



# R PACKAGE VALIDATION WORKSHOP

The Validation Report

# Welcome

- Adding Test Cases
- Adding Test Code
- Validation Reports in R packages

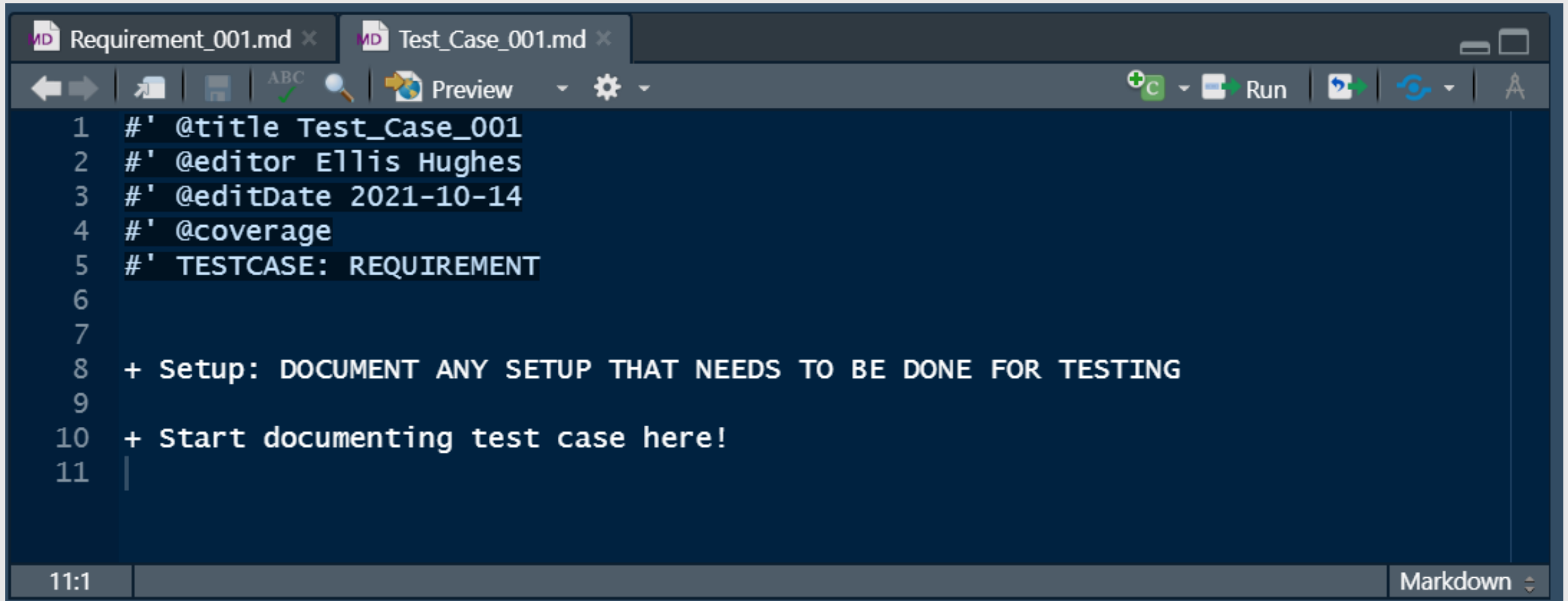


# VALIDATION ELEMENTS

# Test Cases - vt\_use\_test\_case()

- Key Arguments
  - name,
  - username
- Name – requirement name
- Username – Person's name writing test case

# Test Cases - vt\_use\_test\_case()



The image shows a screenshot of a code editor window with two tabs: 'Requirement\_001.md' and 'Test\_Case\_001.md'. The 'Test\_Case\_001.md' tab is active. The editor has a dark blue background and a light blue border. The toolbar at the top includes icons for navigation, editing, and running. The code content is as follows:

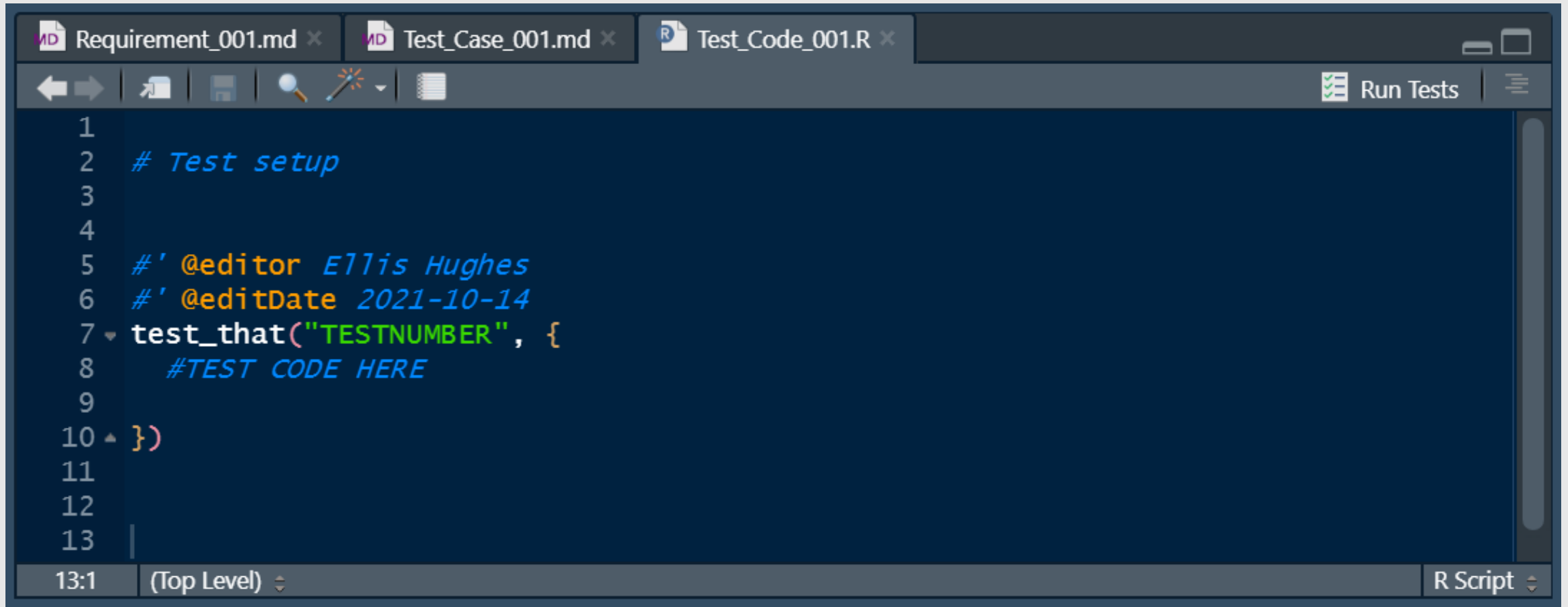
```
1 #' @title Test_Case_001
2 #' @editor Ellis Hughes
3 #' @editDate 2021-10-14
4 #' @coverage
5 #' TESTCASE: REQUIREMENT
6
7
8 + Setup: DOCUMENT ANY SETUP THAT NEEDS TO BE DONE FOR TESTING
9
10 + Start documenting test case here!
11 |
```

The status bar at the bottom shows '11:1' on the left and 'Markdown' on the right.

# Test Code - vt\_use\_test\_code()

- Key Arguments
  - name,
  - username
- Name – requirement name
- Username – Person's name writing test code

# Test Code - vt\_use\_test\_code()



The screenshot shows an RStudio editor window with three tabs: 'Requirement\_001.md', 'Test\_Case\_001.md', and 'Test\_Code\_001.R'. The 'Test\_Code\_001.R' tab is active, displaying a script with the following content:

```
1  
2  # Test setup  
3  
4  
5  #' @editor Ellis Hughes  
6  #' @editDate 2021-10-14  
7  test_that("TESTNUMBER", {  
8    #TEST CODE HERE  
9  
10  })  
11  
12  
13
```

The status bar at the bottom indicates the cursor is at line 13, column 1, and the file is an 'R Script'.



# COMPLETE

## EXERCISE 5 TASK A





# REPORT

# Report basics

- Rmarkdown
- Scrapes information saved across various files
- Produces kables
- Reads in the validation child files and parses them based on order in validation.yml

# Validation Report Helpers

- `vt_path()`
  - `here::here()`, but for validation contents
- `vt_file()`
  - Renders contents based on extensions
    - Rmd parsed
    - Test code executed and results captured
    - Any other file contents are printed

# Child Files

- Validation Report contains pointers for repeated evaluation across different validation environments
- Order of child files can be specified by validation.yml
  - Default is grouped by “requirements”, “test\_cases”, “test\_code” and then alphabetically
  - Control via vt\_get\_child\_files()
- vt\_add\_file\_to\_config() & vt\_drop\_file\_from\_config()
  - Specify validation file (req, test case, test code)
  - Specify which file to add after/before using tidyselect syntax
  - Files added by default to the end by vt\_use\_\*



# COMPLETE

## EXERCISE 5 TASK B

# Creating the validation report Rmd

- `vt_use_report()`
- Creates report from template with suggested contents
- Adds dependencies to package DESCRIPTION
- Two templates included: full validation report and a requirements approval report

# Rendering Reports

- `vt_validate_report()` executes the Rmd and saves output
- Serves as “snapshot” in time

# {valtools} Reports

- Reports are Rmd and therefore infinitely customizable
- Outputs of Rmd from validation to live in new output location
- Serves as “snapshot” in time of package state



# Customization options

- Modify the report RMD generated via template
- Write a report RMD from scratch
- Propose a new template via {valtools} issue/PR
- Switch between multiple RMDs by updating validation.yml as the project cycles through adoption and validation phases



# COMPLETE

## EXERCISE 5 TASK C