\@dtm@#1@year}{\@dtm@year}%
1279   \cslet{\@dtm@#1@month}{\@dtm@month}%
1280   \cslet{\@dtm@#1@day}{\@dtm@day}%
1281   \cslet{\@dtm@#1@dow}{\@dtm@dow}%
1282   \cslet{\@dtm@#1@hour}{\@dtm@hour}%
1283   \cslet{\@dtm@#1@minute}{\@dtm@minute}%
1284   \cslet{\@dtm@#1@second}{\@dtm@second}%
1285   \cslet{\@dtm@#1@TZhour}{\@dtm@timezonehour}%
1286   \cslet{\@dtm@#1@TZminute}{\@dtm@timezoneminute}%
1287 }

```

`\DTMsavenow` Save the current time.

```

1288 \newrobustcmd{\DTMsavenow}[1]{%
1289   \cslet{\@dtm@#1@year}{\@dtm@currentyear}%
1290   \cslet{\@dtm@#1@month}{\@dtm@currentmonth}%
1291   \cslet{\@dtm@#1@day}{\@dtm@currentday}%
1292   \cslet{\@dtm@#1@dow}{\@dtm@currentdow}%
1293   \cslet{\@dtm@#1@hour}{\@dtm@currenthour}%
1294   \cslet{\@dtm@#1@minute}{\@dtm@currentminute}%
1295   \cslet{\@dtm@#1@second}{\@dtm@currentsecond}%
1296   \cslet{\@dtm@#1@TZhour}{\@dtm@currenttimezonehour}%
1297   \cslet{\@dtm@#1@TZminute}{\@dtm@currenttimezoneminute}%
1298 }

```

`\DTMmakeglobal` Globally set the stored information.

```

1299 \newrobustcmd{\DTMmakeglobal}[1]{%
1300   \global\csletcs{\@dtm@#1@year}{\@dtm@#1@year}%
1301   \global\csletcs{\@dtm@#1@month}{\@dtm@#1@month}%
1302   \global\csletcs{\@dtm@#1@day}{\@dtm@#1@day}%
1303   \global\csletcs{\@dtm@#1@dow}{\@dtm@#1@dow}%
1304   \global\csletcs{\@dtm@#1@hour}{\@dtm@#1@hour}%
1305   \global\csletcs{\@dtm@#1@minute}{\@dtm@#1@minute}%
1306   \global\csletcs{\@dtm@#1@second}{\@dtm@#1@second}%
1307   \global\csletcs{\@dtm@#1@TZhour}{\@dtm@#1@TZhour}%
1308   \global\csletcs{\@dtm@#1@TZminute}{\@dtm@#1@TZminute}%
1309 }

```

Expandable ways of fetching saved data. (No check for existence performed.)

The argument is the label.

`\DTMfetchyear`

```

1310 \newcommand*{\DTMfetchyear}[1]{\csname @dtm@#1@year\endcsname}

```

`\DTMfetchmonth`

```

1311 \newcommand*{\DTMfetchmonth}[1]{\csname @dtm@#1@month\endcsname}

```


\DTMfetchday

```
1312 \newcommand*{\DTMfetchday}[1]{\csname @dtm@#1@day\endcsname}
```

\DTMfetchdow

```
1313 \newcommand*{\DTMfetchdow}[1]{\csname @dtm@#1@dow\endcsname}
```

\DTMfetchhour

```
1314 \newcommand*{\DTMfetchhour}[1]{\csname @dtm@#1@hour\endcsname}
```

\DTMfetchminute

```
1315 \newcommand*{\DTMfetchminute}[1]{\csname @dtm@#1@minute\endcsname}
```

\DTMfetchsecond

```
1316 \newcommand*{\DTMfetchsecond}[1]{\csname @dtm@#1@second\endcsname}
```

\DTMfetchTZhour

```
1317 \newcommand*{\DTMfetchTZhour}[1]{\csname @dtm@#1@TZhour\endcsname}
```

\DTMfetchTZminute

```
1318 \newcommand*{\DTMfetchTZminute}[1]{\csname @dtm@#1@TZminute\endcsname}
```

\DTMusedate \DTMusedate{<label>}

Displays the previously saved date using \DTMdisplaydate.

```
1319 \newcommand*{\DTMusedate}[1]{%
1320   \ifcsundef{@dtm@#1@year}%
1321   {%
1322     \PackageError{datetime2}{Undefined date ‘#1’}{}%
1323   }%
1324   {%
1325     \DTMdisplaydate
1326     {\csname @dtm@#1@year\endcsname}%
1327     {\csname @dtm@#1@month\endcsname}%
1328     {\csname @dtm@#1@day\endcsname}%
1329     {\csname @dtm@#1@dow\endcsname}%
1330   }%
1331 }%
```

\DTMUsedate \DTMUsedate{<label>}

Displays the previously saved date using \DTMDisplaydate.

```
1332 \newcommand*{\DTMUsedate}[1]{%
```

```

1333 \ifcsundef{@dtm@#1@year}%
1334 {%
1335     \PackageError{datetime2}{Undefined date ‘#1’}{}%
1336 }%
1337 {%
1338     \DTMDisplaydate
1339     {\csname @dtm@#1@year\endcsname}%
1340     {\csname @dtm@#1@month\endcsname}%
1341     {\csname @dtm@#1@day\endcsname}%
1342     {\csname @dtm@#1@dow\endcsname}%
1343 }%
1344 }%

```

`\DTMusetime` `\DTMusetime{<label>}`

Displays the previously saved time using `\DTMdisplaytime`.

```

1345 \newcommand*\DTMusetime[1]{%
1346     \ifcsundef{@dtm@#1@hour}%
1347     {%
1348         \PackageError{datetime2}{Undefined time ‘#1’}{}%
1349     }%
1350     {%
1351         \DTMdisplaytime
1352         {\csname @dtm@#1@hour\endcsname}%
1353         {\csname @dtm@#1@minute\endcsname}%
1354         {\csname @dtm@#1@second\endcsname}%
1355     }%
1356 }%

```

`\DTMusezone` `\DTMusezone{<label>}`

Displays the previously saved date using `\DTMdisplayzone`.

```

1357 \newcommand*\DTMusezone[1]{%
1358     \ifcsundef{@dtm@#1@TZhour}%
1359     {%
1360         \PackageError{datetime2}{Undefined time ‘#1’}{}%
1361     }%
1362     {%
1363         \DTMdisplayzone
1364         {\csname @dtm@#1@TZhour\endcsname}%
1365         {\csname @dtm@#1@TZminute\endcsname}%
1366     }%
1367 }%

```

\DTMuse \DTMuse{<label>}

Displays the previously saved date and time.

```

1368 \newcommand*\DTMuse[1]{%
1369   \ifcsundef{@dtm@#1@year}%
1370   {%
1371     \PackageError{datetime2}{Undefined date-time ‘#1’}{}%
1372   }%
1373   {%
1374     \DTMdisplay
1375     {\csname @dtm@#1@year\endcsname}%
1376     {\csname @dtm@#1@month\endcsname}%
1377     {\csname @dtm@#1@day\endcsname}%
1378     {\csname @dtm@#1@dow\endcsname}%
1379     {\csname @dtm@#1@hour\endcsname}%
1380     {\csname @dtm@#1@minute\endcsname}%
1381     {\csname @dtm@#1@second\endcsname}%
1382     {\csname @dtm@#1@TZhour\endcsname}%
1383     {\csname @dtm@#1@TZminute\endcsname}%
1384   }%
1385 }%
```

\DTMUse \DTMUse{<label>}

Displays the previously saved date and time.

```

1386 \newcommand*\DTMUse[1]{%
1387   \ifcsundef{@dtm@#1@year}%
1388   {%
1389     \PackageError{datetime2}{Undefined date-time ‘#1’}{}%
1390   }%
1391   {%
1392     \DTMDisplay
1393     {\csname @dtm@#1@year\endcsname}%
1394     {\csname @dtm@#1@month\endcsname}%
1395     {\csname @dtm@#1@day\endcsname}%
1396     {\csname @dtm@#1@dow\endcsname}%
1397     {\csname @dtm@#1@hour\endcsname}%
1398     {\csname @dtm@#1@minute\endcsname}%
1399     {\csname @dtm@#1@second\endcsname}%
1400     {\csname @dtm@#1@TZhour\endcsname}%
1401     {\csname @dtm@#1@TZminute\endcsname}%
1402   }%
1403 }%
```

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

```

1404 \newcommand{\DTMifsaveddate}[3]{%
1405   \ifcsundef{@dtm@#1@year}{#3}{#2}%
1406 }

```

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.

```

1407 \newcommand*{\@dtm@requiremodule}[1]{%
1408   \IfTrackedLanguageFileExists{#1}%
1409   {datetime2-}% prefix
1410   {.ldf}% suffix
1411   {%
1412     \RequireDateTimeModule{\CurrentTrackedTag}%
1413   }%
1414   {%
1415     \@dtm@warning{Date-Time Language Module ‘#1’ not installed}%
1416   }%
1417 }

```

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

```

1418 \newcommand*{\@dtm@loadedregions}{%

```

`\RequireDateTimeModule` 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.

```

1419 \newcommand*{\RequireDateTimeModule}[1]{%
1420   \ifundef\CurrentTrackedDialect
1421   {%
1422     \PackageError{datetime2}%
1423     {\string\RequireDateTimeModule\space not permitted here}%
1424     {This command is only permitted inside datetime2 language
1425      modules.}%
1426   }%
1427   {%
1428     \ifcsundef{ver@datetime2-#1.ldf}%
1429     {%
1430       \input{datetime2-#1.ldf}%
1431       \ifdefempty\@dtm@loadedregions
1432       {%
1433         \edef\@dtm@loadedregions{#1}%
1434       }%
1435       {%
1436         \edef\@dtm@loadedregions{\@dtm@loadedregions,#1}%
1437       }%
1438     }%

```

```

1439   {}%
1440 }%
1441 }

```

`\ProvidesDateTimeModule` For use in language module to identify itself.

```

1442 \newcommand*{\ProvidesDateTimeModule}[1]{%
1443   \ProvidesFile{datetime2-#1.1df}%
1444 }

```

`\DTMdefkey` `\DTMdefkey{<region>}{<key>}[<default>]{<func>}`

Used by language modules to define a key.

```

1445 \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.

```

1446 \newcommand*{\DTMdefchoicekey}[1]{\define@choicekey[dtm]{#1}}

```

`\DTMdefboolkey` `\DTMdefboolkey{<region>}[<mp>]{<key>}[<default>]{<func>}`

Used by language modules to define a boolean key.

```

1447 \newcommand*{\DTMdefboolkey}[1]{\define@boolkey[dtm]{#1}}

```

`\DTMifbool` `\DTMifbool{<region>}{<key>}{<true>}{<false>}`

Test boolean key that was defined using `\DTMdefboolkey`

```

1448 \newcommand*{\DTMifbool}[4]{\ifbool{dtm@#1@#2}{#3}{#4}}

```

`\DTMsetbool` `\DTMsetbool{<region>}{<key>}{<value>}`

Set boolean key that was defined using `\DTMdefboolkey`

```

1449 \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.)

```

1450 \newcommand*{\DTMlangsetup}[2][\@dtm@loadedregions]{%
1451   \@for\@dtm@region:=#1\do{%
1452     \setkeys*+[dtm]{\@dtm@region}{#2}%
1453     \ifdefempty\XKV@rm{%
1454       {%
1455         \@dtm@warning{Region '@@dtm@region' has ignored
1456           \MessageBreak the following settings:\MessageBreak
1457           \XKV@rm
1458           ^^J}%
1459       }%
1460     }%
1461 }
```

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

```

1462 \AnyTrackedLanguages
1463 {%
1464   \ForEachTrackedDialect{\this@dialect}%
1465   {%
1466     \@dtm@requiremodule\this@dialect
1467   }%
1468 }
1469 {%
```

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

```
1470 }
```

Load `datetime2-calc` if required.

```
1471 \@dtm@usecalc
```

Use the style package option, if set.

```
1472 \ifdefempty\@dtm@initialstyle{}\DTMsetstyle{\@dtm@initialstyle}
```

9.2 `datetime2-calc.sty` code

```

1473 \NeedsTeXFormat{LaTeX2e}
1474 \ProvidesPackage{datetime2-calc}[2015/11/10 v1.2 (NLCT)]
```

Load other required packages

```

1475 \RequirePackage{pgfkeys}
1476 \RequirePackage{pgfcalendar}
```

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

```
1477 \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.

```

1478 \def\@dtm@parsedate#1-#2-#3\@dtm@endparsedate{%
1479   \pgfcalendardate{#1-#2-#3}\@dtm@julianday}%
1480   \pgfcalendarjuliantodate{\@dtm@julianday}\@dtm@year}\@dtm@month}\@dtm@day}%
1481   \pgfcalendarjuliantowekday{\@dtm@julianday}\count@}%
1482   \edef\@dtm@dow{\number\count@}%
1483 }

```

Set the current day of week

`\@dtm@currentdow`

```

1484 \pgfcalendardate{#1-#2-#3}\@dtm@julianday}%
1485   \pgfcalendarjuliantodate{\@dtm@julianday}\@dtm@year}\@dtm@month}\@dtm@day}%
1486   \pgfcalendarjuliantowekday{\@dtm@julianday}\count@}%
1487   \edef\@dtm@dow{\number\count@}%
1488 }

```

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

```

1489 \newrobustcmd*{\DTMsavejulianday}[2]{%
1490   \pgfcalendarjuliantodate{#2}\@dtm@year}\@dtm@month}\@dtm@day}%
1491   \pgfcalendarjuliantowekday{#2}\count@}%
1492   \csdef{@dtm@#1@dow}{\number\count@}%
1493   \cslet{@dtm@#1@year}{\@dtm@year}%
1494   \cslet{@dtm@#1@month}{\@dtm@month}%
1495   \cslet{@dtm@#1@day}{\@dtm@day}%
1496   \ifcsundef{@dtm@#1@hour}{\csdef{@dtm@#1@hour}{0}}{}%
1497   \ifcsundef{@dtm@#1@minute}{\csdef{@dtm@#1@minute}{0}}{}%
1498   \ifcsundef{@dtm@#1@second}{\csdef{@dtm@#1@second}{0}}{}%
1499   \ifcsundef{@dtm@#1@TZhour}{\csdef{@dtm@#1@TZhour}{0}}{}%
1500   \ifcsundef{@dtm@#1@TZminute}{\csdef{@dtm@#1@TZminute}{0}}{}%
1501 }

```

`\DTMsaveddatetojulianday` 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.

```

1502 \newrobustcmd*{\DTMsaveddatetojulianday}[2]{%
1503   \ifcsundef{@dtm@#1@year}%
1504   {%
1505     \PackageError{datetime2-calc}{Unknown date ‘#1’}{}%
1506   }%
1507   {%
1508     \pgfcalendardate{#1-#2-#3}\@dtm@julianday}%
1509     \csname @dtm@#1@year\endcsname
1510     -\csname @dtm@#1@month\endcsname
1511     -\csname @dtm@#1@day\endcsname
1512     {#2}%

```

```

1513 }%
1514 }

```

`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.

```

1515 \newrobustcmd*{\DTMsaveddateoffsettojulianday}[3]{%
1516   \ifcsundef{@dtm@#1@year}%
1517   {%
1518     \PackageError{datetime2-calc}{Unknown date '#1'}{}}%
1519   }%
1520   {%
1521     \pgfcalendardatetojulian
1522     {\csname @dtm@#1@year\endcsname
1523     -\csname @dtm@#1@month\endcsname
1524     -\csname @dtm@#1@day\endcsname
1525     +#2}
1526     {#3}%
1527   }%
1528 }

```

`\DTMifdate` Test a saved date using `\pgfcalendarifdate`

```

1529 \newrobustcmd*{\DTMifdate}[4]{%
1530   \ifcsundef{@dtm@#1@year}%
1531   {%
1532     \PackageError{datetime2-calc}{Unknown date '#1'}{}}%
1533   }%
1534   {%
1535     \pgfcalendarifdate
1536     {\csname @dtm@#1@year\endcsname
1537     -\csname @dtm@#1@month\endcsname
1538     -\csname @dtm@#1@day\endcsname}
1539     {#2}{#3}{#4}%
1540   }%
1541 }

```

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

```

1542 \newrobustcmd*{\DTMsaveddatediff}[3]{%
1543   \ifcsundef{@dtm@#1@year}%
1544   {%
1545     \PackageError{datetime2-calc}{Unknown date '#1'}{}}%
1546   }%
1547   {%
1548     \ifcsundef{@dtm@#2@year}%
1549     {%
1550       \PackageError{datetime2-calc}{Unknown date '#1'}{}}%
1551     }%
1552   }%

```



```

1553      \pgfcalendaratetotjulian
1554      {\csname @dtm@#1@year\endcsname
1555      -\csname @dtm@#1@month\endcsname
1556      -\csname @dtm@#1@day\endcsname}
1557      {#3}%
1558      \pgfcalendaratetotjulian
1559      {\csname @dtm@#2@year\endcsname
1560      -\csname @dtm@#2@month\endcsname
1561      -\csname @dtm@#2@day\endcsname}
1562      {\@dtm@julianday}%
1563      \advance#3 by -\@dtm@julianday\relax
1564    }
1565  }%
1566 }

```

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

```

1567 \newrobustcmd*{\DTMtozulu}[2]{%
1568   \ifcsundef{@dtm@#1@year}%
1569   {%
1570     \PackageError{datetime2-calc}{Unknown date ‘#1’}{}%
1571   }%
1572   {%
1573     \DTMsaveaszulutime{#2}%
1574     {\DTMfetchyear{#1}}%
1575     {\DTMfetchmonth{#1}}%
1576     {\DTMfetchday{#1}}%
1577     {\DTMfetchhour{#1}}%
1578     {\DTMfetchminute{#1}}%
1579     {\DTMfetchsecond{#1}}%
1580     {\DTMfetchTZhour{#1}}%
1581     {\DTMfetchTZminute{#1}}%
1582   }%
1583 }

```

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

```

\DTMsavetozulutime{<name>}{<year>}{<month>}{<day>}{<hour>}
{<minute>}{<second>}{<tzh>}{<tzm>}

```

```

1584 \newrobustcmd*{\DTMsaveaszulutime}[9]{%
1585   \edef\@dtm@year{\number#2}%
1586   \edef\@dtm@month{\number#3}%
1587   \edef\@dtm@day{\number#4}%
1588   \edef\@dtm@hour{\number#5}%
1589   \edef\@dtm@minute{\number#6}%
1590   \edef\@dtm@second{\number#7}%

```

```

1591 \edef\@dtm@TZhour{\number#8}%
1592 \edef\@dtm@TZminute{\number#9}%
1593 \pgfcalendardatetojulian{\@dtm@year-\@dtm@month-\@dtm@day}{\@dtm@julianday}%

First adjust the minute offset if non-zero
1594 \ifnum\@dtm@TZminute=0\relax
1595 \else
1596 \count@=\@dtm@minute\relax

Add or subtract the offset minute
1597 \ifnum\@dtm@TZhour<0\relax
1598 \advance\count@ by \@dtm@TZminute\relax
1599 \else
1600 \advance\count@ by -\@dtm@TZminute\relax
1601 \fi
1602 \edef\@dtm@minute{\number\count@}%

Does the hour need adjusting?
1603 \ifnum\count@<0\relax
1604 \advance\count@ by 60\relax
1605 \edef\@dtm@minute{\number\count@}%

Need to subtract 1 from the hour but does the day need adjusting?
1606 \ifnum\@dtm@hour=0\relax
1607 \def\@dtm@hour{23}%

Day needs adjusting.
1608 \advance\@dtm@julianday by -1\relax
1609 \else

Subtract 1 from the hour
1610 \count@ = \@dtm@hour\relax
1611 \advance\count@ by -1\relax
1612 \edef\@dtm@hour{\number\count@}%
1613 \fi
1614 \else

Minute isn't negative. Is it  $\geq 60$ ?
1615 \ifnum\count@>59\relax
1616 \advance\count@ by -60\relax
1617 \edef\@dtm@minute{\number\count@}%

Add 1 to the hour
1618 \count@ = \@dtm@hour\relax
1619 \advance\count@ by 1\relax
1620 \edef\@dtm@hour{\number\count@}%

Does the day need adjusting?
1621 \ifnum\@dtm@hour=24\relax
1622 \def\@dtm@hour{00}%
1623 \advance\@dtm@julianday by 1\relax
1624 \fi
1625 \fi

```

```

1626     \fi
1627 \fi

    Now adjust the hour offset if non-zero
1628 \ifnum\@dtm@TZhour=0\relax
1629 \else
1630     \count@=\@dtm@hour\relax
1631     \advance\count@ by -\@dtm@TZhour\relax

    Does the day need adjusting?
1632 \ifnum\count@<0\relax
1633     \advance\count@ by 24\relax
1634     \edef\@dtm@hour{\number\count@}%
1635     \advance\@dtm@julianday by -1\relax
1636 \else
1637     \ifnum\count@>23\relax
1638     \advance\count@ by -24\relax
1639     \edef\@dtm@hour{\number\count@}%
1640     \advance\@dtm@julianday by 1\relax
1641     \else
1642     \edef\@dtm@hour{\number\count@}%
1643     \fi
1644 \fi
1645 \fi
1646 \pgfcalendarjuliantodate{\@dtm@julianday}{\@dtm@year}{\@dtm@month}{\@dtm@day}%
1647 \pgfcalendarjuliantoweekday{\@dtm@julianday}{\count@}%

    Save the results.
1648 \csedef{\@dtm@#1@dow}{\number\count@}%
1649 \cslet{\@dtm@#1@year}{\@dtm@year}%
1650 \cslet{\@dtm@#1@month}{\@dtm@month}%
1651 \cslet{\@dtm@#1@day}{\@dtm@day}%
1652 \cslet{\@dtm@#1@hour}{\@dtm@hour}%
1653 \cslet{\@dtm@#1@minute}{\@dtm@minute}%
1654 \cslet{\@dtm@#1@second}{\@dtm@second}%
1655 \csdef{\@dtm@#1@TZhour}{0}%
1656 \csdef{\@dtm@#1@TZminute}{0}%
1657 }

```

Index

Symbols

<code>\@dtm@currentdow</code>	56, 87	<code>\dtm@hourminsep</code>	46
<code>\@dtm@initialstyle</code>	50	<code>\dtm@minsecsep</code>	46
<code>\@dtm@julianday</code>	86	<code>\dtm@monthdaysep</code>	46
<code>\@dtm@loadedregions</code>	84	<code>\dtm@timezonesep</code>	46
<code>\@dtm@parsedate</code>	51, 87	<code>\dtm@yearmonthsep</code>	46
<code>\@dtm@parsetime</code>	51	<code>\DTMcentury</code>	25, 60
<code>\@dtm@parsetimestamp</code>	52	<code>\DTMclearmap</code>	27, 63
<code>\@dtm@parsetimezn</code>	51	<code>\DTMcurrenttime</code>	11, 57
<code>\@dtm@parsezone</code>	51	<code>\DTMcurrentzone</code>	11, 57
<code>\@dtm@requiremodule</code>	84	<code>\DTMDate</code>	9, 57
<code>\@dtm@setusecalc</code>	49	<code>\DTMdate</code>	9, 56
<code>\@dtm@unknown@style</code>	67	<code>\DTMdefboolkey</code>	85
<code>\@dtm@unknownstyle</code>	67	<code>\DTMdefchoicekey</code>	85
<code>\@dtm@usecalc</code>	49	<code>\DTMdefkey</code>	85
<code>\@dtm@warning</code>	50	<code>\DTMdefzonemap</code>	27, 63
B			
babel package		<code>\DTMDisplay</code>	12, 59
.. 4, 6, 9, 11, 13, 29–34, 37, 38, 86		<code>\DTMdisplay</code>	12, 58
C			
calc (option)	49	<code>\DTMdisplaydate</code>	8, 56
D			
date style	18, 19, 23	<code>\DTMdisplaydate</code>	8, 31, 56
date-time style	12	<code>\DTMdisplaytime</code>	10, 57
datesep (option)	46	<code>\DTMdisplayzone</code>	11, 57
datetime package	4	<code>\DTMdivhundred</code>	26, 60
datetime2-calc package	7,	<code>\DTMfetchday</code>	16, 81
9, 10, 13, 38, 39, 42, 44, 49–51, 86		<code>\DTMfetchdow</code>	16, 81
datetimesep (option)	47	<code>\DTMfetchhour</code>	16, 81
dayyearsep (option)	47	<code>\DTMfetchminute</code>	16, 81
ddmmyyyy style	21, 71	<code>\DTMfetchmonth</code>	16, 80
default style		<code>\DTMfetchsecond</code>	17, 81
.. 5, 7, 20–22, 29–31, 33, 45, 48, 67		<code>\DTMfetchTZhour</code>	17, 81
display style	8	<code>\DTMfetchTZminute</code>	17, 81
dmyy style	21, 73	<code>\DTMfetchyear</code>	16, 80
dmyyyy style	21, 72	<code>\DTMhaszonemap</code>	27, 63
document (environment) ..	9, 26, 37, 39	<code>\DTMifbool</code>	85
<code>\dtm@dayyearsep</code>	46	<code>\DTMifcaseregional</code>	48
		<code>\DTMifdate</code>	43, 88
		<code>\DTMifsaveddate</code>	16, 83
		<code>\DTMlangsetup</code>	34, 39, 86
		<code>\DTMmakeglobal</code>	15, 80
		<code>\DTMNatoZoneMaps</code>	27, 64
		<code>\DTMnewdatestyle</code>	23, 61

<code>\DTMnewstyle</code>	23, 64	<code>\DTMuseumzonemap</code>	26, 63
<code>\DTMnewtimestyle</code>	23, 61	<code>\DTMuseumzonemapordefault</code> ..	27, 63
<code>\DTMnewzonestyle</code>	23, 62		
<code>\DTMNow</code>	12, 58	E	
<code>\DTMnow</code>	12, 58	en-GB style	6, 31–33, 41
<code>\DTMprovidedatestyle</code>	25, 61	en-GB-numeric style	6, 31–33, 41
<code>\DTMprovidestyle</code>	25, 65	english style	33
<code>\DTMprovidetimestyle</code>	25, 62	environments:	
<code>\DTMprovidezonestyle</code>	25, 63	document	9, 26, 37, 39
<code>\DTMrenewdatestyle</code>	24, 61	etoolbox package	46
<code>\DTMrenewstyle</code>	24, 65		
<code>\DTMrenewtimestyle</code>	24, 62	F	
<code>\DTMrenewzonestyle</code>	24, 62	full style	12, 18, 19, 23
<code>\DTMresetzones</code>	27, 64		
<code>\DTMsaveaszulutime</code>	44, 89	H	
<code>\DTMsavedate</code>	13, 78	hhmm style	23, 78
<code>\DTMsaveddatediff</code>	43, 88	hmmss style	22, 77
<code>\DTMsaveddateoffsettojuliandate</code>	43	hmmss package	22
<code>\DTMsaveddateoffsettojulianday</code>	88	hourminsep (option)	47
<code>\DTMsaveddatetojuliandate</code> ...	42	hyperref package	26, 60
<code>\DTMsaveddatetojulianday</code> ...	87		
<code>\DTMsavefilemoddate</code>	14, 52	I	
<code>\DTMsavejulianday</code>	42, 87	iso style	21, 36, 37, 48, 68
<code>\DTMsavenoparsedate</code>	13, 79		
<code>\DTMsavenow</code>	14, 80	M	
<code>\DTMsavetime</code>	13, 79	map style	22, 23, 78
<code>\DTMsavetimestamp</code>	14, 79	mdyy style	21, 76
<code>\DTMsavetimezn</code>	14, 79	mdyyy style	21, 75
<code>\DTMsep</code>	26, 60	mfirstuc package	8
<code>\DTMsetbool</code>	85	minsecsep (option)	47
<code>\DTMsetcurrentzone</code>	56	mmddyyy style	21, 74
<code>\DTMsetdatestyle</code>	18, 66	monthdaysep (option)	47
<code>\DTMsetstyle</code>	18, 66		
<code>\DTMsettimestyle</code>	18, 66	O	
<code>\DTMsetup</code>	9, 20, 31, 39, 50	options (definitions):	
<code>\DTMsetzonestyle</code>	18, 66	calc	49
<code>\DTMshowmap</code>	64	datesep	46
<code>\DTMtexpdfstring</code>	26, 60	datetimesep	47
<code>\DTMtime</code>	11, 57	dayyearsep	47
<code>\DTMttozulu</code>	44, 89	hourminsep	47
<code>\DTMtwdigits</code>	25, 59	minsecsep	47
<code>\DTMUse</code>	16, 83	monthdaysep	47
<code>\DTMuse</code>	15, 83	showdate	47
<code>\DTMUsedate</code>	15, 81	showdow	49
<code>\DTMusedate</code>	15, 81	showisoZ	48
<code>\DTMusetime</code>	15, 82	showseconds	47
<code>\DTMuseumzone</code>	15, 82	showzone	48
		showzoneminutes	48
		style	50
		timesep	47

timezoneseq	47	pgf package	43, 44
useregional	48	pgfcalendar package	9, 10, 38, 42, 44, 49, 87
warn	50	polyglossia package	9, 29–33, 37, 38, 86
yearmonthsep	47	\ProvidesDateTimeModule	85
P			
package options:		R	
british	33, 38	\RequireDateTimeModule	84
calc	9, 13, 38, 39, 42	S	
datesep	20, 21, 36, 40, 41	scottish style	33
datetimesep	20, 21, 37	scottish-numeric style	33
dayyearsep	21, 36	showdate (option)	47
en-GB	33	showdow (option)	49
english	33	showisoZ (option)	48
hourminsep	20, 23, 28, 36	showseconds (option)	47
mapzone	27	showzone (option)	48
minsecsep	20, 36	showzoneminutes (option)	48
monthdaysep	20, 21, 36	style (option)	50
scottish	33	T	
showdate	21, 37	\time	11
showdow	7, 9, 13, 39	time style	11, 18, 19, 23
true	9, 39, 42	timesep (option)	47
showisoZ	20–23, 28, 37	timezoneseq (option)	47
false	5	\Today	9, 56
showseconds	11, 20, 21, 37	\today	8, 56
false	5	tracklang package	30–34, 39, 46, 50, 84, 86
showzone	11, 12, 21, 37	U	
false	5	ukdate package	4
showzoneminutes	20, 21, 23, 28, 37	useregional (option)	48
style	29, 38, 39, 86	W	
timesep	36	warn (option)	50
timezoneseq	21, 37	X	
UKenglish	33	xkeyval package	40, 46
usnumerical		Y	
false	38	yearmonthsep (option)	47
text	38	yyyymd style	21, 70
useregional	6, 29, 31–33, 37, 38	Z	
false	37, 38, 50	zone style	11, 18, 19, 23
true	33		
warn	39		
yearmonthsep	20, 26, 28, 36		
partial style	19, 23		
pdf style	22, 36, 37, 48, 69		
\pdfcreationdate	11		
\pdfinfo	22		