# Contributions welcome!

First, I'd like to apologize to the PHP devs for making you mess with Ruby. But it's for the best.

## Development requirements

You'll need Ruby. Ruby 1.9.x would be best. After you've got Ruby, clone the repo and install dependencies:

```sh

git clone https://github.com/bobthecow/genghis

cd genghis

git submodule update --init

gem install bundle

bundle install

```

## Building Genghis

Genghis is... a bit weird.

\* The two app files, `genghis.rb` and `genghis.php` are built via Rake. To regenerate them, run `rake build` from the

project directory.

\* If you just want to build one or the other, you can run `rake build:php` or `rake build:rb`.

\* For development, it's usually best to use non-minified assets. Run `rake build NOCOMPRESS=1` to skip minification.

\* Be sure to commit the compiled `genghis.rb` and `genghis.php` along with your source changes.

## Running the API test suite

If you're changing the API, you'll need to ensure that the API test suite passes. If you're adding a new feature or

fixing a bug, you should add corresponding tests as well.

To run the test suite, you'll need PHP 5.4+ and a Mongo instance running on `localhost`.

```sh

rspec

```

If you don't have PHP 5.4+, you can test just the Ruby API by editing `spec/spec\_helper.rb` and removing `:php`

from the backend list:

```diff

--- a/spec/spec\_helper.rb

+++ b/spec/spec\_helper.rb

@@ -5,7 +5,7 @@ require\_relative '../genghis.rb'

RSpec.configure do |config|

def genghis\_backends

- [:php, :ruby]

+ [:ruby]

end

def find\_available\_port

```

## Opening a pull request

You can do some things to increase the chance that your pull request is accepted the first time:

\* Open all pull requests against the `develop` branch.

\* Submit one pull request per fix or feature.

\* To help with that, do your work in a feature branch (e.g. `feature/my-alsome-feature`).

\* Follow the conventions you see used in the project. This is especially important when switching between languages,

since Genghis is essentially a Ruby server, an PHP server, and a JavaScript client. Make your code look like the code

around it.

\* Write API tests that fail without your code, and pass with it.

\* Don't bump version numbers. Those will be updated — per [semver](http://semver.org) — once your change is merged into

`master`.

\* Update any documentation: docblocks, README, etc.

\* ... Don't update the wiki until your change is merged and released, but make a note in your pull request so we don't

forget.