# Contributing

Hi, I'm Jason, my online nickname is "smathy" which I use on IRC, twitter,

StackOverflow, here on github, and a few other places on the internet.

Oleg is the creator of acl9, but other commitments have meant that he's had

very little time to maintain this project and so I've basically taken over as

the primary maintainer.

I like to start by introducing myself so that you know that I'm just a human

being, a normal guy, and that if you have something you want to contribute to

acl9 then I'm more than happy to hear from you.

There really aren't any hard and fast rules here for contributing. Feel free to

raise issues, you can even just ask questions in an issue if you'd like,

although IRC or StackOverflow is probably a much better forum for that. You can

ping me on twitter, or even email me at jk@handle.it

Also see the README for information on getting in contact with the rest of the

community.

## Dev Stuff

If you're going to contribute code then just fork our repo, write your thing,

and submit a pull request.

### Setup

We have a `.ruby-version` file, so if you use a ruby manager that

understands that then you might need to install that version of ruby, but you

should know how to do that yourself.

You should be able to just fork the repo and run `bundle && rake` to see the

tests running.

We use [Appraisal](//github.com/thoughtbot/appraisal) to test against multiple versions of

Rails, so you can read up on that and use it to test against all the Rails

versions we support or against a specific one.

### How to

If you're fixing a bug then please arrange your pull request in two commits, the

first one will be a test that demonstrates the bug, that test will be failing

when you create it. The second commit will be the code change that fixes the

bug.

Don't let this be a blocker for you, I'm not saying you have to do TDD. I don't

care whether you actually write the test first, or the code first, I just care

about the order of the commits. Those with experience in reviewing PRs will know

why. I can grab your PR, roll it back to `HEAD^` and run the test, seeing it

fail and confirming that your test works, then roll it back to the head of your

branch and see your code fixing the test. It makes it very easy to review a PR.

You \_can\_ submit a bugfix without a test, although those take \*\*MUCH\*\* longer to

review because it's often hard to work out what problem you're solving.

Also, it's up to you whether you want to create an issue in github first. I'd

recommend that you do because it gives a good place to discuss the details of

the issue.

Also, feel free to submit ideas as PRs, just make sure you put it clearly in the

text that this is not ready for merge yet.