Taskwarrior Command-line to-do list

START HERE

`task add <description>`
Create a task

`task undo`
Revert last change

`task next`
What to do now

`task sync`
Sync with a server

`task <id> done`
Complete a task

`task <id> modify <attrs>`
Change a task

`task <id> annotate <note>`
Add a note

`task project:<name> list`
Filter a report

pending new task → `start` → **active** has start date → `done` → **completed** history → `delete` → **deleted** kept in data

CONCEPTS

Syntax is task [filter] <command> [modifications]. The filter picks tasks, the command acts on them.

IDs are working-set numbers that change as tasks finish; use the UUID for anything lasting.

Quote text that the shell would eat: task add 'Call (Bob) & Eve'. Use -- to stop attribute parsing.

ADD AND CHANGE

`task add <description>`
Create a task

`task add <desc> project:<p> +<tag> due:<date> priority:H`
Add with attributes

`task <id> modify <attrs>`
Change attributes

`task <id> done`
Mark completed

`task <id> delete`
Delete a task

`task <id> annotate <note>`
Attach a note

`task <id> denotate <text>`
Remove a note

`task <id> start`
Mark as active

ADD AND CHANGE (cont.)

`task <id> stop`
Stop working on it

`task <id> edit`
Edit in $EDITOR

`task <id> duplicate`
Copy a task

`task log <description>`
Record an already-done task

`task undo`
Revert the last change

`task <filter> modify project:`
Clear an attribute
Assign an empty value to remove project, due, tags etc.

REPORTS

`task next`
Most urgent tasks

`task list`
All pending tasks

`task <id> info`
Full details of a task

`task all`
Every task, any status

`task completed`
Finished tasks

`task waiting`
Hidden until wait date

`task overdue`
Past their due date

REPORTS (cont.)

`task projects`
Projects with counts

`task tags`
Tags in use

`task calendar`
Months with due dates

`task summary`
Progress per project

`task reports`
List all report names

`task count <filter>`
Count matching tasks

`task _get <id>.<attr>`
Script-friendly attribute value

`task <filter> export`
JSON output

FILTERS

`project:<name>`
Tasks in a project

`+<tag>` / `-<tag>`
Has / lacks a tag

`priority:H`
By priority (H, M, L)

`due:<date>`
Due on a date

`due.before:<date>`
Due earlier than

`status:pending`
By status

`/<regex>/`
Match description

FILTERS (cont.)

`<f1> or <f2>`
Combine with or / and / xor
Wrap groups in parentheses and quote them for the shell.

`+OVERDUE`
Virtual tag: past due

`+TODAY / +WEEK`
Virtual tags: due today / this week

`+READY`
Pending and not blocked

`rc.<setting>=<value>`
Override config for one run

DATES

`due:tomorrow`
Relative day name

`due:eow`
End of week

`due:eom`
End of month

`due:friday`
Next named weekday

`due:2026-12-31`
Absolute ISO date

`due:now+3d`
Offset by a duration

`wait:<date>`
Hide until a date

`sow / eow / som / eom / soy / eoy`
Start or end of week, month, year

DATES (cont.)

`3d / 2w / 1m / 1y`
Duration units

ORGANISE

`task context define <name> <filter>`
Save a filter as context

`task context <name>`
Activate a context

`task context none`
Deactivate context

`task context list`
Show defined contexts

`task <id> modify depends:<id>`
Block on another task

`task add <desc> recur:weekly due:eow`
Recurring task

`task <id> modify priority:L`
Set priority

`task <id> modify +<tag> -<tag>`
Add / remove tags

`task <filter> modify project:<name>`
Bulk move to a project

DATA AND SYNC

`task sync`
Sync with the configured server

DATA AND SYNC (cont.)

`task config <key> <value>`
Set a taskrc value

`task show`
Show effective configuration

`task export > <file>`
Back up as JSON

Config lives in ~/.taskrc (or ~/.config/task/taskrc); data in ~/.local/share/task. Sync settings are the sync.* keys; see task-sync(5).

Urgency is a sum of weighted coefficients (priority, due, age, tags, project). Tune with urgency.*.coefficient settings.

RECIPES

Daily triage

```
task +OVERDUE list
task +TODAY or +DUETODAY list
task next limit:5
```

Weekly project

```
task add 'Draft report'
project:work.q4 due:eow
+writing
task project:work.q4 list
task 12 start
task 12 done
```

Work-only view

```
task context define work
project:work or +work
task context work
task next
task context none
```