Vercel CLI Deploy from the terminal

START HERE

`vercel login`
Sign in

`vercel env pull`
Get env vars locally

`vercel link`
Tie folder to a project

`vercel dev`
Run locally

`vercel`
Deploy a preview

`vercel logs`
Read runtime logs

`vercel --prod`
Deploy to production

`vercel rollback`
Undo a bad release

Local folder .vercel link → `vercel` → **Preview** unique URL → `vercel promote` → **Production** live domains → `vercel rollback` → **Earlier deployment** previous build

CONCEPTS

A plain vercel is a preview deploy; only --prod (or the first deploy of a project) goes to production.

In CI, authenticate with the VERCEL_TOKEN environment variable rather than --token so it stays out of process lists.

stdout of a deploy is always the deployment URL, so vercel > url.txt works in scripts.

SETUP

`npm i -g vercel`
Install the CLI

`vercel login`
Sign in to your account

`vercel link`
Link folder to a Vercel project
Writes .vercel/; commit neither it nor its contents.

`vercel whoami`
Show the signed-in user

`vercel switch` *<team>*
Change active team scope

`vercel --version`
Show CLI version

`vercel upgrade`
Update the CLI

`vercel logout`
Sign out

DEPLOY

`vercel`
Deploy a preview
Prints the deployment URL on stdout.

`vercel --prod`
Deploy to production

`vercel deploy --prebuilt`
Deploy output of vercel build
System env vars are missing at build time with --prebuilt.

`vercel build`
Build locally into .vercel/output

`vercel --force`
Deploy without the build cache

`vercel --prod --skip-domain`
Deploy prod, keep domains on old one
Finish later with vercel promote.

`vercel --logs`
Print build logs while deploying

DEPLOY (cont.)

`vercel --yes`
Skip project setup questions

`vercel --no-wait`
Exit before the build finishes

`vercel --env` *<KEY>*`=`*<value>*
Set a runtime env var

`vercel --build-env` *<KEY>*`=`*<value>*
Set a build-time env var

`vercel --target=`*<env>*
Deploy to a custom environment

`vercel --archive=tgz`
Upload as archive (many files)

`vercel redeploy` *<url>*
Rebuild an existing deployment

LOCAL & ENV

`vercel dev`
Run the project locally

`vercel dev --port` *<port>*
Run on a given port

`vercel pull`
Fetch env vars and settings to .vercel/
Needed before vercel build or vercel dev.

`vercel env ls`
List environment variables

`vercel env add` *<name> <environment>*
Add an env var
Production and Preview values default to secret type.

`vercel env update` *<name> <environment>*
Change an env var's value

`vercel env rm` *<name> <environment>*
Remove an env var

`vercel env pull` *<file>*
Write dev vars to a file
Merges into an existing file; local-only keys are kept.

`vercel env pull --environment=preview`
Pull Preview variables instead

`vercel env run --` *<command>*
Run a command with the env vars
No file is written; use -e to pick the environment.

`vercel env add` *<name> <environment> < <file>*
Use file content as the value

INSPECT & LOGS

`vercel list`
Recent deployments

`vercel inspect` *<url>*
Show deployment details

`vercel inspect` *<url>* **`--logs`**
Show its build logs

`vercel inspect` *<url>* **`--wait`**
Wait until it is ready
Exits 124 if the wait times out.

`vercel logs`
Runtime logs, last 30 minutes

`vercel logs --since` *<duration>*
Widen the time window

`vercel logs --deployment` *<url>*
Logs of one deployment

`vercel logs --query '`*<kql>*`'`**
Filter with a KQL query
Example: 'level:error AND timeout'.

`vercel logs requests`
CDN request records

`vercel logs --json`
One JSON object per line

`vercel open`
Open project in dashboard

RELEASE

`vercel promote` *<url>*
Make a deployment the current one

`vercel promote status`
Show promote progress

`vercel rollback`
Return production to prior deployment

`vercel rollback` *<url>*
Roll back to a specific deployment

`vercel rollback status`
Show rollback progress

`vercel remove` *<url>*
Delete a deployment

`vercel bisect`
Binary-search deployments for a bug

DOMAINS

`vercel domains ls`
List domains

`vercel domains add` *<domain> <project>*
Add a domain to a project

DOMAINS (cont.)

`vercel domains rm` *<domain>*
Remove a domain

`vercel alias set` *<url> <domain>*
Point a domain at a deployment

`vercel alias ls`
List aliases

`vercel alias rm` *<domain>*
Remove an alias

`vercel dns ls` *<domain>*
List DNS records

`vercel dns add` *<domain> <name> <type> <value>*
Add a DNS record

`vercel certs ls`
List certificates

OPTIONS

`--cwd` *<dir>*
Run against another directory

`--scope` *<team>*
Run as another team

`--token` *<token>*
Authenticate with a token

`--project` *<name>*
Choose the project

`--non-interactive`
Never prompt (CI, scripts)

`--debug`
Verbose output

Exit codes

Code	Meaning
0	success or help
1	operation failed
2	usage error

RECIPES

First deploy of a project

```
npm i -g vercel
vercel login
vercel link
vercel env pull
vercel --prod
```

CI deploy with a token

```
export VERCEL_TOKEN=<token>
vercel pull --yes --
environment=production
vercel build --prod
vercel deploy --prebuilt --prod
```

Stage, check, then promote

```
vercel --prod --skip-domain
vercel inspect <url> --logs
vercel promote <url>
```

Alias a preview deployment

```
vercel > url.txt
vercel alias set $(cat url.txt)
preview.example.com
```