Product Description for SaxonJ-PE (Professional Edition)

Version 11 released Feb 2022

This document lists the features provided by SaxonJ 11 Professional Edition (Saxon-PE), for Java.

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 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: XPath 3.1 conformance.

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

3. XQuery

3.1 XQuery 3.1 Minimal Conformance — Provides Minimal Conformance (including try/catch and "group-by", as well as language features retained from earlier XQuery versions) as defined in section 5 of the XQuery 3.1 specification.

3.2 XQuery 3.1 Modules — Provides the Module feature, which allows a query to be made up of multiple modules.

3.3 XQuery 3.1 Serialization — Provides the Serialization feature. This allows the output of a query to be serialized as lexical XML, or in other formats including HTML, JSON, and plain text, under the control of serialization parameters contained either in the query itself, or supplied externally.

3.4 XQuery 3.1 Higher-Order Functions — Provides the Higher-Order Function feature. This provides 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: XQuery 3.1 Schema Aware, XQuery 3.1 Typed Data, XQuery 3.1 Static Typing, XQuery Update 1.0.

For more details see: XQuery 3.1 conformance.

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

Performance Features

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: Compiling a stylesheet.

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.

6. Reading W3C schemas and DTDs

The W3C web server routinely rejects requests for commonly-referenced files such as the DTD for XHTML, causing parsing failures. In response to this, Saxon now includes copies of these documents within the issued JAR/DLL file, and recognizes requests for these documents, satisfying the request using the local copy.

Extensibility

7. EXSLT extension functions

A selection of EXSLT extension functions are provided (in the modules Common, Dates and Times, Math, Random, and Sets), as listed in the documentation.

For more details see: EXSLT extensions.

8. EXPath extension functions

A selection of EXPath extension functions are provided (in the modules Archive, Binary, and File), as listed in the documentation.

For more details see: EXPath extensions.

9. Extensibility using custom classes

Ability to write extension functions (for use in XSLT, XQuery, or XPath) by implementing a Saxon-defined interface and registering the implementation with the Saxon Configuration.

For more details see: Extensibility.

10. Extensibility using reflexion (Java)

Ability to access existing Java methods dynamically and invoke them as extension functions by means of dynamic loading and reflexion.

For more details see: Extensibility.

11. 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 extension functions.

12. SQL extension

XSLT extension functions and instructions providing access to SQL databases.

For more details see: Saxon SQL extension.

13. XSLT element extensibility

Ability to implement XSLT extension instructions by implementing a Saxon-defined interface and registering the implementation with the Saxon Configuration.

For more details see: Writing XSLT extension instructions.

Localization

14. Formatting numbers using ICU

Full support for formatting numbers as words in different languages is provided using the ICU4J library.

For more details see: Numbers and dates from ICU.

15. Formatting dates using ICU

Full support for formatting dates as words in different languages is provided using the ICU4J library.

For more details see: Numbers and dates from ICU.

16. Sorting and collations using ICU

Support for sorting and comparison of strings using the Unicode Collation Algorithm uses the collation facilities available from the ICU4J library.

For more details see: Unicode Collation Algorithm.

Interfaces and APIs

17. S9API API

SaxonJ's native interface for processing XSLT, XQuery, XPath, and XML Schema, on Java.

18. JAXP API

Implementations of the standard JAXP interfaces for XSLT transformation, XPath evaluation, and XML Schema validation.

For more details see: JAXP API conformance.

19. XQJ API

Implementations of the standard XQJ interfaces for XQuery processing. Note that the XQJ interfaces have been removed from the standard download of Saxon-HE because the Oracle specification license is not open source, but they are available on request.

For more details see: XQJ API conformance.

20. Support for DOM

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

For more details see: Object models.

21. Support for JDOM2, AXIOM, DOM4J, and XOM

Ability to use a JDOM2, AXIOM, DOM4J, and XOM for the input or output of transformations and queries. Note that the code for these interfaces is open source and can be compiled to work with Saxon-HE, but it does not come packaged with the Saxon-HE download.

For more details see: Object models.