This repository Search

Explore Gist Blog Help

ceparman

hadley / **testthat**

Watch 18 Star 197 Fork 79

branch: **master** testthat / **README.md**

karl-forner-quartz-bio on Jan 8 merged

11 contributors

44 lines (27 sloc) 3.1 kb

Raw Blame History

testthat

build | error BUILD PASSING coverage | 79%

Testing your code is normally painful and boring. `testthat` tries to make testing as fun as possible, so that you get a visceral satisfaction from writing tests. Testing should be fun, not a drag, so you do it all the time. To make that happen, `testthat` :

- Provides functions that make it easy to describe what you expect a function to do, including catching errors, warnings and messages.
- Easily integrates in your existing workflow, whether it's informal testing on the command line, building test suites or using R CMD check.
- Can re-run tests automatically as you change your code or tests.
- Displays test progress visually, showing a pass, fail or error for every expectation. If you're using the terminal, it'll even colour the output.

`testthat` draws inspiration from the xUnit family of testing packages, as well from many of the innovative ruby testing libraries, like rspec, testy, bacon and cucumber. I have used what I think works for R, and abandoned what doesn't, creating a testing environment that is philosophically centred in R.

Instructions for using this package can be found in the Testing chapter of my forthcoming book R packages.

Integration with R CMD check

If you're using testthat in a package, you need to adopt a specific structure to work with `R CMD check` . This structure has changed recently to comply with new demands from CRAN, so please read closely if you submit your packages to CRAN. Previously, best practice was to put all test files in `inst/tests` and ensure that `R CMD check` ran them by putting the following code in `testthat.R` :

```
library(testthat)
library(yourpackage)
test_package("yourpackage")
```

Now, recommended practice is to put your tests in `tests/testthat` , and ensure `R CMD check` runs them by putting the following code in `tests/testthat.R` :

```
library(testthat)
test_check("yourpackage")
```

The advantage of this new structure is that the user has control over whether or not tests are installed using the `--install-tests` parameter to `R CMD install`, or `INSTALL_opts = "--install-tests"` argument to `install.packages()`. I'm not sure why you wouldn't want to install the tests, but now you have the option.

You also need to add `Suggests: testthat` to `DESCRIPTION` as stated in the Writing R Extensions document. This avoids a `R CMD check` warning about unspecified dependencies.