| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/StreamReaderDelegate.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/javax/xml/stream/util/EventReaderDelegate.html)   [**NEXT CLASS**](http://docs.google.com/javax/xml/stream/util/XMLEventAllocator.html) | [**FRAMES**](http://docs.google.com/index.html?javax/xml/stream/util/StreamReaderDelegate.html)    [**NO FRAMES**](http://docs.google.com/StreamReaderDelegate.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#4d34og8) | [METHOD](#3rdcrjn) |

## **javax.xml.stream.util**

Class StreamReaderDelegate

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.xml.stream.util.StreamReaderDelegate**

**All Implemented Interfaces:** [XMLStreamConstants](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html), [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

public class **StreamReaderDelegate**extends [Object](http://docs.google.com/java/lang/Object.html)implements [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

This is the base class for deriving an XMLStreamReader filter This class is designed to sit between an XMLStreamReader and an application's XMLStreamReader. By default each method does nothing but call the corresponding method on the parent interface.

**Since:** 1.6 **See Also:**[XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html), [EventReaderDelegate](http://docs.google.com/javax/xml/stream/util/EventReaderDelegate.html)

| **Field Summary** | |
| --- | --- |

| **Fields inherited from interface javax.xml.stream.**[**XMLStreamConstants**](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html) |
| --- |
| [ATTRIBUTE](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#ATTRIBUTE), [CDATA](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#CDATA), [CHARACTERS](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#CHARACTERS), [COMMENT](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#COMMENT), [DTD](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#DTD), [END\_DOCUMENT](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#END_DOCUMENT), [END\_ELEMENT](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#END_ELEMENT), [ENTITY\_DECLARATION](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#ENTITY_DECLARATION), [ENTITY\_REFERENCE](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#ENTITY_REFERENCE), [NAMESPACE](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#NAMESPACE), [NOTATION\_DECLARATION](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#NOTATION_DECLARATION), [PROCESSING\_INSTRUCTION](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#PROCESSING_INSTRUCTION), [SPACE](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#SPACE), [START\_DOCUMENT](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#START_DOCUMENT), [START\_ELEMENT](http://docs.google.com/javax/xml/stream/XMLStreamConstants.html#START_ELEMENT) |

| **Constructor Summary** | |
| --- | --- |
| [**StreamReaderDelegate**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#StreamReaderDelegate())()            Construct an empty filter with no parent. |
| [**StreamReaderDelegate**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#StreamReaderDelegate(javax.xml.stream.XMLStreamReader))([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader)            Construct an filter with the specified parent. |

| **Method Summary** | |
| --- | --- |
| void | [**close**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#close())()            Frees any resources associated with this Reader. |
| int | [**getAttributeCount**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributeCount())()            Returns the count of attributes on this START\_ELEMENT, this method is only valid on a START\_ELEMENT or ATTRIBUTE. |
| [String](http://docs.google.com/java/lang/String.html) | [**getAttributeLocalName**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributeLocalName(int))(int index)            Returns the localName of the attribute at the provided index |
| [QName](http://docs.google.com/javax/xml/namespace/QName.html) | [**getAttributeName**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributeName(int))(int index)            Returns the qname of the attribute at the provided index |
| [String](http://docs.google.com/java/lang/String.html) | [**getAttributeNamespace**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributeNamespace(int))(int index)            Returns the namespace of the attribute at the provided index |
| [String](http://docs.google.com/java/lang/String.html) | [**getAttributePrefix**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributePrefix(int))(int index)            Returns the prefix of this attribute at the provided index |
| [String](http://docs.google.com/java/lang/String.html) | [**getAttributeType**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributeType(int))(int index)            Returns the XML type of the attribute at the provided index |
| [String](http://docs.google.com/java/lang/String.html) | [**getAttributeValue**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributeValue(int))(int index)            Returns the value of the attribute at the index |
| [String](http://docs.google.com/java/lang/String.html) | [**getAttributeValue**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getAttributeValue(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) namespaceUri, [String](http://docs.google.com/java/lang/String.html) localName)            Returns the normalized attribute value of the attribute with the namespace and localName If the namespaceURI is null the namespace is not checked for equality |
| [String](http://docs.google.com/java/lang/String.html) | [**getCharacterEncodingScheme**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getCharacterEncodingScheme())()            Returns the character encoding declared on the xml declaration Returns null if none was declared |
| [String](http://docs.google.com/java/lang/String.html) | [**getElementText**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getElementText())()            Reads the content of a text-only element, an exception is thrown if this is not a text-only element. |
| [String](http://docs.google.com/java/lang/String.html) | [**getEncoding**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getEncoding())()            Return input encoding if known or null if unknown. |
| int | [**getEventType**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getEventType())()            Returns an integer code that indicates the type of the event the cursor is pointing to. |
| [String](http://docs.google.com/java/lang/String.html) | [**getLocalName**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getLocalName())()            Returns the (local) name of the current event. |
| [Location](http://docs.google.com/javax/xml/stream/Location.html) | [**getLocation**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getLocation())()            Return the current location of the processor. |
| [QName](http://docs.google.com/javax/xml/namespace/QName.html) | [**getName**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getName())()            Returns a QName for the current START\_ELEMENT or END\_ELEMENT event |
| [NamespaceContext](http://docs.google.com/javax/xml/namespace/NamespaceContext.html) | [**getNamespaceContext**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getNamespaceContext())()            Returns a read only namespace context for the current position. |
| int | [**getNamespaceCount**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getNamespaceCount())()            Returns the count of namespaces declared on this START\_ELEMENT or END\_ELEMENT, this method is only valid on a START\_ELEMENT, END\_ELEMENT or NAMESPACE. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNamespacePrefix**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getNamespacePrefix(int))(int index)            Returns the prefix for the namespace declared at the index. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNamespaceURI**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getNamespaceURI())()            If the current event is a START\_ELEMENT or END\_ELEMENT this method returns the URI of the prefix or the default namespace. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNamespaceURI**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getNamespaceURI(int))(int index)            Returns the uri for the namespace declared at the index. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNamespaceURI**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getNamespaceURI(java.lang.String))([String](http://docs.google.com/java/lang/String.html) prefix)            Return the uri for the given prefix. |
| [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) | [**getParent**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getParent())()            Get the parent of this instance. |
| [String](http://docs.google.com/java/lang/String.html) | [**getPIData**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getPIData())()            Get the data section of a processing instruction |
| [String](http://docs.google.com/java/lang/String.html) | [**getPITarget**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getPITarget())()            Get the target of a processing instruction |
| [String](http://docs.google.com/java/lang/String.html) | [**getPrefix**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getPrefix())()            Returns the prefix of the current event or null if the event does not have a prefix |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getProperty**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getProperty(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Get the value of a feature/property from the underlying implementation |
| [String](http://docs.google.com/java/lang/String.html) | [**getText**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getText())()            Returns the current value of the parse event as a string, this returns the string value of a CHARACTERS event, returns the value of a COMMENT, the replacement value for an ENTITY\_REFERENCE, the string value of a CDATA section, the string value for a SPACE event, or the String value of the internal subset of the DTD. |
| char[] | [**getTextCharacters**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getTextCharacters())()            Returns an array which contains the characters from this event. |
| int | [**getTextCharacters**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getTextCharacters(int,%20char%5B%5D,%20int,%20int))(int sourceStart, char[] target, int targetStart, int length)            Gets the the text associated with a CHARACTERS, SPACE or CDATA event. |
| int | [**getTextLength**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getTextLength())()            Returns the length of the sequence of characters for this Text event within the text character array. |
| int | [**getTextStart**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getTextStart())()            Returns the offset into the text character array where the first character (of this text event) is stored. |
| [String](http://docs.google.com/java/lang/String.html) | [**getVersion**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#getVersion())()            Get the xml version declared on the xml declaration Returns null if none was declared |
| boolean | [**hasName**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#hasName())()            returns true if the current event has a name (is a START\_ELEMENT or END\_ELEMENT) returns false otherwise |
| boolean | [**hasNext**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#hasNext())()            Returns true if there are more parsing events and false if there are no more events. |
| boolean | [**hasText**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#hasText())()            Return true if the current event has text, false otherwise The following events have text: CHARACTERS,DTD ,ENTITY\_REFERENCE, COMMENT, SPACE |
| boolean | [**isAttributeSpecified**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#isAttributeSpecified(int))(int index)            Returns a boolean which indicates if this attribute was created by default |
| boolean | [**isCharacters**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#isCharacters())()            Returns true if the cursor points to a character data event |
| boolean | [**isEndElement**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#isEndElement())()            Returns true if the cursor points to an end tag (otherwise false) |
| boolean | [**isStandalone**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#isStandalone())()            Get the standalone declaration from the xml declaration |
| boolean | [**isStartElement**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#isStartElement())()            Returns true if the cursor points to a start tag (otherwise false) |
| boolean | [**isWhiteSpace**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#isWhiteSpace())()            Returns true if the cursor points to a character data event that consists of all whitespace |
| int | [**next**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#next())()            Get next parsing event - a processor may return all contiguous character data in a single chunk, or it may split it into several chunks. |
| int | [**nextTag**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#nextTag())()            Skips any white space (isWhiteSpace() returns true), COMMENT, or PROCESSING\_INSTRUCTION, until a START\_ELEMENT or END\_ELEMENT is reached. |
| void | [**require**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#require(int,%20java.lang.String,%20java.lang.String))(int type, [String](http://docs.google.com/java/lang/String.html) namespaceURI, [String](http://docs.google.com/java/lang/String.html) localName)            Test if the current event is of the given type and if the namespace and name match the current namespace and name of the current event. |
| void | [**setParent**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#setParent(javax.xml.stream.XMLStreamReader))([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader)            Set the parent of this instance. |
| boolean | [**standaloneSet**](http://docs.google.com/javax/xml/stream/util/StreamReaderDelegate.html#standaloneSet())()            Checks if standalone was set in the document |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Constructor Detail** |
| --- |

### StreamReaderDelegate

public **StreamReaderDelegate**()

Construct an empty filter with no parent.

### StreamReaderDelegate

public **StreamReaderDelegate**([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader)

Construct an filter with the specified parent.

**Parameters:**reader - the parent

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

### setParent

public void **setParent**([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader)

Set the parent of this instance.

**Parameters:**reader - the new parent

### getParent

public [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **getParent**()

Get the parent of this instance.

**Returns:**the parent or null if none is set

### next

public int **next**()  
 throws [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#next()) Get next parsing event - a processor may return all contiguous character data in a single chunk, or it may split it into several chunks. If the property javax.xml.stream.isCoalescing is set to true element content must be coalesced and only one CHARACTERS event must be returned for contiguous element content or CDATA Sections. By default entity references must be expanded and reported transparently to the application. An exception will be thrown if an entity reference cannot be expanded. If element content is empty (i.e. content is "") then no CHARACTERS event will be reported.

Given the following XML:

<foo><!--description-->content text<![CDATA[<greeting>Hello</greeting>]]>other content</foo>

The behavior of calling next() when being on foo will be:

1- the comment (COMMENT)

2- then the characters section (CHARACTERS)

3- then the CDATA section (another CHARACTERS)

4- then the next characters section (another CHARACTERS)

5- then the END\_ELEMENT

**NOTE:** empty element (such as <tag/>) will be reported with two separate events: START\_ELEMENT, END\_ELEMENT - This preserves parsing equivalency of empty element to <tag></tag>. This method will throw an IllegalStateException if it is called after hasNext() returns false.

**Specified by:**[next](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#next()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the integer code corresponding to the current parse event **Throws:** [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html) - if there is an error processing the underlying XML source**See Also:**[XMLEvent](http://docs.google.com/javax/xml/stream/events/XMLEvent.html)

### nextTag

public int **nextTag**()  
 throws [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#nextTag()) Skips any white space (isWhiteSpace() returns true), COMMENT, or PROCESSING\_INSTRUCTION, until a START\_ELEMENT or END\_ELEMENT is reached. If other than white space characters, COMMENT, PROCESSING\_INSTRUCTION, START\_ELEMENT, END\_ELEMENT are encountered, an exception is thrown. This method should be used when processing element-only content seperated by white space.

Precondition: none

Postcondition: the current event is START\_ELEMENT or END\_ELEMENT and cursor may have moved over any whitespace event.

Essentially it does the following (implementations are free to optimized but must do equivalent processing):

int eventType = next();  
 while((eventType == XMLStreamConstants.CHARACTERS && isWhiteSpace()) // skip whitespace  
 || (eventType == XMLStreamConstants.CDATA && isWhiteSpace())   
 // skip whitespace  
 || eventType == XMLStreamConstants.SPACE  
 || eventType == XMLStreamConstants.PROCESSING\_INSTRUCTION  
 || eventType == XMLStreamConstants.COMMENT  
 ) {  
 eventType = next();  
 }  
 if (eventType != XMLStreamConstants.START\_ELEMENT && eventType != XMLStreamConstants.END\_ELEMENT) {  
 throw new String XMLStreamException("expected start or end tag", getLocation());  
 }  
 return eventType;

**Specified by:**[nextTag](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#nextTag()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the event type of the element read (START\_ELEMENT or END\_ELEMENT) **Throws:** [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html) - if the current event is not white space, PROCESSING\_INSTRUCTION, START\_ELEMENT or END\_ELEMENT

### getElementText

public [String](http://docs.google.com/java/lang/String.html) **getElementText**()  
 throws [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getElementText()) Reads the content of a text-only element, an exception is thrown if this is not a text-only element. Regardless of value of javax.xml.stream.isCoalescing this method always returns coalesced content.

Precondition: the current event is START\_ELEMENT.

Postcondition: the current event is the corresponding END\_ELEMENT.

The method does the following (implementations are free to optimized but must do equivalent processing):

if(getEventType() != XMLStreamConstants.START\_ELEMENT) {  
 throw new XMLStreamException(  
 "parser must be on START\_ELEMENT to read next text", getLocation());  
 }  
 int eventType = next();  
 StringBuffer content = new StringBuffer();  
 while(eventType != XMLStreamConstants.END\_ELEMENT ) {  
 if(eventType == XMLStreamConstants.CHARACTERS  
 || eventType == XMLStreamConstants.CDATA  
 || eventType == XMLStreamConstants.SPACE  
 || eventType == XMLStreamConstants.ENTITY\_REFERENCE) {  
 buf.append(getText());  
 } else if(eventType == XMLStreamConstants.PROCESSING\_INSTRUCTION  
 || eventType == XMLStreamConstants.COMMENT) {  
 // skipping  
 } else if(eventType == XMLStreamConstants.END\_DOCUMENT) {  
 throw new XMLStreamException(  
 "unexpected end of document when reading element text content", this);  
 } else if(eventType == XMLStreamConstants.START\_ELEMENT) {  
 throw new XMLStreamException(  
 "element text content may not contain START\_ELEMENT", getLocation());  
 } else {  
 throw new XMLStreamException(  
 "Unexpected event type "+eventType, getLocation());  
 }  
 eventType = next();  
 }  
 return buf.toString();

**Specified by:**[getElementText](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getElementText()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Throws:** [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html) - if the current event is not a START\_ELEMENT or if a non text element is encountered

### require

public void **require**(int type,  
 [String](http://docs.google.com/java/lang/String.html) namespaceURI,  
 [String](http://docs.google.com/java/lang/String.html) localName)  
 throws [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#require(int,%20java.lang.String,%20java.lang.String)) Test if the current event is of the given type and if the namespace and name match the current namespace and name of the current event. If the namespaceURI is null it is not checked for equality, if the localName is null it is not checked for equality.

**Specified by:**[require](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#require(int,%20java.lang.String,%20java.lang.String)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**type - the event typenamespaceURI - the uri of the event, may be nulllocalName - the localName of the event, may be null **Throws:** [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html) - if the required values are not matched.

### hasNext

public boolean **hasNext**()  
 throws [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#hasNext()) Returns true if there are more parsing events and false if there are no more events. This method will return false if the current state of the XMLStreamReader is END\_DOCUMENT

**Specified by:**[hasNext](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#hasNext()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**true if there are more events, false otherwise **Throws:** [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html) - if there is a fatal error detecting the next state

### close

public void **close**()  
 throws [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#close()) Frees any resources associated with this Reader. This method does not close the underlying input source.

**Specified by:**[close](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#close()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Throws:** [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html) - if there are errors freeing associated resources

### getNamespaceURI

public [String](http://docs.google.com/java/lang/String.html) **getNamespaceURI**([String](http://docs.google.com/java/lang/String.html) prefix)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceURI(java.lang.String)) Return the uri for the given prefix. The uri returned depends on the current state of the processor.

**NOTE:**The 'xml' prefix is bound as defined in [Namespaces in XML](http://www.w3.org/TR/REC-xml-names/#ns-using) specification to "http://www.w3.org/XML/1998/namespace".

**NOTE:** The 'xmlns' prefix must be resolved to following namespace <http://www.w3.org/2000/xmlns/>

**Specified by:**[getNamespaceURI](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceURI(java.lang.String)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**prefix - The prefix to lookup, may not be null **Returns:**the uri bound to the given prefix or null if it is not bound

### getNamespaceContext

public [NamespaceContext](http://docs.google.com/javax/xml/namespace/NamespaceContext.html) **getNamespaceContext**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceContext()) Returns a read only namespace context for the current position. The context is transient and only valid until a call to next() changes the state of the reader.

**Specified by:**[getNamespaceContext](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceContext()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**return a namespace context

### isStartElement

public boolean **isStartElement**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isStartElement()) Returns true if the cursor points to a start tag (otherwise false)

**Specified by:**[isStartElement](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isStartElement()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**true if the cursor points to a start tag, false otherwise

### isEndElement

public boolean **isEndElement**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isEndElement()) Returns true if the cursor points to an end tag (otherwise false)

**Specified by:**[isEndElement](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isEndElement()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**true if the cursor points to an end tag, false otherwise

### isCharacters

public boolean **isCharacters**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isCharacters()) Returns true if the cursor points to a character data event

**Specified by:**[isCharacters](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isCharacters()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**true if the cursor points to character data, false otherwise

### isWhiteSpace

public boolean **isWhiteSpace**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isWhiteSpace()) Returns true if the cursor points to a character data event that consists of all whitespace

**Specified by:**[isWhiteSpace](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isWhiteSpace()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**true if the cursor points to all whitespace, false otherwise

### getAttributeValue

public [String](http://docs.google.com/java/lang/String.html) **getAttributeValue**([String](http://docs.google.com/java/lang/String.html) namespaceUri,  
 [String](http://docs.google.com/java/lang/String.html) localName)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeValue(java.lang.String,%20java.lang.String)) Returns the normalized attribute value of the attribute with the namespace and localName If the namespaceURI is null the namespace is not checked for equality

**Specified by:**[getAttributeValue](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeValue(java.lang.String,%20java.lang.String)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**namespaceUri - the namespace of the attributelocalName - the local name of the attribute, cannot be null **Returns:**returns the value of the attribute , returns null if not found

### getAttributeCount

public int **getAttributeCount**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeCount()) Returns the count of attributes on this START\_ELEMENT, this method is only valid on a START\_ELEMENT or ATTRIBUTE. This count excludes namespace definitions. Attribute indices are zero-based.

**Specified by:**[getAttributeCount](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeCount()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**returns the number of attributes

### getAttributeName

public [QName](http://docs.google.com/javax/xml/namespace/QName.html) **getAttributeName**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeName(int)) Returns the qname of the attribute at the provided index

**Specified by:**[getAttributeName](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeName(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the attribute **Returns:**the QName of the attribute

### getAttributePrefix

public [String](http://docs.google.com/java/lang/String.html) **getAttributePrefix**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributePrefix(int)) Returns the prefix of this attribute at the provided index

**Specified by:**[getAttributePrefix](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributePrefix(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the attribute **Returns:**the prefix of the attribute

### getAttributeNamespace

public [String](http://docs.google.com/java/lang/String.html) **getAttributeNamespace**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeNamespace(int)) Returns the namespace of the attribute at the provided index

**Specified by:**[getAttributeNamespace](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeNamespace(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the attribute **Returns:**the namespace URI (can be null)

### getAttributeLocalName

public [String](http://docs.google.com/java/lang/String.html) **getAttributeLocalName**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeLocalName(int)) Returns the localName of the attribute at the provided index

**Specified by:**[getAttributeLocalName](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeLocalName(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the attribute **Returns:**the localName of the attribute

### getAttributeType

public [String](http://docs.google.com/java/lang/String.html) **getAttributeType**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeType(int)) Returns the XML type of the attribute at the provided index

**Specified by:**[getAttributeType](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeType(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the attribute **Returns:**the XML type of the attribute

### getAttributeValue

public [String](http://docs.google.com/java/lang/String.html) **getAttributeValue**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeValue(int)) Returns the value of the attribute at the index

**Specified by:**[getAttributeValue](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getAttributeValue(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the attribute **Returns:**the attribute value

### isAttributeSpecified

public boolean **isAttributeSpecified**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isAttributeSpecified(int)) Returns a boolean which indicates if this attribute was created by default

**Specified by:**[isAttributeSpecified](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isAttributeSpecified(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the attribute **Returns:**true if this is a default attribute

### getNamespaceCount

public int **getNamespaceCount**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceCount()) Returns the count of namespaces declared on this START\_ELEMENT or END\_ELEMENT, this method is only valid on a START\_ELEMENT, END\_ELEMENT or NAMESPACE. On an END\_ELEMENT the count is of the namespaces that are about to go out of scope. This is the equivalent of the information reported by SAX callback for an end element event.

**Specified by:**[getNamespaceCount](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceCount()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**returns the number of namespace declarations on this specific element

### getNamespacePrefix

public [String](http://docs.google.com/java/lang/String.html) **getNamespacePrefix**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespacePrefix(int)) Returns the prefix for the namespace declared at the index. Returns null if this is the default namespace declaration

**Specified by:**[getNamespacePrefix](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespacePrefix(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the namespace declaration **Returns:**returns the namespace prefix

### getNamespaceURI

public [String](http://docs.google.com/java/lang/String.html) **getNamespaceURI**(int index)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceURI(int)) Returns the uri for the namespace declared at the index.

**Specified by:**[getNamespaceURI](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceURI(int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**index - the position of the namespace declaration **Returns:**returns the namespace uri

### getEventType

public int **getEventType**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getEventType()) Returns an integer code that indicates the type of the event the cursor is pointing to.

**Specified by:**[getEventType](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getEventType()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

### getText

public [String](http://docs.google.com/java/lang/String.html) **getText**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getText()) Returns the current value of the parse event as a string, this returns the string value of a CHARACTERS event, returns the value of a COMMENT, the replacement value for an ENTITY\_REFERENCE, the string value of a CDATA section, the string value for a SPACE event, or the String value of the internal subset of the DTD. If an ENTITY\_REFERENCE has been resolved, any character data will be reported as CHARACTERS events.

**Specified by:**[getText](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getText()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the current text or null

### getTextCharacters

public int **getTextCharacters**(int sourceStart,  
 char[] target,  
 int targetStart,  
 int length)  
 throws [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextCharacters(int,%20char%5B%5D,%20int,%20int)) Gets the the text associated with a CHARACTERS, SPACE or CDATA event. Text starting a "sourceStart" is copied into "target" starting at "targetStart". Up to "length" characters are copied. The number of characters actually copied is returned. The "sourceStart" argument must be greater or equal to 0 and less than or equal to the number of characters associated with the event. Usually, one requests text starting at a "sourceStart" of 0. If the number of characters actually copied is less than the "length", then there is no more text. Otherwise, subsequent calls need to be made until all text has been retrieved. For example: int length = 1024; char[] myBuffer = new char[ length ]; for ( int sourceStart = 0 ; ; sourceStart += length ) { int nCopied = stream.getTextCharacters( sourceStart, myBuffer, 0, length ); if (nCopied < length) break; } XMLStreamException may be thrown if there are any XML errors in the underlying source. The "targetStart" argument must be greater than or equal to 0 and less than the length of "target", Length must be greater than 0 and "targetStart + length" must be less than or equal to length of "target".

**Specified by:**[getTextCharacters](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextCharacters(int,%20char%5B%5D,%20int,%20int)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**sourceStart - the index of the first character in the source array to copytarget - the destination arraytargetStart - the start offset in the target arraylength - the number of characters to copy **Returns:**the number of characters actually copied **Throws:** [XMLStreamException](http://docs.google.com/javax/xml/stream/XMLStreamException.html) - if the underlying XML source is not well-formed

### getTextCharacters

public char[] **getTextCharacters**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextCharacters()) Returns an array which contains the characters from this event. This array should be treated as read-only and transient. I.e. the array will contain the text characters until the XMLStreamReader moves on to the next event. Attempts to hold onto the character array beyond that time or modify the contents of the array are breaches of the contract for this interface.

**Specified by:**[getTextCharacters](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextCharacters()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the current text or an empty array

### getTextStart

public int **getTextStart**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextStart()) Returns the offset into the text character array where the first character (of this text event) is stored.

**Specified by:**[getTextStart](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextStart()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

### getTextLength

public int **getTextLength**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextLength()) Returns the length of the sequence of characters for this Text event within the text character array.

**Specified by:**[getTextLength](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getTextLength()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

### getEncoding

public [String](http://docs.google.com/java/lang/String.html) **getEncoding**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getEncoding()) Return input encoding if known or null if unknown.

**Specified by:**[getEncoding](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getEncoding()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the encoding of this instance or null

### hasText

public boolean **hasText**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#hasText()) Return true if the current event has text, false otherwise The following events have text: CHARACTERS,DTD ,ENTITY\_REFERENCE, COMMENT, SPACE

**Specified by:**[hasText](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#hasText()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

### getLocation

public [Location](http://docs.google.com/javax/xml/stream/Location.html) **getLocation**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getLocation()) Return the current location of the processor. If the Location is unknown the processor should return an implementation of Location that returns -1 for the location and null for the publicId and systemId. The location information is only valid until next() is called.

**Specified by:**[getLocation](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getLocation()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

### getName

public [QName](http://docs.google.com/javax/xml/namespace/QName.html) **getName**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getName()) Returns a QName for the current START\_ELEMENT or END\_ELEMENT event

**Specified by:**[getName](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getName()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the QName for the current START\_ELEMENT or END\_ELEMENT event

### getLocalName

public [String](http://docs.google.com/java/lang/String.html) **getLocalName**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getLocalName()) Returns the (local) name of the current event. For START\_ELEMENT or END\_ELEMENT returns the (local) name of the current element. For ENTITY\_REFERENCE it returns entity name. The current event must be START\_ELEMENT or END\_ELEMENT, or ENTITY\_REFERENCE

**Specified by:**[getLocalName](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getLocalName()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the localName

### hasName

public boolean **hasName**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#hasName()) returns true if the current event has a name (is a START\_ELEMENT or END\_ELEMENT) returns false otherwise

**Specified by:**[hasName](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#hasName()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html)

### getNamespaceURI

public [String](http://docs.google.com/java/lang/String.html) **getNamespaceURI**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceURI()) If the current event is a START\_ELEMENT or END\_ELEMENT this method returns the URI of the prefix or the default namespace. Returns null if the event does not have a prefix.

**Specified by:**[getNamespaceURI](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getNamespaceURI()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the URI bound to this elements prefix, the default namespace, or null

### getPrefix

public [String](http://docs.google.com/java/lang/String.html) **getPrefix**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getPrefix()) Returns the prefix of the current event or null if the event does not have a prefix

**Specified by:**[getPrefix](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getPrefix()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the prefix or null

### getVersion

public [String](http://docs.google.com/java/lang/String.html) **getVersion**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getVersion()) Get the xml version declared on the xml declaration Returns null if none was declared

**Specified by:**[getVersion](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getVersion()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the XML version or null

### isStandalone

public boolean **isStandalone**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isStandalone()) Get the standalone declaration from the xml declaration

**Specified by:**[isStandalone](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#isStandalone()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**true if this is standalone, or false otherwise

### standaloneSet

public boolean **standaloneSet**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#standaloneSet()) Checks if standalone was set in the document

**Specified by:**[standaloneSet](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#standaloneSet()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**true if standalone was set in the document, or false otherwise

### getCharacterEncodingScheme

public [String](http://docs.google.com/java/lang/String.html) **getCharacterEncodingScheme**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getCharacterEncodingScheme()) Returns the character encoding declared on the xml declaration Returns null if none was declared

**Specified by:**[getCharacterEncodingScheme](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getCharacterEncodingScheme()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the encoding declared in the document or null

### getPITarget

public [String](http://docs.google.com/java/lang/String.html) **getPITarget**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getPITarget()) Get the target of a processing instruction

**Specified by:**[getPITarget](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getPITarget()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the target or null

### getPIData

public [String](http://docs.google.com/java/lang/String.html) **getPIData**()

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getPIData()) Get the data section of a processing instruction

**Specified by:**[getPIData](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getPIData()) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Returns:**the data or null

### getProperty

public [Object](http://docs.google.com/java/lang/Object.html) **getProperty**([String](http://docs.google.com/java/lang/String.html) name)

**Description copied from interface:** [**XMLStreamReader**](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getProperty(java.lang.String)) Get the value of a feature/property from the underlying implementation

**Specified by:**[getProperty](http://docs.google.com/javax/xml/stream/XMLStreamReader.html#getProperty(java.lang.String)) in interface [XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) **Parameters:**name - The name of the property, may not be null **Returns:**The value of the property

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/StreamReaderDelegate.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/javax/xml/stream/util/EventReaderDelegate.html)   [**NEXT CLASS**](http://docs.google.com/javax/xml/stream/util/XMLEventAllocator.html) | [**FRAMES**](http://docs.google.com/index.html?javax/xml/stream/util/StreamReaderDelegate.html)    [**NO FRAMES**](http://docs.google.com/StreamReaderDelegate.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#4d34og8) | [METHOD](#3rdcrjn) |

[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).