# LDD Testing Overview

Jesse Stone, PDS Small Bodies Node

# Why Tests?

- *Ensure that the dictionary actually works the way you intend*
- Ensure that every class definition works as intended
- Ensures that Schematron tests are running
  - Ensures that your Schematron rules are correct
- Prevent regressions
  - Regressions are unintended side-effects created by making changes
- Provides early-warning if changes are not backwards-compatible

# Testing methodologies

- Regression testing
  - This method generates the dictionary, and validates special labels against the dictionary.
    - These labels are specifically designed to pass or fail validation
    - If the validation result does not match the intent of the label, then there is a problem with the dictionary.
- Static analysis
  - This evaluates the dictionary according to predefined rules, without necessarily comparing it against labels.
- Regression testing and static analysis are complementary tools, and both are needed to fully evaluate a dictionary.

# How do regression tests work

- Upon a push to the repository, GitHub will:
  - Generate and LDD from the ingest file
  - Run validate on every label using the generated LDD
  - Interpret the results
  - Mark the commit as passing/failing based on the results of the test

# Running Tests without GitHub

- You can do this, but it is currently a manual process
- Build the data dictionary with LDDTool
- Run the validate tool on each label in the test directory
  - It would be easier to run the valid and invalid label tests separately
- Examine the output
- Push to GitHub if everything passes
  - To developers, this is known as not breaking the build

# Demonstration - Pushing a dictionary to GitHub

**Objectives**

- Show tests in progress
- Show test results

# What types of tests are there?

- Valid label tests will pass if the validator passes
- Invalid label tests will pass if the validator fails

# Valid Label (passing) tests

- These are meant to test situations where the label should work
- These can consist of a variety of different labels that exercise each aspect of the dictionary

# Invalid Label tests

- These are meant to illustrate labels that are incorrect
  - You would use these to illustrate the type of labels that you *do not* want a data provider to create.
  - The could have incorrect values, be incomplete, or have too much (or conflicting) information.
- Additionally, they will help detect if Schematron rules are not running

# How do you write tests?

- Create a label
  - This could involve creating a completely synthetic label, or using an existing label
  - The simpler the part that is not under test is, the better.
    - Parts that are not being tested just obscure the purpose of the test.
- If this is an invalid label test, introduce errors
- Mark the label as a valid label test or an invalid label test
  - Add either `_VALID` or `_FAIL` to the end of the filename
- Commit the label

# Demonstration - Display Dictionary Tests

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

**Objectives**

Demonstrate simple passing and failing tests.

The display dictionary contains a single type of Schematron rule

- The horizontal display axis and vertical display axis each must match an axis name in the data object.

The tests consist of a valid label and two failing labels

- One where the horizontal display axis name doesn't match
- One where the vertical display axis name doesn't math

ldd-disp

# Static analysis tools

- Validate tool
  - Ingest LDD files are part of the PDS4 information model, just like products. This means that the validator can run against them.
- LDDTool
  - Catches many problems with a dictionary while it is being generated.
- LDDPreflight
  - Runs several of the new rules proposed at this meeting, and displays any violations.

# Using static analysis tools

- These should be run before the regression tests, since errors at this point are easier to catch, and some of them will prevent the dictionary from being generated, or will prevent regression tests from passing.

## Demonstration - LDDPreflight

**Objectives**

Demonstrate using a tool to check for common problems that can be caught without regression tests.

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

preflight

## Access this presentation

**HTML**

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

HTML

**PPT**

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

PPT

**PDF**

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

PDF