NAME

tt - time tracking utility that keeps track of things with git

SYNOPSIS

tt <command> [<args>]

DESCRIPTION

This utility allows you to do simple time tracking so that you can know at any time what is your (worked time / expected work time) ratio. By default, it assumes that you work from Monday to Friday, 7.5 hours a day, and that you have 25 days of vacation per year (see **init** and **plan** for changing the default values). It supports adding public holidays (with the possibility to declare half-days), declaring vacation days, days you stay home because you are sick and days you stay home because you take care of a sick child. In case you wonder how all these days off are even possible, this utility has been designed in a scandinavian country ;-)

You can report worked time by starting and stopping a stopwatch as well as directly declaring time for a given day.

Because time reporting is a matter of trust, this utility versions its internal data with **git** to keep the full history of all the time tracking operations you do once you start tracking. If you do manual modifications on the bookkeeping files of the utility, you will have to commit those changes manually before you can use **tt** again to report worked time or declare holiday, vacation, etc.

Even though this git repository is created as a local one, you are free to push it to a remote location, for instance if you want to do your time tracking across several machines. In this case, it is up to you to use **tt git pull** and **tt git push** to keep things in sync with your remote repository.

COMMANDS

help Prints some basic help.

init <start> [--vacations=<N>] [--weekpattern=<pattern>]

Before you can do any time tracking, you need to initialize the time tracker with this command.

<start> is the date to start time tracking from in dd/mm/yyyy format. This date will be the reference point when it comes to calculating the balance between the time you should have worked and the worked time that you have reported.

If --vacations is specified, <N> is an integer indicating the number of vacation days allowed per year. The default is 25 days. This value is only used a safety net to prevent you from accidentally registering too many vacation days on a given year.

If --weekpattern is specified, <pattern> represents your regular work pattern. It must be a comma-separated list (possibly empty) where each part is of the form DAY=hours:minutes where DAY must have one of the following values: Mon, Tue, Wed, Thu, Fri, Sat, Sun. The parts can be in any order and any day not described in the pattern is considered a weekend day, i.e. a day you are not supposed to work. For instance, the default pattern of 7.5 hours a day from Monday to Friday is the following:

Mon=7:30,Tue=7:30,Wed=7:30,Thu=7:30,Fri=7:30

If your work schedule is not the same every week or if you have to adjust it from time to time, use

the **plan** command to override the regular weekly schedule.

The **init** command creates a bookkeeping directory named .timetracker located in your home directory. This directory is then turned into a git repository by invoking **git init** to allow time tracking operations to be tracked themselves as git commits to ensure traceability.

start Starts the stopwatch. You need use **stop** to stop it and report the interval as worked time.

stop [--force] [description]
Stops the stopwatch and reports the elapsed time as worked time. The optional description is only meant to be used as a hint for humans. If the elapsed time is larger than 24 hours, the program will abort. If the elapsed time is larger than 10 hours or if the clockwatch is not stopped on the same day it was started, the program will abort unless you have used --force to confirm that this was not a mistake. Note that the time actually elapsed is calculated correctly even if you have changed time zones of if a daylight saving change has occurred while the clockwatch was running.

add [--force] <day> <duration> [description]
Registers worked time for a given day. <day> must be given in dd/mm/yyyy format. <duration> must be given in the hours:minutes format. The optional description is only meant to be used as a hint for humans. As for the **stop** command, you cannot report a duration larger than 24 hours and you need to use --force if the duration is larger than 10 hours.

plan [--force] <day> <duration> [description]
Declares the amount of time that should be worked on the given day. This value will take precedence over the value given by the week pattern. <day> is given in dd/mm/yyyy format. <duration> is the amount of time to work in the hh:mm format. The description is optional and is only meant as a hint for humans. As for the **stop** command, you cannot report a duration larger than 24 hours and you need to use --force if the duration is larger than 10 hours.

For example, if you normally work 8 hours from Monday to Friday but you exceptionally need to swap a Friday with a Saturday, you can use this command to declare a duration of 0:00 for the Friday and a duration of 8:00 for the Saturday.

holiday [--half] <day> [description]
Registers a public holiday, i.e. a day you are not supposed to work at all. If --half is used, the holiday is marked as a half-day where you are supposed to work half the normal time. <day> must be given in dd/mm/yyyy format. The optional description is only meant to be used as a hint for humans.

vacation <day>
Registers a vacation day, given in dd/mm/yyyy format. Note that if the day is also registered as a half-holiday, the program will automatically take it into account and count it as 0.5 vacation instead of a full day. The program will abort if the day is already registered as a vacation day, is a week-end day, is a full public holiday or would exceed the yearly limit of vacation days.

sick <day>
Registers a day you stay home sick in dd/mm/yyyy format. The program will abort if the day is already registered as a sick day, a sick child day, a week-end day or a full public holiday.

child <day>

Registers a day you stay home to take care of a sick child in dd/mm/yyyy format. The program will abort if the day is already registered as a sick day, a sick child day, a week-end day or a full public holiday.

report [-v|--verbose] [<type>] [<end>]

Prints a summary of the time tracked up until now or <end> if specified in dd/mm/yyyy format. If -v is specified, the comments for a day, if any, are printed after the report for this day. <type> must have one of the following values:

week	Prints the current week (default)
month	Prints the current month
year	Prints the current year
all	Prints everything since the time tracking began

Legend:

---	Hours due and not worked
###	Hours due and worked
+++	Overtime hours (any time worked past the expected duration for the day)

remove [--force] <type> <day>

Removes the data registered for the given <day> in dd/mm/yyyy format. The program will ask for confirmation unless --force is used. <type> must have one of the following values:

holiday	Removes the day, whether it is a full or a half holiday
vacation	Removes the given vacation day
sick	Removes the given sick day
child	Removes the given sick child day
worked	Removes all the worked time reported for the given day
planned	Removes the entry for the given day from the planning file

git [<args>]

This is a shortcut for executing the given git command from the bookkeeping directory. For instance, **tt git log** will list the modifications done since you invoked the **init** command.