Task YAML task runner

CONCEPTS

Running task with no name runs the task called default. Tasks live in Taskfile.yml, found by searching upwards from the current directory.

task <name> CLI call
↓ deps
Dependencies run first, in parallel
↓ up-to-date?
Check sources / status
↓ cmds
Commands run in order

Variables are templated with {{.NAME}}; environment variables use $NAME in the shell.

RUNNING

task --list
List tasks that have a desc

task --list-all
List all tasks, even without desc

task *<name>*
Run a task

task --dry *<name>*
Print commands without executing

task --force *<name>*
Run even if up to date

task --parallel *<a>* *<b>*
Run the named tasks concurrently

task --summary *<name>*
Show the task's summary text

task --watch *<name>*
Re-run when sources change

task --yes *<name>*
Answer yes to prompts

task --taskfile *<file>*
Use a different Taskfile

task --dir *<path>*
Run in and look for Taskfile in a directory

task --global *<name>*
Run from the Taskfile in $HOME

task --silent *<name>*
Do not echo commands

task --init
Create a starter Taskfile.yml

TASK BASICS

version: '3'
Required first line

tasks: *<name>*: **cmds:**
Define a task and its commands

cmd: *<command>*
Single command shorthand

desc: *<text>*
Text shown in --list

summary: *<text>*
Long text for --summary

aliases: [*<alt>***]**
Alternative task names

dir: *<path>*
Working directory of the task

silent: true
Hide command echo

internal: true
Hide from CLI and --list

- task: *<other>*
Call another task as a command

ignore_error: true
Keep going if the command fails

default:
Task run by plain task

DEPENDENCIES AND LOOPS

deps: [*<a>***, ***<b>***]**
Run these first, concurrently

deps: - task: *<t>* **vars: {...}**
Dependency with variables

for: [*<a>***, ***<b>***] cmd: ... {{.ITEM}}**
Loop a command over values

defer: *<command>*
Run cleanup when the task ends

platforms: [linux, darwin]
Limit task or command to platforms

VARIABLES

vars: *<NAME>*: *<value>*
Define a variable

{{.NAME}}
Use a variable in a command

<NAME>: **{sh:** *<command>***}**
Variable from command output

VARIABLES (cont.)

{{.NAME | default "x"}}
Fallback if unset

{{.CLI_ARGS}}
Everything after -- on the CLI

env: *<NAME>*: *<value>*
Export an environment variable

dotenv: ['.env']
Load env vars from files

requires: vars: [*<NAME>***]**
Fail if variable is missing

UP-TO-DATE AND CHECKS

sources: ['/*.go']**
Files whose change triggers a run

generates: [*<file>***]**
Files the task produces

method: checksum
Compare content (default)

status: [*<test cmd>***]**
Skip when all commands exit 0

preconditions: - sh: *<test>*
msg: *<text>*
Abort with a message if the test fails

prompt: *<question>*
Ask for confirmation first

INCLUDES AND LAYOUT

includes: *<ns>*: *<path>*
Import tasks as ns:<task>

taskfile: *<path>*
Path in a long-form include

dir: *<path>* **(include)**
Working directory for included tasks

task *<ns>*:*<task>*
Run a namespaced task

RECIPES

Build only when sources change

```
tasks:
  build:
    desc: Build the binary
    sources: ['**/*.go', go.mod]
    generates: [bin/app]
    cmds:
      - go build -o bin/app .
```