---

title: Contributing Guide

---

## Code of Conduct

We welcome pull requests from everyone. By participating in this project, you

agree to abide by the thoughtbot [code of conduct].

We expect everyone to follow the code of conduct anywhere in thoughtbot's

project codebases, issue trackers, chat-rooms, and mailing lists.

[code of conduct]: https://thoughtbot.com/open-source-code-of-conduct

## Getting Started

Administrate is a [Rails Engine][], but ships with everything needed to

contribute and test new changes.

To maintain compatibility with multiple dependency versions, we use

[Appraisal][].

[Rails Engine]: https://guides.rubyonrails.org/engines.html

[Appraisal]: https://github.com/thoughtbot/appraisal

### Opening a PR

1. Fork the repo,

2. Run `./bin/setup` to install the base dependencies and setup a local

database,

3. Run the test suite: `bundle exec rspec && bundle exec appraisal rspec`,

4. Make your changes,

5. Push your fork and open a pull request.

A good PR will solve the smallest problem it possibly can, have good test

coverage and (where necessary) have internationalisation support.

### Running the application locally

Administrate's demo application can be run like any Rails application:

```sh

bundle exec rails s

```

This will start the application defined in `spec/example\_app`.

## Repository Structure

\* The gem's source code lives in the `app` and `lib` subdirectories.

\* The demo app is nested within `spec/example\_app`.

\* The guides as seen at

[https://administrate-demo.herokuapp.com][docs] live as

Markdown files in the `docs` subdirectory.

Rails configuration files have been changed

to recognize the app in the new location,

so running the server or deploying to Heroku works normally.

With this structure, developing a typical feature looks like:

\* Add tests in `spec/`

\* Implement a feature in `administrate/`

\* Exercise the feature using the demo rails app (`spec/example\_app/app/`)

[demo]: https://administrate-prototype.herokuapp.com/admin

[docs]: https://administrate-prototype.herokuapp.com

## Front-end Architecture

This project uses:

\* Sass

\* [BEM]-style CSS selectors, with [namespaces]

\* Autoprefixer

\* SCSS-Lint, with [Hound] ([configuration](.scss-lint.yml))

\* A variety of CSS units:

- `em` for typographical-related elements

- `rem` for lengths related to components

- `px` for borders, text shadows, etc.

- `vw`/`vh` for lengths that should be relational to the viewport

[BEM]: http://csswizardry.com/2013/01/mindbemding-getting-your-head-round-bem-syntax/

[namespaces]: http://csswizardry.com/2015/03/more-transparent-ui-code-with-namespaces/

[Hound]: https://houndci.com/

## Labels

Issues and PRs are split into two levels of labels, at the higher level:

\* `feature`: new functionality that’s not yet implemented,

\* `bug`: breakages in functionality that is implemented,

\* `maintenance`: to keep up with changes around us

…and then to more specific themes:

\* `namespacing`: models with a namespace,

\* `installing`: initial setup, first-run experience, generators,

\* `i18n`: translations and language support,

\* `views-and-styles`: how administrate looks and is interacted with,

\* `dashboards`: how administrate presents fields and displays data,

\* `search`: finding things through our models,

\* `sorting`: ordering things on dashboards,

\* `pagination`: how we handle lots of data in small chunks,

\* `security`: controlling data access through authorisation,

\* `fields`: new fields, displaying and editing data,

\* `models`: models, associations and fetching the underlying data,

\* `documentation`: how to use Administrate, examples and common usage,

\* `dependencies`: changes or issues relating to a dependency

## Security

For security inquiries or vulnerability reports, please email

<security@thoughtbot.com>.

If you'd like, you can use our [PGP key] when reporting vulnerabilities.

[PGP key]: https://thoughtbot.com/thoughtbot.asc

## Releasing

New releases (and the time period between them) is arbitrary, but usually

motivated by a new Rails release or enough bug fixes or features that

there's significant enough changes.

A new release involves cutting and pushing a new version to [Ruby Gems][] and

then deploying that version of the example application and documentation. This

means that [the demo application][demo] always matches the current release,

whilst [the pre-release application][pre-release] can track current `master`.

[Ruby Gems]: https://rubygems.org/gems/administrate

[demo]: https://administrate-demo.herokuapp.com/

[pre-release]: https://administrate-demo-prerelease.herokuapp.com/