Product Description for SaxonJS

Version 2 released Jun 2020

This document lists the features provided by SaxonJS 2, for JavaScript in the browser and on Node.js.

This document does not form part of any contract unless expressly incorporated.

Language Support

1. XSLT (Transformation Processing)

1.1 XSLT 3.0 Basic Processor	Provides all mandatory features from the XSLT 3.0 specification (including try/catch, iterate, accumulators, maps, named modes, content value templates, and extended patterns, as well as features retained from XSLT 2.0).
1.2 XSLT 3.0 Serialization	Provides the serialization feature: specifically, the ability to convert the result trees produced as output of an XSLT transformation to lexical XML, or other formats including HTML, JSON, and plain text, under the control of serialization parameters defined in the stylesheet or via an external API.
1.3 XSLT 3.0 Compatibility	Provides XSLT 1.0 compatibility mode as defined in the XSLT 3.0 specification. If a stylesheet specifies `version="1.0"`, this causes certain constructs to behave in a way that retains the XSLT 1.0 behavior.
1.4 XSLT 3.0 Dynamic Evaluation	Provides use of the XSLT 3.0 instruction `xsl:evaluate`, which allows dynamic evaluation of XPath expressions.
1.5 XSLT 3.0 XPath 3.1 Feature	Provides full use of XPath 3.1 features, including XPath 3.1 functions, and maps and arrays. Note: Saxon supports XPath 3.1 unconditionally in XSLT 3.0 stylesheets, it does not offer a processing mode in which XPath is restricted to version 3.0.
1.6 XSLT 3.0 Higher-Order Functions	Provides higher-order functions: specifically, the ability to use functions as values, including dynamic function calls, inline functions, partial function application, and the standard higher-order functions defined in the XPath 3.1 function library.

Optional features not provided: XSLT 3.0 Schema Awareness, XSLT 3.0 Streaming.

For more details see: XSLT 3.0 and XPath 3.1 conformance.

Relevant W3C Specification: XSLT 3.0 Recommendation (08 June 2017).

2. XPath

2.1 XPath 3.1 Basic — Provides all XPath 3.1 features which do not require schema-awareness or higher-order functions. This includes an implementation of maps and arrays, and support for JSON, as well as language constructs retained from earlier XPath versions.

2.2 XPath 3.1 Higher-Order Functions — Provides higher-order functions: specifically, the ability to use functions as values, including dynamic function calls, inline functions, partial function application, and a library of built-in higher-order functions.

Optional features not provided: XPath 3.1 Schema Aware.

For more details see: XSLT 3.0 and XPath 3.1 conformance.

Relevant W3C Specification: XPath 3.1 Recommendation (21 March 2017).

Performance Features

3. Export stylesheet packages

XSLT 3.0 packaging allows stylesheet modules to be independently compiled and distributed, and provides much more "software engineering" control over public and private interfaces, and the like. The ability to save packages in compiled form ("stylesheet export file", SEF) gives much faster loading of frequently used stylesheets, and also enables in-browser execution using SaxonJS.

For more details see: Exporting stylesheets for SaxonJS.

4. Import stylesheet packages

Allows the importing of stylesheet packages in compiled form. Possible with all editions provided the package only uses features available in that edition. Note that this edition does not provide the ability to create compiled stylesheet packages.

For more details see: Exporting stylesheets for SaxonJS.

5. Optimizer (Basic)

The Basic optimizer provided with all Saxon editions provides a wide range of static and dynamic optimizations including full pipelining of list operations, lazy evaluation of variables, elimination of redundant sorting operations, etc.

Extensibility

6. Extensibility using reflexion (JavaScript)

Ability to access existing JavaScript objects and functions dynamically.

For more details see: Saxon interactive extensions.

7. Event handling extensions

Ability to write template rules which respond to user interaction events in the browser.

For more details see: Handling user input events.

8. Saxon extension functions (Basic)

Extension functions, as listed in the documentation, in the Saxon namespace. The Basic level excludes extension functions that depend on streaming or schema-awareness.

For more details see: Saxon extensions.

Localization

9. Formatting dates in English

Dates can be formatted as words in English.

10. Sorting and collations using JavaScript API

Support for sorting and comparison of strings using the Unicode Collation Algorithm uses the collation facilities available from the `Intl.Collator` JavaScript API.

Interfaces and APIs

11. Support for DOM

Ability to use a DOM (Document Object Model) for the input and output of transformations.

For more details see: Nodes.