Wrangler Cloudflare Workers CLI

CONCEPTS

Wrangler is the Workers CLI. Run it per project with npx wrangler (or pnpm/yarn) so the version matches the repo.

Code + config src/ and wrangler.jsonc
↓ `wrangler dev`
Local preview localhost:8787
↓ `wrangler deploy`
Cloudflare edge live Worker
↓ `wrangler tail`
Live logs debug and watch

Bindings (KV, R2, D1) are declared in the config file and reach your code as env.<NAME>.

ACCOUNT

`wrangler login`
Sign in through the browser

`wrangler whoami`
Show the signed-in account

PROJECT

`npm create cloudflare@latest`
Create a new Worker project

`wrangler init [<name>]`
Set up Wrangler in a folder

`wrangler types`
Generate types for your bindings

DEVELOP

`wrangler dev`
Run the Worker locally

`wrangler dev --port <n>`
Use another local port

`wrangler dev --remote`
Run against remote resources

`wrangler dev --local`
Keep all bindings local

DEPLOY

`wrangler deploy`
Publish the Worker

`wrangler deploy --env <name>`
Deploy an environment

DEPLOY (cont.)

`wrangler deploy --dry-run --outdir <dir>`
Build without publishing

`wrangler delete [<name>]`
Remove a Worker

LOGS

`wrangler tail`
Stream live logs

`wrangler tail --format pretty`
Human-readable log lines

`wrangler tail --status error`
Only failed requests

SECRETS

`wrangler secret put <KEY>`
Add or update a secret

`wrangler secret list`
List secret names

VERSIONS

`wrangler versions upload`
Upload without going live

`wrangler versions deploy <id>@<pct>`
Split traffic between versions

`wrangler rollback [<id>]`
Return to an earlier version

KV

`wrangler kv namespace create <name>`
Create a namespace

`wrangler kv key put <key> <value> --binding <b>`
Write a key

`wrangler kv key get <key> --binding <b>`
Read a key

R2

`wrangler r2 bucket create <name>`
Create a bucket

`wrangler r2 object put <bucket>/<key> --file <path>`
Upload an object

R2 (cont.)

`wrangler r2 object get <bucket>/<key> --file <path>`
Download an object

D1

`wrangler d1 create <name>`
Create a database

`wrangler d1 execute <db> --command "<sql>"`
Run one SQL statement

`wrangler d1 execute <db> --file <file.sql>`
Run a SQL file

`wrangler d1 migrations create <db> <name>`
Add a migration file

`wrangler d1 migrations apply <db>`
Apply pending migrations

D1 commands run locally unless you pass --remote; apply migrations locally first, then with --remote.

CONFIG

Config fields

Field	Purpose
`name`	Worker name
`main`	Entry file
`compatibility_date`	Runtime behaviour date
`compatibility_flags`	Opt-in runtime features
`vars`	Plain variables
`kv_namespaces`	KV bindings
`r2_buckets`	R2 bindings
`d1_databases`	D1 bindings
`routes`	Custom domains and routes
`assets`	Static asset directory
`observability`	Logs and traces
`env.<name>`	Per-environment overrides

Cloudflare recommends wrangler.jsonc for new projects; wrangler.toml still works.

vars and bindings are not inherited by [env.<name>]: repeat them in each environment.