

ASN1C

ASN.1 Compiler
Version 6.1
XML Runtime
Reference Manual

Contents

1	ASN1C Main Page	1
2	ASN1C Module Index	2
2.1	ASN1C Modules	2
3	ASN1C Data Structure Index	3
3.1	ASN1C Data Structures	3
4	ASN1C File Index	4
4.1	ASN1C File List	4
5	ASN1C Module Documentation	5
5.1	ASN.1-XML encode/decode functions.	5
5.2	XML decode functions.	13
5.3	XML encode functions.	29
5.4	XML utility functions.	67
5.5	XML pull-parser decode functions.	68
5.6	XML run-time error status codes.	98
6	ASN1C Data Structure Documentation	101
6.1	OSDecimalFmt Struct Reference	101
6.2	OSDoubleFmt Struct Reference	102
6.3	OSIntegerFmt Struct Reference	103
6.4	OSXMLCtxtInfo Struct Reference	104
6.5	OSXMLElemIDRec Struct Reference	105
6.6	OSXMLFacets Struct Reference	106
6.7	OSXMLGroupDesc Struct Reference	107
6.8	OSXMLItemDescr Struct Reference	108
6.9	OSXMLNameFragments Struct Reference	109
6.10	OSXMLQName Struct Reference	110

6.11 OSXMLSortedAttrOffset Struct Reference	111
6.12 OSXMLStrFragment Struct Reference	112
6.13 OSXSAnyType Struct Reference	113
7 ASN1C File Documentation	114
7.1 osrtxml.h File Reference	114
7.2 rtXmlCppMsgBuf.h File Reference	145
7.3 rtXmlErrCodes.h File Reference	146

Chapter 1

ASN1C Main Page

C XML Runtime Library Functions

The **C run-time XML library** contains functions used to encode/decode XML data. These functions are identified by their *rtXml* prefixes.

The categories of functions provided are as follows:

- XML pull-parser code.
- Functions functions to encode C types to XML.
- Functions to decode XML to C data types.
- Functions to encode XML element tags.
- Functions to encode XML attributes in sorted order for C14N.
- SAX parser interfaces.
- Context management functions.

Chapter 2

ASN1C Module Index

2.1 ASN1C Modules

Here is a list of all modules:

ASN.1-XML encode/decode functions.	5
XML decode functions.	13
XML encode functions.	29
XML utility functions.	67
XML pull-parser decode functions.	68
XML run-time error status codes.	98

Chapter 3

ASN1C Data Structure Index

3.1 ASN1C Data Structures

Here are the data structures with brief descriptions:

OSDecimalFmt	101
OSDoubleFmt	102
OSIntegerFmt	103
OSXMLCtxtInfo	104
OSXMLElemIDRec	105
OSXMLFacets	106
OSXMLGroupDesc	107
OSXMLItemDescr	108
OSXMLNameFragments	109
OSXMLQName	110
OSXMLSortedAttrOffset	111
OSXMLStrFragment	112
OSXSAnyType	113

Chapter 4

ASN1C File Index

4.1 ASN1C File List

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

asn1xml.h	??
osrtxml.h (XML low-level C encode/decode functions)	114
rtXmlCppMsgBuf.h (This file is deprecated)	145
rtXmlErrCodes.h (List of numeric status codes that can be returned by ASN1C run-time functions and generated code)	146

Chapter 5

ASN1C Module Documentation

5.1 ASN.1-XML encode/decode functions.

Functions

- EXTERNXML int [rtAsn1XmlpDecObjId](#) (OSCTXT *pctxt, ASN1OBJID *pvalue)
This function decodes the contents of an ASN.1 OBJECT IDENTIFIER type.
- EXTERNXML int [rtAsn1XmlpDecUnivStr](#) (OSCTXT *pctxt, const OS32BITCHAR **ppdata, OSUINT32 *pnchars)
This function decodes the contents of an ASN.1 UNIVERSAL string type.
- EXTERNXML int [rtAsn1XmlEncGenTime](#) (OSCTXT *pctxt, const char *value, const OSUTF8CHAR *elemName, const OSUTF8CHAR *nsPrefix)
This function encodes a variable of the ASN.1 GeneralizedTime type.
- EXTERNXML int [rtAsn1XmlEncUTCTime](#) (OSCTXT *pctxt, const char *value, const OSUTF8CHAR *elemName, const OSUTF8CHAR *nsPrefix)
This function encodes a variable of the ASN.1 UTCTime type.
- EXTERNXML int [rtAsn1XmlEncObjId](#) (OSCTXT *pctxt, const ASN1OBJID *pvalue, const OSUTF8CHAR *elemName, const OSUTF8CHAR *nsPrefix)
This function encodes a variable of the ASN.1 OBJECT IDENTIFIER type.
- EXTERNXML int [rtAsn1XmlEncRelOID](#) (OSCTXT *pctxt, const ASN1OBJID *pvalue, const OSUTF8CHAR *elemName, const OSUTF8CHAR *nsPrefix)
This function encodes a variable of the ASN.1 RELATIVE-OID type.
- EXTERNXML int [rtAsn1XmlEncOpenType](#) (OSCTXT *pctxt, const OSOCTET *data, OSUINT32 nocts, const OSUTF8CHAR *elemName, const OSUTF8CHAR *nsPrefix)
This function encodes a variable of the ASN.1 open type.
- EXTERNXML int [rtAsn1XmlEncOpenTypeExt](#) (OSCTXT *pctxt, OSRTDList *pElemList)
This function encodes an ASN.1 open type extension.
- EXTERNXML int [rtAsn1XmlEncUnivStr](#) (OSCTXT *pctxt, const OS32BITCHAR *value, OSUINT32 nchars, const OSUTF8CHAR *elemName, const OSUTF8CHAR *nsPrefix)

This function encodes a variable of the ASN.1 UNIVERSAL string type.

- EXTERNXML int [rtAsn1XmlFmtAttrStr](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value, OSUTF8CHAR **pAttrStr)

This function formats a name-value XML pair into a name="value" attribute string.

- EXTERNXML int [rtAsn1XmlParseAttrStr](#) (OSCTXT *pctxt, const OSUTF8CHAR *pAttrStr, OSUTF8NVP *pNVPair)

This function parses an XML name-value pair from an attribute string.

- EXTERNXML int [rtAsn1XmlAddAnyAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value, OSRTDList *plist)

This function formats an attribute string and adds it to the attribute list.

- EXTERNXML int [rtAsn1XmlpDecDynBitStr](#) (OSCTXT *pctxt, ASN1DynBitStr *pvalue)

This function decodes a bit string value.

- EXTERNXML int [rtXmlpDecListOfASN1DynBitStr](#) (OSCTXT *pctxt, OSRTDList *plist)

This function decodes a list of space-separated bit strings and returns each bit string as a separate item on the given list.

- EXTERNXML int [rtAsn1XmlpDecRelOID](#) (OSCTXT *pctxt, ASN1OBJID *pvalue)

This function decodes the contents of an ASN.1 RELATIVE-OID type.

5.1.1 Function Documentation

5.1.1.1 EXTERNXML int [rtAsn1XmlAddAnyAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value, OSRTDList *plist)

This function formats an attribute string and adds it to the attribute list.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

name The name of the new attribute.

value The value of the new attribute.

plist The attribute list.

5.1.1.2 EXTERNXML int [rtAsn1XmlEncGenTime](#) (OSCTXT *pctxt, const char *value, const OSUTF8CHAR *elemName, const OSUTF8CHAR *nsPrefix)

This function encodes a variable of the ASN.1 GeneralizedTime type.

It performs conversion from ASN.1 time format into the XML dateTime format.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

value A pointer to a null-terminated C character string to be encoded. It should contain UTCTime in ASN.1 format.

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

nsPrefix XML namespace prefix. If not null, will be prepended to elemName to form a qualified name.

Returns:

Completion status of operation:

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

5.1.1.3 EXTERNXML int rtAsn1XmlEncObjId (OSCTXT * *pctxt*, const ASN1OBJID * *pvalue*, const OSUTF8CHAR * *elemName*, const OSUTF8CHAR * *nsPrefix*)

This function encodes a variable of the ASN.1 OBJECT IDENTIFIER type.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to an object identifier structure. This structure contains an integer to hold the number of subidentifiers in the object and an array to hold the subidentifier values.

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

nsPrefix XML namespace prefix. If not null, will be prepended to elemName to form a qualified name.

Returns:

Completion status of operation:

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

5.1.1.4 EXTERNXML int rtAsn1XmlEncOpenType (OSCTXT * *pctxt*, const OSOCTET * *data*, OSUINT32 *nocts*, const OSUTF8CHAR * *elemName*, const OSUTF8CHAR * *nsPrefix*)

This function encodes a variable of the ASN.1 open type.

It copies the data as it exists in the structure to the encode buffer or stream.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

data A pointer to a buffer containing the open type data.

nocts Number of bytes in the data buffer to encode.

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

nsPrefix XML namespace prefix. If not null, will be prepended to elemName to form a qualified name.

Returns:

Completion status of operation:

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

5.1.1.5 EXTERNXML int rtAsn1XmlEncOpenTypeExt (OSCTXT * *pctx*, OSRTDList * *pElemList*)

This function encodes an ASN.1 open type extension.

This occurs in a SEQUENCE or SET type when a ... is present. The type is represented as a list of open type structures.

Parameters:

pctx A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pElemList Linked list of ASN.1 open type structures.

Returns:

Completion status of operation:

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

5.1.1.6 EXTERNXML int rtAsn1XmlEncRelOID (OSCTXT * *pctx*, const ASN1OBJID * *pvalue*, const OSUTF8CHAR * *elemName*, const OSUTF8CHAR * *nsPrefix*)

This function encodes a variable of the ASN.1 RELATIVE-OID type.

Parameters:

pctx A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to an object identifier structure. This structure contains an integer to hold the number of subidentifiers in the object and an array to hold the subidentifier values.

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

nsPrefix XML namespace prefix. If not null, will be prepended to elemName to form a qualified name.

Returns:

Completion status of operation:

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

5.1.1.7 EXTERNXML int rtAsn1XmlEncUnivStr (OSCTXT * *pctxt*, const OS32BITCHAR * *value*, OSUINT32 *nchars*, const OSUTF8CHAR * *elemName*, const OSUTF8CHAR * *nsPrefix*)

This function encodes a variable of the ASN.1 UNIVERSAL string type.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

value Array of universal characters to be encoded. Each character is represented as a 32-bit integer.

nchars Number of characters to encode.

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

nsPrefix XML namespace prefix. If not null, will be prepended to *elemName* to form a qualified name.

Returns:

Completion status of operation:

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

5.1.1.8 EXTERNXML int rtAsn1XmlEncUTCTime (OSCTXT * *pctxt*, const char * *value*, const OSUTF8CHAR * *elemName*, const OSUTF8CHAR * *nsPrefix*)

This function encodes a variable of the ASN.1 UTCTime type.

It performs conversion from ASN.1 time format into the XML dateTime format.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

value A pointer to a null-terminated C character string to be encoded. It should contain UTCTime in ASN.1 format.

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

nsPrefix XML namespace prefix. If not null, will be prepended to *elemName* to form a qualified name.

Returns:

Completion status of operation:

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

5.1.1.9 EXTERNXML int rtAsn1XmlFmtAttrStr (OSCTXT * *pctxt*, const OSUTF8CHAR * *name*, const OSUTF8CHAR * *value*, OSUTF8CHAR ** *pAttrStr*)

This function formats a name-value XML pair into a name="value" attribute string.

Parameters:

- pctxt* A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
- name* A pointer to an XML element name. A name must be provided.
- value* A pointer to the corresponding element value.
- pAttrStr* The resulting name="value" string.

Returns:

Completion status of operation:

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

5.1.1.10 EXTERNXML int rtAsn1XmlParseAttrStr (OSCTXT * *pctxt*, const OSUTF8CHAR * *pAttrStr*, OSUTF8NVP * *pNVPair*)

This function parses an XML name-value pair from an attribute string.

Parameters:

- pctxt* A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
- pAttrStr* The name-value string to be parsed.
- pNVPair* A pointer to an XML name-value pair structure filled in by invoking this method.

Returns:

Completion status of operation:

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

5.1.1.11 EXTERNXML int rtAsn1XmlpDecDynBitStr (OSCTXT * *pctxt*, ASN1DynBitStr * *pvalue*)

This function decodes a bit string value.

The string consists of a series of '1' and '0' characters. This is the dynamic version in which memory is allocated for the returned binary string variable. Bits are stored from MSB to LSB order.

Parameters:

- pctxt* Pointer to context block structure.
- pvalue* Pointer to a variable to receive the decoded boolean value.

Returns:

Completion status of operation:

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

5.1.1.12 EXTERNXML int rtAsn1XmlpDecObjId (OSCTXT * *pctxt*, ASN1OBJID * *pvalue*)

This function decodes the contents of an ASN.1 OBJECT IDENTIFIER type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to ASN.1 object identifier value to receive decoded result.

Returns:

Completion status of operation:

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

5.1.1.13 EXTERNXML int rtAsn1XmlpDecRelOID (OSCTXT * *pctxt*, ASN1OBJID * *pvalue*)

This function decodes the contents of an ASN.1 RELATIVE-OID type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to ASN.1 object identifier value to receive decoded result.

Returns:

Completion status of operation:

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

5.1.1.14 EXTERNXML int rtAsn1XmlpDecUnivStr (OSCTXT * *pctxt*, const OS32BITCHAR ** *ppdata*, OSUINT32 * *pnchars*)

This function decodes the contents of an ASN.1 UNIVERSAL string type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

ppdata Pointer to 32-bit character string value to receive decoded result.

pnchars Pointer to length value to receive decoded length in characters.

Returns:

Completion status of operation:

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

5.1.1.15 EXTERNXML int rtXmlpDecListOfASN1DynBitStr (OSCTXT * *pctxt*, OSRTDList * *plist*)

This function decodes a list of space-separated bit strings and returns each bit string as a separate item on the given list.

The string consists of a series of '1' and '0' characters. Memory is allocated for the list nodes and token values using the rtx memory management functions. Bits are stored from MSB to LSB order.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

plist A pointer to a linked list structure to which the parsed bit string values will be added.

Returns:

Completion status of operation:

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

5.2 XML decode functions.

Functions

- EXTERNXML int [rtXmlDecBase64Binary](#) (OSRTMEMBUF *pMemBuf, const OSUTF8CHAR *inpdata, int length)
This function decodes the contents of a Base64-encoded binary data type into a memory buffer.
- EXTERNXML int [rtXmlDecBase64Str](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)
This function decodes a contents of a Base64-encode binary string into a static memory structure.
- EXTERNXML int [rtXmlDecBase64StrValue](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, size_t bufSize, size_t srcDataLen)
This function decodes a contents of a Base64-encode binary string into the specified octet array.
- EXTERNXML int [rtXmlDecBigInt](#) (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)
This function will decode a variable of the XSD integer type.
- EXTERNXML int [rtXmlDecBool](#) (OSCTXT *pctxt, OSBOOL *pvalue)
This function decodes a variable of the boolean type.
- EXTERNXML int [rtXmlDecDate](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'date' type.
- EXTERNXML int [rtXmlDecTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'time' type.
- EXTERNXML int [rtXmlDecDateTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'dateTime' type.
- EXTERNXML int [rtXmlDecDecimal](#) (OSCTXT *pctxt, OSREAL *pvalue)
This function decodes the contents of a decimal data type.
- EXTERNXML int [rtXmlDecDouble](#) (OSCTXT *pctxt, OSREAL *pvalue)
This function decodes the contents of a float or double data type.
- EXTERNXML int [rtXmlDecDynBase64Str](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a contents of a Base64-encode binary string.
- EXTERNXML int [rtXmlDecDynHexStr](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a contents of a hexBinary string.
- EXTERNXML int [rtXmlDecUTF8Str](#) (OSCTXT *pctxt, OSUTF8CHAR *outdata, size_t max_len)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlDecDynUTF8Str](#) (OSCTXT *pctxt, const OSUTF8CHAR **outdata)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlDecHexBinary](#) (OSRTMEMBUF *pMemBuf, const OSUTF8CHAR *inpdata, int length)

This function decodes the contents of a hex-encoded binary data type into a memory buffer.

- EXTERNXML int [rtXmlDecHexStr](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocets, OSINT32 bufsize)

This function decodes the contents of a hexBinary string into a static memory structure.

- EXTERNXML int [rtXmlDecHexStrValue](#) (OSCTXT *pctxt, const OSUTF8CHAR *const inpdata, size_t nbytes, OSOCTET *pvalue, OSUINT32 *pnbits, OSINT32 bufsize)
- EXTERNXML int [rtXmlDecGYear](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gYear' type.

- EXTERNXML int [rtXmlDecGYearMonth](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gYearMonth' type.

- EXTERNXML int [rtXmlDecGMonth](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gMonth' type.

- EXTERNXML int [rtXmlDecGMonthDay](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gMonthDay' type.

- EXTERNXML int [rtXmlDecGDay](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gDay' type.

- EXTERNXML int [rtXmlDecInt](#) (OSCTXT *pctxt, OSINT32 *pvalue)

This function decodes the contents of a 32-bit integer data type.

- EXTERNXML int [rtXmlDecInt8](#) (OSCTXT *pctxt, OSINT8 *pvalue)

This function decodes the contents of an 8-bit integer data type (i.e.

- EXTERNXML int [rtXmlDecInt16](#) (OSCTXT *pctxt, OSINT16 *pvalue)

This function decodes the contents of a 16-bit integer data type.

- EXTERNXML int [rtXmlDecInt64](#) (OSCTXT *pctxt, OSINT64 *pvalue)

This function decodes the contents of a 64-bit integer data type.

- EXTERNXML int [rtXmlDecUInt](#) (OSCTXT *pctxt, OSUINT32 *pvalue)

This function decodes the contents of an unsigned 32-bit integer data type.

- EXTERNXML int [rtXmlDecUInt8](#) (OSCTXT *pctxt, OSUINT8 *pvalue)

This function decodes the contents of an unsigned 8-bit integer data type (i.e.

- EXTERNXML int [rtXmlDecUInt16](#) (OSCTXT *pctxt, OSUINT16 *pvalue)

This function decodes the contents of an unsigned 16-bit integer data type.

- EXTERNXML int [rtXmlDecUInt64](#) (OSCTXT *pctxt, OSUINT64 *pvalue)

This function decodes the contents of an unsigned 64-bit integer data type.

- EXTERNXML int [rtXmlDecNSAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *attrName, const OSUTF8CHAR *attrValue, OSRTDList *pNSAttrs, const OSUTF8CHAR *nsTable[], OSUINT32 nsTableRowCount)

This function decodes an XML namespace attribute (xmlns).

- EXTERNXML const OSUTF8CHAR * [rtXmlDecQName](#) (OSCTXT *pctxt, const OSUTF8CHAR *qname, const OSUTF8CHAR **prefix)
This function decodes an XML qualified name string (QName) type.
- EXTERNXML int [rtXmlDecXSIAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *attrName, const OSUTF8CHAR *attrValue)
This function decodes XML schema instance (XSI) attribute.
- EXTERNXML int [rtXmlDecXSIAttrs](#) (OSCTXT *pctxt, const OSUTF8CHAR *const *attrs, const char *typeName)
This function decodes XML schema instance (XSI) attributes.
- EXTERNXML int [rtXmlDecXmlStr](#) (OSCTXT *pctxt, OSXMLSTRING *outdata)
This function decodes the contents of an XML string data type.
- EXTERNXML int [rtXmlParseElementName](#) (OSCTXT *pctxt, OSUTF8CHAR **ppName)
This function parses the initial tag from an XML message.
- EXTERNXML int [rtXmlParseElemQName](#) (OSCTXT *pctxt, OSXMLQName *pQName)
This function parses the initial tag from an XML message.

5.2.1 Function Documentation

5.2.1.1 EXTERNXML int [rtXmlDecBase64Binary](#) (OSRTMEMBUF * *pMemBuf*, const OSUTF8CHAR * *inpdata*, int *length*)

This function decodes the contents of a Base64-encoded binary data type into a memory buffer.

Input is expected to be a string of UTF-8 characters returned by an XML parser. The decoded data will be put into the memory buffer starting from the current position and bit offset. After all data is decoded the octet string may be fetched out.

This function is normally used in the 'characters' SAX handler.

Parameters:

pMemBuf Memory buffer to which decoded binary data is to be appended.

inpdata Pointer to a source string to be decoded.

length Length of the source string (in characters).

Returns:

Completion status of operation:

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

5.2.1.2 EXTERNXML int [rtXmlDecBase64Str](#) (OSCTXT * *pctxt*, OSOCTET * *pvalue*, OSUINT32 * *pnocfs*, OSINT32 *bufsize*)

This function decodes a contents of a Base64-encode binary string into a static memory structure.

The octet string must be Base64 encoded. This function call is used to decode a sized base64Binary string production.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the bufsize input parameter.

pnocts A pointer to an integer value to receive the decoded number of octets.

bufsize The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns:

Completion status of operation:

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

5.2.1.3 EXTERNXML int rtXmlDecBase64StrValue (OSCTXT * *pctxt*, OSOCTET * *pvalue*, OSUINT32 * *pnocts*, size_t *bufSize*, size_t *srcDataLen*)

This function decodes a contents of a Base64-encode binary string into the specified octet array.

The octet string must be Base64 encoded. This function call is used internally to decode both sized and non-sized base64binary string production.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the bufsize input parameter.

pnocts A pointer to an integer value to receive the decoded number of octets.

bufSize A maximum size (in octets) of *pvalue* buffer. An error will occur if the number of octets in the decoded string is larger than this value.

srcDataLen An actual source data length (in octets) without whitespaces.

Returns:

Completion status of operation:

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

5.2.1.4 EXTERNXML int rtXmlDecBigInt (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function will decode a variable of the XSD integer type.

In this case, the integer is assumed to be of a larger size than can fit in a C or C++ long type (normally 32 or 64 bits). For example, parameters used to calculate security values are typically larger than these sizes.

These variables are stored in character string constant variables. The radix should be 10. If it is necessary to convert to another radix, then use *rtxBigIntSetStr* or *rtxBigIntToString* functions.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to a pointer to receive decoded UTF-8 string. Dynamic memory is allocated for the variable using the `rtMemAlloc` function. The decoded variable is represented as a string starting with appropriate prefix.

Returns:

Completion status of operation:

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

5.2.1.5 EXTERNXML int rtXmlDecBool (OSCTXT * *pctxt*, OSBOOL * *pvalue*)

This function decodes a variable of the boolean type.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to a variable to receive the decoded boolean value.

Returns:

Completion status of operation:

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

5.2.1.6 EXTERNXML int rtXmlDecDate (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'date' type.

Input is expected to be a string of characters returned by an XML parser. The string should have CCYY-MM-DD format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.7 EXTERNXML int rtXmlDecDateTime (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'dateTime' type.

Input is expected to be a string of characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.8 EXTERNXML int rtXmlDecDecimal (OSCTXT * *pctxt*, OSREAL * *pvalue*)

This function decodes the contents of a decimal data type.

Input is expected to be a string of characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 64-bit double value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.9 EXTERNXML int rtXmlDecDouble (OSCTXT * *pctxt*, OSREAL * *pvalue*)

This function decodes the contents of a float or double data type.

Input is expected to be a string of characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 64-bit double value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.10 EXTERNXML int rtXmlDecDynBase64Str (OSCTXT * *pctxt*, OSDynOctStr * *pvalue*)

This function decodes a contents of a Base64-encode binary string.

The octet string must be Base64 encoded. This function will allocate dynamic memory to store the decoded result.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated to hold the string using the `rtxMemAlloc` function.

Returns:

Completion status of operation:

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

5.2.1.11 EXTERNXML int rtXmlDecDynHexStr (OSCTXT * *pctxt*, OSDynOctStr * *pvalue*)

This function decodes a contents of a hexBinary string.

This function will allocate dynamic memory to store the decoded result.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated to hold the string using the `rtxMemAlloc` function.

Returns:

Completion status of operation:

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

5.2.1.12 EXTERNXML int rtXmlDecDynUTF8Str (OSCTXT * *pctxt*, const OSUTF8CHAR ** *outdata*)

This function decodes the contents of a UTF-8 string data type.

Input is expected to be a string of UTF-8 or Unicode characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

outdata Pointer to a pointer to receive decoded UTF-8 string. Memory is allocated for this string using the run-time memory manager.

Returns:

Completion status of operation:

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

5.2.1.13 EXTERNXML int rtXmlDecGDay (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gDay' type.

Input is expected to be a string of characters returned by an XML parser. The string should have —DD[+hh:mm|Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.14 EXTERNXML int rtXmlDecGMonth (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gMonth' type.

Input is expected to be a string of characters returned by an XML parser. The string should have –MM[+hh:mm|Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.15 EXTERNXML int rtXmlDecGMonthDay (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gMonthDay' type.

Input is expected to be a string of characters returned by an XML parser. The string should have –MM-DD[+hh:mm|Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.16 EXTERNXML int rtXmlDecGYear (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gYear' type.

Input is expected to be a string of characters returned by an XML parser. The string should have CCYY[-+hh:mm|Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.17 EXTERNXML int rtXmlDecGYearMonth (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gYearMonth' type.

Input is expected to be a string of characters returned by an XML parser. The string should have CCYY-MM[-+hh:mm|Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.18 EXTERNXML int rtXmlDecHexBinary (OSRTMEMBUF * *pMemBuf*, const OSUTF8CHAR * *inpdata*, int *length*)

This function decodes the contents of a hex-encoded binary data type into a memory buffer.

Input is expected to be a string of UTF-8 characters returned by an XML parser. The decoded data will be put into the given memory buffer starting from the current position and bit offset. After all data is decoded the octet string may be fetched out.

This function is normally used in the 'characters' SAX handler.

Parameters:

pMemBuf Pointer to memory buffer onto which the decoded binary data will be appended.

inpdata Pointer to a source string to be decoded.

length Length of the source string (in characters).

Returns:

Completion status of operation:

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

5.2.1.19 EXTERNXML int rtXmlDecHexStr (OSCTXT * *pctxt*, OSOCTET * *pvalue*, OSUINT32 * *pnocts*, OSINT32 *bufsize*)

This function decodes the contents of a hexBinary string into a static memory structure.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the *bufsize* input parameter.

pnocts A pointer to an integer value to receive the decoded number of octets.

bufsize The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns:

Completion status of operation:

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

5.2.1.20 EXTERNXML int rtXmlDecInt (OSCTXT * *pctxt*, OSINT32 * *pvalue*)

This function decodes the contents of a 32-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 32-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.21 EXTERNXML int rtXmlDecInt16 (OSCTXT * *pctxt*, OSINT16 * *pvalue*)

This function decodes the contents of a 16-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 16-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.22 EXTERNXML int rtXmlDecInt64 (OSCTXT * *pctxt*, OSINT64 * *pvalue*)

This function decodes the contents of a 64-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 64-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.23 EXTERNXML int rtXmlDecInt8 (OSCTXT * *pctxt*, OSINT8 * *pvalue*)

This function decodes the contents of an 8-bit integer data type (i.e.

a signed byte type in the range of -128 to 127). Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 8-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.24 EXTERNXML int rtXmlDecNSAttr (OSCTXT * *pctxt*, const OSUTF8CHAR * *attrName*, const OSUTF8CHAR * *attrValue*, OSRTDList * *pNSAttrs*, const OSUTF8CHAR * *nsTable*[], OSUINT32 *nsTableRowCount*)

This function decodes an XML namespace attribute (xmlns).

It is assumed that the given attribute name is either 'xmlns' or 'xmlns:prefix'. The XML namespace prefix and URI are added to the given attribute list structure. The parsed namespace is also added to the namespace stack in the given context.

Parameters:

pctxt Pointer to context structure.

attrName Name of the XML namespace attribute. This is assumed to contain either 'xmlns' (a default namespace declaration) or 'xmlns:prefix' where prefix is a namespace prefix.

attrValue XML namespace attribute value (a URI).

pNSAttrs List to receive parsed namespace values.

nsTable Namespace URI's parsed from schema.

nsTableRowCount Number of rows (URI's) in namespace table.

Returns:

Zero if success or negative error code.

5.2.1.25 EXTERNXML const OSUTF8CHAR* rtXmlDecQName (OSCTXT * *pctxt*, const OSUTF8CHAR * *qname*, const OSUTF8CHAR ** *prefix*)

This function decodes an XML qualified name string (QName) type.

This is a type that contains an optional namespace prefix followed by a colon and the local part of the name:

[NS 5] QName ::= (Prefix ':')? LocalPart

[NS 6] Prefix ::= NCName

[NS 7] LocalPart ::= NCName

Parameters:

pctxt Pointer to context block structure.

qname String containing XML QName to be decoded.

prefix Pointer to string pointer to receive decoded prefix. This is optional. If NULL, the prefix will not be returned. If not-NULL, the prefix will be returned in memory allocated using `rtxMemAlloc` which must be freed using one of the `rtxMemFree` functions.

Returns:

Pointer to local part of string. This is pointer to a location within the given string (i.e. a pointer to the character after the ':' or to the beginning of the string if it contains no colon). This pointer does not need to be freed.

5.2.1.26 EXTERNXML int rtXmlDecTime (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'time' type.

Input is expected to be a string of characters returned by an XML parser. The string should have one of following formats:

(1) hh-mm-ss.ss used if tz_flag = false (2) hh-mm-ss.ssZ used if tz_flag = false and tzo = 0 (3) hh-mm-ss.ss+HH:MM if tz_flag = false and tzo > 0 (4) hh-mm-ss.ss-HH:MM-HH:MM if tz_flag = false and tzo < 0

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.27 EXTERNXML int rtXmlDecUInt (OSCTXT * *pctxt*, OSUINT32 * *pvalue*)

This function decodes the contents of an unsigned 32-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned 32-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.28 EXTERNXML int rtXmlDecUInt16 (OSCTXT * *pctxt*, OSUINT16 * *pvalue*)

This function decodes the contents of an unsigned 16-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned 16-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.29 EXTERNXML int rtXmlDecUInt64 (OSCTXT * *pctxt*, OSUINT64 * *pvalue*)

This function decodes the contents of an unsigned 64-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned 64-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.30 EXTERNXML int rtXmlDecUInt8 (OSCTXT * *pctxt*, OSUINT8 * *pvalue*)

This function decodes the contents of an unsigned 8-bit integer data type (i.e.

a signed byte type in the range of 0 to 255). Input is expected to be a string of OSUTF8CHAR characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned 8-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.2.1.31 EXTERNXML int rtXmlDecUTF8Str (OSCTXT * *pctxt*, OSUTF8CHAR * *outdata*, size_t *max_len*)

This function decodes the contents of a UTF-8 string data type.

Input is expected to be a string of UTF-8 or Unicode characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

outdata Pointer to a block of memory to receive decoded UTF8 string.

max_len Size of memory block.

Returns:

Completion status of operation:

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

5.2.1.32 EXTERNXML int rtXmlDecXmlStr (OSCTXT * *pctxt*, OSXMLSTRING * *outdata*)

This function decodes the contents of an XML string data type.

This type contains a pointer to a UTF-8 character string plus flags that can be set to alter the encoding of the string (for example, the *cdata* flag allows the string to be encoded in a CDATA section). Input is expected to be a string of UTF-8 characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

outdata Pointer to an XML string structure to receive the decoded string. Memory is allocated for the string using the run-time memory manager.

Returns:

Completion status of operation:

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

5.2.1.33 EXTERNXML int rtXmlDecXSIAttr (OSCTXT * *pctxt*, const OSUTF8CHAR * *attrName*, const OSUTF8CHAR * *attrValue*)

This function decodes XML schema instance (XSI) attribute.

These attributes include the XSI namespace declaration, the XSD schema location attribute, and the XSD no namespace schema location attribute.

Parameters:

pctxt Pointer to context block structure.

attrName Attribute's name to be decoded

attrValue Attribute's value to be decoded

Returns:

Completion status of operation:

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

5.2.1.34 EXTERNXML int rtXmlDecXSIAttrs (OSCTXT * *pctxt*, const OSUTF8CHAR * *const* * *attrs*, const char * *typeName*)

This function decodes XML schema instance (XSI) attributes.

These attributes include the XSI namespace declaration, the XSD schema location attribute, and the XSD no namespace schema location attribute.

Parameters:

pctxt Pointer to context block structure.

attrs Attributes-values array [attr, value]. Should be null-terminated.

typeName Name of parent type to add in error log, if would be necessary.

Returns:

Completion status of operation:

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

5.2.1.35 EXTERNXML int rtXmlParseElementName (OSCTXT * *pctxt*, OSUTF8CHAR ** *ppName*)

This function parses the initial tag from an XML message.

If the tag is a QName, only the local part of the name is returned.

Parameters:

pctxt Pointer to OSCTXT structure

ppName Pointer to a pointer to receive decoded UTF-8 string. Dynamic memory is allocated for the variable using the rtxMemAlloc function.

Returns:

Completion status of operation:

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

5.2.1.36 EXTERNXML int rtXmlParseElemQName (OSCTXT * *pctxt*, OSXMLQName * *pQName*)

This function parses the initial tag from an XML message.

Parameters:

pctxt Pointer to OSCTXT structure

pQName Pointer to a QName structure to receive parsed name prefix and local name. Dynamic memory is allocated for both name parts using the rtxMemAlloc function.

Returns:

Completion status of operation:

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

5.3 XML encode functions.

Defines

- #define `rtXmlFinalizeMemBuf`(pMemBuf)
- #define `rtXmlGetEncBufPtr`(pctx) (pctx) → buffer.data
This macro returns the start address of the encoded XML message.
- #define `rtXmlGetEncBufLen`(pctx) (pctx) → buffer.byteIndex
This macro returns the length of the encoded XML message.

Functions

- EXTERNXML int `rtXmlEncAny` (OSCTXT *pctx, OSXMLSTRING *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD any type.
- EXTERNXML int `rtXmlEncAnyStr` (OSCTXT *pctx, const OSUTF8CHAR *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
- EXTERNXML int `rtXmlEncAnyTypeValue` (OSCTXT *pctx, const OSUTF8CHAR *pvalue)
This function encodes a variable of the XSD anyType type.
- EXTERNXML int `rtXmlEncAnyAttr` (OSCTXT *pctx, OSRTDList *pAnyAttrList)
This function encodes a list of OSAnyAttr attributes in which the name and value are given as a UTF-8 string.
- EXTERNXML int `rtXmlEncBase64Binary` (OSCTXT *pctx, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD base64Binary type.
- EXTERNXML int `rtXmlEncBase64BinaryAttr` (OSCTXT *pctx, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD base64Binary type as an attribute.
- EXTERNXML int `rtXmlEncBase64StringValue` (OSCTXT *pctx, OSUINT32 nocts, const OSOCTET *value)
This function encodes a variable of the XSD base64Binary type.
- EXTERNXML int `rtXmlEncBigInt` (OSCTXT *pctx, const OSUTF8CHAR *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int `rtXmlEncBigIntAttr` (OSCTXT *pctx, const OSUTF8CHAR *value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes an XSD integer attribute value.
- EXTERNXML int `rtXmlEncBigIntValue` (OSCTXT *pctx, const OSUTF8CHAR *value)
This function encodes an XSD integer attribute value.
- EXTERNXML int `rtXmlEncBitString` (OSCTXT *pctx, OSUINT32 nbits, const OSOCTET *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a variable of the ASN.1 BIT STRING type.

- EXTERNXML int [rtXmlEncBinStrValue](#) (OSCTXT *pctxt, OSUINT32 nbits, const OSOCTET *data)
This function encodes a binary string value as a sequence of '1's and '0's.
- EXTERNXML int [rtXmlEncBool](#) (OSCTXT *pctxt, OSBOOL value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD boolean type.
- EXTERNXML int [rtXmlEncBoolValue](#) (OSCTXT *pctxt, OSBOOL value)
This function encodes a variable of the XSD boolean type.
- EXTERNXML int [rtXmlEncBoolAttr](#) (OSCTXT *pctxt, OSBOOL value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes an XSD boolean attribute value.
- EXTERNXML int [rtXmlEncDate](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD 'date' type as a string.
- EXTERNXML int [rtXmlEncDateValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a variable of the XSD 'date' type as a string.
- EXTERNXML int [rtXmlEncTime](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD 'time' type as a string.
- EXTERNXML int [rtXmlEncTimeValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a variable of the XSD 'time' type as a string.
- EXTERNXML int [rtXmlEncDateTime](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a numeric date/time value into an XML string representation.
- EXTERNXML int [rtXmlEncDateTimeValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a numeric date/time value into an XML string representation.
- EXTERNXML int [rtXmlEncDecimal](#) (OSCTXT *pctxt, OSREAL value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const [OSDecimalFmt](#) *pFmtSpec)
This function encodes a variable of the XSD decimal type.
- EXTERNXML int [rtXmlEncDecimalAttr](#) (OSCTXT *pctxt, OSREAL value, const OSUTF8CHAR *attrName, size_t attrNameLen, const [OSDecimalFmt](#) *pFmtSpec)
This function encodes a variable of the XSD decimal type as an attribute.
- EXTERNXML int [rtXmlEncDecimalValue](#) (OSCTXT *pctxt, OSREAL value, const [OSDecimalFmt](#) *pFmtSpec, char *pDestBuf, size_t destBufSize)
This function encodes a value of the XSD decimal type.
- EXTERNXML int [rtXmlEncDouble](#) (OSCTXT *pctxt, OSREAL value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const [OSDoubleFmt](#) *pFmtSpec)

This function encodes a variable of the XSD double type.

- EXTERNXML int [rtXmlEncDoubleAttr](#) (OSCTXT *pctxt, OSREAL value, const OSUTF8CHAR *attrName, size_t attrNameLen, const [OSDoubleFmt](#) *pFmtSpec)

This function encodes a variable of the XSD double type as an attribute.

- EXTERNXML int [rtXmlEncDoubleValue](#) (OSCTXT *pctxt, OSREAL value, const [OSDoubleFmt](#) *pFmtSpec, int defaultPrecision)

This function encodes a value of the XSD double or float type.

- EXTERNXML int [rtXmlEncEmptyElement](#) (OSCTXT *pctxt, const OSUTF8CHAR *elemName, OSXML-
Namespace *pNS, OSRTDList *pNSAttrs, OSBOOL terminate)

This function encodes an empty element tag value (<elemName/>).

- EXTERNXML int [rtXmlEncEndDocument](#) (OSCTXT *pctxt)

This function adds trailer information and a null terminator at the end of the XML document being encoded.

- EXTERNXML int [rtXmlEncEndElement](#) (OSCTXT *pctxt, const OSUTF8CHAR *elemName, OSXML-
Namespace *pNS)

This function encodes an end element tag value (</elemName>).

- EXTERNXML int [rtXmlEncEndSoapEnv](#) (OSCTXT *pctxt)

This function encodes a SOAP envelope end element tag (<SOAP-ENV:Envelope/>).

- EXTERNXML int [rtXmlEncEndSoapElems](#) (OSCTXT *pctxt, [OSXMLSOAPMsgType](#) msgtype)

This function encodes SOAP end element tags.

- EXTERNXML int [rtXmlEncFloat](#) (OSCTXT *pctxt, OSREAL value, const OSUTF8CHAR *elemName, OS-
XMLNamespace *pNS, const [OSDoubleFmt](#) *pFmtSpec)

This function encodes a variable of the XSD float type.

- EXTERNXML int [rtXmlEncFloatAttr](#) (OSCTXT *pctxt, OSREAL value, const OSUTF8CHAR *attrName, size_t attrNameLen, const [OSDoubleFmt](#) *pFmtSpec)

This function encodes a variable of the XSD float type as an attribute.

- EXTERNXML int [rtXmlEncGYear](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gYear element into an XML string representation.

- EXTERNXML int [rtXmlEncGYearMonth](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OS-
UTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gYearMonth element into an XML string representation.

- EXTERNXML int [rtXmlEncGMonth](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OS-
UTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gMonth element into an XML string representation.

- EXTERNXML int [rtXmlEncGMonthDay](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OS-
UTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gMonthDay element into an XML string representation.

- EXTERNXML int [rtXmlEncGDay](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a numeric gDay element into an XML string representation.
- EXTERNXML int [rtXmlEncGYearValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a numeric gYear value into an XML string representation.
- EXTERNXML int [rtXmlEncGYearMonthValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a numeric gYearMonth value into an XML string representation.
- EXTERNXML int [rtXmlEncGMonthValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a numeric gMonth value into an XML string representation.
- EXTERNXML int [rtXmlEncGMonthDayValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a numeric gMonthDay value into an XML string representation.
- EXTERNXML int [rtXmlEncGDayValue](#) (OSCTXT *pctxt, const OSXSDDateTime *pvalue)
This function encodes a numeric gDay value into an XML string representation.
- EXTERNXML int [rtXmlEncHexBinary](#) (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD hexBinary type.
- EXTERNXML int [rtXmlEncHexBinaryAttr](#) (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD hexBinary type as an attribute.
- EXTERNXML int [rtXmlEncHexStrValue](#) (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *data)
This function encodes a variable of the XSD hexBinary type.
- EXTERNXML int [rtXmlEncIndent](#) (OSCTXT *pctxt)
This function adds indentation whitespace to the output stream.
- EXTERNXML int [rtXmlEncInt](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncIntValue](#) (OSCTXT *pctxt, OSINT32 value)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncIntAttr](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD integer type as an attribute (name="value").
- EXTERNXML int [rtXmlEncIntPattern](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const OSUTF8CHAR *pattern)
This function encodes a variable of the XSD integer type using a pattern to specify the format of the integer value.
- EXTERNXML int [rtXmlEncIntPatternValue](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *pattern)

- EXTERNXML int [rtXmlEncInt64](#) (OSCTXT *pctxt, OSINT64 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncInt64Value](#) (OSCTXT *pctxt, OSINT64 value)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncInt64Attr](#) (OSCTXT *pctxt, OSINT64 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD integer type as an attribute (name="value").
- EXTERNXML int [rtXmlEncNamedBits](#) (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSUINT32 nbits, const OSOCTET *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the ASN.1 BIT STRING type.
- EXTERNXML int [rtXmlEncNamedBitsValue](#) (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSUINT32 nbits, const OSOCTET *pvalue)
- EXTERNXML int [rtXmlEncNSAttrs](#) (OSCTXT *pctxt, OSRTDList *pNSAttrs)
This function encodes namespace declaration attributes at the beginning of an XML document.
- EXTERNXML int [rtxPrintNSAttrs](#) (const char *name, const OSRTDList data)
This function prints a list of namespace attributes.
- EXTERNXML int [rtXmlEncReal10](#) (OSCTXT *pctxt, const OSUTF8CHAR *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the ASN.1 REAL base 10 type.
- EXTERNXML int [rtXmlEncSoapArrayTypeAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value, size_t itemCount)
This function encodes the special SOAP encoding attrType attribute which specifies the number and type of elements in a SOAP array.
- EXTERNXML int [rtXmlEncSoapArrayTypeAttr2](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const OSUTF8CHAR *value, size_t valueLen, size_t itemCount)
- EXTERNXML int [rtXmlEncStartDocument](#) (OSCTXT *pctxt)
This function encodes the XML header text at the beginning of an XML document.
- EXTERNXML int [rtXmlEncBOM](#) (OSCTXT *pctxt)
This function encodes the Unicode byte order mark header at the start of the document.
- EXTERNXML int [rtXmlEncStartElement](#) (OSCTXT *pctxt, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, OSRTDList *pNSAttrs, OSBOOL terminate)
This function encodes a start element tag value (<elemName>).
- EXTERNXML int [rtXmlEncStartSoapEnv](#) (OSCTXT *pctxt)
This function encodes a SOAP envelope start element tag.
- EXTERNXML int [rtXmlEncStartSoapElems](#) (OSCTXT *pctxt, [OSXMLSOAPMsgType](#) msgtype)
This function encodes a SOAP envelope start element tag and an optional SOAP body or fault tag.

- EXTERNXML int [rtXmlEncString](#) (OSCTXT *pctxt, OSXMLSTRING *pxmlstr, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD string type.
- EXTERNXML int [rtXmlEncStringValue](#) (OSCTXT *pctxt, const OSUTF8CHAR *value)
This function encodes a variable of the XSD string type.
- EXTERNXML int [rtXmlEncStringValue2](#) (OSCTXT *pctxt, const OSUTF8CHAR *value, size_t valueLen)
This function encodes a variable of the XSD string type.
- EXTERNXML int [rtXmlEncTermStartElement](#) (OSCTXT *pctxt)
This function terminates a currently open XML start element by adding either a '>' or '>' (if empty) terminator.
- EXTERNXML int [rtXmlEncUnicodeStr](#) (OSCTXT *pctxt, const OSUNICHAR *value, OSUINT32 nchars, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a Unicode string value.
- EXTERNXML int [rtXmlEncUTF8Attr](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value)
This function encodes an attribute in which the name and value are given as a null-terminated UTF-8 strings.
- EXTERNXML int [rtXmlEncUTF8Attr2](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const OSUTF8CHAR *value, size_t valueLen)
This function encodes an attribute in which the name and value are given as a UTF-8 strings with lengths.
- EXTERNXML int [rtXmlEncUTF8Str](#) (OSCTXT *pctxt, const OSUTF8CHAR *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a UTF-8 string value.
- EXTERNXML int [rtXmlEncUInt](#) (OSCTXT *pctxt, OSUINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD unsigned integer type.
- EXTERNXML int [rtXmlEncUIntValue](#) (OSCTXT *pctxt, OSUINT32 value)
This function encodes a variable of the XSD unsigned integer type.
- EXTERNXML int [rtXmlEncUIntAttr](#) (OSCTXT *pctxt, OSUINT32 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD unsigned integer type as an attribute (name="value").
- EXTERNXML int [rtXmlEncUInt64](#) (OSCTXT *pctxt, OSUINT64 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncUInt64Value](#) (OSCTXT *pctxt, OSUINT64 value)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncUInt64Attr](#) (OSCTXT *pctxt, OSUINT64 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD integer type as an attribute (name="value").

- EXTERNXML int [rtXmlEncXSIAttrs](#) (OSCTXT *pctxt, OSBOOL needXSI)
This function encodes XML schema instance (XSI) attributes at the beginning of an XML document.
- EXTERNXML int [rtXmlEncXSITypeAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *value)
This function encodes an XML schema instance (XSI) type attribute value (xsi:type="value").
- EXTERNXML int [rtXmlFreeInputSource](#) (OSCTXT *pctxt)
This function closes an input source that was previously created with one of the create input source functions such as 'rtXmlCreateFileInputSource'.
- EXTERNXML OSBOOL [rtXmlStrCmpAsc](#) (const OSUTF8CHAR *text1, const char *text2)
- EXTERNXML OSBOOL [rtXmlStrnCmpAsc](#) (const OSUTF8CHAR *text1, const char *text2, size_t len)
- EXTERNXML int [rtXmlSetEncBufPtr](#) (OSCTXT *pctxt, OSOCTET *bufaddr, size_t bufsiz)
This function is used to set the internal buffer within the run-time library encoding context.
- EXTERNXML int [rtXmlGetIndent](#) (OSCTXT *pctxt)
This function returns current XML output indent value.
- EXTERNXML OSBOOL [rtXmlGetWriteBOM](#) (OSCTXT *pctxt)
This function returns whether the Unicode byte order mark will be encoded.
- EXTERNXML int [rtXmlGetIndentChar](#) (OSCTXT *pctxt)
This function returns current XML output indent character value (default is space).

5.3.1 Define Documentation

5.3.1.1 #define rtXmlFinalizeMemBuf(pMemBuf)

Value:

```
do { \
    (pMemBuf)->pctxt->buffer.data = (pMemBuf)->buffer + (pMemBuf)->startidx; \
    (pMemBuf)->pctxt->buffer.size = \
    ((pMemBuf)->usedcnt - (pMemBuf)->startidx); \
    (pMemBuf)->pctxt->buffer.dynamic = FALSE; \
    (pMemBuf)->pctxt->buffer.byteIndex = 0; \
    rtxMemBufReset (pMemBuf); \
} while(0)
```

Definition at line 2285 of file osrtxml.h.

5.3.1.2 #define rtXmlGetEncBufLen(pctxt) (pctxt) → buffer.byteIndex

This macro returns the length of the encoded XML message.

Parameters:

pctxt Pointer to a context structure.

Definition at line 2334 of file osrtxml.h.

5.3.1.3 #define rtXmlGetEncBufPtr(*pctxt*) (*pctxt*) → **buffer.data**

This macro returns the start address of the encoded XML message.

If a static buffer was used, this is simply the start address of the buffer. If dynamic encoding was done, this will return the start address of the dynamic buffer allocated by the encoder.

Parameters:

pctxt Pointer to a context structure.

Definition at line 2327 of file osrxml.h.

5.3.2 Function Documentation

5.3.2.1 EXTERNXML int rtXmlEncAny (OSCTXT **pctxt*, OSXMLSTRING **pvalue*, 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.

pvalue 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.

5.3.2.2 EXTERNXML int rtXmlEncAnyAttr (OSCTXT **pctxt*, OSRTDList **pAnyAttrList*)

This function encodes a list of OSAnyAttr attributes in which the name and value are given as a UTF-8 string.

Parameters:

pctxt Pointer to context block structure.

pAnyAttrList List of attributes.

Returns:

Completion status of operation:

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

5.3.2.3 EXTERNXML int rtXmlEncAnyTypeValue (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a variable of the XSD anyType type.

This is considered to be a fully-wrapped element of any type, possibly containing attributes. (for example: * <myType>myData</myType>)

Parameters:

pctxt Pointer to context block structure.

pvalue Value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.4 EXTERNXML int rtXmlEncBase64Binary (OSCTXT * *pctxt*, OSUINT32 *nocts*, const OSOCTET * *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD base64Binary type.

Parameters:

pctxt Pointer to context block structure.

nocts Number of octets in the value string.

value 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.

5.3.2.5 EXTERNXML int rtXmlEncBase64BinaryAttr (OSCTXT * *pctxt*, OSUINT32 *nocts*, const OSOCTET * *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes a variable of the XSD base64Binary type as an attribute.

Parameters:

pctxt Pointer to context block structure.

nocts Number of octets in the value string.

value Value to be encoded.

attrName XML attribute name. A name must be provided.

attrNameLen Length in bytes of the attribute name.

Returns:

Completion status of operation:

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

5.3.2.6 EXTERNXML int rtXmlEncBase64StrValue (OSCTXT * *pctxt*, OSUINT32 *nocts*, const OSOCTET * *value*)

This function encodes a variable of the XSD base64Binary type.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

nocts Number of octets in the value string.

value Value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.7 EXTERNXML int rtXmlEncBigInt (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD integer type.

In this case, the integer is assumed to be of a larger size than can fit in a C or C++ long type (normally 32 or 64 bits). For example, parameters used to calculate security values are typically larger than these sizes.

Items of this type are stored in character string constant variables. They can be represented as decimal strings (with no prefixes), as hexadecimal strings starting with a "0x" prefix, as octal strings starting with a "0o" prefix or as binary strings starting with a "0b" prefix. Other radices are currently not supported.

Parameters:

pctxt Pointer to context block structure.

value A pointer to a character string containing the 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.

5.3.2.8 EXTERNXML int rtXmlEncBigIntAttr (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes an XSD integer attribute value.

In this case, the integer is assumed to be of a larger size than can fit in a C or C++ long type (normally 32 or 64 bits).

Parameters:

- pctxt* Pointer to context block structure.
- value* A pointer to a character string containing the value to be encoded.
- attrName* XML attribute name. A name must be provided.
- attrNameLen* Length in bytes of the attribute name.

Returns:

Completion status of operation:

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

5.3.2.9 EXTERNXML int rtXmlEncBigIntValue (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*)

This function encodes an XSD integer attribute value.

In this case, the integer is assumed to be of a larger size than can fit in a C or C++ long type (normally 32 or 64 bits). This function just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

- pctxt* Pointer to context block structure.
- value* A pointer to a character string containing the value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.10 EXTERNXML int rtXmlEncBinStrValue (OSCTXT * *pctxt*, OSUINT32 *nbits*, const OSOCTET * *data*)

This function encodes a binary string value as a sequence of '1's and '0's.

Parameters:

- pctxt* Pointer to context block structure.
- nbits* Number of bits in the value string.
- data* Value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.11 EXTERNXML int rtXmlEncBitString (OSCTXT * *pctxt*, OSUINT32 *nbits*, const OSOCTET * *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the ASN.1 BIT STRING type.

The encoded data is a sequence of '1's and '0's. This is only used if named bits are not specified in the string (

See also:

[rtXmlEncNamedBits](#)).

Parameters:

pctxt Pointer to context block structure.

nbits Number of bits in the bit string.

value 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.

5.3.2.12 EXTERNXML int rtXmlEncBOM (OSCTXT * *pctxt*)

This function encodes the Unicode byte order mark header at the start of the document.

It is called by rtXmlEncStartDocument and does not need to be called manually.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

5.3.2.13 EXTERNXML int rtXmlEncBool (OSCTXT * *pctxt*, OSBOOL *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD boolean type.

Parameters:

pctxt Pointer to context block structure.

value Boolean 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.

5.3.2.14 EXTERNXML int rtXmlEncBoolAttr (OSCTXT * *pctxt*, OSBOOL *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes an XSD boolean attribute value.

Parameters:

pctxt Pointer to context block structure.

value Boolean value to be encoded.

attrName XML attribute name. A name must be provided.

attrNameLen Length in bytes of the attribute name.

Returns:

Completion status of operation:

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

5.3.2.15 EXTERNXML int rtXmlEncBoolValue (OSCTXT * *pctxt*, OSBOOL *value*)

This function encodes a variable of the XSD boolean type.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

value Boolean value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.16 EXTERNXML int rtXmlEncDate (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD 'date' type as a string.

This version of the function is used to encode an OSXSDDateTime value into CCYY-MM-DD format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime 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.

5.3.2.17 EXTERNXML int rtXmlEncDateTime (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a numeric date/time value into an XML string representation.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 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.

5.3.2.18 EXTERNXML int rtXmlEncDateTimeValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a numeric date/time value into an XML string representation.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.19 EXTERNXML int rtXmlEncDateValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a variable of the XSD 'date' type as a string.

This version of the function is used to encode an OSXSDDateTime value into CCYY-MM-DD format. This function just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.20 EXTERNXML int rtXmlEncDecimal (OSCTXT * *pctxt*, OSREAL *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*, const OSDecimalFmt * *pFmtSpec*)

This function encodes a variable of the XSD decimal type.

Parameters:

pctxt Pointer to context block structure.

value 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.

pFmtSpec Pointer to format specification structure.

Returns:

Completion status of operation:

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

5.3.2.21 EXTERNXML int rtXmlEncDecimalAttr (OSCTXT * *pctxt*, OSREAL *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*, const OSDecimalFmt * *pFmtSpec*)

This function encodes a variable of the XSD decimal type as an attribute.

Parameters:

- pctxt* Pointer to context block structure.
- value* Value to be encoded.
- attrName* XML attribute name. A name must be provided.
- attrNameLen* Length of XML attribute name.
- pFmtSpec* Pointer to format specification structure.

Returns:

Completion status of operation:

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

5.3.2.22 EXTERNXML int rtXmlEncDecimalValue (OSCTXT * *pctxt*, OSREAL *value*, const OSDecimalFmt * *pFmtSpec*, char * *pDestBuf*, size_t *destBufSize*)

This function encodes a value of the XSD decimal type.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

- pctxt* Pointer to context block structure.
- value* Value to be encoded.
- pFmtSpec* Pointer to format specification structure.
- pDestBuf* Pointer to a destination buffer. If NULL (*destBufSize* should be 0) the encoded value will be put in *pctxt->buffer* or in stream associated with the *pctxt*.
- destBufSize* The size of the destination buffer. Must be 0, if *pDestBuf* is NULL.

Returns:

Completion status of operation:

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

5.3.2.23 EXTERNXML int rtXmlEncDouble (OSCTXT * *pctxt*, OSREAL *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*, const OSDoubleFmt * *pFmtSpec*)

This function encodes a variable of the XSD double type.

Parameters:

- pctxt* Pointer to context block structure.
- value* 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.
- pFmtSpec* Pointer to format specification structure.

Returns:

Completion status of operation:

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

5.3.2.24 EXTERNXML int rtXmlEncDoubleAttr (OSCTXT * *pctxt*, OSREAL *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*, const OSDoubleFmt * *pFmtSpec*)

This function encodes a variable of the XSD double type as an attribute.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

attrName XML attribute name. A name must be provided.

attrNameLen Length of XML attribute name.

pFmtSpec Pointer to format specification structure.

Returns:

Completion status of operation:

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

5.3.2.25 EXTERNXML int rtXmlEncDoubleValue (OSCTXT * *pctxt*, OSREAL *value*, const OSDoubleFmt * *pFmtSpec*, int *defaultPrecision*)

This function encodes a value of the XSD double or float type.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

pFmtSpec Pointer to format specification structure.

defaultPrecision Default precision of the value. For float, it is 6, for double it is 15.

Returns:

Completion status of operation:

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

5.3.2.26 EXTERNXML int rtXmlEncEmptyElement (OSCTXT * *pctxt*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*, OSRTDList * *pNSAttrs*, OSBOOL *terminate*)

This function encodes an empty element tag value (<elemName/>).

Parameters:

- pctxt* Pointer to context block structure.
- elemName* XML element name.
- pNS* XML namespace information (prefix and URI).
- pNSAttrs* List of namespace attributes to be added to element.
- terminate* Add closing '>' character.

Returns:

Completion status of operation:

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

5.3.2.27 EXTERNXML int rtXmlEncEndDocument (OSCTXT * *pctxt*)

This function adds trailer information and a null terminator at the end of the XML document being encoded.

Parameters:

- pctxt* Pointer to context block structure.

Returns:

Completion status of operation:

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

5.3.2.28 EXTERNXML int rtXmlEncEndElement (OSCTXT * *pctxt*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes an end element tag value (</elemName>).

Parameters:

- pctxt* Pointer to context block structure.
- elemName* XML element name.
- pNS* XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.29 EXTERNXML int rtXmlEncEndSoapElems (OSCTXT * *pctxt*, OSXMLSOAPMsgType *msgtype*)

This function encodes SOAP end element tags.

If will add a SOAP body or fault end tag based on the SOAP message type argument. It will then add an envelope end element tag.

Parameters:

- pctxt* Pointer to context block structure.
- msgtype* SOAP message type (body, fault, or none)

Returns:

Completion status of operation:

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

5.3.2.30 EXTERNXML int rtXmlEncEndSoapEnv (OSCTXT * *pctxt*)

This function encodes a SOAP envelope end element tag (<SOAP-ENV:Envelope/>).

Parameters:

- pctxt* Pointer to context block structure.

Returns:

Completion status of operation:

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

5.3.2.31 EXTERNXML int rtXmlEncFloat (OSCTXT * *pctxt*, OSREAL *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*, const OSDoubleFmt * *pFmtSpec*)

This function encodes a variable of the XSD float type.

Parameters:

- pctxt* Pointer to context block structure.
- value* 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* XML namespace information (prefix and URI).
- pFmtSpec* Pointer to format specification structure.

Returns:

Completion status of operation:

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

5.3.2.32 EXTERNXML int rtXmlEncFloatAttr (OSCTXT * *pctxt*, OSREAL *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*, const OSDoubleFmt * *pFmtSpec*)

This function encodes a variable of the XSD float type as an attribute.

Parameters:

- pctxt* Pointer to context block structure.
- value* Value to be encoded.
- attrName* XML attribute name. A name must be provided.
- attrNameLen* Length of XML attribute name.
- pFmtSpec* Pointer to format specification structure.

Returns:

Completion status of operation:

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

5.3.2.33 EXTERNXML int rtXmlEncGDay (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a numeric gDay element into an XML string representation.

Parameters:

- pctxt* Pointer to context block structure.
- pvalue* Pointer to 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* XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.34 EXTERNXML int rtXmlEncGDayValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a numeric gDay value into an XML string representation.

It just puts the encoded value into the buffer or stream without any tags.

Parameters:

- pctxt* Pointer to context block structure.
- pvalue* Pointer to value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.35 EXTERNXML int rtXmlEncGMonth (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a numeric gMonth element into an XML string representation.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.36 EXTERNXML int rtXmlEncGMonthDay (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a numeric gMonthDay element into an XML string representation.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.37 EXTERNXML int rtXmlEncGMonthDayValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a numeric gMonthDay value into an XML string representation.

It just puts the encoded value into the buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.38 EXTERNXML int rtXmlEncGMonthValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a numeric gMonth value into an XML string representation.

It just puts the encoded value into the buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.39 EXTERNXML int rtXmlEncGYear (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a numeric gYear element into an XML string representation.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.40 EXTERNXML int rtXmlEncGYearMonth (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a numeric gYearMonth element into an XML string representation.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.41 EXTERNXML int rtXmlEncGYearMonthValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a numeric gYearMonth value into an XML string representation.

It just puts the encoded value into the buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.42 EXTERNXML int rtXmlEncGYearValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a numeric gYear value into an XML string representation.

It just puts the encoded value into the buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.43 EXTERNXML int rtXmlEncHexBinary (OSCTXT * *pctxt*, OSUINT32 *nocts*, const OSOCTET * *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD hexBinary type.

Parameters:

pctxt Pointer to context block structure.

nocts Number of octets in the value string.

value 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.44 EXTERNXML int rtXmlEncHexBinaryAttr (OSCTXT * *pctxt*, OSUINT32 *nocts*, const OSOCTET * *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes a variable of the XSD hexBinary type as an attribute.

Parameters:

- pctxt* Pointer to context block structure.
- nocts* Number of octets in the value string.
- value* Value to be encoded.
- attrName* XML attribute name. A name must be provided.
- attrNameLen* Length of XML attribute name.

Returns:

- Completion status of operation:
- 0 = success,
 - negative return value is error.

5.3.2.45 EXTERNXML int rtXmlEncHexStrValue (OSCTXT * *pctxt*, OSUINT32 *nocts*, const OSOCTET * *data*)

This function encodes a variable of the XSD hexBinary type.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

- pctxt* Pointer to context block structure.
- nocts* Number of octets in the value string.
- data* Value to be encoded.

Returns:

- Completion status of operation:
- 0 = success,
 - negative return value is error.

5.3.2.46 EXTERNXML int rtXmlEncIndent (OSCTXT * *pctxt*)

This function adds indentation whitespace to the output stream.

The amount of indentation to add is determined by the level member variable in the context structure and the OSXML-LINDENT constant value.

Parameters:

- pctxt* Pointer to context block structure.

Returns:

- Completion status of operation:
- 0 = success,
 - negative return value is error.

5.3.2.47 EXTERNXML int rtXmlEncInt (OSCTXT * *pctxt*, OSINT32 *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD integer type.

Parameters:

pctxt Pointer to context block structure.

value 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.48 EXTERNXML int rtXmlEncInt64 (OSCTXT * *pctxt*, OSINT64 *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD integer type.

This version of the function is used for 64-bit integer values.

Parameters:

pctxt Pointer to context block structure.

value 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.49 EXTERNXML int rtXmlEncInt64Attr (OSCTXT * *pctxt*, OSINT64 *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes a variable of the XSD integer type as an attribute (name="value").

This version of the function is used for 64-bit integer values.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

attrName XML attribute name.

attrNameLen Length (in bytes) of the attribute name.

Returns:

Completion status of operation:

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

5.3.2.50 EXTERNXML int rtXmlEncInt64Value (OSCTXT * *pctxt*, OSINT64 *value*)

This function encodes a variable of the XSD integer type.

This version of the function is used for 64-bit integer values. It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.51 EXTERNXML int rtXmlEncIntAttr (OSCTXT * *pctxt*, OSINT32 *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes a variable of the XSD integer type as an attribute (name="value").

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

attrName XML attribute name.

attrNameLen Length (in bytes) of the attribute name.

Returns:

Completion status of operation:

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

5.3.2.52 EXTERNXML int rtXmlEncIntPattern (OSCTXT * *pctxt*, OSINT32 *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*, const OSUTF8CHAR * *pattern*)

This function encodes a variable of the XSD integer type using a pattern to specify the format of the integer value.

Parameters:

pctxt Pointer to context block structure.

value 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 XML namespace information (prefix and URI).

pattern Pattern of the encoded value.

Returns:

Completion status of operation:

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

5.3.2.53 EXTERNXML int rtXmlEncIntValue (OSCTXT * *pctxt*, OSINT32 *value*)

This function encodes a variable of the XSD integer type.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.54 EXTERNXML int rtXmlEncNamedBits (OSCTXT * *pctxt*, const OSBitMapItem * *pBitMap*, OSUINT32 *nbits*, const OSOCTET * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the ASN.1 BIT STRING type.

In this case, a set of named bits was provided in the schema for the bit values. The encoding is a list of the named bit identifiers corresponding to each set bit in the bit string value.

Parameters:

pctxt Pointer to context block structure.

pBitMap Bit map equating symbolic bit names to bit numbers.

nbits Number of bits in the bit string value.

pvalue Bit 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.

5.3.2.55 EXTERNXML int rtXmlEncNSAttrs (OSCTXT * *pctxt*, OSRTDList * *pNSAttrs*)

This function encodes namespace declaration attributes at the beginning of an XML document.

The attributes to be encoded are stored in the namespace list provided, or within the context if a NULL pointer is passed for *pNSAttrs*. Namespaces are added to this list by using the namespace utility functions.

Parameters:

pctxt Pointer to context block structure.

pNSAttrs Pointer to list of namespace attributes.

Returns:

Completion status of operation:

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

5.3.2.56 EXTERNXML int rtXmlEncReal10 (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the ASN.1 REAL base 10 type.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to an REAL base 10 value.

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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.57 EXTERNXML int rtXmlEncSoapArrayTypeAttr (OSCTXT * *pctxt*, const OSUTF8CHAR * *name*, const OSUTF8CHAR * *value*, size_t *itemCount*)

This function encodes the special SOAP encoding attrType attribute which specifies the number and type of elements in a SOAP array.

The form of this attribute is 'attrType="<type>[count]"'.

Parameters:

- pctxt* Pointer to context block structure.
- name* Attribute name (NS prefix + arrayType)
- value* UTF-8 string value to be encoded.
- itemCount* Count of the number of elements in the array.

Returns:

Completion status of operation:

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

5.3.2.58 EXTERNXML int rtXmlEncStartDocument (OSCTXT * *pctxt*)

This function encodes the XML header text at the beginning of an XML document.

This is normally the following:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Parameters:

- pctxt* Pointer to context block structure.

Returns:

Completion status of operation:

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

5.3.2.59 EXTERNXML int rtXmlEncStartElement (OSCTXT * *pctxt*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*, OSRTDList * *pNSAttrs*, OSBOOL *terminate*)

This function encodes a start element tag value (<elemName>).

Parameters:

- pctxt* Pointer to context block structure.
- elemName* XML element name.
- pNS* XML namespace information (prefix and URI).
- pNSAttrs* List of namespace attributes to be added to element.
- terminate* Add closing '>' character.

Returns:

Completion status of operation:

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

5.3.2.60 EXTERNXML int rtXmlEncStartSoapElems (OSCTXT * *pctxt*, OSXMLSOAPMsgType *msgtype*)

This function encodes a SOAP envelope start element tag and an optional SOAP body or fault tag.

This includes all of the standard SOAP namespace attributes.

Parameters:

pctxt Pointer to context block structure.

msgtype SOAP message type (body, fault, or none)

Returns:

Completion status of operation:

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

5.3.2.61 EXTERNXML int rtXmlEncStartSoapEnv (OSCTXT * *pctxt*)

This function encodes a SOAP envelope start element tag.

This includes all of the standard SOAP namespace attributes.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

5.3.2.62 EXTERNXML int rtXmlEncString (OSCTXT * *pctxt*, OSXMLSTRING * *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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.63 EXTERNXML int rtXmlEncStringValue (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*)

This function encodes a variable of the XSD string type.

Parameters:

pctxt Pointer to context block structure.

value XML string value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.64 EXTERNXML int rtXmlEncStringValue2 (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*, size_t *valueLen*)

This function encodes a variable of the XSD string type.

Parameters:

pctxt Pointer to context block structure.

value XML string value to be encoded.

valueLen UTF-8 string value length (in octets).

Returns:

Completion status of operation:

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

5.3.2.65 EXTERNXML int rtXmlEncTermStartElement (OSCTXT * *pctxt*)

This function terminates a currently open XML start element by adding either a '>' or '>' (if empty) terminator. It will also add XSI attributes to the element.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

5.3.2.66 EXTERNXML int rtXmlEncTime (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD 'time' type as a string.

This version of the function is used to encode OSXSDDateTime value into any of following format in different condition as stated below. (1) hh-mm-ss.ss used if *tz_flag* = false (2) hh-mm-ss.ssZ used if *tz_flag* = false and *tzo* = 0 (3) hh-mm-ss.ss+HH:MM if *tz_flag* = false and *tzo* > 0 (4) hh-mm-ss.ss-HH:MM-HH:MM if *tz_flag* = false and *tzo* < 0

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime 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.

5.3.2.67 EXTERNXML int rtXmlEncTimeValue (OSCTXT * *pctxt*, const OSXSDDateTime * *pvalue*)

This function encodes a variable of the XSD 'time' type as a string.

This version of the function just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.68 EXTERNXML int rtXmlEncUInt (OSCTXT * *pctxt*, OSUINT32 *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD unsigned integer type.

Parameters:

pctxt Pointer to context block structure.

value 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.69 EXTERNXML int rtXmlEncUInt64 (OSCTXT * *pctxt*, OSUINT64 *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a variable of the XSD integer type.

This version of the function is used when constraints cause an unsigned 64-bit integer variable to be used.

Parameters:

pctxt Pointer to context block structure.

value 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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.70 EXTERNXML int rtXmlEncUInt64Attr (OSCTXT * *pctxt*, OSUINT64 *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes a variable of the XSD integer type as an attribute (name="value").

This version of the function is used for unsigned 64-bit integer values.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

attrName XML attribute name.

attrNameLen Length (in bytes) of the attribute name.

Returns:

Completion status of operation:

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

5.3.2.71 EXTERNXML int rtXmlEncUInt64Value (OSCTXT * *pctxt*, OSUINT64 *value*)

This function encodes a variable of the XSD integer type.

This version of the function is used when constraints cause an unsigned 64-bit integer variable to be used. It writes the encoded value to the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.72 EXTERNXML int rtXmlEncUIntAttr (OSCTXT * *pctxt*, OSUINT32 *value*, const OSUTF8CHAR * *attrName*, size_t *attrNameLen*)

This function encodes a variable of the XSD unsigned integer type as an attribute (name="value").

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

attrName XML attribute name.

attrNameLen Length (in bytes) of the attribute name.

Returns:

Completion status of operation:

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

5.3.2.73 EXTERNXML int rtXmlEncUIntValue (OSCTXT * *pctxt*, OSUINT32 *value*)

This function encodes a variable of the XSD unsigned integer type.

It just puts the encoded value in the destination buffer or stream without any tags.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.74 EXTERNXML int rtXmlEncUnicodeStr (OSCTXT * *pctxt*, const OSUNICHAR * *value*, OSUINT32 *nchars*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a Unicode string value.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded. This is a pointer to an array of 16-bit integer values.

nchars Number of characters in value array.

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 XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.75 EXTERNXML int rtXmlEncUTF8Attr (OSCTXT * *pctxt*, const OSUTF8CHAR * *name*, const OSUTF8CHAR * *value*)

This function encodes an attribute in which the name and value are given as a null-terminated UTF-8 strings.

Parameters:

pctxt Pointer to context block structure.

name Attribute name.

value UTF-8 string value to be encoded.

Returns:

Completion status of operation:

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

5.3.2.76 EXTERNXML int rtXmlEncUTF8Attr2 (OSCTXT * *pctxt*, const OSUTF8CHAR * *name*, size_t *nameLen*, const OSUTF8CHAR * *value*, size_t *valueLen*)

This function encodes an attribute in which the name and value are given as a UTF-8 strings with lengths.

Parameters:

- pctxt* Pointer to context block structure.
- name* Attribute name.
- nameLen* Attribute name length (in octets).
- value* UTF-8 string value to be encoded.
- valueLen* UTF-8 string value length (in octets).

Returns:

Completion status of operation:

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

5.3.2.77 EXTERNXML int rtXmlEncUTF8Str (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a UTF-8 string value.

Parameters:

- pctxt* Pointer to context block structure.
- value* 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* XML namespace information (prefix and URI).

Returns:

Completion status of operation:

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

5.3.2.78 EXTERNXML int rtXmlEncXSIAttrs (OSCTXT * *pctxt*, OSBOOL *needXSI*)

This function encodes XML schema instance (XSI) attributes at the beginning of an XML document.

The encoded attributes include the XSI namespace declaration, the XSD schema location attribute, and the XSD no namespace schema location attribute.

The XSI namespace declaration will only be added to the document if schema location attributes exist or the 'needXSI' flag (see below) is set.

Parameters:

- pctxt* Pointer to context block structure.

needXSI This flag is set to true to indicate the XSI namespace declaration is required to support XML items in the main document body such as `xsi:type` or `xsi:nil`.

Returns:

Completion status of operation:

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

5.3.2.79 EXTERNXML int rtXmlEncXSITypeAttr (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*)

This function encodes an XML schema instance (XSI) type attribute value (`xsi:type="value"`).

Parameters:

pctxt Pointer to context block structure.

value XSI type attribute value.

Returns:

Completion status of operation:

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

5.3.2.80 EXTERNXML int rtXmlFreeInputSource (OSCTXT * *pctxt*)

This function closes an input source that was previously created with one of the create input source functions such as `'rtXmlCreateFileInputSource'`.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

5.3.2.81 EXTERNXML int rtXmlGetIndent (OSCTXT * *pctxt*)

This function returns current XML output indent value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Current indent value (≥ 0) if OK, negative status code if error.

5.3.2.82 EXTERNXML int rtXmlGetIndentChar (OSCTXT * *pctxt*)

This function returns current XML output indent character value (default is space).

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Current indent character (> 0) if OK, negative status code if error.

5.3.2.83 EXTERNXML OSBOOL rtXmlGetWriteBOM (OSCTXT * *pctxt*)

This function returns whether the Unicode byte order mark will be encoded.

Parameters:

pctxt Pointer to OSCTXT structure.

Returns:

TRUE if BOM is to be encoded, FALSE if not or if context uninitialized.

5.3.2.84 EXTERNXML int rtXmlSetEncBufPtr (OSCTXT * *pctxt*, OSOCTET * *bufaddr*, size_t *bufsiz*)

This function is used to set the internal buffer within the run-time library encoding context.

It must be called after the context variable is initialized by the rtxInitContext function and before any other compiler generated or run-time library encode function.

Parameters:

pctxt Pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

bufaddr A pointer to a memory buffer to use to encode a message. The buffer should be declared as an array of unsigned characters (OCTETs). This parameter can be set to NULL to specify dynamic encoding (i.e., the encode functions will dynamically allocate a buffer to hold the encoded message).

bufsiz The length of the memory buffer in bytes. Should be set to zero if NULL was specified for *bufaddr* (i.e. dynamic encoding was selected).

5.3.2.85 EXTERNXML int rtxPrintNSAttrs (const char * *name*, const OSRTDList *data*)

This function prints a list of namespace attributes.

Parameters:

name Name to print.

data List of namespace attributes.

Returns:

Completion status of operation:

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

5.4 XML utility functions.

Functions

- EXTERNXML int **rtXmlWriteToFile** (OSCTXT *pctxt, const char *filename)
This function writes the encoded XML message stored in the context message buffer out to a file.
- EXTERNXML void **rtXmlTreatWhitespaces** (OSCTXT *pctxt, int whiteSpaceType)

5.4.1 Function Documentation

5.4.1.1 EXTERNXML int **rtXmlWriteToFile** (OSCTXT * *pctxt*, const char * *filename*)

This function writes the encoded XML message stored in the context message buffer out to a file.

Parameters:

pctxt Pointer to OSCTXT structure.

filename Full path to file to which XML is to be written.

Returns:

0 - if success, negative value if error.

5.5 XML pull-parser decode functions.

Functions

- EXTERNXML int [rtXmlpDecAny](#) (OSCTXT *pctxt, const OSUTF8CHAR **pvalue)
This function decodes an arbitrary XML section of code as defined by the XSD any type (xsd:any).
- EXTERNXML int [rtXmlpDecAnyAttrStr](#) (OSCTXT *pctxt, const OSUTF8CHAR **ppAttrStr, size_t index)
This function decodes an any attribute string.
- EXTERNXML int [rtXmlpDecAnyElem](#) (OSCTXT *pctxt, const OSUTF8CHAR **pvalue)
This function decodes an arbitrary XML section of code as defined by the XSD any type (xsd:any).
- EXTERNXML int [rtXmlpDecBase64Str](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)
This function decodes a contents of a Base64-encode binary string into a static memory structure.
- EXTERNXML int [rtXmlpDecBigInt](#) (OSCTXT *pctxt, const OSUTF8CHAR **pvalue)
This function will decode a variable of the XSD integer type.
- EXTERNXML int [rtXmlpDecBitString](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnbits, OSUINT32 bufsize)
This function decodes a bit string value.
- EXTERNXML int [rtXmlpDecBool](#) (OSCTXT *pctxt, OSBOOL *pvalue)
This function decodes a variable of the boolean type.
- EXTERNXML int [rtXmlpDecDate](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'date' type.
- EXTERNXML int [rtXmlpDecDateTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'dateTime' type.
- EXTERNXML int [rtXmlpDecDecimal](#) (OSCTXT *pctxt, OSREAL *pvalue, int totalDigits, int fraction-Digits)
This function decodes the contents of a decimal data type.
- EXTERNXML int [rtXmlpDecDouble](#) (OSCTXT *pctxt, OSREAL *pvalue)
This function decodes the contents of a float or double data type.
- EXTERNXML int [rtXmlpDecDynBase64Str](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a contents of a Base64-encode binary string.
- EXTERNXML int [rtXmlpDecDynBitString](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a bit string value.
- EXTERNXML int [rtXmlpDecDynHexStr](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a contents of a hexBinary string.
- EXTERNXML int [rtXmlpDecDynUnicodeStr](#) (OSCTXT *pctxt, const OSUNICHAR **ppdata, OSUINT32 *pnchars)

This function decodes a Unicode string data type.

- EXTERNXML int [rtXmlpDecDynUTF8Str](#) (OSCTXT *pctxt, const OSUTF8CHAR **outdata)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlpDecUTF8Str](#) (OSCTXT *pctxt, OSUTF8CHAR *out, size_t max_len)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlpDecGDay](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gDay' type.
- EXTERNXML int [rtXmlpDecGMonth](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gMonth' type.
- EXTERNXML int [rtXmlpDecGMonthDay](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gMonthDay' type.
- EXTERNXML int [rtXmlpDecGYear](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gYear' type.
- EXTERNXML int [rtXmlpDecGYearMonth](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gYearMonth' type.
- EXTERNXML int [rtXmlpDecHexStr](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocets, OSINT32 bufsize)
This function decodes the contents of a hexBinary string into a static memory structure.
- EXTERNXML int [rtXmlpDecInt](#) (OSCTXT *pctxt, OSINT32 *pvalue)
This function decodes the contents of a 32-bit integer data type.
- EXTERNXML int [rtXmlpDecInt8](#) (OSCTXT *pctxt, OSINT8 *pvalue)
This function decodes the contents of an 8-bit integer data type (i.e.
- EXTERNXML int [rtXmlpDecInt16](#) (OSCTXT *pctxt, OSINT16 *pvalue)
This function decodes the contents of a 16-bit integer data type.
- EXTERNXML int [rtXmlpDecInt64](#) (OSCTXT *pctxt, OSINT64 *pvalue)
This function decodes the contents of a 64-bit integer data type.
- EXTERNXML int [rtXmlpDecNamedBits](#) (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSOCTET *pvalue, OSUINT32 *pnbits, OSUINT32 bufsize)
This function decodes the contents of a named bit field.
- EXTERNXML int [rtXmlpDecStrList](#) (OSCTXT *pctxt, OSRTDList *plist)
This function decodes a list of space-separated tokens and returns each token as a separate item on the given list.
- EXTERNXML int [rtXmlpDecTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'time' type.
- EXTERNXML int [rtXmlpDecUInt](#) (OSCTXT *pctxt, OSUINT32 *pvalue)

This function decodes the contents of an unsigned 32-bit integer data type.

- EXTERNXML int [rtXmlpDecUInt8](#) (OSCTXT *pctxt, OSOCTET *pvalue)

This function decodes the contents of an unsigned 8-bit integer data type (i.e.

- EXTERNXML int [rtXmlpDecUInt16](#) (OSCTXT *pctxt, OSUINT16 *pvalue)

This function decodes the contents of an unsigned 16-bit integer data type.

- EXTERNXML int [rtXmlpDecUInt64](#) (OSCTXT *pctxt, OSUINT64 *pvalue)

This function decodes the contents of an unsigned 64-bit integer data type.

- EXTERNXML int [rtXmlpDecXmlStr](#) (OSCTXT *pctxt, OSXMLSTRING *outdata)

This function decodes the contents of an XML string data type.

- EXTERNXML int [rtXmlpDecXmlStrList](#) (OSCTXT *pctxt, OSRTDList *plist)

This function decodes a list of space-separated tokens and returns each token as a separate item on the given list.

- EXTERNXML int [rtXmlpDecXSIAAttr](#) (OSCTXT *pctxt, const [OSXMLNameFragments](#) *attrName)

This function decodes XSI (XML Schema Instance) and XML namespace attributes that may be present in any arbitrary XML element within a document.

- EXTERNXML int [rtXmlpDecXSITypeAttr](#) (OSCTXT *pctxt, const [OSXMLNameFragments](#) *attrName, const OSUTF8CHAR **ppAttrValue)

This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type).

- EXTERNXML int [rtXmlpGetAttributeID](#) (const [OSXMLStrFragment](#) *attrName, OSINT16 nsidx, size_t nAttr, const [OSXMLAttrDescr](#) attrNames[], OSUINT32 attrPresent[])

This function finds an attribute in the descriptor table.

- EXTERNXML int [rtXmlpGetNextElem](#) (OSCTXT *pctxt, [OSXMLElemDescr](#) *pElem, OSINT32 level)

This function parse the next element start tag.

- EXTERNXML int [rtXmlpGetNextElemID](#) (OSCTXT *pctxt, const [OSXMLElemIDRec](#) *tab, size_t nrows, OSINT32 level, OSBOOL continueParse)

This function parses the next start tag and finds the index of the element name in the descriptor table.

- EXTERNXML int [rtXmlpMarkLastEventActive](#) (OSCTXT *pctxt)

This function marks current tag as unprocessed.

- EXTERNXML int [rtXmlpMatchStartTag](#) (OSCTXT *pctxt, const OSUTF8CHAR *elemLocalName, OSINT16 nsidx)

This function parses the next start tag that matches with given name.

- EXTERNXML int [rtXmlpMatchEndTag](#) (OSCTXT *pctxt, OSINT32 level)

This function parse next end tag that matches with given name.

- EXTERNXML OSBOOL [rtXmlpHasAttributes](#) (OSCTXT *pctxt)

This function checks accessibility of attributes.

- EXTERNXML int [rtXmlpGetAttributeCount](#) (OSCTXT *pctxt)

This function returns number of attributes in last processed start tag.

- EXTERNXML int [rtXmlpSelectAttribute](#) (OSCTXT *pctx, [OSXMLNameFragments](#) *pAttr, OSINT16 *nsidx, size_t index)
This function selects attribute to decode.
- EXTERNXML OSINT32 [rtXmlpGetCurrentLevel](#) (OSCTXT *pctx)
This function returns current nesting level.
- EXTERNXML void [rtXmlpSetWhiteSpaceMode](#) (OSCTXT *pctx, [OSXMLWhiteSpaceMode](#) whiteSpaceMode)
Sets the whitespace treatment mode.
- EXTERNXML OSBOOL [rtXmlpSetMixedContentMode](#) (OSCTXT *pctx, OSBOOL mixedContentMode)
Sets mixed content mode.
- EXTERNXML void [rtXmlpSetListMode](#) (OSCTXT *pctx)
Sets list mode.
- EXTERNXML OSBOOL [rtXmlpListHasItem](#) (OSCTXT *pctx)
Check for end of decoded token list.
- EXTERNXML void [rtXmlpCountListItems](#) (OSCTXT *pctx, OSUINT32 *itemCnt)
Count tokens in list.
- EXTERNXML int [rtXmlpGetNextSeqElemID](#) (OSCTXT *pctx, const [OSXMLElemIDRec](#) *tab, const [OSXMLGroupDesc](#) *pGroup, int curID, int lastMandatoryID, OSBOOL groupMode)
This function parses the next start tag and find index of element name in descriptor table.
- EXTERNXML int [rtXmlpGetNextAllElemID](#) (OSCTXT *pctx, const [OSXMLElemIDRec](#) *tab, size_t nRows, const OSUINT8 *pOrder, OSUINT32 nOrder, OSUINT32 maxOrder, int anyID)
This function parses the next start tag and finds index of element name in descriptor table.
- EXTERNXML int [rtXmlpGetNextAllElemID16](#) (OSCTXT *pctx, const [OSXMLElemIDRec](#) *tab, size_t nRows, const OSUINT16 *pOrder, OSUINT32 nOrder, OSUINT32 maxOrder, int anyID)
This function parses the next start tag and finds index of element name in descriptor table.
- EXTERNXML void [rtXmlpSetNamespaceTable](#) (OSCTXT *pctx, const OSUTF8CHAR *namespaceTable[], size_t nmNamespaces)
Sets user namespace table.
- EXTERNXML int [rtXmlpCreateReader](#) (OSCTXT *pctx)
Creates pull parser reader structure within the context.
- EXTERNXML void [rtXmlpHideAttributes](#) (OSCTXT *pctx)
Disable access to attributes.
- EXTERNXML OSBOOL [rtXmlpNeedDecodeAttributes](#) (OSCTXT *pctx)
This function checks if attributes were previously decoded.
- EXTERNXML void [rtXmlpMarkPos](#) (OSCTXT *pctx)

Save current decode position.

- EXTERNXML void [rtXmlpRewindToMarkedPos](#) (OSCTXT *pctxt)
Rewind to saved decode position.
- EXTERNXML void [rtXmlpResetMarkedPos](#) (OSCTXT *pctxt)
Reset saved decode position.
- EXTERNXML int [rtXmlpGetXSITypeAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR **ppAttrValue, OSINT16 *nsidx, size_t *pLocalOffs)
This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type).
- EXTERNXML int [rtXmlpGetXmlnsAttrs](#) (OSCTXT *pctxt, OSRTDList *pNSAttrs)
This function decodes namespace attributes from start tag and adds them to the given list.
- EXTERNXML int [rtXmlpDecXSIAAttrs](#) (OSCTXT *pctxt)
This function decodes XSI (XML Schema Instance) that may be present in any arbitrary XML element within a document.
- EXTERNXML OSBOOL [rtXmlpIsEmptyElement](#) (OSCTXT *pctxt)
Check element content: empty or not.
- EXTERNXML int [rtXmlEncAttrC14N](#) (OSCTXT *pctxt)
This function used only in C14 mode.
- EXTERNXML OSBOOL [rtXmlpIsLastEventDone](#) (OSCTXT *pctxt)
Check processing status of current tag.
- EXTERNXML int [rtXmlpGetXSITypeIndex](#) (OSCTXT *pctxt, const OSXMLItemDescr typetab[], size_t type-
tabsiz)
*This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type) and find type index in
descriptor table.*
- EXTERNXML int [rtXmlpLookupXSITypeIndex](#) (OSCTXT *pctxt, const OSUTF8CHAR *pXsiType, OS-
INT16 xsiTypeIdx, const OSXMLItemDescr typetab[], size_t typetabsiz)
This function find index of XSI (XML Schema Instance) type in descriptor table.
- EXTERNXML void [rtXmlpForceDecodeAsGroup](#) (OSCTXT *pctxt)
Disable skipping of unknown elements in optional sequence tail.
- EXTERNXML OSBOOL [rtXmlpIsDecodeAsGroup](#) (OSCTXT *pctxt)
This function checks if "decode as group" mode was forced.

5.5.1 Function Documentation

5.5.1.1 EXTERNXML int rtXmlEncAttrC14N (OSCTXT * pctxt)

This function used only in C14 mode.

It provide attributes sorting.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

5.5.1.2 EXTERNXML void rtXmlpCountListItems (OSCTXT * *pctxt*, OSUINT32 * *itemCnt*)

Count tokens in list.

Parameters:

pctxt Pointer to context block structure.

itemCnt Pointer to number of elements in list.

Returns:

Token number. For element content function check accessed part of content only. Returned value may be below then real token number.

5.5.1.3 EXTERNXML int rtXmlpCreateReader (OSCTXT * *pctxt*)

Creates pull parser reader structure within the context.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

5.5.1.4 EXTERNXML int rtXmlpDecAny (OSCTXT * *pctxt*, const OSUTF8CHAR ** *pvalue*)

This function decodes an arbitrary XML section of code as defined by the XSD any type (xsd:any).

The decoded XML fragment is returned as a string in the form as it appears in the document. Memory is allocated for the string using the rtxMemAlloc function.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to UTF8 character string pointer to receive decoded XML fragment. Memory is allocated for the string using the run-time memory manager.

Returns:

Completion status of operation:

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

5.5.1.5 EXTERNXML int rtXmlpDecAnyAttrStr (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppAttrStr*, size_t *index*)

This function decodes an any attribute string.

The full attribute string (name="value") is decoded and returned on the string output argument. Memory is allocated for the string using the rtxMemAlloc function.

Parameters:

pctxt Pointer to context block structure.

ppAttrStr Pointer to UTF8 character string pointer to receive decoded attribute string. Memory is allocated for the string using the run-time memory manager.

index Index of attribute.

Returns:

Completion status of operation:

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

5.5.1.6 EXTERNXML int rtXmlpDecAnyElem (OSCTXT * *pctxt*, const OSUTF8CHAR ** *pvalue*)

This function decodes an arbitrary XML section of code as defined by the XSD any type (xsd:any).

The decoded XML fragment is returned as a string in the form as it appears in the document. Memory is allocated for the string using the rtxMemAlloc function. The difference between this function and rtXmlpDecAny is that this function preserves the full encoded XML fragment including the start and end elements tags and attributes. rtXmlpDecAny decodes the contents within the start and end tags.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to UTF8 character string pointer to receive decoded XML fragment. Memory is allocated for the string using the run-time memory manager.

Returns:

Completion status of operation:

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

5.5.1.7 EXTERNXML int rtXmlpDecBase64Str (OSCTXT * *pctxt*, OSOCTET * *pvalue*, OSUINT32 * *pnocts*, OSINT32 *bufsize*)

This function decodes a contents of a Base64-encode binary string into a static memory structure.

The octet string must be Base64 encoded. This function call is used to decode a sized base64Binary string production. Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the *bufsize* input parameter.

pnocts A pointer to an integer value to receive the decoded number of octets.

bufsize The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns:

Completion status of operation:

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

5.5.1.8 EXTERNXML int rtXmlpDecBigInt (OSCTXT * *pctxt*, const OSUTF8CHAR ** *pvalue*)

This function will decode a variable of the XSD integer type.

In this case, the integer is assumed to be of a larger size than can fit in a C or C++ long type (normally 32 or 64 bits). For example, parameters used to calculate security values are typically larger than these sizes.

These variables are stored in character string constant variables. The radix should be 10. If it is necessary to convert to another radix, then use `rtxBigIntSetStr` or `rtxBigIntToString` functions. Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to a pointer to receive decoded UTF-8 string. Dynamic memory is allocated for the variable using the `rtMemAlloc` function. The decoded variable is represented as a string starting with appropriate prefix.

Returns:

Completion status of operation:

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

5.5.1.9 EXTERNXML int rtXmlpDecBitString (OSCTXT * *pctxt*, OSOCTET * *pvalue*, OSUINT32 * *pnbits*, OSUINT32 *bufsize*)

This function decodes a bit string value.

The string consists of a series of '1' and '0' characters. This is the static version in which the user provides a pre-allocated memory buffer to receive the decoded data. One byte in a memory buffer can hold 8 characters of encoded data. Bits are stored from MSB to LSB order.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to a variable to receive the decoded boolean value.

pnbits Pointer to hold decoded number of bits.

bufsize Size of buffer passed in pvalue argument.

Returns:

Completion status of operation:

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

5.5.1.10 EXTERNXML int rtXmlpDecBool (OSCTXT * *pctxt*, OSBOOL * *pvalue*)

This function decodes a variable of the boolean type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to a variable to receive the decoded boolean value.

Returns:

Completion status of operation:

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

5.5.1.11 EXTERNXML int rtXmlpDecDate (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'date' type.

Input is expected to be a string of characters returned by an XML pull parser. The string should have CCYY-MM-DD format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.12 EXTERNXML int rtXmlpDecDateTime (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'dateTime' type.

Input is expected to be a string of characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.13 EXTERNXML int rtXmlpDecDecimal (OSCTXT * *pctxt*, OSREAL * *pvalue*, int *totalDigits*, int *fractionDigits*)

This function decodes the contents of a decimal data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 64-bit double value to receive decoded result.

totalDigits Number of total digits in the decimal number from XSD totalDigits facet value. Argument should be set to -1 if facet not given.

fractionDigits Number of fraction digits in the decimal number from XSD fractionDigits facet value. Argument should be set to -1 if facet not given.

Returns:

Completion status of operation:

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

5.5.1.14 EXTERNXML int rtXmlpDecDouble (OSCTXT * *pctxt*, OSREAL * *pvalue*)

This function decodes the contents of a float or double data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 64-bit double value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.15 EXTERNXML int rtXmlpDecDynBase64Str (OSCTXT * *pctxt*, OSDynOctStr * *pvalue*)

This function decodes a contents of a Base64-encode binary string.

The octet string must be Base64 encoded. This function will allocate dynamic memory to store the decoded result. Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated for the string using the rtxMemAlloc function.

Returns:

Completion status of operation:

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

5.5.1.16 EXTERNXML int rtXmlpDecDynBitString (OSCTXT * *pctxt*, OSDynOctStr * *pvalue*)

This function decodes a bit string value.

The string consists of a series of '1' and '0' characters. This is the dynamic version in which memory is allocated for the returned binary string variable. Bits are stored from MSB to LSB order.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to a variable to receive the decoded boolean value.

Returns:

Completion status of operation:

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

5.5.1.17 EXTERNXML int rtXmlpDecDynHexStr (OSCTXT * *pctxt*, OSDynOctStr * *pvalue*)

This function decodes a contents of a hexBinary string.

This function will allocate dynamic memory to store the decoded result. Input is expected to be a string of OS-UTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated to hold the string using the rtxMemAlloc function.

Returns:

Completion status of operation:

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

5.5.1.18 EXTERNXML int rtXmlpDecDynUnicodeStr (OSCTXT * *pctxt*, const OSUNICHAR ** *ppdata*, OSUINT32 * *pchars*)

This function decodes a Unicode string data type.

The input is assumed to be in UTF-8 format. This function reads each character and converts it into its Unicode equivalent.

Parameters:

pctxt Pointer to context block structure.

ppdata Pointer to Unicode character string. A Unicode character string is represented as an array of unsigned 16-bit integers in C. Memory is allocated for the string using the run-time memory manager.

pchars Pointer to integer variables to receive the number of characters in the string.

Returns:

Completion status of operation:

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

5.5.1.19 EXTERNXML int rtXmlpDecDynUTF8Str (OSCTXT * *pctxt*, const OSUTF8CHAR ** *outdata*)

This function decodes the contents of a UTF-8 string data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

outdata Pointer to a pointer to receive decoded UTF-8 string. Memory is allocated for this string using the run-time memory manager.

Returns:

Completion status of operation:

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

5.5.1.20 EXTERNXML int rtXmlpDecGDay (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gDay' type.

Input is expected to be a string of characters returned by an XML pull parser. The string should have —DD[-+hh:mm[Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.21 EXTERNXML int rtXmlpDecGMonth (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gMonth' type.

Input is expected to be a string of characters returned by an XML pull parser. The string should have –MM[+hh:mm|Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.22 EXTERNXML int rtXmlpDecGMonthDay (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gMonthDay' type.

Input is expected to be a string of characters returned by an XML pull parser. The string should have –MM-DD[+hh:mm|Z] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.23 EXTERNXML int rtXmlpDecGYear (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gYear' type.

Input is expected to be a string of characters returned by an XML pull parser. The string should have CCYY[-hh:mm[Z]] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.24 EXTERNXML int rtXmlpDecGYearMonth (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'gYearMonth' type.

Input is expected to be a string of characters returned by an XML pull parser. The string should have CCYY-MM[-hh:mm[Z]] format.

Parameters:

pctxt Pointer to context block structure.

pvalue OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.25 EXTERNXML int rtXmlpDecHexStr (OSCTXT * *pctxt*, OSOCTET * *pvalue*, OSUINT32 * *pnocts*, OSINT32 *bufsize*)

This function decodes the contents of a hexBinary string into a static memory structure.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

pvalue A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the bufsize input parameter.

pnocts A pointer to an integer value to receive the decoded number of octets.

bufsize The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns:

Completion status of operation:

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

5.5.1.26 EXTERNXML int rtXmlpDecInt (OSCTXT * *pctxt*, OSINT32 * *pvalue*)

This function decodes the contents of a 32-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 32-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.27 EXTERNXML int rtXmlpDecInt16 (OSCTXT * *pctxt*, OSINT16 * *pvalue*)

This function decodes the contents of a 16-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 16-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.28 EXTERNXML int rtXmlpDecInt64 (OSCTXT * *pctxt*, OSINT64 * *pvalue*)

This function decodes the contents of a 64-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 64-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.29 EXTERNXML int rtXmlpDecInt8 (OSCTXT * *pctxt*, OSINT8 * *pvalue*)

This function decodes the contents of an 8-bit integer data type (i.e.

a signed byte type in the range of -128 to 127). Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to 8-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.30 EXTERNXML int rtXmlpDecNamedBits (OSCTXT * *pctxt*, const OSBitMapItem * *pBitMap*, OSOCTET * *pvalue*, OSUINT32 * *pnbits*, OSUINT32 *bufsize*)

This function decodes the contents of a named bit field.

This is a space-separated list of token values in which each token corresponds to a bit field in a bit map.

Parameters:

pctxt Pointer to context block structure.

pBitMap Pointer to bit map structure that defined token to bit mappings.

pvalue Pointer to buffer to receive decoded bit map.

pnbits Number of bits in decoded bit map.

bufsize Size of buffer passed in to received decoded bit values.

Returns:

Completion status of operation:

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

5.5.1.31 EXTERNXML int rtXmlpDecStrList (OSCTXT * *pctxt*, OSRTDList * *plist*)

This function decodes a list of space-separated tokens and returns each token as a separate item on the given list. Memory is allocated for the list nodes and token values using the rtx memory management functions.

Parameters:

- pctxt* A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
- plist* A pointer to a linked list structure to which the parsed token values will be added.

Returns:

Completion status of operation:

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

5.5.1.32 EXTERNXML int rtXmlpDecTime (OSCTXT * *pctxt*, OSXSDDateTime * *pvalue*)

This function decodes a variable of the XSD 'time' type.

Input is expected to be a string of characters returned by an XML pull parser. The string should have one of following formats:

(1) hh-mm-ss.ss used if *tz_flag* = false (2) hh-mm-ss.ssZ used if *tz_flag* = false and *tzo* = 0 (3) hh-mm-ss.ss+HH:MM if *tz_flag* = false and *tzo* > 0 (4) hh-mm-ss.ss-HH:MM-HH:MM if *tz_flag* = false and *tzo* < 0

Parameters:

- pctxt* Pointer to context block structure.
- pvalue* OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.33 EXTERNXML int rtXmlpDecUInt (OSCTXT * *pctxt*, OSUINT32 * *pvalue*)

This function decodes the contents of an unsigned 32-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

- pctxt* Pointer to context block structure.
- pvalue* Pointer to unsigned 32-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.34 EXTERNXML int rtXmlpDecUInt16 (OSCTXT * *pctxt*, OSUINT16 * *pvalue*)

This function decodes the contents of an unsigned 16-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned 16-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.35 EXTERNXML int rtXmlpDecUInt64 (OSCTXT * *pctxt*, OSUINT64 * *pvalue*)

This function decodes the contents of an unsigned 64-bit integer data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned 64-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.36 EXTERNXML int rtXmlpDecUInt8 (OSCTXT * *pctxt*, OSOCTET * *pvalue*)

This function decodes the contents of an unsigned 8-bit integer data type (i.e.

a signed byte type in the range of 0 to 255). Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned 8-bit integer value to receive decoded result.

Returns:

Completion status of operation:

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

5.5.1.37 EXTERNXML int rtXmlpDecUTF8Str (OSCTXT * *pctxt*, OSUTF8CHAR * *out*, size_t *max_len*)

This function decodes the contents of a UTF-8 string data type.

Input is expected to be a string of OSUTF8CHAR characters returned by an XML pull parser.

Parameters:

pctxt Pointer to context block structure.

out Pointer to an array of OSUTF8CHAR to receive decoded UTF-8 string.

max_len Length of out array.

Returns:

Completion status of operation:

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

5.5.1.38 EXTERNXML int rtXmlpDecXmlStr (OSCTXT * *pctxt*, OSXMLSTRING * *outdata*)

This function decodes the contents of an XML string data type.

This type contains a pointer to a UTF-8 character string plus flags that can be set to alter the encoding of the string (for example, the *CDATA* flag allows the string to be encoded in a CDATA section). Input is expected to be a string of UTF-8 characters returned by an XML parser.

Parameters:

pctxt Pointer to context block structure.

outdata Pointer to an XML string structure to receive the decoded string. Memory is allocated for the string using the run-time memory manager.

Returns:

Completion status of operation:

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

5.5.1.39 EXTERNXML int rtXmlpDecXmlStrList (OSCTXT * *pctxt*, OSRTDList * *plist*)

This function decodes a list of space-separated tokens and returns each token as a separate item on the given list.

Memory is allocated for the list nodes and token values using the *rtx* memory management functions. List contains OSXMLSTRING structures.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

plist A pointer to a linked list structure to which the parsed token values will be added.

Returns:

Completion status of operation:

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

5.5.1.40 EXTERNXML int rtXmlpDecXSIAttr (OSCTXT * *pctxt*, const OSXMLNameFragments * *attrName*)

This function decodes XSI (XML Schema Instance) and XML namespace attributes that may be present in any arbitrary XML element within a document.

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

attrName A pointer to a structure holding various parts of an attribute name. The parts are in the form of string fragments meaning they are not null terminated. The user must be careful to use the value and length when working with them.

Returns:

Completion status of operation:

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

5.5.1.41 EXTERNXML int rtXmlpDecXSIAttr (OSCTXT * *pctxt*)

This function decodes XSI (XML Schema Instance) that may be present in any arbitrary XML element within a document.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

5.5.1.42 EXTERNXML int rtXmlpDecXSITypeAttr (OSCTXT * *pctxt*, const OSXMLNameFragments * *attrName*, const OSUTF8CHAR ** *ppAttrValue*)

This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type).

Parameters:

pctxt A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.

attrName A pointer to a structure holding various parts of an attribute name. The parts are in the form of string fragments meaning they are not null terminated. The user must be careful to use the value and length when working with them.

ppAttrValue A pointer to a pointer to a UTF8 character string to received the decoded XSI type name.

Returns:

Completion status of operation:

- 0 = success,
- 1 = OK, but attrName was not xsi:type (i.e. no attribute match)
- negative return value is error.

5.5.1.43 EXTERNXML void rtXmlpForceDecodeAsGroup (OSCTXT * *pctxt*)

Disable skipping of unknown elements in optional sequence tail.

Function used in outer types to break decode on first unknown element after decoding mandatory sequence part.

Parameters:

pctxt Pointer to context block structure.

5.5.1.44 EXTERNXML int rtXmlpGetAttributeCount (OSCTXT * *pctxt*)

This function returns number of attributes in last processed start tag.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

- zero or positive value is attributes number,
- negative return value is error.

5.5.1.45 EXTERNXML int rtXmlpGetAttributeID (const OSXMLStrFragment * *attrName*, OSINT16 *nsidx*, size_t *nAttr*, const OSXMLAttrDescr *attrNames*[], OSUINT32 *attrPresent*[])

This function finds an attribute in the descriptor table.

Parameters:

attrName A pointer to a structure holding various parts of an attribute name. The parts are in the form of string fragments meaning they are not null terminated. The user must be careful to use the value and length when working with them.

nsidx Namespace index:

- 0 = attribute is unqualified,
- positive value is user namespace from generated namespace table,
- negative value is predefined namespace like XSD instance and ect.

nAttr Number of descriptors in table.

attrNames Attributes descriptor table.

attrPresent Bit array to mark already decoded attributes. It is used to identify duplicate attributes.

Returns:

Completion status of operation:

- positive or zero return value is attribute index in descriptor table,
- negative return value is error.

5.5.1.46 EXTERNXML OSINT32 rtXmlpGetCurrentLevel (OSCTXT * *pctxt*)

This function returns current nesting level.

Parameters:

pctxt Pointer to context block structure.

Returns:

Current nesting level.

5.5.1.47 EXTERNXML int rtXmlpGetNextAllElemID (OSCTXT * *pctxt*, const OSXMLElemIDRec * *tab*, size_t *nrows*, const OSUINT8 * *pOrder*, OSUINT32 *nOrder*, OSUINT32 *maxOrder*, int *anyID*)

This function parses the next start tag and finds index of element name in descriptor table.

It used for decode "all" content model in strict mode.

Parameters:

pctxt Pointer to context block structure.

tab Elements descriptor table.

nrows Number of descriptors in table.

pOrder Pointer to array to receive elements order.

nOrder Last element's index in order array.

maxOrder Size of order array.

anyID Identifier of xsd:any element.

Returns:

Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

5.5.1.48 EXTERNXML int rtXmlpGetNextAllElemID16 (OSCTXT * *pctxt*, const OSXMLElemIDRec * *tab*, size_t *nrows*, const OSUINT16 * *pOrder*, OSUINT32 *nOrder*, OSUINT32 *maxOrder*, int *anyID*)

This function parses the next start tag and finds index of element name in descriptor table.

It used for decode "all" content model in strict mode. This variant used when xsd:all has above 256 elements.

Parameters:

- pctxt* Pointer to context block structure.
- tab* Elements descriptor table.
- nrows* Number of descriptors in table.
- pOrder* Pointer to array to receive elements order.
- nOrder* Last element's index in order array.
- maxOrder* Size of order array.
- anyID* Identifier of xsd:any element.

Returns:

Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

5.5.1.49 EXTERNXML int rtXmlpGetNextElem (OSCTXT * *pctxt*, OSXMLElemDescr * *pElem*, OSINT32 *level*)

This function parse the next element start tag.

Parameters:

- pctxt* Pointer to context block structure.
- pElem* Pointer to a structure to receive the decoded element descriptor.
- level* Nesting level of parsed start tag. When value equal -1 parsed next start tag.

Returns:

Completion status of operation:

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

5.5.1.50 EXTERNXML int rtXmlpGetNextElemID (OSCTXT * *pctxt*, const OSXMLElemIDRec * *tab*, size_t *nrows*, OSINT32 *level*, OSBOOL *continueParse*)

This function parses the next start tag and finds the index of the element name in the descriptor table.

Parameters:

- pctxt* Pointer to context block structure.
- tab* Elements descriptor table.
- nrows* Number of descriptors in table.
- level* Nesting level of parsed start tag. When value equal -1 function parse next start tag.
- continueParse* When value equals TRUE function skips unrecognized elements.

Returns:

Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

5.5.1.51 EXTERNXML int rtXmlpNextSeqElemID (OSCTXT * *pctxt*, const OSXMLElemIDRec * *tab*, const OSXMLGroupDesc * *pGroup*, int *curID*, int *lastMandatoryID*, OSBOOL *groupMode*)

This function parses the next start tag and find index of element name in descriptor table.

It is used to decode sequences in strict mode.

Parameters:

pctxt Pointer to context block structure.

tab Elements descriptor table.

pGroup Decode groups table.

curID Current decode group.

lastMandatoryID Identifier of last mandatory element.

groupMode This parameter must be setted to TRUE when decode groups or base types.

Returns:

Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

5.5.1.52 EXTERNXML int rtXmlpGetXmlnsAttrs (OSCTXT * *pctxt*, OSRTDList * *pNSAttrs*)

This function decodes namespace attributes from start tag and adds them to the given list.

Parameters:

pctxt Pointer to context block structure.

pNSAttrs A pointer to a linked list of OSXMLNamespace structures.

Returns:

Completion status of operation:

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

5.5.1.53 EXTERNXML int rtXmlpGetXSITypeAttr (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppAttrValue*, OSINT16 * *nsidx*, size_t * *pLocalOffs*)

This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type).

Parameters:

pctxt Pointer to context block structure.

ppAttrValue A pointer to a pointer to a UTF8 character string to received the decoded XSI type QName.

nsidx A pointer to OSINT16 value to received the decoded XSI type namespace index.

pLocalOffs A pointer to size_t value to received the local name offset in *ppAttrValue*. When *pLocalOffs* value equal NULL function return in *ppAttrValue* local name only.

Returns:

Completion status of operation:

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

5.5.1.54 EXTERNXML int rtXmlpGetXSITypeIndex (OSCTXT * *pctxt*, const OSXMLItemDescr *typetab*[], size_t *typetabsiz*)

This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type) and find type index in descriptor table.

Parameters:

pctxt Pointer to context block structure.

typetab XSI types descriptor table.

typetabsiz Number of descriptors in table.

Returns:

Completion status of operation:

- positive or zero value is type index,
- negative return value is error.

5.5.1.55 EXTERNXML OSBOOL rtXmlpHasAttributes (OSCTXT * *pctxt*)

This function checks accessibility of attributes.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

- TRUE = attributes are present and access is enabled,
- FALSE = attributes are absent or access is disabled.

5.5.1.56 EXTERNXML void rtXmlpHideAttributes (OSCTXT * *pctxt*)

Disable access to attributes.

Function used in derived types to disable repeated decode in base type.

Parameters:

pctxt Pointer to context block structure.

5.5.1.57 EXTERNXML OSBOOL rtXmlpIsDecodeAsGroup (OSCTXT * *pctxt*)

This function checks if "decode as group" mode was forced.

Parameters:

pctxt Pointer to context block structure.

Returns:

State of mode:

- TRUE = need decode as group,
- FALSE = normal sequence decoding.

5.5.1.58 EXTERNXML OSBOOL rtXmlpIsEmptyElement (OSCTXT * *pctxt*)

Check element content: empty or not.

Parameters:

pctxt Pointer to context block structure.

Returns:

Status of element content:

- TRUE = element is empty,
- FALSE = element has content.

5.5.1.59 EXTERNXML OSBOOL rtXmlpIsLastEventDone (OSCTXT * *pctxt*)

Check processing status of current tag.

Parameters:

pctxt Pointer to context block structure.

Returns:

Status of element content:

- TRUE = tag marked as processed,
- FALSE = tag will be processed again.

5.5.1.60 EXTERNXML OSBOOL rtXmlpListHasItem (OSCTXT * *pctxt*)

Check for end of decoded token list.

Parameters:

pctxt Pointer to context block structure.

Returns:

State of decoded list:

- TRUE = list is not finished,
- FALSE = all tokens was decoded.

5.5.1.61 EXTERNXML int rtXmlpLookupXSITypeIndex (OSCTXT * *pctxt*, const OSUTF8CHAR * *pXsiType*, OSINT16 *xsiTypeIdx*, const OSXMLItemDescr *typetab*[], size_t *typetabsiz*)

This function find index of XSI (XML Schema Instance) type in descriptor table.

Parameters:

- pctxt* Pointer to context block structure.
- pXsiType* A pointer to XSI type name.
- xsiTypeIdx* A XSI type namespace index.
- typetab* XSI types descriptor table.
- typetabsiz* Number of descriptors in table.

Returns:

Completion status of operation:

- positive or zero value is type index,
- negative return value is error.

5.5.1.62 EXTERNXML int rtXmlpMarkLastEventActive (OSCTXT * *pctxt*)

This function marks current tag as unprocessed.

This will cause the element to be processed again in the next pull-parser function.

Parameters:

- pctxt* Pointer to context block structure.

Returns:

Completion status of operation:

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

5.5.1.63 EXTERNXML void rtXmlpMarkPos (OSCTXT * *pctxt*)

Save current decode position.

Parameters:

- pctxt* Pointer to context block structure.

5.5.1.64 EXTERNXML int rtXmlpMatchEndTag (OSCTXT * *pctxt*, OSINT32 *level*)

This function parse next end tag that matches with given name.

Parameters:

- pctxt* Pointer to context block structure.

level Nesting level of parsed start tag. When value equal -1 function parse next end tag with current level.

Returns:

Completion status of operation:

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

5.5.1.65 EXTERNXML int rtXmlpMatchStartTag (OSCTXT * *pctxt*, const OSUTF8CHAR * *elemLocalName*, OSINT16 *nsidx*)

This function parses the next start tag that matches with given name.

Parameters:

pctxt Pointer to context block structure.

elemLocalName Name of parsed element.

nsidx Namespace index:

- 0 = attribute is unqualified,
- positive value is user namespace from generated namespace table,
- negative value is predefined namespace like XSD instance and ect.

Returns:

Completion status of operation:

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

5.5.1.66 EXTERNXML OSBOOL rtXmlpNeedDecodeAttributes (OSCTXT * *pctxt*)

This function checks if attributes were previously decoded.

Parameters:

pctxt Pointer to context block structure.

Returns:

State of attributes:

- TRUE = attributes was not decoded,
- FALSE = attributes had been decoded.

5.5.1.67 EXTERNXML void rtXmlpResetMarkedPos (OSCTXT * *pctxt*)

Reset saved decode position.

Parameters:

pctxt Pointer to context block structure.

5.5.1.68 EXTERNXML void rtXmlpRewindToMarkedPos (OSCTXT * *pctxt*)

Rewind to saved decode position.

Parameters:

pctxt Pointer to context block structure.

5.5.1.69 EXTERNXML int rtXmlpSelectAttribute (OSCTXT * *pctxt*, OSXMLNameFragments * *pAttr*, OSINT16 * *nsidx*, size_t *index*)

This function selects attribute to decode.

Parameters:

pctxt Pointer to context block structure.

pAttr Pointer to structure to receive the various parts of an attribute name.

nsidx Pointer to value to receive namespace index:

- 0 = attribute is unqualified,
- positive value is user namespace from generated namespace table,
- negative value is predefined namespace like XSD instance and ect.

index Index of selected attribute.

Returns:

Completion status of operation:

- positive or zero return value is attribute index in descriptor table,
- negative return value is error.

5.5.1.70 EXTERNXML void rtXmlpSetListMode (OSCTXT * *pctxt*)

Sets list mode.

Attribute value or element content is decoded by tokens.

Parameters:

pctxt Pointer to context block structure.

5.5.1.71 EXTERNXML OSBOOL rtXmlpSetMixedContentMode (OSCTXT * *pctxt*, OSBOOL *mixedContentMode*)

Sets mixed content mode.

Parameters:

pctxt Pointer to context block structure.

mixedContentMode Enable/disable mixed mode.

Returns:

Previously state of mixed mode.

5.5.1.72 EXTERNXML void rtXmlpSetNamespaceTable (OSCTXT * *pctxt*, const OSUTF8CHAR * *namespaceTable*[], size_t *nmNamespaces*)

Sets user namespace table.

Parameters:

- pctxt* Pointer to context block structure.
- namespaceTable* Array of namespace URI strings.
- nmNamespaces* Number of namespaces in table.

5.5.1.73 EXTERNXML void rtXmlpSetWhiteSpaceMode (OSCTXT * *pctxt*, OSXMLWhiteSpaceMode *whiteSpaceMode*)

Sets the whitespace treatment mode.

This mode affects the content and attribute values reading. For example, if OSXMLWSM_COLLAPSE mode is set then all spaces in returned data will be already collapsed.

Parameters:

- pctxt* Pointer to context block structure.
- whiteSpaceMode* White space mode.

Returns:

Previously set whitespace mode.

5.6 XML run-time error status codes.

This is a list of status codes that can be returned by XML run-time functions and generated code.

Defines

- #define `XML_OK_EOB` 0x7fffffff
End of block marker.
- #define `XML_OK_FRAG` `XML_OK_EOB`
Maintained for backward compatibility.
- #define `XML_E_BASE` -200
Error base.
- #define `XML_E_GENERR` (`XML_E_BASE`)
General error.
- #define `XML_E_INVSYMBOL` (`XML_E_BASE-1`)
An invalid XML symbol (character) was detected at the given point in the parse stream.
- #define `XML_E_TAGMISMATCH` (`XML_E_BASE-2`)
Start/end tag mismatch.
- #define `XML_E_DUPLATTR` (`XML_E_BASE-3`)
Duplicate attribute found.
- #define `XML_E_BADCHARREF` (`XML_E_BASE-4`)
Bad character reference found.
- #define `XML_E_INVMODE` (`XML_E_BASE-5`)
Invalid mode.
- #define `XML_E_UNEXPEOF` (`XML_E_BASE-6`)
Unexpected end of file (document).
- #define `XML_E_NOMATCH` (`XML_E_BASE-7`)
Current tag is not matched to specified one.
- #define `XML_E_ELEMMISRQ` (`XML_E_BASE-8`)
Missing required element.
- #define `XML_E_ELEMSISRQ` (`XML_E_BASE-9`)
Missing required elements.
- #define `XML_E_TOOFWELEMS` (`XML_E_BASE-10`)
The number of elements in a repeating collection was less than the number of elements specified in the XSD minOccurs facet for this type or element.
- #define `XML_E_UNEXPSTARTTAG` (`XML_E_BASE-11`)

Unexpected start tag.

- #define XML_E_UNEXPENDTAG (XML_E_BASE-12)

Unexpected end tag.

- #define XML_E_IDNOTFOU (XML_E_BASE-13)

Expected identifier not found.

- #define XML_E_INVTYPEINFO (XML_E_BASE-14)

Unknown xsi:type.

- #define XML_E_NSURINOTFOU (XML_E_BASE-15)

Namespace URI not defined for given prefix.

5.6.1 Detailed Description

This is a list of status codes that can be returned by XML run-time functions and generated code.

In many cases, additional information and parameters for the different errors are stored in the context structure at the time the error is raised. This additional information can be output using the `rtxErrPrint` or `rtxErrLogUsingCB` run-time functions.

5.6.2 Define Documentation

5.6.2.1 #define XML_E_BASE -200

Error base.

XML specific errors start at this base number to distinguish them from common and other error types.

Definition at line 55 of file `rtXmlErrCodes.h`.

5.6.2.2 #define XML_E_ELEMMISRQ (XML_E_BASE-8)

Missing required element.

This status code is returned by the decoder when the decoder knows exactly which element is absent.

Definition at line 116 of file `rtXmlErrCodes.h`.

5.6.2.3 #define XML_E_ELEMSISRQ (XML_E_BASE-9)

Missing required elements.

This status code is returned by the decoder when the number of elements decoded for a given content model group is less than the required number of elements as specified in the schema.

Definition at line 123 of file `rtXmlErrCodes.h`.

5.6.2.4 #define XML_E_NOMATCH (XML_E_BASE-7)

Current tag is not matched to specified one.

Informational code.

Definition at line 110 of file rtXmlErrCodes.h.

5.6.2.5 #define XML_E_NSURINOTFOU (XML_E_BASE-15)

Namespace URI not defined for given prefix.

A namespace URI was not defined using an xmlns attribute for the given prefix.

Definition at line 161 of file rtXmlErrCodes.h.

5.6.2.6 #define XML_E_TAGMISMATCH (XML_E_BASE-2)

Start/end tag mismatch.

The parsed end tag does not match the start tag that was parsed earlier at this level. Indicates document is not well-formed.

Definition at line 85 of file rtXmlErrCodes.h.

Chapter 6

ASN1C Data Structure Documentation

6.1 OSDecimalFmt Struct Reference

Data Fields

- OSINT8 [totalDigits](#)
- OSINT8 [fractionDigits](#)
- OSINT8 [fractionMinDigits](#)
- OSINT8 [integerMaxDigits](#)
- OSINT8 [integerMinDigits](#)
- OSBOOL [signPresent](#)
- OSBOOL [pointPresent](#)
- OSUINT8 [nPatterns](#)
- const char *const * [patterns](#)

6.1.1 Detailed Description

Definition at line 205 of file osrxml.h.

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

- [osrxml.h](#)

6.2 OSDoubleFmt Struct Reference

Data Fields

- OSINT8 [totalDigits](#)
- OSINT8 [fractionDigits](#)
- OSINT8 [fractionMinDigits](#)
- OSINT8 [integerMaxDigits](#)
- OSINT8 [integerMinDigits](#)
- OSINT8 [expSymbol](#)
- OSINT16 [expMinValue](#)
- OSINT16 [expMaxValue](#)
- OSINT8 [expDigits](#)
- OSBOOL [signPresent](#)
- OSBOOL [pointPresent](#)
- OSBOOL [expPresent](#)
- OSBOOL [expSignPresent](#)

6.2.1 Detailed Description

Definition at line 226 of file [osrxml.h](#).

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

- [osrxml.h](#)

6.3 OSIntegerFmt Struct Reference

Data Fields

- OSINT8 [integerMaxDigits](#)
- OSBOOL [signPresent](#)

6.3.1 Detailed Description

Definition at line 198 of file osrxml.h.

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

- [osrxml.h](#)

6.4 OSXMLCtxtInfo Struct Reference

Data Fields

- OSFreeCtxtAppInfoPtr [pFreeFunc](#)
- OSResetCtxtAppInfoPtr [pResetFunc](#)
- OSUTF8CHAR * [schemaLocation](#)
- OSUTF8CHAR * [noNSSchemaLoc](#)
- OSUTF8CHAR * [xsiTypeAttr](#)
- OSXMLEncoding [encoding](#)
- OSRTDList [namespaceList](#)
- OSRTDList [encodedNSList](#)
- OSRTDList [sortedAttrList](#)
- OSXMLNSPfxLinkStack [nsPfxLinkStack](#)
- OSXMLNSURITable [nsURITable](#)
- OSRTMEMBUF [memBuf](#)
- OSINT32 [mSaxLevel](#)
- OSINT32 [mSkipLevel](#)
- OSUINT32 [maxSaxErrors](#)
- OSUINT32 [errorsCnt](#)
- OSUINT8 [indent](#)
- OSBOOL [mbCdataProcessed](#)
- char [indentChar](#)
- OSUINT8 [soapVersion](#)
- OSXMLFacets [facets](#)
- const OSUTF8CHAR * [encodingStr](#)
- OSXMLBOM [byteOrderMark](#)
- OSXMLReader * [pXmlPPReader](#)
- OSRTBuffer [savedBuffer](#)
- OSRTFLAGS [savedFlags](#)
- OSOCTET * [attrsBuff](#)
- size_t [attrsBuffSize](#)
- size_t [attrStartPos](#)

6.4.1 Detailed Description

Definition at line 149 of file [osrxml.h](#).

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

- [osrxml.h](#)

6.5 OSXMLElemIDRec Struct Reference

Data Fields

- [OSXMLElemDescr descr](#)
- OSUINT16 [id](#)

6.5.1 Detailed Description

Definition at line 111 of file osrxml.h.

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

- [osrxml.h](#)

6.6 OSXMLFacets Struct Reference

Data Fields

- int [totalDigits](#)
- int [fractionDigits](#)

6.6.1 Detailed Description

Definition at line 87 of file osrxml.h.

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

- [osrxml.h](#)

6.7 OSXMLGroupDesc Struct Reference

Data Fields

- int [row](#)
- int [num](#)
- int [anyCase](#)

6.7.1 Detailed Description

Definition at line 116 of file osrxml.h.

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

- [osrxml.h](#)

6.8 OSXMLItemDescr Struct Reference

Data Fields

- [OSXMLStrFragment localName](#)
- [OSINT16 nsidx](#)

6.8.1 Detailed Description

Definition at line 103 of file osrxml.h.

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

- [osrxml.h](#)

6.9 OSXMLNameFragments Struct Reference

Data Fields

- [OSXMLStrFragment mQName](#)
- [OSXMLStrFragment mLocalName](#)
- [OSXMLStrFragment mPrefix](#)

6.9.1 Detailed Description

Definition at line 97 of file osrxml.h.

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

- [osrxml.h](#)

6.10 OSXMLQName Struct Reference

Data Fields

- const OSUTF8CHAR * [nsPrefix](#)
- const OSUTF8CHAR * [ncName](#)

6.10.1 Detailed Description

Definition at line 191 of file osrxml.h.

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

- [osrxml.h](#)

6.11 OSXMLSortedAttrOffset Struct Reference

Data Fields

- [size_t offset](#)
- [size_t length](#)
- [size_t prefixLength](#)
- [size_t nameLength](#)

6.11.1 Detailed Description

Definition at line 255 of file osrxml.h.

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

- [osrxml.h](#)

6.12 OSXMLStrFragment Struct Reference

Data Fields

- const OSUTF8CHAR * [value](#)
- size_t [length](#)

6.12.1 Detailed Description

Definition at line 92 of file osrxml.h.

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

- [osrxml.h](#)

6.13 OSXSDAnyType Struct Reference

Data Fields

- OSXMLSTRING [value](#)
- OSRTDList [attrs](#)

6.13.1 Detailed Description

Definition at line 122 of file osrxml.h.

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

- [osrxml.h](#)

Chapter 7

ASN1C File Documentation

7.1 osrtxml.h File Reference

XML low-level C encode/decode functions.

```
#include "rtxsrc/rtxCommon.h"  
#include "rtxmlsrc/rtSaxDefs.h"  
#include "rtxsrc/rtxDList.h"  
#include "rtxsrc/rtxMemBuf.h"  
#include "rtxmlsrc/rtXmlExternDefs.h"  
#include "rtxmlsrc/rtXmlErrCodes.h"  
#include "rtxmlsrc/rtXmlNamespace.h"
```

Data Structures

- struct [OSXMLFacets](#)
- struct [OSXMLStrFragment](#)
- struct [OSXMLNameFragments](#)
- struct [OSXMLItemDescr](#)
- struct [OSXMLElemIDRec](#)
- struct [OSXMLGroupDesc](#)
- struct [OSXSDAnyType](#)
- struct [OSXMLCtxtInfo](#)
- struct [OSXMLQName](#)
- struct [OSIntegerFmt](#)
- struct [OSDecimalFmt](#)
- struct [OSDoubleFmt](#)
- struct [OSXMLSortedAttrOffset](#)

Defines

- #define [OSXMLNS12](#)
- #define [OSUPCASE](#) 0x00008000

- #define **OSTERMSTART** 0x00004000
- #define **OEMPTYELEM** 0x00002000
- #define **OSQUALATTR** 0x00001000
- #define **OSXMLFRAG** 0x00000800
- #define **OSXMLNSSET** 0x00000400
- #define **OSXMLC14N** 0x00000200
- #define **OSXSIATTR** 0x00000100
- #define **OSXMLFRAGSEQUAL**(frag1, frag2) (frag1.length==frag2.length && !memcmp(frag1.value,frag2.value,frag1.length))
- #define **OSXMLQNAMEEQUALS**(xnamefrag, qnametext)
- #define **OSXMLSETUTF8DECPTR**(pctxt, str)
- #define **IS_XMLNSATTR**(name)
- #define **IS_XSIATTR**(name)
- #define **OSXMLINDENT** 3
- #define **rtXmlErrAddStrParm** rtxErrAddStrParm
- #define **rtXmlFinalizeMemBuf**(pMemBuf)
- #define **rtXmlGetEncBufPtr**(pctxt) (pctxt) → buffer.data
This macro returns the start address of the encoded XML message.
- #define **rtXmlGetEncBufLen**(pctxt) (pctxt) → buffer.byteIndex
This macro returns the length of the encoded XML message.

Typedefs

- typedef **OSXMLItemDescr** **OSXMLAttrDescr**
- typedef **OSXMLItemDescr** **OSXMLElemDescr**

Enumerations

- enum **OSXMLEncoding** { **OSXMLUTF8**, **OSXMLUTF16** }
- enum **OSXMLSOAPMsgType** { **OSSOAPNONE**, **OSSOAPHEADER**, **OSSOAPBODY**, **OSSOAPFAULT** }
- enum **OSXMLBOM** {
OSXMLBOM_NO_BOM, **OSXMLBOM_UTF32_BE**, **OSXMLBOM_UTF32_LE**, **OSXMLBOM_UTF16_BE**,
OSXMLBOM_UTF16_LE, **OSXMLBOM_UTF8** }
- enum **OSXMLState** {
OSXMLINIT, **OSXMLHEADER**, **OSXMLSTART**, **OSXMLATTR**,
OSXMLDATA, **OSXMLEND** }
- enum **OSXMLWhiteSpaceMode** { **OSXMLWSM_PRESERVE** = 0, **OSXMLWSM_REPLACE**,
OSXMLWSM_COLLAPSE }
Whitespace treatment options.

Functions

- EXTERNXML int [rtXmlInitContext](#) (OSCTXT *pctxt)
This function initializes a context variable for XML encoding or decoding.
- EXTERNXML int [rtXmlInitCtxtAppInfo](#) (OSCTXT *pctxt)
This function initializes the XML application info section of the given context.
- EXTERNXML int [rtXmlCreateFileInputSource](#) (OSCTXT *pctxt, const char *filepath)
This function creates an XML document file input source.
- EXTERNXML OSBOOL [rtXmlCmpQName](#) (const OSUTF8CHAR *qname1, const OSUTF8CHAR *name2, const OSUTF8CHAR *nsPrefix2)
- EXTERNXML int [rtXmlGetBase64StrDecodedLen](#) (const OSUTF8CHAR *inpdata, size_t srcDataSize, size_t *pNumOcts, size_t *pSrcDataLen)
- EXTERNXML int [rtXmlDecBase64Binary](#) (OSRTMEMBUF *pMemBuf, const OSUTF8CHAR *inpdata, int length)
This function decodes the contents of a Base64-encoded binary data type into a memory buffer.
- EXTERNXML int [rtXmlDecBase64Str](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)
This function decodes a contents of a Base64-encode binary string into a static memory structure.
- EXTERNXML int [rtXmlDecBase64StrValue](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, size_t bufSize, size_t srcDataLen)
This function decodes a contents of a Base64-encode binary string into the specified octet array.
- EXTERNXML int [rtXmlDecBigInt](#) (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)
This function will decode a variable of the XSD integer type.
- EXTERNXML int [rtXmlDecBool](#) (OSCTXT *pctxt, OSBOOL *pvalue)
This function decodes a variable of the boolean type.
- EXTERNXML int [rtXmlDecDate](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'date' type.
- EXTERNXML int [rtXmlDecTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'time' type.
- EXTERNXML int [rtXmlDecDateTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'dateTime' type.
- EXTERNXML int [rtXmlDecDecimal](#) (OSCTXT *pctxt, OSREAL *pvalue)
This function decodes the contents of a decimal data type.
- EXTERNXML int [rtXmlDecDouble](#) (OSCTXT *pctxt, OSREAL *pvalue)
This function decodes the contents of a float or double data type.
- EXTERNXML int [rtXmlDecDynBase64Str](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a contents of a Base64-encode binary string.

- EXTERNXML int [rtXmlDecDynHexStr](#) (OSCTXT *pctx, OSDynOctStr *pvalue)
This function decodes a contents of a hexBinary string.
- EXTERNXML int [rtXmlDecUTF8Str](#) (OSCTXT *pctx, OSUTF8CHAR *outdata, size_t max_len)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlDecDynUTF8Str](#) (OSCTXT *pctx, const OSUTF8CHAR **outdata)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlDecHexBinary](#) (OSRTMEMBUF *pMemBuf, const OSUTF8CHAR *inpdata, int length)
This function decodes the contents of a hex-encoded binary data type into a memory buffer.
- EXTERNXML int [rtXmlDecHexStr](#) (OSCTXT *pctx, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)
This function decodes the contents of a hexBinary string into a static memory structure.
- EXTERNXML int [rtXmlDecHexStrValue](#) (OSCTXT *pctx, const OSUTF8CHAR *const inpdata, size_t nbytes, OSOCTET *pvalue, OSUINT32 *pnbits, OSINT32 bufsize)
- EXTERNXML int [rtXmlDecGYear](#) (OSCTXT *pctx, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gYear' type.
- EXTERNXML int [rtXmlDecGYearMonth](#) (OSCTXT *pctx, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gYearMonth' type.
- EXTERNXML int [rtXmlDecGMonth](#) (OSCTXT *pctx, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gMonth' type.
- EXTERNXML int [rtXmlDecGMonthDay](#) (OSCTXT *pctx, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gMonthDay' type.
- EXTERNXML int [rtXmlDecGDay](#) (OSCTXT *pctx, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gDay' type.
- EXTERNXML int [rtXmlDecInt](#) (OSCTXT *pctx, OSINT32 *pvalue)
This function decodes the contents of a 32-bit integer data type.
- EXTERNXML int [rtXmlDecInt8](#) (OSCTXT *pctx, OSINT8 *pvalue)
This function decodes the contents of an 8-bit integer data type (i.e.
- EXTERNXML int [rtXmlDecInt16](#) (OSCTXT *pctx, OSINT16 *pvalue)
This function decodes the contents of a 16-bit integer data type.
- EXTERNXML int [rtXmlDecInt64](#) (OSCTXT *pctx, OSINT64 *pvalue)
This function decodes the contents of a 64-bit integer data type.
- EXTERNXML int [rtXmlDecUInt](#) (OSCTXT *pctx, OSUINT32 *pvalue)
This function decodes the contents of an unsigned 32-bit integer data type.
- EXTERNXML int [rtXmlDecUInt8](#) (OSCTXT *pctx, OSUINT8 *pvalue)

This function decodes the contents of an unsigned 8-bit integer data type (i.e.

- EXTERNXML int [rtXmlDecUInt16](#) (OSCTXT *pctxt, OSUINT16 *pvalue)

This function decodes the contents of an unsigned 16-bit integer data type.

- EXTERNXML int [rtXmlDecUInt64](#) (OSCTXT *pctxt, OSUINT64 *pvalue)

This function decodes the contents of an unsigned 64-bit integer data type.

- EXTERNXML int [rtXmlDecNSAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *attrName, const OSUTF8CHAR *attrValue, OSRTDList *pNSAttrs, const OSUTF8CHAR *nsTable[], OSUINT32 nsTableRowCount)

This function decodes an XML namespace attribute (xmlns).

- EXTERNXML const OSUTF8CHAR * [rtXmlDecQName](#) (OSCTXT *pctxt, const OSUTF8CHAR *qname, const OSUTF8CHAR **prefix)

This function decodes an XML qualified name string (QName) type.

- EXTERNXML int [rtXmlDecXSIAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *attrName, const OSUTF8CHAR *attrValue)

This function decodes XML schema instance (XSI) attribute.

- EXTERNXML int [rtXmlDecXSIAttrs](#) (OSCTXT *pctxt, const OSUTF8CHAR *const *attrs, const char *typeName)

This function decodes XML schema instance (XSI) attributes.

- EXTERNXML int [rtXmlDecXmlStr](#) (OSCTXT *pctxt, OSXMLSTRING *outdata)

This function decodes the contents of an XML string data type.

- EXTERNXML int [rtXmlParseElementName](#) (OSCTXT *pctxt, OSUTF8CHAR **ppName)

This function parses the initial tag from an XML message.

- EXTERNXML int [rtXmlParseElemQName](#) (OSCTXT *pctxt, OSXMLQName *pQName)

This function parses the initial tag from an XML message.

- EXTERNXML int [rtXmlEncAny](#) (OSCTXT *pctxt, OSXMLSTRING *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a variable of the XSD any type.

- EXTERNXML int [rtXmlEncAnyStr](#) (OSCTXT *pctxt, const OSUTF8CHAR *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

- EXTERNXML int [rtXmlEncAnyTypeValue](#) (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

This function encodes a variable of the XSD anyType type.

- EXTERNXML int [rtXmlEncAnyAttr](#) (OSCTXT *pctxt, OSRTDList *pAnyAttrList)

This function encodes a list of OSAnyAttr attributes in which the name and value are given as a UTF-8 string.

- EXTERNXML int [rtXmlEncBase64Binary](#) (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a variable of the XSD base64Binary type.

- EXTERNXML int [rtXmlEncBase64BinaryAttr](#) (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *attrName, size_t attrNameLen)

This function encodes a variable of the XSD base64Binary type as an attribute.

- EXTERNXML int [rtXmlEncBase64StrValue](#) (OSCTXT *pctx, OSUINT32 noctx, const OSOCTET *value)
This function encodes a variable of the XSD base64Binary type.
- EXTERNXML int [rtXmlEncBigInt](#) (OSCTXT *pctx, const OSUTF8CHAR *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncBigIntAttr](#) (OSCTXT *pctx, const OSUTF8CHAR *value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes an XSD integer attribute value.
- EXTERNXML int [rtXmlEncBigIntValue](#) (OSCTXT *pctx, const OSUTF8CHAR *value)
This function encodes an XSD integer attribute value.
- EXTERNXML int [rtXmlEncBitString](#) (OSCTXT *pctx, OSUINT32 nbits, const OSOCTET *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the ASN.1 BIT STRING type.
- EXTERNXML int [rtXmlEncBinStrValue](#) (OSCTXT *pctx, OSUINT32 nbits, const OSOCTET *data)
This function encodes a binary string value as a sequence of '1's and '0's.
- EXTERNXML int [rtXmlEncBool](#) (OSCTXT *pctx, OSBOOL value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD boolean type.
- EXTERNXML int [rtXmlEncBoolValue](#) (OSCTXT *pctx, OSBOOL value)
This function encodes a variable of the XSD boolean type.
- EXTERNXML int [rtXmlEncBoolAttr](#) (OSCTXT *pctx, OSBOOL value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes an XSD boolean attribute value.
- EXTERNXML int [rtXmlEncDate](#) (OSCTXT *pctx, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD 'date' type as a string.
- EXTERNXML int [rtXmlEncDateValue](#) (OSCTXT *pctx, const OSXSDDateTime *pvalue)
This function encodes a variable of the XSD 'date' type as a string.
- EXTERNXML int [rtXmlEncTime](#) (OSCTXT *pctx, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD 'time' type as a string.
- EXTERNXML int [rtXmlEncTimeValue](#) (OSCTXT *pctx, const OSXSDDateTime *pvalue)
This function encodes a variable of the XSD 'time' type as a string.
- EXTERNXML int [rtXmlEncDateTime](#) (OSCTXT *pctx, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a numeric date/time value into an XML string representation.

- EXTERNXML int [rtXmlEncDateTimeValue](#) (OSCTXT *pctx, const OSXSDDateTime *pvalue)
This function encodes a numeric date/time value into an XML string representation.
- EXTERNXML int [rtXmlEncDecimal](#) (OSCTXT *pctx, OSREAL value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const [OSDecimalFmt](#) *pFmtSpec)
This function encodes a variable of the XSD decimal type.
- EXTERNXML int [rtXmlEncDecimalAttr](#) (OSCTXT *pctx, OSREAL value, const OSUTF8CHAR *attrName, size_t attrNameLen, const [OSDecimalFmt](#) *pFmtSpec)
This function encodes a variable of the XSD decimal type as an attribute.
- EXTERNXML int [rtXmlEncDecimalValue](#) (OSCTXT *pctx, OSREAL value, const [OSDecimalFmt](#) *pFmtSpec, char *pDestBuf, size_t destBufSize)
This function encodes a value of the XSD decimal type.
- EXTERNXML int [rtXmlEncDouble](#) (OSCTXT *pctx, OSREAL value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const [OSDoubleFmt](#) *pFmtSpec)
This function encodes a variable of the XSD double type.
- EXTERNXML int [rtXmlEncDoubleAttr](#) (OSCTXT *pctx, OSREAL value, const OSUTF8CHAR *attrName, size_t attrNameLen, const [OSDoubleFmt](#) *pFmtSpec)
This function encodes a variable of the XSD double type as an attribute.
- EXTERNXML int [rtXmlEncDoubleValue](#) (OSCTXT *pctx, OSREAL value, const [OSDoubleFmt](#) *pFmtSpec, int defaultPrecision)
This function encodes a value of the XSD double or float type.
- EXTERNXML int [rtXmlEncEmptyElement](#) (OSCTXT *pctx, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, OSRTDList *pNSAttrs, OSBOOL terminate)
This function encodes an empty element tag value (<elemName/>).
- EXTERNXML int [rtXmlEncEndDocument](#) (OSCTXT *pctx)
This function adds trailer information and a null terminator at the end of the XML document being encoded.
- EXTERNXML int [rtXmlEncEndElement](#) (OSCTXT *pctx, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes an end element tag value (</elemName>).
- EXTERNXML int [rtXmlEncEndSoapEnv](#) (OSCTXT *pctx)
This function encodes a SOAP envelope end element tag (<SOAP-ENV:Envelope/>).
- EXTERNXML int [rtXmlEncEndSoapElems](#) (OSCTXT *pctx, [OSXMLSOAPMsgType](#) msgtype)
This function encodes SOAP end element tags.
- EXTERNXML int [rtXmlEncFloat](#) (OSCTXT *pctx, OSREAL value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const [OSDoubleFmt](#) *pFmtSpec)
This function encodes a variable of the XSD float type.
- EXTERNXML int [rtXmlEncFloatAttr](#) (OSCTXT *pctx, OSREAL value, const OSUTF8CHAR *attrName, size_t attrNameLen, const [OSDoubleFmt](#) *pFmtSpec)

This function encodes a variable of the XSD float type as an attribute.

- EXTERNXML int **rtXmlEncGYear** (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gYear element into an XML string representation.

- EXTERNXML int **rtXmlEncGYearMonth** (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gYearMonth element into an XML string representation.

- EXTERNXML int **rtXmlEncGMonth** (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gMonth element into an XML string representation.

- EXTERNXML int **rtXmlEncGMonthDay** (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gMonthDay element into an XML string representation.

- EXTERNXML int **rtXmlEncGDay** (OSCTXT *pctxt, const OSXSDDateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a numeric gDay element into an XML string representation.

- EXTERNXML int **rtXmlEncGYearValue** (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

This function encodes a numeric gYear value into an XML string representation.

- EXTERNXML int **rtXmlEncGYearMonthValue** (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

This function encodes a numeric gYearMonth value into an XML string representation.

- EXTERNXML int **rtXmlEncGMonthValue** (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

This function encodes a numeric gMonth value into an XML string representation.

- EXTERNXML int **rtXmlEncGMonthDayValue** (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

This function encodes a numeric gMonthDay value into an XML string representation.

- EXTERNXML int **rtXmlEncGDayValue** (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

This function encodes a numeric gDay value into an XML string representation.

- EXTERNXML int **rtXmlEncHexBinary** (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a variable of the XSD hexBinary type.

- EXTERNXML int **rtXmlEncHexBinaryAttr** (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *value, const OSUTF8CHAR *attrName, size_t attrNameLen)

This function encodes a variable of the XSD hexBinary type as an attribute.

- EXTERNXML int **rtXmlEncHexStrValue** (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *data)

This function encodes a variable of the XSD hexBinary type.

- EXTERNXML int **rtXmlEncIndent** (OSCTXT *pctxt)

This function adds indentation whitespace to the output stream.

- EXTERNXML int [rtXmlEncInt](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncIntValue](#) (OSCTXT *pctxt, OSINT32 value)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncIntAttr](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD integer type as an attribute (name="value").
- EXTERNXML int [rtXmlEncIntPattern](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const OSUTF8CHAR *pattern)
This function encodes a variable of the XSD integer type using a pattern to specify the format of the integer value.
- EXTERNXML int [rtXmlEncIntPatternValue](#) (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *pattern)
- EXTERNXML int [rtXmlEncInt64](#) (OSCTXT *pctxt, OSINT64 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncInt64Value](#) (OSCTXT *pctxt, OSINT64 value)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncInt64Attr](#) (OSCTXT *pctxt, OSINT64 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD integer type as an attribute (name="value").
- EXTERNXML int [rtXmlEncNamedBits](#) (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSUINT32 nbits, const OSOCTET *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the ASN.1 BIT STRING type.
- EXTERNXML int [rtXmlEncNamedBitsValue](#) (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSUINT32 nbits, const OSOCTET *pvalue)
- EXTERNXML int [rtXmlEncNSAttrs](#) (OSCTXT *pctxt, OSRTDList *pNSAttrs)
This function encodes namespace declaration attributes at the beginning of an XML document.
- EXTERNXML int [rtxPrintNSAttrs](#) (const char *name, const OSRTDList data)
This function prints a list of namespace attributes.
- EXTERNXML int [rtXmlEncReal10](#) (OSCTXT *pctxt, const OSUTF8CHAR *pvalue, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the ASN.1 REAL base 10 type.
- EXTERNXML int [rtXmlEncSoapArrayTypeAttr](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value, size_t itemCount)
This function encodes the special SOAP encoding attrType attribute which specifies the number and type of elements in a SOAP array.
- EXTERNXML int [rtXmlEncSoapArrayTypeAttr2](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const OSUTF8CHAR *value, size_t valueLen, size_t itemCount)

- EXTERNXML int [rtXmlEncStartDocument](#) (OSCTXT *pctxt)
This function encodes the XML header text at the beginning of an XML document.
- EXTERNXML int [rtXmlEncBOM](#) (OSCTXT *pctxt)
This function encodes the Unicode byte order mark header at the start of the document.
- EXTERNXML int [rtXmlEncStartElement](#) (OSCTXT *pctxt, const OSUTF8CHAR *elemName, OSXML-
Namespace *pNS, OSRTDList *pNSAttrs, OSBOOL terminate)
This function encodes a start element tag value (<elemName>).
- EXTERNXML int [rtXmlEncStartSoapEnv](#) (OSCTXT *pctxt)
This function encodes a SOAP envelope start element tag.
- EXTERNXML int [rtXmlEncStartSoapElems](#) (OSCTXT *pctxt, [OSXMLSOAPMsgType](#) msgtype)
This function encodes a SOAP envelope start element tag and an optional SOAP body or fault tag.
- EXTERNXML int [rtXmlEncString](#) (OSCTXT *pctxt, OSXMLSTRING *pxmlstr, const OSUTF8CHAR
*elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD string type.
- EXTERNXML int [rtXmlEncStringValue](#) (OSCTXT *pctxt, const OSUTF8CHAR *value)
This function encodes a variable of the XSD string type.
- EXTERNXML int [rtXmlEncStringValue2](#) (OSCTXT *pctxt, const OSUTF8CHAR *value, size_t valueLen)
This function encodes a variable of the XSD string type.
- EXTERNXML int [rtXmlEncTermStartElement](#) (OSCTXT *pctxt)
This function terminates a currently open XML start element by adding either a '>' or '>' (if empty) terminator.
- EXTERNXML int [rtXmlEncUnicodeStr](#) (OSCTXT *pctxt, const OSUNICHAR *value, OSUINT32 nchars,
const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a Unicode string value.
- EXTERNXML int [rtXmlEncUTF8Attr](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR
*value)
This function encodes an attribute in which the name and value are given as a null-terminated UTF-8 strings.
- EXTERNXML int [rtXmlEncUTF8Attr2](#) (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const
OSUTF8CHAR *value, size_t valueLen)
This function encodes an attribute in which the name and value are given as a UTF-8 strings with lengths.
- EXTERNXML int [rtXmlEncUTF8Str](#) (OSCTXT *pctxt, const OSUTF8CHAR *value, const OSUTF8CHAR
*elemName, OSXMLNamespace *pNS)
This function encodes a UTF-8 string value.
- EXTERNXML int [rtXmlEncUInt](#) (OSCTXT *pctxt, OSUINT32 value, const OSUTF8CHAR *elemName, OS-
XMLNamespace *pNS)
This function encodes a variable of the XSD unsigned integer type.
- EXTERNXML int [rtXmlEncUIntValue](#) (OSCTXT *pctxt, OSUINT32 value)
This function encodes a variable of the XSD unsigned integer type.

- EXTERNXML int [rtXmlEncUIntAttr](#) (OSCTXT *pctx, OSUINT32 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD unsigned integer type as an attribute (name="value").
- EXTERNXML int [rtXmlEncUInt64](#) (OSCTXT *pctx, OSUINT64 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncUInt64Value](#) (OSCTXT *pctx, OSUINT64 value)
This function encodes a variable of the XSD integer type.
- EXTERNXML int [rtXmlEncUInt64Attr](#) (OSCTXT *pctx, OSUINT64 value, const OSUTF8CHAR *attrName, size_t attrNameLen)
This function encodes a variable of the XSD integer type as an attribute (name="value").
- EXTERNXML int [rtXmlEncXSIAttrs](#) (OSCTXT *pctx, OSBOOL needXSI)
This function encodes XML schema instance (XSI) attributes at the beginning of an XML document.
- EXTERNXML int [rtXmlEncXSITypeAttr](#) (OSCTXT *pctx, const OSUTF8CHAR *value)
This function encodes an XML schema instance (XSI) type attribute value (xsi:type="value").
- EXTERNXML int [rtXmlFreeInputSource](#) (OSCTXT *pctx)
This function closes an input source that was previously created with one of the create input source functions such as 'rtXmlCreateFileInputSource'.
- EXTERNXML OSBOOL [rtXmlStrCmpAsc](#) (const OSUTF8CHAR *text1, const char *text2)
- EXTERNXML OSBOOL [rtXmlStrnCmpAsc](#) (const OSUTF8CHAR *text1, const char *text2, size_t len)
- EXTERNXML int [rtXmlSetEncBufPtr](#) (OSCTXT *pctx, OSOCTET *bufaddr, size_t bufsiz)
This function is used to set the internal buffer within the run-time library encoding context.
- EXTERNXML int [rtXmlGetIndent](#) (OSCTXT *pctx)
This function returns current XML output indent value.
- EXTERNXML OSBOOL [rtXmlGetWriteBOM](#) (OSCTXT *pctx)
This function returns whether the Unicode byte order mark will be encoded.
- EXTERNXML int [rtXmlGetIndentChar](#) (OSCTXT *pctx)
This function returns current XML output indent character value (default is space).
- EXTERNXML int [rtXmlPrepareContext](#) (OSCTXT *pctx)
This function prepares the context for another encode by setting the state back to OSXMLINIT and moving the buffer's cursor back to the beginning of the buffer.
- EXTERNXML int [rtXmlSetEncC14N](#) (OSCTXT *pctx, OSBOOL value)
This function sets the option to encode in C14N mode.
- EXTERNXML int [rtXmlSetEncXSINamespace](#) (OSCTXT *pctx, OSBOOL value)
This function sets a flag in the context that indicates the XSI namespace declaration (xmlns:xsi) should be added to the encoded XML instance.

- EXTERNXML int **rtXmlSetDigitsFacets** (OSCTXT *pctx, int totalDigits, int fractionDigits)
- EXTERNXML int **rtXmlSetEncDocHdr** (OSCTXT *pctx, OSBOOL value)
This function sets the option to add the XML document header (i.e.
- EXTERNXML int **rtXmlSetEncodingStr** (OSCTXT *pctx, const OSUTF8CHAR *encodingStr)
This function sets the XML output encoding to the given value.
- EXTERNXML int **rtXmlSetFormatting** (OSCTXT *pctx, OSBOOL doFormatting)
This function sets XML output formatting to the given value.
- EXTERNXML int **rtXmlSetIndent** (OSCTXT *pctx, OSUINT8 indent)
This function sets XML output indent to the given value.
- EXTERNXML int **rtXmlSetIndentChar** (OSCTXT *pctx, char indentChar)
This function sets XML output indent character to the given value.
- EXTERNXML void **rtXmlSetNamespacesSet** (OSCTXT *pctx, OSBOOL value)
This function sets the context 'namespaces are set' flag.
- EXTERNXML int **rtXmlSetNSPrefixLinks** (OSCTXT *pctx, OSRTDList *pNSAttrs)
This function sets namespace prefix/URI links in the namespace prefix stack in the context structure.
- EXTERNXML int **rtXmlSetSchemaLocation** (OSCTXT *pctx, const OSUTF8CHAR *schemaLocation)
This function sets the XML Schema Instance (xsi) schema location attribute to be added to an encoded document.
- EXTERNXML int **rtXmlSetNoNSSchemaLocation** (OSCTXT *pctx, const OSUTF8CHAR *schemaLocation)
This function sets the XML Schema Instance (xsi) no namespace schema location attribute to be added to an encoded document.
- EXTERNXML void **rtXmlSetSoapVersion** (OSCTXT *pctx, OSUINT8 version)
This function sets the SOAP version number.
- EXTERNXML int **rtXmlSetXSITypeAttr** (OSCTXT *pctx, const OSUTF8CHAR *xsiType)
This function sets the XML Schema Instance (xsi) type attribute value.
- EXTERNXML int **rtXmlSetWriteBOM** (OSCTXT *pctx, OSBOOL write)
This function sets whether the Unicode byte order mark is encoded.
- EXTERNXML int **rtXmlMatchHexStr** (OSCTXT *pctx, size_t minLength, size_t maxLength)
This function tests the context buffer for containing a correct hexadecimal string.
- EXTERNXML int **rtXmlMatchBase64Str** (OSCTXT *pctx, size_t minLength, size_t maxLength)
This function tests the context buffer for containing a correct base64 string.
- EXTERNXML int **rtXmlMatchDate** (OSCTXT *pctx)
This function tests the context buffer for containing a correct date string.
- EXTERNXML int **rtXmlMatchTime** (OSCTXT *pctx)
This function tests the context buffer for containing a correct time string.

- EXTERNXML int **rtXmlMatchDateTime** (OSCTXT *pctx)

This function tests the context buffer for containing a correct dateTime string.
- EXTERNXML int **rtXmlMatchGYear** (OSCTXT *pctx)

This function tests the context buffer for containing a correct gYear string.
- EXTERNXML int **rtXmlMatchGYearMonth** (OSCTXT *pctx)

This function tests the context buffer for containing a correct gYearMonth string.
- EXTERNXML int **rtXmlMatchGMonth** (OSCTXT *pctx)

This function tests the context buffer for containing a correct gMonth string.
- EXTERNXML int **rtXmlMatchGMonthDay** (OSCTXT *pctx)

This function tests the context buffer for containing a correct gMonthDay string.
- EXTERNXML int **rtXmlMatchGDay** (OSCTXT *pctx)

This function tests the context buffer for containing a correct gDay string.
- EXTERNXML OSUTF8CHAR * **rtXmlNewQName** (OSCTXT *pctx, const OSUTF8CHAR *localName, const OSUTF8CHAR *prefix)

This function creates a new QName given the localName and prefix parts.
- EXTERNXML OSBOOL **rtXmlCmpBase64Str** (OSUINT32 noctx1, const OSOCTET *data1, const OSUTF8CHAR *data2)

This function compares an array of octets to a base64 string.
- EXTERNXML OSBOOL **rtXmlCmpHexStr** (OSUINT32 noctx1, const OSOCTET *data1, const OSUTF8CHAR *data2)

This function compares an array of octets to a hex string.
- EXTERNXML OSBOOL **rtXmlCmpHexChar** (OSUTF8CHAR ch, OSOCTET hexval)
- EXTERNXML int **rtSaxGetAttributeID** (const OSUTF8CHAR *attrName, size_t nAttr, const OSUTF8CHAR *attrNames[], OSUINT32 attrPresent[])
- EXTERNXML const OSUTF8CHAR * **rtSaxGetAttrValue** (const OSUTF8CHAR *attrName, const OSUTF8CHAR *const *attrs)

This function looks up an attribute in the attribute array returned by SAX to the startElement function.
- EXTERNXML OSINT16 **rtSaxGetElemID** (OSINT16 *pState, OSINT16 prevElemIdx, const OSUTF8CHAR *localName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT16 *fstab, OSINT16 fstabRows, OSINT16 fstabCols)

This function looks up a sequence element name in the given element info array.
- EXTERNXML OSINT16 **rtSaxGetElemID8** (OSINT16 *pState, OSINT16 prevElemIdx, const OSUTF8CHAR *localName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT8 *fstab, OSINT16 fstabRows, OSINT16 fstabCols)

This function is a space optimized version of rtSaxGetElemID.
- EXTERNXML OSINT16 **rtSaxFindElemID** (OSINT16 *pState, OSINT16 prevElemIdx, const OSUTF8CHAR *localName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT16 *fstab, OSINT16 fstabRows, OSINT16 fstabCols)

- EXTERNXML OSINT16 **rtSaxFindElemID8** (OSINT16 *pState, OSINT16 prevElemIdx, const OSUTF8CHAR *localName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT8 *fstab, OSINT16 fstabRows, OSINT16 fstabCols)
- EXTERNXML OSBOOL **rtSaxHasXMLNSAttrs** (const OSUTF8CHAR *const *attrs)
This function checks if the given attribute list contains one or more XML namespace attributes (xmlns).
- EXTERNXML OSBOOL **rtSaxIsEmptyBuffer** (OSCTXT *pctxt)
This function checks if the buffer in the context is empty or not.
- EXTERNXML OSINT16 **rtSaxLookupElemID** (OSCTXT *pctxt, OSINT16 *pState, OSINT16 prevElemIdx, const OSUTF8CHAR *localName, const OSUTF8CHAR *qName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT16 *fstab, OSINT16 fstabRows, OSINT16 fstabCols)
- EXTERNXML OSINT16 **rtSaxLookupElemID8** (OSCTXT *pctxt, OSINT16 *pState, OSINT16 prevElemIdx, const OSUTF8CHAR *localName, const OSUTF8CHAR *qName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT8 *fstab, OSINT16 fstabRows, OSINT16 fstabCols)
- EXTERNXML int **rtSaxStrListParse** (OSCTXT *pctxt, OSRTMEMBUF *pMemBuf, OSRTDList *pvalue)
This function parses the list of strings.
- EXTERNXML int **rtSaxSortAttrs** (OSCTXT *pctxt, const OSUTF8CHAR *const *attrs, OSUINT16 **order)
This function sorts a SAX attribute list in ascending order based on attribute name.
- EXTERNXML int **rtSaxStrListMatch** (OSCTXT *pctxt)
This function matches the list of strings.
- EXTERNXML OSBOOL **rtSaxTestFinal** (OSINT16 state, OSINT16 currElemIdx, const int *fstab, int fstabRows, int fstabCols)
- EXTERNXML OSBOOL **rtSaxTestFinal8** (OSINT16 state, OSINT16 currElemIdx, const OSINT8 *fstab, int fstabRows, int fstabCols)
- EXTERNXML int **rtSaxSetSkipLevelToCurrent** (OSCTXT *pctxt, int stat)
- EXTERNXML OSUINT32 **rtSaxSetMaxErrors** (OSCTXT *pctxt, OSUINT32 maxErrors)
- EXTERNXML OSUINT32 **rtSaxGetMaxErrors** (OSCTXT *pctxt)
- EXTERNXML int **rtSaxTestAttributesPresent** (OSCTXT *pctxt, const OSUINT32 *attrPresent, const OSUINT32 *reqAttrMask, const OSUTF8CHAR *const *attrNames, size_t numOfAttrs, const char *parentTypeName)
- EXTERNXML OSBOOL **rtSaxIncErrors** (OSCTXT *pctxt)
- EXTERNXML int **rtSaxReportUnexpAttrs** (OSCTXT *pctxt, const OSUTF8CHAR *const *attrs, const char *typeName)
- EXTERNXML int **rtXmlWriteToFile** (OSCTXT *pctxt, const char *filename)
This function writes the encoded XML message stored in the context message buffer out to a file.
- EXTERNXML void **rtXmlTreatWhitespaces** (OSCTXT *pctxt, int whiteSpaceType)
- EXTERNXML void **rtErrXmlInit** (void)
- EXTERNXML int **rtXmlpDecAny** (OSCTXT *pctxt, const OSUTF8CHAR **pvalue)
This function decodes an arbitrary XML section of code as defined by the XSD any type (xsd:any).
- EXTERNXML int **rtXmlpDecAnyAttrStr** (OSCTXT *pctxt, const OSUTF8CHAR **ppAttrStr, size_t index)
This function decodes an any attribute string.
- EXTERNXML int **rtXmlpDecAnyElem** (OSCTXT *pctxt, const OSUTF8CHAR **pvalue)
This function decodes an arbitrary XML section of code as defined by the XSD any type (xsd:any).

- EXTERNXML int [rtXmlpDecBase64Str](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)
This function decodes a contents of a Base64-encode binary string into a static memory structure.
- EXTERNXML int [rtXmlpDecBigInt](#) (OSCTXT *pctxt, const OSUTF8CHAR **pvalue)
This function will decode a variable of the XSD integer type.
- EXTERNXML int [rtXmlpDecBitString](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnbits, OSUINT32 bufsize)
This function decodes a bit string value.
- EXTERNXML int [rtXmlpDecBool](#) (OSCTXT *pctxt, OSBOOL *pvalue)
This function decodes a variable of the boolean type.
- EXTERNXML int [rtXmlpDecDate](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'date' type.
- EXTERNXML int [rtXmlpDecDateTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'dateTime' type.
- EXTERNXML int [rtXmlpDecDecimal](#) (OSCTXT *pctxt, OSREAL *pvalue, int totalDigits, int fraction-Digits)
This function decodes the contents of a decimal data type.
- EXTERNXML int [rtXmlpDecDouble](#) (OSCTXT *pctxt, OSREAL *pvalue)
This function decodes the contents of a float or double data type.
- EXTERNXML int [rtXmlpDecDynBase64Str](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a contents of a Base64-encode binary string.
- EXTERNXML int [rtXmlpDecDynBitString](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a bit string value.
- EXTERNXML int [rtXmlpDecDynHexStr](#) (OSCTXT *pctxt, OSDynOctStr *pvalue)
This function decodes a contents of a hexBinary string.
- EXTERNXML int [rtXmlpDecDynUnicodeStr](#) (OSCTXT *pctxt, const OSUNICHAR **ppdata, OSUINT32 *pnchars)
This function decodes a Unicode string data type.
- EXTERNXML int [rtXmlpDecDynUTF8Str](#) (OSCTXT *pctxt, const OSUTF8CHAR **outdata)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlpDecUTF8Str](#) (OSCTXT *pctxt, OSUTF8CHAR *out, size_t max_len)
This function decodes the contents of a UTF-8 string data type.
- EXTERNXML int [rtXmlpDecGDay](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)
This function decodes a variable of the XSD 'gDay' type.
- EXTERNXML int [rtXmlpDecGMonth](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gMonth' type.

- EXTERNXML int [rtXmlpDecGMonthDay](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gMonthDay' type.

- EXTERNXML int [rtXmlpDecGYear](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gYear' type.

- EXTERNXML int [rtXmlpDecGYearMonth](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'gYearMonth' type.

- EXTERNXML int [rtXmlpDecHexStr](#) (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)

This function decodes the contents of a hexBinary string into a static memory structure.

- EXTERNXML int [rtXmlpDecInt](#) (OSCTXT *pctxt, OSINT32 *pvalue)

This function decodes the contents of a 32-bit integer data type.

- EXTERNXML int [rtXmlpDecInt8](#) (OSCTXT *pctxt, OSINT8 *pvalue)

This function decodes the contents of an 8-bit integer data type (i.e.

- EXTERNXML int [rtXmlpDecInt16](#) (OSCTXT *pctxt, OSINT16 *pvalue)

This function decodes the contents of a 16-bit integer data type.

- EXTERNXML int [rtXmlpDecInt64](#) (OSCTXT *pctxt, OSINT64 *pvalue)

This function decodes the contents of a 64-bit integer data type.

- EXTERNXML int [rtXmlpDecNamedBits](#) (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSOCTET *pvalue, OSUINT32 *pnbits, OSUINT32 bufsize)

This function decodes the contents of a named bit field.

- EXTERNXML int [rtXmlpDecStrList](#) (OSCTXT *pctxt, OSRTDList *plist)

This function decodes a list of space-separated tokens and returns each token as a separate item on the given list.

- EXTERNXML int [rtXmlpDecTime](#) (OSCTXT *pctxt, OSXSDDateTime *pvalue)

This function decodes a variable of the XSD 'time' type.

- EXTERNXML int [rtXmlpDecUInt](#) (OSCTXT *pctxt, OSUINT32 *pvalue)

This function decodes the contents of an unsigned 32-bit integer data type.

- EXTERNXML int [rtXmlpDecUInt8](#) (OSCTXT *pctxt, OSOCTET *pvalue)

This function decodes the contents of an unsigned 8-bit integer data type (i.e.

- EXTERNXML int [rtXmlpDecUInt16](#) (OSCTXT *pctxt, OSUINT16 *pvalue)

This function decodes the contents of an unsigned 16-bit integer data type.

- EXTERNXML int [rtXmlpDecUInt64](#) (OSCTXT *pctxt, OSUINT64 *pvalue)

This function decodes the contents of an unsigned 64-bit integer data type.

- EXTERNXML int [rtXmlpDecXmlStr](#) (OSCTXT *pctxt, OSXMLSTRING *outdata)

This function decodes the contents of an XML string data type.

- EXTERNXML int [rtXmlpDecXmlStrList](#) (OSCTXT *pctx, OSRTDList *plist)
This function decodes a list of space-separated tokens and returns each token as a separate item on the given list.
- EXTERNXML int [rtXmlpDecXSIAAttr](#) (OSCTXT *pctx, const [OSXMLNameFragments](#) *attrName)
This function decodes XSI (XML Schema Instance) and XML namespace attributes that may be present in any arbitrary XML element within a document.
- EXTERNXML int [rtXmlpDecXSITypeAttr](#) (OSCTXT *pctx, const [OSXMLNameFragments](#) *attrName, const OSUTF8CHAR **ppAttrValue)
This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type).
- EXTERNXML int [rtXmlpGetAttributeID](#) (const [OSXMLStrFragment](#) *attrName, OSINT16 nsidx, size_t nAttr, const [OSXMLAttrDescr](#) attrNames[], OSUINT32 attrPresent[])
This function finds an attribute in the descriptor table.
- EXTERNXML int [rtXmlpGetNextElem](#) (OSCTXT *pctx, [OSXMLElemDescr](#) *pElem, OSINT32 level)
This function parse the next element start tag.
- EXTERNXML int [rtXmlpGetNextElemID](#) (OSCTXT *pctx, const [OSXMLElemIDRec](#) *tab, size_t nrows, OSINT32 level, OSBOOL continueParse)
This function parses the next start tag and finds the index of the element name in the descriptor table.
- EXTERNXML int [rtXmlpMarkLastEventActive](#) (OSCTXT *pctx)
This function marks current tag as unprocessed.
- EXTERNXML int [rtXmlpMatchStartTag](#) (OSCTXT *pctx, const OSUTF8CHAR *elemLocalName, OSINT16 nsidx)
This function parses the next start tag that matches with given name.
- EXTERNXML int [rtXmlpMatchEndTag](#) (OSCTXT *pctx, OSINT32 level)
This function parse next end tag that matches with given name.
- EXTERNXML OSBOOL [rtXmlpHasAttributes](#) (OSCTXT *pctx)
This function checks accessibility of attributes.
- EXTERNXML int [rtXmlpGetAttributeCount](#) (OSCTXT *pctx)
This function returns number of attributes in last processed start tag.
- EXTERNXML int [rtXmlpSelectAttribute](#) (OSCTXT *pctx, [OSXMLNameFragments](#) *pAttr, OSINT16 *nsidx, size_t index)
This function selects attribute to decode.
- EXTERNXML OSINT32 [rtXmlpGetCurrentLevel](#) (OSCTXT *pctx)
This function returns current nesting level.
- EXTERNXML void [rtXmlpSetWhiteSpaceMode](#) (OSCTXT *pctx, [OSXMLWhiteSpaceMode](#) whiteSpaceMode)
Sets the whitespace treatment mode.

- EXTERNXML OSBOOL [rtXmlpSetMixedContentMode](#) (OSCTXT *pctx, OSBOOL mixedContentMode)
Sets mixed content mode.
- EXTERNXML void [rtXmlpSetListMode](#) (OSCTXT *pctx)
Sets list mode.
- EXTERNXML OSBOOL [rtXmlpListHasItem](#) (OSCTXT *pctx)
Check for end of decoded token list.
- EXTERNXML void [rtXmlpCountListItems](#) (OSCTXT *pctx, OSUINT32 *itemCnt)
Count tokens in list.
- EXTERNXML int [rtXmlpGetNextSeqElemID](#) (OSCTXT *pctx, const [OSXMLElemIDRec](#) *tab, const [OSXMLGroupDesc](#) *pGroup, int curID, int lastMandatoryID, OSBOOL groupMode)
This function parses the next start tag and find index of element name in descriptor table.
- EXTERNXML int [rtXmlpGetNextAllElemID](#) (OSCTXT *pctx, const [OSXMLElemIDRec](#) *tab, size_t nRows, const OSUINT8 *pOrder, OSUINT32 nOrder, OSUINT32 maxOrder, int anyID)
This function parses the next start tag and finds index of element name in descriptor table.
- EXTERNXML int [rtXmlpGetNextAllElemID16](#) (OSCTXT *pctx, const [OSXMLElemIDRec](#) *tab, size_t nRows, const OSUINT16 *pOrder, OSUINT32 nOrder, OSUINT32 maxOrder, int anyID)
This function parses the next start tag and finds index of element name in descriptor table.
- EXTERNXML void [rtXmlpSetNamespaceTable](#) (OSCTXT *pctx, const OSUTF8CHAR *namespaceTable[], size_t nmNamespaces)
Sets user namespace table.
- EXTERNXML int [rtXmlpCreateReader](#) (OSCTXT *pctx)
Creates pull parser reader structure within the context.
- EXTERNXML void [rtXmlpHideAttributes](#) (OSCTXT *pctx)
Disable access to attributes.
- EXTERNXML OSBOOL [rtXmlpNeedDecodeAttributes](#) (OSCTXT *pctx)
This function checks if attributes were previously decoded.
- EXTERNXML void [rtXmlpMarkPos](#) (OSCTXT *pctx)
Save current decode position.
- EXTERNXML void [rtXmlpRewindToMarkedPos](#) (OSCTXT *pctx)
Rewind to saved decode position.
- EXTERNXML void [rtXmlpResetMarkedPos](#) (OSCTXT *pctx)
Reset saved decode position.
- EXTERNXML int [rtXmlpGetXSITypeAttr](#) (OSCTXT *pctx, const OSUTF8CHAR **ppAttrValue, OSINT16 *nsidx, size_t *pLocalOffs)
This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type).
- EXTERNXML int [rtXmlpGetXmlnsAttrs](#) (OSCTXT *pctx, OSRTDList *pNSAttrs)

This function decodes namespace attributes from start tag and adds them to the given list.

- EXTERNXML int [rtXmlpDecXSIAttrs](#) (OSCTXT *pctxt)
This function decodes XSI (XML Schema Instance) that may be present in any arbitrary XML element within a document.
- EXTERNXML OSBOOL [rtXmlpIsEmptyElement](#) (OSCTXT *pctxt)
Check element content: empty or not.
- EXTERNXML int [rtXmlEncAttrC14N](#) (OSCTXT *pctxt)
This function used only in C14 mode.
- EXTERNXML OSBOOL [rtXmlpIsLastEventDone](#) (OSCTXT *pctxt)
Check processing status of current tag.
- EXTERNXML int [rtXmlpGetXSITypeIndex](#) (OSCTXT *pctxt, const [OSXMLItemDescr](#) typetab[], size_t type-
tabsiz)
*This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type) and find type index in
descriptor table.*
- EXTERNXML int [rtXmlpLookupXSITypeIndex](#) (OSCTXT *pctxt, const OSUTF8CHAR *pXsiType, OS-
INT16 xsiTypeIdx, const [OSXMLItemDescr](#) typetab[], size_t typetabsiz)
This function find index of XSI (XML Schema Instance) type in descriptor table.
- EXTERNXML void [rtXmlpForceDecodeAsGroup](#) (OSCTXT *pctxt)
Disable skipping of unknown elements in optional sequence tail.
- EXTERNXML OSBOOL [rtXmlpIsDecodeAsGroup](#) (OSCTXT *pctxt)
This function checks if "decode as group" mode was forced.

7.1.1 Detailed Description

XML low-level C encode/decode functions.

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

7.1.2 Define Documentation

7.1.2.1 #define IS_XMLNSATTR(name)

Value:

```
((OSUTF8LEN(name) >= 5) && name[0] == 'x' && name[1] == 'm' && \
name[2] == 'l' && name[3] == 'n' && name[4] == 's')
```

Definition at line 137 of file [osrtxml.h](#).

7.1.2.2 #define IS_XSIATTR(name)

Value:

```
((OSUTF8LEN(name) >= 4) && name[0] == 'x' && name[1] == 's' && \
name[2] == 'i' && name[3] == ':')
```

Definition at line 141 of file osrtxml.h.

7.1.2.3 #define OSXMLQNAMEEQUALS(xnamefrag, qnametext)

Value:

```
rtxUTF8StrnEqual \
(xnamefrag.mQName.value, OSUTF8(qnametext), xnamefrag.mQName.length)
```

Definition at line 130 of file osrtxml.h.

7.1.2.4 #define OSXMLSETUTF8DECPTR(pctxt, str)

Value:

```
rtxInitContextBuffer (pctxt, OSRTSAFECONSTCAST (OSOCKET*, str), \
OSUTF8LEN (str))
```

Definition at line 133 of file osrtxml.h.

7.1.3 Function Documentation

7.1.3.1 EXTERNXML const OSUTF8CHAR* rtSaxGetAttrValue (const OSUTF8CHAR * attrName, const OSUTF8CHAR *const * attrs)

This function looks up an attribute in the attribute array returned by SAX to the startElement function.

Parameters:

attrName Name of the attribute to find.

attrs Attribute array returned in SAX startElement function. This is an array of character strings containing name1, value1, name2, value2, ... List is terminated by a null name.

Returns:

Pointer to character string containing attribute value or NULL if attrName not found.

7.1.3.2 EXTERNXML OSINT16 rtSaxGetElemID (OSINT16 * pState, OSINT16 prevElemIdx, const OSUTF8CHAR * localName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT16 * fstab, OSINT16 fstabRows, OSINT16 fstabCols)

This function looks up a sequence element name in the given element info array.

If ensures elements are received in the correct order and also sets the required element count variable.

Parameters:

pState The pointer to state variable to be changed.
prevElemIdx Previous index of element. The search will be started from this element for better performance.
localName Local name of XML element
nsidx Namespace index
idtab Element ID table
fstab Finite state table
fstabRows Number of rows in *fstab*.
fstabCols Number of columns in *fstab*.

7.1.3.3 EXTERNXML OSINT16 rtSaxGetElemID8 (OSINT16 * pState, OSINT16 prevElemIdx, const OSUTF8CHAR * localName, OSINT32 nsidx, const OSSAXElemTableRec idtab[], const OSINT8 * fstab, OSINT16 fstabRows, OSINT16 fstabCols)

This function is a space optimized version of `rtSaxGetElemID`.

It operates with array of 8-bit integers (OSINT8) instead of 32-bit integers (int).

Parameters:

pState The pointer to state variable to be changed.
prevElemIdx Previous index of element. The search will be started from this element + 1 for better performance.
localName Local name of XML element
nsidx Namespace index
idtab Element ID table
fstab Finite state table (array of 8-bit integers)
fstabRows Number of rows in *fstab*.
fstabCols Number of columns in *fstab*.

7.1.3.4 EXTERNXML OSBOOL rtSaxHasXMLNSAttrs (const OSUTF8CHAR *const * attrs)

This function checks if the given attribute list contains one or more XML namespace attributes (xmlns).

Parameters:

attrs Attribute list in form passed by parser into SAX `startElement` function.

Returns:

TRUE, if xmlns attribute found in list.

7.1.3.5 EXTERNXML OSBOOL rtSaxIsEmptyBuffer (OSCTXT * pctxt)

This function checks if the buffer in the context is empty or not.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

TRUE, if the buffer contains empty string.

7.1.3.6 EXTERNXML int rtSaxSortAttrs (OSCTXT * *pctxt*, const OSUTF8CHAR *const * *attrs*, OSUINT16 ** *order*)

This function sorts a SAX attribute list in ascending order based on attribute name.

It currently only supports unqualified attributes.

Parameters:

pctxt Pointer to OSCTXT structure.

attrs Standard SAX attribute list. Entry *i* is attribute name and *i+1* is value. List is terminated by a null name.

order Order array containing the order of sorted attributes. This array is allocated using `rtxMemAlloc`, it can be freed using `rtxMemFreePtr` or will be freed when the context is freed. The list holds indices to name items in the attribute list that is passed in.

Returns:

If success, positive value contains number of attributes in *attrs*; if failure, negative status code.

7.1.3.7 EXTERNXML int rtSaxStrListMatch (OSCTXT * *pctxt*)

This function matches the list of strings.

It is used for matching NMTOKENS, IDREFS, NMENTITIES.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

0 - if success, negative value is error.

7.1.3.8 EXTERNXML int rtSaxStrListParse (OSCTXT * *pctxt*, OSRTMEMBUF * *pMemBuf*, OSRTDList * *pvalue*)

This function parses the list of strings.

It is used for parsing NMTOKENS, IDREFS, NMENTITIES.

Parameters:

pctxt Pointer to OSCTXT structure. Can be NULL, if *pMemBuf* is not NULL.

pMemBuf Pointer to memory buffer structure. Can be NULL, if *pctxt* is not NULL.

pvalue Doubly-linked list for parsed strings.

Returns:

0 - if success, negative value is error.

7.1.3.9 EXTERNXML OSBOOL rtXmlCmpBase64Str (OSUINT32 *noctsl*, const OSOCTET * *data1*, const OSUTF8CHAR * *data2*)

This function compares an array of octets to a base64 string.

Parameters:

- noctsl* Number of octets in data1.
- data1* Pointer to array of OSOCTET.
- data2* Pointer to null-terminated array of OSUTF8CHAR.

Returns:

TRUE if data2 is a base64 string representation of data1, false otherwise.

7.1.3.10 EXTERNXML OSBOOL rtXmlCmpHexStr (OSUINT32 *noctsl*, const OSOCTET * *data1*, const OSUTF8CHAR * *data2*)

This function compares an array of octets to a hex string.

Parameters:

- noctsl* Number of octets in data1.
- data1* Pointer to array of OSOCTET.
- data2* Pointer to null-terminated array of OSUTF8CHAR.

Returns:

TRUE if data2 is a hex string representation of data1, false otherwise.

7.1.3.11 EXTERNXML int rtXmlCreateFileInputSource (OSCTXT * *pctxt*, const char * *filepath*)

This function creates an XML document file input source.

The document can then be decoded by invoking an XML decode function.

Parameters:

- pctxt* Pointer to context block structure.
- filepath* Full pathname of XML document file to open.

Returns:

- Completion status of operation:
- 0 = success,
 - negative return value is error.

7.1.3.12 EXTERNXML int rtXmlInitContext (OSCTXT * *pctxt*)

This function initializes a context variable for XML encoding or decoding.

Parameters:

- pctxt* Pointer to OSCTXT structure

7.1.3.13 EXTERNXML int rtXmlInitCtxtAppInfo (OSCTXT * *pctxt*)

This function initializes the XML application info section of the given context.

Parameters:

pctxt Pointer to OSCTXT structure

7.1.3.14 EXTERNXML int rtXmlMatchBase64Str (OSCTXT * *pctxt*, size_t *minLength*, size_t *maxLength*)

This function tests the context buffer for containing a correct base64 string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

minLength A minimal length of expected string.

maxLength A maximal length of expected string.

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.15 EXTERNXML int rtXmlMatchDate (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct date string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.16 EXTERNXML int rtXmlMatchDateTime (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct dateTime string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.17 EXTERNXML int rtXmlMatchGDay (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct gDay string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.18 EXTERNXML int rtXmlMatchGMonth (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct gMonth string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.19 EXTERNXML int rtXmlMatchGMonthDay (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct gMonthDay string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.20 EXTERNXML int rtXmlMatchGYear (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct gYear string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.21 EXTERNXML int rtXmlMatchGYearMonth (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct gYearMonth string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.22 EXTERNXML int rtXmlMatchHexStr (OSCTXT * *pctxt*, size_t *minLength*, size_t *maxLength*)

This function tests the context buffer for containing a correct hexadecimal string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

minLength A minimal length of expected string.

maxLength A maximal length of expected string.

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.23 EXTERNXML int rtXmlMatchTime (OSCTXT * *pctxt*)

This function tests the context buffer for containing a correct time string.

It does not decode the value.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

If the string in the context buffer is a correct one, function returns zero. Error code (negative) will be returned otherwise. Note, error record will NOT be added in the error's list of the context.

7.1.3.24 EXTERNXML OSUTF8CHAR* rtXmlNewQName (OSCTXT * *pctxt*, const OSUTF8CHAR * *localName*, const OSUTF8CHAR * *prefix*)

This function creates a new QName given the localName and prefix parts.

Parameters:

pctxt Pointer to a context structure.

localName Element local name.

prefix Namespace prefix.

Returns:

QName value. Memory for the value will have been allocated by `rtxMemAlloc` and thus must be freed using one of the `rtxMemFree` functions. The value will be NULL if no dynamic memory was available.

7.1.3.25 EXTERNXML int rtXmlPrepareContext (OSCTXT * *pctxt*)

This function prepares the context for another encode by setting the state back to OSXMLINIT and moving the buffer's cursor back to the beginning of the buffer.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

0 if OK, negative status code if error.

7.1.3.26 EXTERNXML int rtXmlSetEncC14N (OSCTXT * *pctxt*, OSBOOL *value*)

This function sets the option to encode in C14N mode.

Parameters:

pctxt Pointer to OSCTXT structure

value Boolean value: true = C14N mode enabled.

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.27 EXTERNXML int rtXmlSetEncDocHdr (OSCTXT * *pctxt*, OSBOOL *value*)

This function sets the option to add the XML document header (i.e.

<?xml version="1.0" encoding="UTF-8"?>) to the XML output stream.

Parameters:

pctxt Pointer to OSCTXT structure

value Boolean value: true = add document header

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.28 EXTERNXML int rtXmlSetEncodingStr (OSCTXT * *pctxt*, const OSUTF8CHAR * *encodingStr*)

This function sets the XML output encoding to the given value.

Currently, only UTF-8 encoding is supported.

Parameters:

pctxt Pointer to OSCTXT structure

encodingStr XML output encoding format

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.29 EXTERNXML int rtXmlSetEncXSINamespace (OSCTXT * *pctxt*, OSBOOL *value*)

This function sets a flag in the context that indicates the XSI namespace declaration (xmlns:xsi) should be added to the encoded XML instance.

Parameters:

pctxt Pointer to OSCTXT structure

value Boolean value: true = encode XSI namespace attribute.

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.30 EXTERNXML int rtXmlSetFormatting (OSCTXT * *pctxt*, OSBOOL *doFormatting*)

This function sets XML output formatting to the given value.

If TRUE (the default), the XML document is formatted with indentation and newlines. If FALSE, all whitespace between elements is suppressed. Turning formatting off can provide more compressed documents and also a more canonical representation which is important for security applications. Also the function 'rtXmlSetIndent' might be used to set the exact size of indentation.

Parameters:

pctxt Pointer to OSCTXT structure

doFormatting Boolean value indicating if formatting is to be done

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.31 EXTERNXML int rtXmlSetIndent (OSCTXT * *pctxt*, OSUINT8 *indent*)

This function sets XML output indent to the given value.

Parameters:

pctxt Pointer to OSCTXT structure

indent Number of spaces per indent. Default is 3.

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.32 EXTERNXML int rtXmlSetIndentChar (OSCTXT * *pctxt*, char *indentChar*)

This function sets XML output indent character to the given value.

Parameters:

pctxt Pointer to OSCTXT structure

indentChar Indent character. Default is space.

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.33 EXTERNXML void rtXmlSetNamespacesSet (OSCTXT * *pctxt*, OSBOOL *value*)

This function sets the context 'namespaces are set' flag.

This indicates that namespace declarations have been set either by the decoder or externally by the end user. It is used by the encoder to know not to set the default namespaces specified in the schema before starting encoding.

Parameters:

pctxt Pointer to OSCTXT structure.

value Boolean value to which flag is to be set.

7.1.3.34 EXTERNXML int rtXmlSetNoNSSchemaLocation (OSCTXT * *pctxt*, const OSUTF8CHAR * *schemaLocation*)

This function sets the XML Schema Instance (xsi) no namespace schema location attribute to be added to an encoded document.

This attribute is optional: if not set, no xsi:noNamespaceSchemaLocation attribute will be added.

Parameters:

pctxt Pointer to OSCTXT structure

schemaLocation Schema location attribute value

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.35 EXTERNXML int rtXmlSetNSPrefixLinks (OSCTXT * *pctxt*, OSRTDList * *pNSAttrs*)

This function sets namespace prefix/URI links in the namespace prefix stack in the context structure.

Parameters:

pctxt Pointer to OSCTXT structure.

pNSAttrs List of namespace attributes.

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.36 EXTERNXML int rtXmlSetSchemaLocation (OSCTXT * *pctxt*, const OSUTF8CHAR * *schemaLocation*)

This function sets the XML Schema Instance (xsi) schema location attribute to be added to an encoded document.

This attribute is optional: if not set, no xsi:schemaLocation attribute will be added.

Parameters:

pctxt Pointer to OSCTXT structure

schemaLocation Schema location attribute value

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.37 EXTERNXML void rtXmlSetSoapVersion (OSCTXT * *pctxt*, OSUINT8 *version*)

This function sets the SOAP version number.

Parameters:

pctxt Pointer to OSCTXT structure

version SOAP version number as 2 digit integer (for example, 11 is SOAP version 1.1, 12 is version 1.2, etc.)

7.1.3.38 EXTERNXML int rtXmlSetWriteBOM (OSCTXT * *pctxt*, OSBOOL *write*)

This function sets whether the Unicode byte order mark is encoded.

Parameters:

pctxt Pointer to OSCTXT structure

write TRUE to encode BOM, FALSE to not encode BOM.

Returns:

Status of operation: 0 if OK, negative status code if error.

7.1.3.39 EXTERNXML int rtXmlSetXSITypeAttr (OSCTXT * *pctxt*, const OSUTF8CHAR * *xsiType*)

This function sets the XML Schema Instance (xsi) type attribute value.

This will cause an xsi:type attribute to be added to the top level element in an encoded XML instance.

Parameters:

pctxt Pointer to OSCTXT structure

xsiType xsi:type attribute value

Returns:

Status of operation: 0 if OK, negative status code if error.

7.2 rtXmlCppMsgBuf.h File Reference

This file is deprecated.

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

7.2.1 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](#).

7.3 rtXmlErrCodes.h File Reference

List of numeric status codes that can be returned by ASN1C run-time functions and generated code.

```
#include "rtxsrc/rtxErrCodes.h"
```

Defines

- #define [XML_OK_EOB](#) 0x7fffffff
End of block marker.
- #define [XML_OK_FRAG](#) XML_OK_EOB
Maintained for backward compatibility.
- #define [XML_E_BASE](#) -200
Error base.
- #define [XML_E_GENERR](#) (XML_E_BASE)
General error.
- #define [XML_E_INVSYMBOL](#) (XML_E_BASE-1)
An invalid XML symbol (character) was detected at the given point in the parse stream.
- #define [XML_E_TAGMISMATCH](#) (XML_E_BASE-2)
Start/end tag mismatch.
- #define [XML_E_DUPLATTR](#) (XML_E_BASE-3)
Duplicate attribute found.
- #define [XML_E_BADCHARREF](#) (XML_E_BASE-4)
Bad character reference found.
- #define [XML_E_INVMODE](#) (XML_E_BASE-5)
Invalid mode.
- #define [XML_E_UNEXPEOF](#) (XML_E_BASE-6)
Unexpected end of file (document).
- #define [XML_E_NOMATCH](#) (XML_E_BASE-7)
Current tag is not matched to specified one.
- #define [XML_E_ELEMMISRQ](#) (XML_E_BASE-8)
Missing required element.
- #define [XML_E_ELEMSISRQ](#) (XML_E_BASE-9)
Missing required elements.
- #define [XML_E_TOOFWELEMS](#) (XML_E_BASE-10)
The number of elements in a repeating collection was less than the number of elements specified in the XSD minOccurs facet for this type or element.

- #define [XML_E_UNEXPSTARTTAG](#) (XML_E_BASE-11)
Unexpected start tag.
- #define [XML_E_UNEXPENDTAG](#) (XML_E_BASE-12)
Unexpected end tag.
- #define [XML_E_IDNOTFOU](#) (XML_E_BASE-13)
Expected identifier not found.
- #define [XML_E_INVTYPEINFO](#) (XML_E_BASE-14)
Unknown xsi:type.
- #define [XML_E_NSURINOTFOU](#) (XML_E_BASE-15)
Namespace URI not defined for given prefix.

7.3.1 Detailed Description

List of numeric status codes that can be returned by ASN1C run-time functions and generated code.

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

Index

ASN.1-XML encode/decode functions., 5

asn1xml

- rtAsn1XmlAddAnyAttr, 6
- rtAsn1XmlEncGenTime, 6
- rtAsn1XmlEncObjId, 7
- rtAsn1XmlEncOpenType, 7
- rtAsn1XmlEncOpenTypeExt, 8
- rtAsn1XmlEncRelOID, 8
- rtAsn1XmlEncUnivStr, 8
- rtAsn1XmlEncUTCTime, 9
- rtAsn1XmlFmtAttrStr, 9
- rtAsn1XmlParseAttrStr, 10
- rtAsn1XmlpDecDynBitStr, 10
- rtAsn1XmlpDecObjId, 10
- rtAsn1XmlpDecRelOID, 11
- rtAsn1XmlpDecUnivStr, 11
- rtXmlpDecListOfASN1DynBitStr, 11

IS_XMLNSATTR

osrtxml.h, 132

IS_XSIATTR

osrtxml.h, 132

OSDecimalFmt, 101

OSDoubleFmt, 102

OSIntegerFmt, 103

osrtxml.h, 114

IS_XMLNSATTR, 132

IS_XSIATTR, 132

OSXMLQNAMEEQUALS, 133

OSXMLSETUTF8DECPTR, 133

rtSaxGetAttrValue, 133

rtSaxGetElemID, 133

rtSaxGetElemID8, 134

rtSaxHasXMLNSAttrs, 134

rtSaxIsEmptyBuffer, 134

rtSaxSortAttrs, 134

rtSaxStrListMatch, 135

rtSaxStrListParse, 135

rtXmlCmpBase64Str, 135

rtXmlCmpHexStr, 136

rtXmlCreateFileInputSource, 136

rtXmlInitContext, 136

rtXmlInitCtxtAppInfo, 136

rtXmlMatchBase64Str, 137

rtXmlMatchDate, 137

rtXmlMatchDateTime, 137

rtXmlMatchGDay, 137

rtXmlMatchGMonth, 138

rtXmlMatchGMonthDay, 138

rtXmlMatchGYear, 138

rtXmlMatchGYearMonth, 138

rtXmlMatchHexStr, 139

rtXmlMatchTime, 139

rtXmlNewQName, 139

rtXmlPrepareContext, 140

rtXmlSetEncC14N, 140

rtXmlSetEncDocHdr, 140

rtXmlSetEncodingStr, 140

rtXmlSetEncXSINamespace, 141

rtXmlSetFormatting, 141

rtXmlSetIndent, 141

rtXmlSetIndentChar, 142

rtXmlSetNamespacesSet, 142

rtXmlSetNoNSSchemaLocation, 142

rtXmlSetNSPrefixLinks, 142

rtXmlSetSchemaLocation, 143

rtXmlSetSoapVersion, 143

rtXmlSetWriteBOM, 143

rtXmlSetXSITypeAttr, 143

OSXMLCtxtInfo, 104

OSXMLElemIDRec, 105

OSXMLFacets, 106

OSXMLGroupDesc, 107

OSXMLItemDescr, 108

OSXMLNameFragments, 109

OSXMLQName, 110

OSXMLQNAMEEQUALS

osrtxml.h, 133

OSXMLSETUTF8DECPTR

osrtxml.h, 133

OSXMLSortedAttrOffset, 111

OSXMLStrFragment, 112

OSXSDAnyType, 113

rtAsn1XmlAddAnyAttr

asn1xml, 6

rtAsn1XmlEncGenTime

asn1xml, 6

rtAsn1XmlEncObjId

- asn1xml, 7
- rtAsn1XmlEncOpenType
 - asn1xml, 7
- rtAsn1XmlEncOpenTypeExt
 - asn1xml, 8
- rtAsn1XmlEncRelOID
 - asn1xml, 8
- rtAsn1XmlEncUnivStr
 - asn1xml, 8
- rtAsn1XmlEncUTCTime
 - asn1xml, 9
- rtAsn1XmlFmtAttrStr
 - asn1xml, 9
- rtAsn1XmlParseAttrStr
 - asn1xml, 10
- rtAsn1XmlpDecDynBitStr
 - asn1xml, 10
- rtAsn1XmlpDecObjId
 - asn1xml, 10
- rtAsn1XmlpDecRelOID
 - asn1xml, 11
- rtAsn1XmlpDecUnivStr
 - asn1xml, 11
- rtSaxGetAttrValue
 - osrtxml.h, 133
- rtSaxGetElemID
 - osrtxml.h, 133
- rtSaxGetElemID8
 - osrtxml.h, 134
- rtSaxHasXMLNSAttrs
 - osrtxml.h, 134
- rtSaxIsEmptyBuffer
 - osrtxml.h, 134
- rtSaxSortAttrs
 - osrtxml.h, 134
- rtSaxStrListMatch
 - osrtxml.h, 135
- rtSaxStrListParse
 - osrtxml.h, 135
- rtXmlCmpBase64Str
 - osrtxml.h, 135
- rtXmlCmpHexStr
 - osrtxml.h, 136
- rtXmlCppMsgBuf.h, 145
- rtXmlCreateFileInputSource
 - osrtxml.h, 136
- rtXmlDec
 - rtXmlDecBase64Binary, 15
 - rtXmlDecBase64Str, 15
 - rtXmlDecBase64StrValue, 16
 - rtXmlDecBigInt, 16
 - rtXmlDecBool, 17
 - rtXmlDecDate, 17
 - rtXmlDecDateTime, 17
 - rtXmlDecDecimal, 18
 - rtXmlDecDouble, 18
 - rtXmlDecDynBase64Str, 18
 - rtXmlDecDynHexStr, 19
 - rtXmlDecDynUTF8Str, 19
 - rtXmlDecGDay, 19
 - rtXmlDecGMonth, 20
 - rtXmlDecGMonthDay, 20
 - rtXmlDecGYear, 20
 - rtXmlDecGYearMonth, 21
 - rtXmlDecHexBinary, 21
 - rtXmlDecHexStr, 22
 - rtXmlDecInt, 22
 - rtXmlDecInt16, 22
 - rtXmlDecInt64, 23
 - rtXmlDecInt8, 23
 - rtXmlDecNSAttr, 23
 - rtXmlDecQName, 24
 - rtXmlDecTime, 24
 - rtXmlDecUInt, 25
 - rtXmlDecUInt16, 25
 - rtXmlDecUInt64, 25
 - rtXmlDecUInt8, 26
 - rtXmlDecUTF8Str, 26
 - rtXmlDecXmlStr, 26
 - rtXmlDecXSIAAttr, 27
 - rtXmlDecXSIAAttrs, 27
 - rtXmlParseElementName, 28
 - rtXmlParseElemQName, 28
- rtXmlDecBase64Binary
 - rtXmlDec, 15
- rtXmlDecBase64Str
 - rtXmlDec, 15
- rtXmlDecBase64StrValue
 - rtXmlDec, 16
- rtXmlDecBigInt
 - rtXmlDec, 16
- rtXmlDecBool
 - rtXmlDec, 17
- rtXmlDecDate
 - rtXmlDec, 17
- rtXmlDecDateTime
 - rtXmlDec, 17
- rtXmlDecDecimal
 - rtXmlDec, 18
- rtXmlDecDouble
 - rtXmlDec, 18
- rtXmlDecDynBase64Str
 - rtXmlDec, 18
- rtXmlDecDynHexStr
 - rtXmlDec, 19
- rtXmlDecDynUTF8Str
 - rtXmlDec, 19
- rtXmlDecGDay

- rtXmlDec, 19
- rtXmlDecGMonth
 - rtXmlDec, 20
- rtXmlDecGMonthDay
 - rtXmlDec, 20
- rtXmlDecGYear
 - rtXmlDec, 20
- rtXmlDecGYearMonth
 - rtXmlDec, 21
- rtXmlDecHexBinary
 - rtXmlDec, 21
- rtXmlDecHexStr
 - rtXmlDec, 22
- rtXmlDecInt
 - rtXmlDec, 22
- rtXmlDecInt16
 - rtXmlDec, 22
- rtXmlDecInt64
 - rtXmlDec, 23
- rtXmlDecInt8
 - rtXmlDec, 23
- rtXmlDecNSAttr
 - rtXmlDec, 23
- rtXmlDecQName
 - rtXmlDec, 24
- rtXmlDecTime
 - rtXmlDec, 24
- rtXmlDecUInt
 - rtXmlDec, 25
- rtXmlDecUInt16
 - rtXmlDec, 25
- rtXmlDecUInt64
 - rtXmlDec, 25
- rtXmlDecUInt8
 - rtXmlDec, 26
- rtXmlDecUTF8Str
 - rtXmlDec, 26
- rtXmlDecXmlStr
 - rtXmlDec, 26
- rtXmlDecXSIAttr
 - rtXmlDec, 27
- rtXmlDecXSIAttrs
 - rtXmlDec, 27
- rtXmlEnc
 - rtXmlEncAny, 36
 - rtXmlEncAnyAttr, 36
 - rtXmlEncAnyTypeValue, 36
 - rtXmlEncBase64Binary, 37
 - rtXmlEncBase64BinaryAttr, 37
 - rtXmlEncBase64StrValue, 38
 - rtXmlEncBigInt, 38
 - rtXmlEncBigIntAttr, 38
 - rtXmlEncBigIntValue, 39
 - rtXmlEncBinStrValue, 39
 - rtXmlEncBitString, 39
 - rtXmlEncBOM, 40
 - rtXmlEncBool, 40
 - rtXmlEncBoolAttr, 41
 - rtXmlEncBoolValue, 41
 - rtXmlEncDate, 41
 - rtXmlEncDateTime, 42
 - rtXmlEncDateTimeValue, 42
 - rtXmlEncDateValue, 43
 - rtXmlEncDecimal, 43
 - rtXmlEncDecimalAttr, 43
 - rtXmlEncDecimalValue, 44
 - rtXmlEncDouble, 44
 - rtXmlEncDoubleAttr, 45
 - rtXmlEncDoubleValue, 45
 - rtXmlEncEmptyElement, 45
 - rtXmlEncEndDocument, 46
 - rtXmlEncEndElement, 46
 - rtXmlEncEndSoapElems, 46
 - rtXmlEncEndSoapEnv, 47
 - rtXmlEncFloat, 47
 - rtXmlEncFloatAttr, 47
 - rtXmlEncGDay, 48
 - rtXmlEncGDayValue, 48
 - rtXmlEncGMonth, 48
 - rtXmlEncGMonthDay, 49
 - rtXmlEncGMonthDayValue, 49
 - rtXmlEncGMonthValue, 49
 - rtXmlEncGYear, 50
 - rtXmlEncGYearMonth, 50
 - rtXmlEncGYearMonthValue, 50
 - rtXmlEncGYearValue, 51
 - rtXmlEncHexBinary, 51
 - rtXmlEncHexBinaryAttr, 51
 - rtXmlEncHexStrValue, 52
 - rtXmlEncIndent, 52
 - rtXmlEncInt, 52
 - rtXmlEncInt64, 53
 - rtXmlEncInt64Attr, 53
 - rtXmlEncInt64Value, 54
 - rtXmlEncIntAttr, 54
 - rtXmlEncIntPattern, 54
 - rtXmlEncIntValue, 55
 - rtXmlEncNamedBits, 55
 - rtXmlEncNSAttrs, 56
 - rtXmlEncReal10, 56
 - rtXmlEncSoapArrayTypeAttr, 56
 - rtXmlEncStartDocument, 57
 - rtXmlEncStartElement, 57
 - rtXmlEncStartSoapElems, 58
 - rtXmlEncStartSoapEnv, 58
 - rtXmlEncString, 58
 - rtXmlEncStringValue, 59
 - rtXmlEncStringValue2, 59

- rtXmlEncTermStartElement, [59](#)
- rtXmlEncTime, [60](#)
- rtXmlEncTimeValue, [60](#)
- rtXmlEncUInt, [60](#)
- rtXmlEncUInt64, [61](#)
- rtXmlEncUInt64Attr, [61](#)
- rtXmlEncUInt64Value, [62](#)
- rtXmlEncUIntAttr, [62](#)
- rtXmlEncUIntValue, [62](#)
- rtXmlEncUnicodeStr, [63](#)
- rtXmlEncUTF8Attr, [63](#)
- rtXmlEncUTF8Attr2, [63](#)
- rtXmlEncUTF8Str, [64](#)
- rtXmlEncXSIAAttrs, [64](#)
- rtXmlEncXSITypeAttr, [65](#)
- rtXmlFinalizeMemBuf, [35](#)
- rtXmlFreeInputSource, [65](#)
- rtXmlGetEncBufLen, [35](#)
- rtXmlGetEncBufPtr, [35](#)
- rtXmlGetIndent, [65](#)
- rtXmlGetIndentChar, [65](#)
- rtXmlGetWriteBOM, [66](#)
- rtXmlSetEncBufPtr, [66](#)
- rtxPrintNSAttrs, [66](#)
- rtXmlEncAny
 - rtXmlEnc, [36](#)
- rtXmlEncAnyAttr
 - rtXmlEnc, [36](#)
- rtXmlEncAnyTypeValue
 - rtXmlEnc, [36](#)
- rtXmlEncAttrC14N
 - rtXmlpDec, [72](#)
- rtXmlEncBase64Binary
 - rtXmlEnc, [37](#)
- rtXmlEncBase64BinaryAttr
 - rtXmlEnc, [37](#)
- rtXmlEncBase64StrValue
 - rtXmlEnc, [38](#)
- rtXmlEncBigInt
 - rtXmlEnc, [38](#)
- rtXmlEncBigIntAttr
 - rtXmlEnc, [38](#)
- rtXmlEncBigIntValue
 - rtXmlEnc, [39](#)
- rtXmlEncBinStrValue
 - rtXmlEnc, [39](#)
- rtXmlEncBitString
 - rtXmlEnc, [39](#)
- rtXmlEncBOM
 - rtXmlEnc, [40](#)
- rtXmlEncBool
 - rtXmlEnc, [40](#)
- rtXmlEncBoolAttr
 - rtXmlEnc, [41](#)
- rtXmlEncBoolValue
 - rtXmlEnc, [41](#)
- rtXmlEncDate
 - rtXmlEnc, [41](#)
- rtXmlEncDateTime
 - rtXmlEnc, [42](#)
- rtXmlEncDateTimeValue
 - rtXmlEnc, [42](#)
- rtXmlEncDateValue
 - rtXmlEnc, [43](#)
- rtXmlEncDecimal
 - rtXmlEnc, [43](#)
- rtXmlEncDecimalAttr
 - rtXmlEnc, [43](#)
- rtXmlEncDecimalValue
 - rtXmlEnc, [44](#)
- rtXmlEncDouble
 - rtXmlEnc, [44](#)
- rtXmlEncDoubleAttr
 - rtXmlEnc, [45](#)
- rtXmlEncDoubleValue
 - rtXmlEnc, [45](#)
- rtXmlEncEmptyElement
 - rtXmlEnc, [45](#)
- rtXmlEncEndDocument
 - rtXmlEnc, [46](#)
- rtXmlEncEndElement
 - rtXmlEnc, [46](#)
- rtXmlEncEndSoapElems
 - rtXmlEnc, [46](#)
- rtXmlEncEndSoapEnv
 - rtXmlEnc, [47](#)
- rtXmlEncFloat
 - rtXmlEnc, [47](#)
- rtXmlEncFloatAttr
 - rtXmlEnc, [47](#)
- rtXmlEncGDay
 - rtXmlEnc, [48](#)
- rtXmlEncGDayValue
 - rtXmlEnc, [48](#)
- rtXmlEncGMonth
 - rtXmlEnc, [48](#)
- rtXmlEncGMonthDay
 - rtXmlEnc, [49](#)
- rtXmlEncGMonthDayValue
 - rtXmlEnc, [49](#)
- rtXmlEncGMonthValue
 - rtXmlEnc, [49](#)
- rtXmlEncGYear
 - rtXmlEnc, [50](#)
- rtXmlEncGYearMonth
 - rtXmlEnc, [50](#)
- rtXmlEncGYearMonthValue
 - rtXmlEnc, [50](#)

rtXmlEncGYearValue	rtXmlEncUInt64
rtXmlEnc, 51	rtXmlEnc, 61
rtXmlEncHexBinary	rtXmlEncUInt64Attr
rtXmlEnc, 51	rtXmlEnc, 61
rtXmlEncHexBinaryAttr	rtXmlEncUInt64Value
rtXmlEnc, 51	rtXmlEnc, 62
rtXmlEncHexStrValue	rtXmlEncUIntAttr
rtXmlEnc, 52	rtXmlEnc, 62
rtXmlEncIndent	rtXmlEncUIntValue
rtXmlEnc, 52	rtXmlEnc, 62
rtXmlEncInt	rtXmlEncUnicodeStr
rtXmlEnc, 52	rtXmlEnc, 63
rtXmlEncInt64	rtXmlEncUTF8Attr
rtXmlEnc, 53	rtXmlEnc, 63
rtXmlEncInt64Attr	rtXmlEncUTF8Attr2
rtXmlEnc, 53	rtXmlEnc, 63
rtXmlEncInt64Value	rtXmlEncUTF8Str
rtXmlEnc, 54	rtXmlEnc, 64
rtXmlEncIntAttr	rtXmlEncXSIAttrs
rtXmlEnc, 54	rtXmlEnc, 64
rtXmlEncIntPattern	rtXmlEncXSITypeAttr
rtXmlEnc, 54	rtXmlEnc, 65
rtXmlEncIntValue	rtXmlErrCodes.h, 146
rtXmlEnc, 55	rtXmlFinalizeMemBuf
rtXmlEncNamedBits	rtXmlEnc, 35
rtXmlEnc, 55	rtXmlFreeInputSource
rtXmlEncNSAttrs	rtXmlEnc, 65
rtXmlEnc, 56	rtXmlGetEncBufLen
rtXmlEncReal10	rtXmlEnc, 35
rtXmlEnc, 56	rtXmlGetEncBufPtr
rtXmlEncSoapArrayTypeAttr	rtXmlEnc, 35
rtXmlEnc, 56	rtXmlGetIndent
rtXmlEncStartDocument	rtXmlEnc, 65
rtXmlEnc, 57	rtXmlGetIndentChar
rtXmlEncStartElement	rtXmlEnc, 65
rtXmlEnc, 57	rtXmlGetWriteBOM
rtXmlEncStartSoapElems	rtXmlEnc, 66
rtXmlEnc, 58	rtXmlInitContext
rtXmlEncStartSoapEnv	osrtxml.h, 136
rtXmlEnc, 58	rtXmlInitCtxAppInfo
rtXmlEncString	osrtxml.h, 136
rtXmlEnc, 58	rtXmlMatchBase64Str
rtXmlEncStringValue	osrtxml.h, 137
rtXmlEnc, 59	rtXmlMatchDate
rtXmlEncStringValue2	osrtxml.h, 137
rtXmlEnc, 59	rtXmlMatchDateTime
rtXmlEncTermStartElement	osrtxml.h, 137
rtXmlEnc, 59	rtXmlMatchGDay
rtXmlEncTime	osrtxml.h, 137
rtXmlEnc, 60	rtXmlMatchGMonth
rtXmlEncTimeValue	osrtxml.h, 138
rtXmlEnc, 60	rtXmlMatchGMonthDay
rtXmlEncUInt	osrtxml.h, 138
rtXmlEnc, 60	rtXmlMatchGYear

- osrtxml.h, 138
- rtXmlMatchGYearMonth
 - osrtxml.h, 138
- rtXmlMatchHexStr
 - osrtxml.h, 139
- rtXmlMatchTime
 - osrtxml.h, 139
- rtXmlNewQName
 - osrtxml.h, 139
- rtXmlParseElementName
 - rtXmlDec, 28
- rtXmlParseElemQName
 - rtXmlDec, 28
- rtXmpltCountListItems
 - rtXmpltDec, 73
- rtXmpltCreateReader
 - rtXmpltDec, 73
- rtXmpltDec
 - rtXmlEncAttrC14N, 72
 - rtXmpltCountListItems, 73
 - rtXmpltCreateReader, 73
 - rtXmpltDecAny, 73
 - rtXmpltDecAnyAttrStr, 74
 - rtXmpltDecAnyElem, 74
 - rtXmpltDecBase64Str, 74
 - rtXmpltDecBigInt, 75
 - rtXmpltDecBitString, 75
 - rtXmpltDecBool, 76
 - rtXmpltDecDate, 76
 - rtXmpltDecDateTime, 76
 - rtXmpltDecDecimal, 77
 - rtXmpltDecDouble, 77
 - rtXmpltDecDynBase64Str, 77
 - rtXmpltDecDynBitString, 78
 - rtXmpltDecDynHexStr, 78
 - rtXmpltDecDynUnicodeStr, 79
 - rtXmpltDecDynUTF8Str, 79
 - rtXmpltDecGDay, 79
 - rtXmpltDecGMonth, 80
 - rtXmpltDecGMonthDay, 80
 - rtXmpltDecGYear, 80
 - rtXmpltDecGYearMonth, 81
 - rtXmpltDecHexStr, 81
 - rtXmpltDecInt, 82
 - rtXmpltDecInt16, 82
 - rtXmpltDecInt64, 82
 - rtXmpltDecInt8, 83
 - rtXmpltDecNamedBits, 83
 - rtXmpltDecStrList, 83
 - rtXmpltDecTime, 84
 - rtXmpltDecUInt, 84
 - rtXmpltDecUInt16, 84
 - rtXmpltDecUInt64, 85
 - rtXmpltDecUInt8, 85
 - rtXmpltDecUTF8Str, 85
 - rtXmpltDecXmlStr, 86
 - rtXmpltDecXmlStrList, 86
 - rtXmpltDecXSIAAttr, 87
 - rtXmpltDecXSIAAttrs, 87
 - rtXmpltDecXSITypeAttr, 87
 - rtXmpltForceDecodeAsGroup, 88
 - rtXmpltGetAttributeCount, 88
 - rtXmpltGetAttributeID, 88
 - rtXmpltGetCurrentLevel, 89
 - rtXmpltGetNextAllElemID, 89
 - rtXmpltGetNextAllElemID16, 89
 - rtXmpltGetNextElem, 90
 - rtXmpltGetNextElemID, 90
 - rtXmpltGetNextSeqElemID, 90
 - rtXmpltGetXmpltAttrs, 91
 - rtXmpltGetXSITypeAttr, 91
 - rtXmpltGetXSITypeIndex, 92
 - rtXmpltHasAttributes, 92
 - rtXmpltHideAttributes, 92
 - rtXmpltIsDecodeAsGroup, 92
 - rtXmpltIsEmptyElement, 93
 - rtXmpltIsLastEventDone, 93
 - rtXmpltListHasItem, 93
 - rtXmpltLookupXSITypeIndex, 93
 - rtXmpltMarkLastEventActive, 94
 - rtXmpltMarkPos, 94
 - rtXmpltMatchEndTag, 94
 - rtXmpltMatchStartTag, 95
 - rtXmpltNeedDecodeAttributes, 95
 - rtXmpltResetMarkedPos, 95
 - rtXmpltRewindToMarkedPos, 95
 - rtXmpltSelectAttribute, 96
 - rtXmpltSetListMode, 96
 - rtXmpltSetMixedContentMode, 96
 - rtXmpltSetNamespaceTable, 96
 - rtXmpltSetWhiteSpaceMode, 97
- rtXmpltDecAny
 - rtXmpltDec, 73
- rtXmpltDecAnyAttrStr
 - rtXmpltDec, 74
- rtXmpltDecAnyElem
 - rtXmpltDec, 74
- rtXmpltDecBase64Str
 - rtXmpltDec, 74
- rtXmpltDecBigInt
 - rtXmpltDec, 75
- rtXmpltDecBitString
 - rtXmpltDec, 75
- rtXmpltDecBool
 - rtXmpltDec, 76
- rtXmpltDecDate
 - rtXmpltDec, 76
- rtXmpltDecDateTime

[rtXmlpDec, 76](#)
[rtXmlpDecDecimal](#)
[rtXmlpDec, 77](#)
[rtXmlpDecDouble](#)
[rtXmlpDec, 77](#)
[rtXmlpDecDynBase64Str](#)
[rtXmlpDec, 77](#)
[rtXmlpDecDynBitString](#)
[rtXmlpDec, 78](#)
[rtXmlpDecDynHexStr](#)
[rtXmlpDec, 78](#)
[rtXmlpDecDynUnicodeStr](#)
[rtXmlpDec, 79](#)
[rtXmlpDecDynUTF8Str](#)
[rtXmlpDec, 79](#)
[rtXmlpDecGDay](#)
[rtXmlpDec, 79](#)
[rtXmlpDecGMonth](#)
[rtXmlpDec, 80](#)
[rtXmlpDecGMonthDay](#)
[rtXmlpDec, 80](#)
[rtXmlpDecGYear](#)
[rtXmlpDec, 80](#)
[rtXmlpDecGYearMonth](#)
[rtXmlpDec, 81](#)
[rtXmlpDecHexStr](#)
[rtXmlpDec, 81](#)
[rtXmlpDecInt](#)
[rtXmlpDec, 82](#)
[rtXmlpDecInt16](#)
[rtXmlpDec, 82](#)
[rtXmlpDecInt64](#)
[rtXmlpDec, 82](#)
[rtXmlpDecInt8](#)
[rtXmlpDec, 83](#)
[rtXmlpDecListOfASN1DynBitStr](#)
[asn1xml, 11](#)
[rtXmlpDecNamedBits](#)
[rtXmlpDec, 83](#)
[rtXmlpDecStrList](#)
[rtXmlpDec, 83](#)
[rtXmlpDecTime](#)
[rtXmlpDec, 84](#)
[rtXmlpDecUInt](#)
[rtXmlpDec, 84](#)
[rtXmlpDecUInt16](#)
[rtXmlpDec, 84](#)
[rtXmlpDecUInt64](#)
[rtXmlpDec, 85](#)
[rtXmlpDecUInt8](#)
[rtXmlpDec, 85](#)
[rtXmlpDecUTF8Str](#)
[rtXmlpDec, 85](#)
[rtXmlpDecXmlStr](#)

[rtXmlpDec, 86](#)
[rtXmlpDecXmlStrList](#)
[rtXmlpDec, 86](#)
[rtXmlpDecXSIAAttr](#)
[rtXmlpDec, 87](#)
[rtXmlpDecXSIAAttrs](#)
[rtXmlpDec, 87](#)
[rtXmlpDecXSISTypeAttr](#)
[rtXmlpDec, 87](#)
[rtXmlpForceDecodeAsGroup](#)
[rtXmlpDec, 88](#)
[rtXmlpGetAttributeCount](#)
[rtXmlpDec, 88](#)
[rtXmlpGetAttributeID](#)
[rtXmlpDec, 88](#)
[rtXmlpGetCurrentLevel](#)
[rtXmlpDec, 89](#)
[rtXmlpGetNextAllElemID](#)
[rtXmlpDec, 89](#)
[rtXmlpGetNextAllElemID16](#)
[rtXmlpDec, 89](#)
[rtXmlpGetNextElem](#)
[rtXmlpDec, 90](#)
[rtXmlpGetNextElemID](#)
[rtXmlpDec, 90](#)
[rtXmlpGetNextSeqElemID](#)
[rtXmlpDec, 90](#)
[rtXmlpGetXmLnsAttrs](#)
[rtXmlpDec, 91](#)
[rtXmlpGetXSISTypeAttr](#)
[rtXmlpDec, 91](#)
[rtXmlpGetXSISTypeIndex](#)
[rtXmlpDec, 92](#)
[rtXmlpHasAttributes](#)
[rtXmlpDec, 92](#)
[rtXmlpHideAttributes](#)
[rtXmlpDec, 92](#)
[rtXmlpIsDecodeAsGroup](#)
[rtXmlpDec, 92](#)
[rtXmlpIsEmptyElement](#)
[rtXmlpDec, 93](#)
[rtXmlpIsLastEventDone](#)
[rtXmlpDec, 93](#)
[rtXmlpListHasItem](#)
[rtXmlpDec, 93](#)
[rtXmlpLookupXSISTypeIndex](#)
[rtXmlpDec, 93](#)
[rtXmlpMarkLastEventActive](#)
[rtXmlpDec, 94](#)
[rtXmlpMarkPos](#)
[rtXmlpDec, 94](#)
[rtXmlpMatchEndTag](#)
[rtXmlpDec, 94](#)
[rtXmlpMatchStartTag](#)

- rtXmlpDec, 95
- rtXmlpNeedDecodeAttributes
 - rtXmlpDec, 95
- rtXmlPrepareContext
 - osrtxml.h, 140
- rtXmlpResetMarkedPos
 - rtXmlpDec, 95
- rtXmlpRewindToMarkedPos
 - rtXmlpDec, 95
- rtXmlpSelectAttribute
 - rtXmlpDec, 96
- rtXmlpSetListMode
 - rtXmlpDec, 96
- rtXmlpSetMixedContentMode
 - rtXmlpDec, 96
- rtXmlpSetNamespaceTable
 - rtXmlpDec, 96
- rtXmlpSetWhiteSpaceMode
 - rtXmlpDec, 97
- rtXmlSetEncBufPtr
 - rtXmlEnc, 66
- rtXmlSetEncC14N
 - osrtxml.h, 140
- rtXmlSetEncDocHdr
 - osrtxml.h, 140
- rtXmlSetEncodingStr
 - osrtxml.h, 140
- rtXmlSetEncXSINamespace
 - osrtxml.h, 141
- rtXmlSetFormatting
 - osrtxml.h, 141
- rtXmlSetIndent
 - osrtxml.h, 141
- rtXmlSetIndentChar
 - osrtxml.h, 142
- rtXmlSetNamespacesSet
 - osrtxml.h, 142
- rtXmlSetNoNSSchemaLocation
 - osrtxml.h, 142
- rtXmlSetNSPrefixLinks
 - osrtxml.h, 142
- rtXmlSetSchemaLocation
 - osrtxml.h, 143
- rtXmlSetSoapVersion
 - osrtxml.h, 143
- rtXmlSetWriteBOM
 - osrtxml.h, 143
- rtXmlSetXSITypeAttr
 - osrtxml.h, 143
- rtXmlUtil
 - rtXmlWriteToFile, 67
- rtXmlWriteToFile
 - rtXmlUtil, 67
- rtxPrintNSAttrs

- rtXmlEnc, 66
- XML decode functions., 13
- XML encode functions., 29
- XML pull-parser decode functions., 68
- XML run-time error status codes., 98
- XML utility functions., 67
- XML_E_BASE
 - xmlErrCodes, 99
- XML_E_ELEMMISRQ
 - xmlErrCodes, 99
- XML_E_ELEMSISRQ
 - xmlErrCodes, 99
- XML_E_NOMATCH
 - xmlErrCodes, 99
- XML_E_NSURINOTFOU
 - xmlErrCodes, 100
- XML_E_TAGMISMATCH
 - xmlErrCodes, 100
- xmlErrCodes
 - XML_E_BASE, 99
 - XML_E_ELEMMISRQ, 99
 - XML_E_ELEMSISRQ, 99
 - XML_E_NOMATCH, 99
 - XML_E_NSURINOTFOU, 100
 - XML_E_TAGMISMATCH, 100