Taskwarrior

What's next?

Dirk Deimeke

December, 9th 2016

Taskwarrior Academy @ TNG Technology Consulting GmbH

Prolog

Dirk Deimeke – d5e.org

Very Simple Rules

- No “Sie” – please use “Du”!
- Questions? Ask!

This workshop . . .

This workshop hopefully is a *real workshop*.

It will live from you doing things and asking,
it is not about me talking all of the time.

Nevertheless I will show you every command.

Taskwarrior

Questions

- You are very welcome here, but I like to know why do you attend?
- What do you expect from this workshop?
- Your Operating System?
- Taskwarrior installed?
  - Which version? – run `task --version`
  - Self compiled or packaged?
- Experience level?

  Novice, intermediate, professional, master, wizard, . . . ?

Custom Directory

If you installed to a custom directory . . .

. . . for example `~/MYDIR`,

you have to replace `/usr/local` with `~/MYDIR` throughout this presentation.

Taskwarrior

Simple ToDo-Lists

A simple example

This is all you need to know to manage very simple to do lists with Taskwarrior.

```
> task add

> task list

> task <ID> done

> task list
```

Starting and Stopping

If you like to see what you are working on, use ‘start’ and ‘stop’.

```
> task <ID> start

> task list

> task <ID> stop

> task list
```

Taskwarrior

Initial Configuration

Choose a theme

Uncomment the theme you want to use from `~/.taskrc`

```
# Color theme (uncomment one to use)
#include /usr/local/share/doc/task/rc/light-16.theme
#include /usr/local/share/doc/task/rc/light-256.theme
#include /usr/local/share/doc/task/rc/dark-16.theme
#include /usr/local/share/doc/task/rc/dark-256.theme
#include /usr/local/share/doc/task/rc/dark-red-256.theme
#include /usr/local/share/doc/task/rc/dark-green-256.theme
#include /usr/local/share/doc/task/rc/dark-blue-256.theme
#include /usr/local/share/doc/task/rc/dark-violets-256.theme
#include /usr/local/share/doc/task/rc/dark-yellow-green.theme
#include /usr/local/share/doc/task/rc/dark-gray-256.theme
#include /usr/local/share/doc/task/rc/dark-gray-blue-256.theme
#include /usr/local/share/doc/task/rc/solarized-dark-256.theme
include /usr/local/share/doc/task/rc/solarized-light-256.theme
#include /usr/local/share/doc/task/rc/no-color.theme
```

Attention: Packaged Taskwarrior

Packaged Taskwarrior

Your package distributor might have different ideas where the theme files should be.

Check with `find / -name no-color.theme -type f 2>/dev/null`

Set weekstart

```
> task show weekstart

> task config weekstart Monday

> tail ~/.taskrc
```

Scripts

```
# Scripts shipped with Taskwarrior
> ls /usr/local/share/doc/task/scripts/*
```

Commandline completion

```
# Commandline completion tabtabtabtabtabtab ;-)
> source /usr/local/share/doc/task/scripts/bash/task.sh

# Make it persistent (Bash)
> echo source /usr/local/share/doc/task/scripts/bash/task.sh >> .bashrc

# ZSH is supported as well
# check /usr/local/share/doc/task/scripts/zsh/_task

# Using Fish?
# check /usr/local/share/doc/task/scripts/fish/task.fish
```

Syntaxhighlighting

```
# Syntaxhighlighting for vim
> [[ -d ~/.vim ]] || mkdir ~/.vim
> cp -r /usr/local/share/doc/task/scripts/vim/* ~/.vim
```

Taskwarrior Cheat Sheet

```
# Use your PDF Viewer of choice
/usr/local/share/doc/task/task-ref.pdf
```

Taskwarrior

Basic Usage

Nearly all commands work on a bunch of tasks

Even the very easy commands from the last section are more mighty than they seem.

- `task add <mods>`
- `task <filter> list`
- `task <filter> start <mods>`
- `task <filter> stop <mods>`
- `task <filter> done <mods>`

To get an overview, take a look at the cheat sheet.
`/usr/local/share/doc/task/task-ref.pdf`

`task <filter> command <mods>`

- Is the basic usage of all task related **write** commands.
- Modifications can be either a change of description, a change of dates or anything else that changes a task.
- Write commands can operate on one task or a group of tasks or even on all tasks.
- Every command may be **abbreviated** up to the minimum that is necessary to identify a single command (or the configured limit).
- Filters can be anything from nothing to simple IDs to regular expressions or Boolean constructs.
- In our simple example we already used the write commands **add**, **done**, **start** and **stop**.

Most important commands

These are the most important commands, just because I use them most ;-)

- `task <filter> modify`
  The name says it, it modifies tasks according to the filter used.
- `task <filter> edit`
  This starts your favourite editor with the tasks you want to change.
  (Remember the syntax highlighting for vim?)
- `task undo`
  Reverts the most recent change to a task.

No kidding!

Please try them!

- `task help`
  Gives an overview of implemented commands and custom reports.
- `man task (taskrc, task-color, task-sync)`
  Show the (almighty) man-page(s).
  Unlike the man-pages of many other programs they are extremely helpful and full of information and examples.

Taskwarrior

Usage of dates

Characters for Dateformats – from ‘man taskrc’

```
m  minimal-digit month,   for example 1 or 12
d  minimal-digit day,     for example 1 or 30
y  two-digit year,        for example 09
D  two-digit day,         for example 01 or 30
M  two-digit month,       for example 01 or 12
Y  four-digit year,       for example 2009
a  short name of weekday, for example Mon or Wed
A  long name of weekday,  for example Monday or Wednesday
b  short name of month,   for example Jan or Aug
B  long name of month,    for example January or August
V  weeknumber,            for example 03 or 37
H  two-digit hour,        for example 03 or 11
N  two-digit minutes,     for example 05 or 42
S  two-digit seconds,     for example 07 or 47
```

ISO supported

Default Dateformat

The default value is the ISO-8601 standard: Y-M-D.

Defined dateformats

The dateformat you define, will be used in **addition** to all the standard supported ISO-8601 formats.

Example: 2013-03-14T22:30:00Z

Set dateformat

```
> task show dateformat

> task config dateformat YMD
> task config dateformat.annotation YMD
> task config dateformat.report YMD

> task show dateformat

> grep dateformat ~/.taskrc

# my dateformat some time ago: YMD-HN
```

Special dates

Relative wording

```
task ...due:today
task ...due:yesterday
task ...due:tomorrow
```

Day number with ordinal

```
task ...due:23rd
task ...due:3wks
task ...due:1day
```

Next occurring weekday

```
task ...due:fri
```

At some point or later (sets the wait date to 2038-01-18)

```
task ...wait:later
task ...wait:someday
```

Start and End of ...

Start and end of ... (remember weekstart setting)

```
task ...due:sow/eow # week
task ...due:soww/eoww # workweek
task ...due:socw/eocw # current week
task ...due:som/eom # month
task ...due:soq/eoq # quarter
task ...due:soy/eoy # year
```

Due and wait

```
> task add due:20161231 'Celebrate Sylvester'
> task add due:Sunday 'Drive home'

> task list

> task x modify wait:2016-12-11

> task list

> task waiting

> task x info
```

Urgency and next

Based on your tasks attributes especially – but not limited to – the due date, Taskwarrior calculates an urgency value which will be used by some reports to sort the tasks.

You can increase urgency by adding the `+next` tag (more on tags later).

See urgency calculation

```
> task x info

> task x mod +next

> task x info

> task show urgency
```

Taskwarrior

Recurring tasks

Recurrence

```
> task waiting
> task x modify due:eom recur:monthly

> task list

> task recurring

# task id changed from x (task modify) to y
# try task x edit
```

Recurrence modifiers (1)

hourly
: Every hour.

daily, day, 1da, 2da, ...
: Every day or a number of days.

weekdays
: Mondays, Tuesdays, Wednesdays, Thursdays, Fridays and skipping weekend days.

weekly, 1wk, 2wks, ...
: Every week or a number of weeks.

biweekly, fortnight
: Every two weeks.

monthly
: Every month.

Recurrence modifiers (2)

quarterly, 1qtr, 2qtrs, ...
: Every three months, a quarter, or a number of quarters.

semiannual
: Every six months.

annual, yearly, 1yr, 2yrs, ...
: Every year or a number of years.

biannual, biyearly, 2yr
: Every two years.

Recurrence based on hours

No alarm!

Nothing is wrong with setting a recurrence to hours or minutes, but please keep in mind that Taskwarrior is not and never will be a calendar application or an alarm clock.

If you want to get notified, you are on your own.

Until and entry

```
> task add due:eom recur:monthly until:20171231 'Pay installment for credit'

> task add 'Prepare slides for workshop'
> task x modify entry:yesterday

> task list
```

Taskwarrior

Holiday and Calendar

Holiday

Attention!

Holiday has nothing in common with the German words *Ferien* or *Urlaub* (this would be vacation). (Public) Holiday means *Feiertag*.

You can add holidays by either adding them via `task config` on the commandline or by adding them directly to the `~/.taskrc`-File or by including an external holiday definition.

On holidata.net you find a growing list of holiday dates, licensed CC-BY and offered by volunteers. Service was introduced by the Taskwarrior team, who is responsible for hosting and conversion to different formats.

We are currently working on an algorithmic approach to calculate holiday dates.

Add holiday / Configure calendar

```
> task config holiday.swissnationalday.name Swiss National Day
> task config holiday.swissnationalday.date 20170801

# Holiday is not highlighted by default
> task calendar 08 2017



> task show calendar
> task config calendar.holidays full

> task cale 08 2017
```

Calendar with due tasks

```
> task config calendar.holidays sparse
> task config calendar.details full

> task cale
```

Taskwarrior

Priorities

Priorities

```
> task long

> task x modify pri:H # can be either (H)igh, (M)edium or (L)ow

# you can change this to be any value you want
> task show uda.priority

> task long
```

Taskwarrior

Projects, Tags and Annotations

Project and subproject

```
> task x modify pro:taskwarrior
> task y modify pro:taskwarrior.tngtech
> task z modify pro:private

> task list
```

Projects

```
> task projects

> task pro:taskwarrior ls

> task x done
```

Tags

```
> task x modify +banking
> task y modify +banking

> task list

> task x mod -banking +münchen

> task +münchen list
```

Annotations

```
> task x annotate 'Do not forget your head'

> task y annotate 'Use dads account'

> task list

> task y denotate 'Use dads account'
```

Taskwarrior

Dependencies

Dependency, part 1

```
> task add 'Send letter to Fritz'

> task add 'Write letter'

> task x modify depends:y

> task blocked

> task unblocked
```

Dependency, part 2

```
> task x done

> task list
```

Undo

```
> task undo
```

Dependency, part 3

```
> task x,y done

> task blocked
```

Taskwarrior

Reports

Predefined reports (from task reports), part 1

These reports were already used.

- **blocked** Lists all blocked tasks matching the specified criteria
- **list** Lists all tasks matching the specified criteria
- **long** Lists all task, all data, matching the specified criteria
- **projects** Shows a list of all project names used, and how many tasks are in each
- **recurring** Lists recurring tasks matching the specified criteria
- **unblocked** Lists all unblocked tasks matching the specified criteria
- **waiting** Lists all waiting tasks matching the specified criteria

Predefined reports (from task reports), part 2

New ones:

- **active** Lists active tasks matching the specified criteria
- **all** Lists all tasks matching the specified criteria, including parents of recurring tasks
- **blocking** Blocking tasks
- **burndown.daily** Shows a graphical burndown chart, by day
- **burndown.monthly** Shows a graphical burndown chart, by month
- **burndown.weekly** Shows a graphical burndown chart, by week
- **completed** Lists completed tasks matching the specified criteria

Predefined reports (from task reports), part 3

And more:

- **ghistory.annual** Shows a graphical report of task history, by year
- **ghistory.monthly** Shows a graphical report of task history, by month
- **history.annual** Shows a report of task history, by year
- **history.monthly** Shows a report of task history, by month
- **information** Shows all data and metadata for specified tasks
- **ls** Minimal listing of all tasks matching the specified criteria
- **minimal** A really minimal listing
- **newest** Shows the newest tasks
- **next** Lists the most urgent tasks

Predefined reports (from task reports), part 4

The leftovers:

- **oldest** Shows the oldest tasks
- **overdue** Lists overdue tasks matching the specified criteria
- **ready** Most urgent actionable tasks
- **summary** Shows a report of task status by project
- **tags** Shows a list of all tags used

26 reports in total (as told by `task reports`)

Test the reports

```
> task burndown.daily

> task ghistory.annual
> task ghistory.monthly

> task history.monthly

> task ls

> task minimal

> task summary
```

Report definitions

```
> task show report.minimal
> task show report.list

> task show report # to see all reports defined
```

Dirks former task list

```
> echo '
report.ll.description=Dirks task list
report.ll.columns=id,project,priority,due,due.countdown,tags,description
report.ll.labels=ID,Project,Pri,Due,Countdown,Tags,Description
report.ll.sort=due+,priority-,project+,description+
report.ll.filter=status:pending
' >> ~/.taskrc

task ll
```

Set default command

```
> task show default

> task config default.command ll
> task
```

Taskwarrior

Filters

Filtering in general

You can filter for any modifier. If you don't use a modifier description is searched for the term, which may be a regular expression, on the command line. Filters may be combined.

Filter Modifiers

The following attribute modifiers maybe applied as well. Names in brackets can be used alternatively.

So a filter can look like `attribute.modifier:value`.

- before, after
- none, any
- is (equals), isnt (not)
- has (contains), hasnt
- startswith (left), endswith (right)
- word, noword

Attribute modifiers

```
> task due.before:20161224
```

```
> task project.not:taskwarrior
```

Combining with logical AND

```
> task project:taskwarrior +banking
> task status:completed project:taskwarrior
> task status:completed project:taskwarrior completed

> task show report.ll.filter
```

Logical OR ...

```
> task list

> task \( pro:taskwarrior or pro:private \) list
# Brackets must be escaped for the shell
```

Filter in reports

```
> task show filter
```

What you see is NOT what you modify!

```
> task project:taskwarrior next # has a filter
```

```
> task show report.next.filter # see status:pending
```

BUT!

`task project:taskwarrior modify due:eoy`
works **without any filters**. (There will be a notification in later versions).

Most probably
`task project:taskwarrior status:pending modify due:eoy`
is what you want.

Taskwarrior

Search and find

Searches

```
> task

> task pay

> task /[Pp]ay/
```

Search configuration

```
> task show search

> task show regex
```

Taskwarrior

Context

Contexts

Context is a user-defined filter, which is automatically applied to all commands that filter the task list. In particular, any report command will have its result affected by the current active context.

- `task context define <name> <filter>`
- `task context delete <name>`
- `task context <name>` – sets active context
- `task context show` – shows active context
- `task context list` – lists available contexts
- `task context none` – clears active context

Taskwarrior

Virtual tags

Virtual Tags (1)

- **ACTIVE** – Task is started
- **ANNOTATED** – Task has annotations
- **BLOCKED** – Task is blocked
- **BLOCKING** – Task is blocking
- **CHILD** – Task has a parent
- **COMPLETED** – Task has completed status
- **DELETED** – Task has deleted status
- **DUE** – Task is due
- **LATEST** – Task is the newest added task
- **MONTH** – Task is due this month
- **ORPHAN** – Task has any orphaned UDA values
- **OVERDUE** – Task is overdue

Virtual Tags (2)

- **PARENT** – Task is a parent
- **PENDING** – Task has pending status
- **PRIORITY** – Task has a priority
- **PROJECT** – Task has a project
- **READY** – Task is actionable
- **SCHEDULED** – Task is scheduled
- **TAGGED** – Task has tags
- **TODAY** – Task is due today
- **TOMORROW** – Task is due sometime tomorrow
- **UDA** – Task has any UDA values
- **UNBLOCKED** – Task is not blocked
- **UNTIL** – Task expires

Virtual Tags (3)

- **WAITING** – Task is waiting
- **WEEK** – Task is due this week
- **YEAR** – Task is due this year
- **YESTERDAY** – Task was due sometime yesterday

Taskwarrior

UDA – User Defined Attributes

What is an user defined attribute?

A UDA is a new metadata item that you define, and Taskwarrior faithfully stores, displays, and modifies.

Taskwarrior simply treats it as a data value with a name, allowing you to sort by it, use it in a report, import and export it.

It is intended that, once configured, a UDA is indistinguishable from core attributes, and will not impart performance penalties.

Core attributes

```
> task columns project # show the definition of the project column

> task columns # shows all defined columns
```

Estimate

```
> task config uda.estimate.type numeric
> task config uda.estimate.label Est

> task add 'Paint the door' project:Home estimate:4
```

Size

```
> task config uda.size.type string
> task config uda.size.label Size
> task config uda.size.values large,medium,small

> task config uda.size.default medium
> task config urgency.uda.size.coefficient 2.8
```

Taskwarrior

DOM – Document Object Model

Document Object Model in general

Taskwarrior has a Document Object Model, or DOM, which defines a way to reference all the data managed by taskwarrior. You may be familiar with the DOM implemented by web browsers that let you access details on a page programmatically.

There is a `_get` helper command that queries data using a DOM reference.

Milky Example

```
> task add Buy milk due:tomorrow +store project:Home pri:H

> task x info

> task _get x.description
> task _get x.uuid
> task _get x.due.year
```

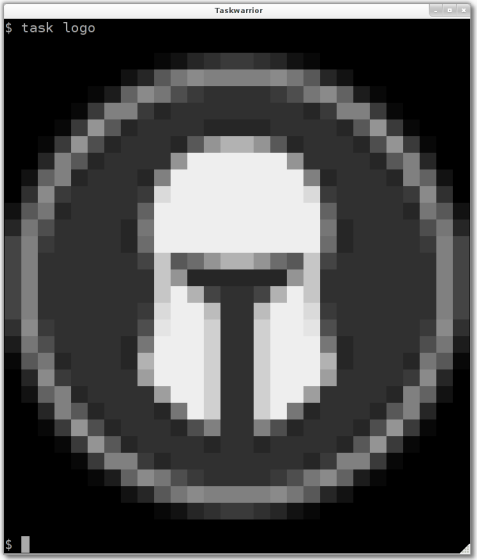
Epilog

This is by far not all

task log
: for logging a task after it is already done.

task diagnostics
: to help support for diagnostic purpose.

...
: and many more!


Taskwarrior
$ task logo
$

Thank you!

Dirk Deimeke, 2016, CC-BY

dirk@deimeke.net

d5e.org – speakerdeck .com/ddeimeke

Download the PDF for clickable links.

Links

- Taskwarrior Homepage
- Taskwarrior Documentation
- Taskwarrior Guides and Presentations

Support

- Submit your details to our **Q & A site**, then wait patiently for the community to respond.
- **Email** us at support@taskwarrior.org, then wait patiently for a volunteer to respond.
- Join us **IRC** in the #taskwarrior channel on Freenode.net, and get a quick response from the community.
- Even though **Twitter** is no means of support, you can get in touch with @taskwarrior.
- We have a **User Mailinglist** which you can join anytime to discuss about Taskwarrior and techniques.
- The **Developer Mailinglist** is focussing on a more technical oriented audience.