XBinder XML Runtime

Version 3.0

Objective Systems, Inc.

December 2024

XBinder XML Runtime

Author's Contact Information. Comments, suggestions, and inquiries regarding XBinder or this document may be sent by electronic mail to `<info@obj-sys.com>`.

Chapter 1. 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. C DOM Runtime Library Functions

The **C run-time DOM library** contains functions used to encode/decode XML data in a DOM tree. These functions are identified by their *rtDom* prefixes.

The categories of functions provided are as follows:

- Functions to add nodes.
- Functions to add attributes to a node.
- Functions to add namespace and xsi:type information.
- Functions to encode data types to DOM.

Chapter 3. Module Documentation

XML decode functions.

Detailed Description

Functions

- EXTERNXML int rtXmlDecBase64Binary (OSRTMEMBUF * pMemBuf, const OSUTF8CHAR * inpdata, OSSIZE 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 rtXmlDecBase64Str64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlDecBase64Str except that is supports a 64-bit integer length on 64-bit systems.

- EXTERNXML int rtXmlDecBase64StrValue (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnocts, OSSIZE bufSize, OSSIZE srcDataLen)

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

- EXTERNXML int rtXmlDecBase64StrValue64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufSize, OSSIZE srcDataLen)

  This function decodes is identical to rtXmlDecBase64StrValue except that it supports a 64-bit integer length on 64-bit systems.

- 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 rtXmlDecDynBase64Str64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlDecDynBase64Str except that it supports a 64-bit integer length on 64-bit systems.

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

  This function decodes a contents of a hexBinary string.

- EXTERNXML int rtXmlDecDynHexStr64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlDecDynHexStr except that it supports a 64-bit integer length on 64-bit systems.

- EXTERNXML int rtXmlDecEmptyElement (OSCTXT * pctxt)

  This function is used to enforce a requirement that an element be empty.

- EXTERNXML int rtXmlDecUTF8Str (OSCTXT * pctxt, OSUTF8CHAR * outdata, OSSIZE 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, OSSIZE 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 * pnocts, OSINT32 bufsize)

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

- EXTERNXML int rtXmlDecHexStr64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlDecHexStr except that it supports a 64-bit integer length on 64-bit systems.

- EXTERNXML int rtXmlDecHexStrValue (OSCTXT * pctxt, const OSUTF8CHAR *const inpdata, OSSIZE nbytes, OSOCTET * pvalue, OSUINT32 * pnbits, OSINT32 bufsize)

- EXTERNXML int rtXmlDecHexStrValue64 (OSCTXT * pctxt, const OSUTF8CHAR *const inpdata, OSSIZE nbytes, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE 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 namespac 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.

Function Documentation

EXTERNXML int rtXmlDecBase64Binary (OSRTMEMBUF *pMemBuf, const OSUTF8CHAR *inpdata, OSSIZE 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.

Table 3.1. 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.

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.

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

Table 3.2. 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.

EXTERNXML int rtXmlDecBase64Str64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, OSSIZE bufsize)

This function is identical to rtXmlDecBase64Str except that is supports a 64-bit integer length on 64-bit systems.

Table 3.3. 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.

EXTERNXML int rtXmlDecBase64StrValue (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSSIZE bufSize, OSSIZE 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.

Table 3.4. 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.

EXTERNXML int rtXmlDecBase64StrValue64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, OSSIZE bufSize, OSSIZE srcDataLen)

This function decodes is identical to rtXmlDecBase64StrValue except that it supports a 64-bit integer length on 64-bit systems.

Table 3.5. 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.

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.

Table 3.6. 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.

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

This function decodes a variable of the boolean type.

Table 3.7. 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.

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.

Table 3.8. 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.

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.

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
```

Table 3.9. 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.

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.

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

Table 3.10. 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.

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.

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

Table 3.11. 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.

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.

Table 3.12. 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.

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.

Table 3.13. 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.

EXTERNXML int rtXmlDecDynBase64Str64 (OSCTXT *pctxt, OS-DynOctStr64 *pvalue)

This function is identical to rtXmlDecDynBase64Str except that it supports a 64-bit integer length on 64-bit systems.

Table 3.14. 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.

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.

Table 3.15. 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.

EXTERNXML int rtXmlDecDynHexStr64 (OSCTXT *pctxt, OSDynOc-tStr64 *pvalue)

This function is identical to rtXmlDecDynHexStr except that it supports a 64-bit integer length on 64-bit systems.

Table 3.16. 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.

EXTERNXML int rtXmlDecEmptyElement (OSCTXT *pctxt)

This function is used to enforce a requirement that an element be empty.

An error is returned in the current element has any element or character children. The last event must be the start tag.

Table 3.17. 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.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlDecUTF8Str (OSCTXT *pctxt, OSUTF8CHAR *outdata, OSSIZE 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.

Table 3.18. 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.

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.

Table 3.19. 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.

EXTERNXML int rtXmlDecHexBinary (OSRTMEMBUF *pMemBuf, const OSUTF8CHAR *inpdata, OSSIZE 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.

Table 3.20. 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.

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.

Table 3.21. 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.

EXTERNXML int rtXmlDecHexStr64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, OSSIZE bufsize)

This function is identical to rtXmlDecHexStr except that it supports a 64-bit integer length on 64-bit systems.

Table 3.22. 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.

EXTERNXML int rtXmlDecHexStrValue (OSCTXT *pctxt, const OSUTF8CHAR *const inpdata, OSSIZE nbytes, OSOCTET *pvalue, OSUINT32 *pnbits, OSINT32 bufsize)

EXTERNXML int rtXmlDecHexStrValue64 (OSCTXT *pctxt, const OSUTF8CHAR *const inpdata, OSSIZE nbytes, OSOCTET *pvalue, OSSIZE *pnbits, OSSIZE bufsize)

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.

Table 3.23. 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.

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.

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

Table 3.24. 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.

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.

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

Table 3.25. 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.

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.

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

Table 3.26. 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.

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.

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

Table 3.27. 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.

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.

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

Table 3.28. 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.

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.

Table 3.29. 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.

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.

Table 3.30. 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.

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.

Table 3.31. 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.

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.

Table 3.32. 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.

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.

Table 3.33. 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.

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.

Table 3.34. 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.

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.

Table 3.35. 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.

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

This function decodes an XML namespac 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.

Table 3.36. 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.

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

Table 3.37. 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.

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.

Table 3.38. 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.

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.

Table 3.39. 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.

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 characer 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.

Table 3.40. 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.

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.

Table 3.41. 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.

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

This function parses the initial tag from an XML message.

Table 3.42. 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.

XML encode functions.

Detailed Description

Functions

- 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, OSSIZE 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, OSSIZE attrNameLen)

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

- EXTERNXML int rtXmlEncBase64StrValue (OSCTXT * pctxt, OSSIZE nocts, const OSOCTET * value)

This function encodes a variable of the XSD base64Binary type.

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

  This function encodes a variable of the XSD integer type.

- EXTERNXML int rtXmlEncBigIntAttr (OSCTXT * pctxt, const OSUTF8CHAR * value, const OSUTF8CHAR * attrName, OSSIZE attrNameLen)

  This function encodes an XSD integer attribute value.

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

  This function encodes an XSD integer attribute value.

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

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

- EXTERNXML int rtXmlEncBitStringExt (OSCTXT * pctxt, OSSIZE nbits, const OSOCTET * value, OSSIZE dataSize, const OSOCTET * extValue, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

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

- EXTERNXML int rtXmlEncBinStrValue (OSCTXT * pctxt, OSSIZE 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, OSSIZE attrNameLen)

  This function encodes an XSD boolean attribute value.

- EXTERNXML int rtXmlEncCanonicalSort (OSCTXT * pctxt, OSCTXT * pBufCtxt, OSRTSList * pList)

  Sort (as for CXER, Canonical-XER) and encode previously encoded SET OF components.

- EXTERNXML int rtXmlEncComment (OSCTXT * pctxt, const OSUTF8CHAR * comment)

  This function encodes an XML comment.

- 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 an string.

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

  This function encodes a variable of the XSD 'time' type as an 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, OSSIZE 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, OSSIZE 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, OSSIZE attrNameLen, const OSDoubleFmt * pFmtSpec)

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

- EXTERNXML int rtXmlEncDoubleNormalValue (OSCTXT * pctxt, OSREAL value, const OSDoubleFmt * pFmtSpec, int defaultPrecision)

  This function encodes a normal (not +/-INF or NaN) value of the XSD double or float type.

- 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 enpty element tag value (\<elemName/\>).

- EXTERNXML int rtXmlEncEndDocument (OSCTXT * pctxt)

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

- EXTERNXML int rtXmlEncEndElement (OSCTXT * pctxt, const OSUTF8CHAR * elemName, OSXMLNamespace * 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, OSXMLNamespace * pNS, const OSDoubleFmt * pFmtSpec)

  This function encodes a variable of the XSD float type.

- EXTERNXML int rtXmlEncFloatAttr (OSCTXT * pctxt, OSREAL value, const OSUTF8CHAR * attrName, OSSIZE 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, OSSIZE 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, OSSIZE attrNameLen)

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

- EXTERNXML int rtXmlEncHexStrValue (OSCTXT * pctxt, OSSIZE 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, OSXML-Namespace * 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, OSSIZE 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 rtXmlEncUIntPattern (OSCTXT * pctxt, OSUINT32 value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)

- EXTERNXML int rtXmlEncUIntPatternValue (OSCTXT * pctxt, OSUINT32 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 rtXmlEncInt64Pattern (OSCTXT * pctxt, OSINT64 value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)
- EXTERNXML int rtXmlEncInt64Value (OSCTXT * pctxt, OSINT64 value)

  This function encodes a variable of the XSD integer type.
- EXTERNXML int rtXmlEncInt64PatternValue (OSCTXT * pctxt, OSINT64 value, const OSUTF8CHAR * pattern)
- EXTERNXML int rtXmlEncInt64Attr (OSCTXT * pctxt, OSINT64 value, const OSUTF8CHAR * attrName, OSSIZE attrNameLen)

  This function encodes a variable of the XSD integer type as an attribute (name="value").
- EXTERNXML int rtXmlEncNamedBits (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSSIZE 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, OSSIZE 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 rtXmlPrintNSAttrs (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, OSSIZE 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, OSSIZE nameLen, const OSUTF8CHAR * value, OSSIZE valueLen, OSSIZE 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, OSRTDList * pNSAttrs)

  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, OSSIZE 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, OSSIZE 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, OSSIZE nameLen, const OSUTF8CHAR * value, OSSIZE 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, OSSIZE 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 rtXmlEncUInt64Pattern (OSCTXT * pctxt, OSUINT64 value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)

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

  This function encodes a variable of the XSD integer type.

- EXTERNXML int rtXmlEncUInt64PatternValue (OSCTXT * pctxt, OSUINT64 value, const OSUTF8CHAR * pattern)

- EXTERNXML int rtXmlEncUInt64Attr (OSCTXT * pctxt, OSUINT64 value, const OSUTF8CHAR * attrName, OSSIZE 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 rtXmlEncXSITypeAttr2 (OSCTXT * pctxt, const OSUTF8CHAR * typeNsUri, const OSUTF8CHAR * typeName)

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

- EXTERNXML int rtXmlEncXSINilAttr (OSCTXT * pctxt)

  This function encodes an XML nil attribute (xsi:nil="true").

- 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, OSSIZE len)

- EXTERNXML int rtXmlSetEncBufPtr (OSCTXT * pctxt, OSOCTET * bufaddr, OSSIZE 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).

Macros

- #define rtxPrintNSAttrs rtXmlPrintNSAttrs(name,&data)

- #define rtXmlFinalizeMemBuf 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)

- #define rtXmlGetEncBufPtr (pctxt)->buffer.data

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

- #define rtXmlGetEncBufLen (pctxt)->buffer.byteIndex

  This macro returns the length of the encoded XML message.

Function Documentation

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>)

Table 3.43. 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.

EXTERNXML int rtXmlEncAnyStr (OSCTXT *pctxt, const OSUTF8CHAR *pvalue, const OSUTF8CHAR *elemName, OSXML-Namespace *pNS)

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>)

Table 3.44. Parameters

pctxt	Pointer to context block structure.
pvalue	Value to be encoded.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncAnyAttr (OSCTXT *pctxt, OSRTDList *pA-nyAttrList)

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

Table 3.45. Parameters

pctxt	Pointer to context block structure.
pAnyAttrList	List of attributes.

Returns: . Completion status of operation:

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

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

This function encodes a variable of the XSD base64Binary type.

Table 3.46. 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.

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

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

Table 3.47. 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.

EXTERNXML int rtXmlEncBase64StrValue (OSCTXT *pctxt, OSSIZE 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.

Table 3.48. 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.

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 radixes are currently not supported.

Table 3.49. 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.

EXTERNXML int rtXmlEncBigIntAttr (OSCTXT *pctxt, const OSUTF8CHAR *value, const OSUTF8CHAR *attrName, OSSIZE 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).

Table 3.50. 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.

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.

Table 3.51. 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.

EXTERNXML int rtXmlEncBitString (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *value, const OSUTF8CHAR *elemName, OSXML-Namespace *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).

Table 3.52. 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.

EXTERNXML int rtXmlEncBitStringExt (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *value, OSSIZE dataSize, const OSOCTET *extValue, 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 used for BIT STRINGs with extdata member present.

Table 3.53. Parameters

pctxt	Pointer to context block structure.
nbits	Number of bits in the bit string.
value	Value to be encoded.
dataSize	Size of data member.
extValue	Value of extdata 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.

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

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

Table 3.54. 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.

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

This function encodes a variable of the XSD boolean type.

Table 3.55. 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.

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.

Table 3.56. 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.

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

This function encodes an XSD boolean attribute value.

Table 3.57. 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.

EXTERNXML int rtXmlEncCanonicalSort (OSCTXT *pctxt, OSCTXT *pBufCtxt, OSRTSList *pList)

Sort (as for CXER, Canonical-XER) and encode previously encoded SET OF components.

Table 3.58. Parameters

pctxt	Context to use to encode the sorted encoding
pBufCtxt	Context previously used to encode the components which are to be sorted.
pList	List of OSRTBufLocDescr identifying the location of each of the components' encodings within pBufCtxt.

EXTERNXML int rtXmlEncComment (OSCTXT *pctxt, const OSUTF8CHAR *comment)

This function encodes an XML comment.

The given text will be inserted in between XML comment start and end elements ().

Table 3.59. Parameters

pctxt	Pointer to context block structure.
comment	The comment text.

Returns: . Completion status of operation:

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

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.

Table 3.60. 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.

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.

Table 3.61. 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.

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

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

Table 3.62. 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.

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

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

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

Table 3.63. 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.

EXTERNXML int rtXmlEncDateTime (OSCTXT *pctxt, const OSXSD-DateTime *pvalue, const OSUTF8CHAR *elemName, OSXMLName-space *pNS)

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

Table 3.64. 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.

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.

Table 3.65. 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.

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.

Table 3.66. 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.

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

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

Table 3.67. 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.

EXTERNXML int rtXmlEncDecimalValue (OSCTXT *pctxt, OSREAL value, const OSDecimalFmt *pFmtSpec, char *pDestBuf, OSSIZE 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.

Table 3.68. 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.

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.

Table 3.69. 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.

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

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

Table 3.70. 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.

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

This function encodes a normal (not +/-INF or NaN) value of the XSD double or float type.

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

Table 3.71. 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.

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. Special real values +/-INF and NaN are encoded as "INF", "-INF", and "NaN", respectively.

Table 3.72. 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.

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

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

Table 3.73. 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.

EXTERNXML int rtXmlEncEndDocument (OSCTXT *pctxt)

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

Table 3.74. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

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

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

Table 3.75. 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.

EXTERNXML int rtXmlEncEndSoapEnv (OSCTXT *pctxt)

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

Table 3.76. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncEndSoapElems (OSCTXT *pctxt, OSXM-LSOAPMsgType 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.

Table 3.77. 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.

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.

Table 3.78. 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.

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

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

Table 3.79. 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.

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.

Table 3.80. 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.

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.

Table 3.81. 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.

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.

Table 3.82. 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.

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.

Table 3.83. 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.

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.

Table 3.84. 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.

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.

Table 3.85. 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.

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.

Table 3.86. 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.

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.

Table 3.87. 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.

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.

Table 3.88. 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.

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.

Table 3.89. 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.

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

This function encodes a variable of the XSD hexBinary type.

Table 3.90. 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.

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

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

Table 3.91. 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.

EXTERNXML int rtXmlEncHexStrValue (OSCTXT *pctxt, OSSIZE 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.

Table 3.92. 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.

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 OSXM-LINDENT constant value.

Table 3.93. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

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.

Table 3.94. 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.

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.

Table 3.95. Parameters

pctxt	Pointer to context block structure.
value	Value to be encoded.

Returns: . Completion status of operation:

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

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

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

Table 3.96. 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.

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.

Table 3.97. 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.

EXTERNXML int rtXmlEncIntPatternValue (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *pattern)

EXTERNXML int rtXmlEncUIntPattern (OSCTXT *pctxt, OSUINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const OSUTF8CHAR *pattern)

EXTERNXML int rtXmlEncUIntPatternValue (OSCTXT *pctxt, OSUINT32 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.

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

Table 3.98. 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.

EXTERNXML int rtXmlEncInt64Pattern (OSCTXT *pctxt, OSINT64 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const OSUTF8CHAR *pattern)

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.

Table 3.99. Parameters

pctxt	Pointer to context block structure.
value	Value to be encoded.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncInt64PatternValue (OSCTXT *pctxt, OSINT64 value, const OSUTF8CHAR *pattern)

EXTERNXML int rtXmlEncInt64Attr (OSCTXT *pctxt, OSINT64 value, const OSUTF8CHAR *attrName, OSSIZE 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.

Table 3.100. 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.

EXTERNXML int rtXmlEncNamedBits (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSSIZE 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 identifers corresponding to each set bit in the bit string value.

Table 3.101. Parameters

pctxt	Pointer to context block structure.
pBitMap	Bit map equating symbolic bit names to bit numbers.
nbits	Number of bits in the sit 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.

EXTERNXML int rtXmlEncNamedBitsValue (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSSIZE nbits, const OSOCTET *pvalue)

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.

Table 3.102. 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.

EXTERNXML int rtXmlPrintNSAttrs (const char *name, const OSRT-DList *data)

This function prints a list of namespace attributes.

Table 3.103. Parameters

name	Name to print.
data	List of namespace attributes.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncReal10 (OSCTXT *pctxt, const OSUTF8CHAR *pvalue, const OSUTF8CHAR *elemName, OSXML-Namespace *pNS)

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

Table 3.104. 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.

EXTERNXML int rtXmlEncSoapArrayTypeAttr (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value, OSSIZE item-Count)

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]"'.

Table 3.105. 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.

EXTERNXML int rtXmlEncSoapArrayTypeAttr2 (OSCTXT *pctxt, const OSUTF8CHAR *name, OSSIZE nameLen, const OSUTF8CHAR *value, OSSIZE valueLen, OSSIZE itemCount)

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"?>

Table 3.106. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

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.

Table 3.107. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

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

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

Table 3.108. Parameters

pctxt	Pointer to context block structure.
elemName	XML element name. Empty string and null are treated equivalently.
pNS	XML namespace information (prefix and URI). If the prefix is NULL, this method will search the context's namespace stack for a prefix to use.
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.

EXTERNXML int rtXmlEncStartSoapEnv (OSCTXT *pctxt, OSRTDList *pNSAttrs)

This function encodes a SOAP envelope start element tag.

This includes all of the standard SOAP namespace attributes.

Table 3.109. Parameters

pctxt	Pointer to context block structure.
pNSAttrs	List of namespace attributes to be added to SOAP envelope.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncStartSoapElems (OSCTXT *pctxt, OSXM-LSOAPMsgType 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.

Table 3.110. 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.

EXTERNXML int rtXmlEncString (OSCTXT *pctxt, OSXMLSTRING *pxmlstr, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function encodes a variable of the XSD string type.

Table 3.111. Parameters

pctxt	Pointer to context block structure.
pxmlstr	XML string value to be encoded.
elemName	XML element name. If either null or 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.

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

This function encodes a variable of the XSD string type.

Table 3.112. 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.

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

This function encodes a variable of the XSD string type.

Table 3.113. 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.

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.

Table 3.114. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

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

This function encodes a Unicode string value.

Table 3.115. 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.

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.

Table 3.116. 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.

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

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

Table 3.117. 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.

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

This function encodes a UTF-8 string value.

Table 3.118. 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.

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

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

Table 3.119. 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.

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.

Table 3.120. Parameters

pctxt	Pointer to context block structure.
value	Value to be encoded.

Returns: . Completion status of operation:

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

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

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

Table 3.121. 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.

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.

Table 3.122. 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.

EXTERNXML int rtXmlEncUInt64Pattern (OSCTXT *pctxt, OSUINT64 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, const OSUTF8CHAR *pattern)

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.

Table 3.123. Parameters

pctxt	Pointer to context block structure.
value	Value to be encoded.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncUInt64PatternValue (OSCTXT *pctxt, OSUINT64 value, const OSUTF8CHAR *pattern)

EXTERNXML int rtXmlEncUInt64Attr (OSCTXT *pctxt, OSUINT64 value, const OSUTF8CHAR *attrName, OSSIZE 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.

Table 3.124. 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.

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.

Table 3.125. 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.

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

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

Table 3.126. Parameters

pctxt	Pointer to context block structure.
value	XSI type attribute value.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncXSITypeAttr2 (OSCTXT *pctxt, const OSUTF8CHAR *typeNsUri, const OSUTF8CHAR *typeName)

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

This will encode namespace declarations for the xsi namespace and for the typeNsUri namespace, if needed.

Table 3.127. Parameters

pctxt	Pointer to context block structure.
typeNsUri	The type's namespace URI. Either null or empty if none.
typeName	The type's name.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlEncXSINilAttr (OSCTXT *pctxt)

This function encodes an XML nil attribute (xsi:nil="true").

Table 3.128. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

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'.

Table 3.129. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

EXTERNXML OSBOOL rtXmlStrCmpAsc (const OSUTF8CHAR *text1, const char *text2)

EXTERNXML OSBOOL rtXmlStrnCmpAsc (const OSUTF8CHAR *text1, const char *text2, OSSIZE len)

EXTERNXML int rtXmlSetEncBufPtr (OSCTXT *pctxt, OSOCTET *bufaddr, OSSIZE 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.

This function should not be called with a context that has an associated stream open, but if it is, the stream may be automatically closed.

Table 3.130. 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).

EXTERNXML int rtXmlGetIndent (OSCTXT *pctxt)

This function returns current XML output indent value.

Table 3.131. Parameters

pctxt	Pointer to OSCTXT structure

Returns: . Current indent value (>= 0) if OK, negative status code if error.

EXTERNXML OSBOOL rtXmlGetWriteBOM (OSCTXT *pctxt)

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

Table 3.132. Parameters

pctxt	Pointer to OSCTXT structure.

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

EXTERNXML int rtXmlGetIndentChar (OSCTXT *pctxt)

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

Table 3.133. Parameters

pctxt	Pointer to OSCTXT structure

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

Macro Definition Documentation

#define rtXmlGetEncBufPtr

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.

Table 3.134. Parameters

pctxt	Pointer to a context structure.

Definition at line 2650 of file osrtxml.h

The Documentation for this define was generated from the following file:

- osrtxml.h

#define rtXmlGetEncBufLen

Table 3.135. Parameters

pctxt	Pointer to a context structure.

Definition at line 2657 of file osrtxml.h

The Documentation for this define was generated from the following file:

- osrtxml.h

XML utility functions.

Detailed Description

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 int rtXmlWriteUTF16ToFile (OSCTXT * pctxt, const char * filename)
- EXTERNXML void rtXmlTreatWhitespaces (OSCTXT * pctxt, int whiteSpaceType)
- EXTERNXML int rtXmlCheckBuffer (OSCTXT * pctxt, OSSIZE byte_count)

Function Documentation

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.

Table 3.136. 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.

EXTERNXML int rtXmlWriteUTF16ToFile (OSCTXT *pctxt, const char *filename)

EXTERNXML void rtXmlTreatWhitespaces (OSCTXT *pctxt, int whiteSpaceType)

EXTERNXML int rtXmlCheckBuffer (OSCTXT *pctxt, OSSIZE byte_count)

XML pull-parser decode functions.

Detailed Description

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 rtXmlpDecAny2 (OSCTXT * pctxt, OSUTF8CHAR ** pvalue)

  This version of the rtXmlpDecAny function returns the string in a mutable buffer.

- EXTERNXML int rtXmlpDecAnyAttrStr (OSCTXT * pctxt, const OSUTF8CHAR ** ppAttrStr, OSSIZE attrIndex)

  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, OSSIZE bufsize)

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

- EXTERNXML int rtXmlpDecBase64Str64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlpDecBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

- 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 rtXmlpDecBitString64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE bufsize)

  This function is identical to rtXmlpDecBitString except that it supports lengths up to 64-bits in size on 64-bit machines.

- EXTERNXML int rtXmlpDecBitStringExt (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnbits, OSOCTET ** ppextdata, OSUINT32 bufsize)

  This function decodes a bit string value.

- EXTERNXML int rtXmlpDecBitStringExt64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnbits, OSOCTET ** ppextdata, OSSIZE bufsize)

  This function is identical to rtXmlpDecBitStringExt except that it supports lengths up to 64-bits in size on 64-bit machines.

- 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 fractionDigits)

  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 rtXmlpDecDoubleExt (OSCTXT * pctxt, OSUINT8 flags, 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 rtXmlpDecDynBase64Str64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlpDecDynBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

- 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 rtXmlpDecDynHexStr64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to the rtXmlpDecDynHexStr except that it supports lengths up to 64 bits in size on 64-bit systems.

- EXTERNXML int rtXmlpDecDynUnicodeStr (OSCTXT * pctxt, const OSUNICHAR ** ppdata, OSSIZE * pn-chars)

  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, OSSIZE 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, OSSIZE bufsize)

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

- EXTERNXML int rtXmlpDecHexStr64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlpDecHexStr except that it supports lengths up to 64-bits in size on 64-bit machines.

- 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 rtXmlpDecNamedBits64 (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE 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 rtXmlpDecXSIAttr (OSCTXT * pctxt, const OSXMLNameFragments * attrName)

  This function decodes XSI (XML Schema Instance) 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, OSSIZE 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, OSSIZE 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 * pctxt, OSXMLNameFragments * pAttr, OSINT16 * nsidx, OSSIZE attrIndex)

  This function selects attribute to decode.

- EXTERNXML OSINT32 rtXmlpGetCurrentLevel (OSCTXT * pctxt)

  This function returns current nesting level.

- EXTERNXML void rtXmlpSetWhiteSpaceMode (OSCTXT * pctxt, OSXMLWhiteSpaceMode whiteSpaceMode)

  Sets the whitespace treatment mode.

- EXTERNXML OSBOOL rtXmlpSetMixedContentMode (OSCTXT * pctxt, OSBOOL mixedContentMode)

  Sets mixed content mode.

- EXTERNXML void rtXmlpSetListMode (OSCTXT * pctxt)

  Sets list mode.

- EXTERNXML OSBOOL rtXmlpListHasItem (OSCTXT * pctxt)

  Check for end of decoded token list.

- EXTERNXML void rtXmlpCountListItems (OSCTXT * pctxt, OSSIZE * itemCnt)

  Count tokens in list.

- EXTERNXML int rtXmlpGetNextSeqElemID2 (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXMLGroupDesc * pGroup, int groups, int curID, int lastMandatoryID, OSBOOL groupMode, OSBOOL check-Repeat)

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

- EXTERNXML int rtXmlpGetNextSeqElemID (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXML-GroupDesc * pGroup, int curID, int lastMandatoryID, OSBOOL groupMode)

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

- EXTERNXML int rtXmlpGetNextSeqElemIDExt (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXMLGroupDesc * ppGroup, const OSBOOL * extRequired, int postExtRootID, int curID, int lastMandatoryID, OSBOOL groupMode)

  This is an ASN.1 extension-supporting version of rtXmlpGetNextSeqElemID.

- EXTERNXML int rtXmlpGetNextAllElemID (OSCTXT * pctxt, const OSXMLElemIDRec * tab, OSSIZE nrows, const OSUINT8 * pOrder, OSSIZE nOrder, OSSIZE maxOrder, int anyID)

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

- EXTERNXML int rtXmlpGetNextAllElemID16 (OSCTXT * pctxt, const OSXMLElemIDRec * tab, OSSIZE nrows, const OSUINT16 * pOrder, OSSIZE nOrder, OSSIZE maxOrder, int anyID)

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

- EXTERNXML int rtXmlpGetNextAllElemID32 (OSCTXT * pctxt, const OSXMLElemIDRec * tab, OSSIZE nrows, const OSUINT32 * pOrder, OSSIZE nOrder, OSSIZE maxOrder, int anyID)

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

- EXTERNXML void rtXmlpSetNamespaceTable (OSCTXT * pctxt, const OSUTF8CHAR * namespaceTable, OSSIZE nmNamespaces)

  Sets user namespace table.

- EXTERNXML int rtXmlpCreateReader (OSCTXT * pctxt)

  Creates pull parser reader structure within the context.

- EXTERNXML void rtXmlpHideAttributes (OSCTXT * pctxt)

  Disable access to attributes.

- EXTERNXML OSBOOL rtXmlpNeedDecodeAttributes (OSCTXT * pctxt)

  This function checks if attributes were previously decoded.

- EXTERNXML void rtXmlpMarkPos (OSCTXT * pctxt)

  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, OSSIZE * 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 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 struct OSXMLReader * rtXmlpGetReader (OSCTXT * pctxt)

  This function fetches the XML reader structure from the context for use in low-level pull parser calls.

- EXTERNXML OSBOOL rtXmlpIsLastEventDone (OSCTXT * pctxt)

  Check processing status of current tag.

- EXTERNXML int rtXmlpGetXSITypeIndex (OSCTXT * pctxt, const OSXMLItemDescr typetab, OSSIZE typetabsiz)

  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, OSINT16 xsiTypeIdx, const OSXMLItemDescr typetab, OSSIZE 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.

- EXTERNXML OSBOOL rtXmlpIsUTF8Encoding (OSCTXT * pctxt)

  This function checks if the encoding specified in XML header is UTF-8.

- EXTERNXML int rtXmlpReadBytes (OSCTXT * pctxt, OSOCTET * pbuf, OSSIZE nbytes)

  This function reads the specified number of bytes directly from the underlying XML parser stream.

Macros

- #define OSXMLREALENC_OBJSYS 0x1F /* Obj-Sys XML encoding rules */
- #define OSXMLREALENC_BXER 0x10 /* basic-XER */
- #define OSXMLREALENC_EXERMODS 0x1B /* extended-XER with MODIFIED-ENCODINGS */
- #define OSXMLREALENC_EXERDECIMAL 0x03 /* extended-XER with DECIMAL */

Function Documentation

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.

Table 3.137. 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.

EXTERNXML int rtXmlpDecAny2 (OSCTXT *pctxt, OSUTF8CHAR **pvalue)

This version of the rtXmlpDecAny function returns the string in a mutable buffer.

Table 3.138. 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.

EXTERNXML int rtXmlpDecAnyAttrStr (OSCTXT *pctxt, const OSUTF8CHAR **ppAttrStr, OSSIZE attrIndex)

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.

Table 3.139. 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.
attrIndex	Index of attribute.

Returns: . Completion status of operation:

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

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.

Table 3.140. 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.

EXTERNXML int rtXmlpDecBase64Str (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSSIZE 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.

Table 3.141. 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.

EXTERNXML int rtXmlpDecBase64Str64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, OSSIZE bufsize)

This function is identical to rtXmlpDecBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

Table 3.142. 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.

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.

Table 3.143. 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.

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.

Table 3.144. 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.

EXTERNXML int rtXmlpDecBitString64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnbits, OSSIZE bufsize)

This function is identical to rtXmlpDecBitString except that it supports lengths up to 64-bits in size on 64-bit machines.

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.

Table 3.145. 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.

See also: . rtXmlpDecBitString

EXTERNXML int rtXmlpDecBitStringExt (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnbits, OSOCTET **ppextdata, 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. This version supports BIT STRINGs with the extdata member present.

Table 3.146. 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.
ppextdata	Pointer to byte array containing extension data.
bufsize	Size of buffer passed in pvalue argument.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlpDecBitStringExt64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnbits, OSOCTET **ppextdata, OSSIZE bufsize)

This function is identical to rtXmlpDecBitStringExt except that it supports lengths up to 64-bits in size on 64-bit machines.

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. This version supports BIT STRINGs with the extdata member present.

Table 3.147. 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.
ppextdata	Pointer to byte array containing extension data.
bufsize	Size of buffer passed in pvalue argument.

Returns: . Completion status of operation:

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

See also: . rtXmlpDecBitStringExt

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.

Table 3.148. 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.

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.

Table 3.149. 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.

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.

Table 3.150. 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.

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.

Table 3.151. 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.

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.

Table 3.152. 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.

EXTERNXML int rtXmlpDecDoubleExt (OSCTXT *pctxt, OSUINT8 flags, 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.

Table 3.153. Parameters

pctxt	Pointer to context block structure.
flags	Specifies alternatives allowed in the lexical value. See OSXMLREALENC* constants. bit 0 (rightmost) set: recognize INF, -INF, NaN text values bit 1 set: recognize leading '+' on normal reals bit 2 set: recognize leading '+' on INF bit 3 set: recognize leading zeros in exponent bit 4 set: recognize exponents

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

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlpDecDynBase64Str (OSCTXT *pctxt, OS-DynOctStr *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.

Table 3.154. 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.

EXTERNXML int rtXmlpDecDynBase64Str64 (OSCTXT *pctxt, OS-DynOctStr64 *pvalue)

This function is identical to rtXmlpDecDynBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

Table 3.155. 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.

EXTERNXML int rtXmlpDecDynBitString (OSCTXT *pctxt, OSDynOc-tStr *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.

Table 3.156. 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.

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 OSUTF8CHAR characters returned by an XML pull parser.

Table 3.157. 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.

EXTERNXML int rtXmlpDecDynHexStr64 (OSCTXT *pctxt, OSDynOctStr64 *pvalue)

This function is identical to the rtXmlpDecDynHexStr except that it supports lengths up to 64 bits in size on 64-bit systems.

Table 3.158. 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.

EXTERNXML int rtXmlpDecDynUnicodeStr (OSCTXT *pctxt, const OSUNICHAR **ppdata, OSSIZE *pnchars)

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.

Table 3.159. 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.
pnchars	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.

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.

Table 3.160. 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.

EXTERNXML int rtXmlpDecUTF8Str (OSCTXT *pctxt, OSUTF8CHAR *out, OSSIZE 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.

Table 3.161. 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.

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.

Table 3.162. 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.

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.

Table 3.163. 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.

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.

Table 3.164. 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.

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.

Table 3.165. 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.

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.

Table 3.166. 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.

EXTERNXML int rtXmlpDecHexStr (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSSIZE 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.

Table 3.167. 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.

EXTERNXML int rtXmlpDecHexStr64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, OSSIZE bufsize)

This function is identical to rtXmlpDecHexStr except that it supports lengths up to 64-bits in size on 64-bit machines.

Table 3.168. 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.

See also: . rtXmlpDecHexStr

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.

Table 3.169. 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.

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.

Table 3.170. 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.

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.

Table 3.171. 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.

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.

Table 3.172. 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.

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.

Table 3.173. Parameters

pctxt	Pointer to context block structure.
pBitMap	Pointer to bit map structure that defined token to bit mappings.
pvalue	Pointer to buffer to recieve 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.

EXTERNXML int rtXmlpDecNamedBits64 (OSCTXT *pctxt, const OSBitMapItem *pBitMap, OSOCTET *pvalue, OSSIZE *pnbits, OSSIZE 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.

Table 3.174. Parameters

pctxt	Pointer to context block structure.
pBitMap	Pointer to bit map structure that defined token to bit mappings.
pvalue	Pointer to buffer to recieve 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.

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.

Table 3.175. 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.

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
```

Table 3.176. 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.

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.

Table 3.177. 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.

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.

Table 3.178. 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.

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.

Table 3.179. 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.

EXTERNXML int rtXmlpDecUInt64 (OSCTXT *pctxt, OSUINT64 *pval-ue)

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.

Table 3.180. 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.

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 characer 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.

Table 3.181. 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.

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.

Table 3.182. 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.

EXTERNXML int rtXmlpDecXSIAttr (OSCTXT *pctxt, const OSXML-NameFragments *attrName)

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

Table 3.183. 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.

EXTERNXML int rtXmlpDecXSITypeAttr (OSCTXT *pctxt, const OSXMLNameFragments *attrName, const OSUTF8CHAR **ppAttr-Value)

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

Prior to calling this function, use rtXmlpSelectAttribute to select the attribute. After calling this function, if you want to read the same attribute again, you will need to call rtXmlpSelectAttribute again.

Table 3.184. 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.

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

This function finds an attribute in the descriptor table.

Table 3.185. 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.

EXTERNXML int rtXmlpGetNextElem (OSCTXT *pctxt, OSXMLElemDescr *pElem, OSINT32 level)

This function parse the next element start tag.

Table 3.186. 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.

EXTERNXML int rtXmlpGetNextElemID (OSCTXT *pctxt, const OSXMLElemIDRec *tab, OSSIZE nrows, OSINT32 level, OSBOOL continueParse)

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

If this reaches the closing tag for the current level, it returns XML_OK_EOB. If this encounters an unexpected element and !continueParse, it returns RTERR_UNEXPELEM (without logging it).

Table 3.187. 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; skipped elements are logged, but an error is not returned.

Returns: . Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

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.

Table 3.188. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

EXTERNXML int rtXmlpMatchStartTag (OSCTXT *pctxt, const OSUTF8CHAR *elemLocalName, OSINT16 nsidx)

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

Table 3.189. 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.

EXTERNXML int rtXmlpMatchEndTag (OSCTXT *pctxt, OSINT32 level)

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

Table 3.190. 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.

EXTERNXML OSBOOL rtXmlpHasAttributes (OSCTXT *pctxt)

This function checks accessibility of attributes.

Table 3.191. 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.

EXTERNXML int rtXmlpGetAttributeCount (OSCTXT *pctxt)

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

Table 3.192. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

- zero or positive value is atributes number,
- negative return value is error.

EXTERNXML int rtXmlpSelectAttribute (OSCTXT *pctxt, OSXML-NameFragments *pAttr, OSINT16 *nsidx, OSSIZE attrIndex)

This function selects attribute to decode.

Table 3.193. 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 (see enum OSXMLNsIndex)
attrIndex	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.

EXTERNXML OSINT32 rtXmlpGetCurrentLevel (OSCTXT *pctxt)

This function returns current nesting level.

Table 3.194. Parameters

pctxt	Pointer to context block structure.

Returns: . Current nesting level.

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.

Table 3.195. Parameters

pctxt	Pointer to context block structure.
whiteSpaceMode	White space mode.

Returns: . Previously set whitespace mode.

EXTERNXML OSBOOL rtXmlpSetMixedContentMode (OSCTXT *pc-txt, OSBOOL mixedContentMode)

Sets mixed content mode.

Table 3.196. Parameters

pctxt	Pointer to context block structure.
mixedContentMode	Enable/disable mixed mode.

Returns: . Previously state of mixed mode.

EXTERNXML void rtXmlpSetListMode (OSCTXT *pctxt)

Sets list mode.

Attribute value or element content is decoded by tokens.

Table 3.197. Parameters

pctxt	Pointer to context block structure.

EXTERNXML OSBOOL rtXmlpListHasItem (OSCTXT *pctxt)

Check for end of decoded token list.

Table 3.198. Parameters

pctxt	Pointer to context block structure.

Returns: . State of decoded list:

- TRUE = list is not finished,

- FALSE = all tokens was decoded.

EXTERNXML void rtXmlpCountListItems (OSCTXT *pctxt, OSSIZE *itemCnt)

Count tokens in list.

Table 3.199. 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.

EXTERNXML int rtXmlpGetNextSeqElemID2 (OSCTXT *pctxt, const OSXMLElemIDRec *tab, const OSXMLGroupDesc *pGroup, int groups, int curID, int lastMandatoryID, OSBOOL groupMode, OSBOOL checkRepeat)

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

It is used to decode sequences in strict mode.

Table 3.200. Parameters

pctxt	Pointer to context block structure.
tab	Elements descriptor table.
pGroup	Decode groups table.
groups	Number of groups in groups table. If checkRepeat is FALSE, you can pass 0 for this if the value is unknown.
curID	Current decode group.
lastMandatoryID	Identifier of last mandatory element.
groupMode	This parameter must be setted to TRUE when decode groups or base types.
checkRepeat	If true, this method is being called to check for a repeat of curID. In this case, we do not treat curID as being required. Otherwise, curID is required if curId <= lastMandatoryID.

Returns: . Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

EXTERNXML int rtXmlpGetNextSeqElemID (OSCTXT *pctxt, const OSXMLElemIDRec *tab, const OSXMLGroupDesc *pGroup, int curID, int lastMandatoryID, OSBOOL groupMode)

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

It is used to decode sequences in strict mode.

Handling of unexpected elements:

- If the current decode group includes an any case, unexpected elements will be matched against it (the any case id will be returned).
- Otherwise, if groupMode is true, and the element identified by lastMandatoryID has been reached, XML_OK_EOB is returned. The unexpected element may belong to something following the elements in tab.
- If neither of the above applies, unknown elements will be logged and skipped over, with this method continuing as if the unexpected element were not present. Handling of the case of reaching closing tag for current level:
- If the last mandatory element is reached, return XML_OK_EOB.
- Otherwise, log and return XML_E_ELEMSMISRQ.

This function is equivalent to: rtXmlpGetNextSeqElemID2(pctxt, tab, pGroup, curID, lastMandatoryID, groupMode, FALSE);

Table 3.201. 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 decoding groups or base types.

Returns: . Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

EXTERNXML int rtXmlpGetNextSeqElemIDExt (OSCTXT *pctxt, const OSXMLElemIDRec *tab, const OSXMLGroupDesc *ppGroup, const OSBOOL *extRequired, int postExtRootID, int curID, int lastMandatoryID, OSBOOL groupMode)

This is an ASN.1 extension-supporting version of rtXmlpGetNextSeqElemID.

It allows required extension elements to be absent, except that when an extension group is present, all required extension elements in that group must be present. It otherwise behaves the same as rtXmlpGetNextSeqElemID.

Table 3.202. Parameters

pctxt	Pointer to context block structure.

tab	Elements descriptor table.
pGroup	Decode groups table.
extRequired	extRequired[i] corresponds to pGroup[i]. This is TRUE if and only if the decode group ends with a non-optional, non-default extension element that must be present in the encoding (because it is inside an extension group for which some element has already been decoded).
postExtRootID	This is the group id corresponding to the decode group that begins with the first root element that follows the extension additions. If there is no such element, this is -1.
curID	Current decode group.
lastMandatoryID	Identifier of last mandatory element.
groupMode	This parameter must be setted to TRUE when decoding groups or base types.

Returns: . Completion status of operation:

- positive or zero value is element identifier,
- negative return value is error.

EXTERNXML int rtXmlpGetNextAllElemID (OSCTXT *pctxt, const OSXMLElemIDRec *tab, OSSIZE nrows, const OSUINT8 *pOrder, OSSIZE nOrder, OSSIZE 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.

Table 3.203. 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.

EXTERNXML int rtXmlpGetNextAllElemID16 (OSCTXT *pctxt, const OSXMLElemIDRec *tab, OSSIZE nrows, const OSUINT16 *pOrder, OSSIZE nOrder, OSSIZE 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.

Table 3.204. 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.

EXTERNXML int rtXmlpGetNextAllElemID32 (OSCTXT *pctxt, const OSXMLElemIDRec *tab, OSSIZE nrows, const OSUINT32 *pOrder, OSSIZE nOrder, OSSIZE 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.

Table 3.205. 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.

EXTERNXML void rtXmlpSetNamespaceTable (OSCTXT *pctxt, const OSUTF8CHAR *namespaceTable[], OSSIZE nmNamespaces)

Sets user namespace table.

Table 3.206. Parameters

pctxt	Pointer to context block structure.
namespaceTable	Array of namespace URI strings.
nmNamespaces	Number of namespaces in table.

EXTERNXML int rtXmlpCreateReader (OSCTXT *pctxt)

Creates pull parser reader structure within the context.

Table 3.207. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

EXTERNXML void rtXmlpHideAttributes (OSCTXT *pctxt)

Disable access to attributes.

Function used in derived types to disable repeated decode in base type.

Table 3.208. Parameters

pctxt	Pointer to context block structure.

EXTERNXML OSBOOL rtXmlpNeedDecodeAttributes (OSCTXT *pctxt)

This function checks if attributes were previously decoded.

Table 3.209. Parameters

pctxt	Pointer to context block structure.

Returns: . State of attributes:

- TRUE = attributes was not decoded,
- FALSE = attributes had been decoded.

EXTERNXML void rtXmlpMarkPos (OSCTXT *pctxt)

Save current decode position.

Table 3.210. Parameters

pctxt	Pointer to context block structure.

EXTERNXML void rtXmlpRewindToMarkedPos (OSCTXT *pctxt)

Rewind to saved decode position.

Table 3.211. Parameters

pctxt	Pointer to context block structure.

EXTERNXML void rtXmlpResetMarkedPos (OSCTXT *pctxt)

Reset saved decode position.

Table 3.212. Parameters

pctxt	Pointer to context block structure.

EXTERNXML int rtXmlpGetXSITypeAttr (OSCTXT *pctxt, const OSUTF8CHAR **ppAttrValue, OSINT16 *nsidx, OSSIZE *pLocalOffs)

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

Table 3.213. 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.

EXTERNXML int rtXmlpGetXmlnsAttrs (OSCTXT *pctxt, OSRTDList *pNSAttrs)

This function decodes namespace attributes from start tag and adds them to the given list.

Table 3.214. 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.

EXTERNXML int rtXmlpDecXSIAttrs (OSCTXT *pctxt)

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

Table 3.215. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

EXTERNXML OSBOOL rtXmlpIsEmptyElement (OSCTXT *pctxt)

Check element content: empty or not.

Table 3.216. Parameters

pctxt	Pointer to context block structure.

Returns: . Status of element content:

- TRUE = element is empty,
- FALSE = element has content.

EXTERNXML int rtXmlEncAttrC14N (OSCTXT *pctxt)

This function used only in C14 mode.

It provide atributes sorting.

Table 3.217. Parameters

pctxt	Pointer to context block structure.

Returns: . Completion status of operation:

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

EXTERNXML struct OSXMLReader* rtXmlpGetReader (OSCTXT *pc-txt)

This function fetches the XML reader structure from the context for use in low-level pull parser calls.

If the reader structure does not exist, it is created and initialized.

Table 3.218. Parameters

pctxt	Pointer to context block structure.

Returns: . Pointer to XML reader structure or NULL if an error occurred.

EXTERNXML OSBOOL rtXmlpIsLastEventDone (OSCTXT *pctxt)

Check processing status of current tag.

Table 3.219. Parameters

pctxt	Pointer to context block structure.

Returns: . Status of element content:

- TRUE = tag marked as processed,
- FALSE = tag will be processed again.

EXTERNXML int rtXmlpGetXSITypeIndex (OSCTXT *pctxt, const OSXMLItemDescr typetab[], OSSIZE typetabsiz)

This function decodes the contents of an XSI (XML Schema Instance) type attribute (xsi:type) and find type index in descriptor table.

Table 3.220. 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.

EXTERNXML int rtXmlpLookupXSITypeIndex (OSCTXT *pctxt, con-st OSUTF8CHAR *pXsiType, OSINT16 xsiTypeIdx, const OSXM-LItemDescr typetab[], OSSIZE typetabsiz)

This function find index of XSI (XML Schema Instance) type in descriptor table.

Table 3.221. 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.

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.

Table 3.222. Parameters

pctxt	Pointer to context block structure.

EXTERNXML OSBOOL rtXmlpIsDecodeAsGroup (OSCTXT *pctxt)

This function checks if "decode as group" mode was forced.

Table 3.223. Parameters

pctxt	Pointer to context block structure.

Returns: . State of mode:

- TRUE = need decode as group,
- FALSE = normal sequence decoding.

EXTERNXML OSBOOL rtXmlpIsUTF8Encoding (OSCTXT *pctxt)

This function checks if the encoding specified in XML header is UTF-8.

Table 3.224. Parameters

pctxt	Pointer to context block structure.

Returns: . State of mode:

- TRUE = is in UTF-8 encoding,
- FALSE = is not in UTF-8 encoding.

EXTERNXML int rtXmlpReadBytes (OSCTXT *pctxt, OSOCTET *pbuf, OSSIZE nbytes)

This function reads the specified number of bytes directly from the underlying XML parser stream.

It has no effect on any of the parser state variables.

Table 3.225. Parameters

pctxt	Pointer to context block structure.
pbuf	Pointer to static buffer (byte array) to receive data. The buffer must be at least large enough to hold the requested number of bytes.
nbytes	Number of bytes to read.

Returns: . State of mode:

- TRUE = is in UTF-8 encoding,
- FALSE = is not in UTF-8 encoding.

Macro Definition Documentation

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.

Detailed Description

In many cases, additional information and parameters for the different errors are stored in the context structure at the time the error in raised. This additional information can be output using the `rtxErrPrint` or `rtxErrLogUsingCB` run-time functions.

Macros

- #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_ELEMSMISRQ (XML_E_BASE-9)

  Missing required elements.

- #define XML_E_TOOFEWELEMS (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.

- #define XML_E_KEYNOTFOU (XML_E_BASE-16)

  Keyref constraint has some key that not present in refered constraint.

- #define XML_E_DUPLKEY (XML_E_BASE-17)

  Key or unique constraint has duplicated key.

- #define XML_E_FLDABSENT (XML_E_BASE-18)

  Some key has no full set of fields.

- #define XML_E_DUPLFLD (XML_E_BASE-19)

  Some key has more than one value for field.

- #define XML_E_NOTEMPTY (XML_E_BASE-20)

  An element was not empty when expected.

Macro Definition Documentation

#define XML_OK_EOB

Definition at line 51 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_OK_FRAG

Definition at line 54 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_E_BASE

XML specific errors start at this base number to distinguish them from common and other error types.

Definition at line 60 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_E_TAGMISMATCH

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 90 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_E_NOMATCH

Informational code.

Definition at line 115 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_E_ELEMMISRQ

This status code is returned by the decoder when the decoder knows exactly which element is absent.

Definition at line 121 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_E_ELEMSMISRQ

This status code is returned by the decoder when the number of elements decoded for a given content model group is less then the required number of elements as specified in the schema.

Definition at line 128 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_E_NSURINOTFOU

A namespace URI was not defined using an xmlns attribute for the given prefix.

Definition at line 166 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

#define XML_E_FLDABSENT

It is not valid for key constraint.

Definition at line 199 of file rtXmlErrCodes.h

The Documentation for this define was generated from the following file:

- rtXmlErrCodes.h

DOM API functions.

Detailed Description

Typedefs

- typedef void * OSRTDOMDocPtr
- typedef void * OSRTDOMNodePtr
- typedef void * OSRTDOMAttrPtr
- typedef void * OSRTDOMNsNodePtr
- typedef void * OSRTDOMContCtxtPtr
- typedef void * OSRTDOMAttrCtxtPtr
- typedef int OSRTDOMError

  0 - OK <0 - Error code (see rtxsrc/rtxErrCodes.h) >0 - Supplementary code

Functions

- EXTERNDOM void domParserInit ( )

  Inits parser.

- EXTERNDOM void domParserShutdown ( )

  Shutdowns parser.

- EXTERNDOM OSRTDOMError domParseFile (const char * fileSpec, OSRTDOMDocPtr * pXmlDoc)

  Parse the file into a DOM document.

- EXTERNDOM OSRTDOMError domGetRootElement (OSRTDOMDocPtr xmlDoc, OSRTDOMNodePtr * pN-ode)

  Returns root node for the document.

- EXTERNDOM OSRTDOMError domGetDoc (OSRTDOMNodePtr node, OSRTDOMDocPtr * pXmlDoc)

  Returns the document for the given element.

- EXTERNDOM void domFreeDoc (OSRTDOMDocPtr xmlDoc)

  Frees document.

- EXTERNDOM OSRTDOMError domGetNext (const OSRTDOMNodePtr curNode, OSRTDOMNodePtr * pNextNode)

  Returns next element node (down sibling).

- EXTERNDOM OSRTDOMError domGetChild (const OSRTDOMNodePtr curNode, OSRTDOMNodePtr * pChildNode)

Returns first (top) child node.

- EXTERNDOM OSRTDOMError domGetElementName (const OSRTDOMNodePtr curNode, const OSUTF8CHAR ** ppName, const OSUTF8CHAR ** ppNsPrefix, const OSUTF8CHAR ** ppNsUri)

  Returns node's name and prefix (if any).

- EXTERNDOM int domGetNodeContent (OSRTDOMContCtxtPtr * ppCtxt, const OSRTDOMNodePtr curNode, const OSUTF8CHAR ** ppValue, size_t * pValueLen, OSBOOL * pCdataProcessed)

  Gets the node's content.

- EXTERNDOM OSRTDOMError domGetNodeFirstAttribute (OSRTDOMAttrCtxtPtr * ppCtxt, const OSRT-DOMNodePtr curNode, OSRTDOMAttrPtr * pTopAttrNode)

  Returns the pointer to the first (top) attribute of the node.

- EXTERNDOM int domGetNodeAttributesNum (const OSRTDOMNodePtr curNode)

  Returns number of attributes in the node.

- EXTERNDOM OSRTDOMError domGetNextAttr (OSRTDOMAttrCtxtPtr * ppCtxt, const OSRTDOMAttrPtr curAttrNode, OSRTDOMAttrPtr * pAttrNode)

  Returns the next attribute.

- EXTERNDOM OSRTDOMError domGetAttrData (const OSRTDOMAttrPtr curAttrNode, const OSUTF8CHAR ** ppAttrName, const OSUTF8CHAR ** ppAttrValue, const OSUTF8CHAR ** ppAttrPrefix, const OSUTF8CHAR ** ppAttrUri)

  Returns the attribute's name and value.

- EXTERNDOM OSRTDOMError domSaveDoc (OSRTDOMDocPtr xmlDoc, const char * filename)

  Saves DOM document to a file.

- EXTERNDOM OSRTDOMError domCreateDocument (OSCTXT * pctxt, OSRTDOMDocPtr * pXmlDoc, const OSUTF8CHAR * pNodeName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Creates a new document and a root element.

- EXTERNDOM OSRTDOMError domCreateChild (OSCTXT * pctxt, OSRTDOMNodePtr parentNode, con-st OSUTF8CHAR * pNodeName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs, OSRTDOMNodePtr * pNewNode)

  Creates new child node.

- EXTERNDOM OSRTDOMError domAddAttribute (OSCTXT * pctxt, OSRTDOMNodePtr node, const OSUTF8CHAR * pAttrName, const OSUTF8CHAR * pAttrValue, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Creates new attribute and adds it to the node.

- EXTERNDOM OSRTDOMError domAddContent (OSRTDOMNodePtr node, const OSUTF8CHAR * pContent, size_t contentLen)

  Adds content (text) to the node.

- EXTERNDOM OSRTDOMError domAddCdata (OSRTDOMNodePtr node, const OSUTF8CHAR * pContent, size_t contentLen)

  Adds CDATA section to the node.

- EXTERNDOM OSBOOL domIsContent (OSRTDOMNodePtr node)

Typedef Documentation

typedef void* OSRTDOMDocPtr

typedef void* OSRTDOMNodePtr

typedef void* OSRTDOMAttrPtr

typedef void* OSRTDOMNsNodePtr

typedef void* OSRTDOMContCtxtPtr

typedef void* OSRTDOMAttrCtxtPtr

typedef int OSRTDOMError

0 - OK <0 - Error code (see rtxsrc/rtxErrCodes.h) >0 - Supplementary code

Function Documentation

EXTERNDOM void domParserInit ()

Inits parser.

EXTERNDOM void domParserShutdown ()

Shutdowns parser.

EXTERNDOM OSRTDOMError domParseFile (const char *fileSpec, OSRTDOMDocPtr *pXmlDoc)

Parse the file into a DOM document.

Table 3.226. Parameters

fileSpec	File name
pXmlDoc	Pointer to a pointer to newly created DOM document.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domGetRootElement (OSRTDOM-DocPtr xmlDoc, OSRTDOMNodePtr *pNode)

Returns root node for the document.

Table 3.227. Parameters

xmlDoc	Pointer to the DOM document context
pNode	Pointer to pointer to receive the root node.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domGetDoc (OSRTDOMNodePtr node, OSRTDOMDocPtr *pXmlDoc)

Returns the document for the given element.

Table 3.228. Parameters

node	Pointer to a node.
pXmlDoc	Pointer to a pointer to the DOM document context

Returns: . 0 - success, otherwise error code.

EXTERNDOM void domFreeDoc (OSRTDOMDocPtr xmlDoc)

Frees document.

Table 3.229. Parameters

xmlDoc	Pointer to the DOM document context

EXTERNDOM OSRTDOMError domGetNext (const OSRT-DOMNodePtr curNode, OSRTDOMNodePtr *pNextNode)

Returns next element node (down sibling).

Table 3.230. Parameters

curNode	Pointer to current node,
pNextNode	Pointer to pointer to receive the next node.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domGetChild (const OSRT-DOMNodePtr curNode, OSRTDOMNodePtr *pChildNode)

Returns first (top) child node.

Table 3.231. Parameters

curNode	Pointer to current node,
pChildNode	Pointer to pointer to receive the child node.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domGetElementName (const OSRT-DOMNodePtr curNode, const OSUTF8CHAR **ppName, const OSUTF8CHAR **ppNsPrefix, const OSUTF8CHAR **ppNsUri)

Returns node's name and prefix (if any).

Table 3.232. Parameters

curNode	Pointer to current node,
ppName	Pointer to pointer to receive the node's local name.
ppNsPrefix	Pointer to pointer to receive the node's prefix. May be NULL, if prefix is not important.
ppNsUri	Pointer to pointer to receive the namespace's Uri.

Returns: . 0 - success, otherwise error code.

EXTERNDOM int domGetNodeContent (OSRTDOMContCtxtPtr *ppC-txt, const OSRTDOMNodePtr curNode, const OSUTF8CHAR **pp-Value, size_t *pValueLen, OSBOOL *pCdataProcessed)

Gets the node's content.

Table 3.233. Parameters

ppCtxt	Pointer to an implementation-specific content context pointer;
curNode	Pointer to the current node;
ppValue	Pointer to pointer to receive value;
pValueLen	Pointer to value length; may be NULL.
pCdataProcessed	Pointer to boolean value; this value will be set to TRUE if CDATA section was proceeded; otherwise FALSE.

Returns: . 0 - if content retrieved successfully <0 - the error code;

EXTERNDOM OSRTDOMError domGetNodeFirstAttribute (OSRTDO-MAttrCtxtPtr *ppCtxt, const OSRTDOMNodePtr curNode, OSRTDO-MAttrPtr *pTopAttrNode)

Returns the pointer to the first (top) attribute of the node.

The context pointer will be passed to subsequent calls to domGetNextAttr.

Table 3.234. Parameters

ppCtxt	Pointer to an implementation-specific context pointer,

curNode	Pointer to current node,
pTopAttrNode	Pointer to pointer to receive the first attribute node. domGetNextAttr/domGetPrevAttr functions may be used to get next/previous attributes.

Returns: . 0 - success, otherwise error code.

EXTERNDOM int domGetNodeAttributesNum (const OSRTDOMNodePtr curNode)

Returns number of attributes in the node.

Table 3.235. Parameters

curNode	Pointer to current node,

Returns: . Number of attributes in the node.

EXTERNDOM OSRTDOMError domGetNextAttr (OSRTDOMAttrCtxtPtr *ppCtxt, const OSRTDOMAttrPtr curAttrNode, OSRTDOMAttrPtr *pAttrNode)

Returns the next attribute.

Table 3.236. Parameters

ppCtxt	Pointer to an implementation-specific context pointer,
curAttrNode	Pointer to current attribute node,
pAttrNode	Pointer to pointer to receive the next attribute node.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domGetAttrData (const OSRTDOMAttrPtr curAttrNode, const OSUTF8CHAR **ppAttrName, const OSUTF8CHAR **ppAttrValue, const OSUTF8CHAR **ppAttrPrefix, const OSUTF8CHAR **ppAttrUri)

Returns the attribute's name and value.

Table 3.237. Parameters

curAttrNode	Pointer to current attribute node,
ppAttrName	Pointer to pointer to receive attribute's name.
ppAttrValue	Pointer to pointer to receive attribute's value.
ppAttrPrefix	Pointer to pointer to receive namespace's prefix.
ppAttrUri	Pointer to pointer to receive namespace's uri.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domSaveDoc (OSRTDOMDocPtr xml-Doc, const char *filename)

Saves DOM document to a file.

This function is supplementary and it is not obligatory to implement it.

Table 3.238. Parameters

xmlDoc	Pointer to the DOM document context.
filename	Pointer to file name.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domCreateDocument (OSCTXT *pc-txt, OSRTDOMDocPtr *pXmlDoc, const OSUTF8CHAR *pNodeName, OSXMLNamespace *pNS, OSRTDList *pNSAttrs)

Creates a new document and a root element.

Table 3.239. Parameters

pctxt	Pointer to context block structure.
pXmlDoc	Pointer to pointer to the newly created empty DOM document.
pNodeName	Name of the node
pNS	XML namespace information (prefix and URI).
pNSAttrs	List of namespace attributes to be added to element.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domCreateChild (OSCTXT *pctxt, OSRTDOMNodePtr parentNode, const OSUTF8CHAR *pNodeName, OSXMLNamespace *pNS, OSRTDList *pNSAttrs, OSRTDOMNodePtr *pNewNode)

Creates new child node.

Table 3.240. Parameters

pctxt	Pointer to context block structure.
parentNode	Pointer to the parent node
pNodeName	Name of the node
pNS	XML namespace information (prefix and URI).

pNSAttrs	List of namespace attributes to be added to element.
pNewNode	Pointer to pointer to newly created node.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domAddAttribute (OSCTXT *pctxt, OSRTDOMNodePtr node, const OSUTF8CHAR *pAttrName, const OSUTF8CHAR *pAttrValue, OSXMLNamespace *pNS, OSRTDList *pNSAttrs)

Creates new attribute and adds it to the node.

Table 3.241. Parameters

pctxt	Pointer to context block structure.
node	Pointer to the node where attribute to be added.
pAttrName	Pointer to the null-terminated string containing name of the attribute.
pAttrValue	Pointer to the null-terminated string containing name of the value.
pNS	XML namespace information (prefix and URI).
pNSAttrs	List of namespace attributes to be added to element.

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domAddContent (OSRTDOMNodePtr node, const OSUTF8CHAR *pContent, size_t contentLen)

Adds content (text) to the node.

Table 3.242. Parameters

node	Pointer to the node where content to be added.
pContent	Pointer to UTF-8 encoded content.
contentLen	Length of the content

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSRTDOMError domAddCdata (OSRTDOMNodePtr node, const OSUTF8CHAR *pContent, size_t contentLen)

Adds CDATA section to the node.

Table 3.243. Parameters

node	Pointer to the node where content to be added.
pContent	Pointer to UTF-8 encoded content to be wrapped by CDATA section.

contentLen	Length of the content

Returns: . 0 - success, otherwise error code.

EXTERNDOM OSBOOL domIsContent (OSRTDOMNodePtr node)

DOM runtime encode/decode functions.

Detailed Description

Functions

- EXTERNDOM int rtDomAddAttr (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, const OSUTF8CHAR * attrName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Adds attribute to the node.

- EXTERNDOM int rtDomAddNode (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Adds child node to the parent's node.

- EXTERNDOM int rtDomAddNSAttrs (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr rootNode, OSRTDList * pNSAttrs)

  Adds namespace attributes to the root node.

- EXTERNDOM int rtDomEncXSIAttrs (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSBOOL needXSI)

  Adds XSI attributes to the node.

- EXTERNDOM int rtDomDecodeDoc (OSRTDOMDocPtr domDoc, struct OSSAXHandlerBase * pSaxBase)

  This function starts decoding of the DOM document.

- EXTERNDOM int rtDomEncStringValue (OSCTXT * pctxt, const OSUTF8CHAR * pvalue)

  This function encodes a variable of the XSD string type.

- EXTERNDOM int rtDomEncString (OSCTXT * pctxt, OSXMLSTRING * pvalue)

  This function encodes a variable of the XSD string type.

- EXTERNDOM int rtDomEncAny (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSXMLSTRING * pXmlData, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  This function encodes a variable of the XSD any type.

- EXTERNDOM int rtDomEncAnyAttr (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSRTDList * pAnyAttrList)

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

- EXTERNDOM int rtDomSetNode (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr curNode)

  Adds content or CDATA section to the node.

- EXTERNDOM int rtDomAddSubTree (OSCTXT * pctxt, OSRTDOMDocPtr xmlDoc, OSRTDOMNodePtr node, const OSUTF8CHAR * pXmlData, size_t dataLen)

  This function adds the subtree to the specified node.

Function Documentation

EXTERNDOM int rtDomAddAttr (OSCTXT *pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, const OSUTF8CHAR *attrName, OSXMLNamespace *pNS, OSRTDList *pNSAttrs)

Adds attribute to the node.

The value is taken from the context's buffer (pctxt->buffer.data with the length pctxt->buffer.byteIndex). This function resets pctxt->buffer.byteIndex to 0.

Table 3.244. 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.
domDoc	A pointer to a DOM-document.
node	A pointer to a node where attribute to be added.
attrName	A pointer to attribute's name.
pNS	XML namespace information (prefix and URI).
pNSAttrs	List of namespace attributes to be added to element.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomAddNode (OSCTXT *pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, OSRTDList *pNSAttrs)

Adds child node to the parent's node.

The content (text) is taken from the context's buffer (pctxt->buffer.data with the length pctxt->buffer.byteIndex). If pctxt->buffer.byteIndex is 0 then empty element is added. This function resets pctxt->buffer.byteIndex to 0.

Table 3.245. 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.

domDoc	A pointer to a DOM-document.
parentNode	A pointer to a parent node.
elemName	A pointer to element's name.
pNS	XML namespace information (prefix and URI).
pNSAttrs	List of namespace attributes to be added to element.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomAddNSAttrs (OSCTXT *pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr rootNode, OSRTDList *pNSAttrs)

Adds namespace attributes to the root node.

Table 3.246. 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.
domDoc	A pointer to a DOM-document.
rootNode	A pointer to a root node.
pNSAttrs	A pointer to a list of namespace attrs.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomEncXSIAttrs (OSCTXT *pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSBOOL needXSI)

Adds XSI attributes to the node.

Table 3.247. 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.
domDoc	A pointer to a DOM-document.
node	A pointer to a node.
needXSI	Determines whether XSI namespace declaration needed.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomDecodeDoc (OSRTDOMDocPtr domDoc, struct OSSAXHandlerBase *pSaxBase)

This function starts decoding of the DOM document.

Table 3.248. Parameters

domDoc	A pointer to a DOM-document.
pSaxBase	A pointer to SAX handler's base.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomEncStringValue (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

This function encodes a variable of the XSD string type.

Table 3.249. Parameters

pctxt	Pointer to context block structure.
pvalue	Null-terminated XML string value to be encoded.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomEncString (OSCTXT *pctxt, OSXMLSTRING *pvalue)

This function encodes a variable of the XSD string type.

Table 3.250. Parameters

pctxt	Pointer to context block structure.
pvalue	XML string value to be encoded.

Returns: . Completion status of operation:

- 0 = success,

- negative return value is error.

EXTERNDOM int rtDomEncAny (OSCTXT *pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSXMLSTRING *pXmlData, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS, OSRTDList *pNSAttrs)

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>)

Table 3.251. Parameters

pctxt	Pointer to context block structure.
domDoc	A pointer to a DOM-document.
parentNode	A pointer to a parent node.
pXmlData	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 NULL pointer is passed, no element tag is added to the encoded value.
pNS	XML namespace information (prefix and URI).
pNSAttrs	List of namespace attributes to be added to element.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomEncAnyAttr (OSCTXT *pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSRTDList *pAnyAttrList)

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

Table 3.252. Parameters

pctxt	Pointer to context block structure.
domDoc	A pointer to a DOM-document.
node	A pointer to a node where attributes to be added.
pAnyAttrList	List of attributes.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomSetNode (OSCTXT *pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr curNode)

Adds content or CDATA section to the node.

The content (text) is taken from the context's buffer (pctxt->buffer.data with the length pctxt->buffer.byteIndex). This function resets pctxt->buffer.byteIndex to 0.

Table 3.253. 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.
domDoc	A pointer to a DOM-document.
curNode	A pointer to a current node.

Returns: . Completion status of operation:

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

EXTERNDOM int rtDomAddSubTree (OSCTXT *pctxt, OSRTDOM-DocPtr xmlDoc, OSRTDOMNodePtr node, const OSUTF8CHAR *pXmlData, size_t dataLen)

This function adds the subtree to the specified node.

The pXmlData contains the fragment of XML data. This fragment is parsed to sub-tree and then inserted as children to the 'node'.

Table 3.254. Parameters

pctxt	Pointer to the context structure. This context might be used for memory allocations, if necessary.
xmlDoc	Pointer to the document
node	Pointer to the node where sub-tree to be inserted.
pXmlData	Pointer to UTF-8 string representing fragment of XML data. For example, "<elem>data</elem>".
dataLen	Length of XML data.

Returns: . Completion status of operation:

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

Chapter 4. Data Structure Documentation

AutoState struct Reference

Data Fields

- int state
- int intDigits
- int fracDigits
- int expDigits
- OSBOOL exponent

Field Documentation

BinInputStream class Reference

DefaultHandler class Reference

InputSource class Reference

OSCustomBinInputStream class Reference

Private Attributes

- OSRTInputStreamIF & source

- OSCustomBinInputStream (OSRTInputStreamIF & source_)
- virtual ~OSCustomBinInputStream ( )
- virtual XMLFilePos curPos ( )
- virtual XMLSize_t readBytes (XMLByte *const toFill, const XMLSize_t maxToRead)
- virtual const XMLCh * getContentType ( )

Field Documentation

OSCustomBinInputStream::OSCustomBinInputStream (OSRTInputStreamIF &source_)

virtual OSCustomBinInputStream::~OSCustomBinInputStream ()

virtual XMLFilePos OSCustomBinInputStream::curPos () const

virtual XMLSize_t OSCustomBinInputStream::readBytes (XMLByte *const toFill, const XMLSize_t maxToRead)

virtual const XMLCh* OSCustomBinInputStream::getContentType () const

OSCustomInputSource class Reference

Private Attributes

- OSRTInputStreamIF & source

- OSCustomInputSource (OSRTInputStreamIF & source_)
- ~OSCustomInputSource ( )
- BinInputStream * makeStream ( )

Field Documentation

OSCustomInputSource::OSCustomInputSource (OSRTInputStreamIF &source_)

OSCustomInputSource::~OSCustomInputSource ()

BinInputStream* OSCustomInputSource::makeStream () const

OSExpatXMLReader class Reference

Protected Attributes

- OSXMLParserCtxtIF * mpContext
- XML_Parser mParser
- OSXMLDefaultHandlerIF * saxHandler
- int mLevel
- int mState

- OSExpatXMLReader (OSXMLParserCtxtIF * pContext)

  The default constructor.
- virtual ~OSExpatXMLReader ( )

  The destructor.
- virtual int parse ( )
- virtual int parse (OSRTInputStreamIF & source)
- virtual int parse (const char *const pBuffer, size_t bufSize)
- virtual int parse (const char *const systemId)
- void setSaxHandler (OSXMLDefaultHandlerIF *const handler)

- static void startElementCallback (void * userData, const XML_Char * name, const XML_Char ** atts)
- static void endElementCallback (void * userData, const XML_Char * name)
- static void charDataCallback (void * userData, const XML_Char * s, int len)

- static void startCdataSectionHandler (void * userData)
- static void endCdataSectionHandler (void * userData)

Field Documentation

OSExpatXMLReader::OSExpatXMLReader (OSXML-ParserCtxtIF *pContext)

The default constructor.

OSExpatXMLReader::~OSExpatXMLReader ()

The destructor.

int OSExpatXMLReader::parse ()

int OSExpatXMLReader::parse (OSRTInputStreamIF &source)

int OSExpatXMLReader::parse (const char *const pBuffer, size_t bufSize)

int OSExpatXMLReader::parse (const char *const systemId)

void OSExpatXMLReader::setSaxHandler (OSXMLDefaultHandlerIF *const handler)

void OSExpatXMLReader::startElementCallback (void *userData, const XML_Char *name, const XML_Char **atts)

void OSExpatXMLReader::endElementCallback (void *userData, const XML_Char *name)

void OSExpatXMLReader::charDataCallback (void *userData, const XML_Char *s, int len)

void OSExpatXMLReader::startCdataSectionHandler (void *userData)

void OSExpatXMLReader::endCdataSectionHandler (void *userData)

OSIntegerFmt struct Reference

Data Fields

- OSINT8 integerMaxDigits
- OSBOOL signPresent

Field Documentation

OSLibxmlXMLReader class Reference

Protected Attributes

- OSXMLParserCtxtIF * mpContext
- xmlSAXHandler saxHandler
- xmlSAXLocatorPtr locator
- xmlParserCtxtPtr currentCtxt
- OSXMLDefaultHandlerIF * userSaxHandler
- OSBOOL bFatalError

- static OSBOOL bManualCleanup

- OSLibxmlXMLReader (OSXMLParserCtxtIF * pContext)

  The default constructor.
- virtual ~OSLibxmlXMLReader ( )

  The destructor.
- virtual int parse ( )
- virtual int parse (OSRTInputStreamIF & source)
- virtual int parse (const char *const pBuffer, size_t bufSize)
- virtual int parse (const char *const systemId)
- void setSaxHandler (OSXMLDefaultHandlerIF *const handler)

- static void setThreadSafety (OSBOOL value)

- static void startElementCallback (void * userData, const xmlChar * name, const xmlChar ** atts)
- static void endElementCallback (void * userData, const xmlChar * name)
- static void charDataCallback (void * userData, const xmlChar * s, int len)
- static void errorCallback (void * ctx, const char * msg, ...)

  error: @ctxt: An XML parser context @msg: the message to display/transmit @...: extra parameters for the message display
- static void setDocumentLocator (void * ctx, xmlSAXLocatorPtr loc)
- static void cdataBlockCallback (void * userData, const xmlChar * value, int len)

Field Documentation

OSLibxmlXMLReader::OSLibxmlXMLReader (OSXML-ParserCtxtIF *pContext)

The default constructor.

OSLibxmlXMLReader::~OSLibxmlXMLReader ()

The destructor.

int OSLibxmlXMLReader::parse ()

int OSLibxmlXMLReader::parse (OSRTInputStreamIF &source)

int OSLibxmlXMLReader::parse (const char *const pBuffer, size_t bufSize)

int OSLibxmlXMLReader::parse (const char *const sys-temId)

void OSLibxmlXMLReader::setSaxHandler (OSXMLDe-faultHandlerIF *const handler)

static void OSLibxmlXMLReader::setThreadSafety (OS-BOOL value=TRUE)

void OSLibxmlXMLReader::startElementCallback (void *userData, const xmlChar *name, const xmlChar **atts)

void OSLibxmlXMLReader::endElementCallback (void *userData, const xmlChar *name)

void OSLibxmlXMLReader::charDataCallback (void *userData, const xmlChar *s, int len)

void OSLibxmlXMLReader::errorCallback (void *ctx, con-st char *msg,...)

error: @ctxt: An XML parser context @msg: the message to display/transmit @...: extra parameters for the message display

Display and format a error messages, gives file, line, position and extra parameters.

void OSLibxmlXMLReader::setDocumentLocator (void *ctx, xmlSAXLocatorPtr loc)

void OSLibxmlXMLReader::cdataBlockCallback (void *userData, const xmlChar *value, int len)

OSMemBlock struct Reference

Data Fields

- OSOCTET buffer[MAX_MEMBLOCK_SIZE]
- OSSIZE lastIdx

Field Documentation

OSXercesString class Reference

Private Attributes

- OSUTF8CHAR * utf8
- unsigned length

- static XMLUTF8Transcoder * transcoder

- static unsigned getSizeInBytes (const XMLCh * str, unsigned * srcLen)

- OSXercesString (const XMLCh * str, unsigned len)
- ~OSXercesString ( )
- unsigned getLength ( )
- operator const OSUTF8CHAR * ( )
- operator const char * ( )

Field Documentation

unsigned OSXercesString::getSizeInBytes (const XMLCh *str, unsigned *srcLen)

OSXercesString::OSXercesString (const XMLCh *str, unsigned len=UINT_MAX)

OSXercesString::~OSXercesString ()

unsigned OSXercesString::getLength ()

OSXercesString::operator const OSUTF8CHAR * ()

OSXercesString::operator const char * ()

OSXercesStringArray class Reference

Private Attributes

- OSXercesString ** array
- XMLSize_t size
- const OSUTF8CHAR ** utf8array

- OSXercesStringArray (XMLSize_t sz)
- ~OSXercesStringArray ( )
- void set (XMLSize_t idx, OSXercesString * elem)
- const OSUTF8CHAR ** getUtf8Array ( )

Field Documentation

OSXercesStringArray::OSXercesStringArray (XMLSize_t sz)

OSXercesStringArray::~OSXercesStringArray ()

void OSXercesStringArray::set (XMLSize_t idx, OSXercesString *elem)

const OSUTF8CHAR** OSXercesStringArray::getUtf8Array ()

OSXercesXMLReader class Reference

Data Structures

- struct OSXercesXMLReader::StopParserException

Data Fields

- OSXMLParserCtxtIF * mpContext
- SAX2XMLReader * mpParser
- OSXMLDefaultHandlerIF * userSaxHandler

- OSXercesXMLReader (OSXMLParserCtxtIF * pContext)
- virtual ~OSXercesXMLReader ( )
- virtual int parse ( )
- virtual int parse (OSRTInputStreamIF & source)
- virtual int parse (const char *const pBuffer, size_t bufSize)
- virtual int parse (const char *const systemId)
- void setSaxHandler (OSXMLDefaultHandlerIF *const handler)

- virtual void startElement (const XMLCh *const uri, const XMLCh *const localname, const XMLCh *const qname, const Attributes & attrs)
- virtual void characters (const XMLCh *const chars, const unsigned int length)

- virtual void endElement (const XMLCh *const uri, const XMLCh *const localname, const XMLCh *const qname)
- virtual void startCDATA ( )
- virtual void endCDATA ( )

Field Documentation

OSXercesXMLReader::OSXercesXMLReader (OSXML-ParserCtxtIF *pContext)

OSXercesXMLReader::~OSXercesXMLReader ()

int OSXercesXMLReader::parse ()

int OSXercesXMLReader::parse (OSRTInputStreamIF &source)

int OSXercesXMLReader::parse (const char *const pBuffer, size_t bufSize)

int OSXercesXMLReader::parse (const char *const sys-temId)

void OSXercesXMLReader::setSaxHandler (OSXMLDe-faultHandlerIF *const handler)

void OSXercesXMLReader::startElement (const XMLCh *const uri, const XMLCh *const localname, const XMLCh *const qname, const Attributes &attrs)

void OSXercesXMLReader::characters (const XMLCh *const chars, const unsigned int length)

void OSXercesXMLReader::endElement (const XMLCh *const uri, const XMLCh *const localname, const XMLCh *const qname)

void OSXercesXMLReader::startCDATA ()

void OSXercesXMLReader::endCDATA ()

OSXMLAttribute struct Reference

Data Fields

- OSXMLStrFragment mQName

- OSXMLStrFragment mLocalName
- OSXMLStrFragment mPrefix
- OSXMLStrFragment mValue

Field Documentation

OSXMLAttrOffset struct Reference

Data Fields

- OSXMLStrFragOffset mQName
- OSXMLStrFragOffset mLocalName
- OSXMLStrFragOffset mPrefix
- OSXMLStrFragOffset mValue
- OSINT16 index

Field Documentation

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
- struct OSXMLReader * pXmlPPReader
- OSRTBuffer savedBuffer
- OSRTFLAGS savedFlags
- OSOCTET * attrsBuff
- OSSIZE attrsBuffSize
- OSSIZE attrStartPos

Field Documentation

OSXMLDataCtxt struct Reference

Data Fields

- OSINT32 mDataLevel
- OSXMLDataMode mDataMode
- int mnChunk
- OSBOOL mbLastChunk
- OSBOOL mbCDATA
- OSXMLStrFragment mData
- OSXMLStrFragment mSrcData
- size_t mSrcDataOffset

- OSBOOL mbInsTokenSeparator

Field Documentation

OSXMLElementName struct Reference

Data Fields

- OSXMLStrFragment mQName
- OSXMLStrFragment mLocalName
- OSXMLStrFragment mPrefix
- OSUTF8CHAR mBuffer[OSXML_DEFAULT_QNAME_BUF_SIZE]

Field Documentation

OSXMLElemIDRec struct Reference

Data Fields

- OSXMLElemDescr descr
- OSUINT16 id

Field Documentation

OSXMLElemNameOffset struct Reference

Data Fields

- OSXMLStrFragOffset mQName
- OSXMLStrFragOffset mLocalName
- OSXMLStrFragOffset mPrefix

Field Documentation

OSXMLEvent struct Reference

Data Fields

- OSUINT32 mId
- OSINT32 mLevel

Field Documentation

OSXMLFacets struct Reference

Data Fields

- int totalDigits
- int fractionDigits

Field Documentation

OSXMLGroupDesc struct Reference

```
#include <osrtxml.h>
```

Data Fields

- int row
- int num
- int anyCase

OSXMLGroupDesc describes how entries in an OSXMLElemIDRec array make up a group.

Detailed Description

Here, "group" means a set of elements, any of which may be matched next. This does not correspond directly to an XSD group.

For example, if elementA is optional and followed by non-optional elementB, then there will be a group that contains both elements. There will also be a group that contains only elementB; this will be the group of interest after elementA is matched.

Definition at line 140 of file osrtxml.h

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

- osrtxml.h

Field Documentation

OSXMLItemDescr struct Reference

Data Fields

- OSXMLStrFragment localName

- OSINT16 nsidx

Field Documentation

OSXMLNameFragments struct Reference

Data Fields

- OSXMLStrFragment mQName
- OSXMLStrFragment mLocalName
- OSXMLStrFragment mPrefix

Field Documentation

OSXMLNamespace struct Reference

Data Fields

- const OSUTF8CHAR * prefix
- const OSUTF8CHAR * uri

Field Documentation

OSXMLNamespace_ struct Reference

Data Fields

- OSXMLStrFragment mPrefix
- OSUTF8CHAR prefixBuff[8]
- OSINT16 index
- int mLevel
- const char * urn

Field Documentation

OSXMLNamespacesStack struct Reference

Data Fields

- OSXMLNamespace_ * mpStackArray

- size_t mSize
- size_t mCount

Field Documentation

OSXMLNSPfxLink struct Reference

Data Fields

- OSINT32 nsidx
- OSXMLNamespace ns
- OSRTDynPtrArray extraPrefixes
- struct OSXMLNSPfxLink * next

Field Documentation

OSXMLNSPfxLinkStack struct Reference

Data Fields

- OSINT32 count
- OSXMLNSPfxLinkStackNode * top

Field Documentation

OSXMLNSPfxLinkStackNode struct Reference

Data Fields

- struct OSXMLNSPfxLink * link
- struct OSXMLNSPfxLinkStackNode * next

Field Documentation

OSXMLNSURITable struct Reference

Data Fields

- OSUINT32 nrows
- const OSUTF8CHAR ** data

Field Documentation

OSXMLQName struct Reference

Data Fields

- const OSUTF8CHAR * nsPrefix
- const OSUTF8CHAR * ncName

Field Documentation

OSXMLReader struct Reference

Data Fields

- OSCTXT * mpCtxt
- OSBOOL mbNoTransform
- OSBOOL mbMixedContext
- OSBOOL mbListMode
- OSBOOL mbSkipPullListElem
- OSBOOL mbDecodeAsGroup
- OSXMLWhiteSpaceMode mWhiteSpaceMode
- OSXMLWhiteSpaceMode mStringWhiteSpaceMode
- OSXMLEncoding mEncoding
- OSXMLBOM mBOM
- OSXMLEvent mLastEvent
- OSXMLElemNameOffset mElementName
- OSBOOL mbHasAttributes
- OSBOOL mbEmptyElement
- OSXMLStack mAttributes
- OSXMLStrFragment mData
- OSBOOL mbLastChunk
- OSBOOL mbCDATA
- OSINT32 mLevel

- OSINT32 mDecodeLevel
- int mError
- OSINT16 mElementNsIndex
- int mState
- int mPrevState
- int mNewState
- int mLocalStates[OSXMLSI_LAST]
- OSUTF8CHAR mCharBuf[10]
- OSUTF8CHAR * mpEscapeChar
- OSUTF8CHAR * mpChars
- size_t mMarkedByteIndex
- OSXMLDataMode mDataMode
- OSXMLStack mTagNamesStack
- OSXMLStack mNamespacesStack
- const char ** mNamespaceTable
- int mNamespacesNumber
- const char ** mPrevNamespaceTable
- int mPrevNamespacesNumber
- OSOCTET mDelayedTasks[MAX_DELAYED_TASK_SIZE]
- size_t mDelayedTaskCount
- size_t mDelayedTaskIndex
- OSUTF8CHAR * mpBuffer
- size_t mByteIndex
- size_t mBufSize
- size_t mReadSize
- size_t mMarkedPos
- size_t mLastBlockSize
- size_t mLastByteIndex
- OSBOOL mbBackoffEnabled
- OSBOOL mbSysMemBuf

- OSXMLStack mRewindPosStack
- OSXMLSrcPos mSrcPos
- size_t mMarkedAttrCount

Field Documentation

OSXMLReaderClass class Reference

OSXMLRewindPos struct Reference

Data Fields

- size_t mPos
- OSXMLEvent mEvent
- OSXMLStrFragment mData
- OSBOOL mbListMode
- OSBOOL mbLastChunk

Field Documentation

OSXMLSortedAttrOffset struct Reference

Data Fields

- OSSIZE offset
- OSSIZE length
- OSSIZE prefixLength
- OSSIZE nameLength

Field Documentation

OSXMLSrcPos struct Reference

Data Fields

- OSUINT32 mLine
- OSUINT32 mColumn
- OSUINT32 mByteIdx

- OSBOOL mbCR

Field Documentation

OSXMLStack struct Reference

Data Fields

- void * mpStackArray
- size_t mSize
- size_t mUnitSize
- size_t mCount
- OSBOOL mbDynamic

Field Documentation

OSXMLStrFragment struct Reference

Data Fields

- const OSUTF8CHAR * value
- OSSIZE length

Field Documentation

OSXMLStrFragOffset struct Reference

Data Fields

- size_t offset
- size_t length

Field Documentation

OSXMLStrFragStack struct Reference

Data Fields

- OSXMLStrFragment * mpStackArray
- size_t mSize

- size_t mCount

Field Documentation

OSXMLStringListParser class Reference

```
#include <rtXmlpCppDecFuncs.h>
```

Private Attributes

- OSXMLDataCtxt dataCtxt
- OSCTXT * mpctxt
- OSBOOL mbInit
- const OSUTF8CHAR * inpdata

- OSXMLStringListParser (OSCTXT * pctxt)

  Create a parser on the given context.
- int next (OSRTXMLString & value)

  Assign the next string from the XML schema list to value.

Class enabling parsing of an XML Schema list into strings.

Field Documentation

OSXMLStringListParser::OSXMLStringListParser (OSCTXT *pctxt)

Create a parser on the given context.

The OSXMLReader associated with the context is used to read the XML value.

Table 4.1. Parameters

pctxt	Holds state information and provides access to the input that is to be parsed.

int OSXMLStringListParser::next (OSRTXMLString &value)

Assign the next string from the XML schema list to value.

To get all of the values from the list, call this function repeatedly until it returns zero or a negative value. After that, this object can no longer be used.

Table 4.2. Parameters

value	Receives the next string value or empty string if there isn't one.

Returns: . Return value indicates the result as follows: return < 0: error return == 0: no more strings; value is assigned empty return == 1: value is assigned the next string

OSXSDAnyType struct Reference

Data Fields

- OSXMLSTRING value
- OSRTDList attrs

Field Documentation

OSXSDKeyArray struct Reference

Data Fields

- OSUINT32 capacity
- OSUINT32 count
- OSUINT32 nmFields
- OSUINT32 curField
- OSXSDKeyRecord * data
- const char * name
- OSBOOL keyConstraint
- OSINT8 duplicateField
- OSINT8 absentField

Field Documentation

OSXSDKeyRecord struct Reference

Data Structures

Data Fields

- OSINT8 tag

- OSINT8 subTag
- OSUINT16 dummy
- OSINT32 int32
- OSUINT32 uint32
- const OSOCTET * pValue
- union OSXSDKeyRecord::@3 u

Field Documentation

rtdom_SaxHandler struct Reference

Data Fields

- OSSAXHandlerBase mSaxBase
- OSSAXHandlerBase * mpCurrElem
- OSRTDOMNodePtr mpParentNode
- OSRTDList mNodePath
- OSRTDOMDocPtr xmlDoc

Field Documentation

StateEnc struct Reference

Data Fields

- const char * ePos
- unsigned sPos
- const char * eBeg[MAX_NESTING]
- unsigned sBeg[MAX_NESTING]
- int cnt[MAX_NESTING]
- char match[MAX_NESTING]
- int phase[MAX_NESTING]
- int nesting
- char buf[STR_SZ]

Field Documentation

OSXercesXMLReader::StopParserException class Reference

XMLReaderImpl struct Reference

Data Fields

- OSXMLREADER reader
- XML_Parser parser
- void * userData
- CSAX_StartElementHandler pStartElementProc
- CSAX_EndElementHandler pEndElementProc
- CSAX_CharacterDataHandler pCharactersProc
- xmlSAXHandler saxHandler
- xmlParserCtxtPtr xmlCtxt
- xmlSAXLocatorPtr locator
- char errorString[128]
- OSMemBlock memBlock
- OSBOOL bStop

Field Documentation

Chapter 5. File Documentation

domAPI.c File Reference

```
#include <stdlib.h>
#include "rtdomsrc/domAPI.h"
#include "rtxsrc/rtxErrCodes.h"
#include "rtxmlsrc/osrtxml.h"
#include "rtdomsrc/osrtdom.h"
#include <libxml/tree.h>
#include <libxml/parser.h>
#include <libxml/parserInternals.h>
#include <libxml/xpath.h>
#include <libxml/debugXML.h>
```

Macros

- #define _DOM_TESTING
- #define _xmlNewDoc xmlNewDoc(a)
- #define _xmlFreeDoc xmlFreeDoc(a)
- #define _xmlNewNode xmlNewNode(a,b)
- #define _xmlNewNs xmlNewNs(a,b,c)
- #define _xmlSetNs xmlSetNs(a,b)
- #define _xmlDocSetRootElement xmlDocSetRootElement(a,b)
- #define _xmlSearchNs xmlSearchNs(a,b,c)
- #define _xmlNewChild xmlNewChild(a,b,c,d)
- #define _xmlAddChild xmlAddChild(a,b)
- #define _xmlNewNsProp xmlNewNsProp(a,b,c,d)
- #define _xmlNewTextLen xmlNewTextLen(a,b)
- #define _xmlNewCDataBlock xmlNewCDataBlock(a,b,c)
- #define _xmlSaveFormatFile xmlSaveFormatFile(a,b,c)
- #define _xmlCleanupParser xmlCleanupParser()

Functions

- void domParserInit ( )

  Inits parser.

- void domParserShutdown ( )

  Shutdowns parser.

- OSRTDOMError domParseFile (const char * fileSpec, OSRTDOMDocPtr * pXmlDoc)

  Parse the file into a DOM document.

- OSRTDOMError domGetDoc (OSRTDOMNodePtr node, OSRTDOMDocPtr * pXmlDoc)

  Returns the document for the given element.

- OSRTDOMError domGetRootElement (OSRTDOMDocPtr xmlDoc, OSRTDOMNodePtr * pNode)

  Returns root node for the document.

- void domFreeDoc (OSRTDOMDocPtr xmlDoc)

  Frees document.

- OSRTDOMError domGetNext (const OSRTDOMNodePtr curNode, OSRTDOMNodePtr * pNextNode)

  Returns next element node (down sibling).

- OSRTDOMError domGetChild (const OSRTDOMNodePtr curNode, OSRTDOMNodePtr * pChildNode)

  Returns first (top) child node.

- OSRTDOMError domGetElementName (const OSRTDOMNodePtr curNode, const OSUTF8CHAR ** ppName, const OSUTF8CHAR ** ppNsPrefix, const OSUTF8CHAR ** ppNsUri)

  Returns node's name and prefix (if any).

- int domGetNodeContent (OSRTDOMContCtxtPtr * ppCtxt, const OSRTDOMNodePtr curNode, const OSUTF8CHAR ** ppValue, size_t * pValueLen, OSBOOL * pCdataProcessed)

  Gets the node's content.

- OSRTDOMError domGetNodeFirstAttribute (OSRTDOMAttrCtxtPtr * ppCtxt, const OSRTDOMNodePtr curN-ode, OSRTDOMAttrPtr * pTopAttrNode)

  Returns the pointer to the first (top) attribute of the node.

- int domGetNodeAttributesNum (const OSRTDOMNodePtr curNode)

  Returns number of attributes in the node.

- OSRTDOMError domGetNextAttr (OSRTDOMAttrCtxtPtr * ppCtxt, const OSRTDOMAttrPtr curAttrNode, OSRTDOMAttrPtr * pAttrNode)

  Returns the next attribute.

- OSRTDOMError domGetAttrData (const OSRTDOMAttrPtr curAttrNode, const OSUTF8CHAR ** ppAttrName, const OSUTF8CHAR ** ppAttrValue, const OSUTF8CHAR ** ppAttrPrefix, const OSUTF8CHAR ** ppAttrUri)

  Returns the attribute's name and value.

- OSRTDOMError domCreateDocument (OSCTXT * pctxt, OSRTDOMDocPtr * pXmlDoc, const OSUTF8CHAR * pNodeName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Creates a new document and a root element.

- OSRTDOMError domCreateChild (OSCTXT * pctxt, OSRTDOMNodePtr parentNode, const OSUTF8CHAR * pNodeName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs, OSRTDOMNodePtr * pNewNode)

  Creates new child node.

- OSRTDOMError domAddAttribute (OSCTXT * pctxt, OSRTDOMNodePtr node, const OSUTF8CHAR * pAttrName, const OSUTF8CHAR * pAttrValue, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Creates new attribute and adds it to the node.

- OSRTDOMError domAddContent (OSRTDOMNodePtr node, const OSUTF8CHAR * pContent, size_t contentLen)

  Adds content (text) to the node.

- OSRTDOMError domAddCdata (OSRTDOMNodePtr node, const OSUTF8CHAR * pContent, size_t contentLen)

  Adds CDATA section to the node.

- OSBOOL domIsContent (OSRTDOMNodePtr node)

- OSRTDOMError domSaveDoc (OSRTDOMDocPtr xmlDoc, const char * filename)

  Saves DOM document to a file.

Detailed Description

Definition in file domAPI.c

domAPI.h File Reference

```
#include "rtxsrc/rtxCommon.h"
```

```
#include "rtdomsrc/rtxDomDefs.h"
```

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

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

Typedefs

- typedef void * OSRTDOMDocPtr

- typedef void * OSRTDOMNodePtr

- typedef void * OSRTDOMAttrPtr
- typedef void * OSRTDOMNsNodePtr
- typedef void * OSRTDOMContCtxtPtr
- typedef void * OSRTDOMAttrCtxtPtr
- typedef int OSRTDOMError

  0 - OK <0 - Error code (see rtxsrc/rtxErrCodes.h) >0 - Supplementary code

Functions

- EXTERNDOM void domParserInit ( )

  Inits parser.

- EXTERNDOM void domParserShutdown ( )

  Shutdowns parser.

- EXTERNDOM OSRTDOMError domParseFile (const char * fileSpec, OSRTDOMDocPtr * pXmlDoc)

  Parse the file into a DOM document.

- EXTERNDOM OSRTDOMError domGetRootElement (OSRTDOMDocPtr xmlDoc, OSRTDOMNodePtr * pNode)

  Returns root node for the document.

- EXTERNDOM OSRTDOMError domGetDoc (OSRTDOMNodePtr node, OSRTDOMDocPtr * pXmlDoc)

  Returns the document for the given element.

- EXTERNDOM void domFreeDoc (OSRTDOMDocPtr xmlDoc)

  Frees document.

- EXTERNDOM OSRTDOMError domGetNext (const OSRTDOMNodePtr curNode, OSRTDOMNodePtr * pNextNode)

  Returns next element node (down sibling).

- EXTERNDOM OSRTDOMError domGetChild (const OSRTDOMNodePtr curNode, OSRTDOMNodePtr * pChildNode)

  Returns first (top) child node.

- EXTERNDOM OSRTDOMError domGetElementName (const OSRTDOMNodePtr curNode, const OSUTF8CHAR ** ppName, const OSUTF8CHAR ** ppNsPrefix, const OSUTF8CHAR ** ppNsUri)

  Returns node's name and prefix (if any).

- EXTERNDOM int domGetNodeContent (OSRTDOMContCtxtPtr * ppCtxt, const OSRTDOMNodePtr curNode, const OSUTF8CHAR ** ppValue, size_t * pValueLen, OSBOOL * pCdataProcessed)

  Gets the node's content.

- EXTERNDOM OSRTDOMError domGetNodeFirstAttribute (OSRTDOMAttrCtxtPtr * ppCtxt, const OSRT-DOMNodePtr curNode, OSRTDOMAttrPtr * pTopAttrNode)

  Returns the pointer to the first (top) attribute of the node.

- EXTERNDOM int domGetNodeAttributesNum (const OSRTDOMNodePtr curNode)

  Returns number of attributes in the node.

- EXTERNDOM OSRTDOMError domGetNextAttr (OSRTDOMAttrCtxtPtr * ppCtxt, const OSRTDOMAttrPtr curAttrNode, OSRTDOMAttrPtr * pAttrNode)

  Returns the next attribute.

- EXTERNDOM OSRTDOMError domGetAttrData (const OSRTDOMAttrPtr curAttrNode, const OSUTF8CHAR ** ppAttrName, const OSUTF8CHAR ** ppAttrValue, const OSUTF8CHAR ** ppAttrPrefix, const OSUTF8CHAR ** ppAttrUri)

  Returns the attribute's name and value.

- EXTERNDOM OSRTDOMError domSaveDoc (OSRTDOMDocPtr xmlDoc, const char * filename)

  Saves DOM document to a file.

- EXTERNDOM OSRTDOMError domCreateDocument (OSCTXT * pctxt, OSRTDOMDocPtr * pXmlDoc, const OSUTF8CHAR * pNodeName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Creates a new document and a root element.

- EXTERNDOM OSRTDOMError domCreateChild (OSCTXT * pctxt, OSRTDOMNodePtr parentNode, con-st OSUTF8CHAR * pNodeName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs, OSRTDOMNodePtr * pNewNode)

  Creates new child node.

- EXTERNDOM OSRTDOMError domAddAttribute (OSCTXT * pctxt, OSRTDOMNodePtr node, const OSUTF8CHAR * pAttrName, const OSUTF8CHAR * pAttrValue, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Creates new attribute and adds it to the node.

- EXTERNDOM OSRTDOMError domAddContent (OSRTDOMNodePtr node, const OSUTF8CHAR * pContent, size_t contentLen)

  Adds content (text) to the node.

- EXTERNDOM OSRTDOMError domAddCdata (OSRTDOMNodePtr node, const OSUTF8CHAR * pContent, size_t contentLen)

  Adds CDATA section to the node.

- EXTERNDOM OSBOOL domIsContent (OSRTDOMNodePtr node)

Detailed Description

Definition in file domAPI.h

osrtdom.h File Reference

```
#include "rtxsrc/rtxCommon.h"

#include "rtdomsrc/rtxDomDefs.h"

#include "rtxmlsrc/osrtxml.h"

#include "rtdomsrc/domAPI.h"

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

Functions

- EXTERNDOM int rtDomAddAttr (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, const OSUTF8CHAR * attrName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Adds attribute to the node.

- EXTERNDOM int rtDomAddNode (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Adds child node to the parent's node.

- EXTERNDOM int rtDomAddNSAttrs (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr rootNode, OSRTDList * pNSAttrs)

  Adds namespace attributes to the root node.

- EXTERNDOM int rtDomEncXSIAttrs (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSBOOL needXSI)

  Adds XSI attributes to the node.

- EXTERNDOM int rtDomDecodeDoc (OSRTDOMDocPtr domDoc, struct OSSAXHandlerBase * pSaxBase)

  This function starts decoding of the DOM document.

- EXTERNDOM int rtDomEncStringValue (OSCTXT * pctxt, const OSUTF8CHAR * pvalue)

  This function encodes a variable of the XSD string type.

- EXTERNDOM int rtDomEncString (OSCTXT * pctxt, OSXMLSTRING * pvalue)

  This function encodes a variable of the XSD string type.

- EXTERNDOM int rtDomEncAny (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSXMLSTRING * pXmlData, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  This function encodes a variable of the XSD any type.

- EXTERNDOM int rtDomEncAnyAttr (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSRTDList * pAnyAttrList)

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

- EXTERNDOM int rtDomSetNode (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr curNode)

  Adds content or CDATA section to the node.

- EXTERNDOM int rtDomAddSubTree (OSCTXT * pctxt, OSRTDOMDocPtr xmlDoc, OSRTDOMNodePtr node, const OSUTF8CHAR * pXmlData, size_t dataLen)

  This function adds the subtree to the specified node.

Detailed Description

DOM low-level C encode/decode functions.

Definition in file osrtdom.h

osrtxml.h File Reference

```
#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

  OSXMLGroupDesc describes how entries in an OSXMLElemIDRec array make up a group.

- struct OSXSDAnyType
- struct OSXMLCtxtInfo
- struct OSXMLQName
- struct OSIntegerFmt
- struct OSXMLSortedAttrOffset

Macros

- #define OSXMLNS12
- #define OSUPCASE 0x00008000 /* convert characters to upper case */
- #define OSTERMSTART 0x00004000 /* term for start elem (>) needed */
- #define OSEMPTYELEM 0x00002000 /* element is empty (no content) */
- #define OSQUALATTR 0x00001000 /* qualified attribute */
- #define OSXMLFRAG 0x00000800 /* XML fragment (not full doc) */
- #define OSXMLNSSET 0x00000400 /* Indicates namespaces are set */
- #define OSXMLC14N 0x00000200 /* Flag used to indicate XML canonical encode when OSASN1XER is not set, or canonical XER when OSASN1XER is set. */
- #define OSXSIATTR 0x00000100 /* add xsi ns decl to encoded msg */
- #define OSXMLNOCMPNS 0x00000080 /* match local names only */
- #define OSXSINIL 0x00000040 /* add xsi:nil decl to encoded msg */
- #define OSHASDEFAULT 0x00000010 /* decode should accept values which are empty after whitespace processing because the element has a default value */
- #define OSASN1XER 0x00000008 /* if true, -xml code and runtime should produce XER encodings instead of Obj-Sys encodings */
- #define OSXMLFRAGSEQUAL (frag1.length==frag2.length && ! memcmp(frag1.value,frag2.value,frag1.length))
- #define OSXMLQNAMEEQUALS rtxUTF8StrnEqual \ (xnamefrag.mQName.value, OSUTF8(qnametext), xnamefrag.mQName.length)
- #define OSXMLSETUTF8DECPTR rtxInitContextBuffer (pctxt, OSRTSAFECONSTCAST (OSOCTET*, str), \ OSUTF8LEN (str))
- #define IS_XMLNSATTR ((OSUTF8LEN(name) >= 5) && name[0] == 'x' && name[1] == 'm' && \ name[2] == 'l' && name[3] == 'n' && name[4] == 's')
- #define IS_XSIATTR ((OSUTF8LEN(name) >= 4) && name[0] == 'x' && name[1] == 's' && \ name[2] == 'i' && name[3] == ':')
- #define OSXMLINDENT 3
- #define rtXmlErrAddStrParm rtxErrAddStrParm
- #define rtxPrintNSAttrs rtXmlPrintNSAttrs(name,&data)
- #define rtXmlFinalizeMemBuf 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)
- #define rtXmlGetEncBufPtr (pctxt)->buffer.data

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

- #define rtXmlGetEncBufLen (pctxt)->buffer.byteIndex

  This macro returns the length of the encoded XML message.

- #define OSXMLREALENC_OBJSYS 0x1F /* Obj-Sys XML encoding rules */
- #define OSXMLREALENC_BXER 0x10 /* basic-XER */
- #define OSXMLREALENC_EXERMODS 0x1B /* extended-XER with MODIFIED-ENCODINGS */
- #define OSXMLREALENC_EXERDECIMAL 0x03 /* extended-XER with DECIMAL */

Enumerations

- enum OSXMLEncoding {
  OSXMLUTF8,
  OSXMLUTF16,
  OSXMLUTF16BE,
  OSXMLUTF16LE,
  OSXMLLATIN1
  }
- 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,
  OSXMLBOM_CHECK
  }
- enum OSXMLNsIndex {
  OSXMLNSI_UNQUALIFIED= 0,
  OSXMLNSI_UNKNOWN= -1,
  OSXMLNSI_UNCHECKED= -2,
  OSXMLNSI_XSI= -3,
  OSXMLNSI_XMLNS= -4,
  OSXMLNSI_XML= -5,
  OSXMLNSI_SOAP_ENVELOPE= -6,
  OSXMLNSI_XSD= -7
  }
- enum OSXMLREALEncoding {
  OSXMLREALOBJSYS,

OSXMLREALBXER,
OSXMLREALEXERMODS,
OSXMLREALEXERDEC
}

- enum OSXMLState {
  OSXMLINIT,
  OSXMLHEADER,
  OSXMLSTART,
  OSXMLATTR,
  OSXMLDATA,
  OSXMLEND,
  OSXMLCOMMENT
  }

- enum OSXMLWhiteSpaceMode {
  OSXMLWSM_PRESERVE= 0,
  OSXMLWSM_REPLACE,
  OSXMLWSM_COLLAPSE
  }

  Whitespace treatment options.

Typedefs

- typedef struct OSXMLFacets OSXMLFacets
- typedef struct OSXMLItemDescr OSXMLItemDescr
- typedef OSXMLItemDescr OSXMLAttrDescr
- typedef OSXMLItemDescr OSXMLElemDescr
- typedef struct OSXMLElemIDRec OSXMLElemIDRec
- typedef struct OSXMLGroupDesc OSXMLGroupDesc

  OSXMLGroupDesc describes how entries in an OSXMLElemIDRec array make up a group.
- typedef struct OSXSDAnyType OSXSDAnyType
- typedef struct OSXMLQName OSXMLQName
- typedef struct OSIntegerFmt OSIntegerFmt

Variables

- static const char OSXMLHDRUTF8
- static const char OSXMLHDRUTF16
- static const char OSXMLHDRUTF16BE
- static const char OSXMLHDRUTF16LE
- static const char OSXMLHDRLATIN1

Functions

- EXTERNXML int rtXmlInitContext (OSCTXT * pctxt)

  This function initializes a context variable for XML encoding or decoding.

- EXTERNXML int rtXmlInitContextUsingKey (OSCTXT * pctxt, const OSOCTET * key, OSSIZE keylen)

  This function initializes a context using a run-time key.

- 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, OSSIZE srcDataSize, OSSIZE * pNumOcts, OSSIZE * pSrcDataLen)

- EXTERNXML void rtXmlMemFreeAnyAttrs (OSCTXT * pctxt, OSRTDList * pAnyAttrList)

  This function frees a list of anyAttribute that is a member of OSXSDAnyType structure.

- EXTERNXML int rtXmlDecBase64Binary (OSRTMEMBUF * pMemBuf, const OSUTF8CHAR * inpdata, OSSIZE 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 rtXmlDecBase64Str64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlDecBase64Str except that is supports a 64-bit integer length on 64-bit systems.

- EXTERNXML int rtXmlDecBase64StrValue (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnocts, OSSIZE bufSize, OSSIZE srcDataLen)

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

- EXTERNXML int rtXmlDecBase64StrValue64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufSize, OSSIZE srcDataLen)

  This function decodes is identical to rtXmlDecBase64StrValue except that it supports a 64-bit integer length on 64-bit systems.

- 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 rtXmlDecDynBase64Str64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlDecDynBase64Str except that it supports a 64-bit integer length on 64-bit systems.

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

  This function decodes a contents of a hexBinary string.

- EXTERNXML int rtXmlDecDynHexStr64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlDecDynHexStr except that it supports a 64-bit integer length on 64-bit systems.

- EXTERNXML int rtXmlDecEmptyElement (OSCTXT * pctxt)

  This function is used to enforce a requirement that an element be empty.

- EXTERNXML int rtXmlDecUTF8Str (OSCTXT * pctxt, OSUTF8CHAR * outdata, OSSIZE 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, OSSIZE 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 * pnocts, OSINT32 bufsize)

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

- EXTERNXML int rtXmlDecHexStr64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlDecHexStr except that it supports a 64-bit integer length on 64-bit systems.

- EXTERNXML int rtXmlDecHexStrValue (OSCTXT * pctxt, const OSUTF8CHAR *const inpdata, OSSIZE nbytes, OSOCTET * pvalue, OSUINT32 * pnbits, OSINT32 bufsize)

- EXTERNXML int rtXmlDecHexStrValue64 (OSCTXT * pctxt, const OSUTF8CHAR *const inpdata, OSSIZE nbytes, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE 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 namespac 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, OSSIZE 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, OSSIZE attrNameLen)

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

- EXTERNXML int rtXmlEncBase64StrValue (OSCTXT * pctxt, OSSIZE nocts, const OSOCTET * value)

  This function encodes a variable of the XSD base64Binary type.

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

  This function encodes a variable of the XSD integer type.

- EXTERNXML int rtXmlEncBigIntAttr (OSCTXT * pctxt, const OSUTF8CHAR * value, const OSUTF8CHAR * attrName, OSSIZE attrNameLen)

  This function encodes an XSD integer attribute value.

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

  This function encodes an XSD integer attribute value.

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

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

- EXTERNXML int rtXmlEncBitStringExt (OSCTXT * pctxt, OSSIZE nbits, const OSOCTET * value, OSSIZE dataSize, const OSOCTET * extValue, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

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

- EXTERNXML int rtXmlEncBinStrValue (OSCTXT * pctxt, OSSIZE 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, OSSIZE attrNameLen)

  This function encodes an XSD boolean attribute value.

- EXTERNXML int rtXmlEncCanonicalSort (OSCTXT * pctxt, OSCTXT * pBufCtxt, OSRTSList * pList)

  Sort (as for CXER, Canonical-XER) and encode previously encoded SET OF components.

- EXTERNXML int rtXmlEncComment (OSCTXT * pctxt, const OSUTF8CHAR * comment)

  This function encodes an XML comment.

- 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 an string.

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

  This function encodes a variable of the XSD 'time' type as an 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, OSSIZE 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, OSSIZE 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, OSSIZE attrNameLen, const OSDoubleFmt * pFmtSpec)

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

- EXTERNXML int rtXmlEncDoubleNormalValue (OSCTXT * pctxt, OSREAL value, const OSDoubleFmt * pFmtSpec, int defaultPrecision)

  This function encodes a normal (not +/-INF or NaN) value of the XSD double or float type.

- 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, OSXMLNamespace * pNS, OSRTDList * pNSAttrs, OSBOOL terminate)

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

- EXTERNXML int rtXmlEncEndDocument (OSCTXT * pctxt)

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

- EXTERNXML int rtXmlEncEndElement (OSCTXT * pctxt, const OSUTF8CHAR * elemName, OSXMLNamespace * 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, OSXMLNamespace * pNS, const OSDoubleFmt * pFmtSpec)

  This function encodes a variable of the XSD float type.

- EXTERNXML int rtXmlEncFloatAttr (OSCTXT * pctxt, OSREAL value, const OSUTF8CHAR * attrName, OSSIZE 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, OSSIZE 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, OSSIZE attrNameLen)

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

- EXTERNXML int rtXmlEncHexStrValue (OSCTXT * pctxt, OSSIZE 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, OSXML-Namespace * 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, OSSIZE 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 rtXmlEncUIntPattern (OSCTXT * pctxt, OSUINT32 value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)

- EXTERNXML int rtXmlEncUIntPatternValue (OSCTXT * pctxt, OSUINT32 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 rtXmlEncInt64Pattern (OSCTXT * pctxt, OSINT64 value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)

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

  This function encodes a variable of the XSD integer type.

- EXTERNXML int rtXmlEncInt64PatternValue (OSCTXT * pctxt, OSINT64 value, const OSUTF8CHAR * pattern)

- EXTERNXML int rtXmlEncInt64Attr (OSCTXT * pctxt, OSINT64 value, const OSUTF8CHAR * attrName, OSSIZE attrNameLen)

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

- EXTERNXML int rtXmlEncNamedBits (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSSIZE 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, OSSIZE 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 rtXmlPrintNSAttrs (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, OSSIZE 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, OSSIZE nameLen, const OSUTF8CHAR * value, OSSIZE valueLen, OSSIZE 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, OSRTDList * pNSAttrs)

  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, OSSIZE 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, OSSIZE 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, OSSIZE nameLen, const OSUTF8CHAR * value, OSSIZE 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, OSSIZE 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 rtXmlEncUInt64Pattern (OSCTXT * pctxt, OSUINT64 value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)

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

  This function encodes a variable of the XSD integer type.

- EXTERNXML int rtXmlEncUInt64PatternValue (OSCTXT * pctxt, OSUINT64 value, const OSUTF8CHAR * pattern)

- EXTERNXML int rtXmlEncUInt64Attr (OSCTXT * pctxt, OSUINT64 value, const OSUTF8CHAR * attrName, OSSIZE 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 rtXmlEncXSITypeAttr2 (OSCTXT * pctxt, const OSUTF8CHAR * typeNsUri, const OSUTF8CHAR * typeName)

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

- EXTERNXML int rtXmlEncXSINilAttr (OSCTXT * pctxt)

  This function encodes an XML nil attribute (xsi:nil="true").

- 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, OSSIZE len)

- EXTERNXML int rtXmlSetEncBufPtr (OSCTXT * pctxt, OSOCTET * bufaddr, OSSIZE 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).

- 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.

- EXTERNXML int rtXmlSetEncC14N (OSCTXT * pctxt, OSBOOL value)

  This function sets the option to encode in C14N mode.

- 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.

- EXTERNXML int rtXmlSetEncXSINilAttr (OSCTXT * pctxt, OSBOOL value)

  This function sets a flag in the context that indicates the XSI attribute declaration (xmlns:xsi) should be added to the encoded XML instance.

- EXTERNXML int rtXmlSetDigitsFacets (OSCTXT * pctxt, int totalDigits, int fractionDigits)

- EXTERNXML int rtXmlSetEncDocHdr (OSCTXT * pctxt, OSBOOL value)

  This function sets the option to add the XML document header (i.e.

- EXTERNXML int rtXmlSetEncodingStr (OSCTXT * pctxt, const OSUTF8CHAR * encodingStr)

  This function sets the XML output encoding to the given value.

- EXTERNXML int rtXmlSetFormatting (OSCTXT * pctxt, OSBOOL doFormatting)

  This function sets XML output formatting to the given value.

- EXTERNXML int rtXmlSetIndent (OSCTXT * pctxt, OSUINT8 indent)

  This function sets XML output indent to the given value.

- EXTERNXML int rtXmlSetIndentChar (OSCTXT * pctxt, char indentChar)

  This function sets XML output indent character to the given value.

- EXTERNXML void rtXmlSetNamespacesSet (OSCTXT * pctxt, OSBOOL value)

  This function sets the context 'namespaces are set' flag.

- EXTERNXML int rtXmlSetNSPrefixLinks (OSCTXT * pctxt, OSRTDList * pNSAttrs)

  This function sets namespace prefix/URI links in the namspace prefix stack in the context structure.

- 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.

- 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.

- EXTERNXML void rtXmlSetSoapVersion (OSCTXT * pctxt, OSUINT8 version)

  This function sets the SOAP version number.

- EXTERNXML int rtXmlSetXSITypeAttr (OSCTXT * pctxt, const OSUTF8CHAR * xsiType)

  This function sets the XML Schema Instance (xsi) type attribute value.

- EXTERNXML int rtXmlSetWriteBOM (OSCTXT * pctxt, OSBOOL write)

  This function sets whether the Unicode byte order mark is encoded.

- EXTERNXML int rtXmlMatchHexStr (OSCTXT * pctxt, OSSIZE minLength, OSSIZE maxLength)

  This function tests the context buffer for containing a correct hexadecimal string.

- EXTERNXML int rtXmlMatchBase64Str (OSCTXT * pctxt, OSSIZE minLength, OSSIZE maxLength)

  This function tests the context buffer for containing a correct base64 string.

- EXTERNXML int rtXmlMatchDate (OSCTXT * pctxt)

  This function tests the context buffer for containing a correct date string.

- EXTERNXML int rtXmlMatchTime (OSCTXT * pctxt)

  This function tests the context buffer for containing a correct time string.

- EXTERNXML int rtXmlMatchDateTime (OSCTXT * pctxt)

  This function tests the context buffer for containing a correct dateTime string.

- EXTERNXML int rtXmlMatchGYear (OSCTXT * pctxt)

  This function tests the context buffer for containing a correct gYear string.

- EXTERNXML int rtXmlMatchGYearMonth (OSCTXT * pctxt)

  This function tests the context buffer for containing a correct gYearMonth string.

- EXTERNXML int rtXmlMatchGMonth (OSCTXT * pctxt)

  This function tests the context buffer for containing a correct gMonth string.

- EXTERNXML int rtXmlMatchGMonthDay (OSCTXT * pctxt)

This function tests the context buffer for containing a correct gMonthDay string.

- EXTERNXML int rtXmlMatchGDay (OSCTXT * pctxt)

  This function tests the context buffer for containing a correct gDay string.

- EXTERNXML OSUTF8CHAR * rtXmlNewQName (OSCTXT * pctxt, const OSUTF8CHAR * localName, const OSUTF8CHAR * prefix)

  This function creates a new QName given the localName and prefix parts.

- EXTERNXML OSBOOL rtXmlCmpBase64Str (OSUINT32 nocts1, const OSOCTET * data1, const OSUTF8CHAR * data2)

  This function compares an array of octets to a base64 string.

- EXTERNXML OSBOOL rtXmlCmpHexStr (OSUINT32 nocts1, 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, OSSIZE 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 \c 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 mathes 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, OSSIZE 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 int rtXmlWriteUTF16ToFile (OSCTXT * pctxt, const char * filename)
- EXTERNXML void rtXmlTreatWhitespaces (OSCTXT * pctxt, int whiteSpaceType)
- EXTERNXML int rtXmlCheckBuffer (OSCTXT * pctxt, OSSIZE byte_count)
- EXTERNXML void rtErrXmlInit (OSVOIDARG)
- EXTERNXML int rtXmlPutChar (OSCTXT * pctxt, const OSUTF8CHAR value)
- EXTERNXML int rtXmlWriteChars (OSCTXT * pctxt, const OSUTF8CHAR * value, OSSIZE len)

- 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 rtXmlpDecAny2 (OSCTXT * pctxt, OSUTF8CHAR ** pvalue)

  This version of the rtXmlpDecAny function returns the string in a mutable buffer.

- EXTERNXML int rtXmlpDecAnyAttrStr (OSCTXT * pctxt, const OSUTF8CHAR ** ppAttrStr, OSSIZE attrIndex)

  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, OSSIZE bufsize)

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

- EXTERNXML int rtXmlpDecBase64Str64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlpDecBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

- 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 rtXmlpDecBitString64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE bufsize)

  This function is identical to rtXmlpDecBitString except that it supports lengths up to 64-bits in size on 64-bit machines.

- EXTERNXML int rtXmlpDecBitStringExt (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnbits, OSOCTET ** ppextdata, OSUINT32 bufsize)

  This function decodes a bit string value.

- EXTERNXML int rtXmlpDecBitStringExt64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnbits, OSOCTET ** ppextdata, OSSIZE bufsize)

  This function is identical to rtXmlpDecBitStringExt except that it supports lengths up to 64-bits in size on 64-bit machines.

- 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 fractionDigits)

  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 rtXmlpDecDoubleExt (OSCTXT * pctxt, OSUINT8 flags, 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 rtXmlpDecDynBase64Str64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlpDecDynBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

- 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 rtXmlpDecDynHexStr64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to the rtXmlpDecDynHexStr except that it supports lengths up to 64 bits in size on 64-bit systems.

- EXTERNXML int rtXmlpDecDynUnicodeStr (OSCTXT * pctxt, const OSUNICHAR ** ppdata, OSSIZE * 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, OSSIZE 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, OSSIZE bufsize)

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

- EXTERNXML int rtXmlpDecHexStr64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlpDecHexStr except that it supports lengths up to 64-bits in size on 64-bit machines.

- 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 rtXmlpDecNamedBits64 (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE 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 rtXmlpDecXSIAttr (OSCTXT * pctxt, const OSXMLNameFragments * attrName)

  This function decodes XSI (XML Schema Instance) 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, OSSIZE 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, OSSIZE 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 * pctxt, OSXMLNameFragments * pAttr, OSINT16 * nsidx, OSSIZE attrIndex)

  This function selects attribute to decode.

- EXTERNXML OSINT32 rtXmlpGetCurrentLevel (OSCTXT * pctxt)

  This function returns current nesting level.

- EXTERNXML void rtXmlpSetWhiteSpaceMode (OSCTXT * pctxt, OSXMLWhiteSpaceMode whiteSpaceMode)

  Sets the whitespace treatment mode.

- EXTERNXML OSBOOL rtXmlpSetMixedContentMode (OSCTXT * pctxt, OSBOOL mixedContentMode)

  Sets mixed content mode.

- EXTERNXML void rtXmlpSetListMode (OSCTXT * pctxt)

  Sets list mode.

- EXTERNXML OSBOOL rtXmlpListHasItem (OSCTXT * pctxt)

  Check for end of decoded token list.

- EXTERNXML void rtXmlpCountListItems (OSCTXT * pctxt, OSSIZE * itemCnt)

  Count tokens in list.

- EXTERNXML int rtXmlpGetNextSeqElemID2 (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXMLGroupDesc * pGroup, int groups, int curID, int lastMandatoryID, OSBOOL groupMode, OSBOOL check-Repeat)

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

- EXTERNXML int rtXmlpGetNextSeqElemID (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXML-GroupDesc * pGroup, int curID, int lastMandatoryID, OSBOOL groupMode)

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

- EXTERNXML int rtXmlpGetNextSeqElemIDExt (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXMLGroupDesc * ppGroup, const OSBOOL * extRequired, int postExtRootID, int curID, int lastMandatoryID, OSBOOL groupMode)

  This is an ASN.1 extension-supporting version of rtXmlpGetNextSeqElemID.

- EXTERNXML int rtXmlpGetNextAllElemID (OSCTXT * pctxt, const OSXMLElemIDRec * tab, OSSIZE nrows, const OSUINT8 * pOrder, OSSIZE nOrder, OSSIZE maxOrder, int anyID)

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

- EXTERNXML int rtXmlpGetNextAllElemID16 (OSCTXT * pctxt, const OSXMLElemIDRec * tab, OSSIZE nrows, const OSUINT16 * pOrder, OSSIZE nOrder, OSSIZE maxOrder, int anyID)

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

- EXTERNXML int rtXmlpGetNextAllElemID32 (OSCTXT * pctxt, const OSXMLElemIDRec * tab, OSSIZE nrows, const OSUINT32 * pOrder, OSSIZE nOrder, OSSIZE maxOrder, int anyID)

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

- EXTERNXML void rtXmlpSetNamespaceTable (OSCTXT * pctxt, const OSUTF8CHAR * namespaceTable, OSSIZE nmNamespaces)

  Sets user namespace table.

- EXTERNXML int rtXmlpCreateReader (OSCTXT * pctxt)

  Creates pull parser reader structure within the context.

- EXTERNXML void rtXmlpHideAttributes (OSCTXT * pctxt)

  Disable access to attributes.

- EXTERNXML OSBOOL rtXmlpNeedDecodeAttributes (OSCTXT * pctxt)

  This function checks if attributes were previously decoded.

- EXTERNXML void rtXmlpMarkPos (OSCTXT * pctxt)

  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, OSSIZE * 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 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 struct OSXMLReader * rtXmlpGetReader (OSCTXT * pctxt)

  This function fetches the XML reader structure from the context for use in low-level pull parser calls.

- EXTERNXML OSBOOL rtXmlpIsLastEventDone (OSCTXT * pctxt)

  Check processing status of current tag.

- EXTERNXML int rtXmlpGetXSITypeIndex (OSCTXT * pctxt, const OSXMLItemDescr typetab, OSSIZE typetabsiz)

  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, OSINT16 xsiTypeIdx, const OSXMLItemDescr typetab, OSSIZE 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.

- EXTERNXML OSBOOL rtXmlpIsUTF8Encoding (OSCTXT * pctxt)

  This function checks if the encoding specified in XML header is UTF-8.

- EXTERNXML int rtXmlpReadBytes (OSCTXT * pctxt, OSOCTET * pbuf, OSSIZE nbytes)

  This function reads the specified number of bytes directly from the underlying XML parser stream.

Detailed Description

XML low-level C encode/decode functions.

Definition in file osrtxml.h

osrtxml.hh File Reference

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

```
#include "rtxsrc/rtxErrCodes.h"
```

Macros

- #define BASE64TOINT ((((unsigned)(c)) < 128) ? decodeTable [(c) - 40] : -1)

- #define DEFAULT_FLOAT_PRECISION 6

- #define DEFAULT_DOUBLE_PRECISION 11
- #define OSRT_CHECK_EVAL_DATE_STAT0
- #define OSRT_CHECK_EVAL_DATE_STAT1
- #define OSRT_CHECK_EVAL_DATE0
- #define OSRT_CHECK_EVAL_DATE1

Variables

- const signed char decodeTable

Functions

- int rtXmlFreeCtxtAppInfo (OSCTXT * pctxt)
- int rtXmlResetCtxtAppInfo (OSCTXT * pctxt)
- OSXMLCtxtInfo * rtXmlCtxtAppInfoDup (OSCTXT * pctxt, OSCTXT * pDstCtxt)
- void rtXmlSetCtxtAppInfo (OSCTXT * pctxt, OSXMLCtxtInfo * pXMLInfo)
- int rtXmlUpdateBOM (OSXMLCtxtInfo * pXMLInfo, OSXMLEncoding encoding)
- int rtXmlSetEncoding (OSCTXT * pctxt, OSXMLEncoding encoding)
- int rtXmlEncStartAttrC14N (OSCTXT * pctxt)
- int rtXmlEncEndAttrC14N (OSCTXT * pctxt)
- int rtXmlEncParseAny (OSCTXT * pctxt, OSCTXT * parseCtxt, const OSUTF8CHAR * pvalue, size_t valueLen, OSBOOL anyType)
- int rtXmlSetSchemaLocationByStrFrag (OSCTXT * pctxt, const OSUTF8CHAR * schemaLocation, size_t nbytes)
- int rtXmlSetNoNSSchemaLocationByStrFrag (OSCTXT * pctxt, const OSUTF8CHAR * schemaLocation, size_t nbytes)
- void rtXmlpFreeReader (OSCTXT * pctxt, OSXMLCtxtInfo * pXMLInfo)
- EXTERNXML OSBOOL rtXmlpIsInGroup (int elemID, int grpId, const OSBOOL * grpTab, int nElems)
- EXTERNXML void rtXmlpGetContent (OSCTXT * pctxt, int level)
- EXTERNXML OSBOOL rtXmlpMatchElemId (OSCTXT * pctxt, int elemID, int matchingID)
- EXTERNXML OSBOOL rtXmlpIsContentMode (OSCTXT * pctxt)

Detailed Description

Definition in file osrtxml.hh

OSXMLDecodeBuffer.cpp File Reference

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#include "rtxsrc/OSRTFileInputStream.h"
```

```
#include "rtxsrc/OSRTMemoryInputStream.h"
```

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

Detailed Description

Definition in file OSXMLDecodeBuffer.cpp

OSXMLEncodeBuffer.cpp File Reference

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

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

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

```
#include "rtxsrc/osMacros.h"
```

```
#include "rtxsrc/rtxFile.h"
```

Detailed Description

Definition in file OSXMLEncodeBuffer.cpp

OSXMLEncodeStream.cpp File Reference

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

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

Detailed Description

Definition in file OSXMLEncodeStream.cpp

OSXMLMessageBuffer.cpp File Reference

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

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

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

Detailed Description

Definition in file OSXMLMessageBuffer.cpp

OSXSDAnyTypeClass.cpp File Reference

```
#include "OSXSDAnyTypeClass.h"
```

Detailed Description

C++ run-time base class for anyType.

Definition in file OSXSDAnyTypeClass.cpp

OSXSDComplexType.cpp File Reference

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

```
#include "rtxsrc/osMacros.h"
```

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

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

Detailed Description

Definition in file OSXSDComplexType.cpp

rtDomAddAttr.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtdomsrc/osrtdom.h"
```

Functions

- int rtDomAddAttr (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, const OSUTF8CHAR * attrName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Adds attribute to the node.

Detailed Description

Definition in file rtDomAddAttr.c

rtDomAddNode.c File Reference

```
#include "rtdomsrc/osrtdom.h"
```

Functions

- int rtDomAddNode (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  Adds child node to the parent's node.

Detailed Description

Definition in file rtDomAddNode.c

rtDomAddNSAttrs.c File Reference

```
#include "rtdomsrc/osrtdom.h"

#include "rtxmlsrc/osrtxml.h"

#include "rtxsrc/rtxErrCodes.h"
```

Functions

- static OSBOOL previouslyEncoded (OSRTDList * pEncNSList, OSXMLNamespace * pNS)
- int rtDomAddNSAttrs (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr rootNode, OSRT-DList * pNSAttrs)

  Adds namespace attributes to the root node.

Detailed Description

Definition in file rtDomAddNSAttrs.c

rtDomAddSubTree.c File Reference

```
#include <stdlib.h>

#include "rtdomsrc/domAPI.h"

#include "rtxsrc/rtxErrCodes.h"

#include "rtxmlsrc/osrtxml.h"

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

Data Structures

- struct rtdom_SaxHandler

Typedefs

- typedef struct rtdom_SaxHandler rtdom_SaxHandler

Functions

- int SAX_rtdom_startElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname, const OSUTF8CHAR *const * atts)
- int SAX_rtdom_endElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname)
- int SAX_rtdom_characters (void * userData, const OSUTF8CHAR * s, OSSIZE len)
- void SAX_rtdom_init (OSCTXT * pctxt, rtdom_SaxHandler * pSaxHandler, OSRTDOMDocPtr xmlDoc, OSRT-DOMNodePtr node)
- int rtDomAddSubTree (OSCTXT * pctxt, OSRTDOMDocPtr xmlDoc, OSRTDOMNodePtr node, const OSUTF8CHAR * pXmlData, size_t dataLen)

  This function adds the subtree to the specified node.

Detailed Description

Definition in file rtDomAddSubTree.c

rtDomDecodeDoc.c File Reference

```
#include "rtdomsrc/osrtdom.h"
```

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

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- OSBOOL lookupNewURI (OSCTXT * pctxt, const OSUTF8CHAR ** attrs, const OSUTF8CHAR ** end, const OSUTF8CHAR * uri, const OSUTF8CHAR ** pPrefix)
- OSRTDOMError prepareQName (OSCTXT * pctxt, const OSUTF8CHAR * prefix, const OSUTF8CHAR * localname, OSUTF8CHAR ** qname)
- OSRTDOMError rtDomDecodeSubtree (OSRTDOMDocPtr domDoc, OSRTDOMNodePtr topChild, struct OSSAXHandlerBase * pSaxBase)
- int rtDomDecodeDoc (OSRTDOMDocPtr domDoc, struct OSSAXHandlerBase * pSaxBase)

  This function starts decoding of the DOM document.

Detailed Description

Definition in file rtDomDecodeDoc.c

rtDomEncAny.c File Reference

```
#include "rtdomsrc/osrtdom.h"
```

Functions

- int rtDomEncAny (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr parentNode, const OSXMLSTRING * pXmlData, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, OSRTDList * pNSAttrs)

  This function encodes a variable of the XSD any type.

Detailed Description

Definition in file rtDomEncAny.c

rtDomEncAnyAttr.c File Reference

```
#include "rtdomsrc/osrtdom.h"
```

Functions

- int rtDomEncAnyAttr (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSRTDList * pAnyAttrList)

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

Detailed Description

Definition in file rtDomEncAnyAttr.c

rtDomEncString.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtdomsrc/osrtdom.h"
```

Functions

- int rtDomEncStringValue (OSCTXT * pctxt, const OSUTF8CHAR * pvalue)

  This function encodes a variable of the XSD string type.

- int rtDomEncString (OSCTXT * pctxt, OSXMLSTRING * pvalue)

  This function encodes a variable of the XSD string type.

Detailed Description

Definition in file rtDomEncString.c

rtDomEncXSIAttrs.c File Reference

```
#include "rtdomsrc/osrtdom.h"
```

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

```
#include "rtxsrc/rtxErrCodes.h"
```

Variables

- static const OSUTF8CHAR XSI_NS
- static const OSUTF8CHAR XSI_TYPE

Functions

- int rtDomEncXSIAttrs (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr node, OSBOOL needXSI)

  Adds XSI attributes to the node.

Detailed Description

Definition in file rtDomEncXSIAttrs.c

rtDomSetNode.c File Reference

```
#include "rtdomsrc/osrtdom.h"
```

Functions

- int rtDomSetNode (OSCTXT * pctxt, OSRTDOMDocPtr domDoc, OSRTDOMNodePtr curNode)

  Adds content or CDATA section to the node.

Detailed Description

Definition in file rtDomSetNode.c

rtLx2Dom.h File Reference

```
#include "libxml/tree.h"
```

```
#include "rtxsrc/rtxMemBuf.h"
```

Functions

- int lx2DomSerializeToMem (xmlDocPtr doc, xmlNodePtr node, OSRTMEMBUF * pMemBuf)

Detailed Description

Definition in file rtLx2Dom.h

rtLx2DomSerializeToMem.c File Reference

```
#include <string.h>
```

```
#include "rtxsrc/rtxMemBuf.h"

#include "rtxsrc/rtxErrCodes.h"

#include "rtxmlsrc/rtLx2Dom.h"
```

Macros

- #define DTD_SUPPORT 1
- #define OUT_ENTITY_REF_CONTENT 0
- #define append rtxMemBufAppend (pMemBuf, (const OSOCTET*)str, len)
- #define appendStr rtxMemBufAppend (pMemBuf, (const OSOCTET*)str, OSCRTLSTRLEN(str))
- #define appendChar c = c1, rtxMemBufAppend (pMemBuf, (const OSOCTET*)&c, 1)
- #define DECL_C char c

Variables

- static const char *const ocur
- static const char *const attrName
- static const char *const attrDef

Functions

- static void lx2SerializeAttrValue (const char * value, OSRTMEMBUF * pMemBuf)
- static void lx2SerializeElementContent (xmlElementContentPtr cont, OSRTMEMBUF * pMemBuf)
- int lx2DomSerializeToMem (xmlDocPtr doc, xmlNodePtr node, OSRTMEMBUF * pMemBuf)

Detailed Description

Definition in file rtLx2DomSerializeToMem.c

rtSaxCAny.c File Reference

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

#include "rtxmlsrc/osrtxml.hh"

#include "rtxsrc/rtxContext.hh"
```

Functions

- EXTXMLMETHOD int SAXAnyStartElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname, const OSUTF8CHAR *const * attrs)
- EXTXMLMETHOD int SAXAnyCharacters (void * userData, const OSUTF8CHAR * chars, OSSIZE length)

- static OSBOOL SAX_any_isEmptyElement (AnySaxHandler * pSaxHandler, const OSUTF8CHAR * qname)
- EXTXMLMETHOD int SAXAnyEndElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname)
- EXTXMLMETHOD void SAXAnyInit (OSCTXT * pctxt, AnySaxHandler * pSaxHandler, OSXMLSTRING * pvalue, int level)
- EXTXMLMETHOD int SAXAnyElementInit (OSCTXT * pctxt, AnySaxHandler * pSaxHandler, OSXMLSTRING * pvalue, const OSUTF8CHAR * elemName)
- EXTXMLMETHOD void SAXAnyFree (OSCTXT * pctxt, AnySaxHandler * pSaxHandler)

Detailed Description

Definition in file rtSaxCAny.c

rtSaxCAnyType.c File Reference

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

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

```
#include "rtxsrc/rtxContext.hh"
```

Functions

- EXTXMLMETHOD int SAXAnyTypeStartElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname, const OSUTF8CHAR *const * attrs)
- EXTXMLMETHOD int SAXAnyTypeCharacters (void * userData, const OSUTF8CHAR * chars, OSSIZE length)
- static OSBOOL SAX_anytype_isEmptyElement (AnyTypeSaxHandler * pSaxHandler, const OSUTF8CHAR * qname)
- EXTXMLMETHOD int SAXAnyTypeEndElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname)
- EXTXMLMETHOD void SAXAnyTypeInit (OSCTXT * pctxt, AnyTypeSaxHandler * pSaxHandler, OSXSDAnyType * pvalue, int level)
- EXTXMLMETHOD int SAXAnyTypeElementInit (OSCTXT * pctxt, AnyTypeSaxHandler * pSaxHandler, OSXSDAnyType * pvalue, const OSUTF8CHAR * elemName)
- EXTXMLMETHOD void SAXAnyTypeFree (OSCTXT * pctxt, AnyTypeSaxHandler * pSaxHandler)

Detailed Description

Definition in file rtSaxCAnyType.c

rtSaxCppAny.cpp File Reference

```
#include "rtxsrc/rtxCommon.hh"
```

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxToken.h"`

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxmlsrc/rtSaxCppAny.h"`

Detailed Description

Definition in file rtSaxCppAny.cpp

rtSaxCppAnyType.cpp File Reference

`#include "rtxsrc/rtxCommon.hh"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxToken.h"`

`#include "rtxmlsrc/rtSaxCppAnyType.h"`

Detailed Description

Definition in file rtSaxCppAnyType.cpp

rtSaxCppParser.cpp File Reference

`#include "rtxmlsrc/rtSaxCppParser.h"`

`#include "rtxsrc/OSRTFileInputStream.h"`

`#include "rtxsrc/OSRTMemoryInputStream.h"`

Macros

- #define XMLSTRPRINT printf((const char*)a)

Detailed Description

Definition in file rtSaxCppParser.cpp

rtSaxCppSimpleType.cpp File Reference

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxToken.h"`

`#include "rtxmlsrc/rtSaxCppSimpleType.h"`

Detailed Description

Definition in file rtSaxCppSimpleType.cpp

rtSaxCppSoap.cpp File Reference

```
#include "rtxsrc/rtxCommon.hh"
#include "rtxsrc/rtxErrCodes.h"
#include "rtxsrc/rtxToken.h"
#include "rtxmlsrc/rtSaxCppSoap.h"
```

Macros

- #define SOAPENVELOPE "http://schemas.xmlsoap.org/soap/envelope/"
- #define SOAP12ENVELOPE "http://www.w3.org/2003/05/soap-envelope"

Detailed Description

Definition in file rtSaxCppSoap.cpp

rtSaxCppStrList.cpp File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "rtxmlsrc/osrtxml.h"
#include "rtxsrc/rtxToken.h"
#include "rtxmlsrc/rtSaxCppStrList.h"
#include "rtxsrc/rtxCppXmlString.h"
```

Detailed Description

Definition in file rtSaxCppStrList.cpp

rtSaxCSimpleType.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
#include "rtxsrc/rtxToken.h"
#include "rtxmlsrc/rtSaxCSimpleType.h"
```

Functions

- EXTXMLMETHOD int SAXSimpleTypeStartElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname, const OSUTF8CHAR *const * attrs)

- EXTXMLMETHOD int SAXSimpleTypeCharacters (void * userData, const OSUTF8CHAR * chars, int length)
- EXTXMLMETHOD int SAXSimpleTypeEndElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname)
- EXTXMLMETHOD int SAXSimpleTypeInit (OSCTXT * pctxt, SimpleTypeSaxHandler * pSaxHandler, const OSUTF8CHAR * elemName)
- EXTXMLMETHOD void SAXSimpleTypeFree (OSCTXT * pctxt, SimpleTypeSaxHandler * pSaxHandler)

Detailed Description

Definition in file rtSaxCSimpleType.c

rtSaxCSoap.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtxsrc/rtxToken.h"
```

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

Macros

- #define SOAPENVELOPE "http://schemas.xmlsoap.org/soap/envelope/"
- #define SOAP12ENVELOPE "http://www.w3.org/2003/05/soap-envelope"

Functions

- EXTXMLMETHOD int SAXSoapStartElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname, const OSUTF8CHAR *const * atts)
- EXTXMLMETHOD int SAXSoapCharacters (void * userData, const OSUTF8CHAR * chars, int length)
- EXTXMLMETHOD int SAXSoapEndElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname)
- EXTXMLMETHOD void SAXSoapFree (OSCTXT * pctxt, SoapSaxHandler * pSaxHandler)
- EXTXMLMETHOD int SAXSoapInit (OSCTXT * pctxt, SoapSaxHandler * pSaxHandler, void * pMsgSaxHandler, void * pFaultMsgSaxHandler)
- OSBOOL isSoapEnv (const OSUTF8CHAR * uri)

Detailed Description

Definition in file rtSaxCSoap.c

rtSaxCStrList.c File Reference

```
#include "rtxsrc/rtxTokenConst.h"
```

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxmlsrc/osrtxml.h"`

Functions

- EXTXMLMETHOD int rtSaxStrListParse (OSCTXT * pctxt, OSRTMEMBUF * pMemBuf, OSRTDList * pvalue)

  This function parses the list of strings.

- EXTXMLMETHOD int rtSaxStrListMatch (OSCTXT * pctxt)

  This function mathes the list of strings.

Detailed Description

Definition in file rtSaxCStrList.c

rtSaxCXmlHandler.c File Reference

`#include "rtxmlsrc/rtSaxCXmlHandler.h"`

`#include "rtxsrc/rtxFile.h"`

Functions

- static void indent (FILE * fptr, OSUINT32 nspaces)
- EXTXMLMETHOD int SAX2XMLInitUserData (OSCTXT * pctxt, SAX2XMLUserData * pUserData, const char * outFileName)
- EXTXMLMETHOD int SAX2XMLStartElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname, const OSUTF8CHAR *const * attrs)
- EXTXMLMETHOD int SAX2XMLCharacters (void * userData, const OSUTF8CHAR * chars, int length)
- EXTXMLMETHOD int SAX2XMLEndElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname)

Detailed Description

Definition in file rtSaxCXmlHandler.c

rtSaxCXmlStreamHandler.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxmlsrc/rtSaxCXmlStreamHandler.h"`

Functions

- EXTXMLMETHOD int SAX2XMLStreamStartElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname, const OSUTF8CHAR *const * attrs)

- EXTXMLMETHOD int SAX2XMLStreamCharacters (void * userData, const OSUTF8CHAR * chars, int length)
- EXTXMLMETHOD int SAX2XMLStreamEndElement (void * userData, const OSUTF8CHAR * localname, const OSUTF8CHAR * qname)

Detailed Description

Definition in file rtSaxCXmlStreamHandler.c

rtSaxDiag.c File Reference

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

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

Macros

- #define XMLSTRPRINT printf("%s", (const char*)a)

Functions

- EXTXMLMETHOD void rtSaxDiagStartFunc (OSSAXHandlerBase * pSaxBase, const char * funcName, const OSUTF8CHAR * localName)
- EXTXMLMETHOD void rtSaxDiagEndFunc (OSSAXHandlerBase * pSaxBase, const char * funcName, const OSUTF8CHAR * localName)

Detailed Description

Definition in file rtSaxDiag.c

rtSaxErrors.c File Reference

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

Functions

- EXTXMLMETHOD OSUINT32 rtSaxSetMaxErrors (OSCTXT * pctxt, OSUINT32 maxErrors)
- EXTXMLMETHOD OSUINT32 rtSaxGetMaxErrors (OSCTXT * pctxt)
- EXTXMLMETHOD OSBOOL rtSaxIncErrors (OSCTXT * pctxt)

Detailed Description

Definition in file rtSaxErrors.c

rtSaxFindElemID.c File Reference

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

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

Macros

- #define FUNCNAME rtSaxFindElemID
- #define IDXTYPE OSINT16

Functions

- EXTXMLMETHOD OSINT16 FUNCNAME (OSINT16 * pState, OSINT16 prevElemIdx, const OSUTF8CHAR * localName, const OSSAXElemTableRec idtab, const IDXTYPE * fstab, OSINT16 fstabRows, OSINT16 fstab-Cols)

Detailed Description

Definition in file rtSaxFindElemID.c

rtSaxFindElemID8.c File Reference

```
#include "rtxmlsrc/rtSaxFindElemID.c"
```

Macros

- #define FUNCNAME rtSaxFindElemID8
- #define IDXTYPE OSINT8

Detailed Description

Definition in file rtSaxFindElemID8.c

rtSaxGetAttributeID.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Functions

- EXTXMLMETHOD int rtSaxGetAttributeID (const OSUTF8CHAR * attrName, size_t nAttr, const OSUTF8CHAR * attrNames, OSUINT32 attrPresent)

Detailed Description

Definition in file rtSaxGetAttributeID.c

rtSaxGetAttrValue.c File Reference

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

Functions

- EXTXMLMETHOD 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.

Detailed Description

Definition in file rtSaxGetAttrValue.c

rtSaxGetElemID.c File Reference

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

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

Macros

- #define FUNCNAME rtSaxGetElemID
- #define IDXTYPE OSINT16

Functions

- EXTXMLMETHOD OSINT16 FUNCNAME (OSINT16 * pState, OSINT16 prevElemIdx, const OSUTF8CHAR * localName, const OSSAXElemTableRec idtab, const IDXTYPE * fstab, OSINT16 fstabRows, OSINT16 fstab-Cols)

Detailed Description

Definition in file rtSaxGetElemID.c

rtSaxGetElemID8.c File Reference

```
#include "rtxmlsrc/rtSaxGetElemID.c"
```

Macros

- #define FUNCNAME rtSaxGetElemID8
- #define IDXTYPE OSINT8

Detailed Description

Definition in file rtSaxGetElemID8.c

rtSaxHasXMLNSAttrs.c File Reference

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

Functions

- EXTXMLMETHOD OSBOOL rtSaxHasXMLNSAttrs (const OSUTF8CHAR *const * attrs)

  This function checks if the given attribute list contains one or more XML namespace attributes (xmlns).

Detailed Description

Definition in file rtSaxHasXMLNSAttrs.c

rtSaxIsEmptyBuffer.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

Functions

- EXTXMLMETHOD OSBOOL rtSaxIsEmptyBuffer (OSCTXT * pctxt)

  This function checks if the buffer in the context is empty or not.

Detailed Description

Definition in file rtSaxIsEmptyBuffer.c

rtSaxLookupElemID.c File Reference

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

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

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

Macros

- #define FUNCNAME rtSaxLookupElemID
- #define FINDFUNC rtSaxFindElemID
- #define IDXTYPE OSINT16

Functions

- EXTXMLMETHOD OSINT16 FUNCNAME (OSCTXT * pctxt, OSINT16 * pState, OSINT16 prevElemIdx, const OSUTF8CHAR * localName, const OSUTF8CHAR * qName, const OSSAXElemTableRec idtab, const IDXTYPE * fstab, OSINT16 fstabRows, OSINT16 fstabCols)

Detailed Description

Definition in file rtSaxLookupElemID.c

rtSaxLookupElemID8.c File Reference

`#include "rtxmlsrc/rtSaxLookupElemID.c"`

Macros

- #define FUNCNAME rtSaxLookupElemID8
- #define FINDFUNC rtSaxFindElemID8
- #define IDXTYPE OSINT8

Detailed Description

Definition in file rtSaxLookupElemID8.c

rtSaxParseNSAttrs.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

Functions

- EXTXMLMETHOD int rtSaxParseNSAttrs (OSCTXT * pctxt, const OSUTF8CHAR *const * attrs, OSRTDList * pNSAttrs, const OSUTF8CHAR * nsTable, OSUINT32 nsTableRowCount)

Detailed Description

Definition in file rtSaxParseNSAttrs.c

rtSaxReportUnexpAttrs.c File Reference

`#include "rtxmlsrc/rtSaxDefs.h"`

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxErrCodes.h"`

Functions

- EXTXMLMETHOD int rtSaxReportUnexpAttrs (OSCTXT * pctxt, const OSUTF8CHAR *const * attrs, const char * typeName)

Detailed Description

Definition in file rtSaxReportUnexpAttrs.c

rtSaxSetSkipLevelToCurrent.c File Reference

`#include "rtxmlsrc/rtSaxDefs.h"`

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

Functions

- EXTXMLMETHOD int rtSaxSetSkipLevelToCurrent (OSCTXT * pctxt, int stat)

Detailed Description

Definition in file rtSaxSetSkipLevelToCurrent.c

rtSaxSortAttrs.c File Reference

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

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

Functions

- 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.

Detailed Description

Definition in file rtSaxSortAttrs.c

rtSaxTestAttributesPresent.c File Reference

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

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

```
#include "rtxsrc/rtxCharStr.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtSaxTestAttributesPresent (OSCTXT * pctxt, const OSUINT32 * attrPresent, const OSUINT32 * reqAttrMask, const OSUTF8CHAR *const * attrNames, size_t numOfAttrs, const char * parentTypeName)

Detailed Description

Definition in file rtSaxTestAttributesPresent.c

rtxDomDefs.h File Reference

Macros

- #define EXTERNDOM

Detailed Description

Definition in file rtxDomDefs.h

rtXmlCheckBuffer.c File Reference

```
#include <stdlib.h>
```

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

```
#include "rtxsrc/rtxCommonDefs.h"
```

Functions

- int rtXmlCheckBuffer (OSCTXT * pctxt, size_t byte_count)

Detailed Description

Definition in file rtXmlCheckBuffer.c

rtXmlCmpBase64Str.c File Reference

```
#include "rtxsrc/rtxCtype.h"
```

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

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

Functions

- EXTXMLMETHOD OSBOOL rtXmlCmpBase64Str (OSUINT32 nocts1, const OSOCTET * data1, const OSUTF8CHAR * data2)

  This function compares an array of octets to a base64 string.

Detailed Description

Definition in file rtXmlCmpBase64Str.c

rtXmlCmpHexStr.c File Reference

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

Functions

- EXTXMLMETHOD OSBOOL rtXmlCmpHexStr (OSUINT32 nocts1, const OSOCTET * data1, const OSUTF8CHAR * data2)

  This function compares an array of octets to a hex string.

- EXTXMLMETHOD OSBOOL rtXmlCmpHexChar (OSUTF8CHAR ch, OSOCTET hexval)

Detailed Description

Definition in file rtXmlCmpHexStr.c

rtXmlCmpQName.c File Reference

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

Functions

- EXTXMLMETHOD OSBOOL rtXmlCmpQName (const OSUTF8CHAR * qname1, const OSUTF8CHAR * name2, const OSUTF8CHAR * nsPrefix2)

Detailed Description

Definition in file rtXmlCmpQName.c

rtXmlContext.c File Reference

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

#include "rtxsrc/rtxStream.h"

#include "rtxsrc/rtxContext.hh"

#include "rtXmlCtxtAppInfo.h"
```

Functions

- EXTXMLMETHOD OSXMLCtxtInfo * rtXmlCtxtAppInfoDup (OSCTXT * pctxt, OSCTXT * pDstCtxt)
- EXTXMLMETHOD void rtXmlSetCtxtAppInfo (OSCTXT * pctxt, OSXMLCtxtInfo * pXMLInfo)
- EXTXMLMETHOD int rtXmlGetIndent (OSCTXT * pctxt)

  This function returns current XML output indent value.
- EXTXMLMETHOD int rtXmlGetIndentChar (OSCTXT * pctxt)

  This function returns current XML output indent character value (default is space).
- EXTXMLMETHOD OSBOOL rtXmlGetWriteBOM (OSCTXT * pctxt)

  This function returns whether the Unicode byte order mark will be encoded.
- EXTXMLMETHOD 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.
- EXTXMLMETHOD int rtXmlSetEncC14N (OSCTXT * pctxt, OSBOOL value)

  This function sets the option to encode in C14N mode.

- EXTXMLMETHOD 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.

- EXTXMLMETHOD int rtXmlSetEncXSINilAttr (OSCTXT * pctxt, OSBOOL value)

  This function sets a flag in the context that indicates the XSI attribute declaration (xmlns:xsi) should be added to the encoded XML instance.

- EXTXMLMETHOD int rtXmlSetEncoding (OSCTXT * pctxt, OSXMLEncoding encoding)

- EXTXMLMETHOD int rtXmlUpdateBOM (OSXMLCtxtInfo * pXMLInfo, OSXMLEncoding encoding)

- EXTXMLMETHOD int rtXmlSetFormatting (OSCTXT * pctxt, OSBOOL doFormatting)

  This function sets XML output formatting to the given value.

- EXTXMLMETHOD int rtXmlSetWriteBOM (OSCTXT * pctxt, OSBOOL write)

  This function sets whether the Unicode byte order mark is encoded.

- EXTXMLMETHOD int rtXmlSetIndent (OSCTXT * pctxt, OSUINT8 indent)

  This function sets XML output indent to the given value.

- EXTXMLMETHOD int rtXmlSetIndentChar (OSCTXT * pctxt, char indentChar)

  This function sets XML output indent character to the given value.

- EXTXMLMETHOD 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.

- EXTXMLMETHOD 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.

- EXTXMLMETHOD int rtXmlSetSchemaLocationByStrFrag (OSCTXT * pctxt, const OSUTF8CHAR * schemaLocation, size_t nbytes)

- EXTXMLMETHOD int rtXmlSetNoNSSchemaLocationByStrFrag (OSCTXT * pctxt, const OSUTF8CHAR * schemaLocation, size_t nbytes)

- EXTXMLMETHOD int rtXmlSetXSITypeAttr (OSCTXT * pctxt, const OSUTF8CHAR * xsiType)

  This function sets the XML Schema Instance (xsi) type attribute value.

- EXTXMLMETHOD int rtXmlSetEncDocHdr (OSCTXT * pctxt, OSBOOL value)

  This function sets the option to add the XML document header (i.e.

- EXTXMLMETHOD int rtXmlSetDigitsFacets (OSCTXT * pctxt, int totalDigits, int fractionDigits)

- EXTXMLMETHOD void rtXmlSetNamespacesSet (OSCTXT * pctxt, OSBOOL value)

  This function sets the context 'namespaces are set' flag.

Detailed Description

Definition in file rtXmlContext.c

rtXmlContextInit.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxContext.hh"`

Functions

- EXTXMLMETHOD int rtXmlInitContextUsingKey (OSCTXT * pctxt, const OSOCTET * key, size_t keylen)
- EXTXMLMETHOD int rtXmlInitContext (OSCTXT * pctxt)

  This function initializes a context variable for XML encoding or decoding.

Detailed Description

Definition in file rtXmlContextInit.c

rtXmlCppDecString.cpp File Reference

`#include "rtxsrc/rtxCppXmlString.h"`

`#include "rtxmlsrc/osrtxml.h"`

Detailed Description

Definition in file rtXmlCppDecString.cpp

rtXmlCppEncAny.cpp File Reference

`#include "rtxsrc/rtxCppXmlString.h"`

`#include "rtxmlsrc/rtXmlCppEncFuncs.h"`

Functions

- int rtXmlEncAny (OSCTXT * pctxt, OSRTXMLString * pxmlstr, const OSUTF8CHAR * elemName, OSXML-Namespace * pNS)

Detailed Description

Definition in file rtXmlCppEncAny.cpp

rtXmlCppEncAnyAttr.cpp File Reference

`#include "rtxsrc/rtxCppXmlString.h"`

`#include "rtxmlsrc/OSXSDAnyTypeClass.h"`

`#include "rtxmlsrc/rtXmlCppEncFuncs.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- int rtXmlCppEncAnyAttr (OSCTXT * pctxt, OSRTObjListClass * pAnyAttrList)

Detailed Description

Definition in file rtXmlCppEncAnyAttr.cpp

rtXmlCppEncAnyTypeValue.cpp File Reference

`#include "rtxsrc/rtxCppXmlString.h"`

`#include "rtxmlsrc/OSXSDAnyTypeClass.h"`

`#include "rtxmlsrc/rtXmlCppEncFuncs.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- int rtXmlCppEncAnyTypeValue (OSCTXT * pctxt, OSXSDAnyTypeClass * pvalue)

Detailed Description

Definition in file rtXmlCppEncAnyTypeValue.cpp

rtXmlCppEncStartElement.cpp File Reference

`#include "rtxsrc/rtxCppXmlString.h"`

`#include "rtxmlsrc/rtXmlCppEncFuncs.h"`

`#include "rtxmlsrc/rtXmlEncStartElement.c"`

Macros

- #define RT_XML_CPP_ENC_START_ELEMENT

Detailed Description

Definition in file rtXmlCppEncStartElement.cpp

rtXmlCppEncString.cpp File Reference

`#include "rtxsrc/rtxCppXmlString.h"`

```
#include "rtxsrc/OSRTMsgBufIF.h"
```

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

Functions

- int rtXmlEncString (OSCTXT * pctxt, OSRTXMLString * pxmlstr, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

Detailed Description

Definition in file rtXmlCppEncString.cpp

rtXmlCppNamespace.cpp File Reference

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

Detailed Description

Definition in file rtXmlCppNamespace.cpp

rtXmlCppXSDElement.cpp File Reference

```
#include "rtxsrc/rtxContext.hh"
```

```
#include "rtxsrc/rtxCppTypes.h"
```

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

Detailed Description

Definition in file rtXmlCppXSDElement.cpp

rtXmlCtxtAppInfo.h File Reference

Macros

- #define INITCTXTAPPINFOFUNC rtXmlInitCtxtAppInfo
- #define FREECTXTAPPINFOFUNC rtXmlFreeCtxtAppInfo
- #define RESETCTXTAPPINFOFUNC rtXmlResetCtxtAppInfo

Functions

- EXTXMLMETHOD int FREECTXTAPPINFOFUNC (OSCTXT * pctxt)
- EXTXMLMETHOD int RESETCTXTAPPINFOFUNC (OSCTXT * pctxt)
- EXTXMLMETHOD int INITCTXTAPPINFOFUNC (OSCTXT * pctxt)

Detailed Description

Definition in file rtXmlCtxtAppInfo.h

rtXmlDecBase64Binary.c File Reference

```
#include "rtxsrc/rtxCtype.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlDecBase64Binary (OSRTMEMBUF * pMemBuf, const OSUTF8CHAR * inpdata, OSSIZE length)

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

Detailed Description

Definition in file rtXmlDecBase64Binary.c

rtXmlDecBase64Common.c File Reference

```
#include "rtxsrc/rtxCtype.h"
```

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

Variables

- const signed char decodeTable

Functions

- EXTXMLMETHOD int rtXmlGetBase64StrDecodedLen (const OSUTF8CHAR * inpdata, size_t srcDataSize, size_t * pNumOcts, size_t * pSrcDataLen)

Detailed Description

Definition in file rtXmlDecBase64Common.c

rtXmlDecBase64Str.c File Reference

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

Functions

- EXTXMLMETHOD 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.

- EXTXMLMETHOD int rtXmlDecBase64Str64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlDecBase64Str except that is supports a 64-bit integer length on 64-bit systems.

Detailed Description

Definition in file rtXmlDecBase64Str.c

rtXmlDecBase64StrValue.c File Reference

```
#include "rtxsrc/rtxCtype.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlDecBase64StrValue (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnocts, size_t bufSize, size_t srcDataLen)
- EXTXMLMETHOD int rtXmlDecBase64StrValue64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, size_t bufSize, size_t srcDataLen)

Detailed Description

Definition in file rtXmlDecBase64StrValue.c

rtXmlDecBigInt.c File Reference

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

```
#include "rtxsrc/rtxBigInt.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtxsrc/rtxCtype.h"
```

Functions

- EXTXMLMETHOD int rtXmlDecBigInt (OSCTXT * pctxt, const OSUTF8CHAR ** ppvalue)

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

Detailed Description

Definition in file rtXmlDecBigInt.c

rtXmlDecBool.c File Reference

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

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtxsrc/rtxCtype.h"
```

Functions

- EXTXMLMETHOD int rtXmlDecBool (OSCTXT * pctxt, OSBOOL * pvalue)

  This function decodes a variable of the boolean type.

Detailed Description

Definition in file rtXmlDecBool.c

rtXmlDecDates.c File Reference

```
#include <string.h>
```

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

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlDecGYear (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecGYear:This function decodes a date value from a supplied context and set the pointed OSXSDDateTime to the decoded date value.

- EXTXMLMETHOD int rtXmlDecGYearMonth (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecGYearMonth: This function decodes a time value from a supplied context and set the pointed OSXSD-DateTime structure to the decoded time value.

- EXTXMLMETHOD int rtXmlDecGMonth (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecGMonth: This function decodes a datetime value from a supplied context and set the pointed OSXSDDate-Time to the decoded date and time value.

- EXTXMLMETHOD int rtXmlDecGMonthDay (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecGMonthDay: This function decodes a datetime value from a supplied context and set the pointed OSXSD-DateTime to the decoded date and time value.

- EXTXMLMETHOD int rtXmlDecGDay (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecGDay: This function decodes a datetime value from a supplied context and set the pointed OSXSDDate-Time to the decoded date and time value.

Detailed Description

Definition in file rtXmlDecDates.c

rtXmlDecDateTime.c File Reference

```
#include <string.h>
```

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

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlDecDate (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecDate:This function decodes a date value from a supplied context and set the pointed OSXSDDateTime to the decoded date value.

- EXTXMLMETHOD int rtXmlDecTime (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecTime: This function decodes a time value from a supplied context and set the pointed OSXSDDateTime structure to the decoded time value.

- EXTXMLMETHOD int rtXmlDecDateTime (OSCTXT * pctxt, OSXSDDateTime * pvalue)

  rtXmlDecDateTime: This function decodes a datetime value from a supplied context and set the pointed OSXSD-DateTime to the decoded date and time value.

Detailed Description

Definition in file rtXmlDecDateTime.c

rtXmlDecDecimal.c File Reference

```
#include <math.h>
```

```
#include "rtxsrc/rtxCtype.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlDecDecimal (OSCTXT * pctxt, OSREAL * pvalue)

  This function decodes the contents of a decimal data type.

Detailed Description

Definition in file rtXmlDecDecimal.c

rtXmlDecDouble.c File Reference

```
#include <math.h>
```

```
#include "rtxsrc/rtxCtype.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlDecDouble (OSCTXT * pctxt, OSREAL * pvalue)

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

Detailed Description

Definition in file rtXmlDecDouble.c

rtXmlDecDynBase64Str.c File Reference

`#include "rtxsrc/rtxCtype.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- EXTXMLMETHOD int rtXmlDecDynBase64Str (OSCTXT * pctxt, OSDynOctStr * pvalue)

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

- EXTXMLMETHOD int rtXmlDecDynBase64Str64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlDecDynBase64Str except that it supports a 64-bit integer length on 64-bit systems.

Detailed Description

Definition in file rtXmlDecDynBase64Str.c

rtXmlDecDynHexStr.c File Reference

`#include "rtxsrc/rtxCtype.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- EXTXMLMETHOD int rtXmlDecDynHexStr (OSCTXT * pctxt, OSDynOctStr * pvalue)

  This function decodes a contents of a hexBinary string.

- EXTXMLMETHOD int rtXmlDecDynHexStr64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlDecDynHexStr except that it supports a 64-bit integer length on 64-bit systems.

Detailed Description

Definition in file rtXmlDecDynHexStr.c

rtXmlDecDynUTF8Str.c File Reference

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxContext.hh"`

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxmlsrc/rtSaxCParser.h"`

Functions

- EXTXMLMETHOD int rtXmlDecXmlStr (OSCTXT * pctxt, OSXMLSTRING * outdata)

  This function decodes the contents of an XML string data type.
- EXTXMLMETHOD int rtXmlDecUTF8Str (OSCTXT * pctxt, OSUTF8CHAR * outdata, size_t max_len)
- EXTXMLMETHOD int rtXmlDecDynUTF8Str (OSCTXT * pctxt, const OSUTF8CHAR ** outdata)

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

Detailed Description

Definition in file rtXmlDecDynUTF8Str.c

rtXmlDecHexBinary.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxCtype.h"`

`#include "rtxsrc/rtxErrCodes.h"`

Functions

- EXTXMLMETHOD int rtXmlDecHexBinary (OSRTMEMBUF * pMemBuf, const OSUTF8CHAR * inpdata, OSSIZE length)

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

Detailed Description

Definition in file rtXmlDecHexBinary.c

rtXmlDecHexStr.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- EXTXMLMETHOD int rtXmlDecHexStr (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnocts, OSINT32 bufsize)

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

- EXTXMLMETHOD int rtXmlDecHexStr64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlDecHexStr except that it supports a 64-bit integer length on 64-bit systems.

Detailed Description

Definition in file rtXmlDecHexStr.c

rtXmlDecHexStrValue.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxCtype.h"`

`#include "rtxsrc/rtxErrCodes.h"`

Functions

- EXTXMLMETHOD int rtXmlDecHexStrValue (OSCTXT * pctxt, const OSUTF8CHAR *const inpdata, size_t nbytes, OSOCTET * pvalue, OSUINT32 * pnbits, OSINT32 bufsize)
- EXTXMLMETHOD int rtXmlDecHexStrValue64 (OSCTXT * pctxt, const OSUTF8CHAR *const inpdata, size_t nbytes, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE bufsize)

Detailed Description

Definition in file rtXmlDecHexStrValue.c

rtXmlDecInt.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxsrc/rtxCtype.h"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxContext.hh"`

Variables

- static const OSINT32 maxInt32
- static const OSINT32 maxInt32_10

Functions

- EXTXMLMETHOD int rtXmlDecInt (OSCTXT * pctxt, OSINT32 * pvalue)

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

- EXTXMLMETHOD int rtXmlDecInt8 (OSCTXT * pctxt, OSINT8 * pvalue)

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

- EXTXMLMETHOD int rtXmlDecInt16 (OSCTXT * pctxt, OSINT16 * pvalue)

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

Detailed Description

Definition in file rtXmlDecInt.c

rtXmlDecInt64.c File Reference

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

#include "rtxsrc/rtxCtype.h"

#include "rtxsrc/rtxErrCodes.h"
```

Variables

- static const OSINT64 maxInt64_10

Functions

- EXTXMLMETHOD int rtXmlDecInt64 (OSCTXT * pctxt, OSINT64 * pvalue)

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

Detailed Description

Definition in file rtXmlDecInt64.c

rtXmlDecNSAttr.c File Reference

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

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

Functions

- EXTXMLMETHOD int rtXmlDecNSAttr (OSCTXT * pctxt, const OSUTF8CHAR * attrName, const OSUTF8CHAR * attrValue, OSRTDList * pNSAttrs, const OSUTF8CHAR * nsTable, OSUINT32 nsTableRow-Count)

  This function decodes an XML namespac attribute (xmlns).

Detailed Description

Definition in file rtXmlDecNSAttr.c

rtXmlDecQName.c File Reference

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

Functions

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlDecQName (OSCTXT * pctxt, const OSUTF8CHAR * qname, const OSUTF8CHAR ** prefix)

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

Detailed Description

Definition in file rtXmlDecQName.c

rtXmlDecUInt.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Variables

- static const OSUINT32 maxUInt32
- static const OSUINT32 maxUInt32_10

Functions

- EXTXMLMETHOD int rtXmlDecUInt (OSCTXT * pctxt, OSUINT32 * pvalue)

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

- EXTXMLMETHOD int rtXmlDecUInt8 (OSCTXT * pctxt, OSOCTET * pvalue)
- EXTXMLMETHOD int rtXmlDecUInt16 (OSCTXT * pctxt, OSUINT16 * pvalue)

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

Detailed Description

Definition in file rtXmlDecUInt.c

rtXmlDecUInt64.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Variables

- static const OSUINT64 maxUInt64
- static const OSUINT64 maxUInt64_10

Functions

- EXTXMLMETHOD int rtXmlDecUInt64 (OSCTXT * pctxt, OSUINT64 * pvalue)

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

Detailed Description

Definition in file rtXmlDecUInt64.c

rtXmlDecXSIAttr.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlDecXSIAttr (OSCTXT * pctxt, const OSUTF8CHAR * attrName, const OSUTF8CHAR * attrValue)

  This function decodes XML schema instance (XSI) attribute.

Detailed Description

Definition in file rtXmlDecXSIAttr.c

rtXmlDecXSIAttrs.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlDecXSIAttrs (OSCTXT * pctxt, const OSUTF8CHAR *const * attrs, const char * typeName)

  This function decodes XML schema instance (XSI) attributes.

Detailed Description

Definition in file rtXmlDecXSIAttrs.c

rtXmlEncAny.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxStreamMemory.h"`

`#include "rtxmlsrc/rtXmlPull.hh"`

`#include "rtxsrc/rtxStack.h"`

Functions

- EXTXMLMETHOD int rtXmlEncParseAny (OSCTXT * pctxt, OSCTXT * parseCtxt, const OSUTF8CHAR * pvalue, size_t valueLen, OSBOOL anyType)
- EXTXMLMETHOD int rtXmlEncAny (OSCTXT * pctxt, OSXMLSTRING * pvalue, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

  This function encodes a variable of the XSD any type.
- EXTXMLMETHOD int rtXmlEncAnyStr (OSCTXT * pctxt, const OSUTF8CHAR * pvalue, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

Detailed Description

Definition in file rtXmlEncAny.c

rtXmlEncAnyAttr.c File Reference

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxmlsrc/osrtxml.h"`

Functions

- EXTXMLMETHOD 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.

Detailed Description

Definition in file rtXmlEncAnyAttr.c

rtXmlEncAnyTypeValue.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxsrc/rtxErrCodes.h"`

Functions

- EXTXMLMETHOD int rtXmlEncAnyTypeValue (OSCTXT * pctxt, const OSUTF8CHAR * pvalue)

  This function encodes a variable of the XSD anyType type.

Detailed Description

Definition in file rtXmlEncAnyTypeValue.c

rtXmlEncAttr.c File Reference

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- EXTXMLMETHOD 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.

- EXTXMLMETHOD int rtXmlEncUTF8Attr2 (OSCTXT * pctxt, const OSUTF8CHAR * name, size_t nameLen, const OSUTF8CHAR * value, size_t valueLen)

Detailed Description

Definition in file rtXmlEncAttr.c

rtXmlEncAttrC14N.c File Reference

`#include "rtxsrc/rtxCommonDefs.h"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- static int c14nAttrsCmp (const void * a1, const void * b1, const void * sortCtxt)
- EXTXMLMETHOD int rtXmlEncAttrC14N (OSCTXT * pctxt)

  This function used only in C14 mode.

- EXTXMLMETHOD int rtXmlEncStartAttrC14N (OSCTXT * pctxt)
- EXTXMLMETHOD int rtXmlEncEndAttrC14N (OSCTXT * pctxt)

Detailed Description

Definition in file rtXmlEncAttrC14N.c

rtXmlEncBase64Binary.c File Reference

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

Variables

- static const char encodeTable

Functions

- EXTXMLMETHOD int rtXmlEncBase64StrValue (OSCTXT * pctxt, OSSIZE nocts, const OSOCTET * value)

  This function encodes a variable of the XSD base64Binary type.

- EXTXMLMETHOD int rtXmlEncBase64Binary (OSCTXT * pctxt, OSSIZE nocts, const OSOCTET * value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

  This function encodes a variable of the XSD base64Binary type.

Detailed Description

Definition in file rtXmlEncBase64Binary.c

rtXmlEncBase64BinaryAttr.c File Reference

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

Functions

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

Detailed Description

Definition in file rtXmlEncBase64BinaryAttr.c

rtXmlEncBigInt.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxBigInt.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlEncBigInt (OSCTXT * pctxt, const OSUTF8CHAR * value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

This function encodes a variable of the XSD integer type.

Detailed Description

Definition in file rtXmlEncBigInt.c

rtXmlEncBigIntAttr.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxBigInt.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlEncBigIntAttr (OSCTXT * pctxt, const OSUTF8CHAR * value, const OSUTF8CHAR * attrName, size_t attrNameLen)

Detailed Description

Definition in file rtXmlEncBigIntAttr.c

rtXmlEncBigIntValue.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxBigInt.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlEncBigIntValue (OSCTXT * pctxt, const OSUTF8CHAR * value)

  This function encodes an XSD integer attribute value.

Detailed Description

Definition in file rtXmlEncBigIntValue.c

rtXmlEncBitString.c File Reference

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

```
#include "rtxsrc/rtxCommonDefs.h"
```

Functions

- EXTXMLMETHOD int rtXmlEncBinStrValue (OSCTXT * pctxt, OSSIZE nbits, const OSOCTET * data)

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

- EXTXMLMETHOD int rtXmlEncBitString (OSCTXT * pctxt, OSSIZE nbits, const OSOCTET * value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

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

- EXTXMLMETHOD int rtXmlEncBitStringExt (OSCTXT * pctxt, OSSIZE nbits, const OSOCTET * value, OSSIZE dataSize, const OSOCTET * extValue, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

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

Detailed Description

Definition in file rtXmlEncBitString.c

rtXmlEncBool.c File Reference

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

Functions

- EXTXMLMETHOD int rtXmlEncBoolValue (OSCTXT * pctxt, OSBOOL value)

  This function encodes a variable of the XSD boolean type.

- EXTXMLMETHOD int rtXmlEncBool (OSCTXT * pctxt, OSBOOL value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

  This function encodes a variable of the XSD boolean type.

Detailed Description

Definition in file rtXmlEncBool.c

rtXmlEncBoolAttr.c File Reference

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

Functions

- EXTXMLMETHOD int rtXmlEncBoolAttr (OSCTXT * pctxt, OSBOOL value, const OSUTF8CHAR * attrName, size_t attrNameLen)

Detailed Description

Definition in file rtXmlEncBoolAttr.c

rtXmlEncCanonicalSort.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtxsrc/rtxContext.hh"
```

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

Functions

- EXTXMLMETHOD int rtXmlEncCanonicalSort (OSCTXT * pctxt, OSCTXT * pBufCtxt, OSRTSList * pList)

  Sort (as for CXER, Canonical-XER) and encode previously encoded SET OF components.

Detailed Description

Definition in file rtXmlEncCanonicalSort.c

rtXmlEncComment.c File Reference

```
#include "osrtxml.h"
```

Functions

- EXTXMLMETHOD int rtXmlEncComment (OSCTXT * pctxt, const OSUTF8CHAR * comment)

  This function encodes an XML comment.

Detailed Description

Definition in file rtXmlEncComment.c

rtXmlEncDates.c File Reference

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

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Macros

- #define FUNCNAME rtXmlEncGYear
- #define ENCVALUEFUNC rtXmlEncGYearValue
- #define FUNCNAME rtXmlEncGYearMonth
- #define ENCVALUEFUNC rtXmlEncGYearMonthValue
- #define FUNCNAME rtXmlEncGMonth

- #define ENCVALUEFUNC rtXmlEncGMonthValue
- #define FUNCNAME rtXmlEncGMonthDay
- #define ENCVALUEFUNC rtXmlEncGMonthDayValue
- #define FUNCNAME rtXmlEncGDay
- #define ENCVALUEFUNC rtXmlEncGDayValue

Detailed Description

Definition in file rtXmlEncDates.c

rtXmlEncDatesValue.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxDateTime.hh"`

Functions

- EXTXMLMETHOD int rtXmlEncGYearValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

  This function encodes a numeric gYear value into an XML string representation.
- EXTXMLMETHOD int rtXmlEncGYearMonthValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

  This function encodes a numeric gYearMonth value into an XML string representation.
- EXTXMLMETHOD int rtXmlEncGMonthValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

  This function encodes a numeric gMonth value into an XML string representation.
- EXTXMLMETHOD int rtXmlEncGMonthDayValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

  This function encodes a numeric gMonthDay value into an XML string representation.
- EXTXMLMETHOD int rtXmlEncGDayValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

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

Detailed Description

Definition in file rtXmlEncDatesValue.c

rtXmlEncDateTime.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxmlsrc/rtXmlEncDateTmpl.h"`

Macros

- #define FUNCNAME rtXmlEncDate
- #define ENCVALUEFUNC rtXmlEncDateValue
- #define FUNCNAME rtXmlEncTime
- #define ENCVALUEFUNC rtXmlEncTimeValue
- #define FUNCNAME rtXmlEncDateTime
- #define ENCVALUEFUNC rtXmlEncDateTimeValue

Detailed Description

Definition in file rtXmlEncDateTime.c

rtXmlEncDateTimeValue.c File Reference

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

#include "rtxsrc/rtxErrCodes.h"

#include "rtxsrc/rtxDateTime.hh"

#include "rtxsrc/rtxContext.hh"
```

Functions

- EXTXMLMETHOD int rtXmlEncDatePartValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)
- EXTXMLMETHOD int rtXmlEncDateValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

  This function encodes a variable of the XSD 'date' type as a string.
- EXTXMLMETHOD int rtXmlEncTimeValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

  This function encodes a variable of the XSD 'time' type as an string.
- EXTXMLMETHOD int rtXmlEncDateTimeValue (OSCTXT * pctxt, const OSXSDDateTime * pvalue)

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

Detailed Description

Definition in file rtXmlEncDateTimeValue.c

rtXmlEncDateTmpl.h File Reference

Detailed Description

Definition in file rtXmlEncDateTmpl.h

rtXmlEncDecimal.c File Reference

`#include "rtxsrc/rtxRegExp.hh"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

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

  This function encodes a variable of the XSD decimal type.

Detailed Description

Definition in file rtXmlEncDecimal.c

rtXmlEncDecimalAttr.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

Functions

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

Detailed Description

Definition in file rtXmlEncDecimalAttr.c

rtXmlEncDecimalPattern.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

Data Structures

- struct StateEnc

Macros

- #define MAX_NESTING 16
- #define STATE_ARRAY_SZ 32
- #define STR_SZ 32

Enumerations

- enum Phase {

Sign,
IntPart,
FracPart
}

Typedefs

- typedef struct StateEnc StateEnc

Functions

- static StateEnc * expandStateBuffer (OSCTXT * pctxt, StateEnc * state, unsigned stateArraySz, unsigned curState)
- EXTXMLMETHOD int rtXmlEncDecimalPatternValue (OSCTXT * pctxt, OSREAL value, const OSUTF8CHAR * pattern)

Detailed Description

Definition in file rtXmlEncDecimalPattern.c

rtXmlEncDecimalValue.c File Reference

```
#include <math.h>
```

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

```
#include "rtxsrc/rtxCharStr.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Macros

- #define LG2 0.30102999566398119521373889472449
- #define PRECISION 15 /* the default number of fraction digits for normalized double */
- #define FRACTIONDIGITS 10

Functions

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

Detailed Description

Definition in file rtXmlEncDecimalValue.c

rtXmlEncDouble.c File Reference

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

Macros

- #define FUNCNAME rtXmlEncDouble
- #define PRECISION DEFAULT_DOUBLE_PRECISION

Functions

- EXTXMLMETHOD int FUNCNAME (OSCTXT * pctxt, OSREAL value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSDoubleFmt * pFmtSpec)

Detailed Description

Definition in file rtXmlEncDouble.c

rtXmlEncDoubleAttr.c File Reference

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

Functions

- EXTXMLMETHOD int rtXmlEncDoubleAttr (OSCTXT * pctxt, OSREAL value, const OSUTF8CHAR * attr-Name, size_t attrNameLen, const OSDoubleFmt * pFmtSpec)

Detailed Description

Definition in file rtXmlEncDoubleAttr.c

rtXmlEncDoublePattern.c File Reference

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

#include "rtxsrc/rtxCharStr.h"

#include "rtxsrc/rtxRegExp.hh"

#include "rtxsrc/rtxMemBuf.h"
```

Data Structures

- struct AutoState

Enumerations

- enum @0 {
  OSRTFPS_EMPTY= 0,
  OSRTFPS_LEADING_SIGN= 1,
  OSRTFPS_LEADING_ZEROS= 2,
  OSRTFPS_INT_DIGITS= 3,
  OSRTFPS_POINT= 4,
  OSRTFPS_FRACTION_DIGITS= 5,

OSRTFPS_EXPONENT= 6,
OSRTFPS_EXP_SIGN= 7,
OSRTFPS_EXP_LEADING_ZEROS= 8,
OSRTFPS_EXP_DIGITS= 9
}

Typedefs

- typedef struct AutoState AutoState

Variables

- static const int automateArray

Functions

- static int getNextState (OSUTF8CHAR charVal, int curState)
- static int formatRange (OSRTMEMBUF * pMemBuf, OSREAL value, AutoState * pState, int minNum, int maxNum, int type, int start, int end, OSBOOL neg)
- static int makeIntFormatStr (OSCTXT * pOSCTXT, rtxRegexpPtr regexp, OSREAL value, OSUTF8CHAR ** ppDest)
- EXTXMLMETHOD int rtXmlEncDoublePatternValue (OSCTXT * pctxt, OSREAL value, const OSUTF8CHAR * pattern)
- EXTXMLMETHOD int rtXmlEncDoublePattern (OSCTXT * pctxt, OSREAL value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, int fractionDigits, const OSUTF8CHAR * pattern)

Detailed Description

Definition in file rtXmlEncDoublePattern.c

rtXmlEncDoubleValue.c File Reference

```
#include <math.h>
```

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

```
#include "rtxsrc/rtxContext.hh"
```

Macros

- #define LG2 0.30102999566398119521373889472449

Functions

- EXTXMLMETHOD int rtXmlEncDoubleValue (OSCTXT * pctxt, OSREAL value, const OSDoubleFmt * pFmtSpec, int defaultPrecision)

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

- EXTXMLMETHOD int rtXmlEncDoubleNormalValue (OSCTXT * pctxt, OSREAL value, const OSDoubleFmt * pFmtSpec, int defaultPrecision)

  This function encodes a normal (not +/-INF or NaN) value of the XSD double or float type.

Detailed Description

Definition in file rtXmlEncDoubleValue.c

rtXmlEncEmptyElement.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

Functions

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

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

Detailed Description

Definition in file rtXmlEncEmptyElement.c

rtXmlEncEndDocument.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxsrc/rtxStream.h"`

`#include "rtxsrc/rtxContext.hh"`

Functions

- EXTXMLMETHOD int rtXmlEncEndDocument (OSCTXT * pctxt)

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

Detailed Description

Definition in file rtXmlEncEndDocument.c

rtXmlEncEndElement.c File Reference

`#include "rtxsrc/rtxDList.h"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- static int checkElement (OSCTXT * pctxt, const OSUTF8CHAR * elemName)
- EXTXMLMETHOD int rtXmlEncEndElement (OSCTXT * pctxt, const OSUTF8CHAR * elemName, OSXML-Namespace * pNS)

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

Detailed Description

Definition in file rtXmlEncEndElement.c

rtXmlEncFloat.c File Reference

```
#include "rtxmlsrc/rtXmlEncDouble.c"
```

Macros

- #define FUNCNAME rtXmlEncFloat
- #define PRECISION DEFAULT_FLOAT_PRECISION

Detailed Description

Definition in file rtXmlEncFloat.c

rtXmlEncFloatAttr.c File Reference

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

Functions

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

Detailed Description

Definition in file rtXmlEncFloatAttr.c

rtXmlEncHexBinary.c File Reference

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

Functions

- EXTXMLMETHOD int rtXmlEncHexStrValue (OSCTXT * pctxt, OSSIZE nocts, const OSOCTET * data)

  This function encodes a variable of the XSD hexBinary type.

- EXTXMLMETHOD int rtXmlEncHexBinary (OSCTXT * pctxt, OSSIZE nocts, const OSOCTET * value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

  This function encodes a variable of the XSD hexBinary type.

Detailed Description

Definition in file rtXmlEncHexBinary.c

rtXmlEncHexBinaryAttr.c File Reference

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

Functions

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

Detailed Description

Definition in file rtXmlEncHexBinaryAttr.c

rtXmlEncIndent.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlEncIndent (OSCTXT * pctxt)

  This function adds the given amount of indentation to the output stream.

Detailed Description

Definition in file rtXmlEncIndent.c

rtXmlEncInt.c File Reference

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

```
#include "rtxsrc/osMacros.h"
```

```
#include "rtxsrc/rtxCharStr.h"
```

Macros

- #define RTXMLENCINTFUNC rtXmlEncInt

- #define RTXMLENCINTVALUEFUNC rtXmlEncIntValue
- #define RTXINTTOCHARSTR rtxIntToCharStr
- #define OSINTTYPE OSINT32

Functions

- EXTXMLMETHOD int RTXMLENCINTVALUEFUNC (OSCTXT * pctxt, OSINTTYPE value)
- EXTXMLMETHOD int RTXMLENCINTFUNC (OSCTXT * pctxt, OSINTTYPE value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

Detailed Description

Definition in file rtXmlEncInt.c

rtXmlEncInt64.c File Reference

```
#include "rtXmlEncInt.c"
```

Macros

- #define RTXMLENCINTFUNC rtXmlEncInt64
- #define RTXMLENCINTVALUEFUNC rtXmlEncInt64Value
- #define RTXINTTOCHARSTR rtxInt64ToCharStr
- #define OSINTTYPE OSINT64

Detailed Description

Definition in file rtXmlEncInt64.c

rtXmlEncInt64Attr.c File Reference

```
#include "rtXmlEncIntAttr.c"
```

Macros

- #define RTXMLENCINTFUNC rtXmlEncInt64Attr
- #define RTXINTTOCHARSTR rtxInt64ToCharStr
- #define OSINTTYPE OSINT64

Detailed Description

Definition in file rtXmlEncInt64Attr.c

rtXmlEncInt64Pattern.c File Reference

`#include "rtXmlEncIntPattern.c"`

Macros

- #define RTXMLENCINTPATFUNC rtXmlEncInt64Pattern
- #define RTXMLENCINTPATVALUEFUNC rtXmlEncInt64PatternValue
- #define RTXINTTOCHARSTR rtxInt64ToCharStr
- #define OSINTPATTYPE OSINT64

Detailed Description

Definition in file rtXmlEncInt64Pattern.c

rtXmlEncIntAttr.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxsrc/osMacros.h"`

`#include "rtxsrc/rtxCharStr.h"`

Macros

- #define RTXMLENCINTFUNC rtXmlEncIntAttr
- #define RTXINTTOCHARSTR rtxIntToCharStr
- #define OSINTTYPE OSINT32

Functions

- EXTXMLMETHOD int RTXMLENCINTFUNC (OSCTXT * pctxt, OSINTTYPE value, const OSUTF8CHAR * attrName, size_t attrNameLen)

Detailed Description

Definition in file rtXmlEncIntAttr.c

rtXmlEncIntPattern.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxsrc/osMacros.h"`

`#include "rtxsrc/rtxCharStr.h"`

`#include "rtxsrc/rtxRegExp.hh"`

`#include "rtxsrc/rtxMemBuf.h"`

Macros

- #define RTXMLENCINTPATFUNC rtXmlEncIntPattern
- #define RTXMLENCINTPATVALUEFUNC rtXmlEncIntPatternValue
- #define RTXINTTOCHARSTR rtxIntToCharStr
- #define OSINTPATTYPE OSINT32

Enumerations

- enum @1 {
  OSRTIPS_EMPTY= 0,
  OSRTIPS_DIGITS= 1,
  OSRTIPS_LEADING_SIGN= 2,
  OSRTIPS_LEADING_ZEROS= 4
  }

Functions

- static int formatRange (OSRTMEMBUF * pMemBuf, OSINTPATTYPE value, int * pState, int num, int type, int start, int end, OSBOOL neg)
- static int rtXmlMakeIntFormatStr (OSCTXT * pOSCTXT, rtxRegexpPtr regexp, OSINTPATTYPE value, OSUTF8CHAR ** ppDest)
- EXTXMLMETHOD int RTXMLENCINTPATVALUEFUNC (OSCTXT * pctxt, OSINTPATTYPE value, const OSUTF8CHAR * pattern)
- EXTXMLMETHOD int RTXMLENCINTPATFUNC (OSCTXT * pctxt, OSINTPATTYPE value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)

Detailed Description

Definition in file rtXmlEncIntPattern.c

rtXmlEncNamedBits.c File Reference

`#include "rtxsrc/rtxBitString.h"`

`#include "rtxsrc/rtxErrCodes.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- EXTXMLMETHOD int rtXmlEncNamedBitsValue (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSSIZE nbits, const OSOCTET * pvalue)

- EXTXMLMETHOD int rtXmlEncNamedBits (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSSIZE nbits, const OSOCTET * pvalue, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

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

Detailed Description

Definition in file rtXmlEncNamedBits.c

rtXmlEncNSAttrs.c File Reference

```
#include "rtxsrc/rtxCommon.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Enumerations

- enum Action {
  encAttrs,
  setPfxLinks
  }

Functions

- static OSBOOL previouslyEncoded (OSRTDList * pEncNSList, OSXMLNamespace * pNS)

  Return TRUE if the given OSXMLNamespace record appears in the given list of encoded namespace records.

- static int procNSAttrs (OSCTXT * pctxt, OSRTDList * pNSAttrs, Action action)

  Process the namespace list in accordance with the specified action (see description of action parameter).

- EXTXMLMETHOD int rtXmlEncNSAttrs (OSCTXT * pctxt, OSRTDList * pNSAttrs)

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

- EXTXMLMETHOD int rtXmlSetNSPrefixLinks (OSCTXT * pctxt, OSRTDList * pNSAttrs)

  This function sets namespace prefix/URI links in the namspace prefix stack in the context structure.

Detailed Description

Definition in file rtXmlEncNSAttrs.c

rtXmlEncReal10.c File Reference

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

Functions

- int encXerReal10 (OSCTXT * pctxt, const OSUTF8CHAR * pvalue)

- int encXmlReal10 (OSCTXT * pctxt, const OSUTF8CHAR * pvalue)
- EXTXMLMETHOD 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.

Detailed Description

Definition in file rtXmlEncReal10.c

rtXmlEncSoapArrayTypeAttr.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

```
#include "rtxsrc/osMacros.h"
```

```
#include "rtxsrc/rtxCharStr.h"
```

Functions

- EXTXMLMETHOD int rtXmlEncSoapArrayTypeAttr (OSCTXT * pctxt, const OSUTF8CHAR * name, const OSUTF8CHAR * value, size_t itemCount)
- EXTXMLMETHOD int rtXmlEncSoapArrayTypeAttr2 (OSCTXT * pctxt, const OSUTF8CHAR * name, size_t nameLen, const OSUTF8CHAR * value, size_t valueLen, size_t itemCount)

Detailed Description

Definition in file rtXmlEncSoapArrayTypeAttr.c

rtXmlEncSoapEnvelope.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Variables

- static const char gSoapEnvElemName
- static const char gSoapEnvNSPrefix
- static const char gSoapEnvNSURI
- static const char gSoap12EnvNSURI

Functions

- EXTXMLMETHOD void rtXmlSetSoapVersion (OSCTXT * pctxt, OSUINT8 version)

This function sets the SOAP version number.

- EXTXMLMETHOD int rtXmlEncStartSoapEnv (OSCTXT * pctxt, OSRTDList * pNSAttrs)

  This function encodes a SOAP envelope start element tag.

- EXTXMLMETHOD int rtXmlEncStartSoapElems (OSCTXT * pctxt, OSXMLSOAPMsgType msgtype)

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

- EXTXMLMETHOD int rtXmlEncEndSoapEnv (OSCTXT * pctxt)

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

- EXTXMLMETHOD int rtXmlEncEndSoapElems (OSCTXT * pctxt, OSXMLSOAPMsgType msgtype)

  This function encodes SOAP end element tags.

Detailed Description

Definition in file rtXmlEncSoapEnvelope.c

rtXmlEncStartDocument.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxsrc/rtxStream.h"`

Functions

- EXTXMLMETHOD int rtXmlEncStartDocument (OSCTXT * pctxt)

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

- EXTXMLMETHOD int rtXmlEncBOM (OSCTXT * pctxt)

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

Detailed Description

Definition in file rtXmlEncStartDocument.c

rtXmlEncStartElement.c File Reference

`#include "rtxsrc/rtxDList.h"`

`#include "rtxmlsrc/osrtxml.hh"`

Functions

- EXTXMLMETHOD int rtXmlEncStartElement (OSCTXT * pctxt, const OSUTF8CHAR * elemName, OSXML-Namespace * pNS, OSRTDList * pNSAttrs, OSBOOL terminate)

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

- EXTXMLMETHOD int rtXmlEncTermStartElement (OSCTXT * pctxt)

  This function terminates a currently open XML start element by adding either a '>' or '/>' (if empty) terminator.

Detailed Description

Definition in file rtXmlEncStartElement.c

rtXmlEncString.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtxsrc/rtxContext.hh"
```

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

Functions

- EXTXMLMETHOD int rtXmlEncCDATAStringValue (OSCTXT * pctxt, const OSUTF8CHAR * value, size_t valueLen)
- EXTXMLMETHOD int rtXmlEncStringValue2 (OSCTXT * pctxt, const OSUTF8CHAR * value, size_t valueLen)
- EXTXMLMETHOD int rtXmlEncStringValue (OSCTXT * pctxt, const OSUTF8CHAR * value)

  This function encodes a variable of the XSD string type.

- EXTXMLMETHOD int rtXmlEncString (OSCTXT * pctxt, OSXMLSTRING * pxmlstr, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

  This function encodes a variable of the XSD string type.

- EXTXMLMETHOD int rtXmlEncUTF8Str (OSCTXT * pctxt, const OSUTF8CHAR * value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

  This function encodes a UTF-8 string value.

Detailed Description

Definition in file rtXmlEncString.c

rtXmlEncUInt.c File Reference

```
#include "rtXmlEncInt.c"
```

Macros

- #define RTXMLENCINTFUNC rtXmlEncUInt
- #define RTXMLENCINTVALUEFUNC rtXmlEncUIntValue

- #define RTXINTTOCHARSTR rtxUIntToCharStr
- #define OSINTTYPE OSUINT32

Detailed Description

Definition in file rtXmlEncUInt.c

rtXmlEncUInt64.c File Reference

```
#include "rtXmlEncInt.c"
```

Macros

- #define RTXMLENCINTFUNC rtXmlEncUInt64
- #define RTXMLENCINTVALUEFUNC rtXmlEncUInt64Value
- #define RTXINTTOCHARSTR rtxUInt64ToCharStr
- #define OSINTTYPE OSUINT64

Detailed Description

Definition in file rtXmlEncUInt64.c

rtXmlEncUInt64Attr.c File Reference

```
#include "rtXmlEncIntAttr.c"
```

Macros

- #define RTXMLENCINTFUNC rtXmlEncUInt64Attr
- #define RTXINTTOCHARSTR rtxUInt64ToCharStr
- #define OSINTTYPE OSUINT64

Detailed Description

Definition in file rtXmlEncUInt64Attr.c

rtXmlEncUInt64Pattern.c File Reference

```
#include "rtXmlEncUIntPattern.c"
```

Macros

- #define RTXMLENCUINTPATFUNC rtXmlEncUInt64Pattern

- #define RTXMLENCUINTPATVALUEFUNC rtXmlEncUInt64PatternValue
- #define RTXINTTOCHARSTR rtxUInt64ToCharStr
- #define OSUINTPATTYPE OSUINT64

Detailed Description

Definition in file rtXmlEncUInt64Pattern.c

rtXmlEncUIntAttr.c File Reference

```
#include "rtXmlEncIntAttr.c"
```

Macros

- #define RTXMLENCINTFUNC rtXmlEncUIntAttr
- #define RTXINTTOCHARSTR rtxUIntToCharStr
- #define OSINTTYPE OSUINT32

Detailed Description

Definition in file rtXmlEncUIntAttr.c

rtXmlEncUIntPattern.c File Reference

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

#include "rtxsrc/osMacros.h"

#include "rtxsrc/rtxCharStr.h"

#include "rtxsrc/rtxRegExp.hh"

#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define RTXMLENCUINTPATFUNC rtXmlEncUIntPattern
- #define RTXMLENCUINTPATVALUEFUNC rtXmlEncUIntPatternValue
- #define RTXINTTOCHARSTR rtxUIntToCharStr
- #define OSUINTPATTYPE OSUINT32

Enumerations

- enum @2 {
  OSRTIPS_EMPTY= 0,

OSRTIPS_DIGITS= 1,
OSRTIPS_LEADING_SIGN= 2,
OSRTIPS_LEADING_ZEROS= 4
}

Functions

- static int formatRange (OSRTMEMBUF * pMemBuf, OSUINTPATTYPE value, int * pState, int num, int type, int start, int end, OSBOOL neg)
- static int rtXmlMakeUIntFormatStr (OSCTXT * pOSCTXT, rtxRegexpPtr regexp, OSUINTPATTYPE value, OSUTF8CHAR ** ppDest)
- EXTXMLMETHOD int RTXMLENCUINTPATVALUEFUNC (OSCTXT * pctxt, OSUINTPATTYPE value, const OSUTF8CHAR * pattern)
- EXTXMLMETHOD int RTXMLENCUINTPATFUNC (OSCTXT * pctxt, OSUINTPATTYPE value, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS, const OSUTF8CHAR * pattern)

Detailed Description

Definition in file rtXmlEncUIntPattern.c

rtXmlEncUnicode.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Variables

- static const OSUINT32 encoding_mask
- static const unsigned char encoding_byte

Functions

- static int rtXmlEncUnicodeData (OSCTXT * pctxt, const OSUNICHAR * value, OSSIZE nchars)
- EXTXMLMETHOD int rtXmlEncUnicodeStr (OSCTXT * pctxt, const OSUNICHAR * value, OSSIZE nchars, const OSUTF8CHAR * elemName, OSXMLNamespace * pNS)

  This function encodes a Unicode string value.

Detailed Description

Definition in file rtXmlEncUnicode.c

rtXmlEncXSIAttrs.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Functions

- EXTXMLMETHOD int rtXmlEncXSIAttrs (OSCTXT * pctxt, OSBOOL needXSI)

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

Detailed Description

Definition in file rtXmlEncXSIAttrs.c

rtXmlEncXSINilAttr.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Macros

- #define XSI_NIL_LEN (sizeof(XSI_NIL)-1)

Variables

- static const OSUTF8CHAR XSI_NIL

Functions

- EXTXMLMETHOD int rtXmlEncXSINilAttr (OSCTXT * pctxt)

  This function encodes an XML nil attribute (xsi:nil="true").

Detailed Description

Definition in file rtXmlEncXSINilAttr.c

rtXmlEncXSITypeAttr.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Macros

- #define XSI_TYPE_LEN (sizeof(XSI_TYPE)-1)

Variables

- static const OSUTF8CHAR XSI_TYPE

Functions

- EXTXMLMETHOD int rtXmlEncXSITypeAttr (OSCTXT * pctxt, const OSUTF8CHAR * value)

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

- EXTXMLMETHOD int rtXmlEncXSITypeAttr2 (OSCTXT * pctxt, const OSUTF8CHAR * typeNsUri, const OSUTF8CHAR * typeName)

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

Detailed Description

Definition in file rtXmlEncXSITypeAttr.c

rtXmlErrCodes.h File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

Macros

- #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_ELEMSMISRQ (XML_E_BASE-9)

  Missing required elements.

- #define XML_E_TOOFEWELEMS (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.

- #define XML_E_KEYNOTFOU (XML_E_BASE-16)

  Keyref constraint has some key that not present in refered constraint.

- #define XML_E_DUPLKEY (XML_E_BASE-17)

  Key or unique constraint has duplicated key.

- #define XML_E_FLDABSENT (XML_E_BASE-18)

  Some key has no full set of fields.

- #define XML_E_DUPLFLD (XML_E_BASE-19)

  Some key has more than one value for field.

- #define XML_E_NOTEMPTY (XML_E_BASE-20)

An element was not empty when expected.

Detailed Description

List of numeric status codes that can be returned by ASN1C run-time functions and generated code.

Definition in file rtXmlErrCodes.h

rtXmlError.c File Reference

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

#include "rtxmlsrc/rtXmlErrCodes.h"

#include "rtxsrc/rtxError.h"
```

Variables

- static const char * g_status_text

Functions

- EXTXMLMETHOD void rtErrXmlInit ( )

Detailed Description

Definition in file rtXmlError.c

rtXmlExpatIF.c File Reference

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

#include "rtxmlsrc/rtSaxDefs.h"

#include "rtxsrc/rtxCtype.h"

#include "rtxsrc/rtxStream.h"

#include "rtxsrc/rtxStreamBuffered.h"

#include "rtxsrc/rtxErrCodes.h"

#include "expatsrc/expat.h"
```

Data Structures

- struct XMLReaderImpl

Macros

- #define LSTRX OSCRTLSTRCPY ((char*)rtxMemAlloc(pctxt, OSCRTLSTRLEN (pLStr) + 1), (pLStr))

- #define XML_BUF_SIZE 2048

Typedefs

- typedef struct XMLReaderImpl XMLReaderImpl

Functions

- static void XMLCALL rtSaxCStartCdataSectionHandler (void * userData)
- static void XMLCALL rtSaxCEndCdataSectionHandler (void * userData)
- static void XMLCALL rtSaxCStartElementHandler (void * userData, const XML_Char * name, const XML_Char ** atts)
- static void XMLCALL rtSaxCEndElementHandler (void * userData, const XML_Char * name)
- static void XMLCALL rtSaxCCharacterDataHandler (void * userData, const XML_Char * s, int len)
- int rtSaxCReleaseReader (OSXMLREADER * pReader)
- const XML_LChar * EXML_ErrorString (XML_Parser parser, enum XML_Error code, XML_LChar * buf, int bufSize)
- static const OSUTF8CHAR * parseQName (const OSUTF8CHAR * qname)
- int rtSaxCEnableThreadSafety ( )
- int rtSaxCParse (OSXMLREADER * pReader)
- OSXMLREADER * rtSaxCCreateXmlReader (OSCTXT * pctxt, void * pSaxHandler-Data, CSAX_StartElementHandler pStartElementProc, CSAX_EndElementHandler pEndElementProc, CSAX_CharacterDataHandler pCharactersProc)

Detailed Description

Definition in file rtXmlExpatIF.c

rtXmlExpatIF.cpp File Reference

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

#include "rtxsrc/osMacros.h"

#include "rtxsrc/rtxCtype.h"

#include "rtxsrc/rtxStream.h"

#include "rtxsrc/rtxStreamBuffered.h"

#include "rtxsrc/OSRTInputStreamIF.h"

#include "expatsrc/expat.h"
```

Data Structures

- struct OSExpatXMLReader

Macros

- #define XML_BUF_SIZE 2048

Functions

- static const OSUTF8CHAR * EXML_ErrorString (XML_Parser parser, enum XML_Error code, XML_LChar * buf, int bufSize)
- OSXMLReaderClass * rtSaxCppCreateXmlReader (OSXMLParserCtxtIF * pContext, OSXMLDefaultHandlerIF * pSaxHandler)
- int rtSaxCppEnableThreadSafety ( )
- void rtSaxCppLockXmlLibrary ( )
- void rtSaxCppUnlockXmlLibrary ( )

Detailed Description

Definition in file rtXmlExpatIF.cpp

rtXmlExternDefs.h File Reference

Macros

- #define EXTXMLMETHOD
- #define EXTERNXML
- #define EXTXMLCLASS

Detailed Description

XML external definitions macro.

This is used for Windows to properly declare function scope within DLL's.

Definition in file rtXmlExternDefs.h

rtXmlKeyArray.c File Reference

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

#include "rtxmlsrc/rtXmlErrCodes.h"

#include "rtxsrc/rtxError.h"
```

```
#include "rtxsrc/rtxMemory.h"
```

Functions

- int rtXmlKeyArrayInit (OSCTXT * pctxt, OSXSDKeyArray * pArray, OSUINT32 nmFields, OSBOOL key, const char * name)

  Initialize the array by allocating memory for it.

- void rtXmlKeyArrayFree (OSCTXT * pctxt, OSXSDKeyArray * pArray)
- static int compareFields (const OSXSDKeyRecord * pa, const OSXSDKeyRecord * pb, OSUINT32 nmFields)

  Compare two keys by comparing their respective fields.

- static OSBOOL checkKey (OSXSDKeyArray * pArray)
- static int expand (OSCTXT * pctxt, OSXSDKeyArray * pArray)
- static void addErrParms (OSCTXT * pctxt, OSXSDKeyArray * pArray)
- static int addKey (OSCTXT * pctxt, OSXSDKeyArray * pArray)
- void rtXmlKeyArraySetString (OSXSDKeyArray * pArray, const OSUTF8CHAR * pValue, OSUINT32 fldNum)

  Set the given field's value to the given string value.

- void rtXmlKeyArraySetInt (OSXSDKeyArray * pArray, OSINT32 value, OSUINT32 fldNum)

  Set the given field's value to the given integer value.

- void rtXmlKeyArraySetUInt (OSXSDKeyArray * pArray, OSUINT32 value, OSUINT32 fldNum)

  Set the given field's value to the given unsigned integer value.

- void rtXmlKeyArraySetDecimal (OSXSDKeyArray * pArray, const OSREAL * value, OSUINT32 fldNum)

  Set the given field's value to the given decimal value.

- int rtXmlKeyArrayAdd (OSCTXT * pctxt, OSXSDKeyArray * pArray)

  Once all the fields for a key are set, invoke this method to add the key.

- int rtXmlKeyArrayContains (OSCTXT * pctxt, OSXSDKeyArray * pArray)

  Once all the fields for a key are set, this method is used to check whether the key is already present in the array of keys.

Detailed Description

Definition in file rtXmlKeyArray.c

rtXmlKeyArray.h File Reference

```
#include "rtxsrc/rtxContext.h"
```

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

Data Structures

- struct OSXSDKeyRecord
- struct OSXSDKeyArray

Enumerations

- enum OSXSDFieldDataType {
  OSXSDString,
  OSXSDDecimal,
  OSXSDInt,
  OSXSDUInt
  }

Functions

- EXTERNXML int rtXmlKeyArrayInit (OSCTXT * pctxt, OSXSDKeyArray * pArray, OSUINT32 nmFields, OSBOOL key, const char * name)

  Initialize the array by allocating memory for it.

- EXTERNXML void rtXmlKeyArrayFree (OSCTXT * pctxt, OSXSDKeyArray * pArray)

- EXTERNXML void rtXmlKeyArraySetString (OSXSDKeyArray * pArray, const OSUTF8CHAR * pValue, OSUINT32 fldNum)

  Set the given field's value to the given string value.

- EXTERNXML void rtXmlKeyArraySetInt (OSXSDKeyArray * pArray, OSINT32 value, OSUINT32 fldNum)

  Set the given field's value to the given integer value.

- EXTERNXML void rtXmlKeyArraySetUInt (OSXSDKeyArray * pArray, OSUINT32 value, OSUINT32 fldNum)

  Set the given field's value to the given unsigned integer value.

- EXTERNXML void rtXmlKeyArraySetDecimal (OSXSDKeyArray * pArray, const OSREAL * value, OSUINT32 fldNum)

  Set the given field's value to the given decimal value.

- EXTERNXML int rtXmlKeyArrayAdd (OSCTXT * pctxt, OSXSDKeyArray * pArray)

  Once all the fields for a key are set, invoke this method to add the key.

- EXTERNXML int rtXmlKeyArrayContains (OSCTXT * pctxt, OSXSDKeyArray * pArray)

  Once all the fields for a key are set, this method is used to check whether the key is already present in the array of keys.

Detailed Description

Implementation of a dynamic pointer sorted array.

Definition in file rtXmlKeyArray.h

rtXmlLibrary.hh File Reference

```
#include <dlfcn.h>
```

```
#include "osrtxml.h"
```

Macros

- #define LOADLIB dlopen (name, RTLD_NOW)
- #define GETLIBPROC ((type)dlsym (lib, func))
- #define CLOSELIB dlclose (lib)
- #define DEFAULT_SAX_LIBRARY "libexpat"

Typedefs

- typedef void * LIBHANDLE
- typedef void *(* GetXMLFactoryProc

Variables

- LIBHANDLE rtXmlLib
- GetXMLFactoryProc getFactory

Detailed Description

Definition in file rtXmlLibrary.hh

rtXmlLibxml2IF.c File Reference

```
#include <libxml/parser.h>
```

```
#include <libxml/SAX.h>
```

```
#include <libxml/parserInternals.h>
```

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

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

```
#include "rtxsrc/osMacros.h"
```

```
#include "rtxsrc/rtxCharStr.h"
```

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxStream.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Data Structures

- struct XMLReaderImpl

Macros

- #define XML_BUF_SIZE 2048

Typedefs

- typedef struct XMLReaderImpl XMLReaderImpl

Variables

- static OSBOOL g_manual_cleanup

Functions

- int rtSaxCReleaseReader (OSXMLREADER * pReader)
- int rtSaxCEnableThreadSafety ( )
- int rtSaxCParse (OSXMLREADER * pReader)
- static void setDocumentLocatorCallback (void * ctx, xmlSAXLocatorPtr loc)
- static void errorCallback (void * ctx, const char * msg, ...)

  error: @ctxt: An XML parser context @msg: the message to display/transmit @...: extra parameters for the message display
- static void cdataBlockCallback (void * userData, const xmlChar * value, int len)
- static const OSUTF8CHAR * parseQName (const OSUTF8CHAR * qname)
- static void startElementCallback (void * userData, const xmlChar * name, const xmlChar ** atts)
- static void endElementCallback (void * userData, const xmlChar * name)
- static void charDataCallback (void * userData, const xmlChar * s, int len)
- OSXMLREADER * rtSaxCCreateXmlReader (OSCTXT * pctxt, void * pSaxHandlerData, CSAX_StartElementHandler pStartElementProc, CSAX_EndElementHandler pEndElementProc, CSAX_CharacterDataHandler pCharactersProc)

Detailed Description

Definition in file rtXmlLibxml2IF.c

rtXmlLibxml2IF.cpp File Reference

```
#include <stdarg.h>
#include <libxml/parser.h>
#include <libxml/SAX.h>
#include <libxml/parserInternals.h>
#include "rtxmlsrc/rtSaxCppParserIF.h"
#include "rtxsrc/osMacros.h"
#include "rtxsrc/rtxCharStr.h"
#include "rtxsrc/rtxCtype.h"
#include "rtxsrc/rtxErrCodes.h"
#include "rtxsrc/OSRTInputStreamIF.h"
```

Data Structures

- struct OSLibxmlXMLReader

Macros

- #define XML_BUF_SIZE 2048

Functions

- OSXMLReaderClass * rtSaxCppCreateXmlReader (OSXMLParserCtxtIF * pContext, OSXMLDefaultHandlerIF * pSaxHandler)
- int rtSaxCppEnableThreadSafety ( )
- void rtSaxCppLockXmlLibrary ( )
- void rtSaxCppUnlockXmlLibrary ( )

Detailed Description

Definition in file rtXmlLibxml2IF.cpp

rtXmlMatchBase64Str.c File Reference

```
#include "rtxmlsrc/osrtxml.hh"
#include "rtxsrc/rtxCtype.h"
```

Functions

- EXTXMLMETHOD int rtXmlMatchBase64Str (OSCTXT * pctxt, size_t minLength, size_t maxLength)

Detailed Description

Definition in file rtXmlMatchBase64Str.c

rtXmlMatchDates.c File Reference

```
#include <string.h>
```

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

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlMatchGYear (OSCTXT * pctxt)

  rtXmlMatchGYear:This function matches a date value from a supplied context

- EXTXMLMETHOD int rtXmlMatchGYearMonth (OSCTXT * pctxt)

  rtXmlMatchGYearMonth: This function matches a time value from a supplied context.

- EXTXMLMETHOD int rtXmlMatchGMonth (OSCTXT * pctxt)

  rtXmlMatchGMonth: This function matches a datetime value from a supplied context.

- EXTXMLMETHOD int rtXmlMatchGMonthDay (OSCTXT * pctxt)

  rtXmlMatchGMonthDay: This function matches a datetime value from a supplied context.

- EXTXMLMETHOD int rtXmlMatchGDay (OSCTXT * pctxt)

  rtXmlMatchGDay: This function matches a datetime value from a supplied context.

Detailed Description

Definition in file rtXmlMatchDates.c

rtXmlMatchDateTime.c File Reference

```
#include <string.h>
```

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

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlMatchDate (OSCTXT * pctxt)

  rtXmlMatchDate:This function matches a date value from a supplied context.

- EXTXMLMETHOD int rtXmlMatchTime (OSCTXT * pctxt) *rtXmlMatchTime: This function matches a time value from a supplied context.*

- EXTXMLMETHOD int rtXmlMatchDateTime (OSCTXT * pctxt)

  rtXmlMatchDateTime: This function matches a datetime value from a supplied context.

Detailed Description

Definition in file rtXmlMatchDateTime.c

rtXmlMatchHexStr.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

Functions

- EXTXMLMETHOD int rtXmlMatchHexStr (OSCTXT * pctxt, size_t minLength, size_t maxLength)

Detailed Description

Definition in file rtXmlMatchHexStr.c

rtXmlMemFree.c File Reference

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

Functions

- EXTXMLMETHOD void rtXmlMemFreeAnyAttrs (OSCTXT * pctxt, OSRTDList * pAnyAttrList)

  This function frees a list of anyAttribute that is a member of OSXSDAnyType structure.

Detailed Description

Definition in file rtXmlMemFree.c

rtXmlMicroIF.c File Reference

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

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxStream.h"
```

```
#include "rtxsrc/rtxStreamBuffered.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Data Structures

- struct OSMemBlock
- struct XMLReaderImpl

Macros

- #define MAX_MEMBLOCK_SIZE 2048
- #define memReset (pMemBlk)->lastIdx = 0

Enumerations

- enum ErrorCodes {
  SAXERR_UNEXPECTED_CHAR= -1,
  SAXERR_INVALID_CHAR= -2,
  SAXERR_BADHEADER= -3
  }

Typedefs

- typedef struct XMLReaderImpl XMLReaderImpl

Variables

- static const OSOCTET transitions
- static XMLReaderImpl gReader

Functions

- static void * memAlloc (OSMemBlock * pMemBlk, OSSIZE size)
- static void rtSaxCStartElementHandler (XMLReaderImpl * reader, const OSUTF8CHAR * name, const OSUTF8CHAR * localname, const OSUTF8CHAR ** atts)
- static void rtSaxCEndElementHandler (XMLReaderImpl * reader, const OSUTF8CHAR * qname, const OSUTF8CHAR * localname)
- static void rtSaxCCharacterDataHandler (XMLReaderImpl * reader, const OSUTF8CHAR * s, OSSIZE len)
- static void rtSaxCStopParser (XMLReaderImpl * reader)
- EXTXMLMETHOD int rtSaxCReleaseReader (OSXMLREADER * pReader)
- static int getSymbolIndex (OSOCTET sym)
- EXTXMLMETHOD int rtSaxCEnableThreadSafety ( )
- EXTXMLMETHOD int rtSaxCParse (OSXMLREADER * pReader)
- EXTXMLMETHOD OSXMLREADER * rtSaxCCreateXmlReader (OSCTXT * pctxt, void * pSaxHandlerData, CSAX_StartElementHandler pStartElementProc, CSAX_EndElementHandler pEndElementProc, CSAX_CharacterDataHandler pCharactersProc)

Detailed Description

Definition in file rtXmlMicroIF.c

rtXmlNamespace.c File Reference

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

```
#include "rtxsrc/osMacros.h"
```

```
#include "rtxsrc/rtxCharStr.h"
```

```
#include "rtxsrc/rtxUTF8.h"
```

```
#include "rtxsrc/rtxDynPtrArray.h"
```

Functions

- static OSXMLNSPfxLink * getPrefixLinkUsingURI (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  Search through the prefix link stack (top to bottom) for a prefix link for the given URI.

- static const OSUTF8CHAR * getNonEmptyPrefix (OSXMLNSPfxLink * plink)

  Return a non-empty prefix from the given namespace-prefix link node.

- static OSBOOL prefixesEqual (const OSUTF8CHAR * prefix1, const OSUTF8CHAR * prefix2)

- EXTXMLMETHOD OSXMLNamespace * rtXmlNSAddNamespace (OSCTXT * pctxt, OSRTDList * pNSAttrs, const OSUTF8CHAR * prefix, const OSUTF8CHAR * uri)

  This function adds a namespace to the context namespace list.

- EXTXMLMETHOD OSBOOL rtXmlNSEqual (OSXMLNamespace * pNS1, OSXMLNamespace * pNS2)

  This function checks if two namespace records are equal.

- EXTXMLMETHOD void rtXmlNSFreeAttrList (OSCTXT * pctxt, OSRTDList * pNSAttrs)

  This function frees dynamic memory used to hold namespace attribute values.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSGetPrefix (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function gets a namespace prefix assigned to the given URI.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSGetPrefixUsingIndex (OSCTXT * pctxt, const OSUTF8CHAR * uri, OSUINT32 idx)

  This function gets a namespace prefix assigned to the given URI using the given index to select a specific prefix from the URI/prefix map.

- EXTXMLMETHOD OSUINT32 rtXmlNSGetPrefixCount (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function returns the total number of prefixes currently assigned to the given URI.

- EXTXMLMETHOD int rtXmlNSGetPrefixIndex (OSCTXT * pctxt, const OSUTF8CHAR * uri, const OSUTF8CHAR * prefix, OSUINT32 * pcount)

This function gets the index of a given prefix in the internal list of prefixes maintained for a given URI in teh namespace stack.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSLookupPrefixForURI (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function looks up a namespace in the context namespace stack using URI as the key value and returns a non-empty prefix, if one has been defined.

- EXTXMLMETHOD OSXMLNamespace * rtXmlNSLookupURI (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function looks up a namespace in the context namespace stack using URI as the key value.

- EXTXMLMETHOD OSXMLNamespace * rtXmlNSLookupURIInList (OSRTDList * pNSAttrs, const OSUTF8CHAR * uri)

  This function looks up a namespace in the given list using URI as the key value.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSLookupPrefix (OSCTXT * pctxt, const OSUTF8CHAR * prefix)

  This function looks up a namespace in the context namespace list using the prefix as the key value.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSLookupPrefixFrag (OSCTXT * pctxt, const OSUTF8CHAR * prefix, size_t prefixLen)

  This function looks up a namespace in the context namespace list using the prefix as the key value.

- EXTXMLMETHOD void rtXmlNSRemoveAll (OSCTXT * pctxt)

  This function removes all namespaces from the context namespace list and frees the dynamic memory used to hold the names.

- EXTXMLMETHOD OSXMLNamespace * rtXmlNSSetNamespace (OSCTXT * pctxt, OSRTDList * pNSAttrs, const OSUTF8CHAR * prefix, const OSUTF8CHAR * uri, OSBOOL override)

  This function sets a namespace in the context namespace list.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSNewPrefix (OSCTXT * pctxt, const OSUTF8CHAR * uri, OSRTDList * pNSAttrs)

  This function returns the next unused prefix of the form "nsX" where X is a number.

- EXTXMLMETHOD int rtXmlNSAddPrefixLink (OSCTXT * pctxt, const OSUTF8CHAR * prefix, const OSUTF8CHAR * uri, const OSUTF8CHAR * nsTable, OSUINT32 nsTableRowCount)

  Add a prefix link at the current stack level.

- EXTXMLMETHOD int rtXmlNSFreeAllPrefixLinks (OSCTXT * pctxt, OSXMLNSPfxLinkStackNode * pStackNode)

  Free all prefixs links in the given namespace stack entry.

- EXTXMLMETHOD int rtXmlNSFreePrefixLink (OSCTXT * pctxt, OSXMLNSPfxLink * plink)

  Free all data within a given namespace prefix link structure.

- EXTXMLMETHOD int rtXmlNSGetIndex (OSCTXT * pctxt, const OSUTF8CHAR * prefix)

  Get namespace index for a given namespace prefix based on current namespace stack in context.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSGetPrefixUsingNsIdx (OSCTXT * pctxt, OSINT32 nsidx)

- EXTXMLMETHOD int rtXmlNSPush (OSCTXT * pctxt)

  Push new namespace prefix mapping level onto stack.

- EXTXMLMETHOD int rtXmlNSPop (OSCTXT * pctxt)

  Remove top namespace prefix mapping level from stack.

- EXTXMLMETHOD void rtXmlNSSetURITable (OSCTXT * pctxt, const OSUTF8CHAR * data, OSUINT32 nrows)

  Set namespace URI table in context.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSGetQName (OSCTXT * pctxt, OSUTF8CHAR * buf, size_t bufsiz, const OSUTF8CHAR * uri, const OSUTF8CHAR * localName)

  This function creates a QName in the given fixed-site buffer.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSGetAttrPrefix (OSCTXT * pctxt, const OSUTF8CHAR * namespaceURI, OSRTDList * pNSAttrs)

  This function returns a namespace prefix for use with attributes in the given namespace.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSGetAttrQName (OSCTXT * pctxt, OSUTF8CHAR * buf, size_t bufsiz, OSXMLNamespace * pNS, const OSUTF8CHAR * localName, OSRTDList * pNSAttrs)

  This function creates a QName for a qualified attribute.

Detailed Description

Definition in file rtXmlNamespace.c

rtXmlNamespace.h File Reference

```
#include "rtxsrc/rtxContext.h"

#include "rtxsrc/rtxDynPtrArray.h"

#include "rtxsrc/rtxXmlQName.h"

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

Data Structures

- struct OSXMLNamespace
- struct OSXMLNSPfxLink
- struct OSXMLNSPfxLinkStackNode

- struct OSXMLNSPfxLinkStack
- struct OSXMLNSURITable

Macros

- #define OSXMLNSURI "http://www.w3.org/XML/1998/namespace"
- #define OSXSINSURI "http://www.w3.org/2001/XMLSchema-instance"
- #define OSXSINSURI_LEN 41
- #define OSXSDNSURI "http://www.w3.org/2001/XMLSchema"
- #define RTXMLNSSETQNAME if (0 != pNS) { qname.nsPrefix = pNS->prefix; qname.nsURI = pNS->uri; } \ else { qname.nsPrefix = qname.nsURI = 0; }

  This macro populates the given QName structure with information from the given namespace structure (namespace URI and prefix).

Typedefs

- typedef struct OSXMLNamespace OSXMLNamespace
- typedef struct OSXMLNSPfxLink OSXMLNSPfxLink
- typedef struct OSXMLNSPfxLinkStackNode OSXMLNSPfxLinkStackNode
- typedef struct OSXMLNSPfxLinkStack OSXMLNSPfxLinkStack
- typedef struct OSXMLNSURITable OSXMLNSURITable

Functions

- EXTERNXML OSXMLNamespace * rtXmlNSAddNamespace (OSCTXT * pctxt, OSRTDList * pNSAttrs, const OSUTF8CHAR * prefix, const OSUTF8CHAR * uri)

  This function adds a namespace to the context namespace list.

- EXTERNXML OSBOOL rtXmlNSEqual (OSXMLNamespace * pNS1, OSXMLNamespace * pNS2)

  This function checks if two namespace records are equal.

- EXTERNXML void rtXmlNSFreeAttrList (OSCTXT * pctxt, OSRTDList * pNSAttrs)

  This function frees dynamic memory used to hold namespace attribute values.

- EXTERNXML const OSUTF8CHAR * rtXmlNSGetPrefix (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function gets a namespace prefix assigned to the given URI.

- EXTERNXML const OSUTF8CHAR * rtXmlNSGetPrefixUsingIndex (OSCTXT * pctxt, const OSUTF8CHAR * uri, OSUINT32 idx)

  This function gets a namespace prefix assigned to the given URI using the given index to select a specific prefix from the URI/prefix map.

- EXTERNXML OSUINT32 rtXmlNSGetPrefixCount (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function returns the total number of prefixes currently assigned to the given URI.

- EXTERNXML int rtXmlNSGetPrefixIndex (OSCTXT * pctxt, const OSUTF8CHAR * uri, const OSUTF8CHAR * prefix, OSUINT32 * pcount)

  This function gets the index of a given prefix in the internal list of prefixes maintained for a given URI in teh namespace stack.

- EXTERNXML const OSUTF8CHAR * rtXmlNSGetQName (OSCTXT * pctxt, OSUTF8CHAR * buf, size_t bufsiz, const OSUTF8CHAR * uri, const OSUTF8CHAR * localName)

  This function creates a QName in the given fixed-site buffer.

- EXTXMLMETHOD const OSUTF8CHAR * rtXmlNSGetAttrPrefix (OSCTXT * pctxt, const OSUTF8CHAR * namespaceURI, OSRTDList * pNSAttrs)

  This function returns a namespace prefix for use with attributes in the given namespace.

- EXTERNXML const OSUTF8CHAR * rtXmlNSGetAttrQName (OSCTXT * pctxt, OSUTF8CHAR * buf, size_t bufsiz, OSXMLNamespace * pNS, const OSUTF8CHAR * localName, OSRTDList * pNSAttrs)

  This function creates a QName for a qualified attribute.

- EXTERNXML OSXMLNamespace * rtXmlNSLookupURI (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function looks up a namespace in the context namespace stack using URI as the key value.

- EXTERNXML OSXMLNamespace * rtXmlNSLookupURIInList (OSRTDList * pNSAttrs, const OSUTF8CHAR * uri)

  This function looks up a namespace in the given list using URI as the key value.

- EXTERNXML const OSUTF8CHAR * rtXmlNSLookupPrefixForURI (OSCTXT * pctxt, const OSUTF8CHAR * uri)

  This function looks up a namespace in the context namespace stack using URI as the key value and returns a non-empty prefix, if one has been defined.

- EXTERNXML const OSUTF8CHAR * rtXmlNSLookupPrefix (OSCTXT * pctxt, const OSUTF8CHAR * prefix)

  This function looks up a namespace in the context namespace list using the prefix as the key value.

- EXTERNXML const OSUTF8CHAR * rtXmlNSLookupPrefixFrag (OSCTXT * pctxt, const OSUTF8CHAR * prefix, size_t prefixLen)

  This function looks up a namespace in the context namespace list using the prefix as the key value.

- EXTERNXML void rtXmlNSRemoveAll (OSCTXT * pctxt)

  This function removes all namespaces from the context namespace list and frees the dynamic memory used to hold the names.

- EXTERNXML OSXMLNamespace * rtXmlNSSetNamespace (OSCTXT * pctxt, OSRTDList * pNSAttrs, const OSUTF8CHAR * prefix, const OSUTF8CHAR * uri, OSBOOL override)

  This function sets a namespace in the context namespace list.

- EXTERNXML const OSUTF8CHAR * rtXmlNSNewPrefix (OSCTXT * pctxt, const OSUTF8CHAR * uri, OSRTDList * pNSAttrs)

  This function returns the next unused prefix of the form "nsX" where X is a number.

- EXTERNXML int rtXmlNSAddPrefixLink (OSCTXT * pctxt, const OSUTF8CHAR * prefix, const OSUTF8CHAR * uri, const OSUTF8CHAR * nsTable, OSUINT32 nsTableRowCount)

  Add a prefix link at the current stack level.

- EXTERNXML int rtXmlNSFreeAllPrefixLinks (OSCTXT * pctxt, OSXMLNSPfxLinkStackNode * pStackNode)

  Free all prefixs links in the given namespace stack entry.

- EXTERNXML int rtXmlNSFreePrefixLink (OSCTXT * pctxt, OSXMLNSPfxLink * plink)

  Free all data within a given namespace prefix link structure.

- EXTERNXML int rtXmlNSGetIndex (OSCTXT * pctxt, const OSUTF8CHAR * prefix)

  Get namespace index for a given namespace prefix based on current namespace stack in context.

- EXTERNXML int rtXmlNSPush (OSCTXT * pctxt)

  Push new namespace prefix mapping level onto stack.

- EXTERNXML int rtXmlNSPop (OSCTXT * pctxt)

  Remove top namespace prefix mapping level from stack.

- EXTERNXML void rtXmlNSSetURITable (OSCTXT * pctxt, const OSUTF8CHAR * data, OSUINT32 nrows)

  Set namespace URI table in context.

Detailed Description

XML namespace handling structures and function definitions.

Definition in file rtXmlNamespace.h

rtXmlNewQName.c File Reference

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

Functions

- EXTXMLMETHOD OSUTF8CHAR * rtXmlNewQName (OSCTXT * pctxt, const OSUTF8CHAR * localName, const OSUTF8CHAR * prefix)

  This function creates a new QName given the localName and prefix parts.

Detailed Description

Definition in file rtXmlNewQName.c

rtXmlParseElementName.c File Reference

```
#include "rtxsrc/rtxCtype.h"
#include "rtxsrc/rtxStreamBuffered.h"
#include "rtxmlsrc/osrtxml.hh"
```

Functions

- EXTXMLMETHOD int rtXmlParseElementName (OSCTXT * pctxt, OSUTF8CHAR ** ppName)

  This function parses the initial tag from an XML message.

Detailed Description

Definition in file rtXmlParseElementName.c

rtXmlParseElemQName.c File Reference

```
#include "rtxsrc/rtxCtype.h"
#include "rtxsrc/rtxStreamBuffered.h"
#include "rtxmlsrc/osrtxml.hh"
```

Functions

- static int expandNameBuffer (OSCTXT * pctxt, OSRTBuffer * pNameBuf)
- EXTXMLMETHOD int rtXmlParseElemQName (OSCTXT * pctxt, OSXMLQName * pQName)

  This function parses the initial tag from an XML message.

Detailed Description

Definition in file rtXmlParseElemQName.c

rtXmlpCppDecFuncs.h File Reference

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

Data Structures

- struct OSXMLStringListParser

  Class enabling parsing of an XML Schema list into strings.

Functions

- EXTERNXML int rtXmlpCppDecStrList (OSCTXT * pctxt, OSRTObjListClass * plist)
- EXTERNXML int rtXmlpCppDecXmlStr (OSCTXT * pctxt, OSRTXMLString * outdata)
- EXTERNXML int rtXmlpCppGetXmlnsAttrs (OSCTXT * pctxt, OSXSDComplexType * ptype)

Detailed Description

XML low-level C++ decode functions.

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

Definition in file rtXmlpCppDecFuncs.h

rtXmlpCppDecStrList.cpp File Reference

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

#include "rtxmlsrc/rtXmlPull.h"

#include "rtxsrc/rtxCppDList.h"

#include "rtxsrc/rtxCppXmlString.h"

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

Functions

- int rtXmlpCppDecStrList (OSCTXT * pctxt, OSRTObjListClass * plist)

Detailed Description

Definition in file rtXmlpCppDecStrList.cpp

rtXmlpCppDecXmlQStr.cpp File Reference

```
#include <QString>

#include "rtxmlsrc/osrtxml.h"

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

Functions

- int rtXmlpCppDecXmlQStr (OSCTXT * pctxt, QString * outdata)

Detailed Description

Definition in file rtXmlpCppDecXmlQStr.cpp

rtXmlpCppDecXmlStr.cpp File Reference

```
#include "rtxmlsrc/osrtxml.h"
#include "rtxmlsrc/rtXmlPull.h"
#include "rtxmlsrc/rtXmlPull.hh"
#include "rtxsrc/rtxCppDList.h"
#include "rtxsrc/rtxCppXmlString.h"
#include "rtxmlsrc/rtXmlpCppDecFuncs.h"
```

Functions

- int rtXmlpCppDecXmlStr (OSCTXT * pctxt, OSRTXMLString * outdata)

Detailed Description

Definition in file rtXmlpCppDecXmlStr.cpp

rtXmlpCppGetXmlnsAttrs.cpp File Reference

```
#include "rtxmlsrc/osrtxml.h"
#include "rtxmlsrc/rtXmlPull.h"
#include "rtxsrc/rtxCppDList.h"
#include "rtxsrc/rtxCppXmlString.h"
#include "rtxmlsrc/rtXmlpCppDecFuncs.h"
```

Functions

- int rtXmlpCppGetXmlnsAttrs (OSCTXT * pctxt, OSXSDComplexType * ptype)

Detailed Description

Definition in file rtXmlpCppGetXmlnsAttrs.cpp

rtXmlpCreateReader.c File Reference

```
#include "rtxmlsrc/osrtxml.hh"
#include "rtxmlsrc/rtXmlPull.h"
#include "rtxsrc/rtxCtype.h"
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- static int createReader (OSCTXT * pctxt, OSXMLCtxtInfo * xmlCtxt)
- EXTXMLMETHOD int rtXmlpCreateReader (OSCTXT * pctxt)

  Creates pull parser reader structure within the context.
- EXTXMLMETHOD struct OSXMLReader * rtXmlpGetReader (OSCTXT * pctxt)

  This function fetches the XML reader structure from the context for use in low-level pull parser calls.
- EXTXMLMETHOD void rtXmlpFreeReader (OSCTXT * pctxt, OSXMLCtxtInfo * pXmlInfo)

Detailed Description

Definition in file rtXmlpCreateReader.c

rtXmlpDecAny.c File Reference

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

```
#include "rtxmlsrc/rtXmlPull.hh"
```

Functions

- EXTXMLMETHOD int rtXmlpDecAny2 (OSCTXT * pctxt, OSUTF8CHAR ** pvalue)

  This version of the rtXmlpDecAny function returns the string in a mutable buffer.
- EXTXMLMETHOD 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).

Detailed Description

Definition in file rtXmlpDecAny.c

rtXmlpDecAnyAttrStr.c File Reference

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

```
#include "rtxmlsrc/rtXmlPull.hh"
```

Functions

- EXTXMLMETHOD int rtXmlpDecAnyAttrStr (OSCTXT * pctxt, const OSUTF8CHAR ** ppAttrStr, size_t idx)

Detailed Description

Definition in file rtXmlpDecAnyAttrStr.c

rtXmlpDecAnyElem.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.hh"`

Functions

- EXTXMLMETHOD 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).

Detailed Description

Definition in file rtXmlpDecAnyElem.c

rtXmlpDecBase64Str.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

Macros

- #define BASE64TOINT (((c) < 128) ? decodeTable [(c) - 40] : -1)

Variables

- static const signed char decodeTable

Functions

- EXTXMLMETHOD int rtXmlpDecBase64Str (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnocts, OSSIZE bufsize)

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

- EXTXMLMETHOD int rtXmlpDecBase64Str64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlpDecBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

Detailed Description

Definition in file rtXmlpDecBase64Str.c

rtXmlpDecBigInt.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

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

```
#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define XMLP_DECODE_SEGSIZE 1

Functions

- EXTXMLMETHOD int rtXmlpDecBigInt (OSCTXT * pctxt, const OSUTF8CHAR ** pvalue)

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

Detailed Description

Definition in file rtXmlpDecBigInt.c

rtXmlpDecBitString.c File Reference

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

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

```
#include "rtxsrc/rtxBitString.h"
```

Functions

- EXTXMLMETHOD int rtXmlpDecBitString (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnbits, OSUINT32 bufsize)

  This function decodes a bit string value.

- EXTXMLMETHOD int rtXmlpDecBitString64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE bufsize)

  This function is identical to rtXmlpDecBitString except that it supports lengths up to 64-bits in size on 64-bit machines.

- EXTXMLMETHOD int rtXmlpDecBitStringExt (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnbits, OSOCTET ** ppextdata, OSUINT32 bufsize)

  This function decodes a bit string value.

- EXTXMLMETHOD int rtXmlpDecBitStringExt64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnbits, OSOCTET ** ppextdata, OSSIZE bufsize)

  This function is identical to rtXmlpDecBitStringExt except that it supports lengths up to 64-bits in size on 64-bit machines.

Detailed Description

Definition in file rtXmlpDecBitString.c

rtXmlpDecBool.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

Variables

- static const char trueVal
- static const char falseVal

Functions

- EXTXMLMETHOD int rtXmlpDecBool (OSCTXT * pctxt, OSBOOL * pvalue)

  This function decodes a variable of the boolean type.

Detailed Description

Definition in file rtXmlpDecBool.c

rtXmlpDecDateTime.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

Enumerations

- enum DecDateFrom {
  Years,
  Months,
  Days,
  TimeStart,
  TimeEnd= 8
  }

Variables

- static const int multiplier
- static const int tzh_multiplier
- static const int tzm_multiplier

Functions

- static int rtXmlDecDateTimeInner (OSCTXT * pctxt, OSXSDDateTime * pvalue, int decDateFrom, int decDateTo)
- EXTXMLMETHOD int rtXmlpDecDate (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecTime (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecDateTime (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecGYear (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecGYearMonth (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecGMonth (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecGMonthDay (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecGDay (OSCTXT * pctxt, OSXSDDateTime * pvalue)

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

Detailed Description

Definition in file rtXmlpDecDateTime.c

rtXmlpDecDecimal.c File Reference

```
#include <math.h>
```

```
#include <limits.h>
```

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

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

Functions

- EXTXMLMETHOD int rtXmlpDecDecimal (OSCTXT * pctxt, OSREAL * pvalue, int totalDigits, int fractionDigits)

  This function decodes the contents of a decimal data type.

Detailed Description

Definition in file rtXmlpDecDecimal.c

rtXmlpDecDouble.c File Reference

```
#include <math.h>
```

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

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

Variables

- static const char nanStr
- static const char infStr

Functions

- EXTXMLMETHOD int rtXmlpDecDouble (OSCTXT * pctxt, OSREAL * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecDoubleExt (OSCTXT * pctxt, OSUINT8 flags, OSREAL * pvalue)

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

Detailed Description

Definition in file rtXmlpDecDouble.c

rtXmlpDecDynBase64Str.c File Reference

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

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

```
#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define XMLP_DECODE_SEGSIZE 1
- #define BASE64TOINT (((c) < 128) ? decodeTable [(c) - 40] : -1)

Variables

- static const signed char decodeTable

Functions

- EXTXMLMETHOD int rtXmlpDecDynBase64Str (OSCTXT * pctxt, OSDynOctStr * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecDynBase64Str64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to rtXmlpDecDynBase64Str except that it supports 64-bit integer lengths on 64-bit systems.

Detailed Description

Definition in file rtXmlpDecDynBase64Str.c

rtXmlpDecDynBitString.c File Reference

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

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

```
#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define XMLP_DECODE_SEGSIZE 1

Functions

- EXTXMLMETHOD int rtXmlpDecDynBitString (OSCTXT * pctxt, OSDynOctStr * pvalue)

  This function decodes a bit string value.

Detailed Description

Definition in file rtXmlpDecDynBitString.c

rtXmlpDecDynHexStr.c File Reference

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

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

```
#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define XMLP_DECODE_SEGSIZE 1

Functions

- EXTXMLMETHOD int rtXmlpDecDynHexStr64 (OSCTXT * pctxt, OSDynOctStr64 * pvalue)

  This function is identical to the rtXmlpDecDynHexStr except that it supports lengths up to 64 bits in size on 64-bit systems.

- EXTXMLMETHOD int rtXmlpDecDynHexStr (OSCTXT * pctxt, OSDynOctStr * pvalue)

  This function decodes a contents of a hexBinary string.

Detailed Description

Definition in file rtXmlpDecDynHexStr.c

rtXmlpDecHexStr.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

Functions

- EXTXMLMETHOD int rtXmlpDecHexStr64 (OSCTXT * pctxt, OSOCTET * pvalue, OSSIZE * pnocts, OSSIZE bufsize)

  This function is identical to rtXmlpDecHexStr except that it supports lengths up to 64-bits in size on 64-bit machines.

- EXTXMLMETHOD int rtXmlpDecHexStr (OSCTXT * pctxt, OSOCTET * pvalue, OSUINT32 * pnocts, OSSIZE bufsize)

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

Detailed Description

Definition in file rtXmlpDecHexStr.c

rtXmlpDecInt.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

Variables

- static const OSINT32 maxInt32
- static const OSINT32 maxInt32_10
- static const OSINT32 maxInt16
- static const OSINT32 maxInt16_10
- static const OSINT32 maxInt8
- static const OSINT32 maxInt8_10

Functions

- static int rtXmlpDecIntLim (OSCTXT * pctxt, OSINT32 * pvalue, OSINT32 maxval, OSINT32 maxval10)
- EXTXMLMETHOD int rtXmlpDecInt (OSCTXT * pctxt, OSINT32 * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecInt8 (OSCTXT * pctxt, OSINT8 * pvalue)

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

- EXTXMLMETHOD int rtXmlpDecInt16 (OSCTXT * pctxt, OSINT16 * pvalue)

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

Detailed Description

Definition in file rtXmlpDecInt.c

rtXmlpDecInt64.c File Reference

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

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

Variables

- static const OSINT64 maxInt64
- static const OSINT64 maxInt64_10

Functions

- EXTXMLMETHOD int rtXmlpDecInt64 (OSCTXT * pctxt, OSINT64 * pvalue)

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

Detailed Description

Definition in file rtXmlpDecInt64.c

rtXmlpDecNamedBits.c File Reference

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

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

```
#include "rtxsrc/rtxBitString.h"
```

```
#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define BUFF_SZ 32

Functions

- static OSUINT32 FindBitMapItem (const OSBitMapItem * pBitMap, const OSOCTET * str, size_t strSz)

- EXTXMLMETHOD int rtXmlpDecNamedBits (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSOCTET * pvalue, OSUINT32 * pnbits, OSUINT32 bufsize)

  This function decodes the contents of a named bit field.

- EXTXMLMETHOD int rtXmlpDecNamedBits64 (OSCTXT * pctxt, const OSBitMapItem * pBitMap, OSOCTET * pvalue, OSSIZE * pnbits, OSSIZE bufsize)

  This function decodes the contents of a named bit field.

Detailed Description

Definition in file rtXmlpDecNamedBits.c

rtXmlpDecStrList.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

`#include "rtxsrc/rtxMemBuf.h"`

Functions

- static int rtXmlpDecStrList_ (OSCTXT * pctxt, OSRTDList * plist, OSBOOL xmlStr)
- EXTXMLMETHOD 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.

- EXTXMLMETHOD 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.

Detailed Description

Definition in file rtXmlpDecStrList.c

rtXmlpDecUInt.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

Variables

- static const OSUINT32 maxUInt32
- static const OSUINT32 maxUInt32_10
- static const OSUINT32 maxUInt16
- static const OSUINT32 maxUInt16_10

- static const OSUINT32 maxUInt8
- static const OSUINT32 maxUInt8_10

Functions

- static int rtXmlpDecUIntLim (OSCTXT * pctxt, OSUINT32 * pvalue, OSUINT32 maxval, OSUINT32 maxval10)
- EXTXMLMETHOD int rtXmlpDecUInt (OSCTXT * pctxt, OSUINT32 * pvalue)

  This function decodes the contents of an unsigned 32-bit integer data type.
- EXTXMLMETHOD int rtXmlpDecUInt8 (OSCTXT * pctxt, OSOCTET * pvalue)

  This function decodes the contents of an unsigned 8-bit integer data type (i.e.
- EXTXMLMETHOD int rtXmlpDecUInt16 (OSCTXT * pctxt, OSUINT16 * pvalue)

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

Detailed Description

Definition in file rtXmlpDecUInt.c

rtXmlpDecUInt64.c File Reference

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

Variables

- static const OSUINT64 maxUInt64
- static const OSUINT64 maxUInt64_10

Functions

- EXTXMLMETHOD int rtXmlpDecUInt64 (OSCTXT * pctxt, OSUINT64 * pvalue)

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

Detailed Description

Definition in file rtXmlpDecUInt64.c

rtXmlpDecUnicode.c File Reference

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

```
#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define XMLP_DECODE_SEGSIZE 1

Functions

- EXTXMLMETHOD int rtXmlpDecDynUnicodeStr (OSCTXT * pctxt, const OSUNICHAR ** ppdata, OSSIZE * pnchars)

  This function decodes a Unicode string data type.

Detailed Description

Definition in file rtXmlpDecUnicode.c

rtXmlpDecXmlStr.c File Reference

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

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

```
#include "rtxmlsrc/rtXmlPull.hh"
```

```
#include "rtxsrc/rtxMemBuf.h"
```

Macros

- #define XMLP_DECODE_SEGSIZE 1

Functions

- EXTXMLMETHOD int rtXmlpDecUTF8Str (OSCTXT * pctxt, OSUTF8CHAR * out, size_t max_len)
- EXTXMLMETHOD int rtXmlpDecXmlStr (OSCTXT * pctxt, OSXMLSTRING * outdata)

  This function decodes the contents of an XML string data type.
- EXTXMLMETHOD int rtXmlpDecDynUTF8Str (OSCTXT * pctxt, const OSUTF8CHAR ** outdata)

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

Detailed Description

Definition in file rtXmlpDecXmlStr.c

rtXmlpDecXSIAttr.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

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

Functions

- EXTXMLMETHOD int rtXmlpDecXSIAttr (OSCTXT * pctxt, const OSXMLNameFragments * attrName)

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

- EXTXMLMETHOD int rtXmlpGetXmlnsAttrs (OSCTXT * pctxt, OSRTDList * pNSAttrs)

  This function decodes namespace attributes from start tag and adds them to the given list.

- EXTXMLMETHOD int rtXmlpDecXSIAttrs (OSCTXT * pctxt)

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

Detailed Description

Definition in file rtXmlpDecXSIAttr.c

rtXmlpDecXSITypeAttr.c File Reference

```
#include "rtxsrc/rtxErrCodes.h"
```

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

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

Functions

- EXTXMLMETHOD 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).

- EXTXMLMETHOD int rtXmlpGetXSITypeAttr (OSCTXT * pctxt, const OSUTF8CHAR ** ppAttrValue, OSINT16 * nsidx, size_t * pLocalOffs)

- EXTXMLMETHOD int rtXmlpLookupXSITypeIndex (OSCTXT * pctxt, const OSUTF8CHAR * xsiType, OSINT16 xsiTypeIdx, const OSXMLItemDescr typetab, size_t typetabsiz)

- EXTXMLMETHOD int rtXmlpGetXSITypeIndex (OSCTXT * pctxt, const OSXMLItemDescr typetab, size_t typetabsiz)

Detailed Description

Definition in file rtXmlpDecXSITypeAttr.c

rtXmlpEvent.c File Reference

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

```
#include "rtxmlsrc/rtXmlPull.hh"
```

Functions

- EXTXMLMETHOD int rtXmlpMarkLastEventActive (OSCTXT * pctxt)

  This function marks current tag as unprocessed.

- EXTXMLMETHOD OSBOOL rtXmlpIsLastEventDone (OSCTXT * pctxt)

  Check processing status of current tag.

Detailed Description

Definition in file rtXmlpEvent.c

rtXmlpGetAttributeCount.c File Reference

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

#include "rtxmlsrc/rtXmlPull.hh"

#include "rtxsrc/rtxCtype.h"

#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlpGetAttributeCount (OSCTXT * pctxt)

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

Detailed Description

Definition in file rtXmlpGetAttributeCount.c

rtXmlpGetAttributeID.c File Reference

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

#include "rtxsrc/rtxCtype.h"

#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlpGetAttributeID (const OSXMLStrFragment * attrName, OSINT16 nsidx, size_t nAttr, const OSXMLAttrDescr attrNames, OSUINT32 attrPresent)

Detailed Description

Definition in file rtXmlpGetAttributeID.c

rtXmlpGetContent.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlPull.h"`

Functions

- EXTXMLMETHOD void rtXmlpGetContent (OSCTXT * pctxt, int level)

Detailed Description

Definition in file rtXmlpGetContent.c

rtXmlpGetNextAllElemID.c File Reference

Functions

- EXTXMLMETHOD int _rtXmlpGetNextAllElemID_NAME (OSCTXT * pctxt, const OSXMLElemIDRec * tab, size_t nrows, const _rtXmlpGetNextAllElemID_TYPE * pOrder, OSSIZE nOrder, OSSIZE maxOrder, int anyID)

Detailed Description

Definition in file rtXmlpGetNextAllElemID.c

rtXmlpGetNextElem.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxmlsrc/rtXmlPull.hh"`

`#include "rtxmlsrc/rtXmlErrCodes.h"`

Functions

- EXTXMLMETHOD int rtXmlpGetNextElem (OSCTXT * pctxt, OSXMLElemDescr * pElem, OSINT32 level)

  This function parse the next element start tag.

Detailed Description

Definition in file rtXmlpGetNextElem.c

rtXmlpGetNextElemID.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxmlsrc/rtXmlPull.hh"`

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

```
#include "rtXmlpGetNextAllElemID.c"
```

Macros

- #define _rtXmlpGetNextAllElemID_NAME rtXmlpGetNextAllElemID
- #define _rtXmlpGetNextAllElemID_TYPE OSUINT8
- #define _rtXmlpGetNextAllElemID_NAME rtXmlpGetNextAllElemID16
- #define _rtXmlpGetNextAllElemID_TYPE OSUINT16
- #define _rtXmlpGetNextAllElemID_NAME rtXmlpGetNextAllElemID32
- #define _rtXmlpGetNextAllElemID_TYPE OSUINT32

Functions

- EXTXMLMETHOD int rtXmlpGetNextElemID (OSCTXT * pctxt, const OSXMLElemIDRec * tab, size_t nrows, OSINT32 level, OSBOOL continueParse)
- EXTXMLMETHOD int rtXmlpGetNextSeqElemID (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXMLGroupDesc * pGroup, int curID, int lastMandatoryID, OSBOOL groupMode)

  This function parses the next start tag and finds index of element name in descriptor table.
- EXTXMLMETHOD int rtXmlpGetNextSeqElemID2 (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXMLGroupDesc * pGroup, int groups, int curID, int lastMandatoryID, OSBOOL groupMode, OSBOOL check-Repeat)

  This function parses the next start tag and finds index of element name in descriptor table.
- EXTXMLMETHOD int rtXmlpGetNextSeqElemIDExt (OSCTXT * pctxt, const OSXMLElemIDRec * tab, const OSXMLGroupDesc * ppGroup, const OSBOOL * extRequired, int postExtRootID, int curID, int lastMandatoryID, OSBOOL groupMode)

  This is an ASN.1 extension-supporting version of rtXmlpGetNextSeqElemID.
- EXTXMLMETHOD void rtXmlpForceDecodeAsGroup (OSCTXT * pctxt)

  Disable skipping of unknown elements in optional sequence tail.
- EXTXMLMETHOD OSBOOL rtXmlpIsDecodeAsGroup (OSCTXT * pctxt)

  This function checks if "decode as group" mode was forced.

Detailed Description

Definition in file rtXmlpGetNextElemID.c

rtXmlpHasAttributes.c File Reference

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

```
#include "rtxmlsrc/rtXmlPull.hh"
```

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD OSBOOL rtXmlpHasAttributes (OSCTXT * pctxt)

  This function checks accessibility of attributes.
- EXTXMLMETHOD void rtXmlpHideAttributes (OSCTXT * pctxt)

  Disable access to attributes.
- EXTXMLMETHOD OSBOOL rtXmlpNeedDecodeAttributes (OSCTXT * pctxt)

  This function checks if attributes were previously decoded.

Detailed Description

Definition in file rtXmlpHasAttributes.c

rtXmlpIsInGroup.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD OSBOOL rtXmlpIsInGroup (int elemID, int grpId, const OSBOOL * grpTab, int nElems)

Detailed Description

Definition in file rtXmlpIsInGroup.c

rtXmlpIsUTF8Encoding.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

Functions

- OSBOOL rtXmlpIsUTF8Encoding (OSCTXT * pctxt)

  This function checks if the encoding specified in XML header is UTF-8.

Detailed Description

Definition in file rtXmlpIsUTF8Encoding.c

rtXmlpMatch.c File Reference

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

```
#include "rtxmlsrc/rtXmlPull.hh"
```

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

Functions

- EXTXMLMETHOD int rtXmlpMatchStartTag (OSCTXT * pctxt, const OSUTF8CHAR * elemLocalName, OSINT16 nsidx)

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

- EXTXMLMETHOD int rtXmlpMatchEndTag (OSCTXT * pctxt, OSINT32 level)

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

- EXTXMLMETHOD OSBOOL rtXmlpMatchElemId (OSCTXT * pctxt, int elemID, int matchingID)
- EXTXMLMETHOD OSINT32 rtXmlpGetCurrentLevel (OSCTXT * pctxt)

  This function returns current nesting level.

- EXTXMLMETHOD void rtXmlpSetWhiteSpaceMode (OSCTXT * pctxt, OSXMLWhiteSpaceMode whiteSpace-Mode)

  Sets the whitespace treatment mode.

- EXTXMLMETHOD OSBOOL rtXmlpSetMixedContentMode (OSCTXT * pctxt, OSBOOL mixedContentMode)

  Sets mixed content mode.

- EXTXMLMETHOD OSBOOL rtXmlpIsContentMode (OSCTXT * pctxt)
- EXTXMLMETHOD void rtXmlpSetListMode (OSCTXT * pctxt)

  Sets list mode.

- EXTXMLMETHOD OSBOOL rtXmlpListHasItem (OSCTXT * pctxt)

  Check for end of decoded token list.

- EXTXMLMETHOD void rtXmlpCountListItems (OSCTXT * pctxt, OSSIZE * itemCnt)

  Count tokens in list.

- void rtXmlpSetNamespaceTable (OSCTXT * pctxt, const char ** namespaceTable, size_t nmNamespaces)
- EXTXMLMETHOD void rtXmlpMarkPos (OSCTXT * pctxt)

  Save current decode position.

- EXTXMLMETHOD void rtXmlpRewindToMarkedPos (OSCTXT * pctxt)

  Rewind to saved decode position.

- EXTXMLMETHOD void rtXmlpResetMarkedPos (OSCTXT * pctxt)

  Reset saved decode position.

- EXTXMLMETHOD OSBOOL rtXmlpIsEmptyElement (OSCTXT * pctxt)

  Check element content: empty or not.

- EXTXMLMETHOD int rtXmlDecEmptyElement (OSCTXT * pctxt)

  This function is used to enforce a requirement that an element be empty.

Detailed Description

Definition in file rtXmlpMatch.c

rtXmlpReadBytes.c File Reference

`#include "rtxmlsrc/osrtxml.hh"`

`#include "rtxmlsrc/rtXmlPull.h"`

Functions

- int rtXmlRdPreReadFromStream (OSXMLReader * pReader)

  Reads at least 1 byte from pReader's stream.

- EXTXMLMETHOD int rtXmlpReadBytes (OSCTXT * pctxt, OSOCTET * pbuf, size_t nbytes)

Detailed Description

Definition in file rtXmlpReadBytes.c

rtXmlPrintNSAttrs.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxmlsrc/rtXmlNamespace.h"`

Functions

- EXTXMLMETHOD int rtXmlPrintNSAttrs (const char * name, const OSRTDList * data)

  This function prints a list of namespace attributes.

Detailed Description

Definition in file rtXmlPrintNSAttrs.c

rtXmlpSelectAttribute.c File Reference

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

#include "rtxmlsrc/rtXmlPull.hh"

#include "rtxsrc/rtxCtype.h"

#include "rtxsrc/rtxErrCodes.h"
```

Functions

- EXTXMLMETHOD int rtXmlpSelectAttribute (OSCTXT * pctxt, OSXMLNameFragments * pAttr, OSINT16 * nsidx, size_t idx)

Detailed Description

Definition in file rtXmlpSelectAttribute.c

rtXmlPull.c File Reference

```
#include <stdio.h>

#include "rtxsrc/rtxContext.hh"

#include "rtxsrc/rtxCtype.h"

#include "rtxsrc/rtxMemory.h"

#include "rtxsrc/rtxUnicode.h"

#include "rtxmlsrc/osrtxml.hh"

#include "rtxmlsrc/rtXmlPull.hh"
```

Macros

- #define INVSYM_SZ 8 /* set string size for INVSYMBOL error */
- #define OSXMLCONST_ATTRIBUTES_DELTA 10 /* default delta for mpAttributes */
- #define OSXMLCONST_STACK_DELTA 10 /* default delta for mpAttributes */
- #define OSXML_ESCAPE_CHAR_BUF_SIZE 10
- #define rtXmlRdLogError pReader->mError=stat, LOG_RTERRNEW (pReader->mpCtxt, stat)
- #define rtXmlRdLogInvSymbolError rtXmlRdLogInvSymbol (pReader), LOG_RTERRNEW (pReader->mpCtxt, XML_E_INVSYMBOL)
- #define rtXmlRdIsError (pReader->mError != 0)
- #define rtXmlRdStrEqual (((a)->length == (b)->length) ? \ rtxUTF8StrnEqual((a)->value, (b)->value, (a)->length) : FALSE)

- #define OSXMLSYM_LT 0
- #define OSXMLSYM_QUE 1
- #define OSXMLSYM_EXCL 2
- #define OSXMLSYM_GT 3
- #define OSXMLSYM_SLASH 4
- #define OSXMLSYM_COLON 8
- #define OSXMLSYM_DASH 9
- #define OSXMLSYM_EQUAL 10
- #define OSXMLSYM_QUOT 11
- #define OSXMLSYM_ALPHA 5
- #define OSXMLSYM_DIGIT 6
- #define OSXMLSYM_SPACE 7
- #define OSXMLSYM_AMP 12
- #define OSXMLSYM_SEMIC 13
- #define OSXMLSYM_PERIOD 14
- #define OSXMLSYM_USCORE 15
- #define OSXMLSYM_APOS 16
- #define OSXMLSYM_OTHER 17
- #define OSXML_BUFSIZE 50
- #define rtXmlRdMarkPosition pReader->mMarkedPos = pReader->mByteIndex
- #define rtXmlRdGetOffset ((pReader->mMarkedPos != (size_t)-1)? \ pReader->mByteIndex - pReader->mMarkedPos : (size_t)-1)
- #define rtXmlRdGetOffsetPtr ((pReader->mMarkedPos != (size_t)-1)? \ pReader->mpBuffer + pReader->mMarkedPos + offset : 0)
- #define rtXmlRdCancelMark pReader->mMarkedPos = (size_t)-1;
- #define rtXmlRdIsMarked (pReader->mMarkedPos != (size_t)-1 && pReader->mMarkedPos < pReader->mReadSize)
- #define READ_SYMBOL getSymbol(pReader,FALSE)
- #define PEEK_SYMBOL getSymbol(pReader,TRUE)
- #define NEED_PREREAD (pReader->mByteIndex >= pReader->mReadSize)
- #define PREFIX_LEN 12

Enumerations

- enum OSXMLDelayedTaskId {
  OSXMLDT_START_ELEMENT_CLEANUP= 1,
  OSXMLDT_END_ELEMENT_EVENT= 2,
  OSXMLDT_END_ELEMENT_CLEANUP= 3
  }

Variables

- static const char *const xsiUrn
- static const char *const soapEnvUrn
- static const char *const soap12EnvUrn
- static const char *const xsdUrn
- static const size_t xsiUrnLen
- static const size_t soapEnvUrnLen
- static const size_t soap12EnvUrnLen
- static const size_t xsdUrnLen
- static const OSINT8 initial_transitions
- static const OSINT8 stdoc_transitions
- static const OSINT8 main_transitions

Functions

- static void rtXmlRdLogInvSymbol (OSXMLReader * pReader)
- static const OSUTF8CHAR * rtXmlRdGetPtr (const OSXMLReader * pReader, size_t offset, size_t * pGuaranteedSize)

  Get pointer into the given reader's buffer, to the marked position, plus the given offset from there.

- static OSBOOL transformEscapeChar (const OSUTF8CHAR * pEscCharPtr, size_t escCharLen, OSXMLStrFragment * pDestStr, OSUTF8CHAR * pBuf)
- static void rtXmlRdStackInit (OSXMLReader * pReader, OSXMLStack * pStack, size_t unitSize)
- static OSBOOL rtXmlRdStackEnsureCapacity (OSXMLReader * pReader, OSXMLStack * pStack, size_t reqUnits)
- static OSBOOL rtXmlRdStackReinit (OSXMLReader * pReader, OSXMLStack * pStack)
- static void * rtXmlRdStackPushNew (OSXMLReader * pReader, OSXMLStack * pStack)
- static void * rtXmlRdStackPop (OSXMLReader * pReader, OSXMLStack * pStack)
- static void * rtXmlRdStackGetTop (OSXMLReader * pReader, OSXMLStack * pStack)

- static const void * rtXmlRdStackGetByIndex (const OSXMLReader * pReader, const OSXMLStack * pStack, size_t i)
- static void rtXmlRdAddDelayedTask (OSXMLReader * pReader, OSXMLDelayedTaskId taskId)
- static void rtXmlRdPopNamespaces (OSXMLReader * pReader)
- static int endElement (OSXMLReader * pReader)
- static int rtXmlRdProcessDelayedTasks (OSXMLReader * pReader)
- EXTXMLMETHOD int rtXmlRdSelectAttribute (struct OSXMLReader * pReader, size_t index)

  Select attribute by index to read the attribute value.
- static int rtXmlRdProcessAttrValue (OSXMLReader * pReader, OSXMLDataCtxt * pDataCtxt)
- static const OSUTF8CHAR * rtXmlRdDupStr (OSXMLReader * pReader, OSXMLStrFragment * pDestStr, const OSUTF8CHAR * pstr)
- static void rtXmlRdFreeStr (OSXMLReader * pReader, OSXMLStrFragment * pDestStr)
- static void rtXmlRdPushNamespace (OSXMLReader * pReader, const OSUTF8CHAR * prefix, size_t prefixLen, const OSUTF8CHAR * urn, size_t urnLen)
- static int rtXmlRdGetNamespaceIndex (OSXMLReader * pReader, const OSUTF8CHAR * prefix, size_t prefixLen)
- static void rtXmlRdProcessStartElement (OSXMLReader * pReader)
- static int getSymbolIndex (OSXMLReader * pReader, int sym)

  Get the symbol index for the given symbol.
- static int getCDATASymbolIndex (OSXMLReader * pReader, int sym)
- EXTXMLMETHOD OSXMLEvent rtXmlRdGetLastEvent (struct OSXMLReader * pReader)

  Returns information on the last processed event (ID and level).
- static void rtXmlRdUpdateSrcPos (OSXMLSrcPos * pSrcPos, const OSUTF8CHAR * p, size_t sz)
- static size_t latin1ToUTF8 (const OSOCTET * inbuf, size_t inlen, OSOCTET * outbuf)
- int rtXmlRdPreReadFromStream (OSXMLReader * pReader)

  Reads at least 1 byte from pReader's stream.
- static int getSymbol (OSXMLReader * pReader, OSBOOL peek)
- static int readLoop (OSXMLReader * pReader, int localStateIdx, const OSINT8 transitions, OSBOOL getAll)
- static OSUTF8CHAR * getEncodingStr (struct OSXMLReader * pReader)
- static int processHeader (OSXMLReader * pReader)
- static int processPI (OSXMLReader * pReader)
- static int processComments (OSXMLReader * pReader)

- static int processStartElement (OSXMLReader * pReader)
- static int processEndElement (OSXMLReader * pReader)
- static int processCharacters (OSXMLReader * pReader)

  processCharacters is called repeatedly to collect the text content of an element.
- static int processCDATA (OSXMLReader * pReader)
- static int processDTD (OSXMLReader * pReader)
- static int checkForUTF16BOM (struct OSXMLReader *)
- int rtXmlRdNext (struct OSXMLReader * pReader)

  The most low-level call.
- EXTXMLMETHOD int rtXmlRdSkipCurrentLevel (struct OSXMLReader * pReader)

  Skip the current level of XML document.
- EXTXMLMETHOD long rtXmlRdFirstData (struct OSXMLReader * pReader, OSXMLDataCtxt * pDataCtxt)

  Reads the first data chunk from content or attribute value.
- EXTXMLMETHOD long rtXmlRdNextData (struct OSXMLReader * pReader, OSXMLDataCtxt * pDataCtxt)

  Reads the next data chunk from content or attribute value.
- EXTXMLMETHOD OSXMLWhiteSpaceMode rtXmlRdSetWhiteSpaceMode (struct OSXMLReader * pReader, OSXMLWhiteSpaceMode whiteSpaceMode)

  Sets the whitespace treatment mode.
- static void initReaderVars (OSXMLReader * pReader)
- static void initReaderStream (OSXMLReader * pReader)
- EXTXMLMETHOD OSXMLReader * rtXmlRdCreateXmlReader (OSCTXT * pctxt)

  Creates XML reader.
- EXTXMLMETHOD void rtXmlRdResetXmlReader (OSXMLReader * pReader)

  Reset XML reader.
- EXTXMLMETHOD int rtXmlRdGetTagName (struct OSXMLReader * pReader, OSXMLStrFragment * localName, OSINT16 * namespaceIndex)

  Returns the local name and namespace URI of the current element.
- EXTXMLMETHOD int rtXmlRdNextTag (struct OSXMLReader * pReader)

  Reads until the next start tag is found.
- EXTXMLMETHOD int rtXmlRdNextEndTag (struct OSXMLReader * pReader)

  Reads until the next end tag is found.

- EXTXMLMETHOD int rtXmlRdNextEvent (struct OSXMLReader * pReader, OSUINT32 eventMask, OSINT32 maxLevel, OSXMLEvent * pLastEvent)

  This function reads XML until one of the events from eventMask happens.

- EXTXMLMETHOD int rtXmlRdMarkLastEventDone (struct OSXMLReader * pReader)

  Marks the last event as done.

- EXTXMLMETHOD int rtXmlRdMarkLastEventActive (struct OSXMLReader * pReader)

  Marks the last event as active.

- EXTXMLMETHOD int rtXmlRdGetAttributeCount (struct OSXMLReader * pReader)

  Returns the number of attributes defined in the current element.

- EXTXMLMETHOD int rtXmlRdGetAttributeName (struct OSXMLReader * pReader, OSXMLNameFragments * pAttr, OSINT16 * pNsidx, size_t index)

  Gets the attribute name by index.

- EXTXMLMETHOD int rtXmlRdFirstAttr (struct OSXMLReader * pReader, OSXMLStrFragment * pAttrName, OSXMLStrFragment * pAttrValue)

  Gets the first attribute name and value.

- EXTXMLMETHOD int rtXmlRdNextAttr (struct OSXMLReader * pReader, OSXMLStrFragment * pAttrName, OSXMLStrFragment * pAttrValue)

  Gets the second attribute name and value.

- EXTXMLMETHOD OSINT32 rtXmlRdGetCurrentLevel (struct OSXMLReader * pReader)

  Returns the current nesting level of where the pull parser currently is within the XML document tree.

- EXTXMLMETHOD OSXMLDataMode rtXmlRdGetDataMode (struct OSXMLReader * pReader)

  Returns the current data mode.

- EXTXMLMETHOD const OSXMLAttrOffset * rtXmlRdGetAttribute (OSXMLReader * pReader, size_t idx)

- EXTXMLMETHOD void rtXmlRdMarkPos (OSXMLReader * pReader)

- EXTXMLMETHOD void rtXmlRdRewindToMarkedPos (OSXMLReader * pReader)

- EXTXMLMETHOD void rtXmlRdResetMarkedPos (OSXMLReader * pReader)

- EXTXMLMETHOD int rtXmlRdGetXSITypeAttr (OSXMLReader * pReader, const OSUTF8CHAR ** ppAttr-Value, OSINT16 * nsidx, size_t * pLocalOffs)

- EXTXMLMETHOD OSBOOL rtXmlRdIsEmpty (struct OSXMLReader * pReader)

  Return TRUE when next will be end tag ("</").

- EXTXMLMETHOD void rtXmlRdGetSourcePosition (struct OSXMLReader * pReader, OSUINT32 * pLine, OSUINT32 * pColumn, OSUINT32 * pByteIndex, OSBOOL nextPos)

- static void rtXmlRdErrAddPos (struct OSXMLReader * pReader, const OSXMLSrcPos * pos)

- EXTXMLMETHOD void rtXmlRdErrAddSrcPos (struct OSXMLReader * pReader, OSBOOL nextPos)
- EXTXMLMETHOD void rtXmlRdErrAddDataSrcPos (struct OSXMLReader * pReader, OSXMLDataCtxt * pDataCtxt, OSSIZE offset)

Detailed Description

Definition in file rtXmlPull.c

rtXmlPull.h File Reference

```
#include "rtxsrc/rtxCtype.h"
```

```
#include "rtxsrc/rtxStream.h"
```

```
#include "rtxsrc/rtxStreamBuffered.h"
```

```
#include "rtxsrc/rtxErrCodes.h"
```

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

Data Structures

- struct OSXMLStrFragOffset
- struct OSXMLElementName
- struct OSXMLEvent
- struct OSXMLDataCtxt
- struct OSXMLElemNameOffset
- struct OSXMLStack
- struct OSXMLRewindPos
- struct OSXMLSrcPos
- struct OSXMLReader

Macros

- #define MAX_DELAYED_TASK_SIZE 10
- #define OSXML_DEFAULT_QNAME_BUF_SIZE 10
- #define OSXMLEVT_NONE 0
- #define OSXMLEVT_INITIAL OSXMLEVT_NONE
- #define OSXMLEVT_START_DOCUMENT 0x1
- #define OSXMLEVT_START_TAG 0x2

- #define OSXMLEVT_TEXT 0x4
- #define OSXMLEVT_END_TAG 0x8
- #define OSXMLEVT_END_DOCUMENT 0x10
- #define OSXMLEVT_COMMENT 0x20
- #define OSXMLEVT_PI 0x40
- #define OSXMLEVT_DTD 0x80
- #define OSXMLEVT_USED_FLAG 0x80000000u
- #define OSXMLEVT_ID_MASK (~OSXMLEVT_USED_FLAG)
- #define OSXMLEVT_ALL_MASK OSXMLEVT_ID_MASK
- #define XMLPREADER ((OSXMLCtxtInfo*) pctxt->pXMLInfo)->pXmlPPReader

Enumerations

- enum OSXMLStates {
  OSXMLS_HEADER= -1,
  OSXMLS_COMMENT= -2,
  OSXMLS_START_ELEMENT= -3,
  OSXMLS_DTD= -4,
  OSXMLS_CONTENT= -5,
  OSXMLS_PI= -6,
  OSXMLS_END_ELEMENT= -7,
  OSXMLS_CDATA= -8,
  OSXMLS_LAST= -9
  }
- enum OSXMLStatesIndex {
  OSXMLSI_HEADER= -OSXMLS_HEADER-1,
  OSXMLSI_COMMENT= -OSXMLS_COMMENT-1,
  OSXMLSI_START_ELEMENT= -OSXMLS_START_ELEMENT-1,
  OSXMLSI_DTD= -OSXMLS_DTD-1,
  OSXMLSI_CONTENT= -OSXMLS_CONTENT-1,
  OSXMLSI_PI= -OSXMLS_PI-1,
  OSXMLSI_END_ELEMENT= -OSXMLS_END_ELEMENT-1,
  OSXMLSI_CDATA= -OSXMLS_CDATA-1,
  OSXMLSI_LAST= -OSXMLS_LAST-1
  }
- enum OSXMLDataMode {
  OSXMLDM_NONE= 0,
  OSXMLDM_SIMULATED,
  OSXMLDM_CONTENT
  }

Typedefs

- typedef struct OSXMLEvent OSXMLEvent

- typedef struct OSXMLDataCtxt OSXMLDataCtxt
- typedef struct OSXMLStack OSXMLStack
- typedef struct OSXMLRewindPos OSXMLRewindPos
- typedef struct OSXMLSrcPos OSXMLSrcPos
- typedef struct OSXMLReader OSXMLReader

Functions

- EXTXMLMETHOD struct OSXMLReader * rtXmlRdCreateXmlReader (OSCTXT * pctxt)

  Creates XML reader.

- EXTXMLMETHOD OSXMLWhiteSpaceMode rtXmlRdSetWhiteSpaceMode (struct OSXMLReader * pReader, OSXMLWhiteSpaceMode whiteSpaceMode)

  Sets the whitespace treatment mode.

- EXTXMLMETHOD long rtXmlRdFirstData (struct OSXMLReader * pReader, OSXMLDataCtxt * pDataCtxt)

  Reads the first data chunk from content or attribute value.

- EXTXMLMETHOD long rtXmlRdNextData (struct OSXMLReader * pReader, OSXMLDataCtxt * pDataCtxt)

  Reads the next data chunk from content or attribute value.

- EXTXMLMETHOD int rtXmlRdSkipCurrentLevel (struct OSXMLReader * pReader)

  Skip the current level of XML document.

- EXTXMLMETHOD int rtXmlRdNext (struct OSXMLReader * pReader)

  The most low-level call.

- EXTXMLMETHOD int rtXmlRdNextTag (struct OSXMLReader * pReader)

  Reads until the next start tag is found.

- EXTXMLMETHOD int rtXmlRdNextEndTag (struct OSXMLReader * pReader)

  Reads until the next end tag is found.

- EXTXMLMETHOD OSXMLEvent rtXmlRdGetLastEvent (struct OSXMLReader * pReader)

  Returns information on the last processed event (ID and level).

- EXTXMLMETHOD int rtXmlRdGetTagName (struct OSXMLReader * pReader, OSXMLStrFragment * localName, OSINT16 * namespaceIndex)

  Returns the local name and namespace URI of the current element.

- EXTXMLMETHOD int rtXmlRdGetAttributeCount (struct OSXMLReader * pReader)

  Returns the number of attributes defined in the current element.

- EXTXMLMETHOD int rtXmlRdGetAttributeName (struct OSXMLReader * pReader, OSXMLNameFragments * pAttr, OSINT16 * pNsidx, size_t index)

  Gets the attribute name by index.

- EXTXMLMETHOD int rtXmlRdFirstAttr (struct OSXMLReader * pReader, OSXMLStrFragment * pAttrName, OSXMLStrFragment * pAttrValue)

  Gets the first attribute name and value.

- EXTXMLMETHOD int rtXmlRdNextAttr (struct OSXMLReader * pReader, OSXMLStrFragment * pAttrName, OSXMLStrFragment * pAttrValue)

  Gets the second attribute name and value.

- EXTXMLMETHOD int rtXmlRdSelectAttribute (struct OSXMLReader * pReader, size_t index)

  Select attribute by index to read the attribute value.

- EXTXMLMETHOD OSINT32 rtXmlRdGetCurrentLevel (struct OSXMLReader * pReader)

  Returns the current nesting level of where the pull parser currently is within the XML document tree.

- EXTXMLMETHOD int rtXmlRdNextEvent (struct OSXMLReader * pReader, OSUINT32 eventMask, OSINT32 maxLevel, OSXMLEvent * pLastEvent)

  This function reads XML until one of the events from eventMask happens.

- EXTXMLMETHOD OSXMLDataMode rtXmlRdGetDataMode (struct OSXMLReader * pReader)

  Returns the current data mode.

- EXTXMLMETHOD int rtXmlRdMarkLastEventDone (struct OSXMLReader * pReader)

  Marks the last event as done.

- EXTXMLMETHOD int rtXmlRdMarkLastEventActive (struct OSXMLReader * pReader)

  Marks the last event as active.

- EXTXMLMETHOD void rtXmlRdResetXmlReader (struct OSXMLReader * pReader)

  Reset XML reader.

- EXTXMLMETHOD void rtXmlRdMarkPos (struct OSXMLReader * pReader)

- EXTXMLMETHOD void rtXmlRdRewindToMarkedPos (struct OSXMLReader * pReader)

- EXTXMLMETHOD void rtXmlRdResetMarkedPos (struct OSXMLReader * pReader)

- EXTXMLMETHOD int rtXmlRdGetXSITypeAttr (struct OSXMLReader * pReader, const OSUTF8CHAR ** ppAttrValue, OSINT16 * nsidx, size_t * pLocalOffs)

- EXTXMLMETHOD OSBOOL rtXmlRdIsEmpty (struct OSXMLReader * pReader)

  Return TRUE when next will be end tag ("</").

- EXTXMLMETHOD void rtXmlRdGetSourcePosition (struct OSXMLReader * pReader, OSUINT32 * pLine, OSUINT32 * pColumn, OSUINT32 * pByteIndex, OSBOOL nextPos)

- EXTXMLMETHOD void rtXmlRdErrAddSrcPos (struct OSXMLReader * pReader, OSBOOL nextPos)
- EXTXMLMETHOD void rtXmlRdErrAddDataSrcPos (struct OSXMLReader * pReader, OSXMLDataCtxt * pDataCtxt, OSSIZE offset)

Detailed Description

Definition in file rtXmlPull.h

rtXmlPull.hh File Reference

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

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

Data Structures

- struct OSXMLAttribute
- struct OSXMLAttrOffset
- struct OSXMLStrFragStack
- struct OSXMLNamespace_
- struct OSXMLNamespacesStack

Macros

- #define MAX_MEMBLOCK_SIZE 2048

Typedefs

- typedef struct OSXMLStrFragStack OSXMLStrFragStack
- typedef struct OSXMLNamespace_ OSXMLNamespace_
- typedef struct OSXMLNamespacesStack OSXMLNamespacesStack

Functions

- const OSXMLAttrOffset * rtXmlRdGetAttribute (struct OSXMLReader * pReader, size_t index)

Detailed Description

Definition in file rtXmlPull.hh

rtXmlPutChar.c File Reference

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

```
#include "rtxsrc/rtxUTF16.h"
```

Variables

- static const OSUINT32 endian
- static const OSOCTET * pEndian

Functions

- EXTXMLMETHOD int rtXmlPutChar (OSCTXT * pctxt, const OSUTF8CHAR value)

Detailed Description

Definition in file rtXmlPutChar.c

rtXmlSetEncBufPtr.c File Reference

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

```
#include "rtxsrc/rtxContext.hh"
```

```
#include "rtxsrc/rtxStream.h"
```

Functions

- EXTXMLMETHOD int rtXmlSetEncBufPtr (OSCTXT * pctxt, OSOCTET * bufaddr, size_t bufsiz)

Detailed Description

Definition in file rtXmlSetEncBufPtr.c

rtXmlSetEncodingStr.c File Reference

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

```
#include "rtxsrc/rtxCtype.h"
```

Functions

- EXTXMLMETHOD int rtXmlSetEncodingStr (OSCTXT * pctxt, const OSUTF8CHAR * encodingStr)

  This function sets the XML output encoding to the given value.

Detailed Description

Definition in file rtXmlSetEncodingStr.c

rtXmlStrCmpAsc.c File Reference

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

Functions

- EXTXMLMETHOD OSBOOL rtXmlStrCmpAsc (const OSUTF8CHAR * text1, const char * text2)

Detailed Description

Definition in file rtXmlStrCmpAsc.c

rtXmlStrnCmpAsc.c File Reference

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

Functions

- EXTXMLMETHOD OSBOOL rtXmlStrnCmpAsc (const OSUTF8CHAR * text1, const char * text2, size_t len)

Detailed Description

Definition in file rtXmlStrnCmpAsc.c

rtXmlTreatWhitespaces.c File Reference

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

#include "rtxsrc/rtxCtype.h"
```

Functions

- EXTXMLMETHOD void rtXmlTreatWhitespaces (OSCTXT * pctxt, int whiteSpaceType)

Detailed Description

Definition in file rtXmlTreatWhitespaces.c

rtXmlWriteChars.c File Reference

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

#include "rtxsrc/rtxUTF16.h"

#include "rtxsrc/rtxLatin1.h"
```

Functions

- EXTXMLMETHOD int rtXmlWriteChars (OSCTXT * pctxt, const OSUTF8CHAR * value, size_t len)

Detailed Description

Definition in file rtXmlWriteChars.c

rtXmlWriteToFile.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxFile.h"`

Functions

- EXTXMLMETHOD int rtXmlWriteToFile (OSCTXT * pctxt, const char * filename)

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

Detailed Description

Definition in file rtXmlWriteToFile.c

rtXmlWriteUTF16ToFile.c File Reference

`#include "rtxmlsrc/osrtxml.h"`

`#include "rtxsrc/rtxFile.h"`

Functions

- EXTXMLMETHOD int rtXmlWriteUTF16ToFile (OSCTXT * pctxt, const char * filename)

Detailed Description

Definition in file rtXmlWriteUTF16ToFile.c

rtXmlXercesIF.cpp File Reference

`#include <stdarg.h>`

`#include <xercesc/util/PlatformUtils.hpp>`

`#include <xercesc/util/TransService.hpp>`

`#include <xercesc/util/BinInputStream.hpp>`

`#include <xercesc/util/XMLUTF8Transcoder.hpp>`

`#include <xercesc/sax/SAXException.hpp>`

`#include <xercesc/sax2/Attributes.hpp>`

`#include <xercesc/sax2/DefaultHandler.hpp>`

`#include <xercesc/sax2/SAX2XMLReader.hpp>`

`#include <xercesc/sax2/XMLReaderFactory.hpp>`

`#include <xercesc/framework/MemBufInputSource.hpp>`

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

```
#include "rtxsrc/rtxErrCodes.h"
```

```
#include "rtxsrc/OSRTInputStreamIF.h"
```

Data Structures

- struct OSXercesXMLReader
- struct OSXercesXMLReader::StopParserException
- struct OSCustomBinInputStream
- struct OSCustomInputSource
- struct OSXercesString
- struct OSXercesStringArray

Macros

- #define XML_BUF_SIZE 2048

Functions

- OSXMLReaderClass * rtSaxCppCreateXmlReader (OSXMLParserCtxtIF * pContext, OSXMLDefaultHandlerIF * pSaxHandler)
- int rtSaxCppEnableThreadSafety ( )
- void rtSaxCppLockXmlLibrary ( )
- void rtSaxCppUnlockXmlLibrary ( )

Detailed Description

Definition in file rtXmlXercesIF.cpp