Contributing to ChefSpec

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

Pull requests are merged via Github, you can find the documentation about how to fork a repository and start contributing to ChefSpec here [https://help.github.com/articles/fork-a-repo](https://help.github.com/articles/fork-a-repo).

All contributions are welcome to be submitted for review for inclusion, but before they will be accepted, we ask that you follow these simple steps:

\* [Coding standards](#coding-standards)

\* [Testing](#testing)

\* [Documentation](#documentation)

Also, please be patient as not all items will be tested or reviewed immediately by the core team.

Please be receptive and responsive to feedback about your additions or changes. The core team and/or other community members may make suggestions or ask questions about your change. This is part of the review process, and helps everyone to understand what is happening, why it is happening, and potentially optimizes your code.

If you're looking to contribute but aren't sure where to start, check out the [open issues](https://github.com/chefspec/chefspec/issues?state=open).

Will Not Merge

--------------

This section details, specifically, Pull Requests or features that will \_not\_ be merged:

1. Matchers for non-Chef core resources. ChefSpec 3.0 introduced a way for cookbook maintainers to [package matchers \_with\_ their cookbooks](https://github.com/chefspec/chefspec#packaging-custom-matchers) at distribution time.

2. New features without accompanying unit tests, cucumber tests, and documentation.

Coding Standards

----------------

The submitted code should be compatible with the standard Ruby coding guidelines. Here are some additional resources:

\* [Ruby Style Guide](https://github.com/bbatsov/ruby-style-guide)

\* [Github Styleguide](https://github.com/styleguide/ruby)

There is a tool called [Cane](https://github.com/square/cane) that allows you to validate your code's ABC complexity and documentation.

Testing

-------

Whether your pull request is a bug fix or introduces new classes or methods to the project, we kindly ask that you include tests for your changes. Even if it's just a small improvement, a test is necessary to ensure the bug is never re-introduced.

We understand that not all users submitting pull requests will be proficient with RSpec. The maintainers and community as a whole are a helpful group and can help you with writing tests. The [Better Specs](http://betterspecs.org/) site should provide some helpful resources to get you started.

ChefSpec is tested on [Travis CI](https://travis-ci.org/chefspec/chefspec) against multiple Chef Versions and Ruby Versions. \*\*Your patches must work for all Chef and Ruby Versions on Travis.\*\* This is in an effort to maintain backward compatibility as long as possible. For more information on which Chef and Ruby versions to support, checkout the [`.travis.yml`](https://github.com/chefspec/chefspec/blob/master/.travis.yml) file.

Documentation

-------------

Documentation is a crucial part to ChefSpec, especially given its broad depth of features. All documentation is placed inline on the method matcher so it can be generated with Yard. Please see existing matchers for an example and check out the [Yard documentation](http://yardoc.info)

When contributing new features, please ensure adequate documentation and examples are present.

---

This contributing guide is based off of the [Joomla Contributing Guide](https://raw.github.com/joomla/joomla-framework/master/CONTRIBUTING.markdown).