ETM Users Manual

Contents

Overview **3**

Sample entries . . . 4
Starting etm . . . 5
Views . . . 9
Creating New Items . . . 13
Editing Existing Items . . . 13
Timers . . . 14
Sharing with other calendar applications . . . 16
Tools . . . 18
Calendars . . . 20
Data Organization . . . 21
Colors . . . 23
Internationalization . . . 23

Item types **25**

~ Action . . . 25
* Event . . . 26
^ Occasion . . . 26
! Note . . . 26
-, % and + Tasks . . . 27
$ In basket . . . 28
? Someday maybe . . . 28
Comment . . . 28
= Defaults . . . 29

@Keys **29**

@a alert . . . 29
@b beginby . . . 31
@c context . . . 31
@d description . . . 31
@e extent . . . 31
@f done[; due] . . . 32
@g goto . . . 33
@h history . . . 33
@i invitees . . . 33
@j job . . . 34
@k keyword . . . 34
@l location . . . 35
@m memo . . . 35
@n noshow . . . 35
@o overdue . . . 36
@p priority . . . 36
@q datetime . . . 36
@r repetition rule . . . 36
@s starting datetime . . . 38
@t tags . . . 39
@u user . . . 39
@v action_rates key . . . 39
@w action_markups key . . . 39
@x expense . . . 39
@z time zone . . . 40
@+ include . . . 40
@- exclude . . . 40

Dates **40**

Fuzzy dates 40

Time periods 41

Time zones 41

Anniversary substitutions 42

Easter 42

Preferences **43**

Template expansions 43

Options 45

Custom view **64**

View type 64

Groupby setting 64

Groupby examples 66

View Options 67

Saving view specifications 70

Shortcuts **70**

Menu 70

Edit 72

Overview

In contrast to most calendar/todo applications, creating items (events, tasks, and so forth) in etm does not require filling out fields in a form. Instead, items are created as free-form text entries using a simple, intuitive format and stored in plain text files.

Dates in the examples below are entered using *fuzzy parsing* - e.g., `+7` for seven days from today, `fri` for the first Friday on or after today, `+1/1` for the first day of next month, `sun - 6d` for Monday of the current week. See Dates for details.

Sample entries

- A sales meeting (an event) [s]tarting seven days from today at 9:00am with an [e]xtent of one hour and a default [a]lert 5 minutes before the start:

```
* sales meeting @s +7 9a @e 1h @a 5
```

- The sales meeting with another [a]lert 2 days before the meeting to (e)mail a reminder to a list of recipients:

```
* sales meeting @s +7 9a @e 1h @a 5
  @a 2d: e; who@when.com, what@where.org
```

- Prepare a report (a task) for the sales meeting [b]eginning 3 days early:

```
- prepare report @s +7 @b 3
```

- A period [e]xtending 35 minutes (an action) spent working on the report yesterday:

```
~ report preparation @s -1 @e 35
```

- Get a haircut (a task) on the 24th of the current month and then [r]epeatedly at (d)aily [i]ntervals of (14) days and, [o]n completion, (r)estart from the completion date:

```
- get haircut @s 24 @r d &i 14 @o r
```

- Payday (an occasion) on the last week day of each month. The `&s -1` part of the entry extracts the last date which is both a weekday and falls within the last three days of the month):

```
^ payday @s 1/1 @r m &w MO, TU, WE, TH, FR
  &m -1, -2, -3 &s -1
```

- Take a prescribed medication daily (a reminder) [s]tarting today and [r]epeating (d)aily at [h]ours 10am, 2pm, 6pm and 10pm [u]ntil (12am on) the fourth day from today. Trigger the default [a]lert zero minutes before each reminder:

```
* take Rx @s +0 @r d &h 10, 14, 18, 22 &u +4 @a 0
```

- Move the water sprinkler (a reminder) every thirty mi[n]utes on Sunday afternoons using the default alert zero minutes before each reminder:

```
* Move sprinkler @s 1 @r n &i 30 &w SU &h 14, 15, 16, 17 @a 0
```

To limit the sprinkler movement reminders to the [M]onths of April through September each year append `&M 4, 5, 6, 7, 8, 9` to the @r entry.

- Grandparent's day (an occasion) each year on the first Sunday in September after Labor day:

```
^ Grandparent's Day @s 2012-09-01
  @r y &M 9 &w SU &m 7, 8, 9, 10, 11, 12, 13
```

- Presidential election day (an occasion) every four years on the first Tuesday after a Monday in November:

```
^ Presidential Election Day @s 2012-11-06
  @r y &i 4 &M 11 &w TU &m 2, 3, 4, 5, 6, 7, 8
```

- Join the etm discussion group (a task) [s]tarting 14 days from today. Because of the @g (goto) link, pressing *G* when this item is selected in the gui would open the link using the system default application which, in this case, would be your default browser:

```
- join the etm discussion group @s +14
  @g groups.google.com/group/eventandtaskmanager/topics
```

Starting etm

To start the etm GUI open a terminal window and enter `etm` at the prompt:

```
$ etm
```

If you have not done a system installation of etm you will need first to cd to the directory where you unpacked etm.

Note: if you change the window size and/or position of the etm window on your display and quit etm from the etm file menu, then the closing size and position will be restored when you restart etm.

You can add a command to use the CLI instead of the GUI. For example, to get the complete command line usage information printed to the terminal window just add a question mark:

```
$ etm ?
Usage:

    etm [logging level] [path] [?] [acmsv]

With no arguments, etm will set logging level 3 (warn), use settings from
the configuration file ~/.etm/etmtk.cfg, and open the GUI.

If the first argument is an integer not less than 1 (debug) and not greater
than 5 (critical), then set that logging level and remove the argument.

If the first (remaining) argument is the path to a directory that contains
a file named etmtk.cfg, then use that configuration file and remove the
argument.

If the first (remaining) argument is one of the commands listed below, then
execute the remaining arguments without opening the GUI.

    a ARG   display the agenda view using ARG, if given, as a filter.
    c ARGS  display a custom view using the remaining arguments as the
            specification. (Enclose ARGS in single quotes to prevent shell
            expansion.)
    d ARG   display the day view using ARG, if given, as a filter.
    k ARG   display the keywords view using ARG, if given, as a filter.
    m INT   display a custom view using the remaining argument, which
            must be a positive integer, to display a custom view using the
            corresponding entry from the file given by report_specifications
            in etmtk.cfg.
            Use ? m to display the numbered list of entries from this file.
    n ARG   display the notes view using ARG, if given, as a filter.
    N ARGS  Create a new item using the remaining arguments as the item
            specification. (Enclose ARGS in single quotes to prevent shell
            expansion.)
    p ARG   display the path view using ARG, if given, as a filter.
    t ARG   display the tags view using ARG, if given, as a filter.
    v       display information about etm and the operating system.
    ? ARG   display (this) command line help information if ARGS = '' or,
            if ARGS = X where X is one of the above commands, then display
            details about command X. 'X ?' is equivalent to '? X'.
```

For example, you can print your agenda to the terminal window by adding the letter “a”:

```
$ etm a
Today
```

```
  > set up luncheon meeting with Joe Smith            4d
Tomorrow
  * test command line event                       3pm ~ 4pm
  * Aerobics                                      5pm ~ 6pm
  - follow up with Mary Jones
Now
  Available
    - Hair cut                                       -1d
Next
  errands
    - milk and eggs
  phone
    - reservation for Saturday dinner
Someday
  ? lose weight and exercise more
```

You can filter the output by adding a (case-insensitive) argument:

```
$ etm a hair
Now
  Available
    - Hair cut                                       -1d
```

or `etm d mar .*2014` to show your items for March, 2014.

You can add a question mark to a command to get details about the commmand, e.g.:

```
Usage:

    etm c <type> <groupby> [options]

Generate a custom view where type is either 'a' (action) or 'c' (composite).
Groupby can include *semicolon* separated date specifications and
elements from:
    c context
    f file path
    k keyword
    t tag
    u user

A *date specification* is either
    w:   week number
or a combination of one or more of the following:
    yy:   2-digit year
```

```
    yyyy:   4-digit year
    MM:   month: 01 - 12
    MMM:   locale specific abbreviated month name: Jan - Dec
    MMMM:   locale specific month name: January - December
    dd:   month day: 01 - 31
    ddd:   locale specific abbreviated week day: Mon - Sun
    dddd:   locale specific week day: Monday - Sunday

Options include:
    -b begin date
    -c context regex
    -d depth (CLI type a only)
    -e end date
    -f file regex
    -k keyword regex
    -l location regex
    -o omit (type c only)
    -s summary regex
    -S search regex
    -t tags regex
    -u user regex
    -w column 1 width
    -W column 2 width

Example:

    etm c 'c ddd, MMM dd yyyy -b 1 -e +1/1'
```

Note: The CLI offers the same views and reporting, with the exception of week and month view, as the GUI. It is also possible to create new items in the CLI with the `n` command. Other modifications such as copying, deleting, finishing and so forth, can only be done in the GUI or, perhaps, in your favorite text editor. An advantage to using the GUI is that it provides auto-completion and validation.

Tip: If you have a terminal open, you can create a new item or put something to finish later in your inbox quickly and easily with the "N" command. For example,

```
    etm N '123 456-7890'
```

would create an entry in your inbox with this phone number. (With no type character an "$" would be supplied automatically to make the item an inbox entry and no further validation would be done.)

Views

All views display only items consistent with the current choices of active calendars. Click the settings icon on the left side of the top menu bar to choose active calendars.

Week and month views have three panes. The top one displays a graphic illustration of scheduled times for the relevant period, the middle one displays an tree view of items grouped by date and the bottom one displays detail information. Custom view also has three panes but the top one is an entry area for providing the specification for the custom view. All other views have two panes - a tree view in the top pane and details in the bottom pane.

If a (case-insensitive) filter is entered then the display in the tree view will be limited to items that match somewhere in either the branch or the leaf. Relevant branches will automatically be expanded to show matches.

In week and month views, *Home* selects the current date. In all views other than week and month, *Home* selects the first item in the tree pane.

In all views, pressing *Return* with an item selected will open a context menu with options to copy, delete, edit and so forth.

In all views, clicking in the details panel with an item selected will open the item for editing if it is not repeating and otherwise prompt for the instance(s) to be changed.

In all views, pressing *O* will open a dialog to choose the outline depth. Pressing *L* will toggle the display of a column displaying item *labels* where, for example, an item with @a, @d and @u fields would have the label "adu". Pressing *S* will show a text verion of the current display suitable for copy and paste. The text version will respect the current outline depth in the view.

In custom view it is possible to export the current view in either text or CSV (comma separated values) format to a file of your choosing.

Note. In custom view you need to move the focus from the view specification entry field in order for the shortcuts *O*, *L* and *S* to work.

In all views:

- if an item is selected:
  - pressing *Shift-H* will show a history of changes for the file containing the selected item, first prompting for the number of changes.
  - pressing *Shift-X* will export the selected item in iCal format.
- if an item is not selected:
  - pressing *Shift-H* will show a history of changes for all files, first prompting for the number of changes.

– pressing *Shift-X* will export all items in active calendars in iCal format.

Agenda View

What you need to know now beginning with your schedule for the next few days and followed by items in these groups:

- **In basket**: In basket items and items with missing types or other errors.
- **Now**: All *scheduled* (dated) tasks whose due dates have passed including delegated tasks and waiting tasks (tasks with unfinished prerequisites) grouped by available, delegated and waiting and, within each group, by the due date.
- **Next**: All *unscheduled* (undated) tasks grouped by context (home, office, phone, computer, errands and so forth) and sorted by priority and extent. These tasks correspond to GTD's *next actions*. These are tasks which don't really have a deadline and can be completed whenever a convenient opportunity arises. Check this view, for example, before you leave to run errands for opportunities to clear other errands.
- **Someday**: Someday (maybe) items for periodic review.

Note: Finished tasks, actions and notes are not displayed in this view.

Week and Month Views

These views only differ in whether the graphic in the top pane describes a week or a month. All dated items appear in the middle, tree pane in these view, grouped by date and sorted by starting time and item type. Displayed items include:

- All non-repeating, dated items.
- All repetitions of repeating items with a finite number of repetitions. This includes 'list-only' repeating items and items with `&u` (until) or `&t` (total number of repetitions) entries.
- For repeating items with an infinite number of repetitions, those repetitions that occur within the first `weeks_after` weeks after the current week are displayed along with the first repetition after this interval. This assures that at least one repetition will be displayed for infrequently repeating items such as voting for president.

The graphic display in the top pane has a square cell for each week/month date. Within this cell, scheduled, *busy* times are indicated by segments of a square busy border that surrounds the date. This border can be regarded as a 24-hour clock face that proceeds clockwise from 12am at the lower, left-hand corner with 6 hour segments for each side: 12am - 6am moving upward on the left side, 6am - 12pm moving rightward along the top, 12pm - 6pm moving downward along the right side and, finally, 6pm - 12pm moving leftward along the bottom. When nothing is scheduled for a date, the border is blank. When only one event is scheduled for a date, say from 9am until 3pm, then the border would be colored from the middle of the top side (9am) around the top, right-hand corner and downward to the middle of the right side (3pm). When other periods are scheduled, corresponding portions of the border would be colored. If two or more scheduled periods overlap, then a small, red box appears in the lower, left-hand corner of the border to warn of the conflict.

When the top pane has the focus, the left/right cursor keys move to the previous/subsequent week or month and the up/down cursor keys move to the previous/subsequent date. Either *Home* or *Space* moves the display to the current date. Pressing *J* will first prompt for a fuzzy-parsed date and then "jump" to the specified date. Whenever a date is selected in the top pane, the date tree in the middle pane is scrolled, if necessary, to display the selected date first. Whenever a date is selected in either week or month view, the same date is automatically becomes the selection in the other view as well.

Note: If a date is selected for which no items are scheduled, then the last date with scheduled items on or before the selected date will become the selected date in the middle, tree pane.

Tip: Want to see your next appointment with Dr. Jones? Switch to day view and enter "jones" in the filter.

Tip. You can display a list of busy times or, after providing the needed period in minutes, a list of free times that would accommodate the requirement within the selected week/month. Both options are in the *View* menu.

Week View

Events and occasions displayed graphically by week with one column for each day. Left and right cursor keys change, respectively, to the previous and next week. Up and down cursor keys select, respectively, the previous and next items within the given week. Items can also be selected by moving the mouse over the item. The details for the selected item are displayed at the bottom of the screen. Pressing return with an item selected or double-clicking an item opens a context menu. Control-clicking an unscheduled time opens a dialog to create an event for that date and time.

Month View

Events and occasions displayed graphically by month. Left and right cursor keys change, respectively, to the previous and next month. Up and down cursor keys select, respectively, the previous and next days within the given month. Days can also be selected by moving the mouse over the item. A list of occasions and events for the selected day is displayed at the bottom of the screen. Double clicking a date or pressing *Return* with a date selected opens a dialog to create an item for that date.

The current date and days with occasions are highlighted.

Tip. You can display a list of busy times or, after providing the needed period in minutes, a list of free times that would accommodate the requirement within the selected month. Both options are in the *View* menu.

Tag View

All items with tag entries grouped by tag and sorted by type and *relevant datetime*. Note that items with multiple tags will be listed under each tag.

Tip: Use the filter to limit the display to items with a particular tag.

Keyword View

All items grouped by keyword and sorted by type and *relevant datetime*.

Path View

All items grouped by file path and sorted by type and *relevant datetime*. Use this view to review the status of your projects.

The *relevant datetime* is the past due date for any past due task, the starting datetime for any non-repeating item and the datetime of the next instance for any repeating item.

Note: Items that you have "commented out" by beginning the item with a `#` will only be visible in this view.

Note View

All notes grouped and sorted by keyword and summary.

Custom

Design your own view. See Custom view for details.

Creating New Items

Items of any type can be created by pressing *N* in the GUI and then providing the details for the item in the resulting dialog.

An event can also be created by double-clicking in a date cell in either Week or Month View - the corresponding date will be entered as the starting date when the dialog opens.

When editing an item, clicking on *Finish* or pressing *Shift-Return* will validate your entry. If there are errors, they will be displayed and you can return to the editor to correct them. If there are no errors, this will be indicated in a dialog, e.g.,

```
Task scheduled for Tue Jun 03

Save changes and exit?
```

Tip. When creating or editing a repeating item, pressing *Finish* will also display a list of instances that will be generated.

Click on *Ok* or press *Return* or *Shift-Return* to save the item and close the editor. Click on *Cancel* or press *Escape* to return to the editor.

If this is a new item and there are no errors, clicking on *Ok* or pressing *Return* will open a dialog to select the file to store the item with the current monthly file already selected. Pressing *Shift-Return* will bypass the file selection dialog and save to the current monthly file.

Editing Existing Items

Double-clicking an item or pressing *Return* when an item is selected will open a context menu of possible actions:

- Copy
- Delete
- Edit
- Edit file
- Finish (unfinished tasks only)
- Reschedule
- Schedule new
- Klone as timer
- Open link (items with `@g` entries only)
- Show user details (items with `@u` entries only)

When either *Copy* or *Edit* is chosen for a repeating item, you can further choose:

1. this instance
2. this and all subsequent instances
3. all instances

When *Delete* is chosen for a repeating item, a further choice is available:

4. all previous instances

Tip: Use *Reschedule* to enter a date for an undated item or to change the scheduled date for the item or the selected instance of a repeating item. All you have to do is enter the new (fuzzy parsed) datetime.

Timers

countdown timer

To start a countdown timer press *z*, change the default number of minutes if necessary and press *Return*. The time that the timer will expire will be displayed in the status bar with a `-` prefix.

If `countdown_command` is given in *etmtk.cfg*, then it will be executed when the timer expires and the countdown message dialog will appear with the last chosen number of minutes as the default. You can either press *Return* to start another countdown or press *Escape* to cancel. If activated, the time that the countdown timer will expire will be displayed in the status bar.

snooze timer

When the last alert of type `m` (message) is triggered for an item, the alert dialog that is displayed offers the option of snoozing, i.e., repeating the alert after a number of minutes you choose. If activated, the alert corresponding to snooze timer can be displayed along with any other remaining alerts using *Tools/Show remaining alerts*.

If `snooze_command` is given in *etmtk.cfg*, then it will be executed when the snooze timer expires and the alert message dialog will appear with the last chosen number of minutes as the default. You can either press *Return* to snooze again or press *Escape* to cancel.

action timer

For people who bill their time or just like to keep track of how they spend their time, etm allows you to create an action by pressing *T* to start a timer. You

will see an entry area with a list of any existing timers below. As you enter characters in the entry area, the list below will shrink to those whose beginnings match the characters you've entered. You can either select a timer from the list to start that timer or enter new name in the entry area to create and start a new timer. If a timer is running, it will automatically be paused when you start a new timer or switch to another timer.

Tip. Devoting time to two different clients this morning? Create two timers, one for each client and just switch back and forth using *T* when you switch from one client to the other. The timers are ordered in the list so that the most recently paused will be at the top.

While a timer is selected, the name, elapsed time and status - running or paused - is displayed in the status bar along with the number of active timers in parentheses. Pressing *I* toggles the timer between running and paused. You can configure etm's options to, for example, play one sound at intervals when a timer is running and another sound when the selected timer is paused and you can also specify the length of the interval and the volume.

When one or more timers are active and none are running, idle time is accumulated and displayed, by default, in the status bar. The idle time display can be toggled on and off and accumulated idle time can be reset to zero. It is also possible to transfer minutes from accumulated idle time to the current action timer.

When you have one or more active timers, you can press *Shift-T* to select one to finish. The selected timer will be paused if it is running and you will be presented with an entry area to create a new action with the following details already filled in: `~ timer name @s starting datetime @e elapsed time`. You can edit this entry in any way you like and then save it. When you do so, this timer will be removed from your list of active timers. You can also press *Shift-I* to select a timer to delete. Any accumulated time for the selected timer will be added to the accumulated idle time and the timer will be removed from the list of active timers.

It is also possible to start a timer by selecting an event, note, task or whatever, from one of *etm*'s Views and then choosing *Item/Klone as timer* from the menu or pressing *K*. A start timer dialog will be opened with the summary of the item you selected as the name together with any @-keys from the selected item that are listed in `action_keys` in your `etmtk.cfg`. You can edit this entry if you like or just press *Return* to accept it and start the timer. If you already have an active timer with this name, it will be restarted. Otherwise a new timer will be created and started.

Tip. Suppose you have a client, John Smith, and will be doing some work for him this morning relating to the project "Motion". If you don't already have a task relating to this begin by creating one for today, June 16, 2015, by pressing *N* and entering

```
- work @k SmithJohn:Motion @s +0
```

The first activity related to this task involves a phone call to Sally. Select the task you just created and then press *K* to start a timer. Change `work` to `call Sally` and press *Return* to start the timer. When you've finished the call, press *I* to pause the timer. Based on this phone conversation, you decide the next activity should be to review Local Rule 4567, so once again select the task, press *K* and then change `work` to `review Local Rule 4567` and press *Return* to start this timer. When you're done, once again press *I* to pause this timer. You can repeat this process as often as you like. If you need to spend more time on 4567, press *T* and select it from the list of timers. When you're done, you can press *Shift-T* to select a timer from the list and finish it. Selecting the "call Sally" timer would produce an entry for the new action something like the following

```
~ call Sally @k SmithJohn:Motion @s 2015-06-16 9:27am @e 12m
```

You can edit this action if you like, but it is already set up to bill 12 minutes to the "Motion" project for client "John Smith" for an activity labeled "call Sally" and will appear as such in reports you generate for this period, so you can just save it as it is. Do the same with your other timers and you will have a complete record of time spent by client, project and activity for the day.

The state of your active timers is saved whenever you quit etm using by choosing *Quit* from the file menu or using the shortcut so that whenever you restart etm on the same day, the active timers will be restored.

If etm is running when a new day begins (midnight local time) or if you stop etm and start it again on a later date, in-basket entries for each of your active timers will be created in the relevant monthly file. These entries will be exactly the same as if you had finished each of the timers save for the use of `$` (in basket) rather than `~` (action) as the type character. You can edit or delete these as you wish. If a timer is selected (displayed in the status bar), then a new timer with the same name will be created for the new date but with zero elapsed time. If the timer was running at midnight, then the new timer will also be running. Idle time will automatically be reset to zero.

Sharing with other calendar applications

Both export and import are supported for files in iCalendar format in ways that depend upon settings in `etmtk.cfg`.

If an absolute path is entered for `current_icsfolder`, for example, then `.ics` files corresponding to the entries in `calendars` will be created in this folder and updated as necessary. If there are no entries in calendars, then a single file, `all.ics`, will be created in this folder and updated as necessary.

If an item is selected, then pressing Shift-X in the gui will export the selected item in iCalendar format to `icsitem_file`. If an item is not selected, pressing Shift-X will export the active calendars in iCalendar format to `icscal_file`.

If `icssync_folder` is given, then files in this folder with the extension `.txt` and `.ics` will automatically kept concurrent using export to iCalendar and import from iCalendar. I.e., if the `.txt` file is more recent than than the `.ics` then the `.txt` file will be exported to the `.ics` file. On the other hand, if the `.ics` file is more recent then it will be imported to the `.txt` file. In either case, the contents of the file to be updated will be overwritten with the new content and the last acess/modified times for both will be set to the current time.

If `ics_subscriptions` is given, it should be a list of [URL, FILE] tuples. The URL is a calendar subscription, e.g., for a Google Calendar subscription the URL, FILE tuple would be something like:

```
['https://www.google.com/calendar/ical/.../basic.ics', 'personal/google.txt']
```

With this entry, pressing Shift-M in the gui would import the calendar from the URL, convert it from ics to etm format and then write the result to `personal/google.txt` in the etm data directory. Note that this data file should be regarded as read-only since any changes made to it will be lost with the next subscription update.

Finally, when creating a new item in the etm editor, you can paste an iCalendar entry such as the following VEVENT:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ForeTees//NONSGML v1.0//EN
CALSCALE:GREGORIAN
METHOD:PUBLISH
BEGIN:VEVENT
UID:1403607754438-11547@127.0.0.1-33
DTSTAMP:20140624T070234
DTSTART:20140630T080000
SUMMARY:8:00 AM Tennis Reservation
LOCATION:Governors Club
DESCRIPTION: Player 1: ...

URL:http://www1.foretees.com/governorsclub
END:VEVENT
END:VCALENDAR
```

When you press *Finish*, the entry will be converted to etm format

```
^ 8:00 AM Tennis Reservation @s 2014-06-30 8am
@d Player 1: ...
@z US/Eastern
```

and you can choose the file to hold it.

The following etm and iCalendar item types are supported:

- export from etm:
  - occasion to VEVENT without end time
  - event (with or without extent) to VEVENT
  - action to VJOURNAL
  - note to VJOURNAL
  - task to VTODO
  - delegated task to VTODO
  - task group to VTODO (one for each job)
- import from iCalendar
  - VEVENT without end time to occasion
  - VEVENT with end time to event
  - VJOURNAL to note
  - VTODO to task

Tools

Date and time calculator

Enter an expression of the form `x [+-] y` where `x` is a date and `y` is either a date or a time period if `-` is used and a time period if `+` is used. Both `x` and `y` can be followed by timezones, e.g.,

```
4/20 6:15p US/Central - 4/20 4:50p Asia/Shanghai:

14h25m
```

or

```
4/20 4:50p Asia/Shanghai + 14h25m US/Central:

2014-04-20 18:15-0500
```

Fuzzy dates (other than relative date expressions using `+` or `-`) can be used to specify date entries. The local timezone is assumed when none is given.

Available dates calculator

Need to see a list of possible dates for a meeting? Get a list of busy dates from each of the members of the group and then use an expression of the form

```
start; end; busy
```

where start and end are dates and busy is a comma separated list of the busy dates or intervals for the members. E.g., if your group needs to meet between 6/1 and 6/30 and the members indicate that they cannot meet on 6/2, 6/14-6/22, 6/5-6/9, 6/11-6/15 or 6/17-6/29, then entering

```
6/1; 6/30; 6/2, 6/14-6/22, 6/5-6/9, 6/11-6/15, 6/17-6/29
```

would give:

```
Sun Jun 01
Tue Jun 03
Wed Jun 04
Tue Jun 10
Mon Jun 30
```

as the possible dates for the meeting.

Yearly calendar

Gives a display such as

```
     January 2014          February 2014          March 2014
Mo Tu We Th Fr Sa Su  Mo Tu We Th Fr Sa Su  Mo Tu We Th Fr Sa Su
       1  2  3  4  5                  1  2                  1  2
 6  7  8  9 10 11 12   3  4  5  6  7  8  9   3  4  5  6  7  8  9
13 14 15 16 17 18 19  10 11 12 13 14 15 16  10 11 12 13 14 15 16
20 21 22 23 24 25 26  17 18 19 20 21 22 23  17 18 19 20 21 22 23
27 28 29 30 31        24 25 26 27 28        24 25 26 27 28 29 30
                                            31

     April 2014            May 2014              June 2014
Mo Tu We Th Fr Sa Su  Mo Tu We Th Fr Sa Su  Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6            1  2  3  4                    1
 7  8  9 10 11 12 13   5  6  7  8  9 10 11   2  3  4  5  6  7  8
14 15 16 17 18 19 20  12 13 14 15 16 17 18   9 10 11 12 13 14 15
21 22 23 24 25 26 27  19 20 21 22 23 24 25  16 17 18 19 20 21 22
```

```
 28 29 30                26 27 28 29 30 31       23 24 25 26 27 28 29
                                                 30

      July 2014               August 2014            September 2014
Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6                 1  2  3     1  2  3  4  5  6  7
 7  8  9 10 11 12 13     4  5  6  7  8  9 10     8  9 10 11 12 13 14
14 15 16 17 18 19 20    11 12 13 14 15 16 17    15 16 17 18 19 20 21
21 22 23 24 25 26 27    18 19 20 21 22 23 24    22 23 24 25 26 27 28
28 29 30 31             25 26 27 28 29 30 31    29 30

      October 2014            November 2014          December 2014
Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su    Mo Tu We Th Fr Sa Su
       1  2  3  4  5                    1  2     1  2  3  4  5  6  7
 6  7  8  9 10 11 12     3  4  5  6  7  8  9     8  9 10 11 12 13 14
13 14 15 16 17 18 19    10 11 12 13 14 15 16    15 16 17 18 19 20 21
20 21 22 23 24 25 26    17 18 19 20 21 22 23    22 23 24 25 26 27 28
27 28 29 30 31          24 25 26 27 28 29 30    29 30 31
```

Left and right cursor keys move backward and forward a year at a time, respectively, and pressing the Home key returns to the current year.

History of changes

This requires that either *git* or *mercurial* is installed. If an item is selected show a history of changes to the file that contains the item. Otherwise show a history of changes for all etm data files. In either case, choose an integer number of the most recent changes to show or choose 0 to show all changes.

Calendars

etm supports using the directory structure in your data directory to create separate *calendars*. For example, my wife, *erp*, and I, *dag*, separate personal and shared items with this structure:

```
root etm data directory
    personal
        dag
        erp
    shared
        holidays
        birthdays
        events
```

Here, our etm configuration files are located in our home directories:

```
~dag/.etm/etmtk.cfg
~erp/.etm/etmtk.cfg
```

Both contain `datadir` entries specifying the common root data directory mentioned above with these additional entries, respectively:

In `~dag/.etm/etmtk.cfg`:

```
calendars
- [dag, true, personal/dag]
- [erp, false, personal/erp]
- [shared, true, shared]
```

In `~erp/.etm/etmtk.cfg`:

```
calendars
- [erp, true, personal/erp]
- [dag, false, personal/dag]
- [shared, true, shared]
```

Thus, by default, both *dag* and *erp* see the entries from their personal files as well as the shared entries and each can optionally view the entries from the other's personal files as well. See the Preferences for details on the `calendars` entry.

Note for Windows users. The path separator needs to be "escaped" in the calendar paths, e.g., you should enter

```
- [dag, true, personal\\dag]
```

instead of

```
- [dag, true, personal\dag]
```

Data Organization

etm offers many ways of organizing your data. Perhaps, the most obvious is by *path*, i.e., the directory structure inside your data directory. *Path View* presents your data using this organization and, as noted above, calendars can be specified using this structure to allow you to choose quickly the calendars whose items will appear in other etm views as well.

The other hierarchical way of organizing your data uses the keywords you specify in your items. *Keyword View* presents your data using this organization. E.g.,

```
- my task @k A:B:C
- my other task
```

would appear in *Keyword View* as:

```
A
    B
        C
            - my task
~ none ~
    - my other task
```

There are no hard and fast rules about how to use these hierarchies but the goal is a system that makes complementary uses of path and keyword and fits your needs. As with any filing system, planning and consistency are paramount. For example, one pattern of use for a business might be to use folders for departments and people and keywords for client and project.

It is also possible to add one or more tags to items and use *Tag View* to see the resulting organization. For example

```
- item 1 @t red, white, blue
- item 2 @t red @t white
- item 3 @t white @t blue
- item 4 @t red, blue
- item 5 @t white
```

would appear in *Tag View* as

```
blue
    - item 1
    - item 3
    - item 4
red
    - item 1
    - item 2
    - item 4
white
    - item 1
    - item 2
    - item 3
    - item 5
```

A final important way of organizing your data is provided by *context*. This is designed to support a GTD (Getting Things Done) common practice where possible contexts includes things like phone, errands, email and so forth. Undated tasks such as

```
- pick up milk @c errands
- call Saul @c phone
- confirm schedule with Bob @c email
```

would appear *Agenda View* as

```
Next
    email
        - confirm schedule with Bob
    errands
        - pick up milk
    phone
        - call Saul
```

When you are next checking email, running errands, using the phone or whatever, you can check *Agenda View* to see what else might be accomplished at the same time. Note that, unlike tags, items can have at most a single context.

Colors

Versions of etm after 3.1.39 support custom settings for both foreground (font) and background colors in the GUI. If a file named `colors.cfg` is found in the etm directory on startup, then the color settings in this file will override the default color settings. If this file is not found, then it will be created and populated with the default color settings. This file can be opened for editing in etm using *File/Open/Configuration file* from the main menu.

Example files for both dark and light backgrounds are available for download and customization. You can also download `colors.py`, set your preferred background color inside this script and then run it to see how the different font colors would appear against your chosen background. See also the setting for `style` under Preferences.

Internationalization

Versions of etm after 3.1.20 provide support for languages beyond English.

End User

If you, for example, are French and would like to use a version of etm in which menu items and standard phrases are French then you need to download the file `fr_FR.mo` either from GitHub locale or from etmtk languages and copy it to the following location in your `etmdir`:

```
<your etmdir>
    languages
        fr_FR
            LC_MESSAGES
                fr_FR.mo
```

creating the corresponding directory structure when necessary. Be sure to get the file with the `.mo` extension, not the one with the `.po` extension. Next you need to create a file named `locale.cfg` in your `etmdir` with the line:

```
[[fr_FR, UTF-8], QLocale.French, QLocale.France]
```

perhaps modifying `UTF-8` to reflect your actual file encoding.

That's it! When you next start etm, `locale.cfg` will be read, `fr_FR` will be set as the desired locale and, if it can be found in the specified directory, the translations in `fr_FR.mo` will be loaded. Now, e.g, instead of *Agenda* you will see *Ordre du jour*.

Translator

If you would like to assist in providing etm for a particular language, the process is pretty simple. You will need to download the program poedit. A free version is available for all major platforms.

In the etm source code, whenever a word or phrase appears that will be seen by the user, it is wrapped in a special format using `_()` so that, e.g., `Agenda` appears in the source code as `_("Agenda")`, `Today` as `_("Today")` and so forth.

When etm is being prepared for distribution a program called *gettext* is used to extract the `_()` entries from wherever they appear in the source and copy them to specially formatted file called `etm.pot`. This file can be then be used in the open-source program *poedit* to create a special translation files for different languages. This is how `fr_FR.po` was created, for example. Translation files are available from the above sources for French (`fr_FR.po`), German (`de_DE.po`), Spanish (`es_ES.po`) and Polish (`pl_PL.mo`). Alternatively, `etm.pot` is also available and you can use it to create whatever translation files you wish.

When a `.po` translation file is opened in *poedit*, two columns are displayed, the first lists the `_()` entries from the source code and the second lists the corresponding translation. E.g.,

```
Agenda          Ordre du jour
Today           Aujourd'hui
```

Initially, of course, the translation column is empty and it is the job of the translator to provide the translations. The *pro* version of *poedit* (~ $20) provides a third column with likely guesses about the appropriate translation. Most of the choices in `fr_FR.po`, in fact, came from accepting these best guesses since my knowledge of French is miniscule.

Whenever a `.po` file is saved in *poedit*, a compiled version with the extension `.mo` is automatically created. This compiled version is the one actually used by etm and the only file that an end user needs.

When an end user has followed the steps given above to enable support for a particular language, the actual translations will, of course be limited to those with "second column choices". Extending the example above, suppose the translator has omitted some items

```
Agenda          Ordre du jour
Yesterday
Today           Aujourd'hui
Tomorrow
```

Then whenever `_("Agenda")` appears in the source, it will effectively be replaced by `"Ordre du jour"` and whenever `_("Yesterday")` appears, it will be replaced by `"Yesterday"`. I.e., when a translation is available, it will be used; otherwise, the original text will be used.

A translator can thus do as much or as little as he or she pleases and then send me the resulting `.po` file. I'll replace the current on-line version with this updated version so the next translator can improve upon prior results.

Item types

There are several types of items in etm. Each item begins with a type character such as an asterisk (event) and continues on one or more lines either until the end of the file is reached or another line is found that begins with a type character. The type character for each item is followed by the item summary and then, perhaps, by one or more `@key value` pairs - see @-Keys for details. The order in which such pairs are entered does not matter.

~ Action

A record of the expenditure of time (`@e`) and/or money (`@x`). Actions are not reminders, they are instead records of how time and/or money was actually spent. Action lines begin with a tilde, `~`.

```
~ picked up lumber and paint @s mon 3p @e 1h15m @x 127.32
```

Entries such as `@s mon 3p`, `@e 1h15m` and `@x 127.32` are discussed below under *Item details*. Action entries form the basis for time and expense billing using action type custom views - see Custom view for details.

Tip: You can use either path or keyword or a combination of the two to organize your actions.

* Event

Something that will happen on particular day(s) and time(s). Event lines begin with an asterick, `*`.

```
* dinner with Karen and Al @s sat 7p @e 3h
```

Events have a starting datetime, `@s` and an extent, `@e`. The ending datetime is given implicitly as the sum of the starting datetime and the extent. Events that span more than one day are possible, e.g.,

```
* Sales conference @s 9a wed @e 2d8h
```

begins at 9am on Wednesday and ends at 5pm on Friday.

An event without an `@e` entry or with `@e 0` is regarded as a *reminder* and, since there is no extent, will not be displayed in *busy times*.

^ Occasion

Holidays, anniversaries, birthdays and such. Similar to an event with a date but no starting time and no extent. Occasions begin with a caret sign, `^`.

```
^ The !1776! Independence Day @s 2010-07-04 @r y &M 7 &m 4
```

On July 4, 2013, this would appear as `The 237th Independence Day`. Here !1776¡ is an example of an *anniversary substitution* - see Dates for details.

! Note

A record of some useful information. Note lines begin with an exclamation point, `!`.

```
! xyz software @k software:passwords @d user: dnlg, pw: abc123def
```

Tip: Since both the GUI and CLI note views group and sort by keyword, it is a good idea to use keywords to organize your notes.

-, % and + Tasks

Tasks are reminders of something that needs to be done. There are three possible type characters for tasks: `-`, `%` and `+`; these are discussed below. Each of these can be further distinguished by whether or not the task has entries for `@e` and/or `@s`.

When an `@e` (extent) is provided for any type of task, it is regarded as an estimate of the time required to complete the task.

- Tasks without an `@s` entry have no due date and are to be done whenever convenient.
- Tasks with an `@s` entry that specifies `12am` (or `0h`) as the starting time are to be completed on or before the date specified.
- Tasks with an `@s` entry that specifies a starting time *other than* `12am` (or `0h`) but without an `@e` entry are to be completed on or before the date *and time* specified. These will be displayed in etm Agenda and Day views before other tasks, sorted by and displaying the starting time.
- Tasks with an `@s` entry that specifies a starting time *other than* `12am` (or `0h`) and with an `@e` entry are to be completed during the period that extends from the starting time until `@e` after the starting time. These will be displayed in etm Agenda and Day views before other tasks, sorted by and displaying the time period in the same manner as events. This period will be regarded as busy time and treated as such by etm in, e.g., Week view. [Tip: Use a non-midnight starting time and an extent when you want to block off a specific period to complete a task.]

- Task

This is the basic task and begins with a minus sign, `-`.

```
- pay bills @s Oct 25
```

A task with an `@s` entry becomes due on that date and time and past due when that date has passed. If the task also has an `@b` begin-by entry, then advance warnings of the task will begin appearing the specified number of days before the task is due.

% Delegated task

A task that is assigned to someone else, usually the person designated in an `@u` entry. Delegated tasks begin with a percent sign, `%`.

```
% make reservations for trip @u joe @s fri
```

+ Task group

A collection of related tasks, some of which may be prerequisite for others. Task groups begin with a plus sign, `+`.

```
+ dog house
  @j pickup lumber and paint      &q 1
  @j cut pieces                   &q 2
  @j assemble                     &q 3
  @j paint                        &q 4
```

Note that a task group is a single item and is treated as such. E.g., if any job is selected for editing then the entire group is displayed.

Individual jobs are given by the `@j` entries. The *queue* entries, `&q`, set the order — tasks with smaller &q values are prerequisites for subsequent tasks with larger &q values. In the example above "pickup lumber and paint" does not have any prerequisites. "Pickup lumber and paint", however, is a prerequisite for "cut pieces" which, in turn, is a prerequisite for "assemble". "Assemble", "cut pieces" and "pickup lumber and paint" are all prerequisites for "paint".

$ In basket

A quick, don't worry about the details item to be edited later when you have the time. In basket entries begin with a dollar sign, `$`.

```
$ joe 919 123-4567
```

If you create an item using *etm* and forget to provide a type character, an `$` will automatically be inserted.

? Someday maybe

Something you don't want to forget about altogether but don't want to appear on your next or scheduled lists. Someday maybe items begin with a question mark, `?`. They are displayed under the heading *Someday* in Agenda view so that you can easily review them whenever you like.

```
? lose weight and exercise more
```

Comment

Comments begin with a hash mark, `#`. Such items are ignored by etm save for appearing in the path view. Stick a hash mark in front of any item that you don't want to delete but don't want to see in your other views.

= Defaults

Default entries begin with an equal sign, `=`. These entries consist of `@key value` pairs which then become the defaults for subsequent entries in the same file until another `=` entry is reached.

Suppose, for example, that a particular file contains items relating to "project_a" for "client_1". Then entering

```
= @k client_1:project_a
```

on the first line of the file and

```
=
```

on the twentieth line of the file would set the default keyword for entries between the first and twentieth line in the file.

@Keys

@a alert

The specification of the alert(s) to use with the item. One or more alerts can be specified in an item. E.g.,

```
@a 10m, 5m
@a 1h: s
```

would trigger the alert(s) specified by `default_alert` in your `etmtk.cfg` at 10 and 5 minutes before the starting time and a (s)ound alert one hour before the starting time.

The alert

```
@a 2d: e; who@what.com, where2@when.org; filepath1, filepath2
```

would send an email to the two listed recipients exactly 2 days (48 hours) before the starting time of the item with the item summary as the subject, with file1 and file2 as attachments and with the body of the message composed using `email_template` from your `etmtk.cfg`.

Similarly, the alert

```
@a 10m: t; 9191234567@vtext.com, 9197654321@txt.att.net
```

would send a text message 10 minutes before the starting time of the item to the two mobile phones listed (using 10 digit area code and carrier mms extension) together with the settings for `sms` in `etmtk.cfg`. If no numbers are given, the number and mms extension specified in `sms.phone` will be used. Here are the mms extensions for the major US carriers:

```
Alltel          @message.alltel.com
AT&T            @txt.att.net
Nextel          @messaging.nextel.com
Sprint          @messaging.sprintpcs.com
SunCom          @tms.suncom.com
T-mobile        @tmomail.net
VoiceStream     @voicestream.net
Verizon         @vtext.com
```

Finally,

```
@a 0: p; program_path
```

would execute `program_path` at the starting time of the item.

The format for each of these:

```
@a <trigger times> [: action [; arguments]]
```

In addition to the default action used when the optional `: action` is not given, there are the following possible values for `action`:

```
d   Execute alert_displaycmd in etmtk.cfg.

e; recipients[;attachments]     Send an email to recipients
   (a comma separated list of email addresses) optionally
   attaching attachments (a comma separated list of file paths).
   The item summary is used as the subject of the email and
   the expanded value of email_template from etmtk.cfg as the
   body. If there is an entry for @i (invitees), these email
   addresses will be appended to the list of recipients.

m   Display an internal etm message box using alert_template.

p; process      Execute the command given by process.
```

```
s   Execute alert_soundcmd in etmtk.cfg.

t [; phonenumbers]      Send text messages to phonenumbers
  (a comma separated list of 10 digit phone numbers with the
  sms extension of the carrier appended) with the expanded
  value of sms.message as the text message.

v   Execute alert_voicecmd in etmtk.cfg.
```

Note: either `e` or `p` can be combined with other actions in a single alert but not with one another.

@b beginby

An integer number of days before the starting date time at which to begin displaying *begin by* notices. When notices are displayed they will be sorted by the item's starting datetime and then by the item's priority, if any.

@c context

Intended primarily for tasks to indicate the context in which the task can be completed. Common contexts include home, office, phone, computer and errands. The "next view" supports this usage by showing undated tasks, grouped by context. If you're about to run errands, for example, you can open the "next view", look under "errands" and be sure that you will have no "wish I had remembered" regrets.

@d description

An elaboration of the details of the item to complement the summary.

@e extent

A time period string such as `1d2h` (1 day 2 hours). For an action, this would be the elapsed time. For a task, this could be an estimate of the time required for completion. For an event, this would be the duration. The ending time of the event would be this much later than the starting datetime.

Tip. Need to determine the appropriate value for `@e` for a flight when you have the departure and arrival datetimes but the timezones are different? The date calculator (shortcut Shift-D) will accept timezone information so that, e.g., entering the arrival time minus the departure time

```
4/20 6:15p US/Central - 4/20 4:50p Asia/Shanghai
```

into the calculator would give

```
14h25m
```

as the flight time.

@f done[; due]

Datetimes; tasks, delegated tasks and task groups only. When a task is completed an `@f done` entry is added to the task. When the task has a due date, `; due` is appended to the entry. Similarly, when a job from a task group is completed in etm, an `&f done` or `&f done; due` entry is appended to the job and it is removed from the list of prerequisites for the other jobs. In both cases `done` is the completion datetime and `due`, if added, is the datetime that the task or job was due. The completed task or job is shown as finished on the completion date. When the last job in a task group is finished an `@f done` or `@f done; due` entry is added to the task group itself reflecting the datetime that the last job was done and, if the task group is repeating, the `&f` entries are removed from the individual jobs.

Another step is taken for repeating task groups. When the first job in a task group is completed, the `@s` entry is updated using the setting for `@o` (above) to show the next datetime the task group is due and the `@f` entry is removed from the task group. This means when some, but not all of the jobs for the current repetition have been completed, only these job completions will be displayed. Otherwise, when none of the jobs for the current repetition have been completed, then only that last completion of the task group itself will be displayed.

Consider, for example, the following repeating task group which repeats monthly on the last weekday on or before the 25th.

```
+ pay bills @s 11/23 @f 10/24;10/25
  @r m &w MO,TU,WE,TH,FR &m 23,24,25 &s -1
  @j organize bills &q 1
  @j pay on-line bills &q 3
  @j get stamps, envelopes, checkbook &q 1
  @j write checks &q 2
  @j mail checks &q 3
```

Here “organize bills” and “get stamps, envelopes, checkbook” have no prerequisites. “Organize bills”, however, is a prerequisite for “pay on-line bills” and both “organize bills” and “get stamps, envelops, checkbook” are prerequisites for “write checks” which, in turn, is a prerequisite for “mail checks”.

The repetition that was due on 10/25 was completed on 10/24. The next repetition was due on 11/23 and, since none of the jobs for this repetition have been completed, the completion of the group on 10/24 and the list of jobs due on 11/23 will be displayed initially. The following sequence of screen shots show the effect of completing the jobs for the 11/23 repetition one by one on 11/27.

@g goto

The path to a file or a URL to be opened using the system default application when the user presses *G* in the GUI. E.g., here's a task to join the etm discussion group with the URL of the group as the link. In this case, pressing *G* would open the URL in your default browser.

```
- join the etm discussion group @s +1/1
  @g http://groups.google.com/group/eventandtaskmanager/topics
```

Template expansion is supported so it is also possible to use a `mailto` link such as the following:

```
- the subject of the email @d The body of the email
  @g mailto:sam@what.com?cc=joe@when.net\&subject=!summary!\&body=!d!
```

Pressing *G* with this item selected would create a new message in your email application with "To: sam@what.com", "Cc: joe@when.net", "Subject: The subject of the email" and "The body of the email" already entered.

Tip. Have a pdf file with the agenda for a meeting? Stick an @g entry with the path to the file in the event you create for the meeting. Then whenever the meeting is selected, *G* will bring up the agenda.

@h history

Used internally with task groups to track completion done;due pairs.

@i invitees

An email address or a list of email addresses for people participating in the item. These email addresses will be appended to the list of recipients for email alerts.

@j job

Component tasks or jobs within a task group are given by `@j job` entries. `@key value` entries prior to the first `@j` become the defaults for the jobs that follow. `&key value` entries given in jobs use `&` rather than `@` and apply only to the specific job.

Many key-value pairs can be given either in the group task using `@` or in the component jobs using `&`:

```
@c or &c    context
@d or &d    description
@e or &e    extent
@f or &f    done[; due] datetime
@k or &k    keyword
@l or &l    location
@u or &u    user
```

The key-value pair `&h` is used internally to track job done;due completions in task groups.

The key-value pair `&q` (queue position) can *only* be given in component jobs where it is required. Key-values other than `&q` and those listed above, can *only* be given in the initial group task entry and their values are inherited by the component jobs.

@k keyword

A heirarchical classifier for the item. Intended for actions to support time billing where a common format would be `client:job:category`. *etm* treats such a keyword as a heirarchy so that an action report grouped by month and then keyword might appear as follows

```
27.5h) Client 1 (3)
    4.9h) Project A (1)
    15h) Project B (1)
    7.6h) Project C (1)
24.2h) Client 2 (3)
    3.1h) Project D (1)
    21.1h) Project E (2)
        5.1h) Category a (1)
        16h) Category b (1)
4.2h) Client 3 (1)
8.7h) Client 4 (2)
    2.1h) Project F (1)
    6.6h) Project G (1)
```

An arbitrary number of heirarchical levels in keywords is supported.

@l location

The location at which, for example, an event will take place.

@m memo

Further information about the item not included in the summary or the description. Since the summary is used as the subject of an email alert and the description is commonly included in the body of an email alert, this field could be used for information not to be included in the email.

@n noshow

Only tasks of type "-" or "%". A value or list of values from *d*, *k*, and *t*, that specify views in which the task should *not* be shown.

This can provide a "poor man's cron" in which a repeating task with a process alert could be used to run a process at specified times without cluttering the etm views unnecessarily. It could also be used to trigger a periodic reminder during the day to, e.g., take a prescription medication, without filling your day lists.

Tip. Want to be reminded when a meeting should end without seeing an extra reminder for the meeting in your day lists? Create a task with a sound alert at the ending time and then add "@n d" to hide it from your day lists and, optionally, "@o s" to automatically remove past due instances.

d) day Do not display the task in the day views: agenda, week and month.

Since an undated task would not appear in week or month view, using "d" for such a task only prevents it from being displayed in the "next" section of agenda view. Using "d" for a dated task, on the other hand, prevents it from being displayed in the day lists of agenda, week and month views as well as the "now" section of agenda view.

k) keyword Do not display the task in keyword view.

t) tag Do not display the task in tag view.

E.g., with the entry "@n d, k, t", a task would appear only in path view.

@o overdue

Repeating tasks only. One of the following choices: k) keep, r) restart, or s) skip. Details below.

@p priority

Either 0 (no priority) or an integer between 1 (highest priority) and 9 (lowest priority). Primarily used with undated tasks.

@q datetime

Used to provide a timestamp for an item. Intended primarily to provide a first-in-first-out queue for related, undated tasks. E.g., the following

```
- first in queue @c queue @q 2015-10-06 10a @z US/Eastern
- second in queue @c queue @q 2015-10-07 12p @z US/Eastern
- third in queue @c queue @q 2015-10-08 9a @z US/Eastern
- fourth in queue @c queue @q 2015-10-09 8a @z US/Eastern
```

would appear in Agenda view at 11:35am on 2015-10-09 grouped by the context "queue" and ordered by age with the oldest first:

```
Next
   ...
   queue
       - first in queue                          3d2h
       - second in queue                        1d23h35m
       - third in queue                          1d2h35m
       - fourth in queue                          3h35m
```

@r repetition rule

The specification of how an item is to repeat. Repeating items **must** have an `@s` entry as well as one or more `@r` entries. Generated datetimes are those satisfying any of the `@r` entries and falling **on or after** the datetime given in `@s`. Note that the datetime given in `@s` will only be included if it matches one of the datetimes generated by the `@r` entry.

A repetition rule begins with

```
@r frequency
```

where `frequency` is one of the following characters:

```
y       yearly
m       monthly
w       weekly
d       daily
h       hourly
n       minutely
l       list (a list of datetimes will be provided using @+)
```

The `@r frequency` entry can, optionally, be followed by one or more `&key value` pairs:

```
&i: interval (positive integer, default = 1) E.g, with frequency w, interval
    3 would repeat every three weeks.
&t: total (positive integer) Include no more than this number of repetitions.
&s: bysetpos (integer). When multiple dates satisfy the rule, take the date
    from this position in the list, e.g, &s 1 would choose the first element
    and &s -1 the last. See the payday example below for an illustration of
    bysetpos.
&u: until  (datetime) Only include repetitions with starting times falling
    on before this datetime.
&M: bymonth (1, 2, ..., 12)
&m: bymonthday (1, 2, ..., 31) Use, e.g., -1 for the last day of the month.
&W: byweekno (1, 2, ..., 53)
&w: byweekday (*English* weekday abbreviation SU ... SA). Use, e.g., 3WE
    for the 3rd Wednesday or -1FR, for the last Friday in the month.
&h: byhour (0 ... 23)
&n: byminute (0 ... 59)
&E: byeaster (integer number of days before, < 0, or after, > 0, Easter)
```

Repetition examples:

- 1st and 3rd Wednesdays of each month.

  ```
  ^ 1st and 3rd Wednesdays
    @r m &w 1WE, 3WE
  ```

- Payday (an occasion) on the last week day of each month. (The `&s -1` entry extracts the last date which is both a weekday and falls within the last three days of the month.)

  ```
  ^ payday @s 2010-07-01
    @r m &w MO, TU, WE, TH, FR &m -1, -2, -3 &s -1
  ```

- Take a prescribed medication daily (an event) from the 23rd through the 27th of the current month at 10am, 2pm, 6pm and 10pm and trigger an alert zero minutes before each event.

  ```
  * take Rx @d 10a 23  @r d &u 11p 27 &h 10, 14 18, 22 @a 0
  ```

- Vote for president (an occasion) every four years on the first Tuesday after a Monday in November. (The `&m range(2,9)` requires the month day to fall within 2 ... 8 and thus, combined with `&w TU` to be the first Tuesday following a Monday.)

  ```
  ^ Vote for president @s 2012-11-06
    @r y &i 4 &M 11 &m range(2,9) &w TU
  ```

- Ash Wednesday (an occasion) that occurs 46 days before Easter each year.

  ^ Ash Wednesday 2010-01-01 @r y &E -46

- Easter Sunday (an occasion).

  ^ Easter Sunday 2010-01-01 @r y &E 0

A repeating *task* may optionally also include an `@o <k|s|r>` entry (default = k):

- `@o k`: Keep the current due date if it becomes overdue and use the next due date from the recurrence rule if it is finished early. This would be appropriate, for example, for the task 'file tax return'. The return due April 15, 2009 must still be filed even if it is overdue and the 2010 return won't be due until April 15, 2010 even if the 2009 return is finished early.
- `@o s`: Skip overdue due dates and set the due date for the next repetition to the first due date from the recurrence rule on or after the current date. This would be appropriate, for example, for the task 'put out the trash' since there is no point in putting it out on Tuesday if it's picked up on Mondays. You might just as well wait until the next Monday to put it out. There's also no point in being reminded until the next Monday.
- `@o r`: Restart the repetitions based on the last completion date. Suppose you want to mow the grass once every ten days and that when you mowed yesterday, you were already nine days past due. Then you want the next due date to be ten days from yesterday and not today. Similarly, if you were one day early when you mowed yesterday, then you would want the next due date to be ten days from yesterday and not ten days from today.

@s starting datetime

When an action is started, an event begins or a task is due.

@t tags

A tag or list of tags for the item.

@u user

Intended to specify the person to whom a delegated task is assigned. Could also be used in actions to indicate the person performing the action.

@v action_rates key

Actions only. A key from `action_rates` in your `etmtk.cfg` to apply to the value of `@e`. Used in actions to apply a billing rate to time spent in an action. E.g., with

```
minutes: 6
action_rates:
    br1: 45.0
    br2: 60.0
```

then entries of `@v br1` and `@e 2h25m` in an action would entail a value of `45.0 * 2.5 = 112.50`.

@w action_markups key

A key from `action_markups` in your `etmtk.cfg` to apply to the value of `@x`. Used in actions to apply a markup rate to expense in an action. E.g., with

```
weights:
    mr1: 1.5
    mr2: 10.0
```

then entries of `@w mr1` and `@x 27.50` in an action would entail a value of `27.50 * 1.5 = 41.25`.

@x expense

Actions only. A currency amount such as `27.50`. Used in conjunction with @w above to bill for action expenditures.

@z time zone

The time zone of the item, e.g., US/Eastern. The starting and other datetimes in the item will be interpreted as belonging to this time zone.

Tip. You live in the US/Eastern time zone but a flight that departs Sydney on April 20 at 9pm bound for New York with a flight duration of 14 hours and 30 minutes. The hard way is to convert this to US/Eastern time and enter the flight using that time zone. The easy way is to use Australia/Sydney and skip the conversion:

```
* Sydney to New York @s 2014-04-23 9pm @e 14h30m @z Australia/Sydney
```

This flight will be displayed while you're in the Australia/Sydney time zone as extending from 9pm on April 23 until 11:30am on April 24, but in the US/Eastern time zone it will be displayed as extending from 7am until 9:30pm on April 23.

@+ include

A datetime, e.g., `@+ 20150420T0930`, or list of datetimes to be added to the repetitions generated by `@r rrule` entries. If only a date is provided, 12:00am is assumed.

@- exclude

A datetime or list of datetimes to be removed from the repetitions generated by `@r rrule` entries. If only a date is provided, 12:00am is assumed.

Note that to exclude a datetime from the recurrence rule, the @- datetime *must exactly match both the date and time* generated by one of the `@r rrule` entries.

Example of using `@-` and `@+`:

```
@s 2014-02-19 4pm
@r m &w 3WE
@+ 20140924T1600, 20141029T1600
@- 20140917T1600, 20141015T1600
```

Dates

Fuzzy dates

When either a *datetime* or an *time period* is to be entered, special formats are used in *etm*. Examples include entering a starting datetime for an item using `@s` and jumping to a date using Ctrl-J.

Suppose, for example, that it is currently 8:30am on Friday, February 15, 2013. Then, *fuzzy dates* would expand into the values illustrated below.

```
mon 2p or mon 14h    2:00pm Monday, February 19
fri                  12:00am Friday, February 15
9a -1/1 or 9h -1/1   9:00am Tuesday, January 1
+2/15                12:00am Monday, April 15 2013
8p +7 or 20h +7      8:00pm Friday, February 22
-14                  8:30am Friday, February 1
now                  8:30am Friday, February 15
```

Note that expressions using `+` or `-` give datetimes relative to the current datetime.

12am is the default time when a time is not explicity entered. E.g., `+2/15` in the examples above gives 12:00am on April 15.

To avoid ambiguity, always append either 'a', 'p' or 'h' when entering an hourly time, e.g., use `1p` or `13h`.

Time periods

Time periods are entered using the format `WwDdHhMm` where W, D, H and M are integers and w, d, h and m refer to weeks, days, hours and minutes respectively. For example:

```
2h30m                2 hours, 30 minutes
2w3d                 2 weeks, 3 days
45m                  45 minutes
```

As an example, if it is currently 8:50am on Friday February 15, 2013, then entering `now + 2d4h30m` into the date calculator would give `2013-02-17 1:20pm`.

Tip. Need to schedule a reminder in 15 minutes? Use `@s +15m`.

Time zones

Dates and times are always stored in *etm* data files as times in the time zone given by the entry for `@z`. On the other hand, dates and times are always displayed in *etm* using the local time zone of the system.

For example, if it is currently 8:50am EST on Friday February 15, 2013, and an item is saved on a system in the `US/Eastern` time zone containing the entry

```
@s now @z Australia/Sydney
```

then the data file would contain

```
@s 2013-02-16 12:50am @z Australia/Sydney
```

but this item would be displayed as starting at `8:50am 2013-02-15` on the system in the `US/Eastern` time zone.

Tip. Need to determine the flight time when the departing timezone is different that the arriving timezone? The date calculator (shortcut Shift-D) will accept timezone information so that, e.g., entering the arrival time minus the departure time

```
4/20 6:15p US/Central - 4/20 4:50p Asia/Shanghai
```

into the calculator would give

```
14h25m
```

as the flight time.

Anniversary substitutions

An anniversary substitution is an expression of the form `!YYYY!` that appears in an item summary. Consider, for example, the occassion

```
^ !2010! anniversary @s 2011-02-20 @r y
```

This would appear on Feb 20 of 2011, 2012, 2013 and 2014, respectively, as *1st anniversary*, *2nd anniversary*, *3rd anniversary* and *4th anniversary*. The suffixes, *st*, *nd* and so forth, depend upon the translation file for the locale.

Easter

An expression of the form `easter(yyyy)` can be used as a date specification in `@s` entries and in the datetime calculator. E.g.

```
@s easter(2014) 4p
```

would expand to `2014-04-20 4pm`. Similarly, in the date calculator

```
easter(2014) - 48d
```

(Rose Monday) would return `2014-03-03`. In repeating items `easter(yyyy)` is replaced by `&E`, e.g.,

```
^ Easter Sunday @s 2010-01-01 @r y &E 0
^ Ash Wednesday @s 2010-01-01 @r y &E -46
^ Rose Monday @s 2010-01-01 @r y &E -48
```

Preferences

Configuration options are stored in a file named `etmtk.cfg` which, by default, belongs to the folder `.etm` in your home directory. When this file is edited in *etm* (Shift Ctrl-P), your changes become effective as soon as they are saved — you do not need to restart *etm*. These options are listed below with illustrative entries and brief descriptions.

Template expansions

The following template expansions can be used in `alert_displaycmd`, `alert_template`, `alert_voicecmd`, `email_template`, `sms_message` and `sms_subject` below.

- `!summary!`

the item's summary (this will be used as the subject of email and message alerts)

- `!start_date!`

the starting date of the event

- `!start_time!`

the starting time of the event

- `!time_span!`

the time span of the event (see below)

- `!alert_time!`

the time the alert is triggered

- `!time_left!`

the time remaining until the event starts

- `!when!`

the time remaining until the event starts as a sentence (see below)

- `!next!`

how long before the starting time the next alert will be triggered

- `!next_alert!`

how long before the starting time the next alert will be triggered as a sentence (see below)

- `!d!`

the item's `@d` (description)

- `!l!`

the item's `@l` (location)

The value of `!next!` for some illustrative cases:

- The current alert is the last

  !next!: None

  !next_alert!: 'This is the last alert.'

- The next alert is at the start time

  !next!: 'at the start time'

  !next_alert!: 'The next alert is at the start time.'

- The next alert is 5 minutes before the start time:

  !next!: '5 minutes before the start time'

  !next_alert!: 'The next alert is 5 minutes before the start time.'

The value of `!time_span!` depends on the starting and ending datetimes. Here are some examples:

- if the start and end *datetimes* are the same (zero extent): `10am Wed, Aug 4`
- else if the times are different but the *dates* are the same: `10am - 2pm Wed, Aug 4`
- else if the dates are different: `10am Wed, Aug 4 - 9am Thu, Aug 5`
- additionally, the year is appended if a date falls outside the current year:

  ```
  10am - 2pm Thu, Jan 3 2013
  10am Mon, Dec 31 - 2pm Thu, Jan 3 2013
  ```

Here are values of `!time_left!` and `!when!` for some illustrative periods:

- `2d3h15m`

  ```
  time_left : '2 days 3 hours 15 minutes'
  when      : '2 days 3 hours 15 minutes from now'
  ```

- `20m`

  ```
  time_left : '20 minutes'
  when      : '20 minutes from now'
  ```

- `0m`

  ```
  time_left : ''
  when      : 'now'
  ```

Note that 'now', 'from now', 'days', 'day', 'hours' and so forth are determined by the translation file in use.

Options

action_interval

```
action_interval: 1
```

Every `action_interval` minutes, execute `action_timercmd` when the timer is running and `action_pausecmd` when the timer is paused. Choose zero to disable executing these commands.

action_keys

```
action_keys: "k"
```

When klone is used to create an action timer, copy the values of the @-keys in this string from the item to the timer. With the default, "k", klone will copy the item's @k entry, if there is one, in addition to the summary when creating the action. Replacing "k", with "c" would cause klone to copy the item's @c entry in addition to the summary. With "ck", both the @c and @k entries would be copied. Any key that is valid for an action can be used.

E.g., when

```
- my task @c my context @k my keyword @t my tag
```

is selected, then the default would create a timer with the name `my task @k my keyword`.

action_markups

```
action_markups:
    default: 1.0
    mu1: 1.5
    mu2: 2.0
```

Possible markup rates to use for `@x` expenses in actions. An arbitrary number of rates can be entered using whatever labels you like. These labels can then be used in actions in the `@w` field so that, e.g.,

```
... @x 25.80 @w mu1 ...
```

in an action would give this expansion in an action template:

```
!expense! = 25.80
!charge! = 38.70
```

action_minutes

```
action_minutes: 6
```

Round action times up to the nearest `action_minutes` in action custom view. Possible choices are 1, 6, 12, 15, 30 and 60. With 1, no rounding is done and times are reported as integer minutes. Otherwise, the prescribed rounding is done and times are reported as floating point hours.

action_rates

```
action_rates:
    default: 30.0
    br1: 45.0
    br2: 60.0
```

Possible billing rates to use for `@e` times in actions. An arbitrary number of rates can be entered using whatever labels you like. These labels can then be used in the `@v` field in actions so that, e.g., with `action_minutes: 6` then:

```
... @e 75m @v br1 ...
```

in an action would give these expansions in an action template:

```
!hours! = 1.3
!value! = 58.50
```

If the label `default` is used, the corresponding rate will be used when `@v` is not specified in an action.

Note that etm accumulates group totals from the `time` and `value` of individual actions. Thus

```
... @e 75m @v br1 ...
... @e 60m @v br2 ...
```

would aggregate to

```
!hours!  = 2.3     (= 1.3 + 1)
!value! = 118.50   (= 1.3 * 45.0 + 1 * 60.0)
```

action_template

```
action_template: '!hours!h) !label! (!count!)'
```

Used for action type custom view. With the above settings for `action_minutes` and `action_template`, a custom view might appear as follows:

```
27.5h) Client 1 (3)
    4.9h) Project A (1)
    15h) Project B (1)
    7.6h) Project C (1)
```

```
24.2h) Client 2 (3)
    3.1h) Project D (1)
    21.1h) Project E (2)
        5.1h) Category a (1)
        16h) Category b (1)
4.2h) Client 3 (1)
8.7h) Client 4 (2)
    2.1h) Project F (1)
    6.6h) Project G (1)
```

Available template expansions for `action_template` include:

- `!label!`: the item or group label.
- `!count!`: the number of children represented in the reported item or group.
- `!minutes!:` the total time from `@e` entries in minutes rounded up using the setting for `action_minutes`.
- `!hours!`: if action_minutes = 1, the time in hours and minutes. Otherwise, the time in floating point hours.
- `!value!`: the billing value of the rounded total time. Requires an action entry such as `@v br1` and a setting for `action_rates`.
- `!expense!`: the total expense from `@x` entries.
- `!charge!`: the billing value of the total expense. Requires an action entry such as `@w mu1` and a setting for `action_markups`.
- `!total!`: the sum of `!value!` and `!charge!`.

Note: when aggregating amounts in action type custom view, billing and markup rates are applied first to times and expenses for individual actions and the resulting amounts are then aggregated. Similarly, when times are rounded up, the rounding is done for individual actions and the results are then aggregated.

action_timer

```
action_timer:
    paused: 'play ~/.etm/sounds/timer_paused.wav'
    running: 'play ~/.etm/sounds/timer_running.wav'
```

The command `running` is executed every `action_interval` minutes whenever the action timer is running and `paused` every minute when the action timer is paused.

agenda

```
agenda_days: 2
agenda_colors: 2
agenda_indent: 2
agenda_omit: [ac, fn, ns]
agenda_width1: 43
agenda_width2: 17
```

Sets the number of days to display in agenda view and other parameters affecting the display in the CLI. The colors setting only affects output to current_html. Items in agenda_omit will not be displayed in the agenda day list. Possible choices include:

- ac: actions
- by: begin by warnings
- fn: finished tasks
- ns: notes (dated)
- oc: occasions

alert_default

```
alert_default: [m]
```

The alert or list of alerts to be used when an alert is specified for an item but the type is not given. Possible values for the list include:

- d: display (requires `alert_displaycmd`)
- m: message (using `alert_template`)
- s: sound (requires `alert_soundcmd`)
- v: voice (requires `alert_voicecmd`)

alert_displaycmd

```
alert_displaycmd: growlnotify -t !summary! -m '!time_span!'
```

The command to be executed when `d` is included in an alert. Possible template expansions are discussed at the beginning of this tab.

alert_soundcmd

```
alert_soundcmd: 'play ~/.etm/sounds/etm_alert.wav'
```

The command to execute when `s` is included in an alert. Possible template expansions are discussed at the beginning of this tab.

alert_template

```
alert_template: '!time_span!\n!l!\n\n!d!'
```

The template to use for the body of `m` (message) alerts. See the discussion of template expansions at the beginning of this tab for other possible expansion items.

alert_voicecmd

```
alert_voicecmd: say -v 'Alex' '!summary! begins !when!.'
```

The command to be executed when `v` is included in an alert. Possible expansions are are discussed at the beginning of this tab.

alert_wakecmd

```
alert_wakecmd: ~/bin/SleepDisplay -w
```

If given, this command will be issued to "wake up the display" before executing `alert_displaycmd`.

ampm

```
ampm: true
```

Use ampm times if true and twenty-four hour times if false. E.g., 2:30pm (true) or 14:30 (false).

completions_width

```
completions_width: 36
```

The width in characters of the auto completions popup window.

calendars

```
calendars:
- [dag, true, personal/dag]
- [erp, false, personal/erp]
- [shared, true, shared]
```

These are (label, default, path relative to `datadir`) tuples to be interpreted as separate calendars. Those for which default is `true` will be displayed as default calendars. E.g., with the `datadir` below, `dag` would be a default calendar and would correspond to the absolute path `/Users/dag/.etm/data/personal/dag`. With this setting, the calendar selection dialog would appear as follows:

When non-default calendars are selected, busy times in the "week view" will appear in one color for events from default calendars and in another color for events from non-default calendars.

Only data files that belong to one of the calendar directories or their subdirectories will be accessible within etm.

cfg_files

```
cfg_files:
    - completions: []
    - reports:     []
    - users:       []
```

Each of the three list brackets can contain one or more comma separated *absolute* file paths. Additionally, paths corresponding to active calendars in the `datadir` directory are searched for files named `completions.cfg`, `reports.cfg` and `users.cfg` and these are processed in addition to the ones from `cfg_files`.

Note. Windows users should place each absolute path in quotes and escape backslashes, i.e., use \\ anywhere \ appears in a path.

- Completions

  Each line in a completions file provides a possible completion when using the editor. E.g. with these completions

  ```
  @c computer
  @c home
  @c errands
  @c office
  @c phone
  @z US/Eastern
  ```

```
@z US/Central
@z US/Mountain
@z US/Pacific
dnlgrhm@gmail.com
```

entering, for example, "@c" in the editor and pressing Ctrl-Space, would popup a list of possible completions. Choosing the one you want and pressing *Return* would insert it and close the popup.

Up and down arrow keys change the selection and either *Tab* or *Return* inserts the selection.

- Reports

  Each line in a reports file provides a possible reports specification. These are available when using the CLI `m` command and in the GUI custom view. See Custom view for details.

- Users

  User files contain user (contact) information in a free form, text database. Each entry begins with a unique key for the person and is followed by detail lines each of which begins with a minus sign and contains some detail about the person that you want to record. Any detail line containing a colon should be quoted, e.g.,

```
jbrown:
- Brown, Joe
- jbr@whatever.com
- 'home: 123 456-7890'
- 'birthday: 1978-12-14'
dcharles:
- Charles, Debbie
- dch@sometime.com
- 'cell: 456 789-0123'
- 'spouse: Rebecca'
```

  Keys from this file are added to auto-completions so that if you type, say, `@u jb` and press *Ctrl-Space*, then `@u jbrown` would be offered for completion.

  If an item with the entry `@u jbrown` is selected in the GUI, you can press "u" to see a popup with the details:

```
Brown, Joe
jbr@whatever.com
home: 123 456-7890
birthday: 1978-12-14
```

countdown timer

```
countdown_command: ''
countdown_minutes: 10
```

If `countdown_command` is given, it will be executed when the timer expires; otherwise a beep will be sounded. The default number of minutes for a countdown is given by `countdown_minutes`. When a timer is active, the time that the timer will expire is displayed in the status bar using the format -H:M:S(am/pm). When a countdown and a snooze timer are both active, the one that will expire first is displayed in the status bar.

current files

```
current_htmlfile:  ''
current_textfile:  ''
current_icsfolder:  ''
current_indent:    3
current_opts:      ''
current_width1:    40
current_width2:    17
```

If absolute file paths are entered for `current_textfile` and/or `current_htmlfile`, then these files will be created and automatically updated by etm as as plain text or html files, respectively. If `current_opts` is given then the file will contain a report using these options; otherwise the file will contain an agenda. Indent and widths are taken from these setting with other settings, including color, from *report* or *agenda*, respectively.

If an absolute path is entered for `current_icsfolder`, then ics files corresponding to the entries in `calendars` will be created in this folder and updated as necessary. If there are no entries in calendars, then a single file, `all.ics`, will be created in this folder and updated as necessary.

Hint: fans of geektool can use the shell command `cat <current_textfile>` to have the current agenda displayed on their desktops.

datadir

```
datadir: ~/.etm/data
```

All etm data files are in this directory.

dayfirst

```
dayfirst: false
```

If dayfirst is False, the MM-DD-YYYY format will have precedence over DD-MM-YYYY in an ambiguous date. See also `yearfirst`.

details_rows

```
details_rows: 4
```

The number of rows to display in the bottom, details panel of the main window.

display_idletime

```
display_idletime: True
```

Show idle time in the status bar by default if True. Display can be toggled on and off in the File/Timer menu. Idle time is accumulated when there are are one or more active timers and none are running.

early_hour

```
early_hour: 6
```

When scheduling an event or action with a starting time that begins before this hour, append the query "Is ___ the starting time you intended?" to the confirmation. Use 0 to disable this warning altogether. The default, 6, will warn for starting times *before* 6am.

edit_cmd

```
edit_cmd: ~/bin/vim !file! +!line!
```

This command is used in the command line version of etm to create and edit items. When the command is expanded, `!file!` will be replaced with the complete path of the file to be edited and `!line!` with the starting line number in the file. If the editor will open a new window, be sure to include the command to wait for the file to be closed before returning, e.g., with vim:

```
edit_cmd: ~/bin/gvim -f !file! +!line!
```

or with sublime text:

```
edit_cmd: ~/bin/subl -n -w !file!:!line!
```

email_template

```
email_template: 'Time: !time_span!
Locaton: !l!


!d!'
```

Note that two newlines are required to get one empty line when the template is expanded. This template might expand as follows:

```
Time: 1pm - 2:30pm Wed, Aug 4
Location: Conference Room

<contents of @d>
```

See the discussion of template expansions at the beginning of this tab for other possible expansion items.

etmdir

```
etmdir: ~/.etm
```

Absolute path to the directory for etmtk.cfg and other etm configuration files.

exportdir

```
exportdir: ~/.etm
```

Absolute path to the directory for exported CSV files.

encoding

```
encoding: {file: utf-8, gui: utf-8, term: utf-8}
```

The encodings to be used for file IO, the GUI and terminal IO.

filechange_alert

```
filechange_alert: 'play ~/.etm/sounds/etm_alert.wav'
```

The command to be executed when etm detects an external change in any of its data files. Leave this command empty to disable the notification.

fontsize_fixed

```
fontsize_fixed: 0
```

Use this font size in the details panel, editor and reports. Use 0 to keep the system default.

fontsize_tree

```
fontsize_tree: 0
```

Use this font size in the gui treeviews. Use 0 to keep the system default.

Tip: Leave the font sizes set to 0 and run etm with logging level 2 to see the system default sizes.

freetimes

```
freetimes:
    opening:  480  # 8*60 minutes after midnight = 8am
    closing: 1020  # 17*60 minutes after midnight = 5pm
    minimum:   30  # 30 minutes
    buffer:    15  # 15 minutes
```

Only display free periods between *opening* and *closing* that last at least *minimum* minutes and preserve at least *buffer* minutes between events. Note that each of these settings must be an *interger* number of minutes.

E.g., with the above settings and these busy periods:

```
Busy periods in Week 16: Apr 14 - 20, 2014
----------------------------------------
Mon 14: 10:30am-11:00am; 12:00pm-1:00pm; 5:00pm-6:00pm
Tue 15: 9:00am-10:00am
Wed 16: 8:30am-9:30am; 2:00pm-3:00pm; 5:00pm-6:00pm
Thu 17: 11:00am-12:00pm; 6:00pm-7:00pm; 7:00pm-9:00pm
Fri 18: 3:00pm-4:00pm; 5:00pm-6:00pm
Sat 19: 9:00am-10:30am; 7:30pm-10:00pm
```

This would be the corresponding list of free periods:

```
Free periods in Week 16: Apr 14 - 20, 2014
----------------------------------------
Mon 14: 8:00am-10:15am; 11:15am-11:45am; 1:15pm-4:45pm
```

```
Tue 15: 8:00am-8:45am; 10:15am-5:00pm
Wed 16: 9:45am-1:45pm; 3:15pm-4:45pm
Thu 17: 8:00am-10:45am; 12:15pm-5:00pm
Fri 18: 8:00am-2:45pm; 4:15pm-4:45pm
Sat 19: 8:00am-8:45am; 10:45am-5:00pm
Sun 20: 8:00am-5:00pm
----------------------------------------
Only periods of at least 30 minutes are displayed.
```

When displaying free times in week view you will be prompted for the shortest period to display using the setting for *minimum* as the default.

Tip: Need to tell someone when you're free in a given week? Jump to that week in week view, press *Ctrl-F*, set the minimum period and then copy and paste the resulting list into an email.

iCalendar settings

icscal_file If an item is not selected, pressing Shift-X in the gui will export the active calendars in iCalendar format to this file.

```
icscal_file: ~/.etm/etmcal.ics
```

icsitem_file If an item is selected, pressing Shift-X in the gui will export the selected item in iCalendar format to this file.

```
icsitem_file: ~/.etm/etmitem.ics
```

icssync_folder

```
icssync_folder: ''
```

A relative path from `etmdata` to a folder. If given, files in this folder with the extension `.txt` and `.ics` will automatically kept concurrent using export to iCalendar and import from iCalendar. I.e., if the `.txt` file is more recent than than the `.ics` then the `.txt` file will be exported to the `.ics` file. On the other hand, if the `.ics` file is more recent then it will be imported to the `.txt` file. In either case, the contents of the file to be updated will be overwritten with the new content and the last acess/modified times for both will be set to the current time.

Note that the calendar application you use to modify the `.ics` file will impose restrictions on the subsequent content of the `.txt` file. E.g., if the `.txt` file has a note entry, then this note will be exported by etm as a VJOURNAL entry to

the `.ics` file. But VJOURNAL entries are not be recognized by many (most) calendar apps. When importing this file to such an application, the note will be omitted and thus will be missing from the `.ics` file after the next export from the application. The note will then be missing from the `.txt` file as well after the next automatic update. Restricting the content to events should be safe with with any calendar application.

Additionally, if an absolute path is entered for `current_icsfolder`, then ics files corresponding to the entries in `calendars` will be created in this folder and updated as necessary. If there are no entries in calendars, then a single file, `all.ics`, will be created in this folder and updated as necessary.

ics_subscriptions

```
ics_subscriptions: []
```

A list of (URL, path) tuples for automatic updates. The URL is a calendar subscription, e.g., for a Google Calendar subscription the entry might be something like:

```
ics_subscriptions:
    - ['https://www.google.com/calendar/ical/.../basic.ics', 'personal/dag/google.txt']
```

With this entry, pressing Shift-M in the gui would import the calendar from the URL, convert it from ics to etm format and then write the result to `personal/google.txt` in the etm data directory. Note that this data file should be regarded as read-only since any changes made to it will be lost with the next subscription update.

local_timezone

```
local_timezone: US/Eastern
```

This timezone will be used as the default when a value for `@z` is not given in an item.

message_last

```
message_last: 0
```

The number of seconds to display the message alert for an item before closing it when it is the last. With 0, the message dialog will be kept open indefinitely.

message_next

```
message_next: 0
```

The number of seconds to display the message alert for an item before closing it when it is not the last alert. With 0, the message dialog will be kept open indefinitely.

monthly

```
monthly: monthly
```

Relative path from `datadir`. With the settings above and for `datadir` the suggested location for saving new items in, say, October 2012, would be the file:

```
~/.etm/data/monthly/2012/10.txt
```

The directories `monthly` and `2012` and the file `10.txt` would, if necessary, be created. The user could either accept this default or choose a different file.

outline_depth

```
outline_depth: 2
```

The default outline depth to use when opening keyword, note, path or tag view. Once any view is opened, use Ctrl-O to change the depth for that view.

prefix

```
prefix: "\n  "
prefix_uses: "rj+-tldm"
```

Apply `prefix` (whitespace only) to the @keys in `prefix_uses` when displaying and saving items. The default would cause the selected elements to begin on a newline and indented by two spaces. E.g.,

```
+ summary @s 2014-05-09 12am @z US/Eastern
  @m memo
  @j job 1 &f 20140510T1411;20140509T0000 &q 1
  @j job 2 &f 20140510T1412;20140509T0000 &q 2
  @j job 3 &q 3
  @d description
```

report

```
report_begin:           '1'
report_end:             '+1/1'
report_colors:          2
report_width1:          61
report_width2:          19
```

Report begin and end are fuzzy parsed dates specifying the default period for reports that group by dates. Each line in the file specified by `report_specifications` provides a possible specification for a report. E.g.

```
a MMM yyyy; k[0]; k[1:] -b -1/1 -e 1
a k, MMM yyyy -b -1/1 -e 1
c ddd MMM d yyyy
c f
```

In custom view these appear in the report specifications pop-up list. A specification from the list can be selected and, perhaps, modified or an entirely new specification can be entered. See Custom view for details. See also the agenda settings above.

retain_ids

```
retain_ids: false
```

If true, the unique ids that etm associates with items will be written to the data files and retained between sessions. If false, new ids will be generated for each session.

Retaining ids enables etm to avoid duplicates when importing and exporting iCalendar files.

show_finished

```
show_finished: 1
```

Show this many of the most recent completions of repeated tasks or, if 0, show all completions.

smtp

```
smtp_from: dnlgrhm@gmail.com
smtp_id: dnlgrhm
smtp_pw: **********
smtp_server: smtp.gmail.com
```

Required settings for the smtp server to be used for email alerts.

sms

```
sms_message: '!summary!'
sms_subject: '!time_span!'
sms_from: dnlgrhm@gmail.com
sms_pw:  **********
sms_phone: 0123456789@vtext.com
sms_server: smtp.gmail.com:587
```

Required settings for text messaging in alerts. Enter the 10-digit area code and number and mms extension for the mobile phone to receive the text message when no numbers are specified in the alert. The illustrated phone number is for Verizon. Here are the mms extensions for the major carriers:

```
Alltel          @message.alltel.com
AT&T            @txt.att.net
Nextel          @messaging.nextel.com
Sprint          @messaging.sprintpcs.com
SunCom          @tms.suncom.com
T-mobile        @tmomail.net
VoiceStream     @voicestream.net
Verizon         @vtext.com
```

snooze

```
snooze_command: ''
snooze_minutes: 10
```

If `snooze_command` is given, it will be executed when the timer expires; otherwise a beep will be sounded. The default number of minutes for a snooze is given by `snooze_minutes`. When a snooze timer is active, the time that the timer will expire is displayed in the status bar in the format +H:M:S(am/pm). When a countdown and a snooze timer are both active, the one that will expire first is displayed in the status bar.

style

```
style: default
```

The style to be used for Tk/Tcl widgets. Options for linux include clam, alt, default and classic. Options for OSX add aqua. Note that aqua does not support background colors for buttons and may not be suitable with darker background colors.

sundayfirst

```
sundayfirst: false
```

The setting affects only the twelve month calendar display.

update_minutes

```
update_minutes: 15
```

Update `current_html`, `current_text` and the files in `icssync_folder` when the number of minutes past the hour modulo `update_minutes` is equal to zero. I.e. with the default, the update would occur on the hour and at 15, 30 and 45 minutes past the hour. Acceptable settings are integers between 1 and 59. Note that with a setting greater than or equal to 30, the update will occur only twice each hour.

vcs_settings

```
vcs_settings:
  command: ''
  commit: ''
  dir: ''
  file: ''
  history: ''
  init: ''
  limit: ''
```

These settings are ignored unless the setting for `vcs_system` below is either `git` or `mercurial`.

Default values will be provided for these settings based on the choice of `vcs_system` below. Any of the settings that you define here will overrule the defaults.

Here, for example, are the default values of these settings for git under OS X:

```
vcs_settings:
    command: '/usr/bin/git --git-dir {repo} --work-dir {work}'
    commit: '/usr/bin/git --git-dir {repo} --work-dir {work} add */\*.txt
        && /usr/bin/git --git-dir {repo} --work-dir {work} commit -a -m "{mesg}"'
    dir: '.git'
    file: ''
    history: '/usr/bin/git -git-dir {repo} --work-dir {work} log
        --pretty=format:"- %ar: %an%n%w(70,0,4)%s" -U1  {numchanges}
            {file}'
    init: '/usr/bin/git init {work}; /usr/bin/git -git-dir {repo}
        --work-dir {work} add */\*.txt; /usr/bin/git-git-dir {repo}
            --work-dir {work} commit -a -m "{mesg}"'
    limit: '-n'
```

In these settings, `{mesg}` will be replaced with an internally generated commit message, `{numchanges}` with an expression that depends upon `limit` that determines how many changes to show and, when a file is selected, `{file}` with the corresponding path. If `~/.etm/data` is your etm datadir, the `{repo}` would be replaced with `~/.etm/data/.git` and {work} with `~/.etm/data`.

Leave these settings empty to use the defaults.

vcs_system

```
vcs_system: ''
```

This setting must be either `"` or `git` or `mercurial`.

If you specify either git or mercurial here (and have it installed on your system), then etm will automatically commit any changes you make to any of your data files. The history of these changes is available in the GUI with the show changes command (*Ctrl-H*) and you can, of course, use any git or mercurial commands in your terminal to, for example, restore a previous version of a file.

weeks_after

```
weeks_after: 52
```

In the day view, all non-repeating, dated items are shown. Additionally all repetitions of repeating items with a finite number of repetitions are shown. This includes 'list-only' repeating items and items with `&u` (until) or `&t` (total number of repetitions) entries. For repeating items with an infinite number of repetitions, those repetitions that occur within the first `weeks_after` weeks after the current week are displayed along with the first repetition after this interval. This assures that for infrequently repeating items such as voting for president, at least one repetition will be displayed.

yearfirst

`yearfirst: true`

If yearfirst is true, the YY-MM-DD format will have precedence over MM-DD-YY in an ambiguous date. See also `dayfirst`.

Custom view

You can create a custom display of your items using either the *custom view* in the GUI or the "c" command in the CLI. In both cases, you enter a specification that determines which items will be displayed and how they will be grouped and sorted.

A view *specification* begins with a *type character*, either `a` or `c`, followed by a *groupby setting* and then, perhaps, by one or more view *options*.

View type

- **a**: action

  Actions only. Expenditures of time and money recorded in *actions* with output formatted using `action_template` computations and expansions. See Preferences for further details about the role of `action_template` in formatting actions output. Note that only *actions* are included in this view and, by default, all actions in your active calendars will be included.

- **c**: composite

  Any item types, including actions, but without `action_template` computations and expansions. Note that only unfinished tasks and unfinished instances of repeating tasks will be displayed. By default, all items from your active calendars will be included.

Groupby setting

A semicolon separated list that determines how items will be grouped and sorted. Possible elements include elements from

- c: context
- f: file path
- k: keyword

- l: location
- t: tag
- u: user

and/or a *date specifications* where a *date specification* is either

- w: week number

or a combination of one or more of the following:

- yy: 2-digit year
- yyyy: 4-digit year
- MM: month: 01 - 12
- MMM: locale specific abbreviated month name: Jan - Dec
- MMMM: locale specific month name: January - December
- dd: month day: 01 - 31
- ddd: locale specific abbreviated week day: Mon - Sun
- dddd: locale specific week day: Monday - Sunday

Note that the groupby specification affects which items will be displayed. Items that are missing an element specified in `groupby` will be omitted from the output. E.g., undated tasks and notes will be omitted when a date specification is included, items without keywords will be omitted when `k` is included and so forth. The latter behavior depends upon the value of the -m MISSING option.

When a date specification is not included in the groupby setting, undated notes and tasks will be potentially included, but only those instances of dated items that correspond to the *relevant datetime* of the item of the item will be included, where the *relevant datetime* is the past due date for any past due tasks, the starting datetime for any non-repeating item and the datetime of the next instance for any repeating item.

Within groups, items are automatically sorted by date, type and time.

Groupby examples

For example, the specification `c ddd, MMM dd yyyy` would group by year, month and day together to give output such as

```
Fri, Apr 1 2011
    items for April 1
Sat, Apr 2 2011
    items for April 2
...
```

On the other hand, the specification `a w; u; k[0]; k[1:]` would group by week number, user and keywords to give output such as

```
13.1) 2014 Week 14: Mar 31 - Apr 6
   6.3) agent 1
      1.3) client 1
         1.3) project 2
            1.3) Activity (12)
      5) client 2
         4.5) project 1
            4.5) Activity (21)
         0.5) project 2
            0.5) Activity (22)
   6.8) agent 2
      2.2) client 1
         2.2) project 2
            2.2) Activity (13)
      4.6) client 2
         3.9) project 1
            3.9) Activity (23)
         0.7) project 2
            0.7) Activity (23)
```

With the heirarchial elements, file path and keyword, it is possible to use parts of the element as well as the whole. Consider, for example, the file path `A/B/C` with the components `[A, B, C]`. Then for this file path:

```
f[0] = A
f[:2] = A/B
f[2:] = C
f = A/B/C
```

Suppose that keywords have the format `client:project`. Then grouping by year and month, then client and finally project to give output such as

```
specification: a MMM yyyy; u; k[0]; k[1] -b 1 -e +1/1

13.1) Feb 2014
   6.3) agent 1
      1.3) client 1
         1.3) project 2
            1.3) Activity 12
      5) client 2
         4.5) project 1
            4.5) Activity 21
         0.5) project 2
            0.5) Activity 22
   6.8) agent 2
      2.2) client 1
         2.2) project 2
            2.2) Activity 13
      4.6) client 2
         3.9) project 1
            3.9) Activity 23
         0.7) project 2
            0.7) Activity 23
```

View Options

View options are listed below. View type `c` supports all options except `-d`. Type `a` supports all options except `-o`. These options can be used to further limit which items are displayed.

-b BEGIN_DATE

Fuzzy parsed date. When a date specification is provided, limit the display of dated items to those with datetimes falling *on or after* this datetime. Relative day and month expressions can also be used so that, for example, `-b -14` would begin 14 days before the current date and `-b -1/1` would begin on the first day of the previous month. It is also possible to add (or subtract) a time period from the fuzzy date, e.g., `-b mon + 7d` would begin with the second Monday falling on or after today. Default: None.

-c CONTEXT

Regular expression. Limit the display to items with contexts matching CONTEXT (ignoring case). Prepend an exclamation mark, i.e., use !CONTEXT rather than CONTEXT, to limit the display to items which do NOT have contexts matching CONTEXT.

-d DEPTH

CLI only. In the GUI use *View/Set outline depth*. The default, `-d 0`, includes all outline levels. Use `-d 1` to include only level 1, `-d 2` to include levels 1 and 2 and so forth. This setting applies to the CLI only. In the GUI use the command *set outline depth*.

For example, modifying the specification above by adding `-d 3` would give the following:

```
specification: a MMM yyyy; u; k[0]; k[1] -b 1 -e +1/1 -d 3

13.1) Feb 2014
   6.3) agent 1
      1.3) client 1
      5) client 2
   6.8) agent 2
      2.2) client 1
      4.6) client 2
```

-e END_DATE

Fuzzy parsed date. When a date specification is provided, limit the display of dated items to those with datetimes falling *before* this datetime. As with BEGIN_DATE relative month expressions can be used so that, for example, `-b -1/1 -e 1` would include all items from the previous month. As with `-b`, period strings can be appended, e.g., `-b mon -e mon + 7d` would include items from the week that begins with the first Monday falling on or after today. Default: None.

-f FILE

Regular expression. Limit the display to items from files whose paths match FILE (ignoring case). Prepend an exclamation mark, i.e., use !FILE rather than FILE, to limit the display to items from files whose path does NOT match FILE.

-k KEYWORD

Regular expression. Limit the display to items with contexts matching KEYWORD (ignoring case). Prepend an exclamation mark, i.e., use !KEYWORD rather than KEYWORD, to limit the display to items which do NOT have keywords matching KEYWORD.

-l LOCATION

Regular expression. Limit the display to items with a location matching LOCATION (ignoring case). Prepend an exclamation mark, i.e., use !LOCATION rather than LOCATION, to limit the display to items which do NOT have a location that matches LOCATION.

-m MISSING

Either 0 (the default) or 1. When 1 include items that would otherwise be excluded because of a *non-date*, groupby specification. E.g., `c k` would omit items without a keyword entry, but `c k -m 1` would include such items under a "None" heading. This option does not apply to date specifications, i.e., if a date specification is part of the groupby setting, then undated items will be excluded whatever the value of `-m`.

-o OMIT

String. Composite type only. Show/hide a)ctions, d)elegated tasks, e)vents, g)roup tasks, n)otes, o)ccasions, s)omeday items and/or t)asks. For example, `-o on` would show everything except occasions and notes and `-o !on` would show only occasions and notes.

-s SUMMARY

Regular expression. Limit the display to items containing SUMMARY (ignoring case) in the item summary. Prepend an exclamation mark, i.e., use !SUMMARY rather than SUMMARY, to limit the display to items which do NOT contain SUMMARY in the summary.

-S SEARCH

Regular expression. Composite type only. Limit the display to items containing SEARCH (ignoring case) anywhere in the *item* or its file path. Prepend an exclamation mark, i.e., use !SEARCH rather than SEARCH, to limit the display to items which do NOT contain SEARCH in the item or its file path.

-t TAGS

Comma separated list of case insensitive regular expressions. E.g., use

```
-t tag1, !tag2
```

or

```
-t tag1, -t !tag2
```

to display items with one or more tags that match 'tag1' but none that match 'tag2'.

-u USER

Regular expression. Limit the display to items with user matching USER (ignoring case). Prepend an exclamation mark, i.e., use !USER rather than USER, to limit the display to items which do NOT have a user that matches USER.

-w WIDTH

Non-negative integer. Truncate the output for column 1 if it exceeds this integer. Do not truncate if this integer is zero.

-W WIDTH

Non-negative integer. Truncate the output for column 2 if it exceeds this integer.

Saving view specifications

You can save view specifications in your specifications file, `~./etm/reports.cfg` by default, and then select them in the selection box at the bottom of the custom view window in the GUI or from a list in the CLI.

You can also add specifications to file in the GUI by selecting any item from the list and then replacing the content with anything you like. Press *Return* to *add* your specification temporarily to the list. *Note that the original entry will not be affected.* When you leave the custom view you will have an opportunity to save the additions you have made. If you choose a file, your additions will be inserted into the list and it will be opened for editing.

Shortcuts

Menu

`File`

```
    New
        Item                                    N
        File                                Shift-N
    Timer
        Start action timer                      T
        Finish action timer                 Shift-T
        Toggle current timer                    I
        Delete action timer                 Shift-I
        Assign idle time                     Ctrl-I
        Reset idle to zero minutes
        Toggle idle timer display
        Countdown timer                         Z
    Open
        Data file ...                       Shift-F
        Configuration file ...              Shift-C
        Preferences                         Shift-P
        Scratchpad                          Shift-S
    ----
    Quit                                     Ctrl-Q
View
    Agenda                                   Ctrl-A
    Week                                     Ctrl-W
    Month                                    Ctrl-M
    Tag                                      Ctrl-T
    Keyword                                  Ctrl-K
    Path                                     Ctrl-P
    Note                                     Ctrl-N
    Custom                                   Ctrl-C
    ----
    Set outline filter                       Ctrl-F
    Clear outline filter                Shift-Ctrl-F
    Toggle labels                            L
    Set outline depth                        O
    Toggle finished                          X
Item
    Copy                                     C
    Delete                             BackSpace
    Edit                                     E
    Edit file                           Shift-E
    Finish                                   F
    Move                                     M
    Reschedule                               R
    Schedule new                             S
    Klone as timer                           K
    Show date and time details               D
    Open link                                G
```

Show user details	U
Tools	
Home	Home
Jump to date	J

Show remaining alerts for today	A
List busy times in week/month	B
List free times in week/month	F
Date and time calculator	Shift-D
Available dates calculator	Shift-A
Yearly calendar	Shift-Y

Show outline as text	Shift-O
Print outline	P
Export to iCal	Shift-X
Update calendar subscriptions	Shift-M
History of changes	Shift-H
Custom	
Create and display selected report	Return
Export report in text format ...	Ctrl-T
Export report in csv format ...	Ctrl-X
Save changes to report specifications	Ctrl-W
Expand report list	Down
Help	
Search	
Shortcuts	?
User manual	F1
About	F2
Check for update	F3

Edit

Show completions	Ctrl-Space
Cancel	Escape
Save and Close	Ctrl-S