\* https://www.ruby-lang.org/ - The homepage of Ruby which has more links and some great tutorials.

\* http://rubyonrails.org/ - The homepage of Rails, also has links and tutorials

## Coding style

We use [Rubocop](https://github.com/rubocop-hq/rubocop) (for ruby files)

and [ERB Lint](https://github.com/Shopify/erb-lint) (for erb templates)

to help maintain consistency in our code. You can run these utilities during

development to check that your code matches our guidelines:

```

bundle exec rubocop

bundle exec rake eslint

bundle exec erblint .

```

## Testing

Having a good suite of tests is very important to the stability and

maintainability of any code base. The tests in the Rails port code are

by no means complete, but they are extensive, and must continue to be

so with any new functionality which is written. Tests are also useful

in giving others confidence in the code you've written, and can

greatly speed up the process of merging in new code.

When hacking, you should:

\* Write new tests to cover the new functionality you've added.

\* Where appropriate, modify existing tests to reflect new or changed

functionality.

\* Never comment out or remove a test just because it doesn't pass.

You can run the existing test suite with:

```

bundle exec rake test

```

You can view test coverage statistics by browsing the `coverage` directory.

The tests are automatically run on Pull Requests and other commits with the

results shown on [Travis CI](https://travis-ci.org/openstreetmap/openstreetmap-website).

## Static Analysis

We also perform static analysis of our code. You can run the analysis yourself with:

```

bundle exec brakeman -q

```

## Comments

Sometimes it's not apparent from the code itself what it does, or,

more importantly, \*\*why\*\* it does that. Good comments help your fellow

developers to read the code and satisfy themselves that it's doing the

right thing.

When hacking, you should:

\* Comment your code - don't go overboard, but explain the bits which

might be difficult to understand what the code does, why it does it

and why it should be the way it is.

\* Check existing comments to ensure that they are not misleading.

## i18n

If you make a change that involve the locale files (in `config/locales`) then please

only submit changes to the `en.yml` file. The other files are updated via

[Translatewiki](https://translatewiki.net/wiki/Translating:OpenStreetMap) and should

not be included in your pull request.

### Nominatim prefixes

I18n keys under the `geocoder.search\_osm\_nominatim` keyspace are managed by the

Nominatim maintainers. From time to time they run stats over the Nominatim

database, and update the list of available keys manually.

Adding or removing keys to this list is therefore discouraged, but contributions

to the descriptive texts are welcome.

## Code Documentation

To generate the HTML documentation of the API/rails code, run the command

```

rake doc:app

```

## Committing

When you submit patches, the project maintainer has to read them and

understand them. This is difficult enough at the best of times, and

misunderstanding patches can lead to them being more difficult to

merge. To help with this, when submitting you should:

\* Split up large patches into smaller units of functionality.

\* Keep your commit messages relevant to the changes in each individual

unit.

When writing commit messages please try and stick to the same style as

other commits, namely:

\* A one line summary, starting with a capital and with no full stop.

\* A blank line.

\* Full description, as proper sentences with capitals and full stops.

For simple commits the one line summary is often enough and the body

of the commit message can be left out.

## Sending the patches

If you have forked on GitHub then the best way to submit your patches is to

push your changes back to GitHub and then send a "pull request" on GitHub.

Otherwise you should either push your changes to a publicly visible git repository

and send the details to the [rails-dev](https://lists.openstreetmap.org/listinfo/rails-dev)

list or generate patches with `git format-patch` and send them to the

[rails-dev](https://lists.openstreetmap.org/listinfo/rails-dev) list.