Quick Fixes

George Bina
Overview

• Generic Quick Fix support in oXygen
• Quick Fix support for XSLT
• Quick Fix support for XML validated with
  - XML Schema
  - Relax NG
  - Schematron (Schematron Quick Fix)
• Developing custom Quick Fixes in Schematron
Quick Fixes

• Similar to spell check proposals

• Why do we need Quick Fixes?
  - Not all users know how to fix the errors
  - Sometimes fixing an error can be a lot of manual work
  - Avoid possible additional errors

• Quick Fixes automate the process to fix errors
Proposals to Solve Errors

• Analyze the errors and propose automatic fixes
  
  Examples:
  
  – Required element missing
  – Attribute not allowed

• Write your own quick fixes
  
  – Schematron
See Quick Fixes

- Placing the caret on error highlight will display a red bulb button
- Show fix proposals:
  - Click the red bulb
  - Invoke the quick fix menu with Alt + 1

Most of the information was taken from [www.wikipedia.org](http://www.wikipedia.org), the free encyclopedia.
Apply Quick Fixes

- Just select one of the offered proposals
Quick Fixes in oXygen

- Available in both Text and Author page
- Reconfigured on document modification (as part of the automatic validation)
- Undo Quick Fix actions
XSLT Quick Fixes

- Missing templates
- Misspelled template names
- Missing functions
- References to undeclared variables or parameters
XSLT Quick Fixes Benefits

- Quickly solve errors
- Speed up your work
- Together with the Quick Assist create a complete XSLT development support
XML Quick Fixes

Solve errors that appear in an XML document

- Fixes generated automatically
  - validation against XML Schema
  - validation against Relax NG
- Fixes provided by the developer
  - validation against Schematron
Quick fixes for common errors

- A required attribute is missing
- An element is invalid in the current context
- References to an invalid ID

...
XML Validated with Relax NG

- Quick fixes for common validation errors
  - A required attribute is missing
  - An element is invalid in the current context
  - References to an invalid ID
- Both RNG and RNC schemas are supported
Benefits

- Fixes generated automatically for common errors
- A better understanding of the problem
- Solve errors with just one click
XML Validated with Schematron

- Define custom errors in Schematron
- Create custom fixes with SQF
Example

A group of technical writers works on a documentation project

Authoring team
Rules

Rules to enforce company style

Information architect

Authoring team
Business Rules Examples

⚠️ Bold element is not allowed in title

❗ An external link requires the @format attribute

⚠️ Semicolon is not allowed after list item

❗ Each table row must have the same number of cells

❗ Ordered lists are not allowed
Schematron Rules

Custom rules imposed with Schematron

Custom fixes offered by SQF
Custom Fixes Examples

⚠️ Bold element is not allowed in title
  ✗ Remove bold styling

❗ An external link requires the @format attribute
  ✤ Add @format attribute

⚠️ Semicolon is not allowed after list item
  ✗ Remove semicolon
  ✤ Replace semicolon with full stop

❗ Each row must have the same number of cells
  ✤ Add missing cells

❗ Ordered lists are not allowed
  ❁ Convert ordered list to unordered one
Schematron Quick Fixes (SQF)

- Extends the Schematron language
- SQF initiated by Nico Kutscherauer/data2type
- Open source project

www.schematron-quickfix.com
github.com/schematron-quickfix/sqf
Schematron Quick Fixes Spec

Schematron Quick Fixes Specification

Quick-fix support for XML Community Group - Draft April 2015

This version:

Latest version:
http://schematron-quickfix.github.io/sqf

Editors:
Nico Kutschenauer
Octavian Nadolu

Copyright © 2015, published by the Quick-fix support for XML Community Group under the W3C Community Contributor License Agreement (CLA). A human-readable summary is available.

Abstract

Schematron QuickFix is an extension of the ISO standard Schematron. With Schematron QuickFix the developer is able to define QuickFixes for the Schematron errors. The implementation should offer the user these QuickFixes for the reported Schematron errors. With just one click the user can decide which QuickFix acceptsably fixes the error.

www.w3.org/community/quickfix
schematron-quickfix.github.io/sqf
SQF in Schematron

- Associated with **assert** and **report** elements

  `<sch:rule context="title">
    <sch:report test="exists(b)" sqf:fix="resolveBold">
      Bold element is not allowed in title.</sch:report>
    ..........</sch:rule>

  <sch:rule context="xref">
    <sch:assert test="matches(@href, '^http(s?)://')" sqf:fix="addHttp addHttps">
      An external link should start with http(s).</sch:assert>
    ..........</sch:rule>`
Schematron Quick Fix

**Operation**: ID

**ID**: resolveBold

**Title**: Change the bold element into text

**Description**: Removes the bold (b) markup and keeps the text content.

**Replacement**: match="b" select="text()"
Schematron Quick Fix

- **ID** - an unique ID, used to refer the fix
- **Title** - represents the name of the quick fix
- **Description** - details about the fix
- **Operations** - 4 basic operations
  - **Add** – adds one or more nodes
  - **Delete** – deletes the matched nodes
  - **Replace** – replace a node by another one
  - **StringReplace** – replace sub-strings of text nodes
SQF “add” operation

- `<sqf:add>` element allows you to add one or more nodes to the XML instance

```
<sch:rule context="xref">
  <sch:assert test="@format" sqf:fix="addFormat">
    An external link requires the @format attribute.</sch:assert>

  <sqf:fix id="addFormat">
    <sqf:description>
      <sqf:title>Add @format attribute</sqf:title>
    </sqf:description>
    <sqf:add node-type="attribute" target="format"/>
  </sqf:fix>
</sch:rule>
```
SQF “add” operation

- **Attributes**
  - `sqf.activityManipulate`
  - `match`
  - `node-type`
  - `select`
  - `target`
  - `position`

- **sqf.activityBase**
  - `match`

- **sqf.activityBase**
  - `match`

- **sqf.templateElements**
  - `#any`

The match attribute defines the context for the add operation. If it is missing, then the rule context is used.

Defines the node type of the node to be added: element, attribute, pi, comment, keep.

Defines nodes or an atomic value as the content of the node to be added.

Defines the qualified name of the node to be added.

The position of the nodes to be added relative to the anchor node: alter, before, first-child, last-child.

Describes the content of the node to be added. The behavior is similar to the content of the `<xsl:template>` element. Additionally, the elements `<sch:value-of>` and `<sch:let>` are available.
**SQF “replace” operation**

- `<sqf:replace>` element specify the nodes to be replaced and the replacing content

```
<sch:rule context="ol">
  <sch:assert test="false()" sqf:fix="convertOLinUL"> Ordered lists are not allowed, use unordered lists instead. </sch:assert>

  <sqf:fix id="convertOLinUL">
    <sqf:description>
      <sqf:title>Convert ordered list to unordered list</sqf:title>
    </sqf:description>
    <sqf:replace target="ul" node-type="element" select="child::node()"/>
  </sqf:fix>
</sch:rule>
```
SQF “replace” operation

The match attribute defines nodes to be replaced and the context for the operation. If it is missing, then the rule context is used.

Defines the node type of the node to be added: element, attribute, pi, comment, keep. If is missing a fragment will be inserted.

Defines the qualified name of the node to be added

Defines nodes or values as the content of the node to be added

Describes the content of the node to be added. The behavior is similar to the content of the <xsl:template> element. Additionally, the elements <sch:value-of> and <sch:let> are available.
SQF “delete” operation

- `<sqf:delete>` element specify the nodes for the deletion

```
<sch:rule context="title">  
  <sch:report test="comment()" sqf:fix="removeComment">  
    Comments are not allowed in the title</sch:report>

  <sqf:fix id="removeComment">  
    <sqf:description>  
      <sqf:title>Delete comments</sqf:title>  
    </sqf:description>  
    <sqf:delete match="comment()"/>  
  </sqf:fix>
</sch:rule>
```
SQF “delete” operation

The match attribute defines nodes to be deleted.

The default value is “.”. If the match attribute is not specified the context node will be deleted.
SQF “stringReplace” operation

- `<sqf:stringReplace>` element defines the text or nodes which will replace the matched substrings.

```
<sch:rule context="li">
  <sch:report test="matches(text()[last()], ';$')" sqf:fix="replaceSemicolon" role="warn">
    Semicolon is not allowed after list item.
  </sch:report>

  <sqf:fix id="replaceSemicolon">
    <sqf:description>
      <sqf:title>Replace semicolon with full stop</sqf:title>
    </sqf:description>
    <sqf:stringReplace match="text()" regex=";\$">.</sqf:stringReplace>
  </sqf:fix>
</sch:rule>
```
**SQF “stringReplace” operation**

The match attribute defines text nodes to be processed.

The regex attribute specifies a regular expression that defines the substrings of the text nodes to be replaced.

Defines the nodes or values as the content of the node to be inserted.

Describes the content to be inserted. The behavior is similar to the content of the `<xsl:template>` element. Additionally, the elements `<sch:value-of>` and `<sch:let>` are available.
Formatting Inserted Content

- By default, oXygen performs line indenting
- Use the `@xml:space` attribute and set its value to `preserve` for full control of the generated fragment
- Whitespace is generated similar to XSLT processing:
  - No whitespace for element only content
  - Whitespace is significant in mixed content
Quick Fixes

Formatting Inserted Content

- `<xsl:text>` - format content and keep automatic indentation
  ```xml
  <sqf:add position="last-child">
    <row>
      <entry>First column</entry>
      <entry>Second column</entry>
    </row>
  </sqf:add>
  ```

- `@xml:space` – preserve space
  ```xml
  <sqf:add node-type="element" target="codeblock" xml:space="preserve">
  /* a long sample program */
  Do forever
  Say "Hello, World"
  End
  </sqf:add>
  ```
The Use-When Condition

- Used to make available a *quick fix* or a specific *operation* only if certain conditions are met
- The condition of the *use-when* attribute is an XPath expression

```xml
<sch:rule context="li">
  <sch:report test="matches(text()[last()], ';$')" sqf:fix="replaceSemicolon">
    Semicolon is not allowed after list item.</sch:report>

  <sqf:fix id="replaceSemicolon" use-when="position() = last()">
    <sqf:description>
      <sqf:title>Replace semicolon with full stop</sqf:title>
    </sqf:description>
    <sqf:stringReplace regex=";'$" match="text()">.</sqf:stringReplace>
  </sqf:fix>
</sch:rule>
```
Additional SQF Elements

<sqf:call-fix> – calls another fix within a fix
<sqf:fixes> – global element that contains fixes
<sqf:group> – defines a group of fixes that can be referred
<sqf:keep>* – used to copy the selected nodes
<sqf:user-entry>** – defines a value that must be set manually by the user

*partially supported in <oXygen/> 17
**not supported in <oXygen/> 17

[schematron-quickfix.github.io/sqf](https://schematron-quickfix.github.io/sqf)
SQF Framework

- Support to edit SQF files
- Support to edit SQF embedded in Schematron
- Validation as you type
- Content completion
  - SQF elements and attributes
  - XPath functions
  - XSLT functions and variables
  - Quick Fix ids
- Search and Refactoring
SQF Quick Fixes benefits

- Create custom quick fixes for errors
- Use the power of Schematron and XSLT
- Custom solution for different document types
- Create refactoring actions using SQF
SQF for any XML

• Schematron can be applied on any XML
• Fixes can be provided for any XML

=>

SQF can be used in XML development:

- XSLT
- XML Schema
- Relax NG
...
Projects using SQF

- `<oXygen/>` DITA framework
  
  https://github.com/oxygenxml/userguide

- `<oXygen/>` User Manual
  
  https://github.com/oxygenxml/dim

- DIM project
  
  https://github.com/oxygenxml/dim

- TEI
  
Related SQF Projects

- Escali Schematron engine
  http://schematron-quickfix.com/escali_xsm.html

- Escali Schematron command line tool
- Oxygen plugin for invoking Escali Schematron
Thank you!

Questions?

<oXygen/> XML Editor
http://www.oxygenxml.com
george@oxygenxml.com
@georgebina