Getting Started

Overview

Vitest is a blazing fast unit test framework powered by Vite.

You can learn more about the rationale behind the project in the Why Vitest section.

Trying Vitest Online

You can try Vitest online on StackBlitz. It runs Vitest directly in the browser, and it is almost identical to the local setup but doesn't require installing anything on your machine.

Adding Vitest to your Project

Learn how to install by Video

With npm

```bash
npm install -D vitest
```

or with yarn

```bash
yarn add -D vitest
```

or with pnpm

```bash
pnpm add -D vitest
```

> TIP
>
> Vitest requires Vite >=v3.0.0 and Node >=v14

It is recommended that you install a copy of `vitest` in your `package.json`, using one of the methods listed above. However, if you would prefer to run `vitest` directly, you can use `npx vitest` (the `npx` command comes with npm and Node.js).

The `npx` command will execute the command either from a local `node_modules/.bin` installing any packages needed in order for the command to run. By default, npx will check whether command exists in $PATH, or in the local project binaries, and execute that. If command is not found, it will be installed prior to execution.

Configuring Vitest

One of the main advantages of Vitest is its unified configuration with Vite. If present, `vitest` will read your root `vite.config.ts` to match with the plugins and setup as your Vite app. For example, your Vite resolve.alias and plugins configuration will work out-of-the-box. If you want a different configuration during testing, you can:

- Create `vitest.config.ts`, which will have the higher priority
- Pass `--config` option to CLI, e.g. `vitest --config ./path/to/vitest.config.ts`
- Use `process.env.VITEST` or `mode` property on `defineConfig` (will be set to `test` if not overridden) to conditionally apply different configuration in `vite.config.ts`

To configure `vitest` itself, add `test` property in your Vite config. You'll also need to add a reference to Vitest types using a triple slash command at the top of your

config file, if you are importing `defineConfig` from `vite` itself.

```ts
import { defineConfig } from 'vitest/config'

export default defineConfig({
  test: {
    // ...
  },
})
```

See the list of config options in the Config Reference

Command Line Interface

In a project where Vitest is installed, you can use the `vitest` binary in your npm scripts, or run it directly with `npx vitest`. Here are the default npm scripts in a scaffolded Vitest project:

```json
{
  "scripts": {
    "test": "vitest",
    "coverage": "vitest run --coverage"
  }
}
```

To run tests once without watching for file changes, use `vitest run`. You can specify additional CLI options like `--port` or `--https`. For a full list of CLI options, run `npx vitest --help` in your project.

Learn more about the Command Line Interface

IDE Integrations

We also provided a official extension for Visual Studio Code to enhance your testing experience with Vitest.

Install from VS Code Marketplace

Learn more about IDE Integrations

Examples

Example	Source	Playground
`basic`	GitHub	Play Online
`fastify`	GitHub	Play Online
`graphql`	GitHub	Play Online
`image-snapshot`	GitHub	Play Online
`lit`	GitHub	Play Online
`mocks`	GitHub	Play Online
`nextjs`	GitHub	Play Online
`playwright`	GitHub	
`puppeteer`	GitHub	
`react-enzyme`	GitHub	Play Online
`react-mui`	GitHub	Play Online
`react-storybook`	GitHub	Play Online
`react-testing-lib-msw`	GitHub	Play Online
`react-testing-lib`	GitHub	Play Online
`react`	GitHub	Play Online
`ruby`	GitHub	Play Online
`solid`	GitHub	Play Online

Example	Source	Playground
`svelte`	GitHub	Play Online
`vitesse`	GitHub	Play Online
`vue-jsx`	GitHub	Play Online
`vue`	GitHub	Play Online
`vue2`	GitHub	Play Online

Projects using Vitest

- unocss
- unplugin-auto-import
- unplugin-vue-components
- vitesse
- vitesse-lite
- fluent-vue
- vueuse
- milkdown
- gridjs-svelte
- spring-easing
- bytemd
- faker
- million
- Vitamin
- neodrag
- svelte-multiselect
- iconify
- tdesign-vue-next
- cz-git

Using Unreleased Commits

If you can't wait for a new release to test the latest features, you will need to clone the vitest repo to your local machine and then build and link it yourself (pnpm is required):

```bash
git clone https://github.com/vitest-dev/vitest.git
cd vitest
pnpm install
cd packages/vitest
pnpm run build
pnpm link --global # you can use your preferred package manager for this step
```

Then go to the project where you are using Vitest and run `pnpm link --global vitest` (or the package manager that you used to link `vitest` globally).

Community

If you have questions or need help, reach out to the community at Discord and GitHub Discussions.

Suggest changes to this page

Last updated: 12/15/2022, 2:10:10 AM

Previous page
Why Vitest

Next page
Features