Last Night, Testing Saved my Life
Experiences of a biologist
How I hope you'll feel about testing

You.

You’re gonna be my friend.

My best friend.

Birdstrips on Instagram
Different Kinds of Testing

(According to Steffi)

Due Diligence (Data-testing)

- Double checking output as part of an analysis
- Sometimes interactive only
- Sometimes Rmd Report for posterity (and reproducibility)
Different Kinds of Testing

(According to Steffi)

Due Diligence (Data-testing)

- Double checking output as part of an analysis
- Sometimes interactive only
- Sometimes Rmd Report for posterity (and reproducibility)

Automated Package Tests

- Rigorous, repeated, and automatic tests of package functionality
- Often local and remote (i.e. CRAN, Travis Ci, AppVeyor, R-hub)

For this talk, **Testing = Automated Package Tests**
Why I Started Testing

Post-Doc on RFID project

- Summarize RFID data
  - Movement between feeders
  - Time spent at a feeder
  - etc.
- Started with small scripts
- Created my first package (feedr!)
Why I Started Testing

Post-Doc on RFID project

- Summarize RFID data
  - Movement between feeders
  - Time spent at a feeder
  - etc.
- Started with small scripts
- Created my first package (**feedr**!)

But!

- My colleagues trusted me too much
- Package was getting big and worrisome
  - Mistakes
  - Breaking old code as adding new code

Jacob Bailey
How I Started Testing

- Hadley's "R packages" book chapter on Testing*
- Used `testthat` package
- Series of test files (e.g., `test_utils.R`) in `tests/testthat` folder

*Now 2nd edition by Hadley Wickham and Jennifer Bryan
Hadley's "R packages" book chapter on Testing*

- Used `testthat` package
- Series of test files (e.g., `test_utils.R`) in `tests/testthat` folder

```r
test_that("check_normals() takes 'YYYY-YYYY'", {
  expect_silent(check_normals("1981-2010"))
  expect_error(check_normals("1981 2010"), "text string in the format")
  expect_error(check_normals("1981/2010"), "text string in the format")
  expect_error(check_normals(1981), "text string in the format")
})
```

*Now 2nd edition by Hadley Wickham and Jennifer Bryan
How I Started Testing

- Hadley's "R packages" book chapter on Testing*
- Used `testthat` package
- Series of test files (e.g., `test_utils.R`) in `tests/testthat` folder

```r
test_that("check_normals() takes 'YYYY-YYYY'", {
  expect_silent(check_normals("1981-2010"))
  expect_error(check_normals("1981 2010"), "text string in the format")
  expect_error(check_normals("1981/2010"), "text string in the format")
  expect_error(check_normals(1981), "text string in the format")
})
```

- Automatically run by processes that build the package
  - R CMD CHECK, CRAN, [R-hub](https://r-hub.org)
  - Travis CI, AppVeyor

*Now 2nd edition by Hadley Wickham and Jennifer Bryan*
Why I Keep Testing

meritget

capnbatterete

Okay, @er13_r @lacion_ @stefilazerte @csferrie @cgranade @crazy4pi314 @rensa_co I'm gearing up for the metascience conf this week and I can't stop thinking about meritget.

Pimping my terminal is one of my meritgets. Would *love* to know yours! jenrichmond.github.io/AIMOSworkshop/...
A reward you get that has no value beyond the fact that it's a reward!
Why I Keep Testing

meritget

test_helpers.R:218: failure: format_zero_fill() extra species columns
`b1 <- format_zero_fill(test_rc, verbose = FALSE)` produced warnings.

test_helpers.R:222: failure: format_zero_fill() extra species columns
`... <- NULL` produced warnings.

test_helpers.R:232: warning: format_zero_fill() extra species columns
longer object length is not a multiple of shorter object length

test_helpers.R:234: warning: format_zero_fill() extra species columns
longer object length is not a multiple of shorter object length

--- Results
Duration: 28.2 s

OK: 81
Failed: 9
Warnings: 3
Skipped: 0

Frustration is a natural part of programming :)

Test complete
Why I Keep Testing

@steffilazerte

meritget

```
===> devtools::test()

Loading naturecounts
Testing naturecounts

| OK | FW | S | Context
|----|----|---|-----------
| 15 | YES | NO |
| 104 | YES | NO |
| 113 | YES | NO |
| 24  | YES | NO |
| 52  | YES | NO |
| 95  | YES | NO |
| 36  | YES | NO |
| 28  | YES | NO |
| 120 | YES | NO |
| 14  | YES | NO |
| 20  | YES | NO |

===> Results
Duration: 228.7 s

OK: 621
Failed: 0
Warnings: 0
Skipped: 0
```
Why I Keep Testing

Sometimes it tells me I'm a Coding Rockstar!

meritget

---

```
=> devtools::test()

Loading naturecounts
Testing naturecounts
✓ | OK F W S | Context
✓ | 15 | APIs [6.1 s]
✓ | 104 | Checks [2.9 s]
✓ | 113 | SQLite Databases [15.0 s]
✓ | 24 | Download counts [5.0 s]
✓ | 52 | Download data [151.0 s]
✓ | 95 | Helper/Formatting functions [26.2 s]
✓ | 36 | Metadata [12.9 s]
✓ | 28 | test_requests [1.6 s]
✓ | 120 | Code searches [6.9 s]
✓ | 14 | Server Calls [8.9 s]
✓ | 20 | Utility Functions

Results
Duration: 228.7 s
OK: 621
Failed: 0
Warnings: 0
Skipped: 0
```
Why I Keep Testing

@steffilazerte

```meritget
1  # nocov start
2  praise <- function() {
3      x <- c(
4         "You rock!",
5         "You are a coding rockstar!",
6         "Keep up the good work.",
7         ":)",
8         "Woot!",
9         "Way to go!",
10        "Nice code.",
11        praise::praise("Your tests are ${adjective}!")
12       praise::praise("${EXCLAMATION} - ${adjective} code.")
13    )
14
15    sample(x, 1)
16 }
```

Why I Keep Testing

```r
meritget

30 lines (27 sloc) | 597 Bytes

# nocov start
praise <- function() {
  x <- c(
    "You rock!",
    "You are a coding rockstar!",
    "Keep up the good work.",
    ":)",
    "Woot!",
    "Way to go!",
    "Nice code.",
    praise::praise("Your tests are ${adjective}!!"),
    praise::praise("${EXCLAMATION} - ${adjective} code."))
  sample(x, 1)
}

```
"Testing your code can be painful and tedious, but it greatly increases the quality of your code"

testthat README on GitHub

Too many times I thought:

I don't need to test that, it couldn't have broken anything!
"Testing your code can be painful and tedious, but it greatly increases the quality of your code"

testthat README on GitHub

Too many times I thought:

I don't need to test that, it couldn't have broken anything!

and was so very very wrong!
Why I Keep Testing

Problem solving

- Took over development of a package
  - No tests to start, but really helpful to add tests as I made changes
- Bug fixing
  - Writing tests to check that bug fixes worked
Problem solving

- Took over development of a package
  - No tests to start, but really helpful to add tests as I made changes

- Bug fixing
  - Writing tests to check that bug fixes worked

Organization

- I use testthat tests even for long-running local tests that don't run remotely
  - Keeps me organized

- Test systematically for weird/unexpected input
  - Make sure messages/warnings/errors show up where they should
Remote Testing

- Testing packages on other operating systems
  - Travis Ci, AppVeyor, R-hub
Remote Testing

- Testing packages on other operating systems
  - Travis Ci, AppVeyor, R-hub

Windows I'm looking at YOU!
Remote Testing

- Testing packages on other operating systems
  - Travis Ci, AppVeyor, R-hub

- Using CRON scheduling on Travis Ci to check on and catch changing data sources
  - e.g., run tests every couple of weeks for weathercan
Testing tests??!??!!

- Trouble shooting tests can take time
- Sometimes need to create special (small) datasets for testing (speed up testing)
- Sometimes complex operations take time to figure out how to test

Birdstrips on Instagram
Problems with Testing

Where it's hard to use it

- Difficult with shiny apps
  - But see: shinytest, RSelenium

- Difficult where credentials are needed
  - There are ways of adding credentials to online test suites
  - Create sample accounts with open credentials
  - Test secured credentials locally

Birdstrips on Instagram
Final Tips

- Time consuming, but worth it!
- Start slowly
- Use a coverage metric (e.g., codecov.io)
- Also test for wonky input
- Number test files so they run in order
  - e.g., test_01_base.R, test_02_main.R
- Checkout the resources in this Community Call

Birdstrips on Instagram
@steffilazerte
Final Tips

- Time consuming, but worth it!
- Start slowly
- Use a coverage metric (e.g., codecov.io)
- Also test for wonky input
- Number test files so they run in order
  - e.g., test_01_base.R, test_02_main.R
- Checkout the resources in this Community Call

Thank you!

Presentation Available: https://steffilazerte.github.io/Presentations/
Slides created with the R package xaringan, using remark.js, knitr, and R Markdown
Birdstrips on Instagram