xsdTransformer

Generates Code Based On XML Schema

Joerg Moebius

description usable, to be completed.
Release: 0.6 / 2006-11-22

Table of Contents

xsdTransformer

Generates Code Based On XML Schema

What is xsdTransformer?

xsdTransformer constists (resp. should constist) of a set of xsl scripts which generate data structure related code, discriptors et al. based on the content of XML Schemata. Actually a script for generating XForms (xsd2xf.xsl) form is provided. Other scripts will follow.

For applying these script you can use an arbitrary xsl processor. If you haven't got got some within your environment of you can try one of the three possibilities depicted in the following of this desciption:

- run from os prompt with

  Saxon [http://www.saxonica.com/documentation/using-xsl/commandline.html] or

  Xalan [http://xml.apache.org/xalan-j/commandline.html]

- use the `style` task of the *`ant`* [http://ant.apache.org/] framework.

- use the eclipse interface of xsdTransformer x>s>d*`xsdTransformer for eclipse`* [http://xsdeclipse.sourceforge.net/]

xsd2crud - Generates CRUD Applications

Purpose

xsd2crud generates complete CRUD Applications derived from an XML schema.

Quick Start

Prerequisites

As xsd2crud.xsl is a (set of) xslt script(s) you can use it easily at any environment containing an appropriate XSL processor. So at least a current java runtime environment has to be installed (Java 5 includes the XSLTC transformer (based on Xalan 2.6.0 (+ controlled bug fixes))).

Install as follows:

1. copy downloaded directories and files into an arbitrary <installation directory>.

1. kopieren Sie die heruntergeladenen Verzeichnisse und Dateien ein beliebiges <Installationsverzeichnis>.

Run Generation from prompt

proceed the following steps:

- java net.sf.saxon.Transform test1.xsd scripts/xsd2crud.xsl -o output/test1.xhtml (Saxon) or

- java org.apache.xalan.xslt.Process -in test1.xsd -xsl scripts/xsd2crud.xsl -out output/test1.xhtml (Xalan)

The process runs and ends without any message. After the programs terminates you should find a file called test1.xhtml in the output directory.

For further details concerning command-line start refer to :

either *http://www.saxonica.com/documentation/using-xsl/commandline.html* [http://www.saxonica.com/documentation/using-xsl/commandline.html] for using Saxon

or *http://xml.apache.org/xalan-j/commandline.html* [http://xml.apache.org/xalan-j/commandline.html] for using Xalan

Note

Running the transformation in this manner includes a well formed check but not a valid check. So make sure to use only valid schemata.

Approach

xsd2crud is designed to work as simple as possible. You can apply the script without any changes directly on your XML schema and get an result which can be used within further development steps. Though you have the possibility to refine the transformation by customization. For further details see the section 'Reference'.

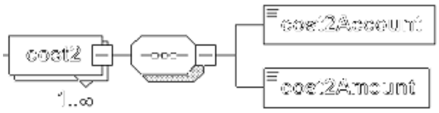
Features

Table 1. Features

XForms control/ construct	Transformation Mode	Status	Annotation
input/output controls			
group	automatic	done	
repeat	automatic	done	
input	automatic	done	
textarea	configure	done	
secret	configure	done	
output	configure	done	
select1	automatic	done	
select	automatic/configure	done	
upload	./.	not planned	
range	./.	not planned	
complex list	?	not yet decided	
generate control controls			
trigger	configure	started	
submit	configure	started	
generate information attributes			
label	automatic/configure	done	
alert	automatic/configure	done	
hint	automatic/configure	done	
help	automatic/configure	done	
generate control attributes			
appearance	automatic/configure	majority done	
navindex	configure	planned	
selection=open	configure	planned	

Transformation Rules

Mainstream of Transformation



❶ element with attribute 'maxOccurs' >1 becomes: repeat control

❷ element with 'sequence' subelement becomes: group control

❷❶ element with 'sequence' subelement and attribute 'maxOccurs' >1 becomes: group control with subelement repeat control

❸ element with 'restriction' subelement becomes: select1 control.

Source

```
              <xs:element name="cost2" maxOccurs="unbounded">❶
  <xs:complexType>
              <xs:sequence>❷
      <xs:element name="cost2Account" default="1002">
        <xs:simpleType>
              <xs:restriction base="xs:integer">❸
            .
            .
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="cost2Amount"
        type="xs:decimal" default="1">
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Target

```
              <xforms:group❷ id="xsd2crud.cost2GroupUI" appearance="full">
.
.

              <xforms:repeat❶ id="xsd2crud.cost2RepeatUI" appearance="compact" bind="B01300">
              <xforms:select1❸ id="xsd2crud.cost2AccountUI" xforms:bind="B01400" xforms:appearance="minima
    .
    .
    <xforms:input id="cost2AmountUI" xforms:bind="B01500">
    </xforms:input>
  </xforms:repeat>
</xforms:group>
```

Transformation Rules in Detail

Table 2. Transformation Rules in Detail

determine appropriate xforms ui control	1	2	3	4	5	6	7	8	9
is an element	n	j	j	j	j	j	j	j	j
has a sequence as child or grandchild before another element as child or grandchild appears	-	n	n	n	n	n	n	j	-
minOccurs<1	-	n	n	n	n	n	?	-	
maxOccurs>1 (incl. unbounded)	-	-	-	-	-	-	?	-	j
has a restriction as child or grandchild before another element as child or grandchild appears	-	n	n	n	n	j	j	-	-
annotation/appinfo/appearance=textarea	-	n	j	-	-	-	-	-	-
annotation/appinfo/appearance=secret	-	n	-	j	-	-	-	-	-

determine appropriate xforms ui control	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
annotation/appinfo/appearance=output	-	n	-	-	j	-	-	-	-
generate group								x	
generate repeat									x
generate input		x							
			x						
generate secret				x					
generate output					x				
generate select1						x			
generate select							x		
generate repeat triggers								x	
do nothing								x	
done		x	x	x	x	x		x	
todo						x	x		

rules 5 and 6 needs some work

Usage

As xsd2crud.xml is a (set of) xslt script(s) you can use it easily at any environment with an appropriate xsl processor.

Run with ant

xsd2crud (scripts and documentation) is still under development. A At first you can try xsd2crud using the ant script `transAddress.xml`.

xsd2fx - generates Xforms

Purpose

xsd2fx generates based on an XML schema a XForms form, which is embedded in an XHTML site.

Quick Start

Prerequisites

As xsd2xf.xsl is a single xslt script you can use it easily at any environment containing an appropriate XSL processor. So at least a current java runtime environment has to be installed.

Install as follows:

1. copy downloaded directories and files into an arbitrary <installation directory>.

Run Generation from os prompt

proceed the following steps:

- java net.sf.saxon.Transform test1.xsd scripts/xsd2xf.xsl -o output/test1.xhtml (Saxon) or
- java org.apache.xalan.xslt.Process -in test1.xsd -xsl scripts/xsd2xf.xsl -out output/test1.xhtml (Xalan)

The process runs and ends without any message. After the programs terminates you should find a file called test1.xhtml in the output directory.

For further details concerning command-line start refer to :

either *http://www.saxonica.com/documentation/using-xsl/commandline.html* for using Saxon

or *http://xml.apache.org/xalan-j/commandline.html* for using Xalan

Note

Running the transformation in this manner includes a well formed check but not a valid check. So make sure to use only valid schemata.

Approach

xsd2fx is designed to work as simple as possible. You can apply the script without any changes directly on your XML schema and get an result which can be used within further development steps. Though you have the possibility to refine the transformation by configuration. For further details see the section 'Usage'.

At least you can also adapt the script to your needs.

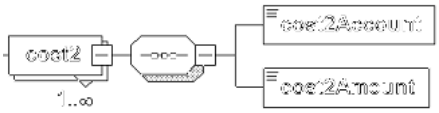
Features

Table 3. Features

XForms control/ construct	Transformation Mode	Status	Annotation
input/output controls			
group	automatic	done	
repeat	automatic	done	
input	automatic	done	
textarea	configure	done	
secret	configure	done	
output	configure	done	
select1	automatic	done	
select	automatic/configure	done	
upload	./.	not planned	
range	./.	not planned	
complex list	?	not yet decided	
generate control controls			
trigger	configure	started	
submit	configure	started	
generate information attributes			
label	automatic/configure	done	
alert	automatic/configure	done	
hint	automatic/configure	done	
help	automatic/configure	done	
generate control attributes			
appearance	automatic/configure	majority done	
navindex	configure	planned	
selection=open	configure	planned	

Transformation Rules

Mainstream of Transformation



- 1 element with attribute 'maxOccurs' >1 becomes: repeat control
- 2 element with 'sequence' subelement becomes: group control
- 2 1 element with 'sequence' subelement and attribute 'maxOccurs' >1 becomes: group control with subelement repeat control
- 3 element with 'restriction' subelement becomes: select1 control.

Source

```
<xs:element name="cost2" maxOccurs="unbounded">1
  <xs:complexType>
    <xs:sequence>2
      <xs:element name="cost2Account" default="1002">
        <xs:simpleType>
          <xs:restriction base="xs:integer">3
            .
            .
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="cost2Amount"
        type="xs:decimal" default="1">
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Target

```
<xforms:group2 id="cost2GroupUI" appearance="full">
.
.

  <xforms:repeat1 id="cost2RepeatUI" appearance="compact" bind="B01300">
    <xforms:select13 id="cost2AccountUI" xforms:bind="B01400" xforms:appearance="minimal">
    .
    .
    <xforms:input id="cost2AmountUI" xforms:bind="B01500">
    </xforms:input>
  </xforms:repeat>
</xforms:group>
```

Transformation Rules in Detail

Table 4. Transformation Rules in Detail

determine appropriate xforms ui control	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
is an element	n	j	j	j	j	j	j	j	j
has a sequence as child or grandchild before another element as child or grandchild appears	-	n	n	n	n	n	n	j	-
minOccurs<1	-	n	n	n	n	n	?	-	
maxOccurs>1 (incl. unbounded)	-	-	-	-	-	-	?	-	j
has a restriction as child or grandchild before another element as child or grandchild appears	-	n	n	n	n	j	j	-	-
annotation/appinfo/appearance=textarea	-	n	j	-	-	-	-	-	-

determine appropriate xforms ui control	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
annotation/appinfo/appearance=secret	-	n	-	j	-	-	-	-	-
annotation/appinfo/appearance=output	-	n	-	-	j	-	-	-	-
generate group								x	
generate repeat									x
generate input		x							
			x						
generate secret				x					
generate output					x				
generate select1						x			
generate select							x		
generate repeat triggers								x	
do nothing								x	
done		x	x	x	x	x		x	
todo						x	x		

rules 5 and 6 needs some work

Usage

As xsd2xf.xml is a single xslt script you can use it easily at any environment with an appropriate xsl processor.

Run with ant

For standalone usage you can use the ant script *`xsdtrans.xml`*. This script consists of three *`targets`*. The first and second one are tranformation processes while the last one is a supplementary task which you can use to parse your input schema.

In *`xsdtrans.xml`* you can also see how to use the start parameters, which will be explained briefly in the following.

Table 5. Brief Description

Parameter	Description	Required
in	location of xml schema file which is to be transformed. (This parameter is used as input for ant's style and as input for the script. The script related parameter is only used for generating messages and can be omitted.) valid values: existent file location including path relative to current directory (path/filename) or an absolute location (/ path/filename) default: none	yes
out	location of target file (here an xhtml file including XForms) (This parameter is used as input for ant's style and as input for the script. The script related parameter is only used for generating messages and can be omitted.)	yes

	valid values: existent file location including path relative to current directory (path/filename) or an absolute location (/ path/filename) default: none	
style	location of name of the transformer script (including path relative to project directory). (This parameter is used as input for ant's style and as input for the script. The script related parameter is only used for generating messages and can be omitted.) valid values: existent file location including path relative to current directory (path/filename) or absolute location (/path/ filename) default: none	yes
force	Recreate target files, even if they are newer than their corresponding source files or the stylesheet. [ant manual] valid values: yes/no default: no	no
inElement	element which is used as root element. The root element is starting point of the transformation. valid values: any string default: none	no
outForm	desired name of the xforms form which is to generate. valid values: any string default: form1	no
language	language in which labels and messages will be transfered to the new form. valid values: en (for english, de (for german) default: en	no
bindPrefix	The id of bind elements is composed of a prefix and a current number. Sample: <xforms:bind id="B00300" xforms:nodeset="vchr:title" xforms:required="true()" xforms:type="string"/> With this parameter you can determine the value of this prefix. valid values: any string	no

	default: B	
minRestrictionsForFull	This parameter determines the appearance of select1 and select controls. Is the amount of restrictions concerning a certain select* control is less than minRestrictionsForFull, the appearance of this control is set to 'minimal' otherwise it is set to 'full'. valid values: any integer default value: 4	
generateAbsentAlert	If this parameter is set to 'yes' and no alert value is determined within the appinfo's (see configuration) the value of the alert element is derived by default from 'minOccurs' and 'type' attributes of the source element. Sample: <xs:element name="title" type="xs:string" default="titleDefaultValue"> (when 'minOccurs' is omitted it means 'minOccurs=1', so input is required) becomes <xforms:input id="title" xforms:bind="B00300"> <xforms:label>title</xforms:label> <xforms:alert>Input is obligative. Only input of type 'string' allowed.</xforms:alert> <xforms:hint>hint for title</xforms:hint> <xforms:help>help for title</xforms:help> </xforms:input> valid values: yes/no default value: yes	no
generateAbsentHint	If this parameter is set to 'yes' and no hint value is determined within the appinfo's (see configuration) a hint element with a place holder value will be created. This value must be manually changed later. Sample: <xs:element name="title"...>	no

	becomes <xforms:input id="title" xforms:bind="B00300"> . <xforms:hint>hint for title</xforms:hint> . </xforms:input> To avoid unnecessary manual work this parameter is set by default to 'no'. valid values: yes/no default value: no	
generateAbsentHelp	If this parameter is set to 'yes' and no help value is determined within the appinfo's (see configuration) a help element with a place holder value will be created. This value must be manually changed later. Sample: <xs:element name="title"...> becomes <xforms:input id="title" xforms:bind="B00300"> . <xforms:help>help for title</xforms:help> . </xforms:input> To avoid unnecessary manual work this parameter is set by default to 'no'. valid values: yes/no default value: no	no

Parse with ant

With the latter target within xsdtrans.xml script you can simply parse your schema.

Ant uses for xml parsing an optional task called 'XMLValidate'. For the prerequisites using this task please refer to http://ant.apache.org/manual/install.html, here the chapter concerning 'Optional Tasks'.

To parse your input schema simply type in at os prompt:

'ant -f xsdTransform.xml xsdparse'

Run within eclipse Einsatz in eclipse

The eclipse userinterface of xsdTransformer is realized as seperate project named xsdeclipse. you can download xsdeclipse plugin either via

xsdeclipse or

xsdtrans

For information concerning the usage within eclipse please refer to the xsdeclipse documentation.

Adapt with `annotation/appinfo` elements

In XML schema you can place schema neutral information within an element called `annotation`. `annotation` can contain two other elements. The first one is the `documentation` and second one is the `appinfo` element. As the names of these element say it is dedicated for documentation and application (controlling) purposes.

For affecting the behaviour of xsd2df you can place particular elements within annotation/appinfo. In detail:

Table 6. Configuration by supplemental `annotation/appinfo` Elements

element	description	use for/within elements
label	Based on the value of this element xsd2xf contributes a nested element called `label` to the control element. If this element is absent and the process paramter `generate absent labels` is set to true, xsd2xf contributes also a nested `label` element and sets its value to the name of the control element. As `label` is an language sensitiv content, a nested element will be created for each language.	all
alert	Based on the value of this element xsd2xf contributes a nested element called `alert` to the control element. If this element is absent and the process paramter `generate absent alerts` is set to true, xsd2xf contributes also a nested `alert` element and sets its value to an appropriate message which is derived from the field contraints (field type, min. occurence) if their are any. As `label` is an language sensitiv content, a nested element will be created for each language.	all editable/changable
hint	Based on the value of this element xsd2xf contributes a nested element called `hint` to the control element. If this element is absent and the process paramter `generate absent hints` is set to true, xsd2xf contributes also a nested `hint` element as place holder for further treatment and sets its value to dummy text.	all

element	description	use for/within elements
help	Based on the value of this element xsd2xf contributes a nested element called `help` to the control element. If this element is absent and the process paramter `generate absent helps` is set to true, xsd2xf contributes also a nested `help` element as place holder for further treatment and sets its value to dummy text.	all
appearance	Usual elements becomes usually `input` control elements in XForms. For affecting this standard you can set this element to: • textarea • secret • output Another case are elements which become `selectx` control elements. In these cases you can overwrite the standard determination of the `appearance` attribute by setting this element to the desired value (`minimal` or `full`)	• elements which become `input` control elements • elements which becomes `selectx` control elements.

If you are working with these supplementary elements and parse you parse your schema or a toll do, you might get error messages concerning these elements. You can avoid the appearance of such messages placinig an internal DTD subset into your XML Schema:

```
<?xml version="1.0"?>
<!DOCTYPE xs:schema PUBLIC "-//W3C//DTD XMLSchema 2001//EN"
  "http://www.w3.org/2001/XMLSchema.dtd" [
  <!ELEMENT label (#PCDATA)>
  <!ELEMENT alert (#PCDATA)>
  <!ELEMENT hint (#PCDATA)>
  <!ELEMENT help (#PCDATA)>
  <!ATTLIST label xml:lang (en | de) "en">
  <!ATTLIST alert xml:lang (en | de) "en">
  <!ATTLIST hint xml:lang (en | de) "en">
  <!ATTLIST help xml:lang (en | de) "en">
]>
<xs:schema targetNamespace="http://xsdtrans.sourceforge.net/costVoucher"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified"
  version="1.0">
  <xs:element name="voucher">
  .
  .
  </xs:element>
</xs:schema>
```

Examples

todo

Considerations / Experiences with Renderers

todo

OPS - Orbeon Presentation Server

Characteristics

- binds have to be flat, otherwise constraints attributes have no effect.
- two (and probably also more than two) input elements with a label element in a group element, which contains also a label element results in the error message:

  'A sequence of more than one item is not allowed as the value of variable $label'.

  Placing each input elements into a <p> element avoids this error message.

Legal Notice

Powered by

[http://sourceforge.net]

[http://docbook.sourceforge.net/]

[http://xmlgraphics.apache.org/fop//]

[http://www.saxonica.com/index.html]