

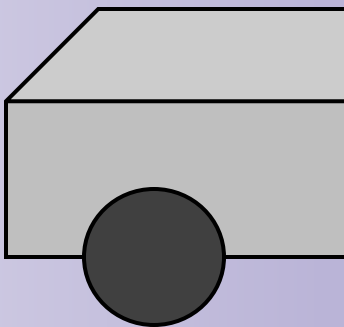


Where the rubber meets the road – Qualifying an R Environment

Ellis Hughes

The Pinewood Derby







Son, we will build the best pinewood derby car.

Sounds like so much fun! I am going to learn so much!

Yes, and because I was a mechanical engineer, I know exactly what we must do. Trust me!

Of course, Dad. Its gonna go so fast. I am going to name it the XR-71 Blackbird!

You know when we raced in the pinewood derby from 1998-2006, we used rubber tires. Those were the Goodyears...



Making a Faster Pinewood Derby Car

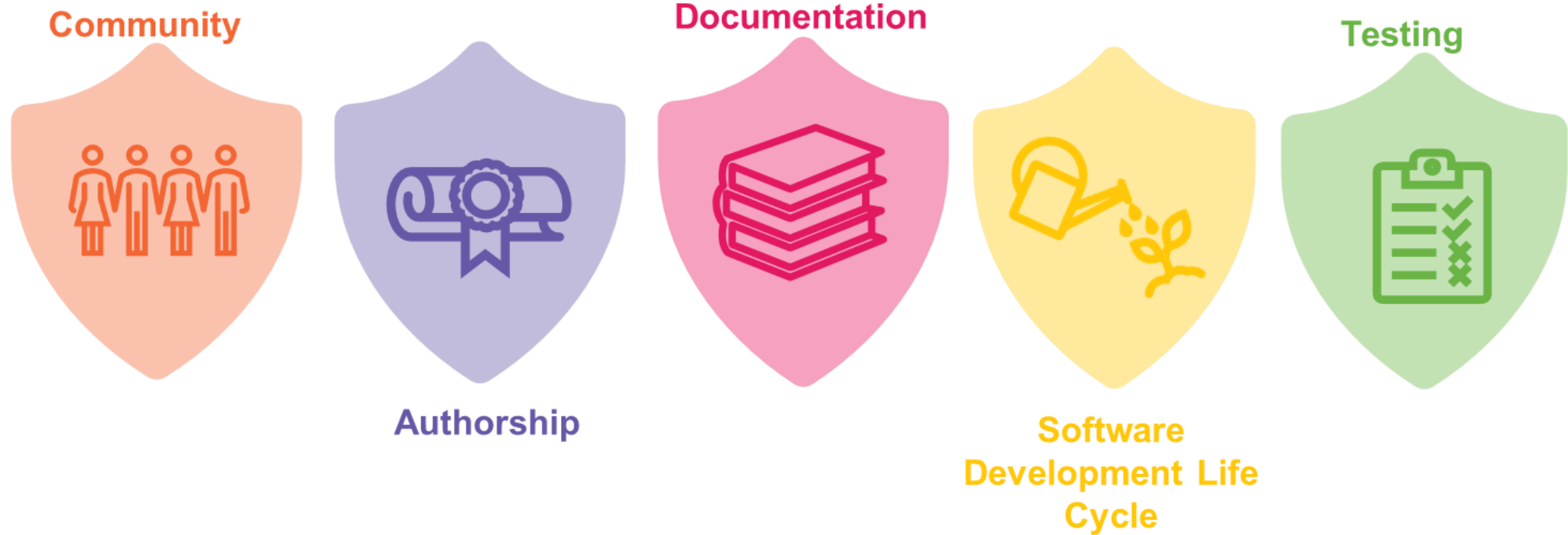


State of the Industry

- Initiatives
 - R Validation Hub
 - Transcelerate Modernization of Analytics
 - PHUSE R Package Validation Framework
- Concerns for both *External* and *Internal* packages
 - Presentations at R/Pharma



5 Pillars of Trust



Becca Krouse - R Package Assessment: Lessons from Pharma

https://bit.ly/Assessment_Pillars

FAKEPACKAGE Assessment

Authorship



- F. Ake, PhD
- University of College, Never-never-land

Documentation



- The one function (fake()) has documented inputs, outputs, examples
- Vignettes are included demonstrating the fake package use cases

Community



- Community is loving using the fake package to do fake things
- An ecosystem of fake packages have been created around this

SDLC



- Versioned and tracked on a web-based git repository
- Continuous Integration
- Code Standards/Formatting rules

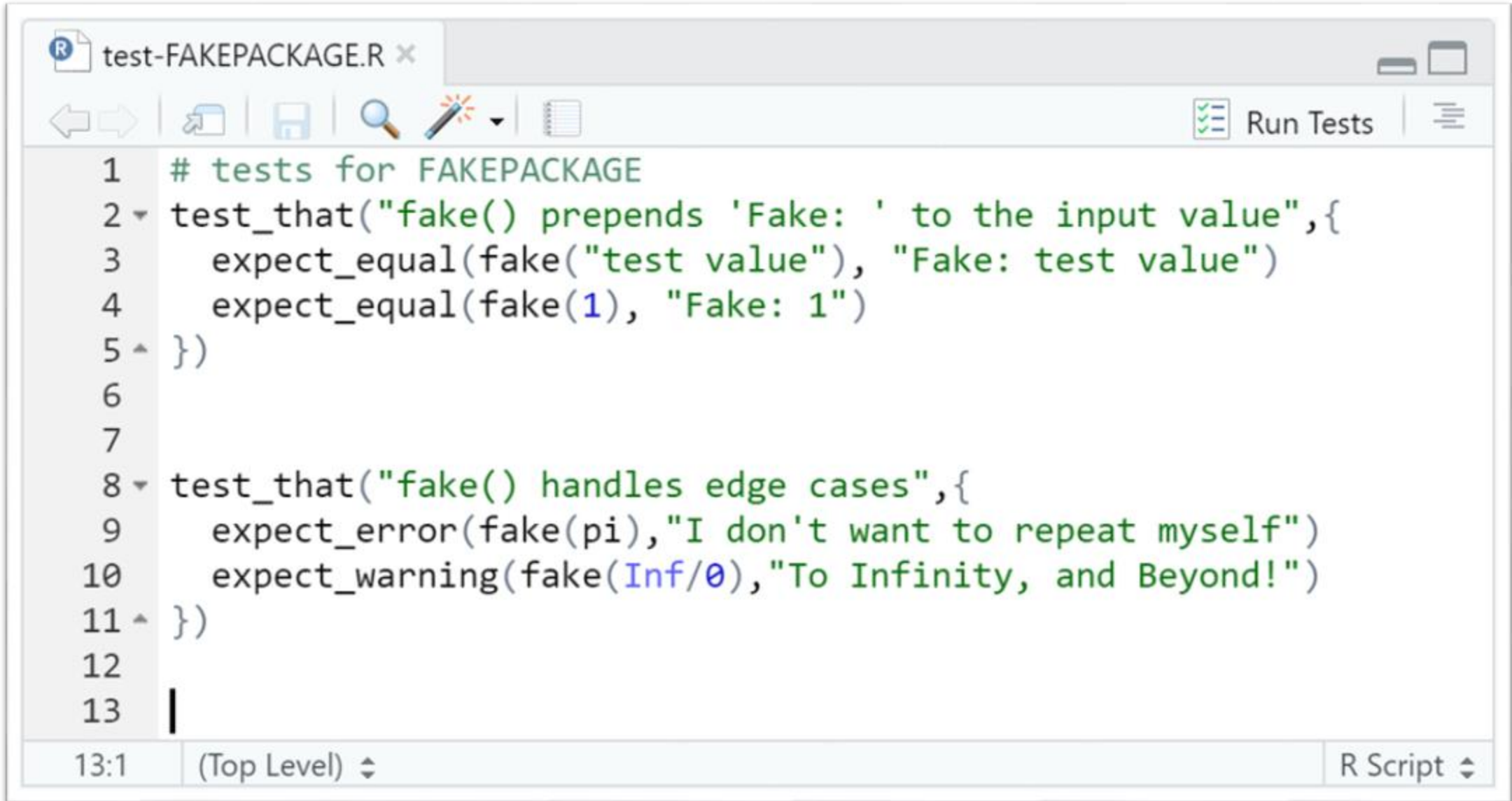
Testing



- Every function is tested for simple cases
- Package is not checked for edge and error cases



FAKEPACKAGE Tests



The screenshot shows an RStudio editor window with a single file named 'test-FAKEPACKAGE.R'. The editor has a toolbar with icons for navigation, file operations, and running tests. The code is written in R and uses the testthat package for testing. It contains two test blocks, each starting with 'test_that()'. The first block tests the 'fake()' function with 'test value' and '1'. The second block tests edge cases with 'pi' and 'Inf/0'. The status bar at the bottom indicates the cursor is at line 13, column 1, at the 'Top Level'.

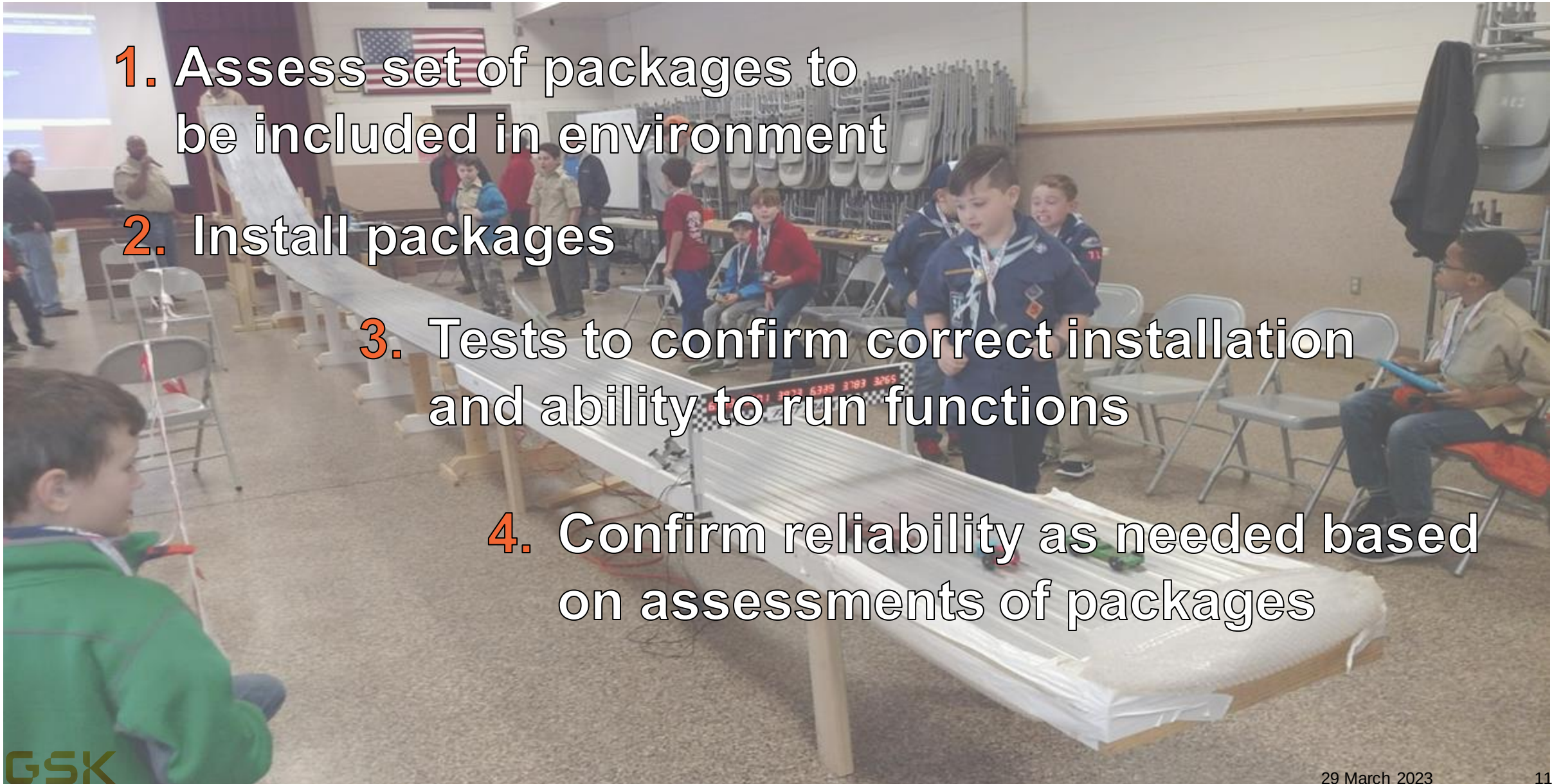
```
1 # tests for FAKEPACKAGE
2 test_that("fake() prepends 'Fake: ' to the input value",{
3   expect_equal(fake("test value"), "Fake: test value")
4   expect_equal(fake(1), "Fake: 1")
5 })
6
7
8 test_that("fake() handles edge cases",{
9   expect_error(fake(pi),"I don't want to repeat myself")
10  expect_warning(fake(Inf/0),"To Infinity, and Beyond!")
11 })
12
13 |
```

13:1 (Top Level) R Script

Validation

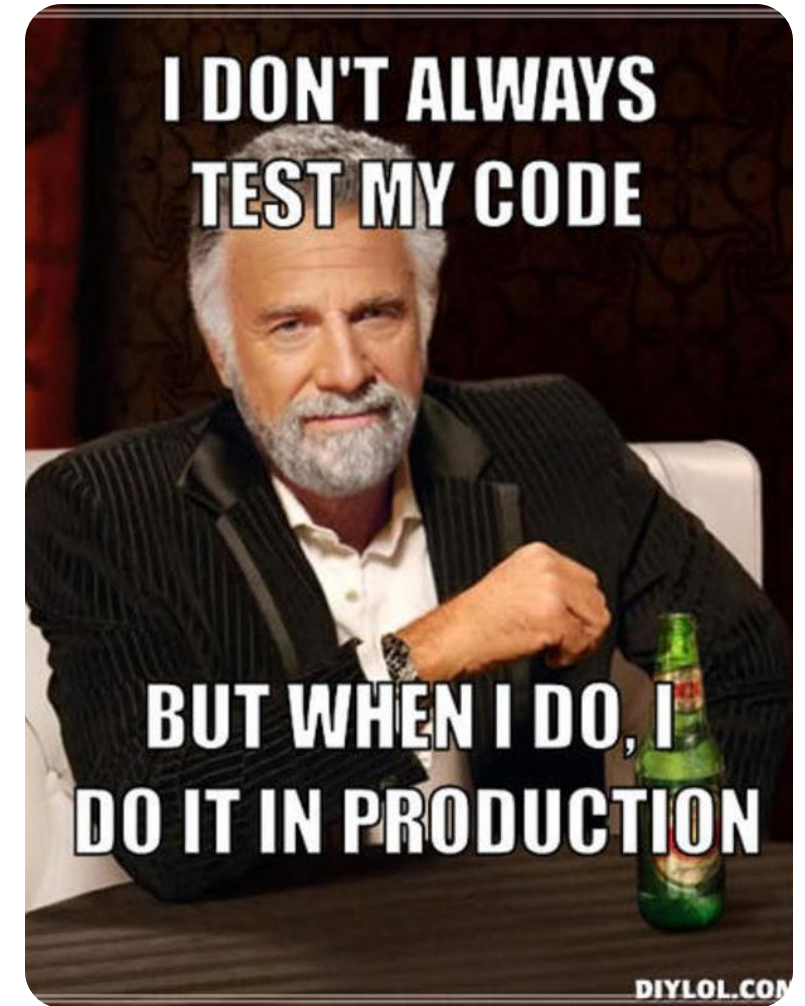
- their approach to validation...takes into consideration the **intended use of the system** and the **potential of the system to affect human subject protection and reliability** of trial results. *(ICH E6 Section 5.5.3)*
- The **credibility of the numerical results** of the analysis depends on the **quality and validity of the methods and software (both internally and externally written)** used both for data management (data entry, storage, verification, correction, and retrieval) and **for processing the data statistically** *(ICH E9 Section 5.8 H)*

Building a Validated Environment

- 
1. Assess set of packages to be included in environment
 2. Install packages
 3. Tests to confirm correct installation and ability to run functions
 4. Confirm reliability as needed based on assessments of packages

Environment Reports

- RMarkdown *in environment being qualified*
- Capture environment information
 - R version
 - Person running report
 - When report is run
- Execute installation and verification tests and record results
- Report can be executed in the command line by team building the system – “Automated”



This Photo by Unknown Author is licensed under CC BY

```
env_report.Rmd x
1 ---
2 title: "Fake Environment Validation"
3 output: pdf_document
4 ---
5
6 # Environment Log
7
8 + Executed by: `r unname(Sys.info()["user"])`
9 + Execution Date: `r Sys.Date()`
10 + R version: `r sessionInfo()$R.version$version.string`
11 + Platform: `r sessionInfo()$platform`, `r sessionInfo()$running`
12
13 # Test Results
14
15 ```{r run-tests, echo = FALSE}
16 test_paths <- c("basic_tests1.R", "basic_tests2.R")
17 test_set <- do.call(rbind, lapply(test_paths, function(i){
18   test_reporter_results <- testthat::test_file(path = i, reporter = testthat::ListReporter)
19   as.data.frame(test_reporter_results)[, c("file", "passed", "failed")]
20 }
21 ))
22 total_passed_tests <- sum(test_set$passed)
23 total_tests <- sum(test_set$passed, test_set$failed)
24
25 A total of `r total_tests` tests were run, with `r round(total_passed_tests/total_tests * 100, 1)`% of tests passing.
26
27 ```{r print-test-results, echo = FALSE}
28 knitr::kable(test_set, row.names = FALSE, col.names = c("Test", "Passed", "Failed"))
29
30
31
32:1 # Test Results
```

Fake Environment Validation

Environment Log

- Executed by: Ellis Hughes
- Execution Date: 2023-01-01
- R version: 4.1.1
- Platform: Windows

Test Results

A total of 5 tests were run, with 100% of tests passing.

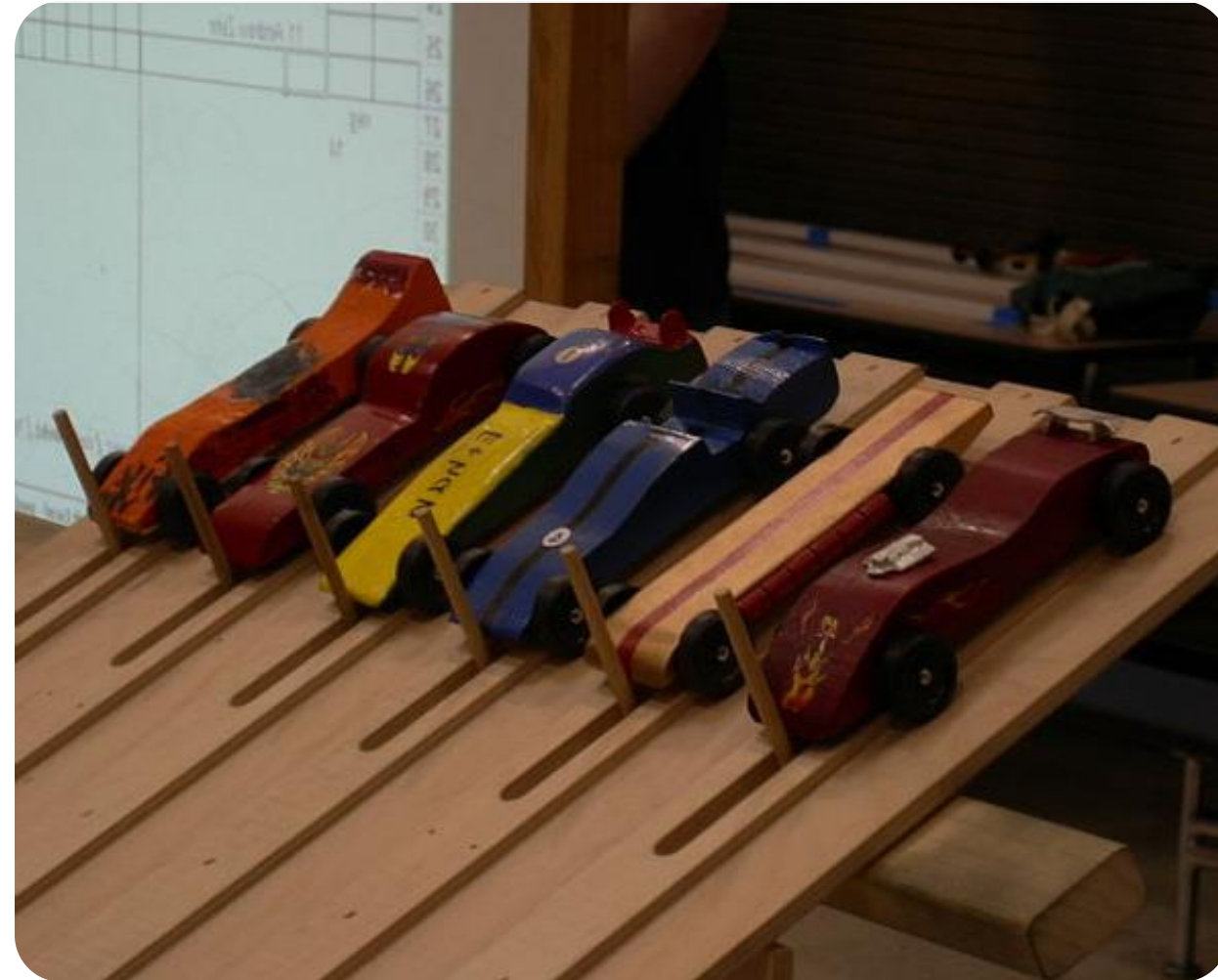
Test	Passed	Failed
basic_tests1.R	2	0
basic_tests2.R	3	0





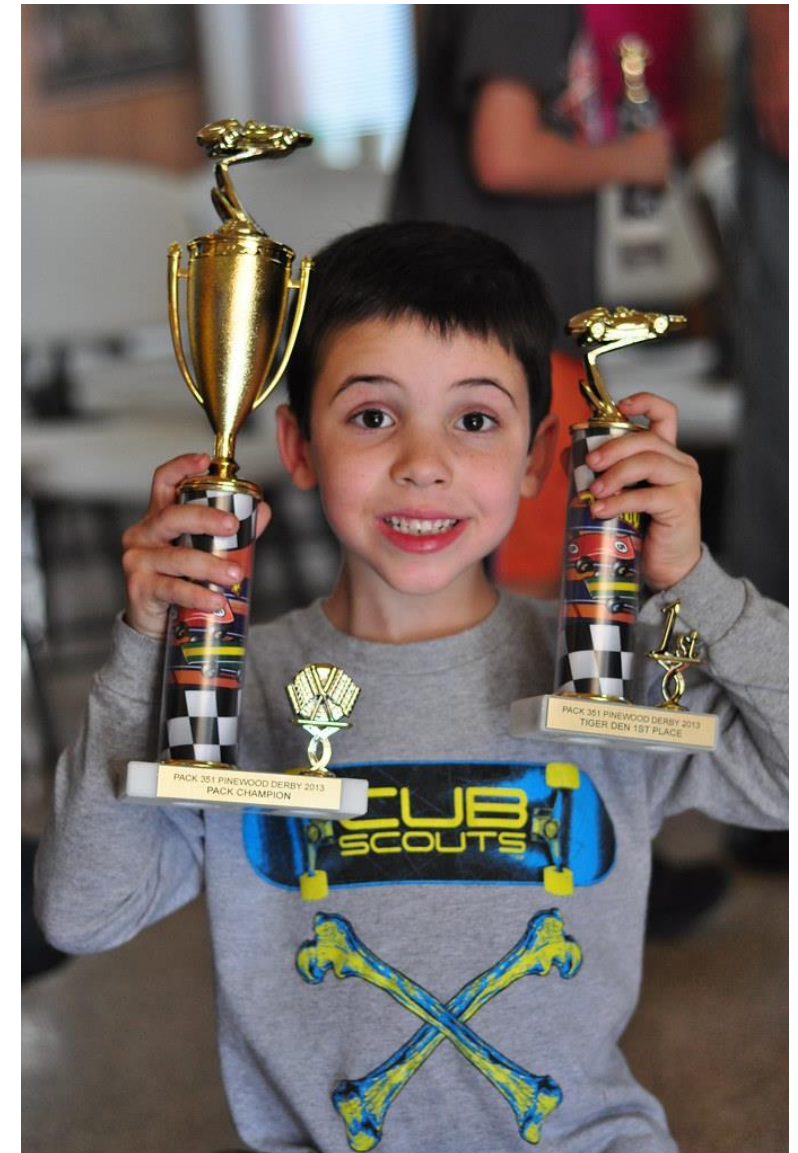
Validation is a lot like building a derby car

- It *can* be complicated
 - Design
 - Tools
 - Implementation and Automation
- Testing should happen at several different times to verify assumptions
 - Deciding on the parts
 - As parts are changed or added
 - Final “Runs”



This Photo by Unknown Author is licensed under CC BY-SA

Now let's let the rubber meet the road and build derby cars



This Photo by Unknown Author is licensed under CC BY
29 March 2023

Now let's let the rubber meet the road and build derby cars...and validated R Environments



This Photo by Unknown Author is licensed under CC BY
29 March 2023