Fake Data Generator :: CHEAT SHEET

Kaiaulu

About : The Fake Data generators provide API to create minimal reproducible examples capturing unusual projects data in order to unit test Kaiaulu parsers & compare analysis to similar tools, and different versions of the same tool behaviors (Green text) ↘

git.R, mail.R, jira.R : Fake Data Generator API's for git, mailing list, jira

example.R : Minimal reproducible examples (MRE) generated on the fly building on fake Data Generators

test-parser.R : unit tests which uses MRE's to unit test behavior of parser functions

git-init → git-commit → git-add → Git Data ← git-add
iO-Sample_file ↗ → Git Data ← iO-create-folder
iO-Sample_file ↘ ↙ Mbox Data ↖
make_sample_mbox_reply ↗   ↖ make_sample_mbox

Jira Data
Jira_function_name#1 ↗   ↖ Jira_function_name #2
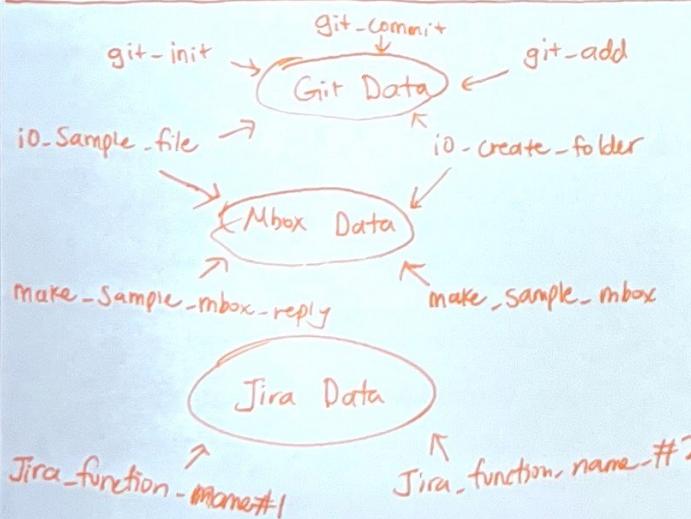
Example.R and MREs

#1
Parse_gitlog (example = example_two_branches, tool = perceval)
Parse_jira (example = two-issues_with_replies)

** Explain what each example function does as well as the expected outcome.

#2
Parse_gitlog (example_two_branches, tool = p 1.0)
Parse_gitlog (example_two_branches, tool = p. 2.0)

The only difference is the version of Perceval, therefore testing the output of various versions of perceval

#3

Integration test

Call mail.R, git.R, and jira.R to simulate a full project that includes development, communication, and issue tracking.

[Visual aid here]

Parse_gitlog (data = example, tool = perceval)
Parse_gitlog (data = example, tool = codeface)
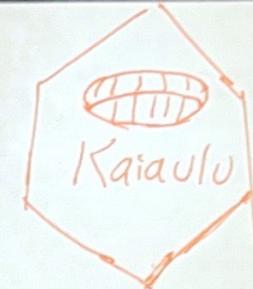
Explain how we are testing both the built-in functions of Kaiaulu AND the behavior of third party tools used within Kaiaulu (like perceval)

Explain that testing different versions of perceval and ensuring they parse data the same way will avoid any time wasted on debugging code.

Able to test everything in the entire process. From beginning to end, how all forms of fake data generators can be used together

Also able to test the output of different tools using the same examples.