RSpec Ruby testing

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

rspec --dry-run
List examples without running them

FILTER

rspec -e "*<text>***"**
Run examples whose name contains text

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

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

rspec --tag ~*<tag>*
Skip examples with a tag

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

rspec --only-failures
Run last run's failures

rspec --next-failure
Fix failures one at a time

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

FILTER (cont.)

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 json -o *<file>*
Write JSON results to a file

rspec --profile
Show the 10 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

rspec --order random
Shuffle example order

rspec --order defined
Run in file order

rspec --bisect
Find the minimal failing set

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

let(:*<name>***) { ***<value>* **}**
Lazy, per-example helper

subject { *<expr>* **}**
Object under test

before { *<setup>* **}**
Run before each example

after { *<cleanup>* **}**
Run after each example

WRITING SPECS (cont.)

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

shared_examples "*<name>***" do**
Define reusable examples

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

skip "*<reason>***"**
Skip an example

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

MOCKS

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

instance_double(*<Class>***)**
Double checked against a class

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

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

CONFIG

.rspec
Default CLI options, one per line

RSpec.configure do |c|
Global configuration block