# XSLT

**XSLT** (**Extensible Stylesheet Language Transformations**) is a language for transforming XML documents into other XML documents,[1] or other formats such as HTML for web pages, plain text or XSL Formatting Objects, which may subsequently be converted to other formats, such as PDF, PostScript and PNG.[2] XSLT 1.0 is widely supported in modern web browsers.[3]

The original document is not changed; rather, a new document is created based on the content of an existing one.[4] Typically, input documents are XML files, but anything from which the processor can build an XQuery and XPath Data Model can be used, such as relational database tables or geographical information systems.[1]

Although XSLT is designed as a special-purpose language for XML transformation, the language is Turing-complete, making it theoretically capable of arbitrary computations.[5]

**XSLT**

| | |
|---|---|
| **Paradigm** | Declarative |
| **Developer** | World Wide Web Consortium (W3C) |
| **First appeared** | 1998 |
| **Stable release** | 3.0 / June 8, 2017 |
| **Filename extensions** | .xslt |
| **Website** | www.w3.org/TR/xslt-30/ (https://www.w3.org/TR/xslt-30/) |
| **Major implementations** | |
| libxslt, Saxon, Xalan | |
| **Influenced by** | |
| DSSSL | |

## Contents



## History

XSLT is influenced by functional languages,[6] and by text-based pattern matching languages like SNOBOL and AWK. Its most direct predecessor is DSSSL, which did for SGML what XSLT does for XML.[7]

- XSLT 1.0: XSLT was part of the World Wide Web Consortium (W3C)'s eXtensible Stylesheet Language (XSL) development effort of 1998–1999, a project that also produced XSL-FO and XPath. Some members of the standards committee that developed XSLT, including James Clark, the editor, had previously worked on DSSSL. XSLT 1.0 was published as a W3C recommendation in November 1999.[8] Despite its age, XSLT 1.0[9] is still widely used (as of 2018), since later versions are not supported natively in web browsers or for environments like LAMP.

- XSLT 2.0: after an abortive attempt to create a version 1.1 in 2001,[10] the XSL working group joined forces with the XQuery working group to create XPath 2.0,[11] with a richer data model and type system based on XML Schema. Building on this is XSLT 2.0,[12] developed under the editorship of Michael Kay, which reached recommendation status in January 2007.[13] The most important innovations in XSLT 2.0 include:
  - String manipulation using regular expressions
  - Functions and operators for manipulating dates, times, and durations
  - Multiple output documents
  - Grouping (creating hierarchic structure from flat input sequences)
  - A richer type system and stronger type checking

- XSLT 3.0: became a W3C Recommendation on 8 June 2017. The main new features are:[14]
  - Streaming transformations: in previous versions the entire input document had to be read into memory before it could be processed,[15] and output could not be written until processing had finished. XSLT 3.0 allows XML streaming which is useful for processing documents too large to fit in memory or when transformations are chained in XML Pipelines.
  - Packages, to improve the modularity of large stylesheets.
  - Improved handling of dynamic errors with, for example, an xsl:try instruction.
  - Support for maps and arrays, enabling XSLT to handle JSON as well as XML.
  - Functions can now be arguments to other (higher-order) functions.

# Design and processing model

The XSLT processor takes one or more XML source documents, plus one or more XSLT stylesheets, and processes them to produce an output document. In contrast to widely implemented imperative programming languages like C, XSLT is declarative.[16] The basic processing paradigm is pattern matching.[17] Rather than listing an imperative sequence of actions to perform in a stateful environment, template rules only define how to handle a node matching a particular XPath-like pattern, if the processor should happen to encounter one, and the contents of the templates effectively comprise functional expressions that directly represent their evaluated form: the result tree, which is the basis of the processor's output.

A typical processor behaves as follows. First, assuming a stylesheet has already been read and prepared, the processor builds a source tree from the input XML document. It then processes the source tree's root node, finds the best-matching template for that node in the stylesheet, and evaluates the template's contents. Instructions in each template generally direct the processor to either create nodes in the result tree, or to process more nodes in the source tree in the same way as the root node. Finally the result tree is serialized as XML or HTML text.

# XPath

XSLT uses XPath to identify subsets of the source document tree and perform calculations. XPath also provides a range of functions, which XSLT itself further augments.

XSLT 1.0 uses XPath 1.0, while XSLT 2.0 uses XPath 2.0. XSLT 3.0 will work with either XPath 3.0 or 3.1. In the case of 1.0 and 2.0, the XSLT and XPath specifications were published on the same date. With 3.0, however, they were no longer synchronized; XPath 3.0 became a Recommendation in April 2014, followed by XPath 3.1 in February 2017; XSLT 3.0 followed in June 2017.

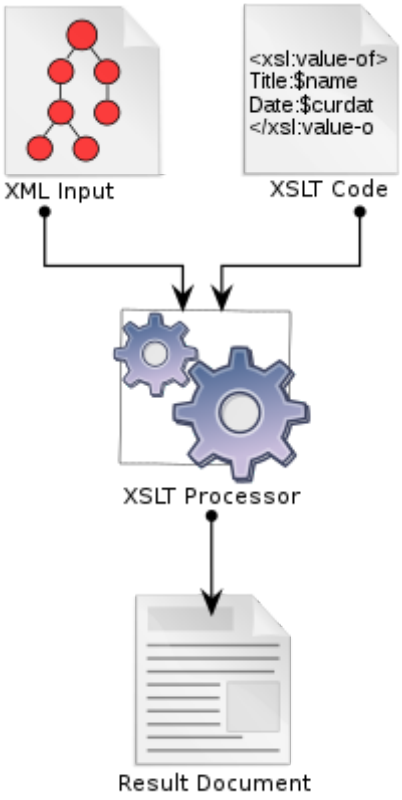



Diagram of the basic elements and process flow of eXtensible Stylesheet Language Transformations.

## XQuery compared

XSLT functionalities overlap with those of XQuery, which was initially conceived as a query language for large collections of XML documents.

The XSLT 2.0 and XQuery 1.0 standards were developed by separate working groups within W3C, working together to ensure a common approach where appropriate. They share the same data model, type system, and function library, and both include XPath 2.0 as a sublanguage.

The two languages, however, are rooted in different traditions and serve the needs of different communities. XSLT was primarily conceived as a stylesheet language whose primary goal was to render XML for the human reader on screen, on the web (as a web template language), or on paper. XQuery was primarily conceived as a database query language in the tradition of SQL.

Because the two languages originate in different communities, XSLT is stronger in its handling of narrative documents with more flexible structure, while XQuery is stronger in its data handling, for example when performing relational joins.

## Media types

The `<output>` element can optionally take the attribute `media-type`, which allows one to set the media type (or MIME type) for the resulting output, for example: `<xsl:output output="xml" media-type="application/xml"/>`. The XSLT 1.0 recommendation recommends the more general attribute types `text/xml` and `application/xml` since for a long time there was no registered media type for XSLT. During this time `text/xsl` became the de facto standard. In XSLT 1.0 it was not specified how the `media-type` values should be used.

With the release of the XSLT 2.0, the W3C recommended the registration of the MIME media type `application/xslt+xml`[18] and it was later registered with the Internet Assigned Numbers Authority.[19]

Pre-1.0 working drafts of XSLT used `text/xsl` in their embedding examples, and this type was implemented and continues to be promoted by Microsoft in Internet Explorer[20] and MSXML. It is also widely recognized in the `xml-stylesheet` processing instruction by other browsers. In practice,

therefore, users wanting to control transformation in the browser using this processing instruction are obliged to use this unregistered media type.[21]

# Examples

These examples use the following incoming XML document

```
<?xml version="1.0" ?>
<persons>
  <person username="JS1">
    <name>John</name>
    <family-name>Smith</family-name>
  </person>
  <person username="MI1">
    <name>Morka</name>
    <family-name>Ismincius</family-name>
  </person>
</persons>
```

## Example 1 (transforming XML to XML)

This XSLT stylesheet provides templates to transform the XML document:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml" indent="yes"/>

  <xsl:template match="/persons">
    <root>
      <xsl:apply-templates select="person"/>
    </root>
  </xsl:template>

  <xsl:template match="person">
    <name username="{@username}">
      <xsl:value-of select="name" />
    </name>
  </xsl:template>

</xsl:stylesheet>
```

Its evaluation results in a new XML document, having another structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <name username="JS1">John</name>
  <name username="MI1">Morka</name>
</root>
```

## Example 2 (transforming XML to XHTML)

Processing the following example XSLT file

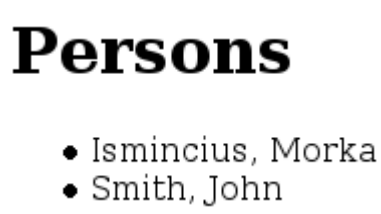
```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet
  version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns="http://www.w3.org/1999/xhtml">
```

```
  <xsl:output method="xml" indent="yes" encoding="UTF-8"/>

  <xsl:template match="/persons">
    <html>
      <head> <title>Testing XML Example</title> </head>
      <body>
        <h1>Persons</h1>
        <ul>
          <xsl:apply-templates select="person">
            <xsl:sort select="family-name" />
          </xsl:apply-templates>
        </ul>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="person">
    <li>
      <xsl:value-of select="family-name"/><xsl:text>, </xsl:text><xsl:value-of select="name"/>
    </li>
  </xsl:template>

</xsl:stylesheet>
```

with the XML input file shown above results in the following XHTML (whitespace has been adjusted here for clarity):

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head> <title>Testing XML Example</title> </head>
  <body>
    <h1>Persons</h1>
      <ul>
        <li>Ismincius, Morka</li>
        <li>Smith, John</li>
      </ul>
  </body>
</html>
```

This XHTML generates the output below when rendered in a web browser.




Rendered XHTML generated from an XML input file and an XSLT transformation.

In order for a web browser to be able to apply an XSL transformation to an XML document on display, an XML stylesheet processing instruction can be inserted into XML. So, for example, if the stylesheet in Example 2 above were available as "example2.xsl", the following instruction could be added to the original incoming XML:[22]

```
<?xml-stylesheet href="example2.xsl" type="text/xsl" ?>
```

In this example, `text/xsl` is technically incorrect according to the W3C specifications[22] (which say the type should be `application/xslt+xml`), but it is the only media type that is widely supported across browsers as of 2009, and the situation is unchanged in 2021.

# Processor implementations

- RaptorXML from Altova is an XSLT 3.0 processor available in the XMLSpy development toolkit and as a free-standing server implementation, invoked using a REST interface.
- IBM offers XSLT processing embedded in a special-purpose hardware appliance under the Datapower brand.
- libxslt is a free library released under the MIT License that can be reused in commercial applications. It is based on libxml and implemented in C for speed and portability. It supports XSLT 1.0 and EXSLT extensions.[23]
  - It can be used at the command line via xsltproc[24] which is included in macOS[25] and many Linux distributions, and can be used on Windows via Cygwin.[26]
  - The WebKit and Blink layout engines, used for example in the Safari and Chrome web browsers respectively, uses the libxslt library to do XSL transformations.[27]
  - Bindings exist for Python,[28] Perl,[29] Ruby,[30] PHP,[31] Common Lisp,[32] Tcl,[33] and C++.[34]
- Microsoft provides two XSLT processors (both XSLT 1.0 only). The earlier processor MSXML provides COM interfaces; from MSXML 4.0 it also includes the command line utility `msxsl.exe`.[35] The .NET runtime includes a separate built-in XSLT processor in its `System.Xml.Xsl` library.
- Saxon is an XSLT 3.0 and XQuery 3.1 processor with open-source and proprietary versions for stand-alone operation and for Java, JavaScript and .NET. A separate product Saxon-JS[36] offers XSLT 3.0 processing on Node.js and in the browser.
- Xalan is an open source XSLT 1.0 processor from the Apache Software Foundation available for Java and C++. A variant of the Xalan processor is included as the default XSLT processor in the standard Java distribution from Oracle.
- Web browsers: Safari, Chrome, Firefox, Opera and Internet Explorer all support XSLT 1.0 (only). Browsers can perform on-the-fly transformations of XML files and display the transformation output in the browser window. This is done either by embedding the XSL in the XML document or by referencing a file containing XSL instructions from the XML document. The latter may not work with Chrome on files from local filesystem because of its security model.[37]

## Performance

Most early XSLT processors were interpreters. More recently, code generation is increasingly common, using portable intermediate languages (such as Java bytecode or .NET Common Intermediate Language) as the target. However, even the interpretive products generally offer separate analysis and execution phases, allowing an optimized expression tree to be created in memory and reused to perform multiple transformations. This gives substantial performance benefits in online publishing applications, where the same transformation is applied many times per second to different source documents.[38] This separation is reflected in the design of XSLT processing APIs (such as JAXP).

Early XSLT processors had very few optimizations. Stylesheet documents were read into Document Object Models and the processor would act on them directly. XPath engines were also not optimized. Increasingly, however, XSLT processors use optimization techniques found in functional programming languages and database query languages, such as static rewriting of an expression tree (e.g., to move calculations out of loops), and lazy pipelined evaluation to reduce the memory footprint of intermediate results (and allow "early

exit" when the processor can evaluate an expression such as `following-sibling::*[1]` without a complete evaluation of all subexpressions). Many processors also use tree representations that are significantly more efficient (in both space and time)[39] than general-purpose DOM implementations.

In June 2014, Debbie Lockett and Michael Kay introduced an open-source benchmarking framework for XSLT processors called XT-Speedo.[40]

## See also

- XSLT elements – a list of some commonly used XSLT structures.
- Muenchian grouping – a dialect differential between XSLT1 and XSLT2+.
- eXtensible Stylesheet Language – a family of languages of which XSLT is a member
- XQuery and XSLT compared
- XSL formatting objects or *XSL-FO* – An XML-based language for documents, usually generated by transforming source documents with XSLT, consisting of objects used to create formatted output
- Identity transform – a starting point for filter chains that add or remove data elements from XML trees in a transformation pipeline
- Apache Cocoon – a Java-based framework for processing data with XSLT and other transformers.

## References


1. "Transformation" (http://www.w3.org/standards/xml/transformation). 2012-09-19.
2. "XML Output Method" (http://www.w3.org/TR/xslt#output). 2012-09-19.
3. "What is XSLT Used For?" (https://www.w3.org/standards/xml/transformation#uses). 2018-02-07.
4. "Introduction" (http://www.w3.org/TR/xslt#section-Introduction). *XSL Transformations (XSLT) Version 1.0 W3C Recommendation*. W3C. 16 November 1999. Retrieved November 7, 2012.
5. XSLT Version 2.0 Is Turing-Complete: A Purely Transformation Based Proof (https://www.researchgate.net/publication/221568016_XSLT_Version_20_Is_Turing-Complete_A_Purely_Transformation_Based_Proof)
6. Michael Kay. "What kind of language is XSLT?" (http://www.ibm.com/developerworks/library/x-xslt/). Retrieved July 8, 2016.
7. "A Proposal for XSL" (http://www.w3.org/TR/NOTE-XSL.html). W3C. Retrieved November 7, 2012.
8. "XML and Semantic Web W3C Standards Timeline" (http://www.dblab.ntua.gr/~bikakis/XML%20and%20Semantic%20Web%20W3C%20Standards%20Timeline-History.pdf) (PDF).
9. "XSL Transformations (XSLT)" (http://www.w3.org/TR/xslt). W3.org. 1999-11-16. Retrieved 2014-07-12.
10. "XSL Transformations (XSLT) Version 1.1" (http://www.w3.org/TR/xslt11/). W3.org. 2001-08-24. Retrieved 2014-07-12.
11. "XML Path Language (XPath) 2.0 (Second Edition)" (http://www.w3.org/TR/xpath20/). W3.org. 2010-12-14. Retrieved 2014-07-12.
12. "XSL Transformations (XSLT) Version 2.0" (http://www.w3.org/TR/xslt20/). W3.org. 2007-01-23. Retrieved 2014-07-12.
13. "XML and Semantic Web W3C Standards Timeline" (http://www.dblab.ntua.gr/~bikakis/XML%20and%20Semantic%20Web%20W3C%20Standards%20Timeline-History.pdf) (PDF). 2012-02-04.

14. "What's New in XSLT 3.0?" (http://www.w3.org/TR/xslt-30/#whats-new-in-xslt3). w3. Retrieved 6 January 2014.
15. Kay, Michael. "A Streaming XSLT Processor" (http://www.balisage.net/Proceedings/vol5/html/Kay01/BalisageVol5-Kay01.html). Balisage: The Markup Conference 2010 Proceedings. Retrieved 15 February 2012.
16. "Discover the Wonders of XSLT: XSLT Quirks" (http://www.developer.com/xml/article.php/3357231#Coding%20styles). "XSLT is a very specialized language with a distinct declarative flavor."
17. Kay, Michael. "What kind of language is XSLT?" (http://www.ibm.com/developerworks/library/x-xslt/). IBM. Retrieved 13 November 2013.
18. "XSL Transformations (XSLT) Version 2.0" (http://www.w3.org/TR/2007/REC-xslt20-20070123/#media-type-registration). W3C. Retrieved 19 October 2012.
19. "Application Media Types" (https://www.iana.org/assignments/media-types/application/index.html). IANA. Retrieved 19 October 2012.
20. "XSLT Requirements for Viewing XML in a Browser" (http://msdn.microsoft.com/en-us/library/windows/desktop/ms757857(v=vs.85).aspx). Microsoft. Retrieved 19 October 2012.
21. Kay, Michael (2008). *XSLT 2.0 and XPath 2.0 Programmer's Reference* (https://archive.org/details/xsltxpathprogram00kaym_646). Wiley. p. 100 (https://archive.org/details/xsltxpathprogram00kaym_646/page/n151). ISBN 978-0-470-19274-0.
22. "XSL Transformations (XSLT) Version 1.0: W3C Recommendation – Embedding Stylesheets" (http://www.w3.org/TR/xslt#section-Embedding-Stylesheets). W3C. 16 November 1999. Retrieved 20 September 2016.
23. "The XSLT C library for GNOME: libxslt" (http://xmlsoft.org/XSLT/index.html). Retrieved 23 November 2012.
24. "The XSLT C library for GNOME: The xsltproc tool" (http://xmlsoft.org/XSLT/xsltproc2.html). Retrieved 23 November 2012.
25. "xsltproc man page" (https://developer.apple.com/library/mac/#documentation/Darwin/Reference/ManPages/man1/xsltproc.1.html). Retrieved 23 November 2012.
26. "New package: libxslt" (http://www.cygwin.com/ml/cygwin-announce/2002/msg00018.html). Retrieved 23 November 2012.
27. "The WebKit Open Source Project - XSLT" (http://webkit.org/projects/xslt/index.html). Retrieved 2009-10-25.
28. "The XML C parser and toolkit of Gnome: Python and bindings" (http://xmlsoft.org/python.html). Retrieved 23 November 2012.
29. "XML::LibXSLT - Interface to the GNOME libxslt library" (https://metacpan.org/module/XML::LibXSLT). CPAN. Retrieved 23 November 2012.
30. "libxslt-ruby" (http://rubygems.org/gems/libxslt-ruby). Retrieved 23 November 2012.
31. "libxml" (http://www.php.net/manual/en/book.libxml.php). Retrieved 23 November 2012.
32. "cl-libxml2 High-level wrapper around libxml2 and libxslt libraries" (https://code.google.com/p/cl-libxml2/).
33. "TclXML" (http://tclxml.sourceforge.net/). Retrieved 21 May 2013.
34. "libxml++" (http://libxmlplusplus.sourceforge.net/). sourceforge.net. Retrieved 23 November 2012.
35. "Command Line Transformation Utility (msxsl.exe)" (http://www.microsoft.com/en-us/download/details.aspx?id=21714). Microsoft. Retrieved 22 October 2012.
36. "Saxon-JS" (http://www.saxonica.com/saxon-js/index.xml). Saxonica. Retrieved 6 September 2018.
37. "Issue 58151: Fails to load xml file on local file system using XMLHttpRequest" (https://bugs.chromium.org/p/chromium/issues/detail?id=58151).

38. Saxon: Anatomy of an XSLT processor (http://www-128.ibm.com/developerworks/xml/library/x-xslt2/) - Article describing implementation & optimization details of a popular XSLT processor.
39. Lumley, John; Kay, Michael (June 2015). "Improving Pattern Matching Performance in XSLT" (http://xmllondon.com/2015/presentations/lumley). *XML London 2015*: 9–25. doi:10.14337/XMLLondon15.Lumley01 (https://doi.org/10.14337%2FXMLLondon15.Lumley01). ISBN 978-0-9926471-2-4.
40. Kay, Michael; Lockett, Debbie (June 2014). "Benchmarking XSLT Performance" (http://xmllondon.com/2014/presentations/kay). *XML London 2014*: 10–23. doi:10.14337/XMLLondon14.Kay01 (https://doi.org/10.14337%2FXMLLondon14.Kay01). ISBN 978-0-9926471-1-7.

# Further reading

- *XSLT* by Doug Tidwell, published by O'Reilly (ISBN 0-596-00053-7)
- *XSLT Cookbook* by Sal Mangano, published by O'Reilly (ISBN 0-596-00974-7)
- *XSLT 2.0 Programmer's Reference* by Michael Kay (ISBN 0-764-56909-0)
- *XSLT 2.0 and XPath 2.0 Programmer's Reference* by Michael Kay (ISBN 978-0-470-19274-0)
- *XSLT 2.0 Web Development* by Dmitry Kirsanov (ISBN 0-13-140635-3)
- *XSL Companion, 2nd Edition* by Neil Bradley, published by Addison-Wesley (ISBN 0-201-77083-0)
- *XSLT and XPath on the Edge (Unlimited Edition)* by Jeni Tennison, published by Hungry Minds Inc, U.S. (ISBN 0-7645-4776-3)
- *XSLT & XPath, A Guide to XML Transformations* by John Robert Gardner and Zarella Rendon, published by Prentice-Hall (ISBN 0-13-040446-2)
- *XSL-FO* by Dave Pawson, published by O'Reilly (ISBN 978-0-596-00355-5)

# External links

### Documentation

- XSLT 1.0 W3C Recommendation (https://www.w3.org/TR/xslt-10/)
- XSLT 2.0 W3C Recommendation (http://www.w3.org/TR/xslt20/)
- XSLT 3.0 W3C Recommendation (http://www.w3.org/TR/xslt-30/)
- XSLT - MDC Docs (https://developer.mozilla.org/en/XSLT) by Mozilla Developer Network (https://developer.mozilla.org)
- XSLT Reference (MSDN) (http://msdn.microsoft.com/en-us/library/ms256069.aspx)
- XSLT Elements (Saxon) (http://saxon.sourceforge.net/saxon7.5/xsl-elements.html)
- XSLT introduction and reference (http://data2type.de/en/xml-xslt-xslfo/xslt/)

### XSLT code libraries

- EXSLT (http://www.exslt.org/) is a widespread community initiative to provide extensions to XSLT.
- FXSL (http://fxsl.sf.net/) is a library implementing support for Higher-order functions in XSLT. FXSL is written in XSLT itself.
- The XSLT Standard Library (http://xsltsl.sourceforge.net/) xsltsl, provides the XSLT developer with a set of XSLT templates for commonly used functions. These are implemented purely in XSLT, that is they do not use any extensions. xsltsl is a SourceForge project.

- Kernow (http://kernowforsaxon.sf.net/) A GUI for Saxon that provides a point and click interface for running transforms.
- xslt.js – Transform XML with XSLT (http://johannburkard.de/software/xsltjs/) JavaScript library that transforms XML with XSLT in the browser.

---