RSpec Ruby testing

START HERE

rspec
Run the whole suite

rspec --only-failures
Rerun what failed

rspec <file>
Run one spec file

rspec --fail-fast
Stop at first failure

rspec <file>:<line>
Run the example at a line

rspec --format doc
Readable output

rspec -e "<text>"
Run by name

rspec --init
Create .rspec and spec_helper

CONCEPTS

--only-failures and --next-failure need a status file set in RSpec.configure; see the detail on --only-failures.

Specs live in spec/ and end in _spec.rb; bare rspec runs spec/**/*_spec.rb.

Example group describe / context
Nested group context, before, let
Example it { expect ... }

Groups nest; metadata and hooks flow inward

Bundler projects: run bundle exec rspec so the Gemfile versions are used.

RUN

rspec
Run all specs

rspec <file> <dir>
Run files or directories

rspec <file>:<line>
Run example or group at that line
Line inside a group runs the whole group.

rspec <file>[1:2]
Run by id (nested index)
Ids like ./a_spec.rb[1:2] are what --bisect prints.

rspec --dry-run
List examples without running them

rspec --version
Show the RSpec version

FILTER

rspec -e "<text>"
Run examples whose name contains text
Matches the full nested description; repeat -e for more.

rspec -E "<regex>"
Run examples matching a regex

rspec --tag <tag>
Run examples with a tag

rspec --tag ~<tag>
Skip examples with a tag
Quote the tilde in some shells.

rspec --tag <key>:<value>
Match tag with a value

rspec --only-failures
Run last run's failures
Set config.example_status_persistence_file_path (e.g. to a spec/ file).

FILTER (cont.)

rspec --next-failure
Fix failures one at a time
Same as --only-failures --fail-fast --order defined.

fit, fdescribe and fcontext mark focus; enable them with filter_run_when_matching in config.

xit, xdescribe and skip mark examples pending; pending still runs and expects a failure.

OUTPUT

rspec --format doc
Nested, readable output

rspec -f p
Dots, the default progress

rspec -f failures
Only failures, one line each

rspec -f json -o <file>
Write JSON results to a file
-o applies to the preceding -f; repeat both for several outputs.

rspec --profile
Show the 10 slowest examples

rspec --profile <n>
Show the n slowest examples

rspec -b
Full backtraces

FAILURES AND ORDER

rspec --fail-fast
Stop after the first failure

rspec --fail-fast=<n>
Stop after n failures

rspec --seed <n>
Rerun with a random order seed
The seed is printed at the end of a randomised run.

rspec --order random
Shuffle example order

rspec --order defined
Run in file order

rspec --bisect
Find the minimal failing set
Needs a failing run; combine with --seed <n>.

WRITING SPECS

RSpec.describe <Class> do
Top-level example group

describe "#<method>" do
Group for one method

context "when <state>" do
Group for one situation

it "<behaviour>" do
One example

WRITING SPECS (cont.)

let(:<name>) { <value> }
Lazy, per-example helper
Evaluated on first use; let! evaluates before each example.

subject { <expr> }
Object under test

before { <setup> }
Run before each example

after { <cleanup> }
Run after each example

before(:all) { <setup> }
Run once per group
State leaks between examples; prefer before(:each) or let.

around { |ex| <x>; ex.run }
Wrap each example

it "<x>", :<tag> do
Tag one example

shared_examples "<name>" do
Define reusable examples

it_behaves_like "<name>"
Include shared examples (nested)

shared_context "<name>" do
Reusable setup

skip "<reason>"
Skip an example

described_class
The class passed to describe

EXPECTATIONS

expect(<x>).to eq(<y>)
Value equals

expect(<x>).not_to eq(<y>)
Value differs

expect(<x>).to be_truthy
Truthy value

expect(<x>).to be_nil
Value is nil

expect(<x>).to be_a(<Class>)
Is instance of class

expect(<x>).to include(<y>)
Collection or string contains

expect(<x>).to match(<regex>)
String matches regex

expect(<x>).to be_within(<d>).of(<n>)
Number near another

expect(<a>).to match_array(<b>)
Same elements, any order

expect { <code> }.to raise_error(<Error>)
Block raises an error

expect { <code> }.to change { <x> }.by(<n>)
Block changes a value

EXPECTATIONS (cont.)

expect(<x>).to be_<predicate>
Calls <predicate>? on the object
be_empty calls empty?, be_valid calls valid?.

MOCKS

double("<name>", <m>: <v>)
Stand-in object

instance_double(<Class>)
Double checked against a class
Fails if the method does not exist on the class.

allow(<obj>).to receive(:<m>).and_return(<v>)
Stub a method

expect(<obj>).to receive(:<m>).with(<args>)
Expect a call

expect(<obj>).to have_received(:<m>)
Assert a call after the fact
Needs allow or a spy first.

CONFIG

.rspec
Default CLI options, one per line
Put --require spec_helper here.

RSpec.configure do |c|
Global configuration block

c.order = :random
Randomise order by default

c.default_formatter = "doc"
Doc output when run on one file

RECIPES

Debug a failing test

```
rspec spec/models/
user_spec.rb:42
rspec --next-failure
rspec -b -e "validates email"
```

Reproduce a flaky order bug

```
rspec --order random   # note
the printed seed
rspec --seed 4711 --bisect
# run the minimal command
bisect prints
```

Focus workflow in spec_helper

```
# spec/spec_helper.rb
RSpec.configure do |c|

c.filter_run_when_matching :focus
  c.order = :random
end
# then: fit "works" do ... end
```

CI with a JSON report

```
bundle exec rspec \
  --format progress \
  --format json --out tmp/
rspec.json \
  --fail-fast=5
```