Including XSLT stylesheet testing in continuous integration process

Balisage 2011

Montreal, Thursday, August 4, 2011

Benoit Mercier

Background information

- Franqus research group (University of Sherbrooke)
- Building a French dictionary describing the French language use in a North American context
- 10+ years project (8 to 25 workers)
- http://franqus.ca/dictio

Intensive use of XML technologies

- Huge DTD (300+ elements)
- 56000+ XML documents
- XML databases (TEXTML, eXist)
- XSLT stylesheets
  - To produce various outputs (HTML, PDF, etc.)
  - To validate data

Software environment

- Open Source Software (desktops and servers)
- Java as the main programming language to develop
    - our web production system (document workflow, drafting resources, etc.)
    - the online electronic version of the dictionary
    - all other supporting custom tools

Adapting to changes

- Dictionary = normative reference material
- After 10 years, we finally (think we) know how to do a dictionary...
- A lot of changes and refactoring
    - DTD
    - XSLT
- Such refactoring are complex and error prone

Challenge

- How to do XSLT stylesheet refactoring with confidence?

In Java world we have tools for...

- Unit testing, to test individual units of code
- Continous integration (CI), to detect integration errors as quickly as possible
- We do it for Java code, why not for XSLT stylesheets?

What is missing?

- XSLT testing framework
- Incorporation of such framework into Continuous Integration process

Choosing an XSLT testing framework

- Test framework should :
    - be easy, intuitive and frictionless
    - require no or minimal Java knowledge
    - allow to test entire document or only fragments
    - be XSLT 2 aware
    - be open source and free

Choosing an XSLT testing framework

And the winner is...

Choosing an XSLT testing framework

And the winner is...

XSpec

http://code.google.com/p/xspec

Authors : Jeni Tennison and Florent Georges

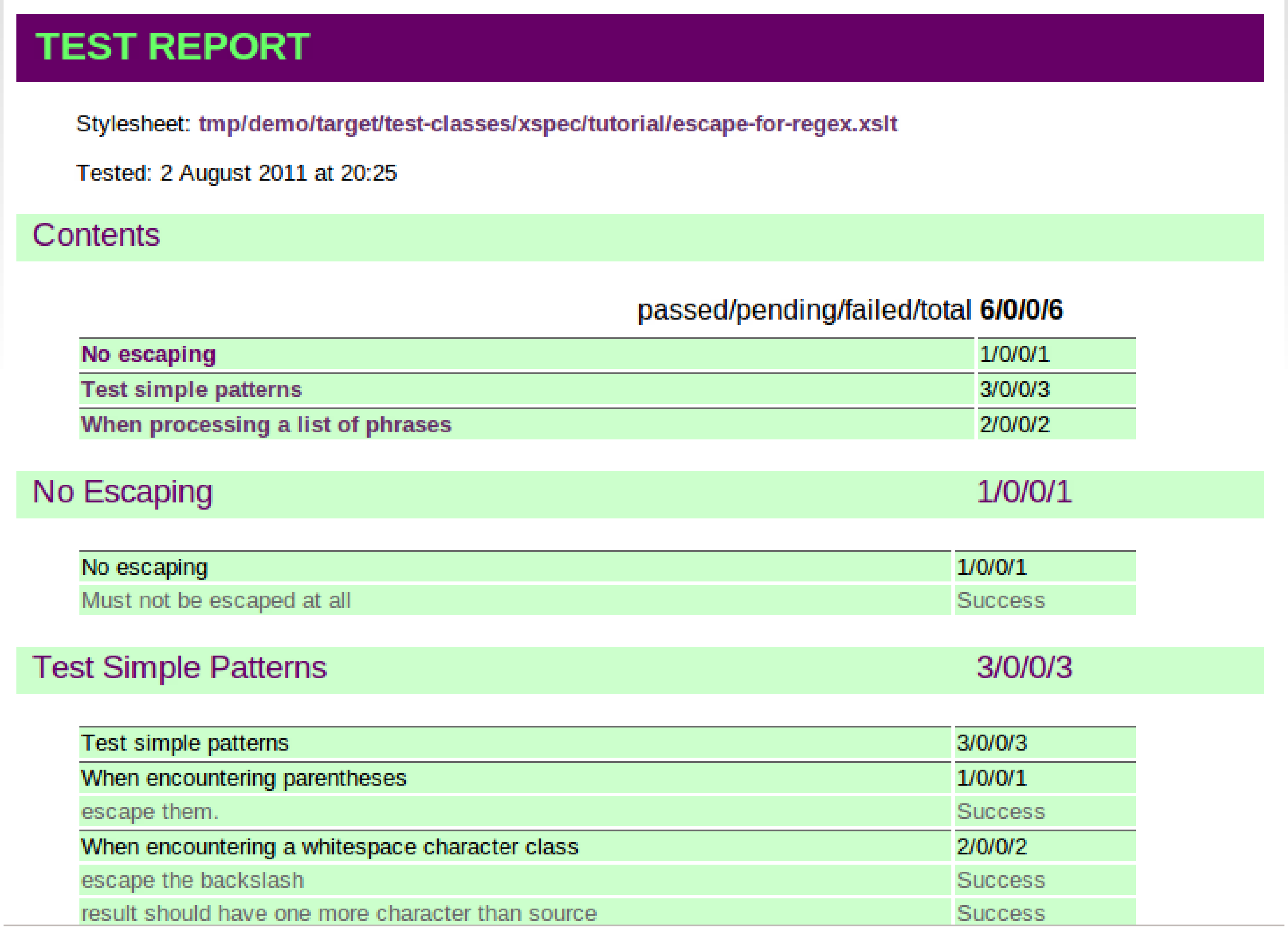
Why XSpec?

- First one we found
- To easy not to be used
- But... Schematron or Tamelizer could do the job

XSpec

- « *XSpec consists of a syntax for describing the behaviour of your XSLT or XQuery code, and some code that enables you to test your code against those descriptions.* » (from XSpec Google code web site)
- XSpec files = XML files
- XSpec file = 1..n test scenario
- Under the hook, XSpec wrapper script use XSLT stylesheet to generate an XSL from XSpec files and applies this generated stylesheet to run the test and produce detailed report.
- (show sample XSpec file)

Xspec report



TEST REPORT

Stylesheet: **tmp/demo/target/test-classes/xspec/tutorial/escape-for-regex.xslt**

Tested: 2 August 2011 at 20:25

Contents

passed/pending/failed/total **6/0/0/6**

No escaping	1/0/0/1
Test simple patterns	3/0/0/3
When processing a list of phrases	2/0/0/2

No Escaping 1/0/0/1

No escaping	1/0/0/1
Must not be escaped at all	Success

Test Simple Patterns 3/0/0/3

Test simple patterns	3/0/0/3
When encountering parentheses	1/0/0/1
escape them.	Success
When encountering a whitespace character class	2/0/0/2
escape the backslash	Success
result should have one more character than source	Success

How to incorporate XSpec into Continuous Integration process?

- By installing a CI server
  (Hudson / Jenkins, Continuum, Bamboo, TeamCity, etc.)
- By triggering XSpec scenario executions from mainstream Java build tools
  (Maven, Ant, etc.)
  - so is born JXsl

    http://code.google.com/p/jxsl
  - would not be necessary with Schematron for Ant

JXsl allows to

- run XSpec tests from your own Java code
  (via XspecTestScenarioRunner or XspecTestSuiteRunner)
- wrap XSpec tests in JUnit tests
- quick start thanks to Maven archetype (demo)
- easily trigger test execution manually
- easily add XSpec testing to existing Java projects (cf. archetype code)
- support Ant via existing JUnit task

How to never forget running tests?

- By triggering test on commit into version control system (demo)

Summary

- Unit testing and continuous integration tools are readily available to the XML technology stack
- Awareness-raising presentation
- Just start writing tests and refactor with confidence!

?