Contributing to Hashie

======================

Hashie is work of [many contributors](https://github.com/hashie/hashie/graphs/contributors). You're encouraged to submit [pull requests](https://github.com/hashie/hashie/pulls), [propose features and discuss issues](https://github.com/hashie/hashie/issues).

#### Fork the Project

Fork the [project on Github](https://github.com/hashie/hashie) and check out your copy.

```

git clone https://github.com/contributor/hashie.git

cd hashie

git remote add upstream https://github.com/hashie/hashie.git

```

#### Create a Topic Branch

Make sure your fork is up-to-date and create a topic branch for your feature or bug fix.

```

git checkout master

git pull upstream master

git checkout -b my-feature-branch

```

#### Install dependencies

You can use the setup script to install dependencies for the gem and its integration tests.

```

bin/setup

```

#### Test

Ensure that you can build the project and run tests.

```

bundle exec rake

```

#### Write Tests

Try to write a test that reproduces the problem you're trying to fix or describes a feature that you want to build. Add to [spec/hashie](spec/hashie).

We definitely appreciate pull requests that highlight or reproduce a problem, even without a fix.

#### Write Code

Implement your feature or bug fix.

Ruby style is enforced with [Rubocop](https://github.com/bbatsov/rubocop), run `bundle exec rubocop` and fix any style issues highlighted.

Make sure that `bundle exec rake` completes without errors.

#### Write Documentation

Document any external behavior in the [README](README.md).

#### Update Changelog

Add a line to [CHANGELOG](CHANGELOG.md) under \*Unreleased\*. Make it look like every other line, including your name and link to your Github account.

There are several categorizations of changes that you can choose from. Add your line to the appropriate section, following these conventions:

\* \*\*Added\*\* - When you add a new behavior to any class or module (or add a new extension) that does not break backwards compatibility, you should mark it as "added". This is generally a fully new behavior that does not touch any pre-existing public API. Changes here require a MINOR version bump, following the Semantic Versioning specification.

\* \*\*Changed\*\* - You should mark any change to the behavior of a public API on any class or module as "changed". Changes here require a MAJOR version bump, following the Semantic Versioning specification.

\* \*\*Deprecated\*\* - Any time you deprecate part of the public API on any class or module you should mark the change as "deprecated". Deprecated behavior will be removed in the next MAJOR version bump, but should be left in until then. Changes here require a MINOR version bump, following the Semantic Versioning specification.

\* \*\*Removed\*\* - You should mark any behavior that you removed from a public API on any class or module as "removed". Changes here require a MAJOR version bump, following the Semantic Versioning specification.

\* \*\*Fixed\*\* - Any time you fix a bug you should mark as "fixed". Changes here require a PATCH version bump.

\* \*\*Security\*\* - You should mark any security issue that you fix as "security". Changes here require a PATCH version bump.

\* \*\*Miscellaneous\*\* - Mark any other changes you make (i.e. documentation updates, test harness changes, etc.) as "miscellaneous". Changes here require a PATCH version bump.

#### Commit Changes

Make sure git knows your name and email address:

```

git config --global user.name "Your Name"

git config --global user.email "contributor@example.com"

```

Writing good commit logs is important. A commit log should describe what changed and why.

```

git add ...

git commit

```

#### Push

```

git push origin my-feature-branch

```

#### Make a Pull Request

Go to https://github.com/contributor/hashie and select your feature branch. Click the 'Pull Request' button and fill out the form. Pull requests are usually reviewed within a few days.

#### Rebase

If you've been working on a change for a while, rebase with upstream/master.

```

git fetch upstream

git rebase upstream/master

git push origin my-feature-branch -f

```

#### Update CHANGELOG Again

Update the [CHANGELOG](CHANGELOG.md) with the pull request number. A typical entry looks as follows.

```

\* [#123](https://github.com/hashie/hashie/pull/123): Reticulated splines - [@contributor](https://github.com/contributor).

```

Amend your previous commit and force push the changes.

```

git commit --amend

git push origin my-feature-branch -f

```

#### Check on Your Pull Request

Go back to your pull request after a few minutes and see whether it passed muster with Travis-CI. Everything should look green, otherwise fix issues and amend your commit as described above.

#### Be Patient

It's likely that your change will not be merged and that the nitpicky maintainers will ask you to do more, or fix seemingly benign problems. Hang on there!

#### Thank You

Please do know that we really appreciate and value your time and work. We love you, really.