# Contributing to Restify

Welcome to the restify community! This document is written both for maintainers and community members!

## Issues and PRs

### Commit Messages

When merging a PR, we squash and merge to keep our commit history clean. Our commit messages use the conventional changelog format (http://conventionalcommits.org/) to automagically manage semver for us.

### Labels and Templates

We try to keep things organized around here. Maintainers have a finite amount of time and are often juggling multiple things in their lives. Keeping things consistent and well labeled helps reduce the amount of concentration and effort required for us to both find and carry out work on the project. Simple things like using our templates and adding the appropriate labels may only take you a few minutes, but it can save cummulative hours worth of work for maintainers trying to digest dozens of issues.

## Website

### Design

The website templates are maintained at https://github.com/restify/restify.github.io and are populated from the docs directory in this repo.

### Releasing a change

To update the documentaiton on the website to reflect the latest version of 5.x simply:

```

git clone --recursive git@github.com:restify/restify.github.io

cd restify.github.io

git submodule update --remote && git add \_docs && git commit -m 'bump' && git push origin master

```

The website will automatically deploy itself with the new changes.

### Updating a documentation page

To update docs, simply run:

```

make docs-build

```

### Adding a documentation page

To add a new page, simply give it a [permalink](https://github.com/restify/node-restify/blob/94fe715173ffcebd8814bed7e17a22a24fac4ae8/docs/index.md) and then update [docs.yml](https://github.com/restify/restify.github.io/blob/master/\_data/docs.yml) with the new permalink.

## Running a benchmark

```

make benchmark

```

## Cutting a release

Cutting a release is currently a manual process. We use a [Conventional Changelog](http://conventionalcommits.org/) to simplify the process of managing semver on this project. Generally, the following series of commands will cut a release from the `master` branch:

```

$ git fetch

$ git pull origin master # ensure you have the latest changes

$ npx unleash [-p for patch, -m for minor, -M for major] --no-publish -d # do a dry run to verify

$ npx unleash [-p for patch, -m for minor, -M for major] --no-publish

# Unleash doesnt support 2FA, hence we use --no-publish flag here.

# This ensures we have the package.json updated, changelog generated, tag created

# and all the changes into origin

# Next, publish to npm manually and do not forget to provide the 2FA code.

$ npm publish

```