Deno Secure TS runtime

CONCEPTS

Scripts run in a sandbox: no disk, network or env access unless you grant it with --allow-* flags (or answer the prompt).

Specifier jsr: npm: https:
↓ resolve
Cache deno.lock pins it
↓ check
Type-check TypeScript built in
↓ permissions
Run sandboxed by flags

RUN AND PERMISSIONS

deno run *<file>*
Run a script or URL

deno run -A *<file>*
Allow everything (trusted code only)

--allow-read[=*<path>***] / -R**
Allow file reads, optionally scoped

--allow-write[=*<path>***] / -W**
Allow file writes

--allow-net[=*<host>***] / -N**
Allow network access

--allow-env[=*<VAR>***] / -E**
Allow reading env vars

--deny-read=*<path>*
Deny a path even if allowed more broadly

--no-prompt
Fail instead of asking for permission

deno run --watch *<file>*
Restart on file changes

--env-file[=*<path>***]**
Load env vars from a file

deno eval "*<code>***"**
Evaluate a snippet

deno serve *<file>*
Run a default-export fetch handler as a server

TASKS AND CONFIG

deno task
List tasks in deno.json

deno task *<name>*
Run a task

TASKS AND CONFIG (cont.)

deno init *<dir>*
New project with main.ts and a test

DEPENDENCIES

deno add *<pkg>*
Add dependencies to the config

deno add jsr:@std/*<name>*
Add a JSR package

deno add npm:*<pkg>*
Add an npm package

deno install
Install all deps from deno.json/ package.json

deno outdated
Show newer dependency versions

deno outdated --update
Update within semver-compatible range

TEST AND CHECK

deno test
Run all tests

deno test *<file>*
Run one test file

deno test --filter "*<name>***"**
Run tests matching a name

deno check *<file>*
Type-check without running

A test is Deno.test("name", () ⇒ {...}) in a file named *_test.ts or *.test.ts.

FORMAT AND LINT

deno fmt
Format all files

deno fmt --check
Fail if anything is unformatted

deno lint
Lint all files

deno lint --fix
Auto-fix what rules allow

BUILD AND PUBLISH

deno compile *<file>*
Bundle into a standalone executable