Command Line Interface

- Dev server
  - vite
    - Usage
    - Options
- Build
  - vite build
    - Usage
    - Options
- Others
  - vite optimize
    - Usage
    - Options
  - vite preview
    - Usage
    - Options

Dev server

`vite`

Start Vite dev server in the current directory. `vite dev` and `vite serve` are aliases for `vite`.

Usage

```
vite [root]
```

Options

Options	
`--host [host]`	Specify hostname (`string`)
`--port <port>`	Specify port (`number`)
`--open [path]`	Open browser on startup (`boolean | string`)
`--cors`	Enable CORS (`boolean`)
`--strictPort`	Exit if specified port is already in use (`boolean`)
`--force`	Force the optimizer to ignore the cache and re-bundle (`boolean`)
`--experimentalBundle`	Use experimental full bundle mode (this is highly experimental) (`boolean`)
`-c, --config <file>`	Use specified config file (`string`)
`--base <path>`	Public base path (default: `/`) (`string`)
`-l, --logLevel <level>`	info | warn | error | silent (`string`)
`--clearScreen`	Allow/disable clear screen when logging (`boolean`)
`--configLoader <loader>`	Use `bundle` to bundle the config with Rolldown, or `runner` (experimental) to process it on the fly, or `native` (experimental) to load using the native runtime (default: `bundle`)
`--profile [name]`	Start built-in Node.js inspector and write the profile to `<name>.cpuprofile` (check Performance bottlenecks) (`boolean | string`)
`-d, --debug [feat]`	Show debug logs (`string | boolean`)
`-f, --filter <filter>`	Filter debug logs (`string`)
`-m, --mode <mode>`	Set env mode (`string`)
`-h, --help`	Display available CLI options
`-v, --version`	Display version number

Build

`vite build`

Build for production.

Usage

```
vite build [root]
```

Options

Options	
`--target <target>`	Transpile target (default: `"baseline-widely-available"`) (`string`)
`--outDir <dir>`	Output directory (default: `dist`) (`string`)
`--assetsDir <dir>`	Directory under outDir to place assets in (default: `"assets"`) (`string`)
`--assetsInlineLimit <number>`	Static asset base64 inline threshold in bytes (default: `4096`) (`number`)
`--ssr [entry]`	Build specified entry for server-side rendering (`string`)
`--sourcemap [output]`	Output source maps for build (default: `false`) (`boolean | "inline" | "hidden"`)
`--minify [minifier]`	Enable/disable minification, or specify minifier to use (default: `"oxc"`) (`boolean | "oxc" | "terser" | "esbuild"`)
`--manifest [name]`	Emit build manifest json (`boolean | string`)
`--ssrManifest [name]`	Emit ssr manifest json (`boolean | string`)
`--emptyOutDir`	Force empty outDir when it's outside of root (`boolean`)
`-w, --watch`	Rebuilds when modules have changed on disk (`boolean`)
`-c, --config <file>`	Use specified config file (`string`)
`--base <path>`	Public base path (default: `/`) (`string`)
`-l, --logLevel <level>`	info | warn | error | silent (`string`)
`--clearScreen`	Allow/disable clear screen when logging (`boolean`)
`--configLoader <loader>`	Use `bundle` to bundle the config with Rolldown, or `runner` (experimental) to process it on the fly, or `native` (experimental) to load using the native runtime (default: `bundle`)
`--profile [name]`	Start built-in Node.js inspector and write the profile to `<name>.cpuprofile` (check Performance bottlenecks) (`boolean | string`)
`-d, --debug [feat]`	Show debug logs (`string | boolean`)
`-f, --filter <filter>`	Filter debug logs (`string`)
`-m, --mode <mode>`	Set env mode (`string`)
`-h, --help`	Display available CLI options

Options	
`--app`	Build all environments, same as `builder: {}` (`boolean`, experimental)

Others

`vite optimize`

Pre-bundle dependencies.

Deprecated: the pre-bundle process runs automatically and does not need to be called.

Usage

```
vite optimize [root]
```

Options

Options	
`--force`	Force the optimizer to ignore the cache and re-bundle (`boolean`)
`-c, --config <file>`	Use specified config file (`string`)
`--base <path>`	Public base path (default: `/`) (`string`)
`-l, --logLevel <level>`	info | warn | error | silent (`string`)
`--clearScreen`	Allow/disable clear screen when logging (`boolean`)
`--configLoader <loader>`	Use `bundle` to bundle the config with Rolldown, or `runner` (experimental) to process it on the fly, or `native` (experimental) to load using the native runtime (default: `bundle`)
`-d, --debug [feat]`	Show debug logs (`string | boolean`)
`-f, --filter <filter>`	Filter debug logs (`string`)
`-m, --mode <mode>`	Set env mode (`string`)
`-h, --help`	Display available CLI options

`vite preview`

Locally preview the production build. Do not use this as a production server as it's not designed for it.

This command starts a server in the build directory (by default `dist`). Run `vite build` beforehand to ensure that the build directory is up-to-date. Depending on the project's configured `appType`, it makes use of certain middleware.

Usage

```
vite preview [root]
```

Options

Options	
`--host [host]`	Specify hostname (`string`)
`--port <port>`	Specify port (`number`)
`--strictPort`	Exit if specified port is already in use (`boolean`)
`--open [path]`	Open browser on startup (`boolean | string`)
`--outDir <dir>`	Output directory (default: `dist`) (`string`)
`-c, --config <file>`	Use specified config file (`string`)
`--base <path>`	Public base path (default: `/`) (`string`)
`-l, --logLevel <level>`	info | warn | error | silent (`string`)
`--clearScreen`	Allow/disable clear screen when logging (`boolean`)
`--configLoader <loader>`	Use `bundle` to bundle the config with Rolldown, or `runner` (experimental) to process it on the fly, or `native` (experimental) to load using the native runtime (default: `bundle`)
`-d, --debug [feat]`	Show debug logs (`string | boolean`)
`-f, --filter <filter>`	Filter debug logs (`string`)
`-m, --mode <mode>`	Set env mode (`string`)
`-h, --help`	Display available CLI options

□□□ #2
□ □□□ □□□ 2026-10-31 13:14:11 UTC
□ □□□ □□□ 2026-10-31 13:14:12 UTC