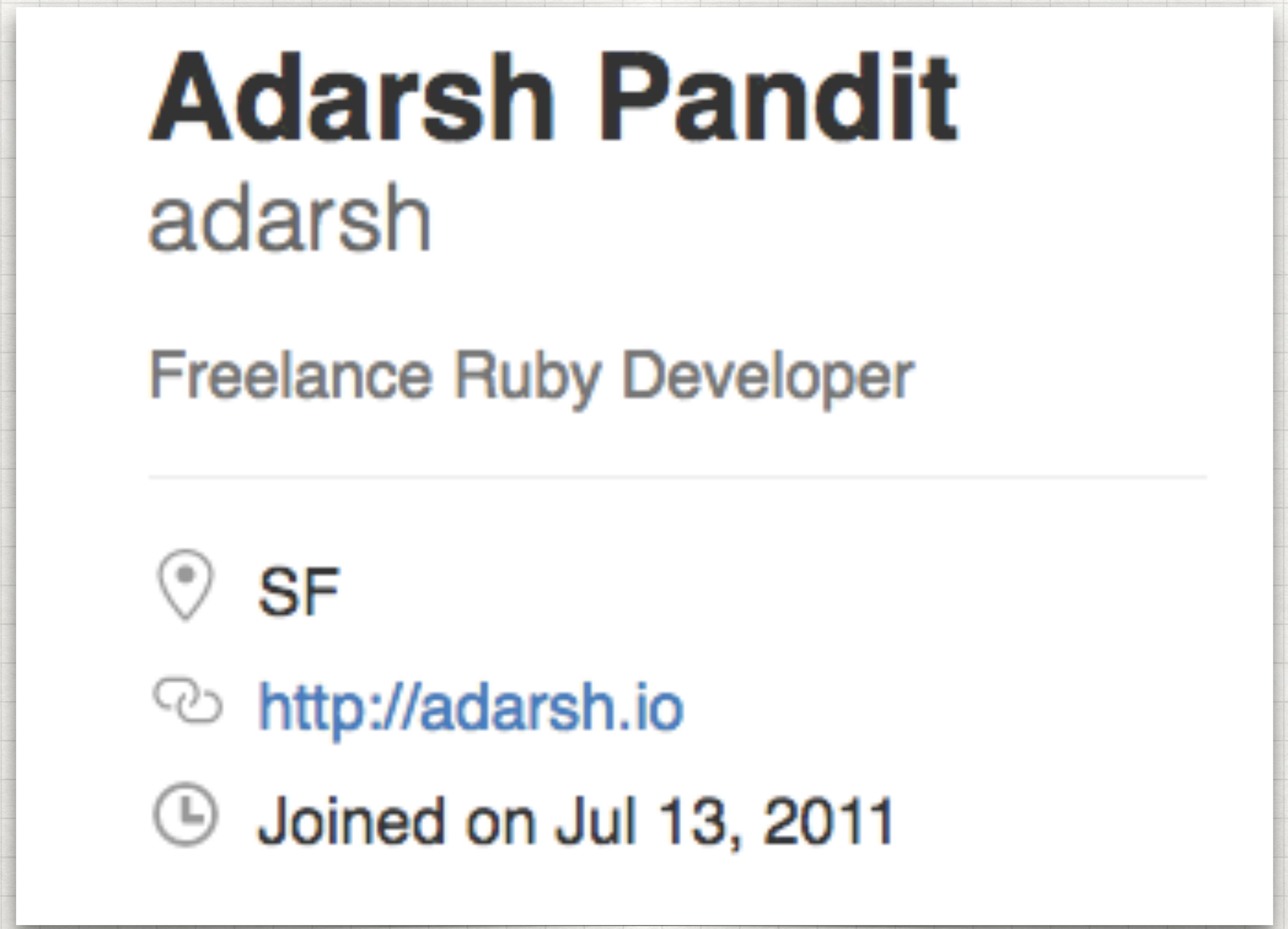
How To Make Your Codebase A Joyful Place

ADARSH PANDIT

twitter.com/adarshp github.com/adarsh

AND YOU ARE?


Adarsh Pandit
adarsh
Freelance Ruby Developer
SF
http://adarsh.io
Joined on Jul 13, 2011

“I'M NOT A GREAT PROGRAMMER; I'M JUST A GOOD PROGRAMMER WITH GREAT HABITS”

— *Kent Beck in “Refactoring: Improving the Design of Existing Code” Martin Fowler, Kent Beck, John Brant (2012)*

TODAY'S GOALS

1. CONVINCE YOU THAT BEST PRACTICES ARE IMPORTANT FOR YOUR BUSINESS
2. SHARE TACTICS FOR AGREEING ON SOME BEST PRACTICES
3. LIST HOW TO AUTOMATE SOME BEST PRACTICES
4. POLARIZE THE CROWD WITH MY OWN STYLE GUIDE OPINIONS

TRAILING CONDITIONALS

YES

```
1 class SomeClassName
2   def hello
3     if person.friendly?
4       puts "Hello!"
5     end
6   end
7 end
```

NO

```
1 class SomeClassName
2   def hello
3     puts "Hello!" if person.friendly?
4   end
5 end
```

1. CONVINCE YOU THAT BEST PRACTICES ARE IMPORTANT FOR YOUR BUSINESS
2. SHARE TACTICS FOR AGREEING ON SOME BEST PRACTICES
3. LIST HOW TO AUTOMATE SOME BEST PRACTICES
4. POLARIZE THE CROWD WITH MY OWN STYLE GUIDE OPINIONS

BEST PRACTICES: WHY DO THEY MATTER?

BECAUSE MONEY

```
developer.happy  #=> budget.recruiting -= lots

developer.fast   #=> developer.roi += lots

bugs.destroy_all #=> customer.happy
```

1. CONVINCE YOU THAT BEST PRACTICES ARE IMPORTANT FOR YOUR BUSINESS
2. SHARE TACTICS FOR AGREEING ON SOME BEST PRACTICES
3. LIST HOW TO AUTOMATE SOME BEST PRACTICES
4. POLARIZE THE CROWD WITH MY OWN STYLE GUIDE OPINIONS

LET'S AGREE AS A TEAM: BEST PRACTICES ARE IMPORTANT

BECAUSE

- We want to engender a learning culture of continuous improvement
- We want to be proud of our codebase and be excited to go to work each day
- Argue less about dumb stuff
- Routine tasks should be automated

ALWAYS KEYWORD ARGUMENTS

YES

```
1 class SomeClassName
2   def hello(person:)
3     if person.friendly?
4       puts "Hello!"
5     end
6   end
7 end
```

NO

```
1 class SomeClassName
2   def hello(person)
3     if person.friendly?
4       puts "Hello!"
5     end
6   end
7 end
```

HOW WILL WE DISCUSS BEST PRACTICES?

STARTING OUT

- Determine which best practices are valuable for us. It's not all of them.
- Openly talk about why this is hard:
  feeling smart! feeling dumb! history! business! individuality! Ruby!
- Draw a picture of where we want to go as an organization
- Get leadership buy-in, which may mean adopting their style (to start)
- Adopt an experimental, "let's try it out!" attitude

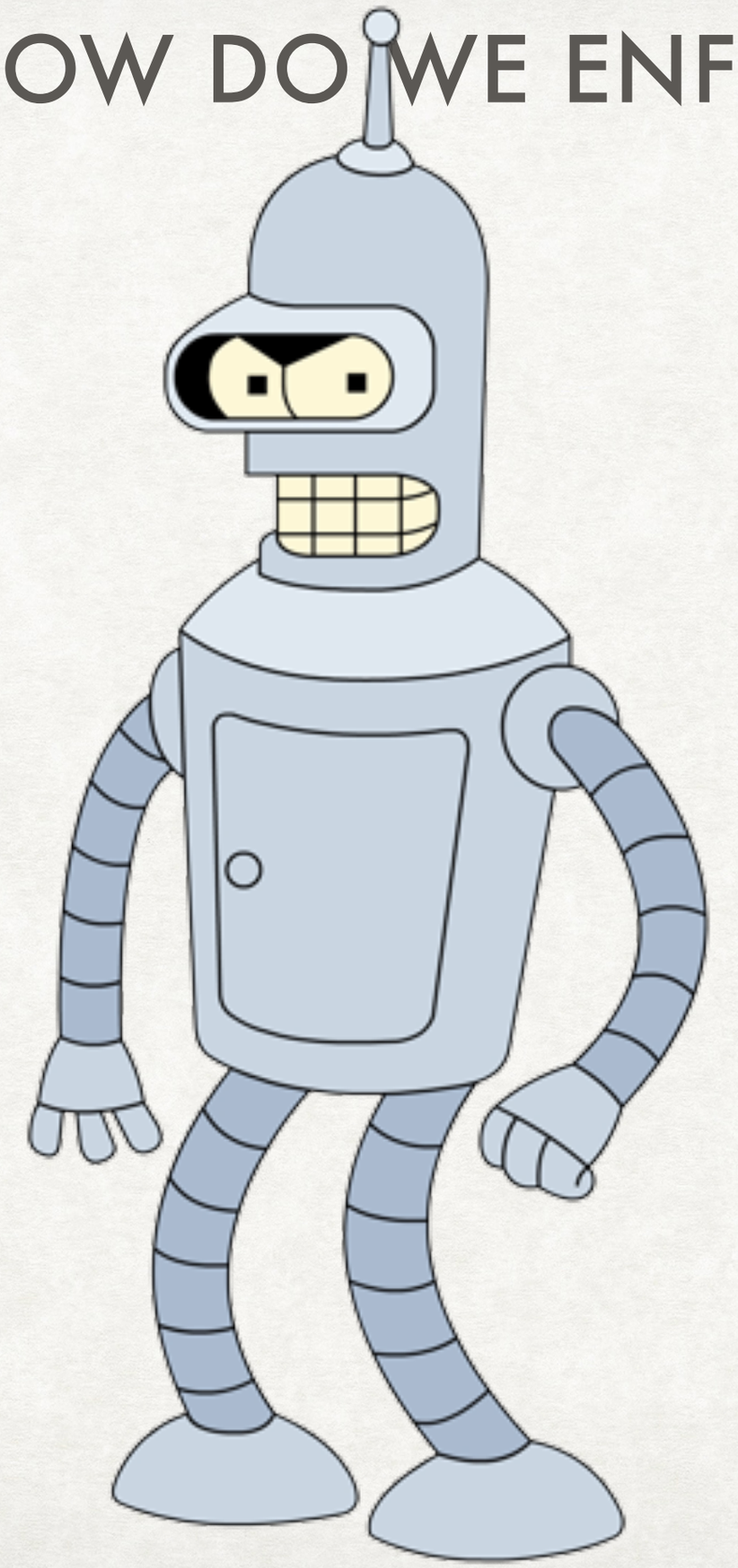
HOW WILL WE DISCUSS BEST PRACTICES?

ONGOING

- Write it in Markdown, under version control.
- PRs are discussions because we are developers.
- Agree on a democratic approach but have a tiebreaker
- Start with one thing we already agree on
- Follow up with one thing no one agrees on

1. CONVINCE YOU THAT BEST PRACTICES ARE IMPORTANT FOR YOUR BUSINESS
2. SHARE TACTICS FOR AGREEING ON SOME BEST PRACTICES
3. LIST HOW TO AUTOMATE SOME BEST PRACTICES
4. POLARIZE THE CROWD WITH MY OWN STYLE GUIDE OPINIONS

OK GREAT: HOW DO WE ENFORCE THIS?



```
59      -  @include span-columns(6 of 10);
        59 +  @include span-columns(6 of 10);
```

[redacted] added a note 9 days ago

I don't understand this change, I think there is an extra space after the semicolon.

[redacted] added a note 8 days ago

Totally true. I guess it was my editor. It is supposed to clean whitespace but maybe I had the cursor there.

PREAMBLE: GIT HOOKS

brigade / **overcommit**

Watch 75 | Unstar 1,311 | Fork 94

Code | Issues 18 | Pull requests 0 | Pulse | Graphs

A fully configurable and extendable Git hook manager

1,233 commits | **2** branches | **59** releases | **42** contributors

Branch: **master** | **New pull request** | **Create new file** | **Upload files** | **Find file** | **Clone or download**

sds Upgrade RuboCop 0.39.0 -> 0.40.0 for Travis — Latest commit 9636eb3 12 days ago

bin	Auto-load Bundler context for CLI	2 months ago
config	Add *.ru matcher to rubocop hook files list	13 days ago
lib	Upgrade RuboCop 0.39.0 -> 0.40.0 for Travis	12 days ago
libexec	Gracefully handle .git submodule files	2 years ago
logo	Add Overcommit logo to README with attribution	4 months ago
spec	Upgrade RuboCop 0.39.0 -> 0.40.0 for Travis	12 days ago
template-dir/hooks	Add global quiet option for silencing hook runs	2 months ago
.editorconfig	Add .editorconfig file	2 years ago
.gitignore	Add code coverage integration with Coveralls	a year ago

PREAMBLE: GIT HOOKS

PREAMBLE: GIT HOOKS

www.guoxiang.me/posts/28-how-to-write-a-custom-overcommit-precommit-git-hook-in-4-steps

Guo Xiang

I built RubyBench.org

How to Write a Custom Overcommit PreCommit Git Hook in 4 Steps

By Guo Xiang on 16 AUG 2014

This post summarizes the first ever Ruby talk I gave at a RubySG meetup

"Follow the instructions here to install Overcommit before reading the tutorial"

Despite Overcommit being an opinionated git hook manager, it is fully configurable and extendable as well. In this tutorial, I will walk you through on how you can write a pre-commit hook (with tests) that enforces a maximum length for each line in your files through the use of the APIs provided by the Overcommit gem.

Step 1. Putting the files in the right directory

By default, Overcommit looks for your custom hooks in the `.git_hooks` folder which we can configure

PRIVATE ATTR_READERS

YES

```ruby
class SomeClassName
  def initialize(person:)
    @person = person
  end

  def hello
    puts "Hello, #{person.name}!"
  end

  private

  attr_reader :person
end
```

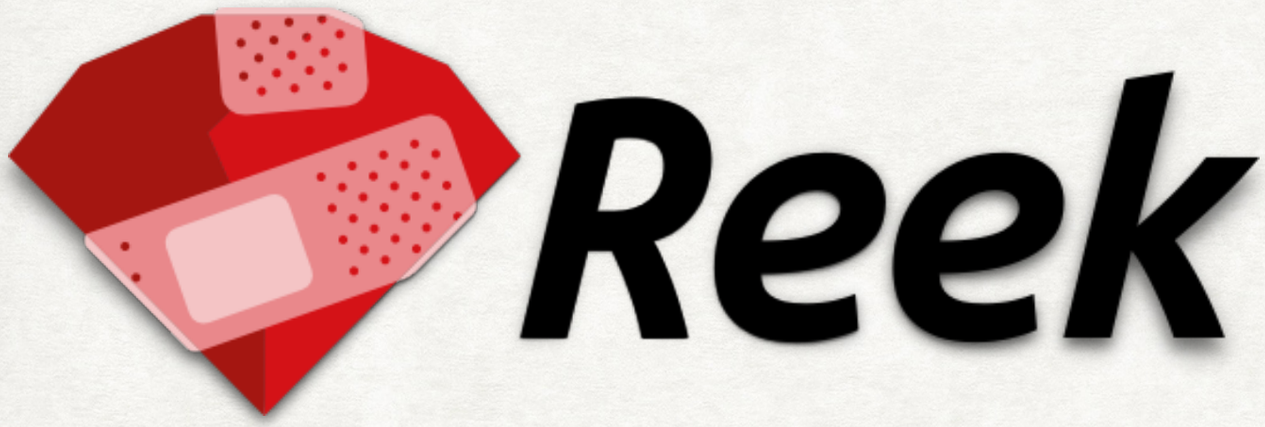
NO

```ruby
class SomeClassName
  def initialize(person:)
    @person = person
  end

  def hello
    puts "Hello, #{@person.name}!"
  end
end
```

CODE QUALITY

OPEN SOURCE TOOLS



seattlerb / flay

seattlerb / flog

square / cane

Code | Issues 6 | Pull requests 0 | Pulse

Code quality threshold checking as part of your build

SandiMeter

build passing | gem version 1.2.0

Static analysis tool for checking your Ruby code for Sandi Metz' four rules.

- 100 lines per class
- 5 lines per method
- 4 params per method call (and don't even try cheating with hash params)
- 1 instance variable per controller action

CODE QUALITY

OPEN SOURCE TOOLS

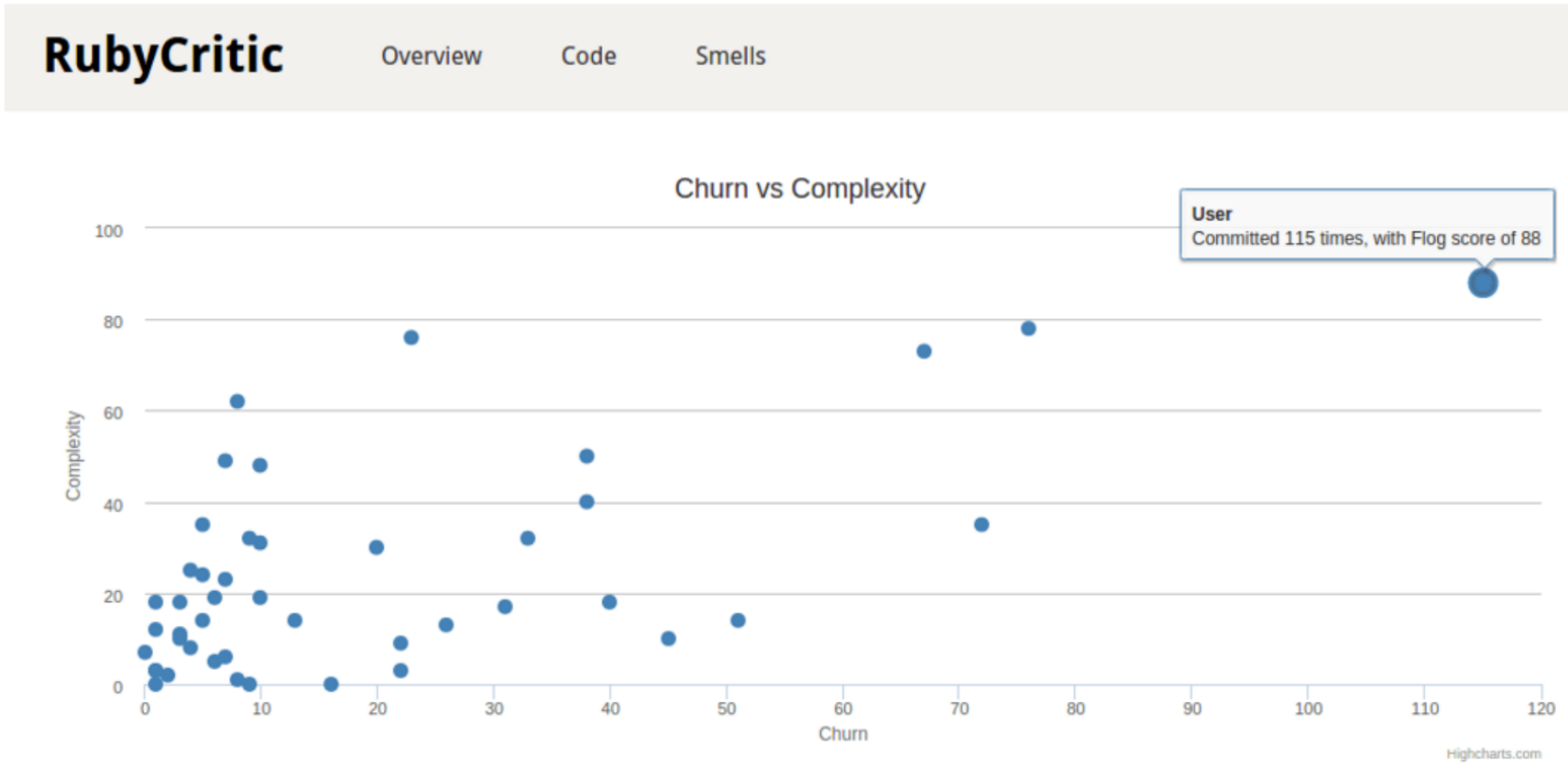
RubyCritic

gem version 2.9.1 | build passing | code climate 4.0

RubyCritic is a gem that wraps around static analysis gems such as Reek, Flay and Flog to provide a quality report of your Ruby code.

This gem provides features such as:

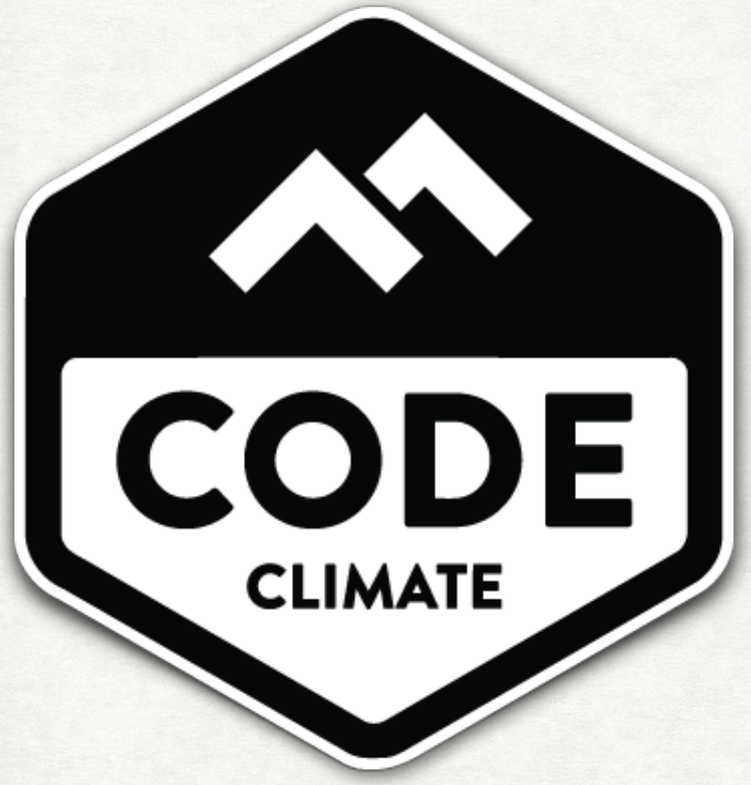
1. An overview of your project:

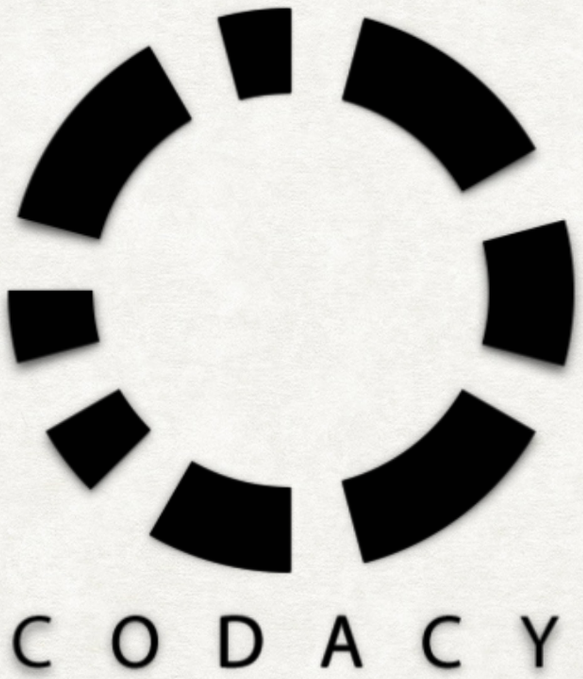


2. An index of the project files with their respective number of smells:

CODE QUALITY

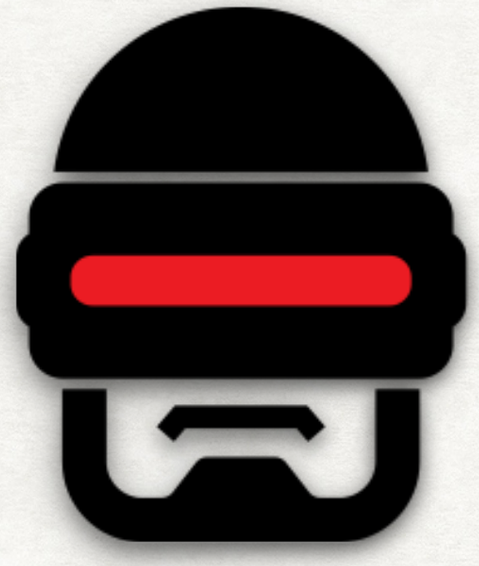
PAID SERVICES





CODE STYLE

OPEN SOURCE TOOLS



RuboCop

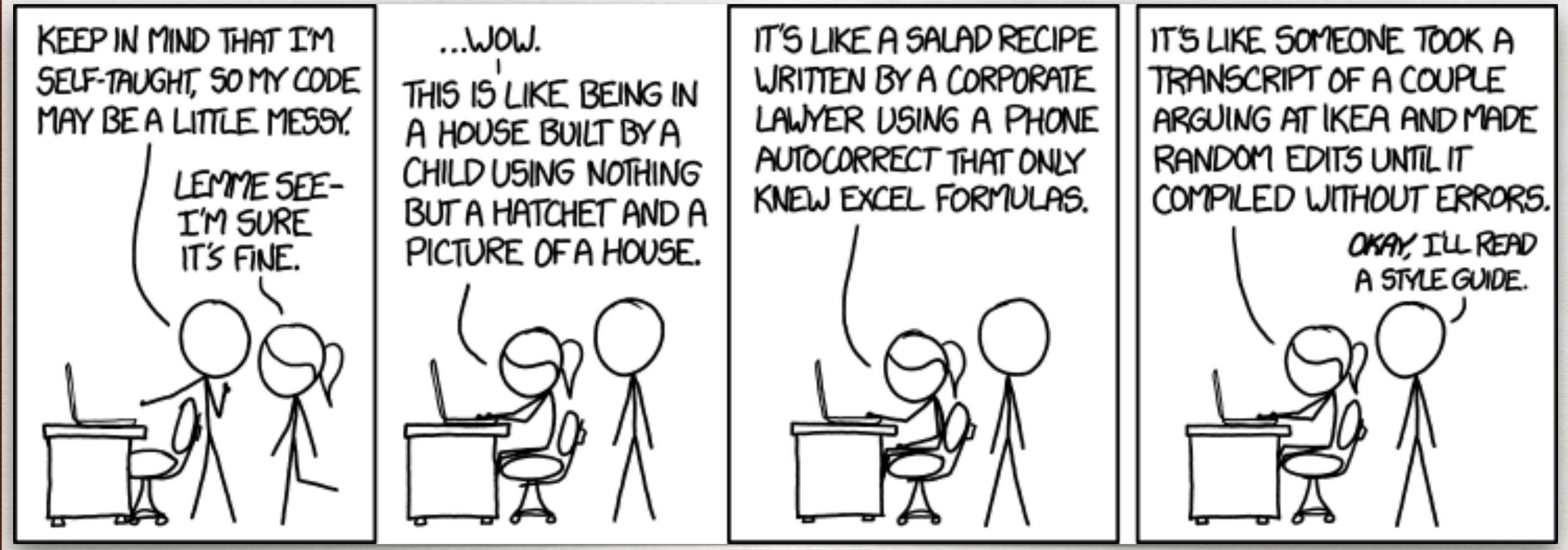
bbatsov / **ruby-style-guide**

Code | Issues 64 | Pull requests 37

A community-driven Ruby coding style guide

CODE STYLE

PAID SERVICES



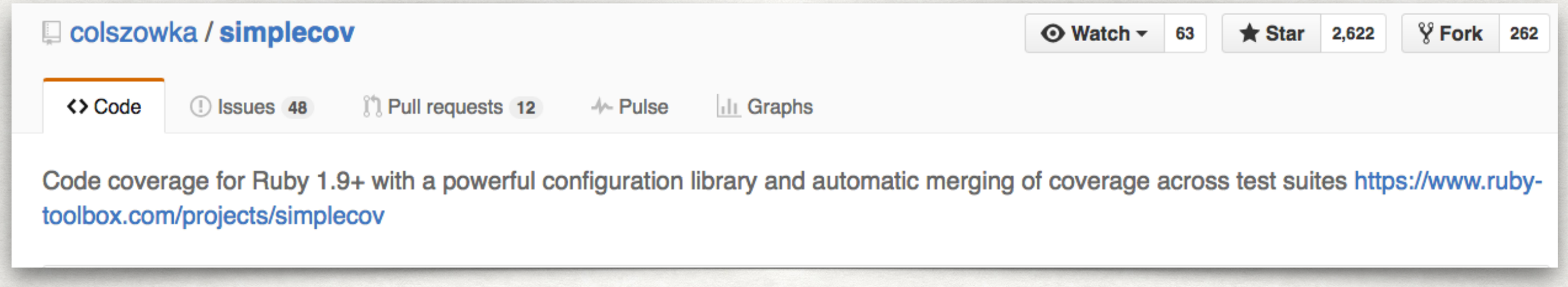
HOUND

CODE CLIMATE

CODACY

TEST COVERAGE

OPEN SOURCE TOOLS



TEST COVERAGE

PAID SERVICES

Travis CI

circleci

Jenkins

SECURITY PATCHES

civisanalytics / **ruby_audit**

Code | Issues 0 | Pull requests 0 | Wiki | Pulse

Checks Ruby and RubyGems against known security vulnerabilities.

rubysec / **bundler-audit**

Code | Issues 11 | Pull req

Patch-level verification for Bundler

SECURITY PATCHES

PAID SERVICES

Brakeman Pro

Brakeman Pro Version 1.2.1 is here!

Brakeman Pro is a static analysis security tool (SAST) for Ruby on Rails applications. It searches for potential security vulnerabilities by scanning the source code of Rails applications. Brakeman Pro provides an interface for reviewing, managing and reporting on warnings across multiple scans and applications.

Appcanary

UPDATING YOUR GEMS

← Adarsh Pandit →

Software Developer

Save Money and Be Happier by Updating Your Gems Every Monday Morning

Nov 16, 2015 *By Adarsh Pandit in writing*

As a newcomer to Rails, understanding Gems and the Gemfile is fairly straightforward: You include a gem that you need and you get it included everywhere.

	Write (editors, CLIs)	Commit (git hooks)	Submit (CI services)
Ruby (Quality)	RubyCritic, Flog/Flay/Reek, rails_best_practices	Overcommit + Integrations	Code Climate, Codacy
Ruby (Style)	RuboCop		
SCSS	SCSS-lint		Hound, Code Climate, Codacy
JavaScript	js-hint		
Testing	SimpleCov	CI (Circle, Travis)	CI (Circle, Travis)
Security	brakeman, bundler-audit	CI (Circle, Travis)	CI (Circle, Travis)

THANKS!

HTTP://SPKR8.COM/T/67231

"PROGRAMS ARE MEANT TO BE READ BY HUMANS AND ONLY INCIDENTALLY FOR COMPUTERS TO EXECUTE."

— H. Abelson and G. Sussman
(in "Structure and Interpretation of Computer Programs")