I recently encountered a small gotcha when using the testthat package to test the equality of two lists. Imagine I have the following two lists below:

list1 <- list(a = 1, b = 2, c = 3, d = 4)

list2 <- list(a = 0, b = 2, c = 3, d = 5)

Imagine that:

1. I know that the value associated with a is going to be different, so I don’t want to test equality for it.
2. The values associated with the other keys should be the same, so I want to test equality for them.

If I run the tests correctly, I should get an error message associated with the key d. The following code snippet using the testthat package looks like it does the job, ***BUT IT DOES NOT WORK***:

library(testthat)

# WRONG!!!!

for (key in names(list1)) {

if (key != "a") {

expect\_equal(list1$key, list2$key)

}

}

The code above incorrectly reports that all the elements in list1 and list2 (ignoring the key a) are equal.

The correct way to test equality in this situation is as follows:

library(testthat)

# CORRECT

keys <- setdiff(names(list1), "a")

expect\_equal(list1[keys], list2[keys])

# Error: list1[keys] not equal to list2[keys].

# Component “d”: Mean relative difference: 0.25

***Update:*** As some of the commenters have rightly pointed out (thanks for contributing!), the bigger mistake in the first snippet is the use of $ to pull out the element instead of the double square braces [[. When using $, it is looking for a key in the list that is literally named "key". Not what I intended!

The code snippet above can be changed to use [[, but notice how the error message is less informative:

library(testthat)

# CORRECT, but error message less informative

for (key in names(list1)) {

if (key != "a") {

expect\_equal(list1[[key]], list2[[key]])

}

}

# Error: list1[[key]] not equal to list2[[key]].

# 1/1 mismatches

# [1] 4 - 5 == -1