Gradually typing your Rails application The easy way

Ryan Brushett & Alexandre Terrasa

Get these slides

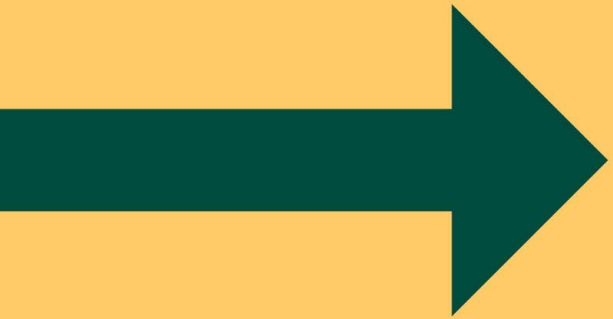
https://github.com/Shopify/rubygems.org/blob/master/slides.pdf

Ryan Brushett
@RyanBrushett

Alexandre Terrasa
@Morriar

Table of Contents

Gradual typing with **Sorbet**



Sorbet

- A fast, powerful type checker for Ruby
- Allow gradual typing
- Both static and runtime type checking
- Syntax 100% Ruby
- Editor integration through LSP
- Developed by Stripe

More details at
https://sorbet.org/

Running static type checking

```
$ bundle exec srb tc
No errors! Great job.
```

What is Gradual Typing?

- A type system where typed and untyped variables can coexist
- Types can be adopted incrementally
- Turning type checking
  - Within an entire file
  - For a particular method
  - For a single argument of a method
  - For a specific call site
- Splitting our work into small and easy to review bits
- Getting benefits without changing our application too much
- Typing an existing app that already runs in production

Why Sorbet at Shopify?

The code base is huge and evolves very fast

- Tens of thousands of files, millions on lines of code
- Thousands of engineers, hundreds of merges each day

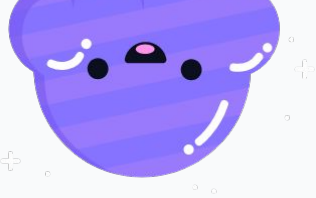
Sorbet unlocks amazing benefits

- Gradual typing
- Fast type checking (~10s for 45,000 files)
- VS Code integration through LSP
- More safety, rigor, and peace of mind

Why not RBS at Shopify?

- RBS didn’t exist at the time
  - We adopted types around Ruby 2.6
  - RBS appeared in 3.0
- RBS isn’t a tool, it’s a language for type annotations
  - No out-of-the-box experience
  - Steep and TypeProf are not fast enough for us
- RBS isn’t valid Ruby code
  - We need to reimplement a lot of tools (editing, linting, ...)

In any case, it’s easier to go from one type system to another rather than from not having types to having types



Per-file type strictness

A sigil is a magic comment at the top of a Ruby file used to set the type strictness

`typed: ignore`

`typed: false` (default)

`typed: true`

`typed: strict`

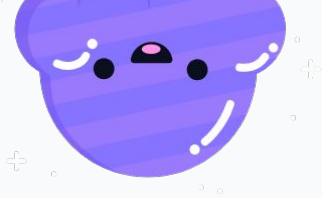
`typed: strong`

foo.rb

```ruby
# typed: true

class Foo; end
```

bar.rb

```ruby
# File strictness is
# typed: false by
# default

class Bar; end
```



`# typed: ignore`

Sorbet does not even read the file and no errors are reported at all

`typed: ignore`

=

`typed: debt`

foo.rb

```ruby
# typed: ignore

class Foo
end
```

bar.rb

```ruby
# typed: false

class Bar < Foo
end
```

Avoid `typed: ignore`!

```
$ bundle exec srb tc
bar.rb:3: Unable to resolve constant Foo
     3 | class Bar < Foo
                   ^^^
```

typed: false

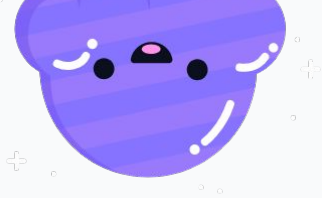
Sorbet reports syntax errors and unresolved constants
(default strictness for files without a sigil)

foo.rb

```
# typed: false

class Foo; end

Faa.new
Foo.new(,)
Foo.faa
```

Type checking at `typed: false`

```
$ bundle exec srb tc
foo.rb:5: Unable to resolve constant Faa
     5 | Faa.new
         ^^^
foo.rb:6: unexpected token ","
     6 | Foo.new(,)
                 ^
```



`# typed: true`

Errors for calling a non-existent method, calling a method with mismatched argument counts, and using variables inconsistently with their types

foo.rb

```ruby
# typed: true
extend T::Sig

sig { params(x: Integer).void }
def foo(x)
 puts(x + 1)
end

ffoo(0)
foo("not an int")
```

Type checking at `typed: true`

```
$ bundle exec srb tc
foo.rb:9: Method ffoo does not exist on
T.class_of(<root>)
     9 | ffoo(0)
         ^^^^

foo.rb:10: Expected Integer but found
String("not an int") for argument x
    10 | foo("not an int")
              ^^^^^^^^^^^^
```

typed: strict

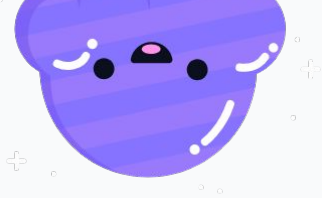
All methods must have a signature, attributes and constants must be typed

foo.rb

```ruby
# typed: strict

class Logger
 LOGS_DIR = Dir.tmpdir

 def initialize(ext)
  date = Date.today
  @file = File.join(
   LOGS_DIR,
   "#{date}.#{ext}"
  )
 end
end
```

Type checking at typed: strict

```
$ bundle exec srb tc
foo.rb:4: Constants must have type
annotations with T.let
     4 |  LOGS_DIR = Dir.tmpdir
                      ^^^^^^^^^^

foo.rb:6: This function does not have a sig
     6 |  def initialize(ext = "log")
          ^^^^^^^^^^^^^^^^^^^^^^^^^^^^

foo.rb:8: Use of undeclared variable @file
     8 |    @file = File.join(LOGS_DIR, …
          ^^^^^
```



typed: strong

Everything must have a type. This is buggy and pretty much unusable

foo.rb

```ruby
# typed: strong

class MyError < StandardError
 extend T::Sig

 sig { params(msg: String).void }
 def initialize(msg)
   super
 end
end
```

Avoid typed: strong

```
$ bundle exec srb tc
foo.rb:8: This code is untyped
     8 |    super
          ^^^^^

 Note:
  Support for typed: strong is minimal.
  Consider using typed: strict instead.
```

Tools of the trade

sorbet-static

Static type checking component of Sorbet

Gemfile

```
gem "sorbet", :group => :development
```

Running static type checking

```
$ bundle install
$ bundle exec srb tc
No errors! Great job.
```

More details at https://sorbet.org/docs/static

sorbet-runtime

Enables adding type annotations and sigs to Ruby code
Dynamically type checks the code while it runs

Gemfile

```
gem "sorbet-runtime"
```

my_app.rb

```
require "sorbet-runtime"
```

Running runtime type checking

```
$ bundle install
$ bundle exec ruby my_app.rb
```

More details at https://sorbet.org/docs/runtime

Tapioca

The swiss army knife of RBI generation

- Generate RBI files for gems & DSLs
- Configurable with custom RBI generators
- Dramatically eases the maintenance of RBI files

Useful tapioca commands

```
$ bundle exec tapioca init
$ bundle exec tapioca gems
$ bundle exec tapioca dsl
$ bundle exec tapioca check-shims
```

More details at https://github.com/Shopify/tapioca

Spoom

Useful tools for Sorbet enthusiasts

- Declutters Sorbet's output to make it more readable
- Generates coverage reports with lots of useful data
- Provides handy automation like bumping files' strictness
- Provide LSP integration with Sorbet

Useful spoom commands

```
$ bundle exec spoom tc
$ bundle exec spoom bump
$ bundle exec spoom coverage
```

More details at https://github.com/Shopify/spoom

Rubocop-Sorbet

Rules for Sorbet adoption

- Collection of Rubocop rules for projects that use Sorbet
- Customisable, based on how strictly your project is typed
- Automating bulk tasks during adoption or migrations
- Invaluable for maintaining types in fast-moving projects

Running style checking

```
$ bundle exec rubocop
```

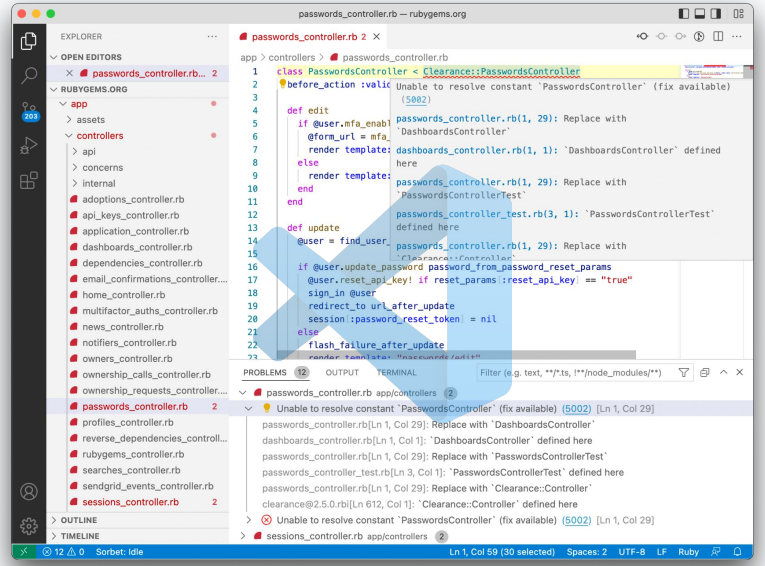
More details at https://github.com/Shopify/rubocop-sorbet

VSCode

Comfortable and configurable code editing

- Open-source
- Runs on many platforms
- Great support for custom extensions
- Sorbet extension for VS Code

  https://marketplace.visualstudio.com/items?itemName=sorbet.sorbet-vscode-extension

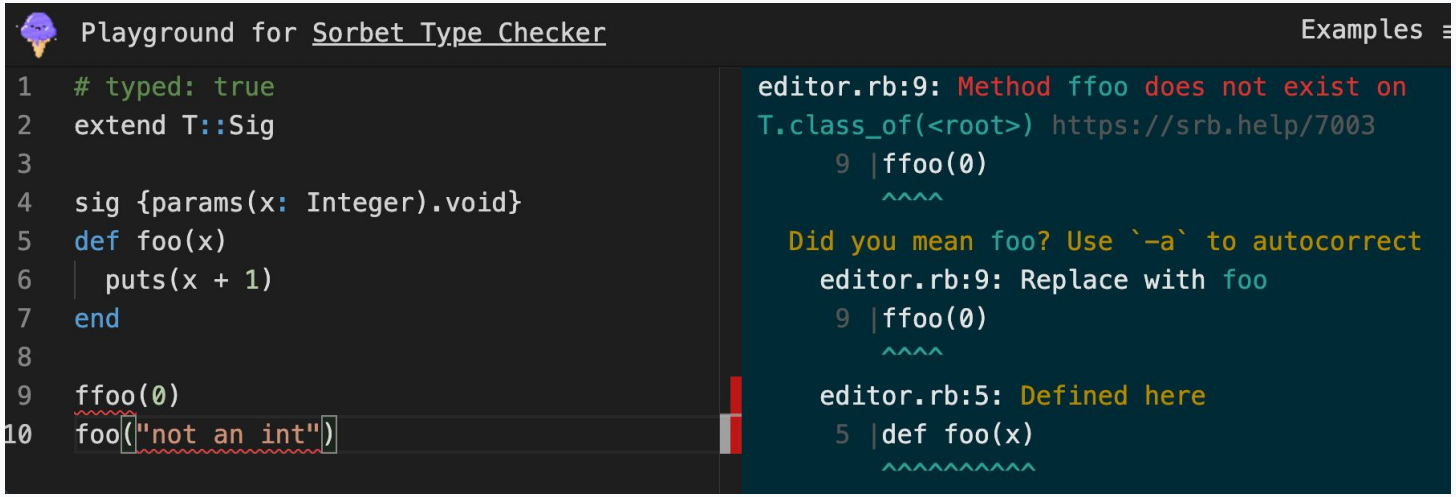


More details at https://code.visualstudio.com/

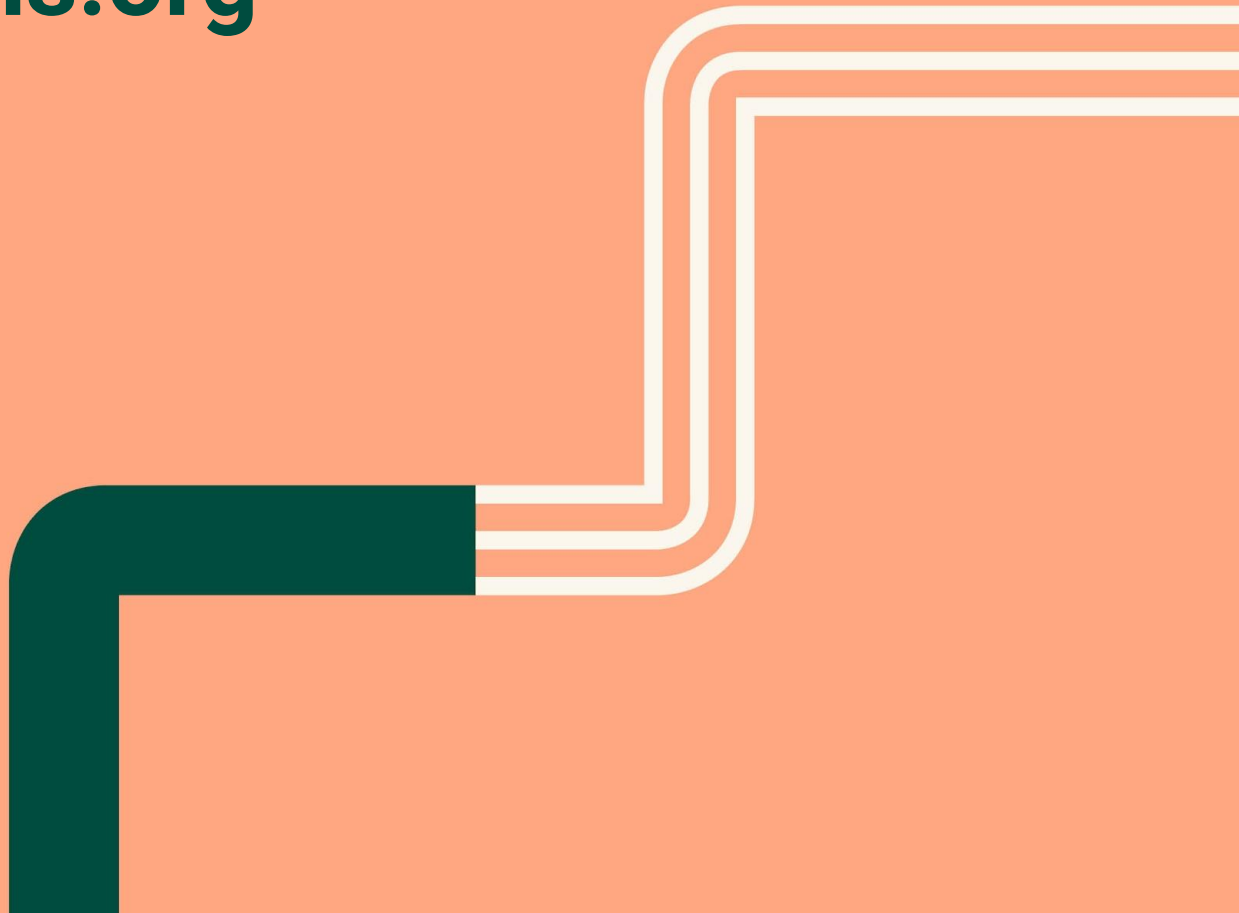
sorbet.run

Sorbet playground in your web browser

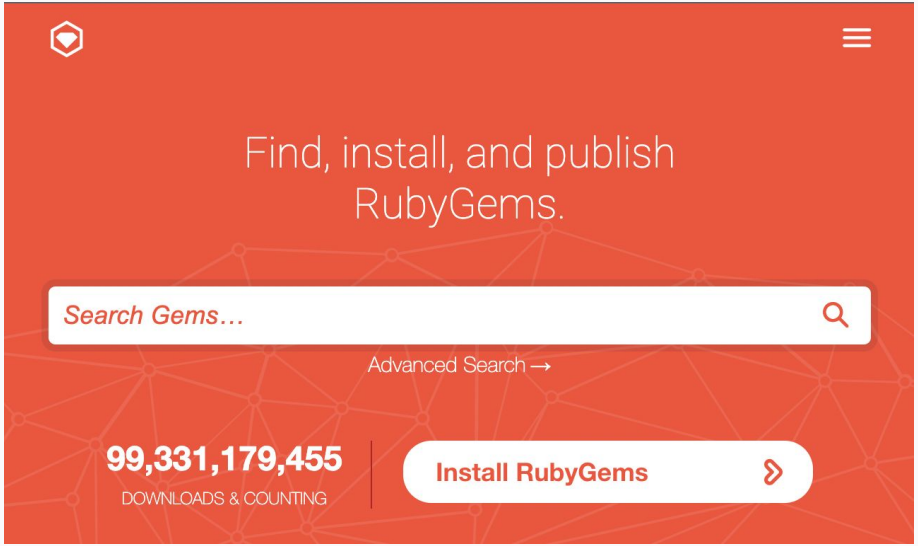
- Comes with nice examples!
- Nice way to share demonstrations of a feature
- Can directly open bug reports from code snippets



Typing rubygems.org



Why Rubygems.org?



- Real-world example
- Open source
- Sizable Rails app
- Up to date
- Not typed yet
- Widely known

Please don’t y’all go and open pull-requests to add types to rubygems.org!

Installing the dependencies

```
$ chruby 3.1.2
$ gem install bundler
Successfully installed bundler-2.3.13
$ brew install postgresql
==> Pouring postgresql--14.2_1.bottle.tar.gz
$ brew install --cask docker
==> Installing Cask Docker
🍺 docker was successfully installed
```

More details at CONTRIBUTING.md#development-setup

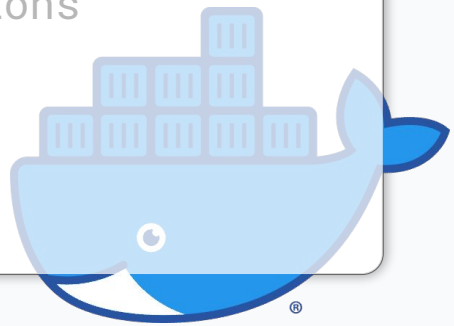
Getting the code

```
$ git clone https://github.com/Shopify/rubygems.org
Cloning into 'rubygems.org'...
remote: Enumerating objects: 51242, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 51242, reused 22, pack-reused 51206
Receiving objects: 100% (51242/51242), done.
Resolving deltas: 100% (35763/35763), done.
$ cd rubygems.org/
$ bundle
Bundle complete!
67 Gemfile dependencies, 192 gems now installed.
```

More details at CONTRIBUTING.md#getting-the-code

Setting up the development infrastructure

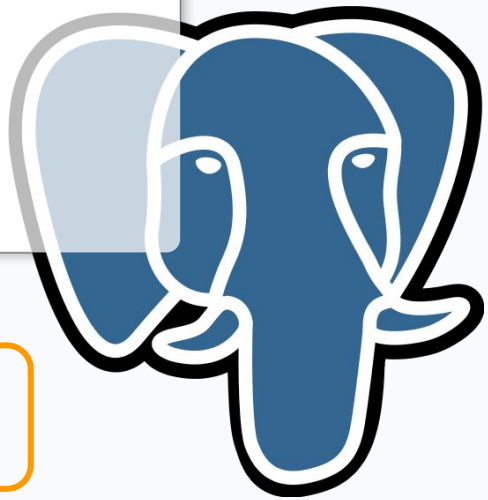
```
$ docker-compose up
Starting rubygemsorg_cache_1     ... done
Starting rubygemsorg_db_1        ... done
Starting rubygemsorg_search_1    ... done
Starting rubygemsorg_toxiproxy_1 ... done
Attaching to rubygemsorg_toxiproxy_1, rubygemsorg_cache_1,
rubygemsorg_search_1, rubygemsorg_db_1
...
... LOG:  database system is ready to accept connections
...
```



More details at [CONTRIBUTING.md#setting-up-the-database]

Setting up the database

```
$ ./script/setup
./script/setup: Cleaning out old logs
./script/setup: Installing libraries and plugins
./script/setup: Reloading the database
./script/setup: Restarting Pow/Passenger
./script/setup: Done!
```



ProTip

This script can be run with the `-v` flag for verbose output!

More details at CONTRIBUTING.md#setting-up-the-database

Running the tests

```
$ bundle exec rake
Run options: --seed 1130

# Running:

....S.........................................S...........
.............S............................................
...................................SS.....................
..........S...............................................
..........................................................

Finished in 156s, 11.8130 runs/s, 23.0702 assertions/s.
1849 runs, 3611 assertions, 0 failures, 0 errors, 6 skips
```

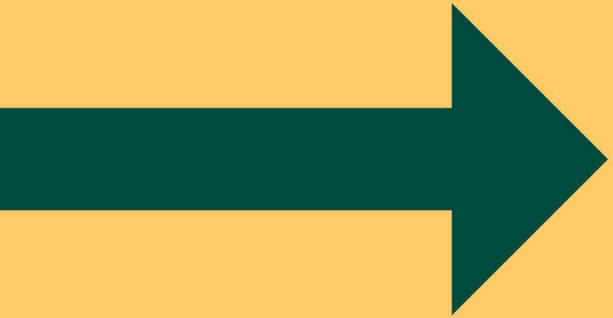
More details at CONTRIBUTING.md#running-tests

Practice yourself

Setup your environment

Getting started with

`typed: false`



Let's get started!

```
$ git checkout -b sorbet
```

Gradual typing

~~typed: ignore~~

typed: false (default)

typed: true

typed: strict

~~typed: strong~~

`# typed: false`

~~`typed: ignore`~~

`typed: false` (default)

`typed: true`

`typed: strict`

~~`typed: strong`~~

- Files without sigils are **`typed: false`**
- This is where we stand right now
- At this strictness level, Sorbet will report
  - Syntax errors
  - Unresolved constants
  - Unsupported constructs

Gemfile

```
44  gem "rotp"
45  gem "unpwn"
46  gem "sorbet-runtime"
…   ...
56  group :development, :test do
…     ...
61    gem "factory_bot_rails"
62    gem "sorbet", require: false
63    gem "spoom", require: false
64    gem "tapioca", require: false
…     ...
69    gem "rubocop-rails", "~> 2.12", require: false
70    gem "rubocop-sorbet", require: false
…     ...
73  end
```

Installing the gems

```
$ bundle
Bundle complete!
72 Gemfile dependencies,
203 gems now installed.
$ git add Gemfile Gemfile.lock
$ git commit -m "Install Sorbet"
```

Initializing Tapioca

```
$ bundle exec tapioca init
      create  sorbet/config
      create  sorbet/tapioca/config.yml
      create  sorbet/tapioca/require.rb
      create  bin/tapioca
$ cat sorbet/config
--dir
.
$ git add sorbet/ bin/tapioca
$ git commit -m "Init Tapioca & Sorbet"
```

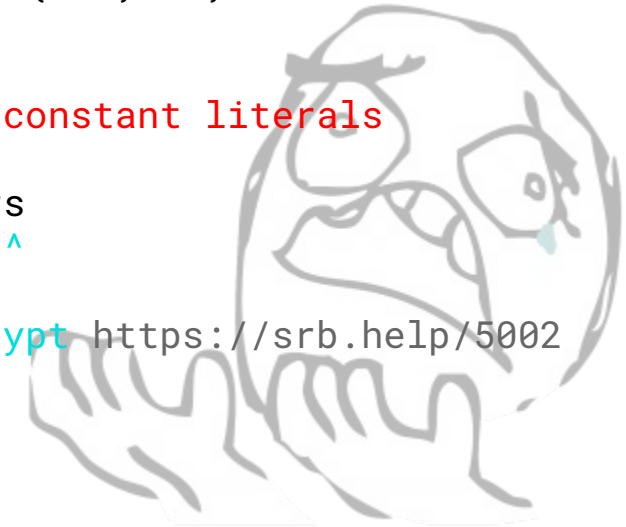
ProTip

Do not use the `srb init` command. Tapioca takes care of that for us!

More details at https://github.com/Shopify/tapioca#getting-started

Running type checking for the first time

```
$ bundle exec srb tc
...
lib/fastly.rb:22: Unable to resolve constant Rails https://srb.help/5002
    22 |      Rails.logger.debug { "Fastly purge url=#{url} " }
             ^^^^^

rubygems_helper_test.rb:4: include must only contain constant literals
https://srb.help/4002
     4 |  include Rails.application.routes.url_helpers
              ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

app/models/user.rb:66: Unable to resolve constant BCrypt https://srb.help/5002
    66 |  rescue BCrypt::Errors::InvalidHash
              ^^^^^^
...

Errors: 1035
```



ProTip

Use the VS Code extension for Sorbet!

Features

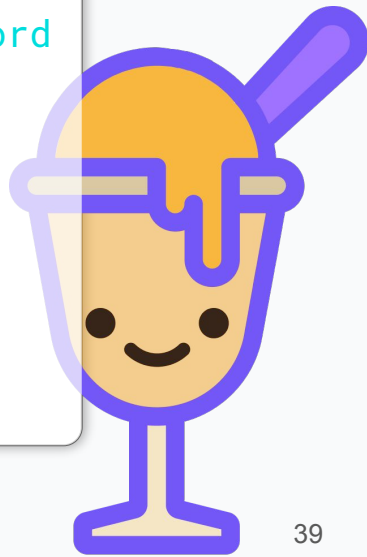
- Errors list
- Squiggly lines
- Error hover
- Jump to error
- Quick fixes

ProTip

Spoom can declutter, group and sort errors to make large outputs easier to read!

Running type checking with spoom

```
$ bundle exec spoom tc -s code
5002 - app/models/user.rb:2: Unable to resolve constant User
5002 - app/models/user.rb:3: Unable to resolve constant Gravtastic
5002 - app/models/user.rb:66: Unable to resolve constant BCrypt
5002 - app/models/user.rb:71: Unable to resolve constant ActiveRecord
5002 - app/models/user.rb:180: Unable to resolve constant Token
5002 - app/models/user.rb:194: Unable to resolve constant Token
5002 - app/models/user.rb:217: Unable to resolve constant I18n
5002 - app/models/user.rb:221: Unable to resolve constant I18n
5002 - app/models/user.rb:280: Unable to resolve constant ROTP
5002 - app/models/user.rb:292: Unable to resolve constant I18n
...
Errors: 1035
```

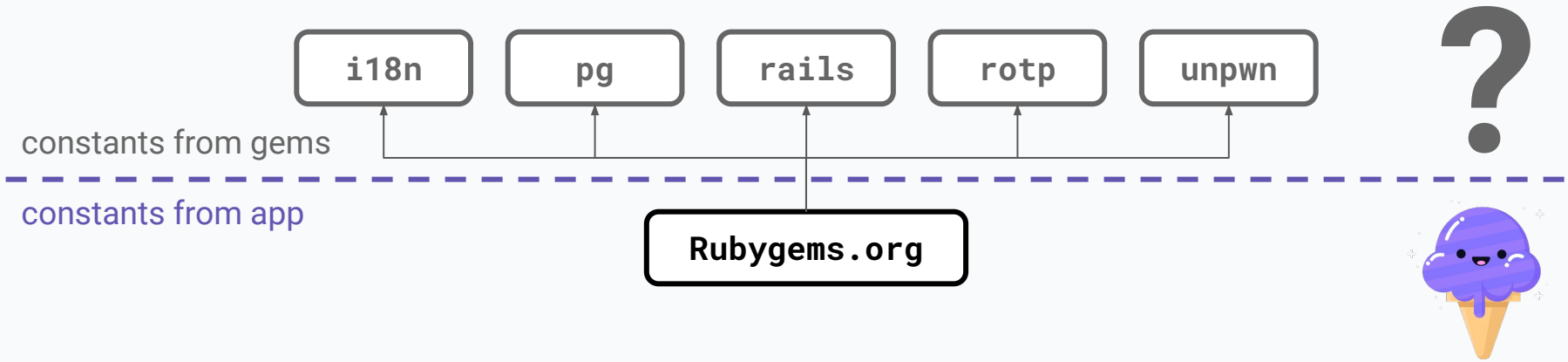


More details at https://github.com/Shopify/spoom#errors-sorting-and-filtering

Why so many unresolved constants?

Sorbet doesn’t know about

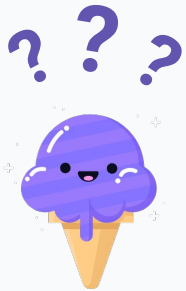
- the constants coming from the gems we depend on
- any constants that are created only at runtime

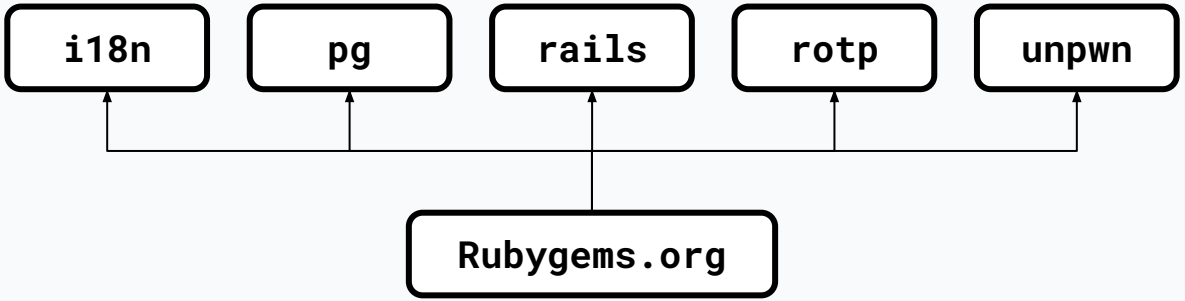


Why not pass the source of the gems to Sorbet?

- gems may not be compatible with Sorbet yet
- would also require passing transitive dependencies
- so much code would increase type checking

all constants in the universe





RBI files

“Ruby Interface” files

A simplified representation of the gem contents:

- constants
- methods
- inheritance & mixins

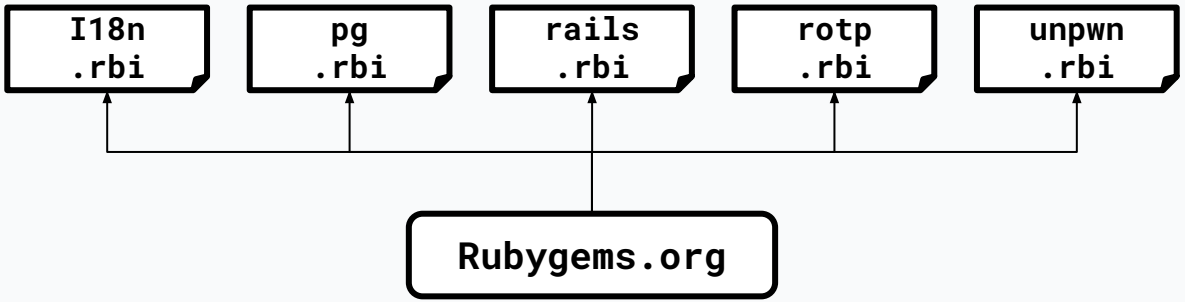
i18n@1.10.0.rbi

```ruby
module I18n
  extend ::I18n::Base

  class << self
    def cache_key_digest; end
    def cache_key_digest=(key_digest); end
    def cache_namespace; end
    def cache_namespace=(namespace); end
    ...
  end
end
...
```

constants from gems

+

constants from app



Generating gems RBIs with Tapioca

```
$ bin/tapioca gems
...
Generating RBI files of gems that are added or updated:

  Requiring all gems to prepare for compiling... Done

  Compiled i18n
      create  sorbet/rbi/gems/i18n@1.10.0.rbi

  Compiled pg
      create  sorbet/rbi/gems/pg@1.3.5.rbi

  ...

All operations performed in working directory.
Please review changes and commit them.
```

More details at https://github.com/Shopify/tapioca#generating-rbi-files-for-gems

Running type checking with spoom

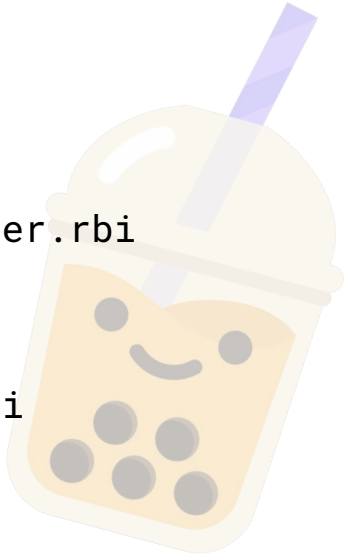
```
$ bundle exec spoom tc -s code
4002 - test/unit/helpers/rubygems_helper_test.rb:4: include must only contain constant literals
5002 - app/controllers/passwords_controller.rb:1: Unable to resolve constant PasswordsController
5002 - app/controllers/passwords_controller.rb:1: Unable to resolve constant PasswordsController
5002 - app/controllers/sessions_controller.rb:1: Unable to resolve constant SessionsController
5002 - app/controllers/sessions_controller.rb:1: Unable to resolve constant SessionsController
5002 - app/controllers/users_controller.rb:1: Unable to resolve constant UsersController
5002 - app/controllers/users_controller.rb:1: Unable to resolve constant UsersController
5002 - app/models/version.rb:21: Unable to resolve constant NAME_PATTERN
5002 - lib/rubygem_fs.rb:87: Unable to resolve constant NotFound
5002 - lib/unpwn_validator.rb:6: Unable to resolve constant Pwned
5002 - lib/unpwn_validator.rb:8: Unable to resolve constant Pwned

Errors: 12
```

Still some constants missing

- Maybe from DSLs?
- Maybe from parts of gems Tapioca didn't load?
- Maybe from meta-programming?
- Maybe from limitations of Sorbet?

Generating DSL RBIs with Tapioca

```
$ bin/tapioca dsl
Loading Rails application... Done
Loading DSL compiler classes... Done
Compiling DSL RBI files...

      ...
      create  sorbet/rbi/dsl/subscription.rbi
      create  sorbet/rbi/dsl/subscriptions_controller.rbi
      create  sorbet/rbi/dsl/user.rbi
      create  sorbet/rbi/dsl/users_controller.rbi
      create  sorbet/rbi/dsl/version.rbi
      create  sorbet/rbi/dsl/versions_controller.rbi
      ...

All operations performed in working directory.
Please review changes and commit them.
```



More details at https://github.com/Shopify/tapioca#generating-rbi-files-for-rails-and-other-dsls

Practice yourself

Generate the RBI files with tapioca

Generating the DSL RBI files resolved more constants

`PasswordsController`, `SessionsController` & `UsersController`

Running type checking with spoom

```
$ bundle exec spoom tc -s code
4002 - test/unit/helpers/rubygems_helper_test.rb:4: include must only contain constant literals
5002 - app/models/version.rb:21: Unable to resolve constant NAME_PATTERN
5002 - lib/rubygem_fs.rb:87: Unable to resolve constant NotFound
5002 - lib/unpwn_validator.rb:6: Unable to resolve constant Pwned
5002 - lib/unpwn_validator.rb:8: Unable to resolve constant Pwned

Errors: 5
```

Still some constants missing

- ~~Maybe from DSLs?~~
- Maybe from parts of gems Tapioca didn't load?
- Maybe from meta-programming?
- Maybe from limitations of Sorbet?

Pwned?

Tapioca can't always automatically load all the gems from the application (autoloads, optional dependencies, hidden requires, …)

Generating the RBI for pwned

```
$ bin/tapioca gem pwned

  Compiled pwned (empty output)
      identical  sorbet/rbi/gems/pwned@2.3.0.rbi
```

Generating the RBI for pwned

```
$ bin/tapioca gem pwned

  Compiled pwned
      force      sorbet/rbi/gems/pwned@2.3.0.rbi
```

sorbet/tapioca/require.rb

```ruby
# typed: true
# frozen_string_literal: true

# Add your extra requires here
(`bin/tapioca require` can be
used to boostrap this list)
require "pwned"
```

More details at https://github.com/Shopify/tapioca#manually-requiring-parts-of-a-gem

Manually requiring pwned worked

More constants resolved: Pwned

Running type checking with spoom

```
$ bundle exec spoom tc -s code
4002 - test/unit/helpers/rubygems_helper_test.rb:4: include must only contain constant literals
5002 - app/models/version.rb:21: Unable to resolve constant NAME_PATTERN
5002 - lib/rubygem_fs.rb:87: Unable to resolve constant NotFound

Errors: 3
```

Still some constants missing

- ~~Maybe from DSLs?~~
- ~~Maybe from parts of gems Tapioca didn't load?~~
- Maybe from meta-programming?
- Maybe from limitations of Sorbet?

Aws::S3::Errors::NotFound?

Tapioca can't always find constants defined through meta-programming

Finding the constant

```
$ bin/rails c
> Aws::S3::Errors::NotFound
=> Aws::S3::Errors::NotFound
> Aws::S3::Errors.const_source_location(:NotFound)
=> ["/gems/aws-sdk-core-3.130.2/lib/aws-sdk-core/
   errors.rb", 380]
```

aws-sdk-core/errors.rb#L380

```
372       def set_error_constant(constant)
373         @const_set_mutex.synchronize do
374           # Ensure the const was not defined while blocked by the mutex
375           if error_const_set?(constant)
376             const_get(constant)
377           else
378             error_class = Class.new(const_get(:ServiceError))
379             error_class.code = constant.to_s
380             const_set(constant, error_class)
```

sorbet/rbi/shims/gems/aws-sdk-core.rbi

```
# typed: true

module Aws
 module S3
   module Errors
     class NotFound < ServiceError; end
   end
 end
end
```

ProTip

The command `tapioca todo` can shim the missing constants for you

Manually shimming aws worked

More constants resolved: NotFound

Running type checking with spoom

```
$ bundle exec spoom tc -s code
4002 - test/unit/helpers/rubygems_helper_test.rb:4: include must only contain constant literals
5002 - app/models/version.rb:21: Unable to resolve constant NAME_PATTERN

Errors: 2
```

Still some constants missing

- ~~Maybe from DSLs?~~
- ~~Maybe from parts of gems Tapioca didn't load?~~
- ~~Maybe from meta programming?~~
- Maybe from limitations of Sorbet?

Rubygem::NAME_PATTERN?

Sorbet doesn't understand constant resolution through inheritance #8

Open XrXr opened this issue on Dec 20, 2018 · 5 comments

lib/patterns.rb

```
1  module Patterns
2    extend ActiveSupport::Concern
3
4    SPECIAL_CHARACTERS     = ".-_".freeze
5    ALLOWED_CHARACTERS     = "[A-Za-z0-9#{Regexp.escape(SPECIAL_CHARACTERS)}]+".freeze
6    ROUTE_PATTERN          = /#{ALLOWED_CHARACTERS}/
7    LAZY_ROUTE_PATTERN     = /#{ALLOWED_CHARACTERS}?/
8    NAME_PATTERN           = /\A#{ALLOWED_CHARACTERS}\Z/
```

app/models/rubygem.rb

```
1  class Rubygem < ApplicationRecord
2    include Patterns
3    include RubygemSearchable
```

Solution

Let's reference the constant through `Patterns::NAME_PATTERN`

app/models/version.rb

```
ializer

utter::MAX_FIELD_LENGTH }, format: { with: /\A#{Gem::Version::VERSION_PATTER
mcutter::MAX_FIELD_LENGTH }, format: { with: Rubygem::NAME_PATTERN }
mcutter::MAX_FIELD_LENGTH }, format: { with: Patterns::NAME_PATTERN }
ueness: { case_sensitive: false }

enses, length: { maximum: Gemcutter::MAX_FIELD_LENGTH }, allow_blank: true
, :requirements, :cert_chain,
::MAX_TEXT_FIELD_LENGTH },


eate
on: :create
```

More details at https://github.com/sorbet/sorbet/issues/8

Error code 4002?

Sorbet requires that every `include` references a constant literal

Running type checking with spoom

```
$ bundle exec spoom tc -s code
4002 - test/unit/helpers/rubygems_helper_test.rb:4: include must only contain constant literals

Errors: 1
```

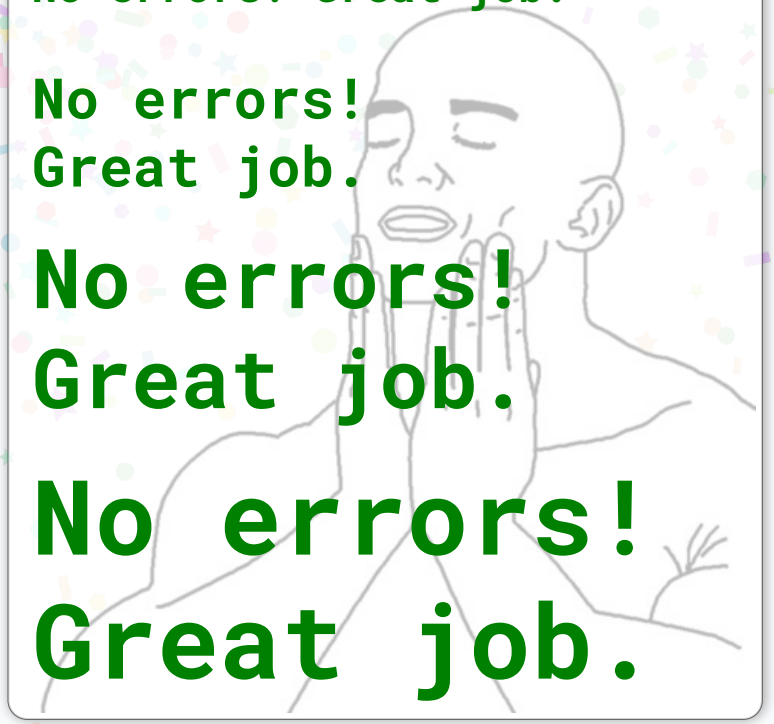
app/models/rubygem.rb

```
1  1  require "test_helper"
2  2
3  3  class RubygemsHelperTest < ActionView::TestCase
4     -  include Rails.application.routes.url_helpers
   4+ +  T.unsafe(self).include Rails.application.routes.url_helpers
5  5     include ApplicationHelper
6  6     include ERB::Util
```

More details at https://sorbet.org/docs/error-reference#4002

Running type checking

```
$ bundle exec srb tc
No errors! Great job.
```



Recap

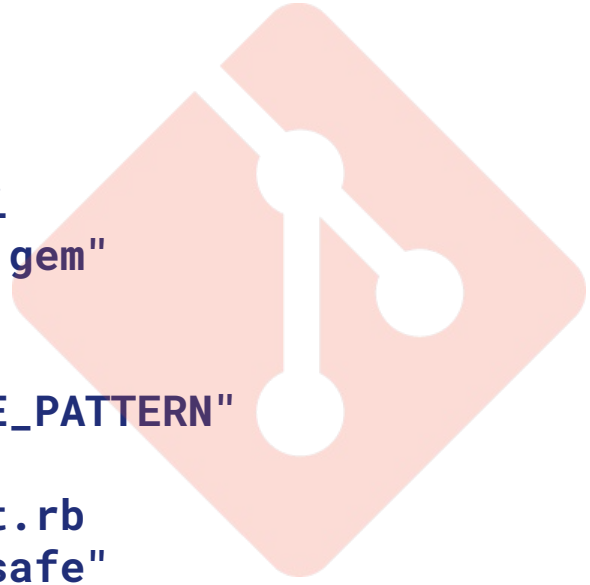
1. Running `tapioca init`
2. Running `tapioca gems`
3. Running `tapioca dsl`
4. Fixing the `require.rb` for `pwned`
5. Shimming `Aws::S3::Errors::NotFound`
6. Fixing `Rubygem::NAME_PATTERN`
7. Ignoring `include url_helpers`
8. Profit?

Practice yourself

Move the files to `typed: false`

Let's commit our work

```
$ git add sorbet/rbi/gems/ sorbet/tapioca/require.rb
$ git commit -m "Generate RBI for gems"

$ git add sorbet/rbi/dsl
$ git commit -m "Generate RBI for DSLs"

$ git add sorbet/rbi/shims/gems/aws-sdk-core.rbi
$ git commit -m "Shim missing constants for AWS gem"

$ git add app/models/version.rb
$ git commit -m "Use reference to Patterns::NAME_PATTERN"

$ git add test/unit/helpers/rubygems_helper_test.rb
$ git commit -m "Mark include of url_helpers unsafe"
```



typed: false

By default, all files without a sigil are typed: false, let's make this explicit

.rubocop.yml

```
1  1  inherit_from: .rubocop_todo.yml
2  2  require:
3  3    - ./test/safe_navigation_cop.rb
4  4    - rubocop-performance
5  5    - rubocop-rails
6  6    - rubocop-minitest
   7+   - rubocop-sorbet
```

Running Rubocop

```
$ bundle exec rubocop --autocorrect
297 files inspected, 277 offenses detected,
277 offenses corrected
```

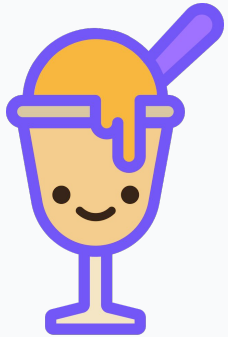
Diff

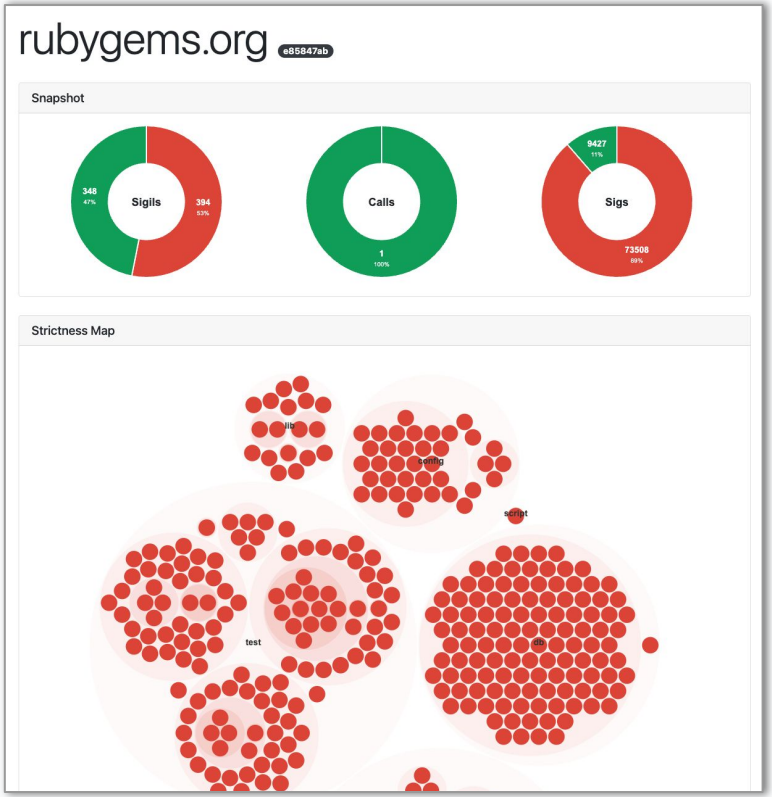
```
$ git diff
diff --git
a/app/controllers/adoptions_controller.rb
b/app/controllers/adoptions_controller.rb
@@ -1,3 +1,4 @@
+# typed: false
diff --git
a/app/controllers/api/base_controller.rb
b/app/controllers/api/base_controller.rb
@@ -1,3 +1,4 @@
+# typed: false
diff --git
a/app/controllers/api_keys_controller.rb
b/app/controllers/api_keys_controller.rb
@@ -1,3 +1,4 @@
+# typed: false
diff --git
a/app/helpers/application_helper.rb
b/app/helpers/application_helper.rb
@@ -1,3 +1,4 @@
+# typed: false
```

Spoom typing coverage report

Typing snapshots and strictness map

- Follow your progress
- Files typed
- Calls types
- Methods with signatures
- Find untyped clusters

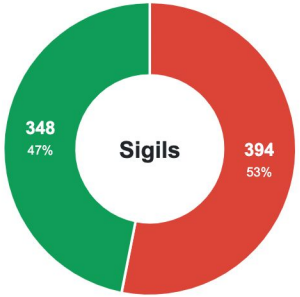




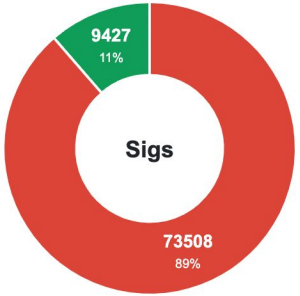
Spoom report

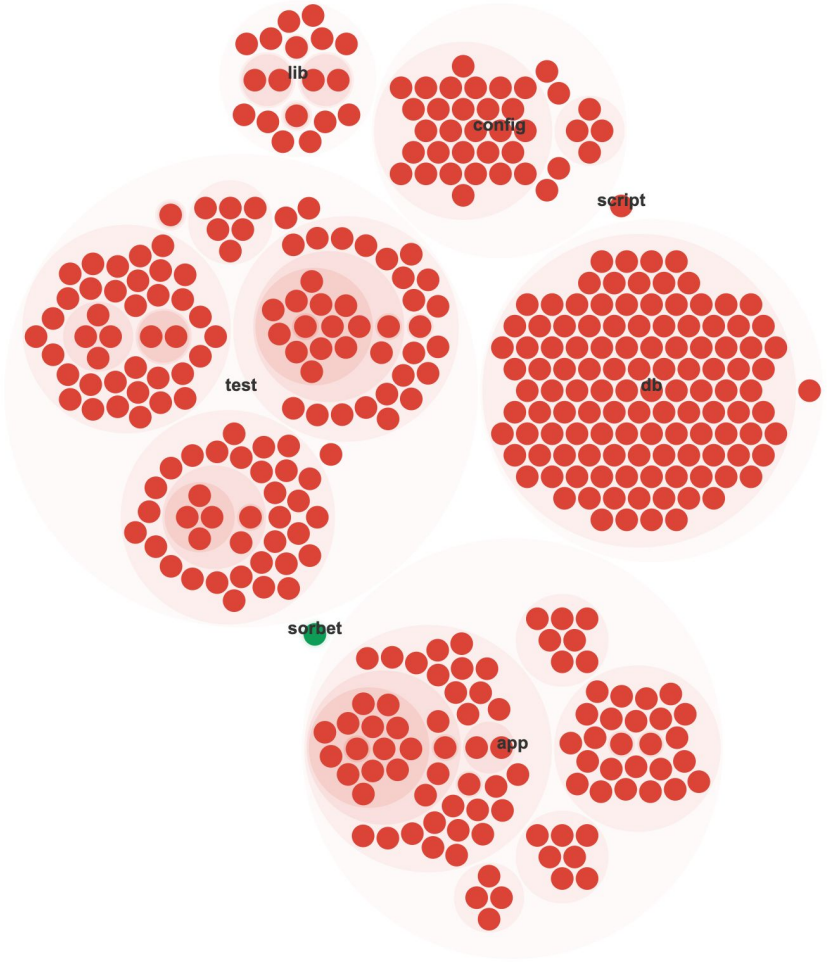
```
$ bundle exec spoom coverage --save
$ bundle exec spoom coverage report
$ bundle exec spoom coverage open
```

More details at https://github.com/Shopify/spoom#typing-coverage


348
47%
Sigils
394
53%

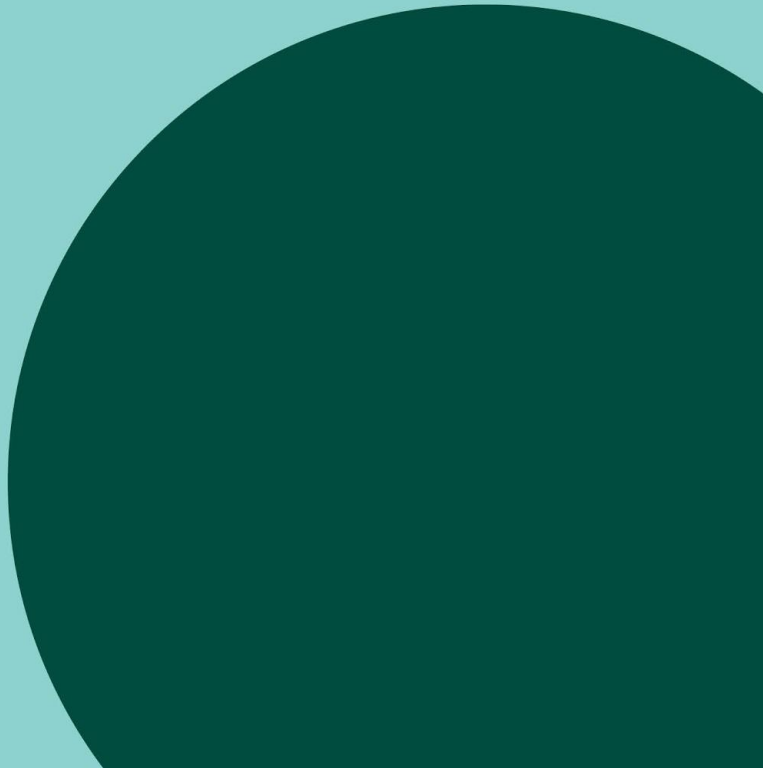

Calls
1
100%


9427
11%
Sigs
73508
89%


lib
config
script
test
db
sorbet
app

Moving files to

typed: true



https://github.com/Shopify/rubygems.org/commits/typed-false

`# typed: true`

~~`typed: ignore`~~

`typed: false` (default)

`typed: true`

`typed: strict`

~~`typed: strong`~~

- At this strictness level, Sorbet will report
  - Calls to nonexistent methods
  - Calls with mismatched args count
  - Calls with wrong args types
  - Incorrect types on variable usages
- Minimum level to enable LSP features
- Does not require signatures on methods

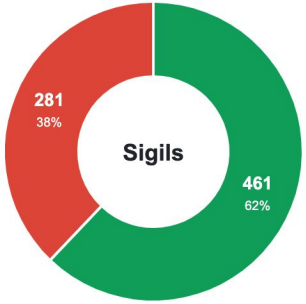
spoom bump

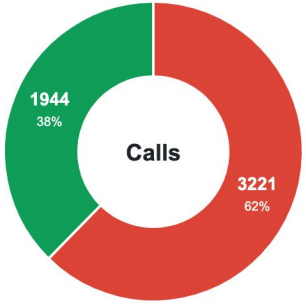
```
$ bundle exec spoom bump
Checking files...

Bumped 113 files from false to true
   ...
 + app/jobs/delete_user.rb
 + app/jobs/fastly_log_processor.rb
 + app/jobs/indexer.rb
 + app/models/application_record.rb
 + app/models/gem_dependent.rb
 + app/models/gem_info.rb
 + app/models/gem_typo.rb
 + app/models/gem_typo_exception.rb
 + app/models/log_ticket.rb
 + app/models/push_policy.rb
 + app/models/pusher.rb
   ...
```

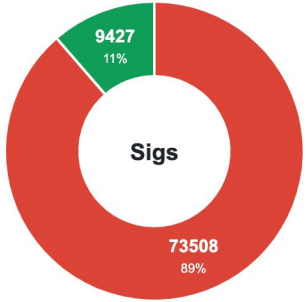
spoom coverage

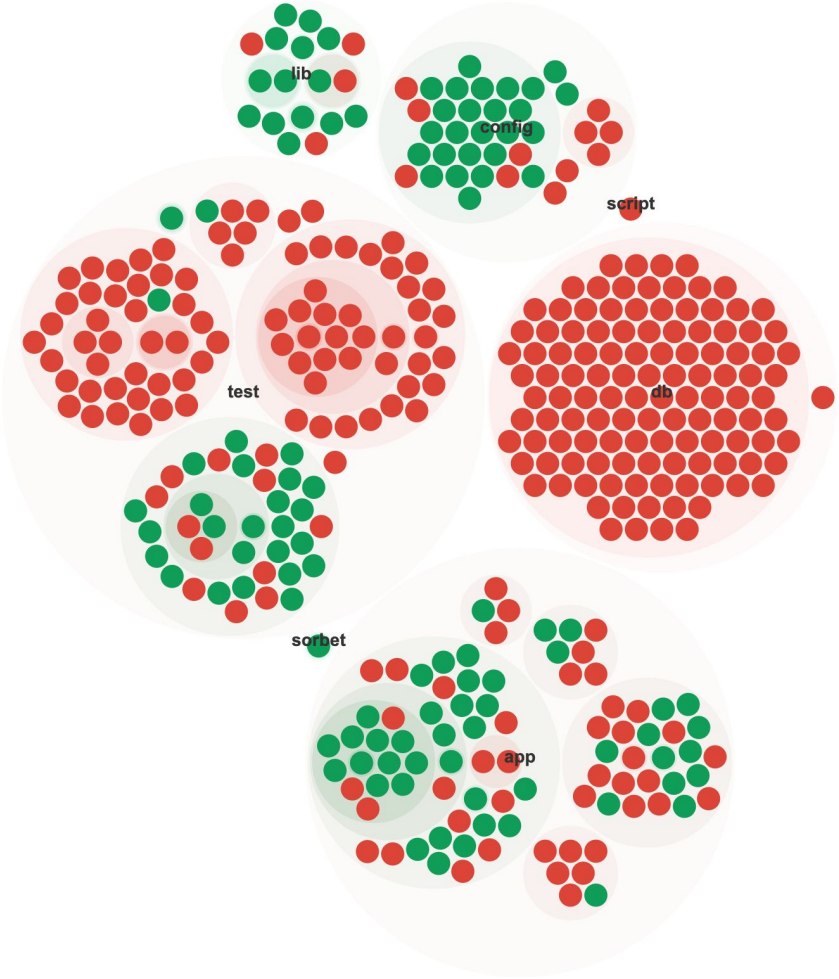
```
$ bundle exec spoom coverage --save
Content:
  files: 742 (including 355 RBIs)
  modules: 2356
  classes: 7446
  methods: 82935

Sigils:
  false: 281 (38%)
  true: 461 (62%)

Methods:
  with signature: 9427 (11%)
  without signature: 73508 (89%)

Calls:
  typed: 1944 (38%)
  untyped: 3221 (62%)
```

More details at https://github.com/Shopify/spoom#change-the-sigil-used-in-files

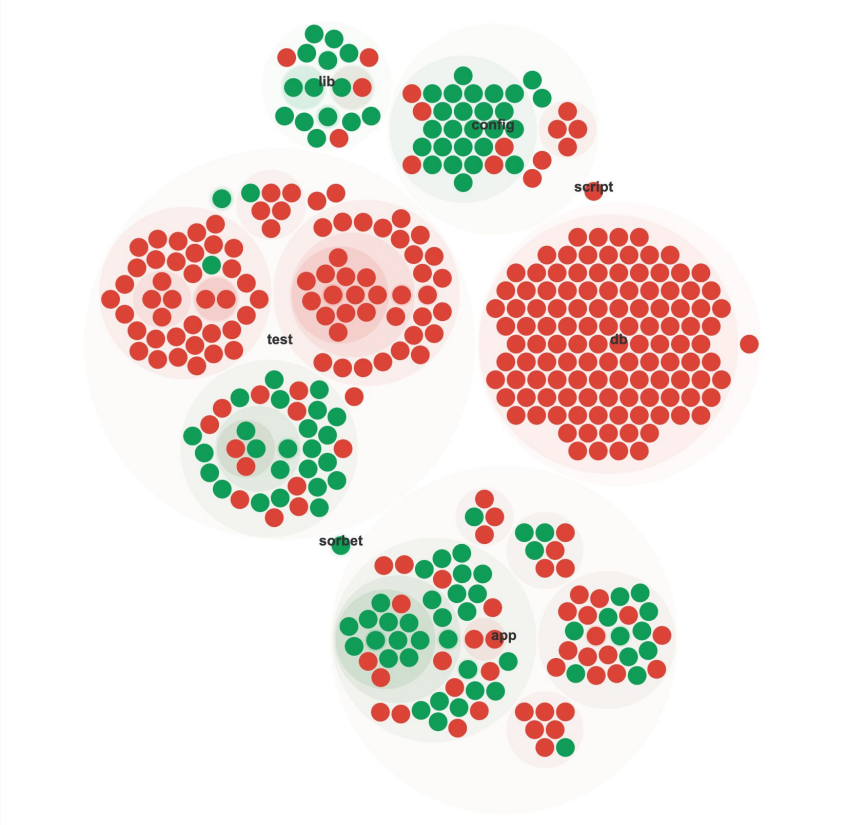

281
38%
Sigils
461
62%


1944
38%
Calls
3221
62%


9427
11%
Sigs
73508
89%


lib
config
script
test
db
sorbet
app

What to focus on first?



Most reused parts of the app

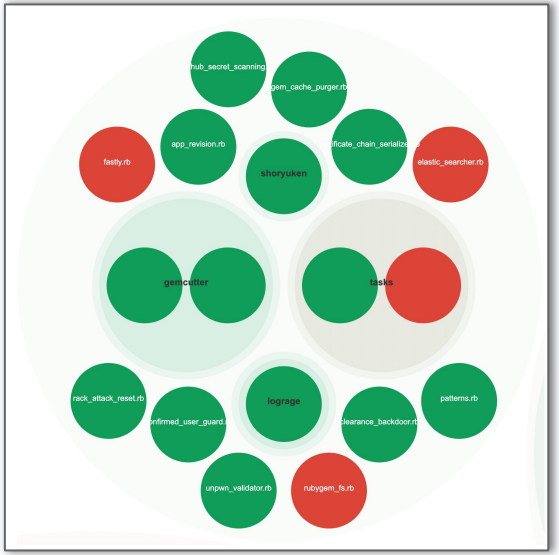
- lib
- helpers
- methods inside models

Critical parts of the app

- complex pieces
- sources of errors
- code with lot of churn

Moving `lib/` to `typed: true`

```
$ bundle exec spoom bump --force lib/
Bumped 4 files from false to true:
 + lib/elastic_searcher.rb
 + lib/fastly.rb
 + lib/rubygem_fs.rb
 + lib/tasks/helpers/gemcutter_tasks_helper.rb

$ bundle exec spoom tc -s code

...

Errors: 39
$ code .
```



Type errors in `lib/fastly.rb`

```
$ bundle exec srb tc
...
lib/fastly.rb:13: Method split does not exist on NilClass component
of T.nilable(String) https://srb.help/7003
    13 |    ENV["FASTLY_DOMAINS"].split(",").each do |domain|
                                  ^^^^^
  Got T.nilable(String) originating from:
    lib/fastly.rb:13:
    13 |    ENV["FASTLY_DOMAINS"].split(",").each do |domain|
           ^^^^^^^^^^^^^^^^^^^^^^
  Autocorrect: Use `-a` to autocorrect
    lib/fastly.rb:13: Replace with T.must(ENV["FASTLY_DOMAINS"])
    13 |    ENV["FASTLY_DOMAINS"].split(",").each do |domain|
           ^^^^^^^^^^^^^^^^^^^^^^
...
```

Method `split` does not exist on `NilClass` component of `T.nilable(String)`

lib/fastly.rb

```ruby
def self.purge(options = {})
  return unless ENV["FASTLY_DOMAINS"]
  ENV["FASTLY_DOMAINS"].split(",").each do |domain|
    url = "https://#{domain}/#{options[:path]}"
```

Problem

ENV content could change between the two calls

lib/fastly.rb

```ruby
def self.purge(options = {})
  return unless ENV["FASTLY_DOMAINS"]
  ENV["FASTLY_DOMAINS"].split(",").each do |domain|
  fastly_domains = ENV["FASTLY_DOMAINS"]
  return unless fastly_domains
  fastly_domains.split(",").each do |domain|
    url = "https://#{domain}/#{options[:path]}"
```

Solution

We can use a local variable to help flow typing

Type errors in `lib/rubygem_fs.rb`

```
$ bundle exec srb tc
...
lib/rubygem_fs.rb:24: Method s3 does not exist on
T.class_of(RubygemFs) https://srb.help/7003
    24 |      s3.create_bucket(bucket: bucket)
           ^^
lib/rubygem_fs.rb:24: Method bucket does not exist on
T.class_of(RubygemFs) https://srb.help/7003
    24 |      s3.create_bucket(bucket: bucket)
                                   ^^^^^^^
lib/rubygem_fs.rb:26: Method init does not exist on RubygemFs::S3
https://srb.help/7003
    26 |    @fs.init
               ^^^^
...
```

Methods `s3` and `bucket` do not exist on `T.class_of(RubygemFs)`

lib/rubygems_fs.rb

```ruby
def self.s3!(host)
  @fs = RubygemFs::S3.new(...)
  @fs.define_singleton_method(:init) do
    s3.create_bucket(bucket: bucket)
  end
  @fs.init
end
```

Problem

Sorbet doesn't understand the context inside `define_singleton_method`

lib/rubygems_fs.rb

```ruby
def init
  s3.create_bucket(bucket: bucket)
end
```

Solution

We can use a real method!

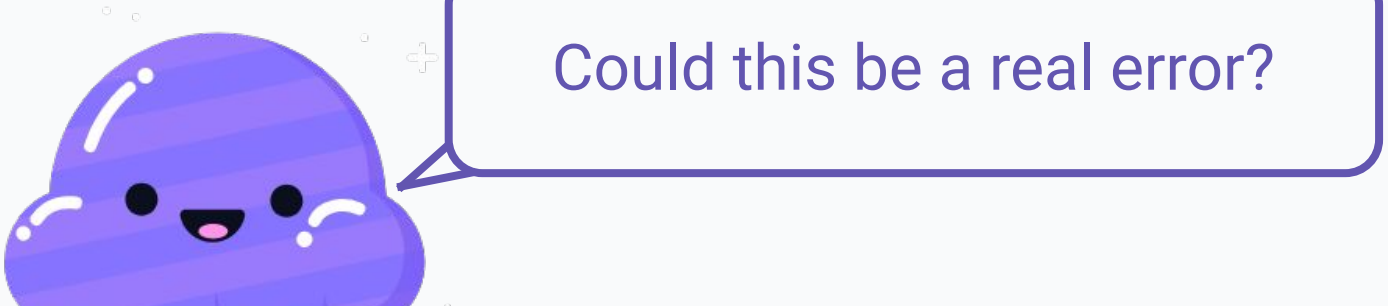
Type errors in `lib/tasks/helpers/gemcutter_tasks_helper.rb`

```
$ bundle exec srb tc
...
lib/tasks/helpers/gemcutter_tasks_helper.rb:39: Method version does
not exist on GemcutterTaskshelper https://srb.help/7003
    39 | … required_ruby_version for version: #{version.full_name}"
                                                  ^^^^^^^

lib/tasks/helpers/gemcutter_tasks_helper.rb:39: Method version does
not exist on T.class_of(GemcutterTaskshelper) https://srb.help/7003
    39 | … required_ruby_version for version: #{version.full_name}"
                                                  ^^^^^^^

...
```

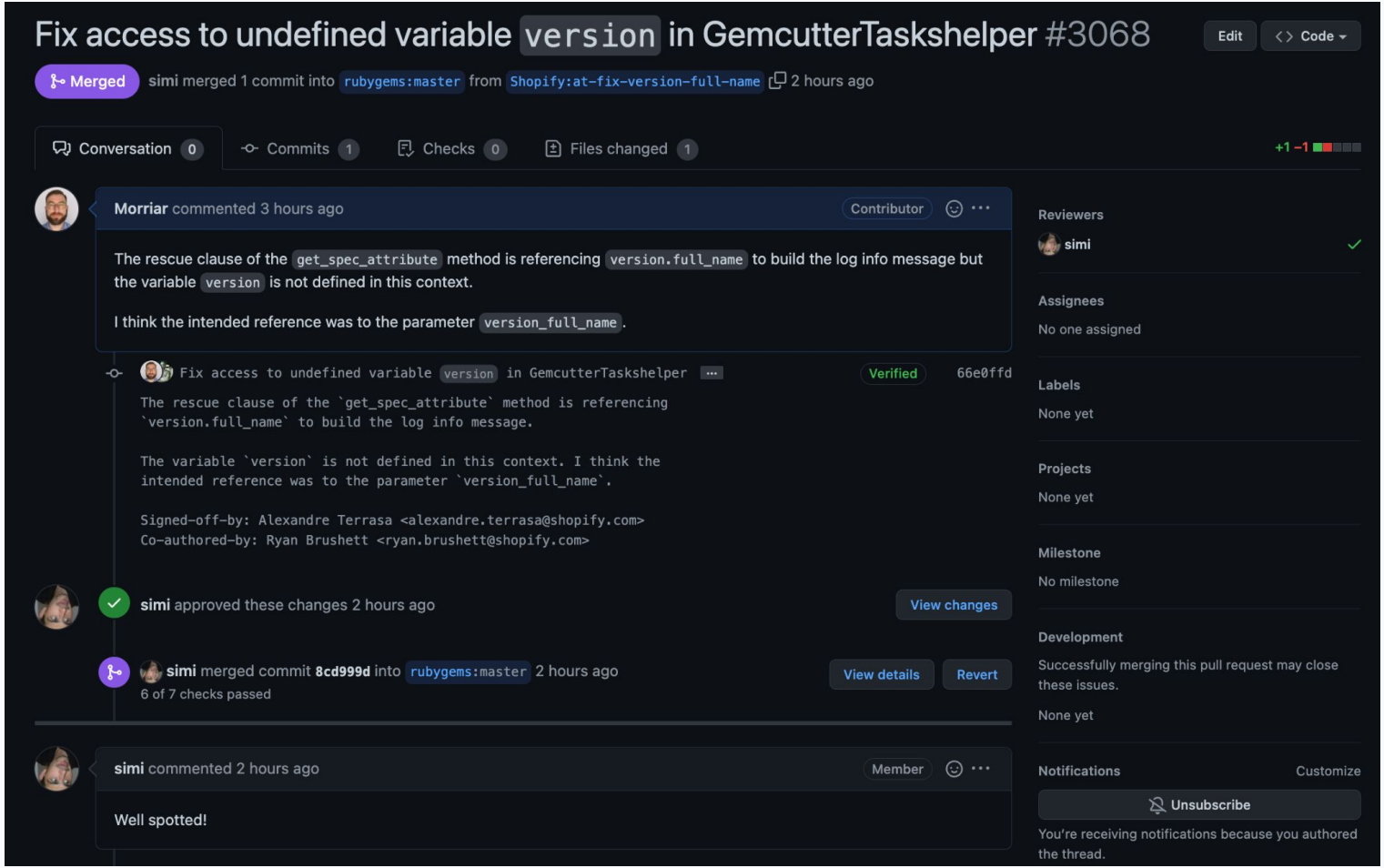
Method `version` does not exist on `GemcutterTaskshelper`

lib/tasks/helpers/gemcutter_tasks_helper.rb

```ruby
def get_spec_attribute(version_full_name, attribute_name)
  ...
rescue StandardError => e
  Rails.logger.info("... for version: #{version.full_name}"\
  ...
end
```



https://github.com/rubygems/rubygems.org/pull/3068



Type errors in `lib/elastic_searcher.rb`

```
$ bundle exec srb tc
...
lib/elastic_searcher.rb:18: Method __elasticsearch__ does not exist
on T.class_of(Rubygem) https://srb.help/7003
    18 |    result = Rubygem.__elasticsearch__.search
                            ^^^^^^^^^^^^^^^^^^
lib/elastic_searcher.rb:21: Method legacy_search does not exist on
T.class_of(Rubygem) https://srb.help/7003
    21 |    Rubygem.legacy_search(@query).page(@page)
                   ^^^^^^^^^^^^^
lib/elastic_searcher.rb:39: Method query does not exist on
ElasticSearcher https://srb.help/7003
    39 |      query do
             ^^^^^
...
```

Method `__elasticsearch__` does not exist on `T.class_of(Rubygem)`

Finding the method

```
$ bin/rails c
> Rubygem.method(:__elasticsearch__).source_location
=> ["/gems/elasticsearch-model-7.1.1/lib/elasticsearch/model/proxy.rb", 50]
```

/gems/elasticsearch/lib/model/proxy.rb

```
47      module Proxy
48        def self.included(base)
49          base.class_eval do
50            def self.__elasticsearch__ &block
51              @__elasticsearch__ ||= ClassMethodsProxy.new(self)
52              @__elasticsearch__.instance_eval(&block) if block_given?
53              @__elasticsearch__
54            end
```

Problem

Sorbet can't see the method defined through meta-programming

Method `__elasticsearch__` does not exist on `T.class_of(Rubygem)`

/gems/elasticsearch/lib/model/proxy.rb

```
47      module Proxy
48        def self.included(base)
49          base.class_eval do
50            def self.__elasticsearch__ &block
51              @__elasticsearch__ ||= ClassMethodsProxy.new(self)
52              @__elasticsearch__.instance_eval(&block) if block_given?
53              @__elasticsearch__
54            end
```

Problem

Sorbet can't see the method defined through meta-programming

sorbet/rbi/shims/app/models/rubygem.rbi

```
# typed: true

class Rubygem
 def self.__elasticsearch__(&block); end
end
```

Solution

We can define this method manually in a RBI file (shim)

Practice yourself

Write the RBI shim for `__elasticsearch__`

A bunch of methods from `elasticsearch-dsl` do not exist on `ElasticSearcher`

Problem

Sorbet doesn't bind `self` to the correct type inside the blocks

Solution

We can manually bind `self` with a signature inside in a shim

lib/elastic_searcher.rb

```
38    Elasticsearch::DSL::Search.search do
39      query do
40        function_score do
41          query do
42            bool do
```

elasticsearch-dsl-ruby/lib/elasticsearch/dsl/search.rb

```ruby
def initialize(*args, &block)
  @options = Options.new *args
  @block = block
  if @block
    @block.arity < 1 ? self.instance_eval(&@block) : @block.call(self)
  end
end
```

sorbet/rbi/shims/gems/elasticsearch-dsl.rbi

```ruby
module Elasticsearch::DSL::Search
 sig do
   params(
     args: T.untyped
     block: T.nilable(
       T.proc.bind(Search).void
     )
   ).void
 end
 def self.search(*args, &block); end
end
```

https://github.com/Shopify/rbi-central/blob/main/rbi/annotations/elasticsearch-dsl.rbi

ProTip

The command `tapioca annotations` can import common shims for you

BETA

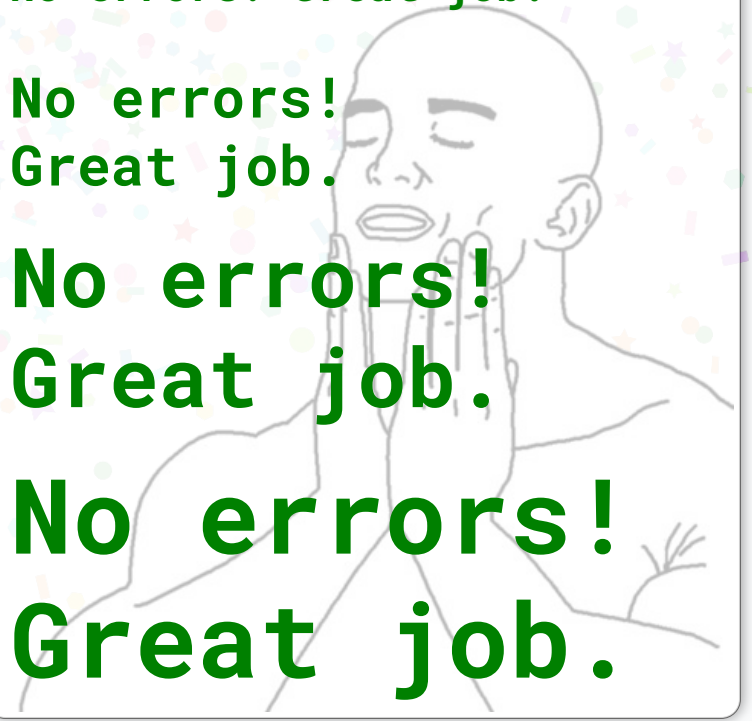
Importing gem annotations with Tapioca

```
$ bin/tapioca annotations
Retrieving index from central repository... Done
Listing gems from Gemfile.lock... Done
Fetching gem annotations from central repository...

  Fetched elasticsearch-dsl
      create  sorbet/rbi/annotations/elasticsearch-dsl.rbi

Done
```

https://github.com/Shopify/rbi-central/

Running type checking

```
$ bundle exec srb tc
No errors! Great job.
```



Recap

1. Moving files to `typed: true` automatically with `spoom bump`
2. Forcing files to `typed: true` automatically with `spoom bump --force`
3. Creating typing coverage snapshots with `spoom coverage --save`
4. Creating a typing coverage report with `spoom coverage report`
5. Importing RBI annotations with `tapioca annotations`
6. Resolve the method calls
7. Profit?

Type assertions

Sometimes we have to help Sorbet figuring the right type

- `T.let` to declare the type of a variable

```
foo = T.let(nil, T.nilable(String))
```

- `T.cast` to coerce the type of a variable

```
foo = T.cast(bar, String)
```

- `T.must` to remove `T.nilable` from a type

```
bar = T.must(foo)
```

- `T.bind` to define the type of `self`

```
T.bind(self, ApiKey)
```

https://sorbet.org/docs/type-assertions

Method `otp_verified?` does not exist on `NilClass` component of `T.nilable(User)`

Problem

Sorbet found that the receiver could be `nil`

app/models/api_key.rb

```
24    def mfa_authorized?(otp)
25      return true unless mfa_enabled?
26      user.otp_verified?(otp)
27    end
```

AR fields are always nilable

```
$ bin/rails c
> ApiKey.new
=> #<ApiKey:0x0000000108357d80
> api_key.user
=> nil
```

Solution

We can ensure the receiver won't be `nil` with `T.must(receiver)`

app/models/api_key.rb

```
def mfa_authorized?(otp)
  return true unless mfa_enabled?
  user.otp_verified?(otp)
  T.must(user).otp_verified?(otp)
end
```

https://sorbet.org/docs/type-assertions#tmust

ProTip

Sorbet provides an autocorrect for this!

In VS Code: ⌘ + .

Method `new_record?` does not exist on `T.class_of(ApiKey)`

Problem

Sorbet doesn't bind `self` to the correct type inside the block

Solution

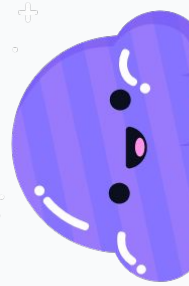
We can manually bind `self` with `T.bind(self, Type)`

app/models/api_key.rb

```ruby
API_SCOPES.each do |scope|
  define_method(:"can_#{scope}?") do
    T.bind(self, ApiKey)
    scope_enabled = send(scope)
    return scope_enabled if !scope_enabled || new_record?
    touch :last_accessed_at
  end
end
```

https://sorbet.org/docs/type-assertions#tbind

Escape hatches



Sometimes we can't represent the type correctly

- `T.untyped` to disable type checking on a variable

```
foo = T.let(nil, T.untyped)
foo.some_method_sorbet_cant_see
```

- `T.unsafe` to disable type checking on a callsite

```
foo = nil
T.unsafe(foo).some_method_sorbet_cant_see
```

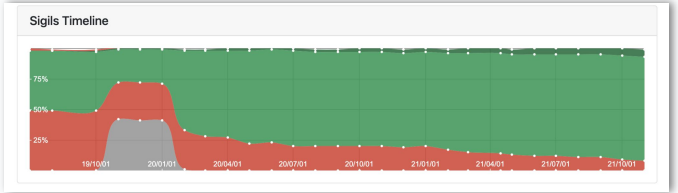
https://sorbet.org/docs/troubleshooting#escape-hatches

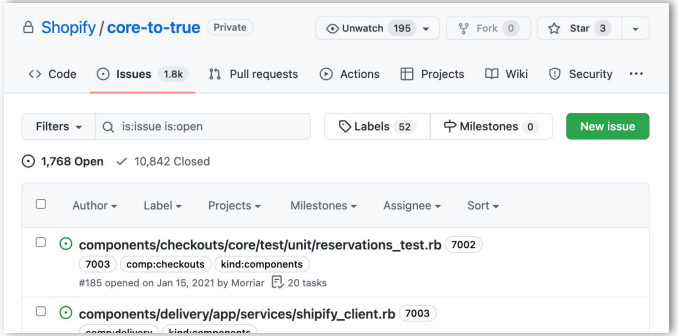
Practice yourself

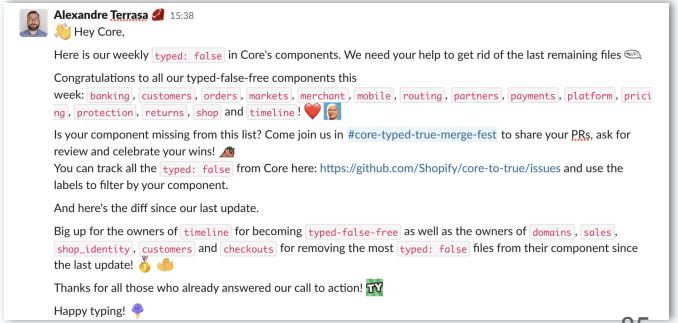
Move more files to `typed: true`

Reaching 100% `typed: true` can take a while

- Use VS Code and the autocorrects
- Tackle the <7000 error codes first
- Go directory by directory
- Find error clusters to bump many files at once
- Enable cops and create sh*tlists
- Leverage your colleagues and contributors!
- Encouragements, rewards and gamification help



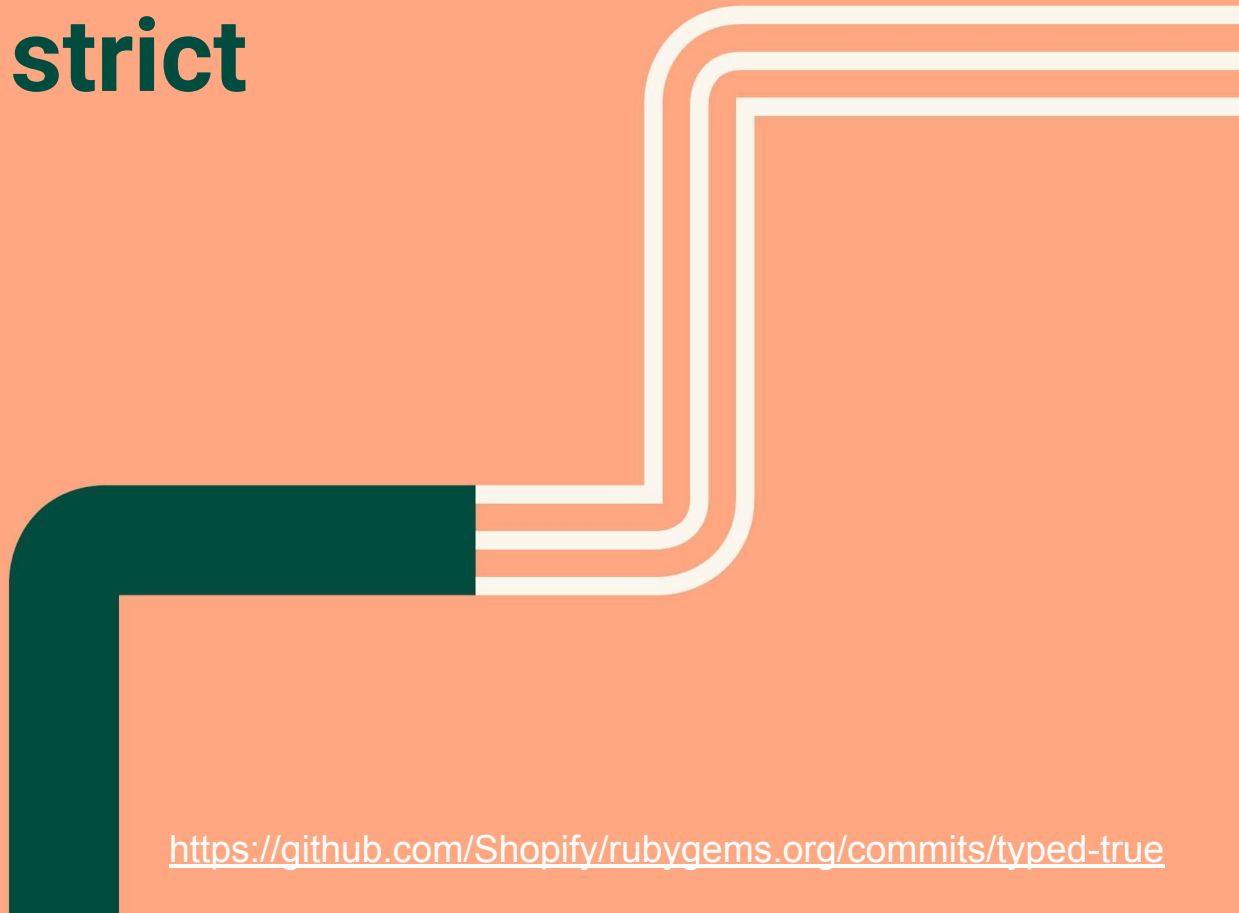




https://www.youtube.com/watch?v=a3jfpSmikdg

Gradually moving to

typed: strict



https://github.com/Shopify/rubygems.org/commits/typed-true

`# typed: strict`

~~`typed: ignore`~~

`typed: false` (default)

`typed: true`

`typed: strict`

~~`typed: strong`~~

- At this strictness level, Sorbet will report
  - Methods without signatures
  - Constants declared without type
  - Attributes declared without type
- Useful level to avoid typing drift
- But also a bit verbose

Signatures

- Enable static & dynamic type checking
- Valid Ruby syntax
- Need to `extend T::Sig` in Ruby files
  - (Not needed in RBI files)
- Written just above the method
  - (or accessor)
- Takes a block

my_class.rb

```ruby
class MyClass
  extend T::Sig

  sig { void }
  def initialize
    # ...
  end
end
```

my_class.rbi

```ruby
class MyClass
  sig { void }
  def initialize
    # ...
  end
end
```

Return types

sig_void.rb

```ruby
sig { void }
def initialize
  # ...
end
```

sig_returns.rb

```ruby
sig { returns(T::Boolean) }
def blank?
  # ...
end
```

https://sorbet.org/docs/sigs#returns--void-annotating-return-types

Parameter types

sig_params.rb

```ruby
sig do
  params(
    a: Integer,
    b: T.nilable(Integer),
    args: Integer,
    x: Integer,
    y: T.nilable(Integer),
    kwargs: Integer,
    block: T.proc.params(x: Integer).returns(String)
  ).void
end
def foo(a, b = 42, *args, x:, y: 42, **kwargs:, &block); end
```

https://sorbet.org/docs/sigs#params-annotating-parameter-types

Binding `self` inside a block

foo.rb

```ruby
class Foo
 def foo; end
end
```

main.rb

```ruby
bar do
  foo
end
```

bar.rb

```ruby
extend T::Sig

sig { params(block: T.proc.bind(Foo).void).void }
def bar(&block)
  Foo.new.instance_eval(&block)
end
```

https://sorbet.org/docs/sigs#block-parameters

A bunch of methods from `elasticsearch-dsl` do not exist on `ElasticSearcher`

Problem

Sorbet doesn't bind `self` to the correct type inside the blocks

Solution

We can manually bind `self` with a signature inside in a shim

lib/elastic_searcher.rb

```
38    Elasticsearch::DSL::Search.search do
39      query do
40        function_score do
41          query do
42            bool do
```

elasticsearch-dsl-ruby/lib/elasticsearch/dsl/search.rb

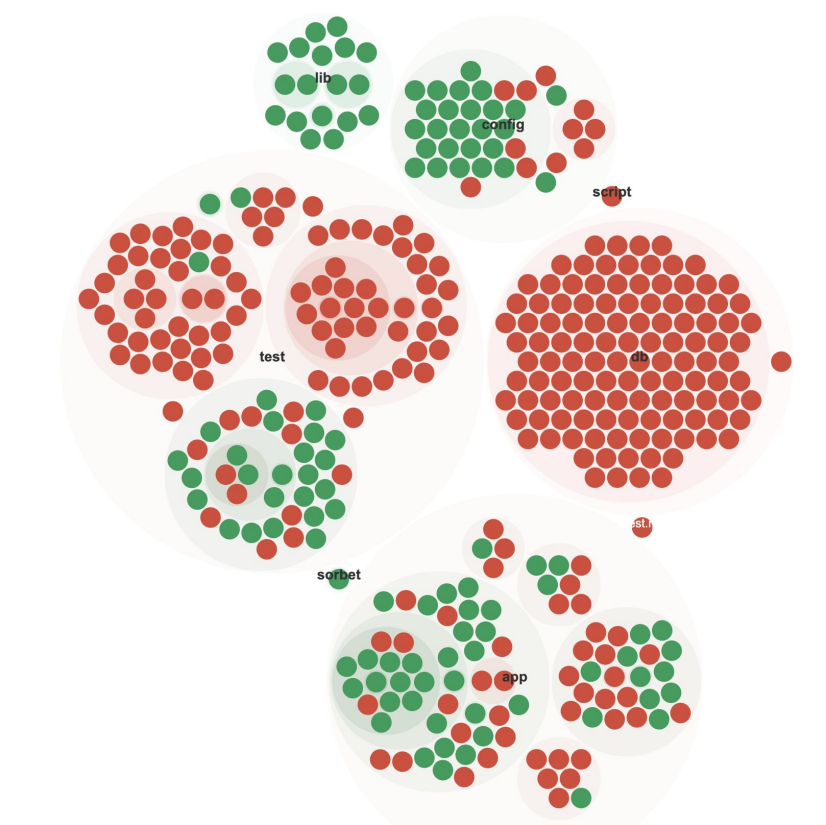
```ruby
def initialize(*args, &block)
  @options = Options.new *args
  @block = block
  if @block
    @block.arity < 1 ? self.instance_eval(&@block) : @block.call(self)
  end
end
```

sorbet/rbi/shims/gems/elasticsearch-dsl.rbi

```ruby
module Elasticsearch::DSL::Search
 sig do
   params(
     args: T.untyped
     block: T.nilable(
       T.proc.bind(Search).void
     )
   ).void
 end
 def self.search(*args, &block); end
end
```

https://github.com/Shopify/rbi-central/blob/main/rbi/annotations/elasticsearch-dsl.rbi

What to focus on first?



Most reused parts of the app

- lib
- helpers
- methods inside models

Critical parts of the app

- complex pieces
- sources of errors
- code with lot of churn

Moving a file in `lib` to `typed: strict`

lib/github_secret_scanning.rb

```
# typed: true
# typed: strict
class GithubSecretScanning
```

Type checking

```
$ bundle exec spoom tc -s code
6002 - lib/github_secret_scanning.rb:7: Use of undeclared variable @public_key
6002 - lib/github_secret_scanning.rb:11: Use of undeclared variable @public_key
6002 - lib/github_secret_scanning.rb:17: Use of undeclared variable @public_key
7017 - lib/github_secret_scanning.rb:6: This function does not have a sig
7017 - lib/github_secret_scanning.rb:10: This function does not have a sig
7017 - lib/github_secret_scanning.rb:16: This function does not have a sig
7017 - lib/github_secret_scanning.rb:20: This function does not have a sig
7017 - lib/github_secret_scanning.rb:28: This function does not have a sig
7027 - lib/github_secret_scanning.rb:4: Constants must have type annotations with T.let when specifying # typed: strict

Errors: 9
```

Type errors in `lib/github_secret_scanning.rb`

```
$ bundle exec srb tc
...
lib/github_secret_scanning.rb:3: Constants must have type annotations
with T.let when specifying # typed: strict https://srb.help/7027
3 |  KEYS_URI =
"https://api.github.com/meta/public_keys/secret_scanning".freeze
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  Autocorrect: Use `-a` to autocorrect
    lib/github_secret_scanning.rb:3: Replace with T.let(
"https://api.github.com/meta/public_keys/secret_scanning".freeze, String)
...
Errors: 9

$ bundle exec srb tc -a --isolate-error-code 7027
```

https://sorbet.org/docs/error-reference#7027

Constants must have type annotations with `T.let` when specifying `# typed: strict`

lib/github_secret_scanning.rb

```
# typed: strict
class GithubSecretScanning
  KEYS_URI =
    "https://api.github.com/meta/public_keys/secret_scanning".freeze
  KEYS_URI = T.let(
    "https://api.github.com/meta/public_keys/secret_scanning".freeze,
    String
  )
# ...
```

ProTip

You can “quick fix” the 7027 error right in VS Code with ⌘ + .

Type errors in `lib/github_secret_scanning.rb`

```
$ bundle exec spoom tc -s code
6002 - lib/github_secret_scanning.rb:7: Use of undeclared variable @public_key
6002 - lib/github_secret_scanning.rb:11: Use of undeclared variable @public_key
6002 - lib/github_secret_scanning.rb:17: Use of undeclared variable @public_key
7017 - lib/github_secret_scanning.rb:6: This function does not have a sig
7017 - lib/github_secret_scanning.rb:10: This function does not have a sig
7017 - lib/github_secret_scanning.rb:16: This function does not have a sig
7017 - lib/github_secret_scanning.rb:20: This function does not have a sig
7017 - lib/github_secret_scanning.rb:28: This function does not have a sig

Errors: 8
```

Eight errors left, but really only two kinds of error to solve!

https://sorbet.org/docs/error-reference#6002

https://sorbet.org/docs/error-reference#7017

Use of undeclared variable `@public_key` *https://srb.help/6002*

lib/github_secret_scanning.rb

```ruby
def initialize(key_identifier)
  @public_key = self.class.public_key(key_identifier)
end
```

Problem

Sorbet requires that instance variables, like `@public_key`, have a declared type

lib/github_secret_scanning.rb

```ruby
@public_key = T.let(
  self.class.public_key(key_identifier), T.nilable(String)
)
```

Solution

Annotate `@public_key` with `T.let`

Solving this in the right place takes care of all three `6002` errors in this file!

This function does not have a `sig` *https://srb.help/7017*

lib/github_secret_scanning.rb

```ruby
def empty_public_key?
  @public_key.blank?
end
```

Problem

Methods must have signatures when using `# typed: strict`

lib/github_secret_scanning.rb

```ruby
sig { returns(T::Boolean) }
def empty_public_key?
  @public_key.blank?
end
```

Solution

Let's add a signature!

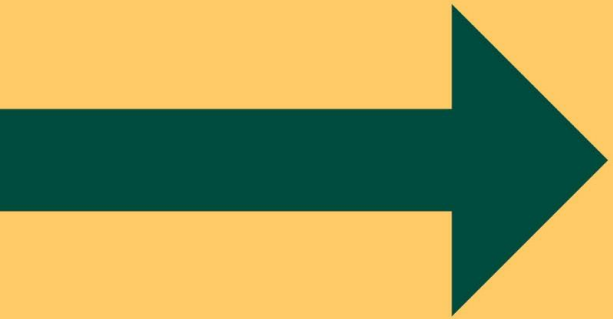
ProTip

Sorbet can suggest the signature for this method since we already typed @public_key.
You can "quick fix" (⌘ + .) it right in the editor!

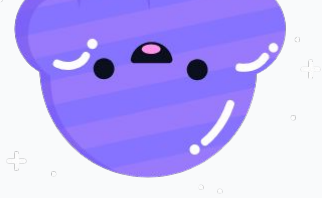
Practice yourself

Move more files to `typed: strict`

Going further



https://github.com/Shopify/rubygems.org/commits/typed-strict



Sorbet Runtime

By default, type errors encountered at runtime will raise an exception

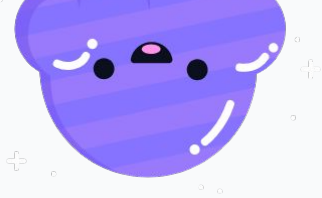
foo.rb

```ruby
# typed: true
extend T::Sig

sig { params(x: Integer).void }
def foo(x)
 puts(x + 1)
end

foo("not an int")
```

Calling the method

```
> foo("not an int")
…
Parameter 'x': Expected type Integer, got
type String with value "not an int"
(TypeError)
```

This catches errors that the static checker might not be able to see,
but can be disruptive in a production environment

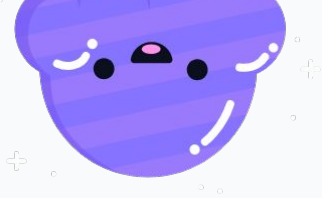


Configuring Sorbet Runtime

Configuration options must be set as early as possible!
In a Rails app, right after requiring `bundler/setup` and `bootsnap/setup`

config/boot.rb

```ruby
# typed: true
ENV["BUNDLE_GEMFILE"] ||= File.expand_path("../Gemfile", __dir__)

require "bundler/setup" # Set up gems listed in the Gemfile
require "bootsnap/setup" # Speed up boot time...
# Configure sorbet-runtime here
```



Configuring Sorbet Runtime

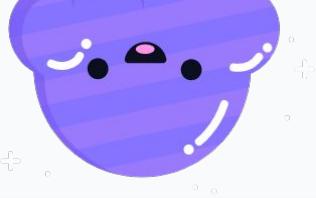
Make a new file `lib/sorbet_config.rb` where we'll do some configuration

lib/sorbet_config.rb

```ruby
# typed: strict

require "sorbet-runtime"

if ENV["RAILS_ENV"] == "production"
 # Configure production sorbet-runtime behaviour
end
```

ProTip

Keep the default behaviour of raising errors in test and development!



Configuration Options

lib/sorbet_config.rb

```ruby
if ENV["RAILS_ENV"] == "production"
  # Configure production sorbet-runtime behaviour
  sorbet_error_handler = lambda do |error, *_|
    Rails.logger.info "SORBET ERROR: #{error.message}"
  end

  # Suppresses errors caused by T.cast, T.let, T.must, etc.
  T::Configuration.inline_type_error_handler = sorbet_error_handler

  # Suppresses errors caused by incorrect parameter ordering
  T::Configuration.sig_validation_error_handler = sorbet_error_handler
end
```

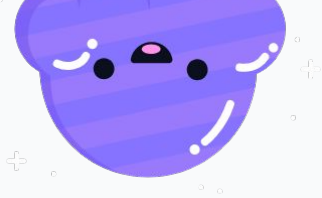
The full list of configuration options can be found in Sorbet's docs

https://sorbet.org/docs/tconfiguration

Configuring Sorbet Runtime

lib/sorbet_config.rb

```ruby
# typed: strict

require "sorbet-runtime"

if ENV["RAILS_ENV"] == "production"
  sorbet_error_handler = lambda do |error, *_|
    Rails.logger.info "SORBET ERROR: #{error.message}"
  end

  # Suppresses errors caused by T.cast, T.let, T.must, etc.
  T::Configuration.inline_type_error_handler = sorbet_error_handler

  # Suppresses errors caused by incorrect parameter ordering
  T::Configuration.sig_validation_error_handler = sorbet_error_handler
end
```



Configuring Sorbet Runtime

Require our configuration file from `config/boot.rb`

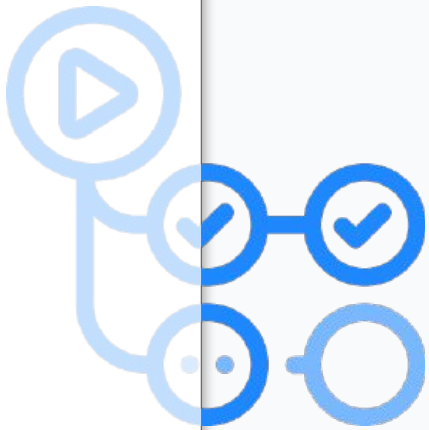
config/boot.rb

```ruby
# typed: true
ENV["BUNDLE_GEMFILE"] ||= File.expand_path("../Gemfile", __dir__)

require "bundler/setup" # Set up gems listed in the Gemfile
require "bootsnap/setup" # Speed up boot time...
require_relative "../lib/sorbet_config" # Configure sorbet-runtime
```

Enabling type checking on CI

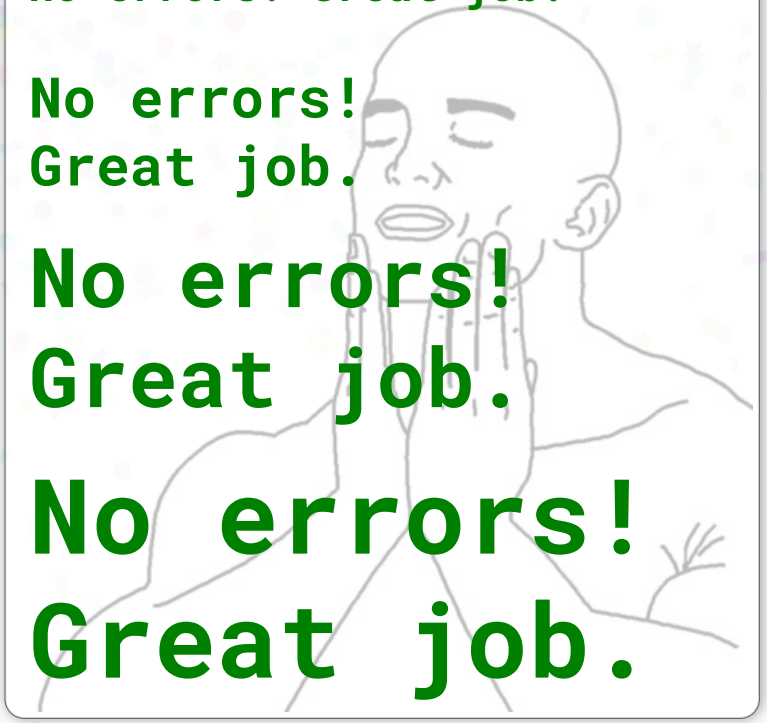
.github/workflows/typechecking.yml

```yaml
# ...
 - name: Run type checking
   run: bundle exec srb tc
 - name: Verify gems RBIs are up-to-date
   run: bin/tapioca gems --verify
 - name: Verify DSL RBIs are up-to-date
   run: bin/tapioca dsl --verify
 - name: Verify shim RBIs are up-to-date
   run: bin/tapioca check-shims
```



Running type checking

```
$ bundle exec srb tc
No errors! Great job.
```



Recap

1. Moving files to `typed: false`
   - Resolving constants (tapioca gems & dsl)
   - Fixing Sorbet limitations
   - Benefits: finding wrong constants
2. Moving files to `typed: true`
   - Defining methods (tapioca annotations, shims)
   - Using type assertions and escape hatches
   - Benefits: finding wrong calls
3. Moving files to `typed: strict`
   - Writing signatures
   - Benefits: finding wrong types
4. Configuring `sorbet-runtime`
5. Enabling Sorbet on CI

Need help?

https://sorbet-ruby.slack.com

channel #tapioca

Need help? https://sorbet-ruby.slack.com #tapioca

Happy Typing!

-- The Ruby & Rails Infrastructure team

We're hiring!

www.shopify.com/careers

Come see us at our booth!