testwhat
taking testing to another level

Filip Schouwenaars
filip@datacamp.com
DataCamp

- Interactive Online Data Science School
- Coding challenges
- Tailored feedback
Example

- Assign 10 to the variable `x`

```r
> a <- 7
> x <- 8
> x <- 10
```

- Incorrect submission
  Did you define `x`?

- Incorrect submission
  It looks like you didn’t assign the correct value to `x`.

Great! Head over to the next exercise!
Unit Testing in R

hypothenuse <- function(a,b) {
  return("I love useR")
}

test_that("hypothenuse works", {
  expect_that(hypothenuse(3,4), equals(5))
  expect_that(hypothenuse(2,0), equals(2))
})

> test_file("test-hypothenuse.r")

1. Failure: hypothenuse() works  ---------
hypothenuse(3, 4) not equal to 5
Mean relative difference: 0.2788897.

testthat (Hadley Wickham)

# Assign 10 to the variable x
x <- 8

test_that("x is defined correctly", {
  expect_that(exists("x"), is_true())
  expect_that(x, equals(10))
})

> test_file("test-exercise.r")

1. Failure: x is defined correctly -------
x not equal to 10
Mean relative difference: 0.2.
But what about...

- A student correctly performed a t-test?

```r
> t.test(df$post, df$pre)
```

- Correctly used .N and by in data.table?

```r
> DT[,N, by=10*round(Sepal.Length*Sepal.Width/10)]
```

- Correctly created ggvis plot?

```r
> pressure %>%
  ggvis(~temperature, ~pressure, fill := ~black) %>%
  layer_points()
```
Unit Testing for DataCamp

- Convenient wrapper functions:
  - `test_object()`
  - `test_function()`
  - `test_outputContains()`
  - ...
  - All use `test_that()` - `expect_that()` construct

- Leverage `testthat`'s resilience
test_object()

```r
# Assign 10 to the variable x
x <- 8

# Assign 10 to the variable x
x <- 10

# SUBMISSION CORRECTNESS TEST
test_object("x")

> test_exercise()
List of 2
$ : logi FALSE
$ : chr "It looks like you didn't assign the correct value to x."
```
test_object()

# Assign 10 to the variable x
x <- 10
x <- (function(){ return(10) })()

# SUBMISSION CORRECTNESS TEST
test_object("x")

> test_exercise()
List of 2
$ : logi TRUE
$ : chr "You’re a coding rockstar!"
test_function()

```r
rnorm(100, sd = 3)
rnorm(100, 0, 3)
rnorm(sd = 3, n = 100)
x <- 100
rnorm(x, sd = 3)
...
```

student

```r
# SUBMISSION CORRECTNESS TEST
test_function("rnorm", args = c("n", "sd"))
```

solution

```r
> test_exercise()
List of 2
$ : logi TRUE
$ : chr "Nice code!"
```
test_function()

\[
\text{rnorm}(90, \text{sd} = 3) \quad \text{student} \\
\text{rnorm}(100, \text{sd} = 3) \quad \text{solution}
\]

# SUBMISSION CORRECTNESS TEST
test_function("rnorm", args = c("n", "sd"))

> test_exercise()
List of 2
$ : logi TRUE
$ : chr "It looks like you didn't set the correct value for the argument n in the function rnorm()."
1. Failure: x is defined correctly ----
x not equal to 10
Mean relative difference: 0.2.

List of 2
$ : logi FALSE
$ : chr "It looks like you didn't assign the correct value to x."
Custom Feedback

# Assign 10 to the variable x
x <- 8

student

# Assign 10 to the variable x
x <- 10

solution

# SUBMISSION CORRECTNESS TEST
test_object("x",
    incorrect_msg = "Segmentation fault!")

> test_exercise()
List of 2
$ : logi FALSE
$ : chr "Segmentation fault!"
More information

- Feel free to contribute
  www.github.com/Data-Camp/testwhat

- One month access to DataCamp
  www.datacamp.com/useR2015

- Create courses for datacamp
  teach.datacamp.com

- We’re hiring!
  www.datacamp.com/careers
Questions?

Filip Schouwenaars
filip@datacamp.com