LDD Testing Techniques

Jesse Stone, PDS Small Bodies Node

Keeping labels uniform

- Unnecessary variations in the label will make it more difficult to track down errors.
- Sometimes variations are necessary when a discipline area can apply to different data types

Monolithic tests vs granular tests

- EN’s test processes are currently better suited for granular tests
- Monolithic tests are currently easier to generate and maintain

Keeping tests granular

- Each label is invalid in only one way
- Combining multiple errors in a single file will mask errors that don’t occur, since the testing framework is binary.

Drawbacks to granular tests

- Granular tests will increase the number of labels that the LDD is tested against
- Each of these labels will need to be maintained individually

Demonstration - Survey Dictionary Tests

https://github.com/pds-data-dictionaries/ldd-survey/tree/main/test

Objectives

Demonstrate granular tests

ldd-survey

Generating Test Labels

- Hand writing labels
- Injecting discipline area fragments into label templates
  - In addition to making the labels easier to generate, the parts of the label that are being tested are separated from the rest of the label.
- Mutating existing labels
  - Keep a mapping of *XPath*s and operations to perform on a location

Demonstration - LDD Test Generator

https://github.com/sbn-psi/ldd_utilities/tree/master/LddTestGenerator

Objectives

Demonstrate a template-based approach to generating test labels

Mention how the framework could be expanded to mutate test files

generator

Monolithic tests

- Multiple tests can be packed into a single label
- Document each point where the test is expected to fail
- Examine the output of the test run to determine if there are any missed failures

Demonstration - Spectral Dictionary Tests

https://github.com/pds-data-dictionaries/ldd-spectral/tree/main/test

Objectives

Demonstrate monolithic tests

ldd-spectral

Interpreting the test output for monolithic tests

- Since monolithic tests have only a pass/fail result, and there are multiple expected failures, it’s possible to miss failures
- This can be mitigated by expecting a certain number of failures, or checking for specific failure messages
  - This would require updates to the test runner

How many tests?

- You want to have enough to thoroughly test your dictionary.
  - Typically, this means that every class should be used at least once
  - Every schematron rule should pass and fail at least once, as well.
- Too many tests can cause problems (This does *not* mean don't write tests)
  - The biggest problem with too many tests is that they need to be maintained
  - Maintenance can be necessary when either your dictionary changes, or when the dependencies change (IM changes, upstream dictionaries, etc)
  - A test should have its own job – it shouldn't just functionally

Exercise every class

- At least one passing test should use each class
- Write as many test files as necessary to achieve this.

Exercise every schematron rule

- At least one invalid label test should fail each schematron rule.
- At least one valid label test should pass each schematron rule
- At least one valid label test should not trigger the schematron rule, if possible.
- This is especially important, since schematron rules can be prevented from triggering if incorrectly written.

Demonstration - Nucspec Dictionary Tests

https://github.com/pds-data-dictionaries/ldd-nucspec/tree/main/test

Objectives

Demonstrate tests for each schematron rule

ldd-nucspec

Document the tests

- Documentation can be as simple as a file that lists the test name and what it is testing.
- This will remind you how each test is expected to fail, or what each test is intended to exercise.
- If writing a monolithic test, this can be further developed into the expected output for comparison in a future version of the EN testing tool.
- Documentation can also be written inline. It would be valuable to note precisely which line should fail.

Organize the tests

At minimum, tests should be organized into valid and invalid label tests. Although this is embedded in the name, sorting them will make it easier to find the test that you need, especially as the number of tests grows.

Access this presentation

HTML

https://sbn-psi.github.io/dmsp/LDDTesting/LDDTestingTechniques

HTML

PPT

https://github.com/sbn-psi/dmsp/raw/main/LDDTesting/stone-LDDTestingTechniques.pptx

PPT

PDF

https://github.com/sbn-psi/dmsp/raw/main/LDDTesting/stone-LDDTestingTechniques.pdf

PDF