XMLSTARLET USER'S GUIDE

see also http://xmlstar.sourceforge.net/

1. BASIC COMMAND LINE OPTIONS

```
xml
XMLStarlet Toolkit: Command line utilities for XML
Usage: xml [<options>] <command> [<cmd-options>]
where <command> is one of:
   ed   (or edit)      - Edit XML document(s)
   sel  (or select)    - Select data or query XML document(s)
   tr   (or transform) - Transform XML document(s)
   val  (or validate)  - Validate XML document(s)
   fo   (or format)    - Format XML document(s)
<options> are:
   --version           - show version
   --help              - show help
Type: xml <command> --help <ENTER> for command help

XMLStarlet is a command line toolkit to query/edit/check/transform
XML documents (for more information see http://xmlstar.sourceforge.net/)
```

2. Select/Query XML documents

```
xml sel --help
XMLStarlet Toolkit: Select from XML document(s)
Usage: xml sel <global-options> {<template>} [ <xml-file> ... ]
where
  <global-options> - global options for selecting
  <xml-file> - input XML document file name (stdin is used if missing)
  <template> - template for querying XL document with following syntax:

<global-options> are:
  -C                - display generated XSLT
  -R                - print root element <xsl-select>
  -T                - output is text (default is XML)
  -I                - indent output
  -D                - do not omit xml declaration line
  --help            - display help

Syntax for templates: -t|--template <options>
where <options>
  -c or --copy-of <xpath>  - print copy of XPATH expression
  -v or --value-of <xpath> - print value of XPATH expression
  -o or --output <string>  - print string literal
  -n or --nl               - print new line
  -s or --sort <order>     - sort in order (used after -m)
  -m or --match <xpath>    - match XPATH expression

There can be multiple --match, --copy-of, value-of, etc options
in a single template. The effect of applying command line templates
can be illustrated with the following XSLT analogue

xml sel -t -c "xpath0" -m "xpath1" -m "xpath2" -v "xpath3" \
        -t -m "xpath4" -c "xpath5"

is equivalent to applying the following XSLT
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <xsl:call-template name="t1"/>
  <xsl:call-template name="t2"/>
</xsl:template>
<xsl:template name="t1">
  <xsl:copy-of select="xpath0"/>
  <xsl:for-each select="xpath1">
    <xsl:for-each select="xpath2">
      <xsl:value-of select="xpath3"/>
    </xsl:for-each>
```

```
  </xsl:for-each>
</xsl:template>
<xsl:template name="t2">
  <xsl:for-each select="xpath4">
    <xsl:copy-of select="xpath5"/>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>

XMLStarlet is a command line toolkit to query/edit/check/transform
XML documents (for more information see http://xmlstar.sourceforge.net/)

Current implementation uses libxslt from GNOME codebase as XSLT processor
(see http://xmlsoft.org/ for more details)



3. Editing XML documents
=====================================================

xml ed --help
XMLStarlet Toolkit: Edit XML document(s)
Usage: xml ed {<action>} [ <xml-file> ... ]
where <action>
   -d or --delete <xpath>
   -i or --insert <xpath> -t (--type) elem|text|attr -v (--value) <value>
   -a or --append <xpath> -t (--type) elem|text|attr -v (--value) <value>
   -s or --subnode <xpath> -t (--type) elem|text|attr -v (--value) <value>
   -m or --move <xpath1> <xpath2>
   -r or --rename <xpath1> -v <new-name>
   -u or --update <xpath> -v (--value) <value>
                          -x (--expr) <xpath>

XMLStarlet is a command line toolkit to query/edit/check/transform
XML documents (for more information see http://xmlstar.sourceforge.net/)



4. Using XSLT to transform XML documents
=====================================================

xml tr --help
XMLStarlet Toolkit: Transform XML document(s) using XSLT
Usage: xml tr [<options>] <xsl-file> {-p|-s <name>=<value>} [ <xml-file> ... ]
where
   <xsl-file>      - main XSLT stylesheet for transformation
   <xml-file>      - input XML document file name (stdin is used if missing)
   <name>=<value>  - name and value of the parameter passed to XSLT processor
   -p              - parameter is an XPATH expression ("'string'" to quote string)
   -s              - parameter is a string literal
<options> are:
   --omit-decl     - omit xml declaration <?xml version="1.0"?>
   --show-ext      - show list of extensions
   --noval         - do not validate against DTDs or schemas
   --nonet         - refuse to fetch DTDs or entities over network
   --xinclude      - do XInclude processing on document input
   --maxdepth val  - increase the maximum depth
   --html          - input document(s) is(are) in HTML format
   --docbook       - input document(s) is(are) in SGML docbook format
   --catalogs      - use SGML catalogs from $SGML_CATALOG_FILES
                     otherwise XML Catalogs starting from
                     file:///etc/xml/catalog are activated by default

XMLStarlet is a command line toolkit to query/edit/check/transform
XML documents (for more information see http://xmlstar.sourceforge.net/)

Current implementation uses libxslt from GNOME codebase as XSLT processor
(see http://xmlsoft.org/ for more details)



5. Formatting XML documents
=====================================================

xml fo --help
XMLStarlet Toolkit: Format XML document(s)
Usage: xml fo [<options>] <xml-file>
where <options> are
   --indent-tab              - indent output with tabulation
   --indent-spaces <num>     - indent output with <num> spaces
```

```
   --noindent               - do not indent

XMLStarlet is a command line toolkit to query/edit/check/transform
XML documents (for more information see http://xmlstar.sourceforge.net/)
```

6. Validating XML documents

```
xml val --help
XMLStarlet Toolkit: Edit XML document(s)
Usage: xml val <options> [ <xml-file> ... ]
where <options>
   -d or --dtd <dtd-file>  - validate against DTD
   -s or --xsd <xsd-file>  - validate against schema
   -n or --line-num        - print line numbers for validation errors
   -x or --xml-out         - print result as xml
   -w or --well-formed     - check only if XML is well-formed

XMLStarlet is a command line toolkit to query/edit/check/transform
XML documents (for more information see http://xmlstar.sourceforge.net/)
```

7. Examples:

Input1
examples/xml/table.xml

```
<?xml version="1.0"?>
<xml>
  <table>
    <rec id="1">
      <numField>123</numField>
      <stringField>String Value</stringField>
    </rec>
    <rec id="2">
      <numField>346</numField>
      <stringField>Text Value</stringField>
    </rec>
    <rec id="3">
      <numField>-23</numField>
      <stringField>stringValue</stringField>
    </rec>
  </table>
</xml>
```

Input2
examples/xml/tab-obj.xml

```
<?xml version="1.0"?>
<xml>
  <table>
    <rec id="1">
      <numField>123</numField>
      <stringField>String Value</stringField>
      <object name="Obj1">
        <property name="size">10</property>
        <property name="type">Data</property>
      </object>
    </rec>
    <rec id="2">
      <numField>346</numField>
      <stringField>Text Value</stringField>
    </rec>
    <rec id="3">
      <numField>-23</numField>
      <stringField>stringValue</stringField>
    </rec>
  </table>
</xml>
```

Input3
examples/html/hello1.html

```
<html>
```

```
<head>
  <title>Hello World</title>
  <meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
</head>
<body>
  <div align="center">Hello World!<br></div>
</body>
</html>
```

Stylesheet1
examples/xsl/sum1.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text"/>
<xsl:param name="inputFile">-</xsl:param>
<xsl:template match="/">
  <xsl:call-template name="t1"/>
</xsl:template>
<xsl:template name="t1">
  <xsl:value-of select="sum(/xml/table/rec/numField)"/>
  <xsl:value-of select="'&#10;'"/>
</xsl:template>
</xsl:stylesheet>
```

Stylesheet2
examples/xsl/hello1.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text"/>
<xsl:param name="inputFile">-</xsl:param>
<xsl:template match="/">
  <xsl:call-template name="t1"/>
</xsl:template>
<xsl:template name="t1">
  <xsl:for-each select="/">
    <xsl:value-of select="/html/body/div"/>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Stylesheet3
examples/xsl/param1.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text"/>
<xsl:param name="Text"/>
<xsl:param name="Count"/>
<xsl:template match="/">
  <xsl:call-template name="t1"/>
</xsl:template>
<xsl:template name="t1">
  <xsl:for-each select="/xml">
    <xsl:value-of select="$Text"/>
    <xsl:value-of select="$Count"/>
    <xsl:value-of select="'&#10;'"/>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Command:
```
./xmlstarlet sel -T -t -v "count(/xml/table/rec/numField)" -n xml/table.xml
```
Result Output:
```
3
```

Command:
```
./xmlstarlet sel -T -t -m / -c '$inputFile' -o " " -v "count(//node())" -n xml/table.xml xml/tab-obj.xml
```
Result Output:
```
xml/table.xml 32
xml/tab-obj.xml 41
```

```
Command:
./xmlstarlet ed -d /xml/table/rec[@id='2'] xml/table.xml
Result Output:
<?xml version="1.0"?>
<xml>
  <table>
    <rec id="1">
      <numField>123</numField>
      <stringField>String Value</stringField>
    </rec>
    <rec id="3">
      <numField>-23</numField>
      <stringField>stringValue</stringField>
    </rec>
  </table>
</xml>


Command:
./xmlstarlet sel -T -t -m /xml/table/rec/object -c '$inputFile' -n xml/table.xml xml/tab-obj.xml
Result Output:
xml/tab-obj.xml


Command:
./xmlstarlet tr --html xsl/hello1.xsl html/hello1.html
Result Output:
Hello World!

Command:
./xmlstarlet sel -T -t -m "/xml/table/rec[@id='2']" -v numField -n xml/table.xml
Result Output:
346


Command:
cat xml/tab-obj.xml | ./xmlstarlet fo --noindent
Result Output:
<?xml version="1.0"?>
<xml>
<table>
<rec id="1">
<numField>123</numField>
<stringField>String Value</stringField>
<object name="Obj1">
<property name="size">10</property>
<property name="type">Data</property>
</object>
</rec>
<rec id="2">
<numField>346</numField>
<stringField>Text Value</stringField>
</rec>
<rec id="3">
<numField>-23</numField>
<stringField>stringValue</stringField>
</rec>
</table>
</xml>


Command:
./xmlstarlet sel -T -t -v "sum(/xml/table/rec/numField)" -n xml/table.xml
Result Output:
446


Command:
cat xml/tab-obj.xml | ./xmlstarlet fo --indent-tab
Result Output:
<?xml version="1.0"?>
<xml>
        <table>
                <rec id="1">
                        <numField>123</numField>
                        <stringField>String Value</stringField>
                        <object name="Obj1">
                                <property name="size">10</property>
                                <property name="type">Data</property>
```

```
                </object>
        </rec>
        <rec id="2">
                <numField>346</numField>
                <stringField>Text Value</stringField>
        </rec>
        <rec id="3">
                <numField>-23</numField>
                <stringField>stringValue</stringField>
        </rec>
    </table>
</xml>
```

```
Command:
./xmlstarlet sel -T -t -m /xml/table/rec -v "@id" -o "|" -v numField -o "|" -v stringField -n xml/table.xml
Result Output:
1|123|String Value
2|346|Text Value
3|-23|stringValue
```

```
Command:
./xmlstarlet sel -T -t -m /xml/table/rec -v "concat(@id,'|',numField,'|',stringField)" -n xml/table.xml
Result Output:
1|123|String Value
2|346|Text Value
3|-23|stringValue
```

```
Command:
./xmlstarlet sel -T -t -m / -o "====================" -n \
             -m xml/table/rec -v "concat(@id,'|',numField,'|',stringField)" -n \
             -t -m / -o "====================" -n xml/table.xml
Result Output:
====================
1|123|String Value
2|346|Text Value
3|-23|stringValue
====================
```

```
Command:
./xmlstarlet tr xsl/param1.xsl -p Count='count(/xml/table/rec)' -s Text="Count=" xml/table.xml
Result Output:
Count=3
```

```
Command:
./xmlstarlet tr xsl/sum1.xsl xml/table.xml
Result Output:
446
```