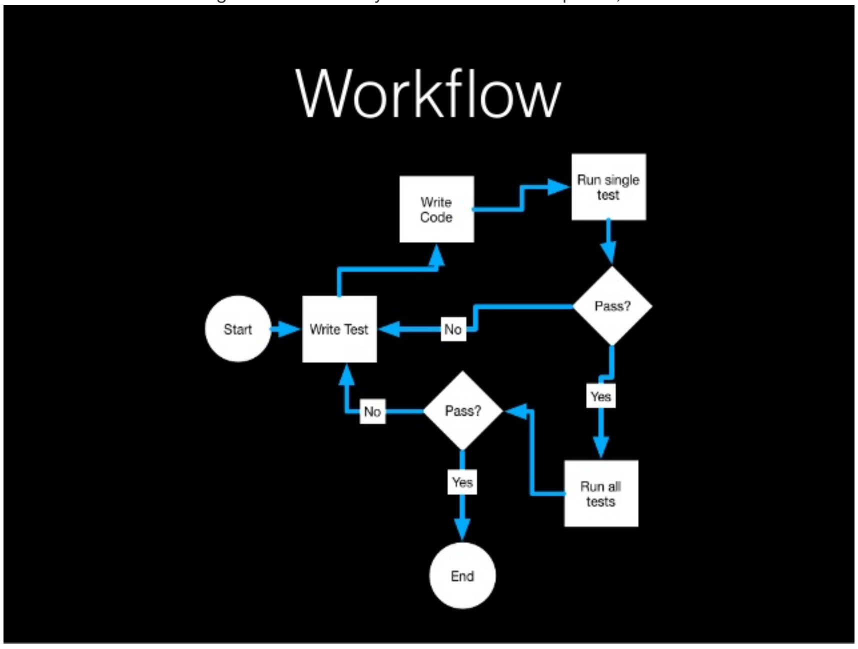
Post 2 - Testing in R

Introduction

Testing is a really important part of all programming. Testing is important because what we think is happening is not always what does happen. Test cases allow us to see edge cases or mistakes that we have made while coding. This post is going to talk about different types of tests that we can use on functions and how to write them using the testthat library. In test driven development, we first write tests then write the functions, like in this diagram below:



Agenda

1. Dependencies
2. General Structure
3. Different testthat types - with code examples
4. Expansions
5. Conclusion
6. Sources

1. Dependencies

To test functions, we will be using the testthat library. To load this into your environment, first install the package using `install.packages("testthat")`, then load the library using `library(testthat)`.

2. General Structure

1. Use `context("whatever function you are testing")`
2. Use `test_that("whatever part of the function you are testing", {actual assertion statements})`
3. Repeat 1 & 2 [1]

So putting it all together it looks like:

```
context("fn")
test_that("fn on some input", {assert that returns correct value on some input})
test_that("fn on other input", {assert that returns correct value on other input})

context("fn2")
test_that("fn2 on some input", {assert that returns correct value on some input})
test_that("fn2 on other input", {assert that returns correct value on other input})
```

3. Different testthat types - with code examples

- `expect_equal`: Checks for equality of values with numerical tolerance [2]

These two values are equal even thought there is 1 more 4 in the second one: `expect_equal(1.234444444, 1.2344444444) #returns true`

- `expect_equivalent`: Checks for equality of values that ignores attributes [3]

```
arr1 <- c(1,2,3)
names(arr1) <- c("hi", "t", "l")
arr2 <- c(1,2,3)
names(arr2) <- c("1", "2", "3")
expect_equivalent(arr1, arr2) #returns true
```

- `expect_match`: Matches a character vector against a regular expression [1]

```
expect_match("Testing is fun", "Testing") #returns true
expect_match("Testing is fun", "testing") #returns false because is case sensitive
expect_match("Testing is fun", "testing", ignore.case = TRUE) #returns true with parameter ignorecase
```

- `expect_identical` : Checks for exact equality [4]

```
arr1 <- c(1,2,3)
names(arr1) <- c("hi", "t", "l")
arr2 <- c(1,2,3)
names(arr2) <- c("1", "2", "3")
expect_match(arr1, arr2) #returns false
```

- `expect_output` : Checks that some part of the output matches the expression [6]

```
expect_output(str(mtcars), "32 obs") #returns true
```

- `expect_error` : Checks that an expression throws an error

```
expect_error(10/0) #returns true
```

- `expect_is` : Checks that an object inherits from a specific class

```
expect_is(5,"numeric")
```

- `expect_true` : Checks to see if an expression is true

```
expect_true(1==1)
```

- `expect_false` : Checks to see if an expression is false

```
expect_false(1==3)
```

- `expect_that` : Can use certain expressions, like equals, is_true, is_false, etc. [5]

```
expect_that(5, equals(5))
```

4. Expansions

There is also the RUnit testing framework that can test a set of functions and store the results of the checks. [7]

5. Conclusion

Testing is a crucial part of programming and we must learn the different types of expects to create comprehensive tests. Hopefully, this post gave you an idea of the most useful expects and interesting caveats especially within the equality expects. You should be able to write and test functions properly after reading this post. Testing will ensure that your code is doing what you think it is doing and will help you catch errors much faster than without testing.

6. Sources

[1] http://r-pkgs.had.co.nz/tests.html

[2] https://tgmstat.wordpress.com/2013/06/26/devtools-and-testthat-r-packages/

[3] http://www.dummies.com/programming/r/how-to-play-with-attributes-in-r/

[4] http://astrostatistics.psu.edu/su07/R/html/base/html/identical.html

[5] http://www.johnmyleswhite.com/notebook/2010/08/17/unit-testing-in-r-the-bare-minimum/

[6] https://github.com/r-lib/testthat/blob/master/R/expect-output.R

[7] https://cran.r-project.org/web/packages/RUnit/RUnit.pdf