| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SQLXML.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/sql/SQLWarning.html)   [**NEXT CLASS**](http://docs.google.com/java/sql/Statement.html) | [**FRAMES**](http://docs.google.com/index.html?java/sql/SQLXML.html)    [**NO FRAMES**](http://docs.google.com/SQLXML.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#2et92p0) |

## **java.sql**

Interface SQLXML

public interface **SQLXML**

The mapping in the JavaTM programming language for the SQL XML type. XML is a built-in type that stores an XML value as a column value in a row of a database table. By default drivers implement an SQLXML object as a logical pointer to the XML data rather than the data itself. An SQLXML object is valid for the duration of the transaction in which it was created.

The SQLXML interface provides methods for accessing the XML value as a String, a Reader or Writer, or as a Stream. The XML value may also be accessed through a Source or set as a Result, which are used with XML Parser APIs such as DOM, SAX, and StAX, as well as with XSLT transforms and XPath evaluations.

Methods in the interfaces ResultSet, CallableStatement, and PreparedStatement, such as getSQLXML allow a programmer to access an XML value. In addition, this interface has methods for updating an XML value.

The XML value of the SQLXML instance may be obtained as a BinaryStream using

SQLXML sqlxml = resultSet.getSQLXML(column);  
 InputStream binaryStream = sqlxml.getBinaryStream();

For example, to parse an XML value with a DOM parser:

DocumentBuilder parser = DocumentBuilderFactory.newInstance().newDocumentBuilder();  
 Document result = parser.parse(binaryStream);

or to parse an XML value with a SAX parser to your handler:

SAXParser parser = SAXParserFactory.newInstance().newSAXParser();  
 parser.parse(binaryStream, myHandler);

or to parse an XML value with a StAX parser:

XMLInputFactory factory = XMLInputFactory.newInstance();  
 XMLStreamReader streamReader = factory.createXMLStreamReader(binaryStream);

Because databases may use an optimized representation for the XML, accessing the value through getSource() and setResult() can lead to improved processing performance without serializing to a stream representation and parsing the XML.

For example, to obtain a DOM Document Node:

DOMSource domSource = sqlxml.getSource(DOMSource.class);  
 Document document = (Document) domSource.getNode();

or to set the value to a DOM Document Node to myNode:

DOMResult domResult = sqlxml.setResult(DOMResult.class);  
 domResult.setNode(myNode);

or, to send SAX events to your handler:

SAXSource saxSource = sqlxml.getSource(SAXSource.class);  
 XMLReader xmlReader = saxSource.getXMLReader();  
 xmlReader.setContentHandler(myHandler);  
 xmlReader.parse(saxSource.getInputSource());

or, to set the result value from SAX events:

SAXResult saxResult = sqlxml.setResult(SAXResult.class);  
 ContentHandler contentHandler = saxResult.getXMLReader().getContentHandler();  
 contentHandler.startDocument();  
 // set the XML elements and attributes into the result  
 contentHandler.endDocument();

or, to obtain StAX events:

StAXSource staxSource = sqlxml.getSource(StAXSource.class);  
 XMLStreamReader streamReader = staxSource.getXMLStreamReader();

or, to set the result value from StAX events:

StAXResult staxResult = sqlxml.setResult(StAXResult.class);  
 XMLStreamWriter streamWriter = staxResult.getXMLStreamWriter();

or, to perform XSLT transformations on the XML value using the XSLT in xsltFile output to file resultFile:

File xsltFile = new File("a.xslt");  
 File myFile = new File("result.xml");  
 Transformer xslt = TransformerFactory.newInstance().newTransformer(new StreamSource(xsltFile));  
 Source source = sqlxml.getSource(null);  
 Result result = new StreamResult(myFile);  
 xslt.transform(source, result);

or, to evaluate an XPath expression on the XML value:

XPath xpath = XPathFactory.newInstance().newXPath();  
 DOMSource domSource = sqlxml.getSource(DOMSource.class);  
 Document document = (Document) domSource.getNode();  
 String expression = "/foo/@bar";  
 String barValue = xpath.evaluate(expression, document);

To set the XML value to be the result of an XSLT transform:

File sourceFile = new File("source.xml");  
 Transformer xslt = TransformerFactory.newInstance().newTransformer(new StreamSource(xsltFile));  
 Source streamSource = new StreamSource(sourceFile);  
 Result result = sqlxml.setResult(null);  
 xslt.transform(streamSource, result);

Any Source can be transformed to a Result using the identity transform specified by calling newTransformer():

Transformer identity = TransformerFactory.newInstance().newTransformer();  
 Source source = sqlxml.getSource(null);  
 File myFile = new File("result.xml");  
 Result result = new StreamResult(myFile);  
 identity.transform(source, result);

To write the contents of a Source to standard output:

Transformer identity = TransformerFactory.newInstance().newTransformer();  
 Source source = sqlxml.getSource(null);  
 Result result = new StreamResult(System.out);  
 identity.transform(source, result);

To create a DOMSource from a DOMResult:

DOMSource domSource = new DOMSource(domResult.getNode());

Incomplete or invalid XML values may cause an SQLException when set or the exception may occur when execute() occurs. All streams must be closed before execute() occurs or an SQLException will be thrown.

Reading and writing XML values to or from an SQLXML object can happen at most once. The conceptual states of readable and not readable determine if one of the reading APIs will return a value or throw an exception. The conceptual states of writable and not writable determine if one of the writing APIs will set a value or throw an exception.

The state moves from readable to not readable once free() or any of the reading APIs are called: getBinaryStream(), getCharacterStream(), getSource(), and getString(). Implementations may also change the state to not writable when this occurs.

The state moves from writable to not writeable once free() or any of the writing APIs are called: setBinaryStream(), setCharacterStream(), setResult(), and setString(). Implementations may also change the state to not readable when this occurs.

All methods on the SQLXML interface must be fully implemented if the JDBC driver supports the data type.

**Since:** 1.6 **See Also:**[javax.xml.parsers](http://docs.google.com/javax/xml/parsers/package-summary.html), [javax.xml.stream](http://docs.google.com/javax/xml/stream/package-summary.html), [javax.xml.transform](http://docs.google.com/javax/xml/transform/package-summary.html), [javax.xml.xpath](http://docs.google.com/javax/xml/xpath/package-summary.html)

| **Method Summary** | |
| --- | --- |
| void | [**free**](http://docs.google.com/java/sql/SQLXML.html#free())()            This method closes this object and releases the resources that it held. |
| [InputStream](http://docs.google.com/java/io/InputStream.html) | [**getBinaryStream**](http://docs.google.com/java/sql/SQLXML.html#getBinaryStream())()            Retrieves the XML value designated by this SQLXML instance as a stream. |
| [Reader](http://docs.google.com/java/io/Reader.html) | [**getCharacterStream**](http://docs.google.com/java/sql/SQLXML.html#getCharacterStream())()            Retrieves the XML value designated by this SQLXML instance as a java.io.Reader object. |
| | <T extends [Source](http://docs.google.com/javax/xml/transform/Source.html)>  T | | --- | | [**getSource**](http://docs.google.com/java/sql/SQLXML.html#getSource(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<T> sourceClass)            Returns a Source for reading the XML value designated by this SQLXML instance. |
| [String](http://docs.google.com/java/lang/String.html) | [**getString**](http://docs.google.com/java/sql/SQLXML.html#getString())()            Returns a string representation of the XML value designated by this SQLXML instance. |
| [OutputStream](http://docs.google.com/java/io/OutputStream.html) | [**setBinaryStream**](http://docs.google.com/java/sql/SQLXML.html#setBinaryStream())()            Retrieves a stream that can be used to write the XML value that this SQLXML instance represents. |
| [Writer](http://docs.google.com/java/io/Writer.html) | [**setCharacterStream**](http://docs.google.com/java/sql/SQLXML.html#setCharacterStream())()            Retrieves a stream to be used to write the XML value that this SQLXML instance represents. |
| | <T extends [Result](http://docs.google.com/javax/xml/transform/Result.html)>  T | | --- | | [**setResult**](http://docs.google.com/java/sql/SQLXML.html#setResult(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<T> resultClass)            Returns a Result for setting the XML value designated by this SQLXML instance. |
| void | [**setString**](http://docs.google.com/java/sql/SQLXML.html#setString(java.lang.String))([String](http://docs.google.com/java/lang/String.html) value)            Sets the XML value designated by this SQLXML instance to the given String representation. |

| **Method Detail** |
| --- |

### free

void **free**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

This method closes this object and releases the resources that it held. The SQL XML object becomes invalid and neither readable or writeable when this method is called. After free has been called, any attempt to invoke a method other than free will result in a SQLException being thrown. If free is called multiple times, the subsequent calls to free are treated as a no-op.

**Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error freeing the XML value. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getBinaryStream

[InputStream](http://docs.google.com/java/io/InputStream.html) **getBinaryStream**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the XML value designated by this SQLXML instance as a stream. The bytes of the input stream are interpreted according to appendix F of the XML 1.0 specification. The behavior of this method is the same as ResultSet.getBinaryStream() when the designated column of the ResultSet has a type java.sql.Types of SQLXML.

The SQL XML object becomes not readable when this method is called and may also become not writable depending on implementation.

**Returns:**a stream containing the XML data. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value. An exception is thrown if the state is not readable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setBinaryStream

[OutputStream](http://docs.google.com/java/io/OutputStream.html) **setBinaryStream**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a stream that can be used to write the XML value that this SQLXML instance represents. The stream begins at position 0. The bytes of the stream are interpreted according to appendix F of the XML 1.0 specification The behavior of this method is the same as ResultSet.updateBinaryStream() when the designated column of the ResultSet has a type java.sql.Types of SQLXML.

The SQL XML object becomes not writeable when this method is called and may also become not readable depending on implementation.

**Returns:**a stream to which data can be written. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value. An exception is thrown if the state is not writable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getCharacterStream

[Reader](http://docs.google.com/java/io/Reader.html) **getCharacterStream**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the XML value designated by this SQLXML instance as a java.io.Reader object. The format of this stream is defined by org.xml.sax.InputSource, where the characters in the stream represent the unicode code points for XML according to section 2 and appendix B of the XML 1.0 specification. Although an encoding declaration other than unicode may be present, the encoding of the stream is unicode. The behavior of this method is the same as ResultSet.getCharacterStream() when the designated column of the ResultSet has a type java.sql.Types of SQLXML.

The SQL XML object becomes not readable when this method is called and may also become not writable depending on implementation.

**Returns:**a stream containing the XML data. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value. The getCause() method of the exception may provide a more detailed exception, for example, if the stream does not contain valid characters. An exception is thrown if the state is not readable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setCharacterStream

[Writer](http://docs.google.com/java/io/Writer.html) **setCharacterStream**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a stream to be used to write the XML value that this SQLXML instance represents. The format of this stream is defined by org.xml.sax.InputSource, where the characters in the stream represent the unicode code points for XML according to section 2 and appendix B of the XML 1.0 specification. Although an encoding declaration other than unicode may be present, the encoding of the stream is unicode. The behavior of this method is the same as ResultSet.updateCharacterStream() when the designated column of the ResultSet has a type java.sql.Types of SQLXML.

The SQL XML object becomes not writeable when this method is called and may also become not readable depending on implementation.

**Returns:**a stream to which data can be written. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value. The getCause() method of the exception may provide a more detailed exception, for example, if the stream does not contain valid characters. An exception is thrown if the state is not writable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getString

[String](http://docs.google.com/java/lang/String.html) **getString**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns a string representation of the XML value designated by this SQLXML instance. The format of this String is defined by org.xml.sax.InputSource, where the characters in the stream represent the unicode code points for XML according to section 2 and appendix B of the XML 1.0 specification. Although an encoding declaration other than unicode may be present, the encoding of the String is unicode. The behavior of this method is the same as ResultSet.getString() when the designated column of the ResultSet has a type java.sql.Types of SQLXML.

The SQL XML object becomes not readable when this method is called and may also become not writable depending on implementation.

**Returns:**a string representation of the XML value designated by this SQLXML instance. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value. The getCause() method of the exception may provide a more detailed exception, for example, if the stream does not contain valid characters. An exception is thrown if the state is not readable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setString

void **setString**([String](http://docs.google.com/java/lang/String.html) value)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the XML value designated by this SQLXML instance to the given String representation. The format of this String is defined by org.xml.sax.InputSource, where the characters in the stream represent the unicode code points for XML according to section 2 and appendix B of the XML 1.0 specification. Although an encoding declaration other than unicode may be present, the encoding of the String is unicode. The behavior of this method is the same as ResultSet.updateString() when the designated column of the ResultSet has a type java.sql.Types of SQLXML.

The SQL XML object becomes not writeable when this method is called and may also become not readable depending on implementation.

**Parameters:**value - the XML value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value. The getCause() method of the exception may provide a more detailed exception, for example, if the stream does not contain valid characters. An exception is thrown if the state is not writable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getSource

<T extends [Source](http://docs.google.com/javax/xml/transform/Source.html)> T **getSource**([Class](http://docs.google.com/java/lang/Class.html)<T> sourceClass)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns a Source for reading the XML value designated by this SQLXML instance. Sources are used as inputs to XML parsers and XSLT transformers.

Sources for XML parsers will have namespace processing on by default. The systemID of the Source is implementation dependent.

The SQL XML object becomes not readable when this method is called and may also become not writable depending on implementation.

Note that SAX is a callback architecture, so a returned SAXSource should then be set with a content handler that will receive the SAX events from parsing. The content handler will receive callbacks based on the contents of the XML.

SAXSource saxSource = sqlxml.getSource(SAXSource.class);  
 XMLReader xmlReader = saxSource.getXMLReader();  
 xmlReader.setContentHandler(myHandler);  
 xmlReader.parse(saxSource.getInputSource());

**Parameters:**sourceClass - The class of the source, or null. If the class is null, a vendor specifc Source implementation will be returned. The following classes are supported at a minimum:

javax.xml.transform.dom.DOMSource - returns a DOMSource  
 javax.xml.transform.sax.SAXSource - returns a SAXSource  
 javax.xml.transform.stax.StAXSource - returns a StAXSource  
 javax.xml.transform.stream.StreamSource - returns a StreamSource

**Returns:**a Source for reading the XML value. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value or if this feature is not supported. The getCause() method of the exception may provide a more detailed exception, for example, if an XML parser exception occurs. An exception is thrown if the state is not readable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setResult

<T extends [Result](http://docs.google.com/javax/xml/transform/Result.html)> T **setResult**([Class](http://docs.google.com/java/lang/Class.html)<T> resultClass)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns a Result for setting the XML value designated by this SQLXML instance.

The systemID of the Result is implementation dependent.

The SQL XML object becomes not writeable when this method is called and may also become not readable depending on implementation.

Note that SAX is a callback architecture and the returned SAXResult has a content handler assigned that will receive the SAX events based on the contents of the XML. Call the content handler with the contents of the XML document to assign the values.

SAXResult saxResult = sqlxml.setResult(SAXResult.class);  
 ContentHandler contentHandler = saxResult.getXMLReader().getContentHandler();  
 contentHandler.startDocument();  
 // set the XML elements and attributes into the result  
 contentHandler.endDocument();

**Parameters:**resultClass - The class of the result, or null. If resultClass is null, a vendor specific Result implementation will be returned. The following classes are supported at a minimum:

javax.xml.transform.dom.DOMResult - returns a DOMResult  
 javax.xml.transform.sax.SAXResult - returns a SAXResult  
 javax.xml.transform.stax.StAXResult - returns a StAXResult  
 javax.xml.transform.stream.StreamResult - returns a StreamResult

**Returns:**Returns a Result for setting the XML value. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if there is an error processing the XML value or if this feature is not supported. The getCause() method of the exception may provide a more detailed exception, for example, if an XML parser exception occurs. An exception is thrown if the state is not writable. [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SQLXML.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/sql/SQLWarning.html)   [**NEXT CLASS**](http://docs.google.com/java/sql/Statement.html) | [**FRAMES**](http://docs.google.com/index.html?java/sql/SQLXML.html)    [**NO FRAMES**](http://docs.google.com/SQLXML.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#2et92p0) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).