## Development Setup

This project uses a monorepo setup that requires using [Yarn](https://yarnpkg.com) because it relies on [Yarn workspaces](https://yarnpkg.com/blog/2017/08/02/introducing-workspaces/).

``` sh

# Install all dependencies.

yarn

# Serves VuePress' own docs with itself.

yarn dev

# Build VuePress' own docs with itself.

yarn build

# Clean dependencies.

yarn clean

# Useful when creating new a package.

yarn boot

```

## Testing Setup

VuePress leverages [jest](https://jestjs.io/) for its tests, testing process depends on some setup located at [scripts/test.js](../scripts/test.js).

```bash

# Execute all the test suites.

yarn test

# Execute tests under specfic package.

yarn test -p=core ## OR --package=core

```

## Core Packages

- \*\*docs\*\*: Docs of VuePress (do not publish to npm).

- \*\*vuepress\*\*: VuePress CLI.

- \*\*packages\*\*

- `core`: containing the Node.js API, the Plugin API, the Theme API, the Client SPA, etc.

- `markdown`: internal Markdown compiler.

- `markdown-loader`: internal Markdown loader.

- `plugin-active-header-links`: a plugin for active sidebar heading links.

- `plugin-google-analytics`: Google Analytics integration.

- `plugin-last-updated`: implementation of "last updated" feature.

- `plugin-medium-zoom`: `medium-zoom` integration.

- `plugin-nprogress`: `nprogress` integration.

- `plugin-pwa`: PWA plugin.

- `plugin-search`: search plugin, providing the `SearchBox` component.

- `shared-utils`: TypeScript utilities.

- `test-utils`: test utilities.

- `theme-default`: default theme.

- `theme-vue`: a theme tweak from default theme, used for the official Vue project.

## Core Packages not in Main Project

These projects are now available under [VuePress](https://github.com/vuepressjs) group, contribution welcome!

- [awesome-vuepress](https://github.com/vuepressjs/awesome-vuepress)

- [@vuepress/plugin-blog](https://github.com/vuepressjs/vuepress-plugin-blog)

- [@vuepress/theme-blog](https://github.com/vuepressjs/vuepress-theme-blog)

## Workflow

### Issue

Use one of the [issues templates](https://github.com/vuejs/vuepress/issues/new/choose) when you open a issue. And please ask questions on the [StackOverflow](https://stackoverflow.com/questions/ask?tags=vuepress).

We'll close your issue if you delete the template or it contains questions.

### Pull Requests

- Create a feature branch from the default branch (`master`) and merge back against that branch.

- It's OK to have multiple small commits as you work on the PR - GitHub automatically squashes them before merging.

- Make sure tests pass.

- If adding a new feature:

- Add accompanying test case(s).

- Provide a convincing reason to add this feature. Ideally, you should open a suggestion issue first and have it approved before working on it.

- If fixing bug:

- If you are resolving an open issue, add `(fix #xxxx)` (`#xxxx` being the issue ID) in your PR title for a better release log, e.g. `chore(feat): implement SSR (fix #1234)`.

- Provide a detailed description of the bug in the PR. Live demo preferred.

- Add appropriate test coverage if applicable.

### Substantial Changes

Check out [RFC flow](https://github.com/vuejs/vuepress/tree/master/rfcs) for more detail.

## Code Specification

> TODO

## Commit Specification

Commit messages should follow the [commit message convention](https://www.conventionalcommits.org) so that changelogs can be automatically generated.

Check out the availalbe types at [@commitlint/config-conventional](https://github.com/conventional-changelog/commitlint/tree/master/@commitlint/config-conventional#type-enum). And the scopes should be one of the followings:

``` sh

cli

# Core Packages/packages:

core

markdown

...

theme-vue

```

Correct examples would be: `fix($core): some message` or `feat: some message`