Exercises
Test Driven Development II - Advanced
02 - Complex Asserts - 03 - Rulesets

Track lead: Richard Ettema

During this hands-on session of the Test Driven Development II - Advanced tutorial we will examine TestScripts that show the use of Rulesets and the Touchstone Rules Engine.

TestScripts for this exercise are in the **FHIR3-0-1-DevDays17/TDD-2-Advanced/02-ComplexAsserts Test Definitions*

Test Scenario

The use case for this test scenario involves two steps:

1. Create a new Patient resource instance using a known, static fixture
2. Read the created Patient where the FHIR system/server may or may not support resource versioning

The success outcome is that the read HTTP response correctly sends or does not send the appropriate HTTP response headers based on the FHIR system/server's declared support for Patient resource versioning. This support is determined based on the FHIR system/server's CapabilityStatement.

TDD-2-Adv-02-Complex-03-rulesets-patient-read-versioning-[xxx]

Create a new Patient resource fixture in the setup section and then read the created Patient. Demonstrate the use of a Ruleset with a Patient read. The actual test is a read operation for a Patient in JSON format where a ruleset is used to conditionally check the system under test for versioning support and either skip or apply checks for required HTTP response headers.

Features

- Uses the setup section to initialize the FHIR system under test by creating the Patient resource instance needed for the subsequent read test; *examine and become familiar with the TestScript*
- Defines the referenced ruleset; *observe that the same underlying rule is used twice and that the rule parameters are used to customize the rule logic*
- Examine the various read operation asserts that test the returned Patient contents; specifically, the assert that calls the defined ruleset

The focus of this test is to illustrate the use of Rules, Rulesets and the Touchstone Rules Engine to simplify the TestScript assert definitions; i.e. reduce and consolidate the number of individual asserts.

Key Concept: Touchstone Rules Engine and Rule Language Support

- **Touchstone Rules Engine has access to the complete executed operation context**
- **Current scripting languages supported are Groovy, Schematron and XSLT**
  - **Groovy is the recommended language**
  - **Schematron and XSLT are only valid for XML formatted content**
- **The Touchstone Test Definitions UI provides a filter mechanism that allows the user access to uploaded rule and ruleset definitions**

Have fun, and remember to ask questions if you need help!