

# **XBinder**

---

XML Schema Compiler  
Version 1.4  
C++ XML Runtime  
Reference Manual



The software described in this document is furnished under a license agreement and may be used only in accordance with the terms of this agreement.

### **Copyright Notice**

Copyright ©1997-2008 Objective Systems, Inc. All rights reserved.

This document may be distributed in any form, electronic or otherwise, provided that it is distributed in its entirety and that the copyright and this notice are included.

### **Author's Contact Information**

Comments, suggestions, and inquiries regarding XBinder may be submitted via electronic mail to [info@obj-sys.com](mailto:info@obj-sys.com).



# Contents

<b>1</b>	<b><a href="#">XBinder Main Page</a></b>	<b>1</b>
<b>2</b>	<b><a href="#">XBinder Hierarchical Index</a></b>	<b>2</b>
2.1	<a href="#">XBinder Class Hierarchy</a>	2
<b>3</b>	<b><a href="#">XBinder Class Index</a></b>	<b>3</b>
3.1	<a href="#">XBinder Class List</a>	3
<b>4</b>	<b><a href="#">XBinder File Index</a></b>	<b>4</b>
4.1	<a href="#">XBinder File List</a>	4
<b>5</b>	<b><a href="#">XBinder Class Documentation</a></b>	<b>5</b>
5.1	<a href="#">OSXMLAnyHandler Class Reference</a>	5
5.2	<a href="#">OSXMLAnyTypeHandler Class Reference</a>	7
5.3	<a href="#">OSXMLBase Class Reference</a>	9
5.4	<a href="#">OSXMLBasePtr Class Reference</a>	10
5.5	<a href="#">OSXMLContentHandler Class Reference</a>	11
5.6	<a href="#">OSXMLDefaultHandler Class Reference</a>	13
5.7	<a href="#">OSXMLDefaultHandler::ErrorInfo Struct Reference</a>	16
5.8	<a href="#">OSXMLDefaultHandlerIF Class Reference</a>	17
5.9	<a href="#">OSXMLDefaultHandlerPtr Class Reference</a>	18
5.10	<a href="#">OSXMLErrorHandler Class Reference</a>	19
5.11	<a href="#">OSXMLErrorInfo Class Reference</a>	21
5.12	<a href="#">OSXMLNamespaceClass Class Reference</a>	22
5.13	<a href="#">OSXMLParserCtxt Class Reference</a>	24
5.14	<a href="#">OSXMLParserCtxtIF Class Reference</a>	25
5.15	<a href="#">OSXMLReaderClass Class Reference</a>	26
5.16	<a href="#">OSXMLSimpleTypeHandler Class Reference</a>	28
5.17	<a href="#">OSXMLSoapHandler Class Reference</a>	30
5.18	<a href="#">OSXMLStrListHandler Class Reference</a>	32

5.19	OSXSDGlobalElement Class Reference	33
<b>6</b>	<b>XBinder File Documentation</b>	<b>39</b>
6.1	rtSaxCppAny.h File Reference	39
6.2	rtSaxCppAnyType.h File Reference	40
6.3	rtSaxCppSimpleType.h File Reference	41
6.4	rtSaxCppSoap.h File Reference	42
6.5	rtSaxCppStrList.h File Reference	43
6.6	rtXmlCppEncFuncs.h File Reference	44
6.7	rtXmlCppMsgBuf.h File Reference	47
6.8	rtXmlCppNamespace.h File Reference	48
6.9	rtXmlCppXSDElement.h File Reference	49
6.10	rtXmlpCppDecFuncs.h File Reference	50

# Chapter 1

## XBinder Main Page

### C++ XML Runtime Library Classes

The **C++ XML run-time classes** are wrapper classes that provide an object-oriented interface to the common C XML run-time library functions. The categories of classes provided are as follows:

- XML encode functions.
- XML decode functions.
- SAX interfaces.
- Objects for global elements, namespaces, and message buffers.

## Chapter 2

# XBinder Hierarchical Index

### 2.1 XBinder Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

OSXMLBase . . . . .	9
OSXMLReaderClass . . . . .	26
OSXMLBasePtr . . . . .	10
OSXMLContentHandler . . . . .	11
OSXMLDefaultHandlerIF . . . . .	17
OSXMLDefaultHandler . . . . .	13
OSXMLAnyHandler . . . . .	5
OSXMLAnyTypeHandler . . . . .	7
OSXMLSimpleTypeHandler . . . . .	28
OSXMLSoapHandler . . . . .	30
OSXMLDefaultHandler::ErrorInfo . . . . .	16
OSXMLDefaultHandlerPtr . . . . .	18
OSXMLErrorHandler . . . . .	19
OSXMLErrorInfo . . . . .	21
OSXMLDefaultHandlerIF . . . . .	17
OSXMLNamespaceClass . . . . .	22
OSXMLParserCtxtIF . . . . .	25
OSXMLParserCtxt . . . . .	24
OSXMLStrListHandler . . . . .	32
OSXSDDGlobalElement . . . . .	33

# Chapter 3

## XBinder Class Index

### 3.1 XBinder Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">OSXMLAnyHandler</a> . . . . .	5
<a href="#">OSXMLAnyTypeHandler</a> . . . . .	7
<a href="#">OSXMLBase</a> . . . . .	9
<a href="#">OSXMLBasePtr</a> . . . . .	10
<a href="#">OSXMLContentHandler</a> (Receive notification of general document events ) . . . . .	11
<a href="#">OSXMLDefaultHandler</a> (This class is derived from the SAX class DefaultHandler base class ) . . . . .	13
<a href="#">OSXMLDefaultHandler::ErrorInfo</a> . . . . .	16
<a href="#">OSXMLDefaultHandlerIF</a> (This class is derived from the SAX class DefaultHandler base class ) . . . . .	17
<a href="#">OSXMLDefaultHandlerPtr</a> . . . . .	18
<a href="#">OSXMLErrorHandler</a> . . . . .	19
<a href="#">OSXMLErrorInfo</a> . . . . .	21
<a href="#">OSXMLNamespaceClass</a> (This class is used to hold an XML namespace prefix to URI mapping ) . . . . .	22
<a href="#">OSXMLParserCtxt</a> . . . . .	24
<a href="#">OSXMLParserCtxtIF</a> . . . . .	25
<a href="#">OSXMLReaderClass</a> . . . . .	26
<a href="#">OSXMLSimpleTypeHandler</a> . . . . .	28
<a href="#">OSXMLSoapHandler</a> . . . . .	30
<a href="#">OSXMLStrListHandler</a> ( <a href="#">OSXMLStrListHandler</a> ) . . . . .	32
<a href="#">OSXSDGlobalElement</a> (XSD global element base class ) . . . . .	33

# Chapter 4

## XBinder File Index

### 4.1 XBinder File List

Here is a list of all documented files with brief descriptions:

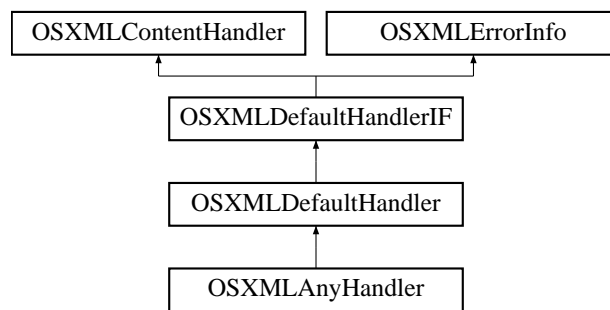
<a href="#">rtSaxCppAny.h</a> . . . . .	39
<a href="#">rtSaxCppAnyType.h</a> . . . . .	40
<b>rtSaxCppParser.h</b> . . . . .	??
<b>rtSaxCppParserIF.h</b> . . . . .	??
<a href="#">rtSaxCppSimpleType.h</a> . . . . .	41
<a href="#">rtSaxCppSoap.h</a> . . . . .	42
<a href="#">rtSaxCppStrList.h</a> . . . . .	43
<a href="#">rtXmlCppEncFuncs.h</a> (XML low-level C++ encode functions) . . . . .	44
<a href="#">rtXmlCppMsgBuf.h</a> (This file is deprecated) . . . . .	47
<a href="#">rtXmlCppNamespace.h</a> (XML namespace handling structures and function definitions) . . . . .	48
<a href="#">rtXmlCppXSDElement.h</a> (C++ run-time XML schema global element class definition) . . . . .	49
<a href="#">rtXmlpCppDecFuncs.h</a> (XML low-level C++ decode functions) . . . . .	50

## Chapter 5

# XBinder Class Documentation

### 5.1 OSXMLAnyHandler Class Reference

Inheritance diagram for OSXMLAnyHandler::



#### Public Member Functions

- virtual int [startElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \*attrs)  
*Receive notification of the beginning of an element.*
- virtual int [endElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname)  
*Receive notification of the end of an element.*

#### 5.1.1 Detailed Description

Definition at line 40 of file rtSaxCppAny.h.

## 5.1.2 Member Function Documentation

**5.1.2.1** `virtual int OSXMLAnyHandler::startElement (const OSUTF8CHAR *const uri, const OSUTF8CHAR *const localname, const OSUTF8CHAR *const qname, const OSUTF8CHAR *const * attrs)` [virtual]

Receive notification of the beginning of an element.

The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding `endElement()` event for every `startElement()` event (even when the element is empty). All of the element's content will be reported, in order, before the corresponding `endElement()` event.

### Parameters:

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

*attrs* The attributes name/value pairs attached to the element, if any.

### See also:

[endElement](#)

Reimplemented from [OSXMLDefaultHandler](#).

**5.1.2.2** `virtual int OSXMLAnyHandler::endElement (const OSUTF8CHAR *const uri, const OSUTF8CHAR *const localname, const OSUTF8CHAR *const qname)` [virtual]

Receive notification of the end of an element.

The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding `startElement()` event for every `endElement()` event (even when the element is empty).

### Parameters:

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

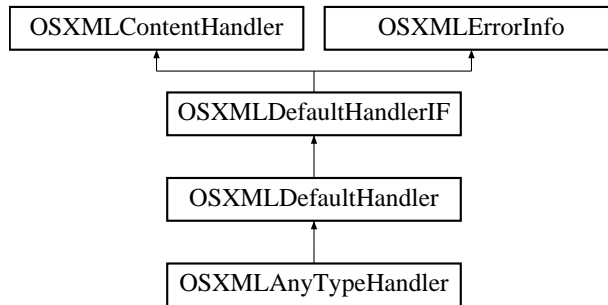
Reimplemented from [OSXMLDefaultHandler](#).

The documentation for this class was generated from the following file:

- [rtSaxCppAny.h](#)

## 5.2 OSXMLAnyTypeHandler Class Reference

Inheritance diagram for OSXMLAnyTypeHandler::



### Public Member Functions

- virtual int [startElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \*attrs)  
*Receive notification of the beginning of an element.*
- virtual int [endElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname)  
*Receive notification of the end of an element.*

### 5.2.1 Detailed Description

Definition at line 39 of file rtSaxCppAnyType.h.

### 5.2.2 Member Function Documentation

**5.2.2.1** virtual int OSXMLAnyTypeHandler::startElement (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \* attrs) [virtual]

Receive notification of the beginning of an element.

The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding [endElement\(\)](#) event for every [startElement\(\)](#) event (even when the element is empty). All of the element's content will be reported, in order, before the corresponding [endElement\(\)](#) event.

#### Parameters:

- uri* The URI of the associated namespace for this element
- localname* The local part of the element name
- qname* The QName of this element
- attrs* The attributes name/value pairs attached to the element, if any.

#### See also:

[endElement](#)

Reimplemented from [OSXMLDefaultHandler](#).

**5.2.2.2 virtual int OSXMLAnyTypeHandler::endElement (const OSUTF8CHAR \*const *uri*, const OSUTF8CHAR \*const *localname*, const OSUTF8CHAR \*const *qname*)** [virtual]

Receive notification of the end of an element.

The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding [startElement\(\)](#) event for every [endElement\(\)](#) event (even when the element is empty).

**Parameters:**

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

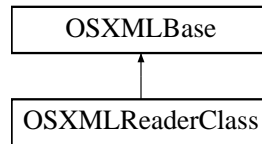
Reimplemented from [OSXMLDefaultHandler](#).

The documentation for this class was generated from the following file:

- [rtSaxCppAnyType.h](#)

## 5.3 OSXMLBase Class Reference

Inheritance diagram for OSXMLBase::



### Protected Member Functions

- [OSXMLBase \(\)](#)
- virtual [~OSXMLBase \(\)](#)

#### 5.3.1 Detailed Description

Definition at line 186 of file `rtSaxCppParserIF.h`.

The documentation for this class was generated from the following file:

- `rtSaxCppParserIF.h`

## 5.4 OSXMLBasePtr Class Reference

### Public Member Functions

- [OSXMLBasePtr \(\)](#)
- [OSXMLBasePtr \(OSXMLBase \\*ptr\)](#)
- [~OSXMLBasePtr \(\)](#)
- [operator OSXMLBase \\* \(\) const](#)
- [OSXMLBase \\* operator= \(OSXMLBase \\*ptr\)](#)

### 5.4.1 Detailed Description

Definition at line 35 of file `rtSaxCppParser.h`.

The documentation for this class was generated from the following file:

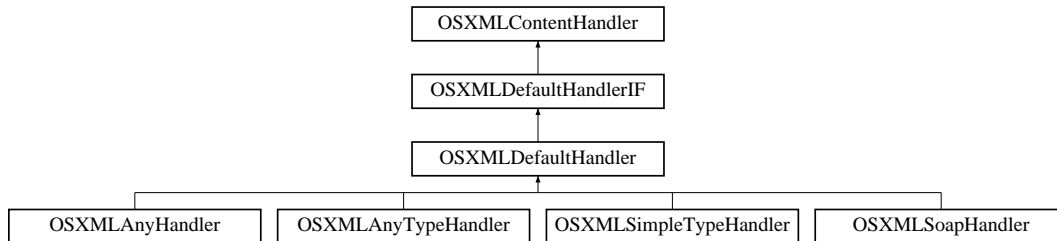
- `rtSaxCppParser.h`

## 5.5 OSXMLContentHandler Class Reference

Receive notification of general document events.

```
#include <rtSaxCppParserIF.h>
```

Inheritance diagram for OSXMLContentHandler::



### Public Member Functions

- virtual `~OSXMLContentHandler()`

#### The virtual document handler interface

- virtual int `characters` (const OSUTF8CHAR \*const chars, unsigned int length)=0  
*Receive notification of character data.*
- virtual int `endElement` (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname)=0  
*Receive notification of the end of an element.*
- virtual int `startElement` (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \*attrs)=0  
*Receive notification of the beginning of an element.*

### 5.5.1 Detailed Description

Receive notification of general document events.

Definition at line 112 of file rtSaxCppParserIF.h.

### 5.5.2 Member Function Documentation

#### 5.5.2.1 virtual int OSXMLContentHandler::characters (const OSUTF8CHAR \*const chars, unsigned int length) [pure virtual]

Receive notification of character data.

The Parser will call this method to report each chunk of character data. SAX parsers may return all contiguous character data in a single chunk, or they may split it into several chunks; however, all of the characters in any single event must come from the same external entity, so that the Locator provides useful information.

**Parameters:**

*chars* The characters from the XML document.

*length* The length of chars.

Implemented in [OSXMLDefaultHandler](#), [OSXMLSimpleTypeHandler](#), and [OSXMLSoapHandler](#).

**5.5.2.2 virtual int OSXMLContentHandler::endElement (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname) [pure virtual]**

Receive notification of the end of an element.

The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding [startElement\(\)](#) event for every [endElement\(\)](#) event (even when the element is empty).

**Parameters:**

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

Implemented in [OSXMLAnyHandler](#), [OSXMLAnyTypeHandler](#), [OSXMLDefaultHandler](#), [OSXMLSimpleTypeHandler](#), and [OSXMLSoapHandler](#).

**5.5.2.3 virtual int OSXMLContentHandler::startElement (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \* attrs) [pure virtual]**

Receive notification of the beginning of an element.

The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding [endElement\(\)](#) event for every [startElement\(\)](#) event (even when the element is empty). All of the element's content will be reported, in order, before the corresponding [endElement\(\)](#) event.

**Parameters:**

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

*attrs* The attributes name/value pairs attached to the element, if any.

**See also:**

[endElement](#)

Implemented in [OSXMLAnyHandler](#), [OSXMLAnyTypeHandler](#), [OSXMLDefaultHandler](#), [OSXMLSimpleTypeHandler](#), and [OSXMLSoapHandler](#).

The documentation for this class was generated from the following file:

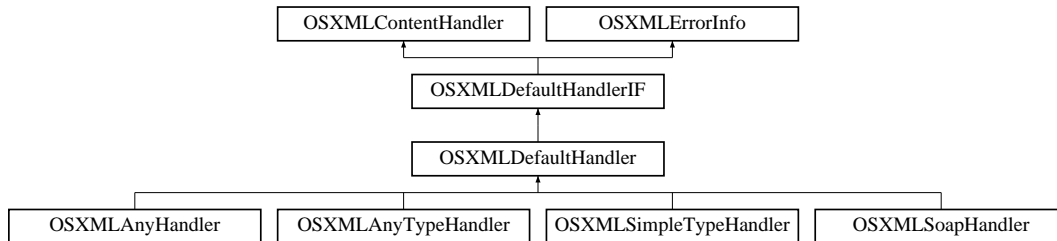
- [rtSaxCppParserIF.h](#)

## 5.6 OSXMLDefaultHandler Class Reference

This class is derived from the SAX class DefaultHandler base class.

```
#include <rtSaxCppParser.h>
```

Inheritance diagram for OSXMLDefaultHandler::



### Public Member Functions

- **OSXMLDefaultHandler** (OSRTCtxtPtr \*pContext, const OSUTF8CHAR \*elemName=0, OSINT32 level=0)
- virtual **~OSXMLDefaultHandler** ()
- virtual int **startElement** (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \*attrs)  
*Receive notification of the beginning of an element.*
- virtual int **characters** (const OSUTF8CHAR \*const chars, unsigned int length)  
*Receive notification of character data.*
- virtual int **endElement** (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname)  
*Receive notification of the end of an element.*
- OSINT16 **getState** ()  
*This method returns the current state of the decoding process.*
- virtual void **init** (int level=0)
- void **setElemName** (const OSUTF8CHAR \*elemName)
- OSBOOL **isComplete** ()

### Protected Attributes

- OSRTCtxtPtr **mpContext**
- const OSUTF8CHAR \* **mpElemName**
- OSINT32 **mLevel**
- OSINT16 **mStartLevel**
- OSINT16 **mReqElemCnt**
- OSINT16 **mCurrElemIdx**
- OSINT16 **mState**

## Classes

- struct [ErrorInfo](#)

### 5.6.1 Detailed Description

This class is derived from the SAX class `DefaultHandler` base class.

It contains variables and methods specific to decoding XML messages. It is used to intercept standard SAX parser events, such as `startElement`, `characters`, `endElement`. This class is used as a base class for `XBinder` generated global element control classes (`<elem>_CC`).

Definition at line 58 of file `rtSaxCppParser.h`.

### 5.6.2 Member Function Documentation

#### 5.6.2.1 `virtual int OSXMLDefaultHandler::startElement (const OSUTF8CHAR *const uri, const OSUTF8CHAR *const localname, const OSUTF8CHAR *const qname, const OSUTF8CHAR *const * attrs) [virtual]`

Receive notification of the beginning of an element.

The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding `endElement()` event for every `startElement()` event (even when the element is empty). All of the element's content will be reported, in order, before the corresponding `endElement()` event.

#### Parameters:

- uri* The URI of the associated namespace for this element
- localname* The local part of the element name
- qname* The QName of this element
- attrs* The attributes name/value pairs attached to the element, if any.

#### See also:

[endElement](#)

Implements [OSXMLContentHandler](#).

Reimplemented in [OSXMLAnyHandler](#), [OSXMLAnyTypeHandler](#), [OSXMLSimpleTypeHandler](#), and [OSXMLSoapHandler](#).

#### 5.6.2.2 `virtual int OSXMLDefaultHandler::characters (const OSUTF8CHAR *const chars, unsigned int length) [virtual]`

Receive notification of character data.

The Parser will call this method to report each chunk of character data. SAX parsers may return all contiguous character data in a single chunk, or they may split it into several chunks; however, all of the characters in any single event must come from the same external entity, so that the Locator provides useful information.

#### Parameters:

- chars* The characters from the XML document.

*length* The length of chars.

Implements [OSXMLContentHandler](#).

Reimplemented in [OSXMLSimpleTypeHandler](#), and [OSXMLSoapHandler](#).

**5.6.2.3 virtual int OSXMLDefaultHandler::endElement (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname) [virtual]**

Receive notification of the end of an element.

The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding [startElement\(\)](#) event for every [endElement\(\)](#) event (even when the element is empty).

**Parameters:**

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

Implements [OSXMLContentHandler](#).

Reimplemented in [OSXMLAnyHandler](#), [OSXMLAnyTypeHandler](#), [OSXMLSimpleTypeHandler](#), and [OSXMLSoapHandler](#).

**5.6.2.4 OSINT16 OSXMLDefaultHandler::getState () [inline]**

This method returns the current state of the decoding process.

**Returns:**

The state of the decoding process as type `OSXMLState`. Can be `XMLINIT`, `XMLSTART`, `XMLDATA`, or `XMLEND`.

Definition at line 124 of file `rtSaxCppParser.h`.

The documentation for this class was generated from the following file:

- `rtSaxCppParser.h`

## 5.7 OSXMLDefaultHandler::ErrorInfo Struct Reference

### Public Member Functions

- [ErrorInfo](#) ()

### Public Attributes

- int [stat](#)
- const char \* [file](#)
- int [line](#)

### 5.7.1 Detailed Description

Definition at line 69 of file `rtSaxCppParser.h`.

The documentation for this struct was generated from the following file:

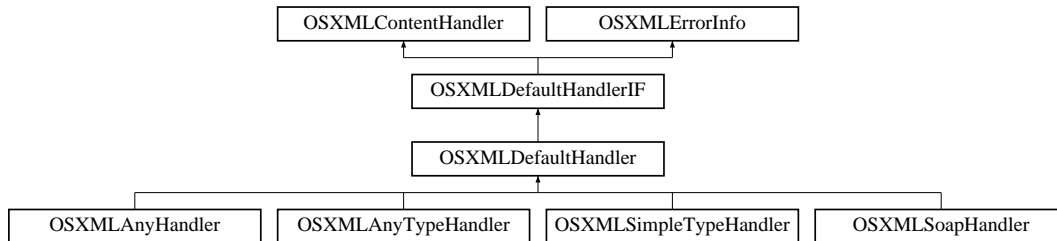
- `rtSaxCppParser.h`

## 5.8 OSXMLDefaultHandlerIF Class Reference

This class is derived from the SAX class DefaultHandler base class.

```
#include <rtSaxCppParserIF.h>
```

Inheritance diagram for OSXMLDefaultHandlerIF::



### Public Member Functions

- virtual [~OSXMLDefaultHandlerIF](#) ()

#### 5.8.1 Detailed Description

This class is derived from the SAX class DefaultHandler base class.

It contains variables and methods specific to decoding XML messages. It is used to intercept standard SAX parser events, such as startElement, characters, endElement. This class is used as a base class for XBinder generated global element control classes (<elem>\_CC).

Definition at line 260 of file rtSaxCppParserIF.h.

The documentation for this class was generated from the following file:

- rtSaxCppParserIF.h

## 5.9 OSXMLDefaultHandlerPtr Class Reference

### Public Member Functions

- [OSXMLDefaultHandlerPtr](#) ()
- [OSXMLDefaultHandlerPtr](#) ([OSXMLDefaultHandler](#) \*ptr)
- [~OSXMLDefaultHandlerPtr](#) ()
- [operator OSXMLDefaultHandler \\*](#) ()
- [operator const OSXMLDefaultHandler \\*](#) () const
- [OSXMLDefaultHandler \\*](#) [operator →](#) () const
- [OSXMLDefaultHandler \\*](#) [operator=](#) ([OSXMLDefaultHandler](#) \*ptr)
- [int operator==](#) (const [OSXMLDefaultHandler](#) \*ptr) const
- [int operator!=](#) (const [OSXMLDefaultHandler](#) \*ptr) const
- [int operator!](#) () const

### Friends

- [int operator!=](#) (const void \*ptr, const [OSXMLDefaultHandlerPtr](#) &ptr2)
- [int operator==](#) (const void \*ptr, const [OSXMLDefaultHandlerPtr](#) &ptr2)

### 5.9.1 Detailed Description

Definition at line 147 of file `rtSaxCppParser.h`.

The documentation for this class was generated from the following file:

- `rtSaxCppParser.h`

## 5.10 OSXMLErrorHandler Class Reference

### Public Member Functions

- virtual `~OSXMLErrorHandler ()`

#### The error handler interface

- virtual void `warning ()=0`  
*Receive notification of a warning.*
- virtual void `error ()=0`  
*Receive notification of a recoverable error.*
- virtual void `fatalError ()=0`  
*Receive notification of a non-recoverable error.*
- virtual void `resetErrors ()=0`  
*Reset the Error handler object on its reuse.*

### 5.10.1 Detailed Description

Definition at line 42 of file `rtSaxCppParserIF.h`.

### 5.10.2 Member Function Documentation

#### 5.10.2.1 virtual void OSXMLErrorHandler::warning () [pure virtual]

Receive notification of a warning.

SAX parsers will use this method to report conditions that are not errors or fatal errors as defined by the XML 1.0 recommendation. The default behaviour is to take no action.

The SAX parser must continue to provide normal parsing events after invoking this method: it should still be possible for the application to process the document through to the end.

#### 5.10.2.2 virtual void OSXMLErrorHandler::error () [pure virtual]

Receive notification of a recoverable error.

This corresponds to the definition of "error" in section 1.2 of the W3C XML 1.0 Recommendation. For example, a validating parser would use this callback to report the violation of a validity constraint. The default behaviour is to take no action.

The SAX parser must continue to provide normal parsing events after invoking this method: it should still be possible for the application to process the document through to the end. If the application cannot do so, then the parser should report a fatal error even if the XML 1.0 recommendation does not require it to do so.

#### 5.10.2.3 virtual void OSXMLErrorHandler::fatalError () [pure virtual]

Receive notification of a non-recoverable error.

This corresponds to the definition of "fatal error" in section 1.2 of the W3C XML 1.0 Recommendation. For example, a parser would use this callback to report the violation of a well-formedness constraint.

The application must assume that the document is unusable after the parser has invoked this method, and should continue (if at all) only for the sake of collecting additional error messages: in fact, SAX parsers are free to stop reporting any other events once this method has been invoked.

#### **5.10.2.4 virtual void OSXMLErrorHandler::resetErrors () [pure virtual]**

Reset the Error handler object on its reuse.

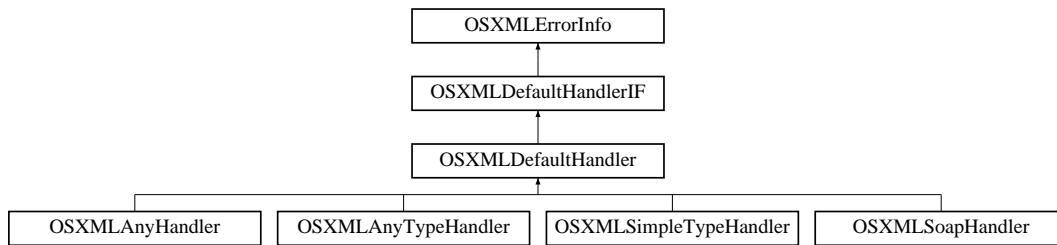
This method helps in resetting the Error handler object implementation defaults each time the Error handler is begun.

The documentation for this class was generated from the following file:

- rtSaxCppParserIF.h

## 5.11 OSXMLErrorInfo Class Reference

Inheritance diagram for OSXMLErrorInfo::



### Public Member Functions

- virtual [~OSXMLErrorInfo](#) ()

#### 5.11.1 Detailed Description

Definition at line 34 of file rtSaxCppParserIF.h.

The documentation for this class was generated from the following file:

- rtSaxCppParserIF.h

## 5.12 OSXMLNamespaceClass Class Reference

This class is used to hold an XML namespace prefix to URI mapping.

```
#include <rtXmlCppNamespace.h>
```

### Public Member Functions

- [OSXMLNamespaceClass \(\)](#)  
*The default constructor sets the namespace prefix and URI values to empty values.*
- [~OSXMLNamespaceClass \(\)](#)  
*The destructor deletes the prefix and uri string variables.*
- [OSXMLNamespaceClass \(const OSUTF8CHAR \\*nsPrefix, const OSUTF8CHAR \\*nsURI\)](#)  
*The parameterized constructor sets the namespace prefix and URI values to the given values.*
- [OSXMLNamespaceClass \(const OSUTF8CHAR \\*nsPrefix, size\\_t nsPrefixBytes, const OSUTF8CHAR \\*nsURI, size\\_t nsURIBytes\)](#)  
*The parameterized constructor sets the namespace prefix and URI values to the given values.*
- [OSXMLNamespaceClass \(const OSXMLNamespaceClass &o\)](#)  
*The copy constructor make a deep-copy of the prefix and URI values.*
- `const OSUTF8CHAR * getPrefix \(\) const`  
*This method is used to get the namespace prefix value.*
- `const OSUTF8CHAR * getURI \(\) const`  
*This method is used to get the namespace URI value.*
- `void setPrefix (const OSUTF8CHAR *nsPrefix)`  
*This method is used to set the namespace prefix value.*
- `void setURI (const OSUTF8CHAR *nsURI)`  
*This method is used to set the namespace URI value.*

### 5.12.1 Detailed Description

This class is used to hold an XML namespace prefix to URI mapping.

Definition at line 38 of file `rtXmlCppNamespace.h`.

### 5.12.2 Constructor & Destructor Documentation

#### 5.12.2.1 OSXMLNamespaceClass::OSXMLNamespaceClass (const OSUTF8CHAR \* nsPrefix, const OSUTF8CHAR \* nsURI)

The parameterized constructor sets the namespace prefix and URI values to the given values.

A deep copy of the values is done.

**Parameters:**

*nsPrefix* Namespace prefix value.

*nsURI* Namespace URI value.

**5.12.2.2 OSXMLNamespaceClass::OSXMLNamespaceClass (const OSUTF8CHAR \* *nsPrefix*, size\_t *nsPrefixBytes*, const OSUTF8CHAR \* *nsURI*, size\_t *nsURIBytes*)**

The parameterized constructor sets the namespace prefix and URI values to the given values.

A deep copy of the values is done.

**Parameters:**

*nsPrefix* Namespace prefix value.

*nsPrefixBytes* Namespace prefix value size in bytes.

*nsURI* Namespace URI value.

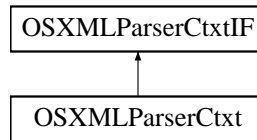
*nsURIBytes* Namespace URI value size in bytes.

The documentation for this class was generated from the following file:

- [rtXmlCppNamespace.h](#)

## 5.13 OSXMLParserCtxt Class Reference

Inheritance diagram for OSXMLParserCtxt::



### 5.13.1 Detailed Description

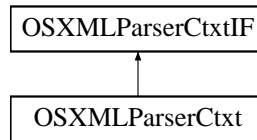
Definition at line 179 of file rtSaxCppParser.h.

The documentation for this class was generated from the following file:

- rtSaxCppParser.h

## 5.14 OSXMLParserCtxtIF Class Reference

Inheritance diagram for OSXMLParserCtxtIF::



### Public Member Functions

- virtual [~OSXMLParserCtxtIF](#) ()

#### 5.14.1 Detailed Description

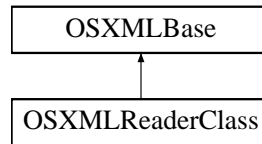
Definition at line 287 of file rtSaxCppParserIF.h.

The documentation for this class was generated from the following file:

- rtSaxCppParserIF.h

## 5.15 OSXMLReaderClass Class Reference

Inheritance diagram for OSXMLReaderClass::



### Public Member Functions

- virtual int `parse` (OSRTInputStreamIF &source)=0  
*Parse an XML document.*
- virtual int `parse` (const char \*const pBuffer, size\_t bufSize)=0  
*Parse an XML document from memory buffer.*
- virtual int `parse` (const char \*const systemId)=0  
*Parse an XML document from a system identifier (URI).*

### 5.15.1 Detailed Description

Definition at line 198 of file rtSaxCppParserIF.h.

### 5.15.2 Member Function Documentation

#### 5.15.2.1 virtual int OSXMLReaderClass::parse (OSRTInputStreamIF & source) [pure virtual]

Parse an XML document.

The application can use this method to instruct the SAX parser to begin parsing an XML document from any valid input source (a character stream, a byte stream, or a URI).

Applications may not invoke this method while a parse is in progress (they should create a new Parser instead for each additional XML document). Once a parse is complete, an application may reuse the same Parser object, possibly with a different input source.

#### Parameters:

*source* The input source for the top-level of the XML document.

#### 5.15.2.2 virtual int OSXMLReaderClass::parse (const char \*const pBuffer, size\_t bufSize) [pure virtual]

Parse an XML document from memory buffer.

#### Parameters:

*pBuffer* Buffer containing the XML data to be parsed.

*bufSize* Buffer size, in octets.

### 5.15.2.3 virtual int OSXMLReaderClass::parse (const char \*const *systemId*) [pure virtual]

Parse an XML document from a system identifier (URI).

This method is a shortcut for the common case of reading a document from a system identifier. It is the exact equivalent of the following:

```
parse(new URLInputSource(systemId));
```

If the system identifier is a URL, it must be fully resolved by the application before it is passed to the parser.

#### Parameters:

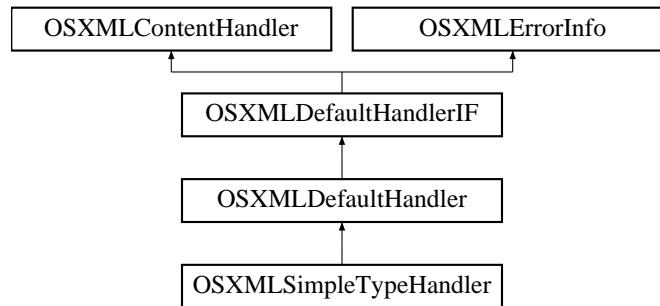
*systemId* The system identifier (URI).

The documentation for this class was generated from the following file:

- rtSaxCppParserIF.h

## 5.16 OSXMLSimpleTypeHandler Class Reference

Inheritance diagram for OSXMLSimpleTypeHandler::



### Public Member Functions

- virtual int [startElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \*attrs)  
*Receive notification of the beginning of an element.*
- virtual int [characters](#) (const OSUTF8CHAR \*const chars, unsigned int length)  
*Receive notification of character data.*
- virtual int [endElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname)  
*Receive notification of the end of an element.*

### 5.16.1 Detailed Description

Definition at line 39 of file rtSaxCppSimpleType.h.

### 5.16.2 Member Function Documentation

**5.16.2.1** virtual int OSXMLSimpleTypeHandler::startElement (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \* attrs) [virtual]

Receive notification of the beginning of an element.

The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding [endElement\(\)](#) event for every [startElement\(\)](#) event (even when the element is empty). All of the element's content will be reported, in order, before the corresponding [endElement\(\)](#) event.

#### Parameters:

- uri* The URI of the associated namespace for this element
- localname* The local part of the element name
- qname* The QName of this element

*attrs* The attributes name/value pairs attached to the element, if any.

**See also:**

[endElement](#)

Reimplemented from [OSXMLDefaultHandler](#).

### 5.16.2.2 `virtual int OSXMLSimpleTypeHandler::characters (const OSUTF8CHAR *const chars, unsigned int length)` [virtual]

Receive notification of character data.

The Parser will call this method to report each chunk of character data. SAX parsers may return all contiguous character data in a single chunk, or they may split it into several chunks; however, all of the characters in any single event must come from the same external entity, so that the Locator provides useful information.

**Parameters:**

*chars* The characters from the XML document.

*length* The length of chars.

Reimplemented from [OSXMLDefaultHandler](#).

### 5.16.2.3 `virtual int OSXMLSimpleTypeHandler::endElement (const OSUTF8CHAR *const uri, const OSUTF8CHAR *const localname, const OSUTF8CHAR *const qname)` [virtual]

Receive notification of the end of an element.

The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding [startElement\(\)](#) event for every [endElement\(\)](#) event (even when the element is empty).

**Parameters:**

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

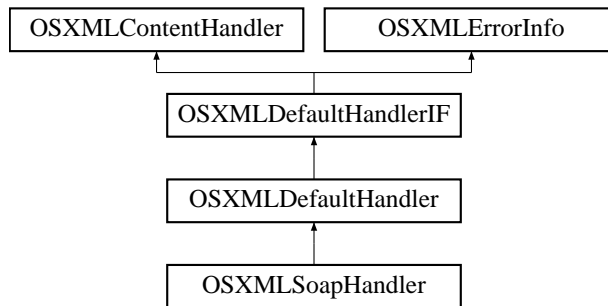
Reimplemented from [OSXMLDefaultHandler](#).

The documentation for this class was generated from the following file:

- [rtSaxCppSimpleType.h](#)

## 5.17 OSXMLSoapHandler Class Reference

Inheritance diagram for OSXMLSoapHandler::



### Public Member Functions

- virtual int [startElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \*attrs)  
*Receive notification of the beginning of an element.*
- virtual int [characters](#) (const OSUTF8CHAR \*const chars, unsigned int length)  
*Receive notification of character data.*
- virtual int [endElement](#) (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname)  
*Receive notification of the end of an element.*

### 5.17.1 Detailed Description

Definition at line 40 of file rtSaxCppSoap.h.

### 5.17.2 Member Function Documentation

**5.17.2.1** virtual int OSXMLSoapHandler::startElement (const OSUTF8CHAR \*const uri, const OSUTF8CHAR \*const localname, const OSUTF8CHAR \*const qname, const OSUTF8CHAR \*const \* attrs) [virtual]

Receive notification of the beginning of an element.

The Parser will invoke this method at the beginning of every element in the XML document; there will be a corresponding [endElement\(\)](#) event for every [startElement\(\)](#) event (even when the element is empty). All of the element's content will be reported, in order, before the corresponding [endElement\(\)](#) event.

#### Parameters:

- uri* The URI of the associated namespace for this element
- localname* The local part of the element name
- qname* The QName of this element

*attrs* The attributes name/value pairs attached to the element, if any.

**See also:**

[endElement](#)

Reimplemented from [OSXMLDefaultHandler](#).

**5.17.2.2 virtual int OSXMLSoapHandler::characters (const OSUTF8CHAR \*const *chars*, unsigned int *length*)** [virtual]

Receive notification of character data.

The Parser will call this method to report each chunk of character data. SAX parsers may return all contiguous character data in a single chunk, or they may split it into several chunks; however, all of the characters in any single event must come from the same external entity, so that the Locator provides useful information.

**Parameters:**

*chars* The characters from the XML document.

*length* The length of chars.

Reimplemented from [OSXMLDefaultHandler](#).

**5.17.2.3 virtual int OSXMLSoapHandler::endElement (const OSUTF8CHAR \*const *uri*, const OSUTF8CHAR \*const *localname*, const OSUTF8CHAR \*const *qname*)** [virtual]

Receive notification of the end of an element.

The SAX parser will invoke this method at the end of every element in the XML document; there will be a corresponding [startElement\(\)](#) event for every [endElement\(\)](#) event (even when the element is empty).

**Parameters:**

*uri* The URI of the associated namespace for this element

*localname* The local part of the element name

*qname* The QName of this element

Reimplemented from [OSXMLDefaultHandler](#).

The documentation for this class was generated from the following file:

- [rtSaxCppSoap.h](#)

## 5.18 OSXMLStrListHandler Class Reference

[OSXMLStrListHandler](#).

```
#include <rtSaxCppStrList.h>
```

### Static Public Member Functions

- static int [parseSTL](#) (OSCTXT \*pctxt, OSRTMEMBUF \*pMemBuf, OSRTObjListClass \*pStrList)
- static int [match](#) (OSCTXT \*)

#### 5.18.1 Detailed Description

[OSXMLStrListHandler](#).

Definition at line 41 of file [rtSaxCppStrList.h](#).

The documentation for this class was generated from the following file:

- [rtSaxCppStrList.h](#)

## 5.19 OSXSDGlobalElement Class Reference

XSD global element base class.

```
#include <rtXmlCppXSDElement.h>
```

### Public Member Functions

- [OSXSDGlobalElement](#) (OSRTMessageBufferIF &msgBuf)  
*This constructor sets the internal message buffer pointer to point at the given message buffer or stream object.*
- [OSXSDGlobalElement](#) (const [OSXSDGlobalElement](#) &o)  
*The copy constructor sets the internal message buffer pointer and context to point at the message buffer and context from the original OSCType object.*
- virtual [~OSXSDGlobalElement](#) ()  
*The virtual destructor does nothing.*
- int [decode](#) ()  
*The decode method decodes the message described by the encapsulated message buffer object.*
- virtual int [decodeFrom](#) (OSRTMessageBufferIF &)  
*The decodeFrom method decodes a message from the given message buffer or stream argument.*
- int [encode](#) ()  
*The encode method encodes a message using the encoding rules specified by the derived message buffer object.*
- virtual int [encodeTo](#) (OSRTMessageBufferIF &)  
*The encodeTo method encodes a message into the given message buffer or stream argument.*
- OSCTXT \* [getCtxtPtr](#) ()  
*The getCtxtPtr method returns the underlying C runtime context.*
- void \* [memAlloc](#) (size\_t numocts)  
*The memAlloc method allocates memory using the C runtime memory management functions.*
- void [memFreePtr](#) (void \*ptr)  
*The memFreePtr method frees the memory at a specific location.*
- void [setDefaultNamespace](#) (const OSUTF8CHAR \*uri)  
*The setDefaultNamespace method sets the default namespace for the element to the given value.*
- void [setDiag](#) (OSBOOL value=TRUE)  
*The setDiag method turns diagnostic tracing on or off.*
- void [setEncXSINamespace](#) (OSBOOL value=TRUE)  
*The setEncXSINamespace method sets a flag in the internal context that indicates the xsi namespace attribute must be encoded.*
- void [setNamespace](#) (const OSUTF8CHAR \*prefix, const OSUTF8CHAR \*uri)

*The setNamespace method adds or modifies the namespace with the given URI in the namespace list to contain the given prefix.*

- void [setNoNSSchemaLocation](#) (const OSUTF8CHAR \*uri)  
*The setNoNSSchemaLocation method adds an xsi:noNamespaceSchemaLocation attribute to the document.*
- void [setSchemaLocation](#) (const OSUTF8CHAR \*uri)  
*The setSchemaLocation method adds an xsi:schemaLocation attribute to the document.*
- void [setXSIType](#) (const OSUTF8CHAR \*typeName)  
*The setXSIType method sets a type name to be used in the xsi:type attribute in the top-level module element declaration.*
- int [validate](#) ()  
*The validate method validates the message described by the encapsulated message buffer object.*
- virtual int [validateFrom](#) (OSRTMessageBufferIF &)  
*The validateFrom method validates a message from the given message buffer or stream argument.*

## Protected Member Functions

- [OSXSDGlobalElement](#) ()  
*The default constructor sets the message pointer member variable to NULL and creates a new context object.*
- [OSXSDGlobalElement](#) (OSRTContext &ctxt)  
*This constructor sets the message pointer member variable to NULL and initializes the context object to point at the given context value.*
- void [setMsgBuf](#) (OSRTMessageBufferIF &msgBuf)  
*The setMsgBuf method is used to set the internal message buffer pointer to point at the given message buffer or stream object.*

## Protected Attributes

- OSRTCtxtPtr [mpContext](#)  
*The mpContext member variable holds a reference-counted C runtime variable.*
- OSRTMessageBufferIF \* [mpMsgBuf](#)  
*The mpMsgBuf member variable is a pointer to a derived message buffer or stream class that will manage the message being encoded or decoded.*

### 5.19.1 Detailed Description

XSD global element base class.

This is the main base class for all generated global element control classes. It holds a variable of a generated data type as well as the associated message buffer or stream class to which a message will be encoded or from which a message will be decoded.

Definition at line 57 of file rtXmlCppXSDElement.h.

## 5.19.2 Constructor & Destructor Documentation

### 5.19.2.1 OSXSDGlobalElement::OSXSDGlobalElement (OSRTContext & *ctxt*) [inline, protected]

This constructor sets the message pointer member variable to NULL and initializes the context object to point at the given context value.

#### Parameters:

*ctxt* - Reference to a context object.

Definition at line 85 of file rtXmlCppXSDElement.h.

### 5.19.2.2 OSXSDGlobalElement::OSXSDGlobalElement (OSRTMessageBufferIF & *msgBuf*) [inline]

This constructor sets the internal message buffer pointer to point at the given message buffer or stream object.

The context is set to point at the context contained within the message buffer object. Thus, the message buffer and control class object share the context. It will not be released until both objects are destroyed.

#### Parameters:

*msgBuf* - Reference to a message buffer or stream object.

Definition at line 105 of file rtXmlCppXSDElement.h.

### 5.19.2.3 OSXSDGlobalElement::OSXSDGlobalElement (const OSXSDGlobalElement & *o*) [inline]

The copy constructor sets the internal message buffer pointer and context to point at the message buffer and context from the original OSCType object.

#### Parameters:

*o* - Reference to a global element object.

Definition at line 116 of file rtXmlCppXSDElement.h.

### 5.19.2.4 virtual OSXSDGlobalElement::~OSXSDGlobalElement () [inline, virtual]

The virtual destructor does nothing.

It is overridden by derived versions of this class.

Definition at line 123 of file rtXmlCppXSDElement.h.

## 5.19.3 Member Function Documentation

### 5.19.3.1 void OSXSDGlobalElement::setMsgBuf (OSRTMessageBufferIF & *msgBuf*) [protected]

The setMsgBuf method is used to set the internal message buffer pointer to point at the given message buffer or stream object.

#### Parameters:

*msgBuf* - Reference to a message buffer or stream object.

**5.19.3.2** `virtual int OSXSDGlobalElement::decodeFrom (OSRTMessageBufferIF &) [inline, virtual]`

The `decodeFrom` method decodes a message from the given message buffer or stream argument.

**Parameters:**

- Message buffer or stream containing message to decode.

Definition at line 138 of file `rtXmlCppXSDElement.h`.

**5.19.3.3** `virtual int OSXSDGlobalElement::encodeTo (OSRTMessageBufferIF &) [inline, virtual]`

The `encodeTo` method encodes a message into the given message buffer or stream argument.

**Parameters:**

- Message buffer or stream to which the message is to be encoded.

Definition at line 153 of file `rtXmlCppXSDElement.h`.

**5.19.3.4** `OSCTXT* OSXSDGlobalElement::getCtxtPtr () [inline]`

The `getCtxtPtr` method returns the underlying C runtime context.

This context can be used in calls to C runtime functions.

Definition at line 159 of file `rtXmlCppXSDElement.h`.

**5.19.3.5** `void* OSXSDGlobalElement::memAlloc (size_t numocts) [inline]`

The `memAlloc` method allocates memory using the C runtime memory management functions.

The memory is tracked in the underlying context structure. When both this `OSXSDGlobalElement` derived control class object and the message buffer object are destroyed, this memory will be freed.

**Parameters:**

- numocts* - Number of bytes of memory to allocate

Definition at line 172 of file `rtXmlCppXSDElement.h`.

**5.19.3.6** `void OSXSDGlobalElement::memFreePtr (void * ptr) [inline]`

The `memFreePtr` method frees the memory at a specific location.

This memory must have been allocated using the `memAlloc` method described earlier.

**Parameters:**

- ptr* - Pointer to a block of memory allocated with `memAlloc`

Definition at line 200 of file `rtXmlCppXSDElement.h`.

### 5.19.3.7 void OSXSDGlobalElement::setDefaultNamespace (const OSUTF8CHAR \* *uri*) [inline]

The setDefaultNamespace method sets the default namespace for the element to the given value.

#### Parameters:

*uri* - Default namespace URI

Definition at line 210 of file rtXmlCxxXSDElement.h.

### 5.19.3.8 void OSXSDGlobalElement::setDiag (OSBOOL *value* = TRUE) [inline]

The setDiag method turns diagnostic tracing on or off.

#### Parameters:

*value* - Boolean on/off value (default = on)

Definition at line 219 of file rtXmlCxxXSDElement.h.

### 5.19.3.9 void OSXSDGlobalElement::setEncXSINamespace (OSBOOL *value* = TRUE) [inline]

The setEncXSINamespace method sets a flag in the internal context that indicates the xsi namespace attribute must be encoded.

#### Parameters:

*value* - Boolean on/off value (default = on)

Definition at line 229 of file rtXmlCxxXSDElement.h.

### 5.19.3.10 void OSXSDGlobalElement::setNamespace (const OSUTF8CHAR \* *prefix*, const OSUTF8CHAR \* *uri*) [inline]

The setNamespace method adds or modifies the namespace with the given URI in the namespace list to contain the given prefix.

#### Parameters:

*prefix* - Namespace prefix

*uri* - Namespace URI

Definition at line 240 of file rtXmlCxxXSDElement.h.

### 5.19.3.11 void OSXSDGlobalElement::setNoNSSchemaLocation (const OSUTF8CHAR \* *uri*) [inline]

The setNoNSSchemaLocation method adds an xsi:noNamespaceSchemaLocation attribute to the document.

#### Parameters:

*uri* - URI for noNamespaceSchemaLocation.

Definition at line 250 of file rtXmlCxxXSDElement.h.

### 5.19.3.12 void OSXSDGlobalElement::setSchemaLocation (const OSUTF8CHAR \* *uri*) [inline]

The setSchemaLocation method adds an xsi:schemaLocation attribute to the document.

#### Parameters:

*uri* - URI for schemaLocation.

Definition at line 260 of file rtXmlCppXSDElement.h.

### 5.19.3.13 void OSXSDGlobalElement::setXSIType (const OSUTF8CHAR \* *typeName*) [inline]

The setXSIType method sets a type name to be used in the xsi:type attribute in the top-level module element declaration.

#### Parameters:

*typeName* - XSI type name

Definition at line 270 of file rtXmlCppXSDElement.h.

### 5.19.3.14 virtual int OSXSDGlobalElement::validateFrom (OSRTMessageBufferIF &) [inline, virtual]

The validateFrom method validates a message from the given message buffer or stream argument.

#### Parameters:

- Message buffer or stream containing message to validate.

Definition at line 287 of file rtXmlCppXSDElement.h.

## 5.19.4 Member Data Documentation

### 5.19.4.1 OSRTCtxtPtr OSXSDGlobalElement::mpContext [protected]

The mpContext member variable holds a reference-counted C runtime variable.

This context is used in calls to all C run-time functions. The context pointed at by this smart-pointer object is shared with the message buffer object contained within this class.

Definition at line 65 of file rtXmlCppXSDElement.h.

The documentation for this class was generated from the following file:

- [rtXmlCppXSDElement.h](#)

## Chapter 6

# XBinder File Documentation

### 6.1 rtSaxCppAny.h File Reference

```
#include "rtxsrc/OSRTContext.h"  
#include "rtxmlsrc/osrtxml.h"  
#include "rtxmlsrc/rtSaxCppParser.h"  
#include "rtxmlsrc/rtXmlCppMsgBuf.h"  
#include "rtxsrc/rtxCppXmlString.h"  
#include "rtxmlsrc/OSXSDAnyTypeClass.h"  
#include "rtxmlsrc/rtSaxCppAnyType.h"
```

#### Classes

- class [OSXMLAnyHandler](#)

#### 6.1.1 Detailed Description

Definition in file [rtSaxCppAny.h](#).

## 6.2 rtSaxCppAnyType.h File Reference

```
#include "rtxsrc/OSRTContext.h"  
#include "rtxmlsrc/osrtxml.h"  
#include "rtxmlsrc/rtSaxCppParser.h"  
#include "rtxmlsrc/rtXmlCppMsgBuf.h"  
#include "rtxsrc/rtxCppXmlString.h"  
#include "rtxmlsrc/OSXSDAnyTypeClass.h"
```

### Classes

- class [OSXMLAnyTypeHandler](#)

### 6.2.1 Detailed Description

Definition in file [rtSaxCppAnyType.h](#).

## 6.3 rtSaxCppSimpleType.h File Reference

```
#include "rtxmlsrc/osrtxml.h"  
#include "rtxmlsrc/rtSaxCppParser.h"
```

### Classes

- class [OSXMLSimpleTypeHandler](#)

### 6.3.1 Detailed Description

Definition in file [rtSaxCppSimpleType.h](#).

## 6.4 rtSaxCppSoap.h File Reference

```
#include "rtxsrc/rtxCppDynOctStr.h"  
#include "rtxsrc/OSRTContext.h"  
#include "rtxsrc/OSRTMemBuf.h"  
#include "rtxsrc/rtxCppXmlString.h"  
#include "rtxmlsrc/osrtxml.h"  
#include "rtxmlsrc/rtSaxCppParser.h"  
#include "rtxmlsrc/rtXmlCppMsgBuf.h"
```

### Classes

- class [OSXMLSoapHandler](#)

#### 6.4.1 Detailed Description

Definition in file [rtSaxCppSoap.h](#).

## 6.5 rtSaxCppStrList.h File Reference

```
#include "rtxsrc/rtxToken.h"  
#include "rtxsrc/rtxDList.h"  
#include "rtxmlsrc/osrtxml.h"  
#include "rtxmlsrc/rtSaxCppParser.h"  
#include "rtxsrc/rtxCppDList.h"
```

### Classes

- class [OSXMLStrListHandler](#)  
*OSXMLStrListHandler.*

### 6.5.1 Detailed Description

Definition in file [rtSaxCppStrList.h](#).

## 6.6 rtXmlCppEncFuncs.h File Reference

XML low-level C++ encode functions.

```
#include "rtxmlsrc/osrtxml.h"
```

### Functions

- int [rtXmlCppEncAnyAttr](#) (OSCTXT \*pctxt, OSRTObjListClass \*pAnyAttrList)  
*This function encodes a variable of the XSD any attribute type.*
- int [rtXmlEncAny](#) (OSCTXT \*pctxt, OSXMLStringClass \*pxmlstr, const OSUTF8CHAR \*elemName, OSXMLNamespace \*pNS)  
*This function encodes a variable of the XSD any type.*
- int [rtXmlCppEncAnyTypeValue](#) (OSCTXT \*pctxt, OSXSDAnyTypeClass \*pvalue)  
*This function encodes a variable of the XSD anyType type.*
- int [rtXmlEncString](#) (OSCTXT \*pctxt, OSXMLStringClass \*pxmlstr, const OSUTF8CHAR \*elemName, OSXMLNamespace \*pNS)  
*This function encodes a variable of the XSD string type.*

### 6.6.1 Detailed Description

XML low-level C++ encode functions.

These are overloaded versions of C XML encode functions for use with C++.

Definition in file [rtXmlCppEncFuncs.h](#).

### 6.6.2 Function Documentation

#### 6.6.2.1 int rtXmlCppEncAnyAttr (OSCTXT \* pctxt, OSRTObjListClass \* pAnyAttrList)

This function encodes a variable of the XSD any attribute type.

This is expressed as list of name/value pairs.

#### Parameters:

*pctxt* Pointer to context block structure.

*pAnyAttrList* List of name/value pair objects.

#### Returns:

Completion status of operation:

- 0 = success,
- negative return value is error.

### 6.6.2.2 int rtXmlCppEncAnyTypeValue (OSCTXT \* *pctxt*, OSXSDAnyTypeClass \* *pvalue*)

This function encodes a variable of the XSD anyType type.

This is considered to be a fully-wrapped element of anyType type (for example: \* <myType>myData</myType>)

#### Parameters:

*pctxt* Pointer to context block structure.

*pvalue* Value to be encoded. This is a pointer to a OSXSDAnyTypeClass containing the fully-encoded XML text to be copied to the output stream.

#### Returns:

Completion status of operation:

- 0 = success,
- negative return value is error.

### 6.6.2.3 int rtXmlEncAny (OSCTXT \* *pctxt*, OSXMLStringClass \* *pxmlstr*, const OSUTF8CHAR \* *elemName*, OSXMLNamespace \* *pNS*)

This function encodes a variable of the XSD any type.

This is considered to be a fully-wrapped element of any type (for example: <myType>myData</myType>)

#### Parameters:

*pctxt* Pointer to context block structure.

*pxmlstr* Value to be encoded. This is a string containing the fully-encoded XML text to be copied to the output stream.

*elemName* XML element name. A name must be provided. If an empty string is passed (""), no element tag is added to the encoded value.

*pNS* Pointer to namespace structure.

#### Returns:

Completion status of operation:

- 0 = success,
- negative return value is error.

### 6.6.2.4 int rtXmlEncString (OSCTXT \* *pctxt*, OSXMLStringClass \* *pxmlstr*, const OSUTF8CHAR \* *elemName*, OSXMLNamespace \* *pNS*)

This function encodes a variable of the XSD string type.

#### Parameters:

*pctxt* Pointer to context block structure.

*pxmlstr* XML string value to be encoded.

*elemName* XML element name. A name must be provided. If an empty string is passed (""), no element tag is added to the encoded value.

*pNS* Pointer to namespace structure.

**Returns:**

Completion status of operation:

- 0 = success,
- negative return value is error.

## 6.7 rtXmlCppMsgBuf.h File Reference

This file is deprecated.

```
#include "rtxmlsrc/OSXMLEncodeBuffer.h"  
#include "rtxmlsrc/OSXMLEncodeStream.h"  
#include "rtxmlsrc/OSXMLDecodeBuffer.h"
```

### 6.7.1 Detailed Description

This file is deprecated.

Users should use one or more of the individual headers files defined in the include statements below.

Definition in file [rtXmlCppMsgBuf.h](#).

## 6.8 rtXmlCppNamespace.h File Reference

XML namespace handling structures and function definitions.

```
#include "rtxmlsrc/osrtxml.h"  
#include "rtxsrc/OSRTBaseType.h"  
#include "rtxsrc/OSRTString.h"
```

### Classes

- class [OSXMLNamespaceClass](#)  
*This class is used to hold an XML namespace prefix to URI mapping.*

### 6.8.1 Detailed Description

XML namespace handling structures and function definitions.

Definition in file [rtXmlCppNamespace.h](#).

## 6.9 rtXmlCppXSDElement.h File Reference

C++ run-time XML schema global element class definition.

```
#include "rtxsrc/OSRTContext.h"
#include "rtxsrc/OSRTMsgBufIF.h"
#include "rtxsrc/rtxDiag.h"
#include "rtxsrc/rtxErrCodes.h"
#include "rtxmlsrc/osrtxml.h"
```

### Classes

- class [OSXSDGlobalElement](#)  
*XSD global element base class.*

### 6.9.1 Detailed Description

C++ run-time XML schema global element class definition.

Definition in file [rtXmlCppXSDElement.h](#).

## 6.10 rtXmlpCppDecFuncs.h File Reference

XML low-level C++ decode functions.

```
#include "rtxmlsrc/osrtxml.h"
```

```
#include "rtxmlsrc/OSXSDComplexType.h"
```

### 6.10.1 Detailed Description

XML low-level C++ decode functions.

These are overloaded versions of C XML encode functions for use with C++.

Definition in file [rtXmlpCppDecFuncs.h](#).

# Index

- ~OSXSDGlobalElement
  - [OSXSDGlobalElement](#), [35](#)
- characters
  - [OSXMLContentHandler](#), [11](#)
  - [OSXMLDefaultHandler](#), [14](#)
  - [OSXMLSimpleTypeHandler](#), [29](#)
  - [OSXMLSoapHandler](#), [31](#)
- decodeFrom
  - [OSXSDGlobalElement](#), [35](#)
- encodeTo
  - [OSXSDGlobalElement](#), [36](#)
- endElement
  - [OSXMLAnyHandler](#), [6](#)
  - [OSXMLAnyTypeHandler](#), [8](#)
  - [OSXMLContentHandler](#), [12](#)
  - [OSXMLDefaultHandler](#), [15](#)
  - [OSXMLSimpleTypeHandler](#), [29](#)
  - [OSXMLSoapHandler](#), [31](#)
- error
  - [OSXMLErrorHandler](#), [19](#)
- fatalError
  - [OSXMLErrorHandler](#), [19](#)
- getCtxtPtr
  - [OSXSDGlobalElement](#), [36](#)
- getState
  - [OSXMLDefaultHandler](#), [15](#)
- memAlloc
  - [OSXSDGlobalElement](#), [36](#)
- memFreePtr
  - [OSXSDGlobalElement](#), [36](#)
- mpContext
  - [OSXSDGlobalElement](#), [38](#)
- [OSXMLAnyHandler](#), [5](#)
- [OSXMLAnyHandler](#)
  - [endElement](#), [6](#)
  - [startElement](#), [6](#)
- [OSXMLAnyTypeHandler](#), [7](#)
- [OSXMLAnyTypeHandler](#)
  - [endElement](#), [8](#)
  - [startElement](#), [7](#)
- [OSXMLBase](#), [9](#)
- [OSXMLBasePtr](#), [10](#)
- [OSXMLContentHandler](#), [11](#)
- [OSXMLContentHandler](#)
  - [characters](#), [11](#)
  - [endElement](#), [12](#)
  - [startElement](#), [12](#)
- [OSXMLDefaultHandler](#), [13](#)
- [OSXMLDefaultHandler](#)
  - [characters](#), [14](#)
  - [endElement](#), [15](#)
  - [getState](#), [15](#)
  - [startElement](#), [14](#)
- [OSXMLDefaultHandler::ErrorInfo](#), [16](#)
- [OSXMLDefaultHandlerIF](#), [17](#)
- [OSXMLDefaultHandlerPtr](#), [18](#)
- [OSXMLErrorHandler](#), [19](#)
- [OSXMLErrorHandler](#)
  - [error](#), [19](#)
  - [fatalError](#), [19](#)
  - [resetErrors](#), [20](#)
  - [warning](#), [19](#)
- [OSXMLErrorInfo](#), [21](#)
- [OSXMLNamespaceClass](#), [22](#)
  - [OSXMLNamespaceClass](#), [22](#), [23](#)
- [OSXMLNamespaceClass](#)
  - [OSXMLNamespaceClass](#), [22](#), [23](#)
- [OSXMLParserCtxt](#), [24](#)
- [OSXMLParserCtxtIF](#), [25](#)
- [OSXMLReaderClass](#), [26](#)
- [OSXMLReaderClass](#)
  - [parse](#), [26](#), [27](#)
- [OSXMLSimpleTypeHandler](#), [28](#)
- [OSXMLSimpleTypeHandler](#)
  - [characters](#), [29](#)
  - [endElement](#), [29](#)
  - [startElement](#), [28](#)
- [OSXMLSoapHandler](#), [30](#)
- [OSXMLSoapHandler](#)
  - [characters](#), [31](#)
  - [endElement](#), [31](#)
  - [startElement](#), [30](#)
- [OSXMLStrListHandler](#), [32](#)
- [OSXSDGlobalElement](#), [33](#)

- OSXSDGlobalElement, 35
- OSXSDGlobalElement
  - ~OSXSDGlobalElement, 35
  - decodeFrom, 35
  - encodeTo, 36
  - getCtxtPtr, 36
  - memAlloc, 36
  - memFreePtr, 36
  - mpContext, 38
  - OSXSDGlobalElement, 35
  - setDefaultNamespace, 36
  - setDiag, 37
  - setEncXSINamespace, 37
  - setMsgBuf, 35
  - setNamespace, 37
  - setNoNSSchemaLocation, 37
  - setSchemaLocation, 37
  - setXSIType, 38
  - validateFrom, 38
- parse
  - OSXMLReaderClass, 26, 27
- resetErrors
  - OSXMLErrorHandler, 20
- rtSaxCppAny.h, 39
- rtSaxCppAnyType.h, 40
- rtSaxCppSimpleType.h, 41
- rtSaxCppSoap.h, 42
- rtSaxCppStrList.h, 43
- rtXmlCppEncAnyAttr
  - rtXmlCppEncFuncs.h, 44
- rtXmlCppEncAnyTypeValue
  - rtXmlCppEncFuncs.h, 44
- rtXmlCppEncFuncs.h, 44
- rtXmlCppEncFuncs.h
  - rtXmlCppEncAnyAttr, 44
  - rtXmlCppEncAnyTypeValue, 44
  - rtXmlEncAny, 45
  - rtXmlEncString, 45
- rtXmlCppMsgBuf.h, 47
- rtXmlCppNamespace.h, 48
- rtXmlCppXSDElement.h, 49
- rtXmlEncAny
  - rtXmlCppEncFuncs.h, 45
- rtXmlEncString
  - rtXmlCppEncFuncs.h, 45
- rtXmlpCppDecFuncs.h, 50
- setDefaultNamespace
  - OSXSDGlobalElement, 36
- setDiag
  - OSXSDGlobalElement, 37
- setEncXSINamespace
  - OSXSDGlobalElement, 37
- setMsgBuf
  - OSXSDGlobalElement, 35
- setNamespace
  - OSXSDGlobalElement, 37
- setNoNSSchemaLocation
  - OSXSDGlobalElement, 37
- setSchemaLocation
  - OSXSDGlobalElement, 37
- setXSIType
  - OSXSDGlobalElement, 38
- startElement
  - OSXMLAnyHandler, 6
  - OSXMLAnyTypeHandler, 7
  - OSXMLContentHandler, 12
  - OSXMLDefaultHandler, 14
  - OSXMLSimpleTypeHandler, 28
  - OSXMLSoapHandler, 30
- validateFrom
  - OSXSDGlobalElement, 38
- warning
  - OSXMLErrorHandler, 19