QUICK REFERENCE

Rails Generators Structure & Overview

generators.dev

v1.0-2024-03-17 **FRICTIONLESS GENERATORS**

lib/generators/**<NAME>**/**<NAME>**_generator.rb

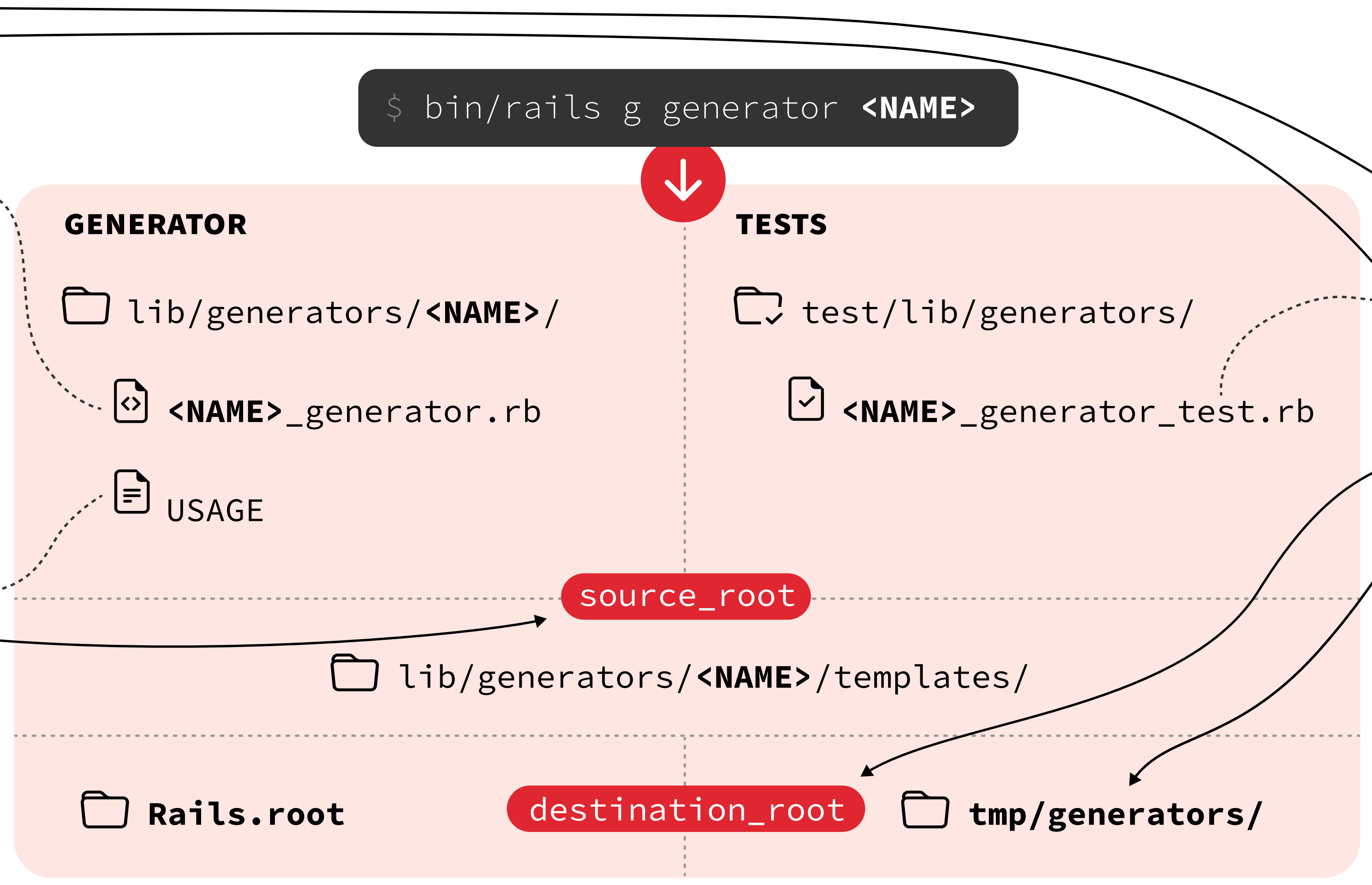
```ruby
class <NAME>Generator < Rails::Generators::NamedBase
  source_root File.expand_path("templates", __dir__)
end
```

lib/generators/**<NAME>**/USAGE

```
Description:
    Explain the generator

Example:
    bin/rails generate <NAME> Thing

    This will create:
        what/will/it/create
```



```
$ bin/rails g generator <NAME>
```

test/lib/generators/**<NAME>**_generator_test.rb

```ruby
require "test_helper"
require "generators/<NAME>/<NAME>_generator"

class <NAME>GeneratorTest < Rails::Generators::TestCase
  tests <NAME>Generator
  destination Rails.root.join("tmp/generators")
  setup :prepare_destination

  # test "generator runs without errors" do
  #   assert_nothing_raised do
  #     run_generator ["arguments"]
  #   end
  # end
end
```

CONFIGURATION
CLASS METHODS

ARGUMENTS, OPTIONS, & HOOKS

```
argument(name, options = {})
class_option(name, options = {})
class_options(options = nil)
hook_for(name, options = {})
remove_argument(*names)
remove_class_option(*names)
remove_command(*names)
remove_hook(*names)
```

VALIDATION & BEHAVIOR

```
class_at_least_one(*args, &block)
class_exclusive(*args, &block)
check_default_type!
allow_incompatible_default_type!
strict_args_position!
check_unknown_options!
check_class_collision(options = {})
```

HELP & DOCUMENTATION

```
desc(description)
hide!()
```

TEMPLATES

ACTIONS

```
template(source, destination)
directory(source, destination)
js_template(source, destination)
migration_template(source, destination)
```

FORMATTING

```
quote(value)
indent(content, multiplier = 2)
indentation()
optimize_indentation(value, amount = 0)
with_indentation(&block)
```

GEMFILE

```
gem(*args)
  version<String>
  comment<String>
  group<String>
  git<String>
gem_group(*names, &block)
add_source(source, options = {}, &block)
github(repo, options = {}, &block)
```

INTERACTIVITY

PROMPTS

```
ask(statement, options)
  default<String>
  limited_to<Array>
  echo<Boolean>
  path<Boolean>
yes?(statement, color = nil)
no?(statement, color = nil)
```

OUTPUT

```
say_status(status, message, log_status = true)
say_error(message = "", color = nil, ...)
say(message = "", color = nil, ...)
mute()
readme(path)
print_in_columns(array)
print_table(array, options = {})
  indent<Integer>
  colwidth<Integer>
  borders<Boolean>
print_wrapped(message, options = {})
  indent<Integer>
```

TESTING

CONFIGURATION (CLASS METHODS)

```
tests(klass)
destination(path)
arguments(array)
setup
teardown
```

GENERATING

```
generator(args, options = {}, config = {})
run_generator(args, config = {})
```

ASSERTIONS

```
assert_file(relative, *contents)
assert_directory(relative, *contents)
assert_no_file(relative)
assert_no_directory(relative)
assert_migration(relative, *contents, &block)
assert_no_migration(relative)
assert_class_method(method, content)
assert_instance_method(method, content)
assert_method(method, content)
```

CREATE & DELETE FILES

RAILS FILES

```
initializer(filename, data = nil)
lib(filename, data = nil)
rakefile(filename, data = nil)
vendor(filename, data = nil)
```

OTHER FILES

```
create_file(destination, data, &block)
add_file(destination, data, &block)
create_link(destination, source)
  symbolic<Boolean>
add_link(destination, source)
create_migration(destination, data, ...)
copy_file(source, destination)
link_file(source, destination)
remove_file(path)
remove_dir(path)
empty_directory(destination)
```

MODIFY FILES

RUBY FILES

```
comment_lines(path, flag)
uncomment_lines(path, flag)
inject_into_class(path, module_name, data)
inject_into_module(path, module_name, data)
```

RAILS FILES

```
environment(data = nil, options = {})
application(data = nil, options = {})
route(routing_code, namespace: nil)
```

TEXT FILES

```
prepend_to_file(path, content)
prepend_file(path, content)
insert_into_file(destination, options)
  before<Integer>
  after<Integer>
inject_into_file(destination, options)
append_to_file(path, content)
append_file(path, content)
gsub_file(path, flag, *args, &block)
```

UTILITIES

RUBY COMMANDS

```
run(command, ...)
run_ruby_script(command, ...)
rails_command(command, options = {})
rake(command, options = {})
generate(generator_name, *args_and_options)
```

DIRECTORY NAVIGATION

```
in_root
inside(dir = "", config = {}, &block)
destination_root
source_root
```

MISCELLANEOUS

```
get(source, destination, http_headers: {})
git(commands = {})
chmod(path, mode, config = {})
find_in_source_paths(file)
```

ADVANCED

APPLICATION/ENGINE NAMESPACE

```
namespaced?
namespace(name = nil)
namespaced_path
namespaced_class_path
wrap_with_namespace(content)
module_namespacing(&block)
mountable_engine?
```

INSPECTION

```
inflection_impossible?(name)
irregular_model_name?(name)
```

This is a free resource to use with Frictionless Generators by Garrett Dimon. It can be helpful on its own, but it works best as a quick reference once you've read the book.