RuboCop // Docs

RuboCop

> *Role models are important.*
>
> — Officer Alex J. Murphy / RoboCop

Overview

RuboCop is a Ruby static code analyzer (a.k.a. linter) and code formatter. Out of the box it will enforce many of the guidelines outlined in the community Ruby Style Guide.

RuboCop packs a lot of features on top of what you'd normally expect from a linter:

- Works with every major Ruby implementation
- Autocorrection of many of the code offenses it detects
- Robust code formatting capabilities
- Multiple result formatters for both interactive use and for feeding data into other tools
- Ability to have different configuration for different parts of your codebase
- Ability to disable certain cops only for specific files or parts of files
- Extremely flexible configuration that allows you to adapt RuboCop to pretty much every style and preference
- It's easy to extend RuboCop with custom cops and formatters
- A vast number of ready-made extensions (e.g. `rubocop-rails`, `rubocop-rspec`, `rubocop-performance` and `rubocop-minitest`)
- Wide editor/IDE support
- Many online services use RuboCop internally (e.g. HoundCI and CodeClimate)
- Best logo/stickers ever

The project is closely tied to several efforts to document and promote the best practices of the Ruby community:

- Ruby Style Guide
- Rails Style Guide
- RSpec Style Guide

- Minitest Style Guide

A long-term goal of RuboCop (and its core extensions) is to cover with cops all the guidelines from the community style guides.

Philosophy

Early on RuboCop aimed to be an opinionated linter/formatter that adhered very closely to the Ruby Style Guide (think gofmt and the like). In those days cops supported just a single style and you couldn't even turn individual cops off. Eventually, we realized that in the Ruby community there were so many competing styles and preferences that it was going to be really challenging to find one set of defaults that makes everyone happy. Part of this was Ruby's own culture and philosophy, part was the lack of common standards for almost 20 years. It's hard to undo any of those, but it's also not really necessary.

The early feedback we got led us to adopt a philosophy of (extreme) configurability and flexibility, and trying to account for every *common* style of programming in Ruby. While we still believe that there's a lot of merit to just sticking to the community style guides, we acknowledge that Ruby is all about diversity and doing things the way that makes you happy. Whatever style preferences you have RuboCop is there for you. That's our promises and our guarantee. Within the subjective limits of sanity that is.

Next Steps

So, what to do next? While you can peruse the documentation in whatever way you'd like, here are a few recommendations:

- See "Basic Usage" to get yourself familiar with RuboCop's capabilities.
- Adjust RuboCop to your style/preferences. RuboCop is an extremely flexible tool and most aspects of its behavior can be tweaked via various configuration options. See "Configuration" for more details.
- See "Versioning" for information about RuboCop versioning, updates, and the process of introducing new cops.
- Explore the existing extensions.