# Contributing to the Cucumber Aruba Project

## Introduction

We would love to get help from you as a \*\*user\*\* and a \*\*contributor\*\*.

### As a \*\*User\*\*

\* Tell us how "Aruba" works for you

\* Spread the word if you like our work and please tell us if something is (utterly) wrong

\* Encourage people in testing their code and keep communicating their needs

### As a \*\*Contributor\*\*

\* Send us bug fixes

\* Add new features to the code

\* Discuss changes

\* Add missing documentation

\* Improve our test coverage

The rest of this document is a guide for those maintaining Aruba, and others

who would like to submit patches.

## Contributing to the Aruba project

It would be great if all people who want to contribute to the Aruba project

&mdash; contributors and maintainers &mdash; follow the guidelines in this

section. There are also "Getting started" sections both for

[contributors](#getting-started-as-a-contributor) and

[maintainers](#getting-started-as-a-maintainer).

### Issues

About to create a new GitHub Issue? We appreciate that. But before you do,

please learn our basic rules:

\* This is not a support forum. If you have a question, please go to

[The Cukes Google Group](http://groups.google.com/group/cukes).

\* Do you have an idea for a new feature? Then don't expect it to be implemented

unless you or someone else submits a

[pull request](https://help.github.com/articles/using-pull-requests). It

might be better to start a discussion on

[the Google Group](http://groups.google.com/group/cukes).

\* Reporting a bug? Follow our comments in the Issue Template, which is

pre-filled when you create the new GitHub Issue.

\* We love [pull requests](https://help.github.com/articles/using-pull-requests).

The same here: Please consider our comments within the pull request template.

### Pull Requests

#### Contributors

Please...

\* Fork the project. Make a branch for your change.

\* Make your feature addition or bug fix &mdash; if you're unsure if your

addition will be accepted, open a GitHub Issue for discussion first

\* Make sure your patch is well covered by tests. We don't accept changes that

aren't tested.

\* Do not change the `Rakefile`, gem version number in `version.rb`, or

[`CHANGELOG.md`][].

(If you want to have your own version, that is fine but bump version in a

separate commit, so that we can ignore that commit when we merge your change.)

\* Make sure your pull request complies with our development style guidelines.

\* Rebase your branch if needed. This reduces clutter in our git history.

\* Make sure you don't break other people's code. On major changes: First

deprecate, then bump major version, then make breaking changes,

\* Split up your changes into reviewable git commits which combine all

lines/files relevant for a single change.

\* Send us a pull request.

#### Maintainers

\* Use pull requests for larger or controversial changes made by yourself or

changes you might expected to break the build.

\* Commit smaller changes directly to master, e.g. fixing typos, adding tests or

adding documentation.

\* Update [`CHANGELOG.md`][] when a pull request is merged.

\* Make sure all tests are green before merging a pull request.

### Development style guidelines

\* We try to follow the recommendations in the

[Ruby Community Style Guide][] and use

[`rubocop`][] to "enforce" it. Please see

[.rubocop.yml][] for exceptions.

\* There should be `action` methods and `getter` methods in Aruba. Only the

latter should return values. Please expect the first ones to return `nil`.

\* Add documentation &mdash; aka acceptance tests &mdash; for new features

using Aruba's Cucumber steps. Place them somewhere suitable in [features/].

\* Add unit tests where needed to cover edge cases which are not (directly)

relevant for users.

\* Add [YARD] developer documentation to all relevant methods added.

\* Format your commit messages following these seven rules &mdash; see

the ["How to Write a Git Commit Message"] blog post for a well-written

explanation about the why.

1. Separate subject from body with a blank line

2. Limit the subject line to 50 characters

3. Capitalize the subject line

4. Do not end the subject line with a period

5. Use the imperative mood in the subject line

6. Wrap the body at 72 characters

7. Use the body to explain what and why vs. how (optional if subject is self-explanatory)

## Getting started as a "Contributor"

### Requirements

To get started with Aruba, you only need [Bundler].

Install Aruba's dependencies:

```bash

bundle install

```

### Running tests

Run the following command to run the test suite.

```bash

# Run the test suite

bin/test

```

Or use these Rake tasks:

```bash

# Run the whole test suite

rake test

# Run RSpec tests

rake spec

# Run Cucumber features

rake cucumber

# Run Cucumber features which are "WORK IN PROGRESS" and are allowed to fail

rake cucumber:wip

```

If you have problems because our assumptions about your local setup are wrong,

perhaps you can use this Docker workflow. This requires [Docker] to be

installed on your local system.

```bash

# Build the docker container

bundle exec rake docker:build

# Alternative: Build with disabled cache

bundle exec rake 'docker:build[false]'

# Build image with version tag

bundle exec rake 'docker:build[false, 0.1.0]'

# Run the whole test suite in "docker"-container

RUN\_IN\_DOCKER=1 bin/test

# Run only selected scenario

RUN\_IN\_DOCKER=1 bin/test cucumber features/steps/command/shell.feature:14

```

### Installing your own gems used for development

A `Gemfile.local` file can be used, to have your own gems installed to support

your normal development workflow.

Example `Gemfile.local`:

```ruby

gem 'pry'

gem 'pry-byebug'

gem 'byebug'

```

### Running a developer console

The interactive Aruba console starts an IRB console with Aruba's API loaded:

```bash

bin/console

```

### Linting

Aruba's Rakefile provides the following linting tasks

```bash

bundle exec rake lint # Run all linters

bundle exec rake lint:coding\_guidelines # Lint our code with "rubocop"

bundle exec rake lint:licenses # Check for relevant licenses in project

```

### Building and installing your local Aruba version

You can use the following Rake tasks to build and install your work-in-progress locally:

```bash

# Build your copy

bundle exec rake build

# Build and install your copy

bundle exec rake install

# Build and install your copy without network access

bundle exec rake install:local

```

## Getting started as a "Maintainer"

### Release Process

\* Bump the version number in `lib/aruba/version.rb`

\* Make sure [`CHANGELOG.md`] is updated with the upcoming version number, and has

entries for all fixes.

\* No need to add a [`CHANGELOG.md`] header at this point - this should be done

later, when a new change is made.

Now release it:

```bash

# update dependencies

bundle update

# Run test suite

bin/test

# Release gem

git commit -m "Version bump"

bundle exec rake release

```

Now send a PR to [cucumber/website] adding an article

with details of the new release. Then merge it - an aruba maintainer should

normally be allowed to merge PRs on [cucumber/website]. A copy of an old

announcement can be used as basis for the new article.

Now, send an email with the link to the article to `cukes@googlegroups.com`.

### Gaining Release Karma

To become a release manager, create a pull request adding your name to the list

below, and include your Rubygems email address in the ticket. One of the

existing Release managers will then add you.

Current release managers:

\* Aslak Hellesøy ([@aslakhellesoy](https://github.com/aslakhellesoy))

\* Matt Wynne ([@mattwynne](https://github.com/mattwynne))

\* Matijs van Zuijlen ([@mvz](https://github.com/mvz))

To grant release karma, issue the following command:

```bash

gem owner aruba --add <NEW OWNER RUBYGEMS EMAIL>

```

[`CHANGELOG.md`]: CHANGELOG.md

[CHANGELOG.md]: CHANGELOG.md

[`rubocop`]: https://github.com/bbatsov/rubocop

[Ruby Community Style Guide]: https://github.com/bbatsov/ruby-style-guide

[.rubocop.yml]: .rubocop.yml

[features/]: features/

[Bundler]: https://bundler.io/

[Docker]: https://docs.docker.com/

[YARD]: http://yardoc.org/

["How to Write a Git Commit Message"]: http://chris.beams.io/posts/git-commit/

[cucumber/website]: https://github.com/cucumber/website