XBinder

XML Schema Compiler
Version 2.0
C EXI Runtime
Reference Manual

Objective Systems, Inc. — August 2009

Author's Contact Information

Comments, suggestions, and inquiries regarding XBinder may be submitted via electronic mail to info@obj-sys.com.

Contents

Chapter 1

Main Page

C EXI Runtime Library Functions

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

The categories of functions provided are as follows:

- Functions to manage the EXI context.
- Functions to encode XML to EXI.
- Functions to encode EXI to XML.
- Functions to manage string tables.
- Functions to run and manage finite-state automata.
- EXI encoder and decoder implementations.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

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

Chapter 5

Module Documentation

5.1 EXI runtime library functions.

Files

- file osrtexi.h

  EXI low-level C context and structure definitions.

Classes

- struct OSEXIStateEvent

  This structure holds state/event information.

- struct OSEXIAutomaton

  This structure defines a finite state automata for EXI grammars.

- struct OSEXIAtmState

  This structure defines state information from an automaton that must be preserved at each stack level.

Defines

- #define rtEXISetBufPtr(pctxt, bufaddr, bufsiz) rtxCtxtSetBufPtr (pctxt, bufaddr, bufsiz)

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

- #define rtEXIGetMsgPtr(pctxt) (pctxt) → buffer.data

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

- #define rtEXIGetMsgLen(pctxt) (pctxt) → buffer.byteIndex

  This macro returns the length of the encoded XML message.

- #define rtEXISetOption(pctxt, option) EXICTXT(pctxt) → options |= option;

  This macro is used to set a bit flag in the EXI options bit mask.

- #define rtEXIClearOption(pctxt, option) EXICTXT(pctxt) → options &= ∼option;

  This macro is used to clear a bit flag in the EXI options bit mask.

- #define rtEXITestOption(pctxt, option) ((EXICTXT(pctxt) → options & option) != 0)

  This macro tests if the given option is set in the EXI context.

- #define rtEXISetValueChannelId(pctxt, channelId) EXICTXT(pctxt) → curChannelId = channelId

  This macro sets the compression channel ID in the EXI context.

- #define rtEXISetFragmentLevel(pctxt, fraglevel) EXICTXT(pctxt) → selfContainedLevel = fraglevel

  This macro sets level of elements, that encoded as self-contained fragment.

- #define rtEXISetFragmentElement(pctxt, fragelement) EXICTXT(pctxt) → selfContainedElemName = (const OSUTF8CHAR∗) (fragelement)

  This macro sets name of element, that encoded as self-contained fragment.

Typedefs

- typedef int(∗ OSEXIAtmAddTransFunc)(OSCTXT ∗pctxt, OSEXIAutomaton ∗pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent ∗pEvent, const OSEXIEventCode ∗pEventCode)

  Add transition function definition.

Functions

- EXTERNEXI int rtEXIInitContext (OSCTXT ∗pctxt)

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

- EXTERNEXI int rtEXIInitCtxtAppInfo (OSCTXT ∗pctxt)

  This function initializes the EXI application info section of the given context.

- EXTERNEXI int rtEXICtxtSetStateTable (OSCTXT ∗pctxt, const OSEXIStateTableRecord ∗statetab, OSINT16 nrows)

  This function will set the state table in the EXI context to the given value if it is not already set.

- EXTERNEXI void rtEXIEnableBitFieldTrace (OSCTXT ∗pctxt)

  This function turns on bit field tracing and initializes the bit field trace list.

- EXTERNEXI int rtEXISetUriPrefix (OSCTXT ∗pctxt, const OSUTF8CHAR ∗prefix, const OSUTF8CHAR ∗uri)

  This function assign URI prefix for EXI decoding.

- EXTERNEXI void rtEXIAutomatonInit (OSCTXT ∗pctxt, OSEXIAutomaton ∗pAutomaton, const OSXMLFullQName ∗pElemName, OSEXIState numStates)

  This function initializes the automaton to its default state.

- EXTERNEXI OSEXIAutomaton ∗ rtEXINewAutomaton (OSCTXT ∗pctxt, const OSXMLFullQName ∗pElemName, OSEXIState numStates)

  This function allocates memory for a new automaton structure and initializes the structure.

- EXTERNEXI OSEXIAutomaton * rtEXIAutomatonCopy (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton)

  This function copies an automaton structure.

- EXTERNEXI void rtEXIAutomatonFreeMem (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSBOOL dynamic)

  This function frees all memory within an Automaton structure.

- EXTERNEXI OSEXIAutomaton * rtEXIAutomatonAddTransition (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEventCode *pEventCode)

  This function adds a transition between two states.

- EXTERNEXI int rtEXIAtmAddUndeclaredItems (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, OSEXIEventCode *pEventCode, size_t numDeclAttrs, const OSXMLFullQName *declAttrs, OSEXIAtmAddTransFunc addTransFunc)

  This function adds all undeclared items (end element, start tag, and content) to an element automaton in schema-informed mode.

- EXTERNEXI int rtEXIAtmAddUndeclaredStartTagItems (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, OSEXIEventCode *pEventCode, size_t numDeclAttrs, const OSXMLFullQName *declAttrs, OSEXIAtmAddTransFunc addTransFunc)

  This function adds undeclared start tag items to an element automaton in schema-informed mode.

- EXTERNEXI int rtEXIAtmAddUndeclaredContentItems (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, OSEXIEventCode *pEventCode, OSEXIAtmAddTransFunc addTransFunc)

  This function adds undeclared content items to an element automaton in schema-informed mode.

- EXTERNEXI OSEXIAutomaton * rtEXIAutomatonInitCopy (OSCTXT *pctxt, OSEXIAutomaton *pDestAtm, OSEXIAutomaton *pSrcAtm)

  This function initializes an automaton structure using the data from an existing automaton.

- EXTERNEXI int rtEXIAutomatonPush (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton)

  This function pushes an automaton onto the context stack.

- EXTERNEXI OSEXIAutomaton * rtEXIAutomatonPop (OSCTXT *pctxt)

  This function pops an automaton from the context stack.

- EXTERNEXI OSEXIEventCodeGroup * rtEXIAtmGetCurrentEventCodeGroup (OSEXIAutomaton *pAutomaton)

  This function returns a pointer to the event code group corresponding to the current state.

- EXTERNEXI OSEXIAutomaton * rtEXIGetDocAutomaton (OSCTXT *pctxt, size_t numGblElems, const OSXMLFullQName *gblElems, OSEXIAtmAddTransFunc addTransFunc)

  This functions returns an automaton that accepts the built-in document grammar.

- EXTERNEXI OSEXIAutomaton * rtEXIGetElemAutomaton (OSCTXT *pctxt, OSXMLFullQName *pqname, int(*addTransFunc)(OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent *pEvent, const OSEXIEventCode *pEventCode))

  This function returns an automaton that accepts the built-in element grammar.

5.1.1 Define Documentation

5.1.1.1 #define rtEXIClearOption(pctxt, option) EXICTXT(pctxt) → options &= ∼option;

This macro is used to clear a bit flag in the EXI options bit mask.

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.

option Option bit mask value as defined earlier in this header file (for example, OSEXI_PRESERVE_DTD).

Definition at line 386 of file osrtexi.h.

5.1.1.2 #define rtEXIGetMsgLen(pctxt) (pctxt) → buffer.byteIndex

This macro returns the length of the encoded XML message.

Parameters:

pctxt Pointer to a context structure.

Definition at line 359 of file osrtexi.h.

5.1.1.3 #define rtEXIGetMsgPtr(pctxt) (pctxt) → buffer.data

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

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

Parameters:

pctxt Pointer to a context structure.

Definition at line 352 of file osrtexi.h.

5.1.1.4 #define rtEXISetBufPtr(pctxt, bufaddr, bufsiz) rtxCtxtSetBufPtr (pctxt, bufaddr, bufsiz)

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

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

Parameters:

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

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

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

Definition at line 341 of file osrtexi.h.

5.1.1.5 #define rtEXISetFragmentElement(pctxt, fragelement) EXICTXT(pctxt) → selfContainedElemName = (const OSUTF8CHAR∗) (fragelement)

This macro sets name of element, that encoded as self-contained fragment.

Parameters:

pctxt Pointer to a context structure.

fragelement Element name.

Definition at line 429 of file osrtexi.h.

5.1.1.6 #define rtEXISetFragmentLevel(pctxt, fraglevel) EXICTXT(pctxt) → selfContainedLevel = fraglevel

This macro sets level of elements, that encoded as self-contained fragment.

Parameters:

pctxt Pointer to a context structure.

fraglevel Element level.

Definition at line 420 of file osrtexi.h.

5.1.1.7 #define rtEXISetOption(pctxt, option) EXICTXT(pctxt) → options |= option;

This macro is used to set a bit flag in the EXI options bit mask.

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.

option Option bit mask value as defined earlier in this header file (for example, OSEXI_PRESERVE_DTD).

Definition at line 372 of file osrtexi.h.

5.1.1.8 #define rtEXISetValueChannelId(pctxt, channelId) EXICTXT(pctxt) → curChannelId = channelId

This macro sets the compression channel ID in the EXI context.

Parameters:

pctxt Pointer to a context structure.

channelId Value channel id.

Definition at line 411 of file osrtexi.h.

5.1.1.9 #define rtEXITestOption(pctxt, option) ((EXICTXT(pctxt) → options & option) != 0)

This macro tests if the given option is set in the EXI context.

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.

option Option bit mask value as defined earlier in this header file (for example, OSEXI_PRESERVE_DTD).

Returns:

True if set, false if not.

Definition at line 402 of file osrtexi.h.

5.1.2 Function Documentation

5.1.2.1 EXTERNEXI int rtEXIAtmAddUndeclaredContentItems (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, OSEXIState *fromState*, OSEXIState *toState*, OSEXIEventCode * *pEventCode*, OSEXIAtmAddTransFunc *addTransFunc*)

This function adds undeclared content items to an element automaton in schema-informed mode.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

fromState The origin state for this transition.

toState The destination state for this transition.

pEventCode Pointer to event code structure that contains the current state of assigned events in the grammar.

Returns:

Status of operation: 0 if successful; otherwise, a negative return code.

5.1.2.2 EXTERNEXI int rtEXIAtmAddUndeclaredItems (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, OSEXIState *fromState*, OSEXIState *toState*, OSEXIEventCode * *pEventCode*, size_t *numDeclAttrs*, const OSXMLFullQName * *declAttrs*, OSEXIAtmAddTransFunc *addTransFunc*)

This function adds all undeclared items (end element, start tag, and content) to an element automaton in schema-informed mode.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

fromState The origin state for this transition.

toState The destination state for this transition.

pEventCode Pointer to event code structure that contains the current state of assigned events in the grammar.

numDeclAttrs Number of items in declAttrs array.

declAttrs Array of declared attribute QNames. These are attributes that were declared in the schema but which are not valid within the current context.

Returns:

Status of operation: 0 if successful; otherwise, a negative return code.

5.1.2.3 EXTERNEXI int rtEXIAtmAddUndeclaredStartTagItems (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, OSEXIState *fromState*, OSEXIState *toState*, OSEXIEventCode * *pEventCode*, size_t *numDeclAttrs*, const OSXMLFullQName * *declAttrs*, OSEXIAtmAddTransFunc *addTransFunc*)

This function adds undeclared start tag items to an element automaton in schema-informed mode.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

fromState The origin state for this transition.

toState The destination state for this transition.

pEventCode Pointer to event code structure that contains the current state of assigned events in the grammar.

numDeclAttrs Number of items in declAttrs array.

declAttrs Array of declared attribute QNames. These are attributes that were declared in the schema but which are not valid within the current context.

Returns:

Status of operation: 0 if successful; otherwise, a negative return code.

5.1.2.4 EXTERNEXI OSEXIEventCodeGroup* rtEXIAtmGetCurrentEventCodeGroup (OSEXIAutomaton * *pAutomaton*)

This function returns a pointer to the event code group corresponding to the current state.

Parameters:

pAutomaton Pointer to automaton structure.

Returns:

Pointer to event group group.

5.1.2.5 EXTERNEXI OSEXIAutomaton* rtEXIAutomatonAddTransition (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, OSEXIState *fromState*, OSEXIState *toState*, const OSEXIEventCode * *pEventCode*)

This function adds a transition between two states.

The transition is defined by a pair of states, and event and event code.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

fromState The origin state for this transition.

toState The destination state for this transition.

pEventCode The event code returned after the transition.

5.1.2.6 EXTERNEXI OSEXIAutomaton* rtEXIAutomatonCopy (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*)

This function copies an automaton structure.

If the structure is closed, a shallow copy is done; otherwise, a deep copy.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure to copy.

Returns:

Copied automaton structure.

5.1.2.7 EXTERNEXI void rtEXIAutomatonFreeMem (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, OSBOOL *dynamic*)

This function frees all memory within an Automaton structure.

Parameters:

pctxt Pointer to a context structure.

pAutomaton Pointer to object in which memory will be freed.

dynamic Boolean indicating if pAutomaton is dynamic. If true, the memory for pAutomaton is freed.

5.1.2.8 EXTERNEXI void rtEXIAutomatonInit (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, const OSXMLFullQName * *pElemName*, OSEXIState *numStates*)

This function initializes the automaton to its default state.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure to initialize.

pElemName Pointer to element QName.

numStates Number of states (if known) or zero.

5.1.2.9 EXTERNEXI OSEXIAutomaton* rtEXIAutomatonInitCopy (OSCTXT * *pctxt*, OSEXIAutomaton * *pDestAtm*, OSEXIAutomaton * *pSrcAtm*)

This function initializes an automaton structure using the data from an existing automaton.

Parameters:

pctxt Pointer to context block structure.

pDestAtm Pointer to destination (target) automaton structure.

pSrcAtm Pointer to source automaton structure.

Returns:

Copied (destination) automaton structure.

5.1.2.10 EXTERNEXI OSEXIAutomaton* rtEXIAutomatonPop (OSCTXT * *pctxt*)

This function pops an automaton from the context stack.

Parameters:

pctxt Pointer to context block structure.

Returns:

Last automaton pushed on stack or NULL if stack empty or error.

5.1.2.11 EXTERNEXI int rtEXIAutomatonPush (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*)

This function pushes an automaton onto the context stack.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Automaton to be pushed onto stack.

Returns:

Status of operation: 0 = success, negative = error.

5.1.2.12 EXTERNEXI int rtEXICtxtSetStateTable (OSCTXT * *pctxt*, const OSEXIStateTableRecord * *statetab*, OSINT16 *nrows*)

This function will set the state table in the EXI context to the given value if it is not already set.

Parameters:

pctxt Pointer to OSCTXT structure

statetab Pointer to state table structure.

nrows Number of rows in the state table.

Returns:

Zero if operation was successful or negative error code if failure.

5.1.2.13 EXTERNEXI void rtEXIEnableBitFieldTrace (OSCTXT * *pctxt*)

This function turns on bit field tracing and initializes the bit field trace list.

Parameters:

pctxt Pointer to OSCTXT structure

5.1.2.14 EXTERNEXI OSEXIAutomaton* rtEXIGetDocAutomaton (OSCTXT * *pctxt*, size_t *numGblElems*, const OSXMLFullQName * *gblElems*, OSEXIAtmAddTransFunc *addTransFunc*)

This functions returns an automaton that accepts the built-in document grammar.

This grammar is not extensible, so all the event codes created are immutable.

Parameters:

pctxt Pointer to context block structure.

numGblElems Number of global elements.

gblElems Array of global element objects.

addTransFunc Pointer to function to add a transition.

Returns:

Pointer to document automaton.

5.1.2.15 EXTERNEXI OSEXIAutomaton* rtEXIGetElemAutomaton (OSCTXT * *pctxt*, OSXMLFullQName * *pqname*, int(*)(OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent *pEvent, const OSEXIEventCode *pEventCode) *addTransFunc*)

This function returns an automaton that accepts the built-in element grammar.

The `elementName` is associated with the automaton.

Parameters:

pctxt Pointer to context block structure.

pqname Pointer to element QName.

addTransFunc Pointer to function to add a transition.

Returns:

Pointer to element automaton.

5.1.2.16 EXTERNEXI int rtEXIInitContext (OSCTXT * *pctxt*)

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

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Completion status of operation:

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

5.1.2.17 EXTERNEXI int rtEXIInitCtxtAppInfo (OSCTXT * *pctxt*)

This function initializes the EXI application info section of the given context.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Completion status of operation:

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

5.1.2.18 EXTERNEXI OSEXIAutomaton* rtEXINewAutomaton (OSCTXT * *pctxt*, const OSXMLFullQName * *pElemName*, OSEXIState *numStates*)

This function allocates memory for a new automaton structure and initializes the structure.

Parameters:

pctxt Pointer to context block structure.

pElemName Pointer to element QName.

numStates Number of states (if known) or zero.

Returns:

Allocated automaton structure or NULL if no dynamic memory available.

5.1.2.19 EXTERNEXI int rtEXISetUriPrefix (OSCTXT * *pctxt*, const OSUTF8CHAR * *prefix*, const OSUTF8CHAR * *uri*)

This function assign URI prefix for EXI decoding.

Parameters:

pctxt Pointer to OSCTXT structure

prefix Namespace prefix

uri Namespace URI

Returns:

Completion status of operation:

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

5.2 EXI to XML serialization functions.

Typedefs

- typedef int(∗ StartElementHandler)(void ∗userData, const OSUTF8CHAR ∗localname, const OSUTF8CHAR ∗qname, const OSUTF8CHAR ∗const ∗attrs)

  SAX start element handler callback function definition.

- typedef int(∗ EndElementHandler)(void ∗userData, const OSUTF8CHAR ∗localname, const OSUTF8CHAR ∗qname)

  SAX end element handler callback function definition.

- typedef int(∗ CharacterDataHandler)(void ∗userData, const OSUTF8CHAR ∗value, int len)

  SAX characters handler callback function definition.

Functions

- EXTERNEXI EXI2SAXReader ∗ rtEXI2SAXCreateReader (OSCTXT ∗pctxt, void ∗pUserData, StartElementHandler startElemFunc, EndElementHandler endElemFunc, CharacterDataHandler charactersFunc)

  This function is used to create a reader structure for use by the main SAX parser function.

- EXTERNEXI int rtEXI2SAXParse (EXI2SAXReader ∗pReader)

  This is the main SAX parser function.

- EXTERNEXI int rtExi2XmlStream (OSCTXT ∗pctxt, OSCTXT ∗poutctxt)

  This function transcodes an EXI encoded message to an XML output stream.

5.2.1 Typedef Documentation

5.2.1.1 typedef int(∗ CharacterDataHandler)(void ∗userData, const OSUTF8CHAR ∗value, int len)

SAX characters handler callback function definition.

Parameters:

userData User-defined data structure.

value Character data. String is not null-terminated.

len Number of characters in character string.

Returns:

Status of operation.

Definition at line 76 of file rtEXI2SAX.h.

5.2.1.2 typedef int(* EndElementHandler)(void *userData, const OSUTF8CHAR *localname, const OSUTF8CHAR *qname)

SAX end element handler callback function definition.

Parameters:

userData User-defined data structure.

localname Local name of element.

qname Qualified name of element.

Returns:

Status of operation.

Definition at line 63 of file rtEXI2SAX.h.

5.2.1.3 typedef int(* StartElementHandler)(void *userData, const OSUTF8CHAR *localname, const OSUTF8CHAR *qname, const OSUTF8CHAR *const *attrs)

SAX start element handler callback function definition.

Parameters:

userData User-defined data structure.

localname Local name of element.

qname Qualified name of element.

attrs Array of attribute name/value pairs, terminated by 0;

Returns:

Status of operation.

Definition at line 51 of file rtEXI2SAX.h.

5.2.2 Function Documentation

5.2.2.1 EXTERNEXI EXI2SAXReader* rtEXI2SAXCreateReader (OSCTXT * *pctxt*, void * *pUserData*, StartElementHandler *startElemFunc*, EndElementHandler *endElemFunc*, CharacterDataHandler *charactersFunc*)

This function is used to create a reader structure for use by the main SAX parser function.

Parameters:

pctxt Pointer to a context structure.

pUserData Pointer to a user defined data structure that will be passed to the SAX callback functions.

startElemFunc Start element callback function.

endElemFunc End element callback function.

charactersFunc Characters callback function.

Returns:

Initialized SAX reader structure.

5.2.2.2 EXTERNEXI int rtEXI2SAXParse (EXI2SAXReader * *pReader*)

This is the main SAX parser function.

It converts decoded EXI events to SAX callback function invocations.

Parameters:

pReader Pointer to EXI-to-SAX reader structure.

Returns:

Status of parse operation. Zero if success, negative status code if error.

5.2.2.3 EXTERNEXI int rtExi2XmlStream (OSCTXT * *pctxt*, OSCTXT * *poutctxt*)

This function transcodes an EXI encoded message to an XML output stream.

The input context structure describes an encoded EXI message as either an input stream or in memory in its internal buffer. The output context is assumed to contain an initialized XML output stream set up with one of the rtxStream*CreateWriter functions (for example, rtxStreamFileCreateWriter to write to a file).

Parameters:

pctxt Pointer to input context structure. This is assumed to contain an input stream or memory buffer containing an EXI encoded message.

poutctxt Pointer to an output context assumed to contain an output stream create with an rtxStreamCreateWriter function.

Returns:

Completion status of operation:

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

5.3 EXI decode structures and functions.

Classes

- struct OSEXIDecStringTable

  This structure defines the structure of the various string table partitions used by the decoder.

- struct OSEXIDecStringTables

  This structure defines the complete set of string table partitions used by the decoder.

Functions

- EXTERNEXI int rtEXIDecAtmAddTransition (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent *pEvent, const OSEXIEventCode *pEventCode)

  This function adds a transition between two states.

- EXTERNEXI OSEXIEvent * rtEXIDecAutomatonAdvance (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, const OSEXIEventCode *pEventCode, OSBOOL dynamicItems)

  This function advances the decoder automaton based on event code, adding new transitions if `dynamicItems` *is set to* `true` *and either SE(*) or AT(*) are matched instead of SE(qname) or AT(qname), respectively.*

- EXTERNEXI OSEXIAutomaton * rtEXIDecGetDocAutomaton (OSCTXT *pctxt, size_t numGblElems, const OSXMLFullQName *gblElems)

  This functions returns an automaton that accepts the built-in document grammar.

- EXTERNEXI OSEXIAutomaton * rtEXIDecGetElemAutomaton (OSCTXT *pctxt, OSXMLFullQName *pqname)

  This function returns an automaton that accepts the built-in element grammar.

- EXTERNEXI int rtEXIDecAttribute (OSCTXT *pctxt, OSXMLFullQName *pqname, const OSUTF8CHAR **ppvalue)

  Decodes the attribute at the current position in the decode stream.

- EXTERNEXI int rtEXIDecBinary (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)

  This function decodes the contents of a binary value into a static memory structure.

- EXTERNEXI int rtEXIDecBoolValue (OSCTXT *pctxt, OSBOOL *pvalue)

  This function decodes a boolean value.

- EXTERNEXI int rtEXIDecBoolValueWithPattern (OSCTXT *pctxt, OSBOOL *pvalue, OSUINT8 pattern)

  This function decodes a boolean value with pattern facet.

- EXTERNEXI int rtEXIDec_CH_String_EE (OSCTXT *pctxt, const OSXMLFullQName *pqname, const OSUINT16 *charSet, const OSUTF8CHAR **ppvalue)

  This function decodes a CH event (assumed to be a one part code = 0 in a 1 bit field) followed by string content followed by an EE event (assumed to be a one part code = 0 in a 1 bit field).

- EXTERNEXI int rtEXIDecDate (OSCTXT *pctxt, OSNumDateTime *pvalue)

This function decodes a date value into a structured variable.

- EXTERNEXI int rtEXIDecDateString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a date value into a string.

- EXTERNEXI int rtEXIDecDateTime (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a date/time value into a structured variable.

- EXTERNEXI int rtEXIDecDateTimeString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a date/time value into a string.

- EXTERNEXI int rtEXIDecDecimalValue (OSCTXT *pctxt, OSREAL *pvalue)

  This function decodes a decimal value.

- EXTERNEXI int rtEXIDecDocumentType (OSCTXT *pctxt, const OSUTF8CHAR **ppName, const OSUTF8CHAR **ppPublic, const OSUTF8CHAR **ppSystem, const OSUTF8CHAR **ppText)

  This function decodes an XML document type declaration (DTD).

- EXTERNEXI int rtEXIDecDoubleValue (OSCTXT *pctxt, OSREAL *pvalue)

  This function decodes a double value.

- EXTERNEXI int rtEXIDecDynBinary (OSCTXT *pctxt, OSDynOctStr *pvalue)

  This function decodes a binary value into a structured variable.

- EXTERNEXI int rtEXIDecEndDocument (OSCTXT *pctxt)

  This function ends decoding of EXI stream.

- EXTERNEXI int rtEXIDecEndEvent (OSCTXT *pctxt)

  This function decodes of event codes of element end grammar.

- EXTERNEXI int rtEXIDecEndEventCompact (OSCTXT *pctxt)

  This function decodes of event codes of element end grammar.

- EXTERNEXI int rtEXIDecEndEventStrict (OSCTXT *pctxt)

  This function decodes of event codes of element end grammar.

- EXTERNEXI int rtEXIDecEventCodePart1 (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 nbits, OSUINT32 maxval)

  This function decodes part 1 of an event code.

- EXTERNEXI int rtEXIDecGetStateIndex (OSCTXT *pctxt, OSUINT32 flags)

  This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

- EXTERNEXI int rtEXIDecGetStateIndexCompact (OSCTXT *pctxt, OSUINT16 stx, const OSEXIStateTableRecord statetab[ ], size_t nrows, OSUINT32 flags, OSUINT32 *prepcnt)

  This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

- EXTERNEXI int rtEXIDecGetStateIndexStrict (OSCTXT *pctxt, OSUINT16 stx, const OSEXIStateTableRecord statetab[ ], size_t nrows, OSUINT32 flags, OSUINT32 *prepcnt)

This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

- EXTERNEXI int rtEXIDecFloatValue (OSCTXT *pctxt, OSFLOAT *pvalue)

  This function decodes a float value.

- EXTERNEXI int rtEXIDecGDay (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gDay value into a structured variable.

- EXTERNEXI int rtEXIDecGDayString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gDay value into a string.

- EXTERNEXI int rtEXIDecGMonth (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gMonth value into a structured variable.

- EXTERNEXI int rtEXIDecGMonthDay (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gMonthDay value into a structured variable.

- EXTERNEXI int rtEXIDecGMonthDayString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gMonthDay value into a string.

- EXTERNEXI int rtEXIDecGMonthString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gMonth value into a string.

- EXTERNEXI int rtEXIDecGYear (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gYear value into a structured variable.

- EXTERNEXI int rtEXIDecGYearMonth (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gYearMonth value into a structured variable.

- EXTERNEXI int rtEXIDecGYearMonthString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gYearMonth value into a string.

- EXTERNEXI int rtEXIDecGYearString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gYear value into a string.

- EXTERNEXI OSBOOL rtEXIDecHasNext (OSCTXT *pctxt)

  This function checks for additonal events in the decode stream.

- EXTERNEXI int rtEXIDecHeader (OSCTXT *pctxt)

  This function decodes the EXI header at the start of a message and uses the results to set internal flags within the context structure.

- EXTERNEXI int rtEXIDecInitCompression (OSCTXT *pctxt, OSUINT32 nmChannelIds)

  This function initialize decoder to use compressed or precompressed EXI stream.

- EXTERNEXI int rtEXIDecInt16Value (OSCTXT *pctxt, OSINT16 *pvalue)

  This function decodes a 16-bit signed integer value.

- EXTERNEXI int rtEXIDecInt64Value (OSCTXT *pctxt, OSINT64 *pvalue)

This function decodes a 64-bit signed integer value.

- EXTERNEXI int rtEXIDecInt8Value (OSCTXT *pctxt, OSINT8 *pvalue)

  This function decodes a 8-bit signed integer value.

- EXTERNEXI int rtEXIDecIntValue (OSCTXT *pctxt, OSINT32 *pvalue)

  This function decodes a signed integer value.

- EXTERNEXI int rtEXIDecLocalName (OSCTXT *pctxt, const OSUTF8CHAR **ppname)

  Returns the local name associated with the current event.

- EXTERNEXI int rtEXIDecNamespaceURI (OSCTXT *pctxt, const OSUTF8CHAR **ppNSURI)

  Returns the namespace associated with the current event.

- EXTERNEXI int rtEXIDecNBitUIntValue (OSCTXT *pctxt, OSUINT32 *pvalue, OSUINT32 nbits)

  This function decodes an unsigned integer value from a bit field of the given width.

- EXTERNEXI int rtEXIDecNextEventType (OSCTXT *pctxt, OSEXIEventType *pEventType)

  Returns the next `OSEXIEventType` *read by this decoder.*

- EXTERNEXI int rtEXIDecoderInit (OSCTXT *pctxt)

  This function initializes the decoder.

- EXTERNEXI int rtEXIDecPrefix (OSCTXT *pctxt, const OSUTF8CHAR *uri, const OSUTF8CHAR **ppPrefix)

  Returns the namespace associated with the current event.

- EXTERNEXI int rtEXIDecProcessingInstruction (OSCTXT *pctxt, const OSUTF8CHAR **ppTarget, const OSUTF8CHAR **ppData)

  This function decodes an XML processing instruction (PI).

- EXTERNEXI int rtEXIDecQName (OSCTXT *pctxt, OSXMLFullQName *pqname)

  Returns the qname associated with the current event.

- EXTERNEXI int rtEXIDecQNameValue (OSCTXT *pctxt, OSXMLFullQName *pvalue, const OSXMLFullQName *pqname)

  Decode the qname value for AT and CH events.

- EXTERNEXI int rtEXIDecReset (OSCTXT *pctxt)

  Resets the decoder for decoding a new message instance.

- EXTERNEXI int rtEXIDecSimpleTypeEvent (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 flags)

  This function decodes of event codes of simple type grammar.

- EXTERNEXI int rtEXIDecSimpleTypeEventCompact (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 flags)

  This function decodes of event codes of simple type grammar.

- EXTERNEXI int rtEXIDecSimpleTypeEventStrict (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 flags)

  This function decodes of event codes of simple type grammar.

- EXTERNEXI int rtEXIDecString (OSCTXT *pctxt, const OSXMLFullQName *pqname, const OSUINT16 *charSet, const OSUTF8CHAR **ppvalue)

  Returns the value associated with the current event.

- EXTERNEXI int rtEXIDecStringToCharArray (OSCTXT *pctxt, const OSUTF8CHAR *target, size_t start, size_t length)

  Similar to `rtEXIDecString` *but the characters are copied into a fixed-size character array.*

- EXTERNEXI int rtEXIDecStringLength (OSCTXT *pctxt)

  Returns the length of the string returned by `rtEXIDecString`*.*

- EXTERNEXI int rtEXIDecTime (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a time value into a structured variable.

- EXTERNEXI int rtEXIDecTimeString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a time value into a string.

- EXTERNEXI int rtEXIDecUInt16Value (OSCTXT *pctxt, OSUINT16 *pvalue)

  This function decodes an 16-bit unsigned integer value.

- EXTERNEXI int rtEXIDecUInt64Value (OSCTXT *pctxt, OSUINT64 *pvalue)

  This function decodes an 64-bit unsigned integer value.

- EXTERNEXI int rtEXIDecUInt8Value (OSCTXT *pctxt, OSUINT8 *pvalue)

  This function decodes an 8-bit unsigned integer value.

- EXTERNEXI int rtEXIDecUIntValue (OSCTXT *pctxt, OSUINT32 *pvalue)

  This function decodes an unsigned integer value.

- EXTERNEXI int rtEXIDecUTF8Str (OSCTXT *pctxt, OSUTF8CHAR **ppvalue, const OSUINT16 *charSet)

  This function decodes a UTF-8 string value.

- EXTERNEXI int rtEXIDecUTF8Chars (OSCTXT *pctxt, OSUTF8CHAR **ppvalue, OSUINT32 nchars, const OSUINT16 *charSet)

  This function reads the given number of characters from the decode stream and creates a UTF-8 string.

- EXTERNEXI void rtEXIDecStringTableInit (OSCTXT *pctxt, OSEXIDecStringTable *pstrtab, size_t capacity)

  This function initializes the given string table structure.

- EXTERNEXI OSEXIDecStringTable * rtEXIDecNewStringTable (OSCTXT *pctxt, size_t capacity)

  This function allocates and initializes a new string table structure.

- EXTERNEXI void rtEXIDecStringTableClear (OSCTXT *pctxt, OSEXIDecStringTable *pstrtab)

  This function clears all strings out of the existing table.

- EXTERNEXI OSUINT32 rtEXIDecStringTableAdd (OSCTXT *pctxt, OSEXIDecStringTable *pstrtab, const OSUTF8CHAR *str)

  This function adds a string to the given string table.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStringTableGetString (OSEXIDecStringTable *pstrtab, OSUINT32 index)

  This function gets the string at the given index (i.e.

- EXTERNEXI void rtEXIDecStrTabsInit (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs)

  This function initializes all EXI string table partitions.

- EXTERNEXI void rtEXIDecStrTabsClear (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs)

  This function clears all EXI string table partitions.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddURI (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function will add a URI to the URI string table partition.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetURI (OSEXIDecStringTables *pstrtabs, OSUINT32 index)

  This function will get the compact identifier of the given URI from the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetURITableSize (OSEXIDecStringTables *pstrtabs)

  This function returns the current number of entries in the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddPrefix (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *prefix)

  This function adds the given prefix to the prefix table partition identified by the given URI.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetPrefix (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, OSUINT32 index)

  This function will get the compact identifier of the given prefix from the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetPrefixTableSize (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddLocalName (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *name)

  This function adds the given local name to the local name table partition identified by the given URI.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetLocalName (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, OSUINT32 index)

  This function will get the compact identifier of the given localName from the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetLocalNameTableSize (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddLocalValue (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSXMLFullQName *qname, const OSUTF8CHAR *value)

  This function adds the given local value to the local value table partition identified by the given QName.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetLocalValue (OSEXIDecStringTables *pstrtabs, const OSXMLFullQName *qname, OSUINT32 index)

  This function will get the compact identifier of the given local value from the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetLocalValueTableSize (OSEXIDecStringTables *pstrtabs, const OSXMLFullQName *qname)

  This function returns the current number of entries in the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddGlobalValue (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *value)

  This function will add a string value to the global value string table partition.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetGlobalValue (OSEXIDecStringTables *pstrtabs, OSUINT32 index)

  This function will get the compact identifier of the given string value from the global value string table partition.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetGlobalValueTableSize (OSEXIDecStringTables *pstrtabs)

  This function returns the current number of entries in the global value string table partition.

5.3.1 Function Documentation

5.3.1.1 EXTERNEXI int rtEXIDec_CH_String_EE (OSCTXT * *pctxt*, const OSXMLFullQName * *pqname*, const OSUINT16 * *charSet*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a CH event (assumed to be a one part code = 0 in a 1 bit field) followed by string content followed by an EE event (assumed to be a one part code = 0 in a 1 bit field).

Parameters:

pctxt Pointer to a context structure.

pqname QName the value is associated with. This should be the QName that was decoded from the stream prior to calling this function.

charSet Pointer to restricted character set to be used for decoding the string or NULL if not restricted.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.2 EXTERNEXI int rtEXIDecAtmAddTransition (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, OSEXIState *fromState*, OSEXIState *toState*, const OSEXIEvent * *pEvent*, const OSEXIEventCode * *pEventCode*)

This function adds a transition between two states.

The transition is defined by a pair of states, and event and event code.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

fromState The origin state for this transition.

toState The destination state for this transition.

pEvent The event on which this transition occurs.

pEventCode The event code returned after the transition.

Returns:

Zero if operation was successful; a negative status code otherwise.

5.3.1.3 EXTERNEXI int rtEXIDecAttribute (OSCTXT * *pctxt*, OSXMLFullQName * *pqname*, const OSUTF8CHAR ** *ppvalue*)

Decodes the attribute at the current position in the decode stream.

This function is called when the application receives an AT event to get the attribute data.

Parameters:

pctxt Pointer to a context structure.

pqname Pointer to variable to receive decoded attribute QName. Memory for the decoded name components is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.4 EXTERNEXI OSEXIEvent* rtEXIDecAutomatonAdvance (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, const OSEXIEventCode * *pEventCode*, OSBOOL *dynamicItems*)

This function advances the decoder automaton based on event code, adding new transitions if `dynamicItems` is set to `true` and either SE(*) or AT(*) are matched instead of SE(qname) or AT(qname), respectively.

Dynamic transitions for CH are also added according to the spec.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

pEventCode Event code on which to advance the automaton.

dynamicItems Flag to add dynamic transitions to the automaton.

Returns:

The event code returned as a result of this transition.

5.3.1.5 EXTERNEXI int rtEXIDecBinary (OSCTXT * *pctxt*, OSOCTET * *pvalue*, OSUINT32 * *pnocts*, OSINT32 *bufsize*)

This function decodes the contents of a binary value into a static memory structure.

Parameters:

pctxt Pointer to context block structure.

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

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

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

Returns:

Completion status of operation:

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

5.3.1.6 EXTERNEXI int rtEXIDecBoolValue (OSCTXT * *pctxt*, OSBOOL * *pvalue*)

This function decodes a boolean value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to boolean to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.7 EXTERNEXI int rtEXIDecBoolValueWithPattern (OSCTXT * *pctxt*, OSBOOL * *pvalue*, OSUINT8 *pattern*)

This function decodes a boolean value with pattern facet.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to boolean to receive decoded value.

pattern Mask of enabled values:

- 0 - "false"
- 1 - "0"
- 2 - "true"
- 3 - "1"

Returns:

Completion status of operation:

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

5.3.1.8 EXTERNEXI int rtEXIDecDate (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a date value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.9 EXTERNEXI int rtEXIDecDateString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a date value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.10 EXTERNEXI int rtEXIDecDateTime (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a date/time value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.11 EXTERNEXI int rtEXIDecDateTimeString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a date/time value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.12 EXTERNEXI int rtEXIDecDecimalValue (OSCTXT * *pctxt*, OSREAL * *pvalue*)

This function decodes a decimal value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to variable to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.13 EXTERNEXI int rtEXIDecDocumentType (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppName*, const OSUTF8CHAR ** *ppPublic*, const OSUTF8CHAR ** *ppSystem*, const OSUTF8CHAR ** *ppText*)

This function decodes an XML document type declaration (DTD).

Parameters:

pctxt Pointer to a context structure.

ppName Pointer to variable to receive decoded DTD name. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

ppPublic Pointer to variable to receive decoded DT public data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

ppSystem Pointer to variable to receive decoded DT system data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

ppText Pointer to variable to receive decoded DT text. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.14 EXTERNEXI int rtEXIDecDoubleValue (OSCTXT * *pctxt*, OSREAL * *pvalue*)

This function decodes a double value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to variable to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.15 EXTERNEXI int rtEXIDecDynBinary (OSCTXT * *pctxt*, OSDynOctStr * *pvalue*)

This function decodes a binary value into a structured variable.

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

Parameters:

pctxt Pointer to OSCTXT structure

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

Returns:

Completion status of operation:

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

5.3.1.16 EXTERNEXI int rtEXIDecEndDocument (OSCTXT * *pctxt*)

This function ends decoding of EXI stream.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Completion status of operation:

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

5.3.1.17 EXTERNEXI int rtEXIDecEndEvent (OSCTXT * *pctxt*)

This function decodes of event codes of element end grammar.

It will decode and skip undeclared items until a EE event code is received. Funcion use "default" mode encoding.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Completion status of operation:

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

5.3.1.18 EXTERNEXI int rtEXIDecEndEventCompact (OSCTXT * *pctxt*)

This function decodes of event codes of element end grammar.

It will decode and skip undeclared items until a EE event code is received. Funcion use "default" mode encoding, but it is break on undeclared event.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Completion status of operation:

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

5.3.1.19 EXTERNEXI int rtEXIDecEndEventStrict (OSCTXT * *pctxt*)

This function decodes of event codes of element end grammar.

It will decode and skip undeclared items until a EE event code is received. Funcion use "strict" mode encoding.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Completion status of operation:

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

5.3.1.20 EXTERNEXI int rtEXIDecEventCodePart1 (OSCTXT * *pctxt*, OSINT32 * *ppart*, OSUINT32 *nbits*, OSUINT32 *maxval*)

This function decodes part 1 of an event code.

It will decode and skip undeclared items until a schema-valid event code is received.

Parameters:

pctxt Pointer to a context structure.

ppart Pointer to event code part to receive decoded value.

nbits Number of bits in the part.

maxval Maximum allowed value for this code. If the decoded value is equal to this value, it indicates that undeclared content was encountered.

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.21 EXTERNEXI int rtEXIDecFloatValue (OSCTXT * *pctxt*, OSFLOAT * *pvalue*)

This function decodes a float value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to variable to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.22 EXTERNEXI int rtEXIDecGDay (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a gDay value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.23 EXTERNEXI int rtEXIDecGDayString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a gDay value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.24 EXTERNEXI OSEXIAutomaton* rtEXIDecGetDocAutomaton (OSCTXT * *pctxt*, size_t *numGblElems*, const OSXMLFullQName * *gblElems*)

This functions returns an automaton that accepts the built-in document grammar.

This grammar is not extensible, so all the event codes created are immutable.

Parameters:

pctxt Pointer to context block structure.

Returns:

Pointer to document automaton.

5.3.1.25 EXTERNEXI OSEXIAutomaton* rtEXIDecGetElemAutomaton (OSCTXT * *pctxt*, OSXMLFullQName * *pqname*)

This function returns an automaton that accepts the built-in element grammar.

The `elementName` is associated with the automaton.

Parameters:

pctxt Pointer to context block structure.

pqname Pointer to element QName.

Returns:

Pointer to element automaton.

5.3.1.26 EXTERNEXI int rtEXIDecGetStateIndex (OSCTXT * *pctxt*, OSUINT32 *flags*)

This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

Funcion use "default" mode encoding.

Parameters:

pctxt Pointer to OSCTXT structure

flags Element flags:

- OSEXI_NO_SUBTYPES - type has no named subtypes
- OSEXI_NILLABLE - nillable element

Returns:

Index of state record matching decoded event. Negative value is an error, RTERR_IDNOTFOU is returned if no matching record is found.

5.3.1.27 EXTERNEXI int rtEXIDecGetStateIndexCompact (OSCTXT * *pctxt*, OSUINT16 *stx*, const OSEXIStateTableRecord *statetab*[ ], size_t *nrows*, OSUINT32 *flags*, OSUINT32 * *prepcnt*)

This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

Funcion use "default" mode encoding, but it is break on undeclared event.

Parameters:

pctxt Pointer to OSCTXT structure

stx Index of first record describing the state.

statetab Array of state records.

nrows Number of rows in the state table.

flags Element flags:

- OSEXI_NO_SUBTYPES - type has no named subtypes
- OSEXI_NILLABLE - nillable element Pointer to repeating element count variable.

Returns:

Index of state record matching decoded event. Negative value is an error, RTERR_IDNOTFOU is returned if no matching record is found.

5.3.1.28 EXTERNEXI int rtEXIDecGetStateIndexStrict (OSCTXT * *pctxt*, OSUINT16 *stx*, const OSEXIStateTableRecord *statetab*[ ], size_t *nrows*, OSUINT32 *flags*, OSUINT32 * *prepcnt*)

This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

Funcion use "strict" mode encoding.

Parameters:

pctxt Pointer to OSCTXT structure

stx Index of first record describing the state.

statetab Array of state records.

nrows Number of rows in the state table.

flags Element flags:

- OSEXI_NO_SUBTYPES - type has no named subtypes
- OSEXI_NILLABLE - nillable element Pointer to repeating element count variable.

Returns:

Index of state record matching decoded event. Negative value is an error, RTERR_IDNOTFOU is returned if no matching record is found.

5.3.1.29 EXTERNEXI int rtEXIDecGMonth (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a gMonth value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.30 EXTERNEXI int rtEXIDecGMonthDay (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a gMonthDay value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.31 EXTERNEXI int rtEXIDecGMonthDayString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a gMonthDay value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.32 EXTERNEXI int rtEXIDecGMonthString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a gMonth value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.33 EXTERNEXI int rtEXIDecGYear (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a gYear value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.34 EXTERNEXI int rtEXIDecGYearMonth (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a gYearMonth value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.35 EXTERNEXI int rtEXIDecGYearMonthString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a gYearMonth value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.36 EXTERNEXI int rtEXIDecGYearString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a gYear value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.37 EXTERNEXI OSBOOL rtEXIDecHasNext (OSCTXT * *pctxt*)

This function checks for additonal events in the decode stream.

Parameters:

pctxt Pointer to a context structure.

Returns:

True if there are more decoding events or false otherwise.

5.3.1.38 EXTERNEXI int rtEXIDecHeader (OSCTXT * *pctxt*)

This function decodes the EXI header at the start of a message and uses the results to set internal flags within the context structure.

Parameters:

pctxt Pointer to a context structure.

Returns:

Completion status of operation:

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

5.3.1.39 EXTERNEXI int rtEXIDecInitCompression (OSCTXT * *pctxt*, OSUINT32 *nmChannelIds*)

This function initialize decoder to use compressed or precompressed EXI stream.

Parameters:

pctxt Pointer to context block structure.

nmChannelIds Number of preallocated value channels.

Returns:

Completion status of operation:

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

5.3.1.40 EXTERNEXI int rtEXIDecInt16Value (OSCTXT * *pctxt*, OSINT16 * *pvalue*)

This function decodes a 16-bit signed integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.41 EXTERNEXI int rtEXIDecInt64Value (OSCTXT * *pctxt*, OSINT64 * *pvalue*)

This function decodes a 64-bit signed integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.42 EXTERNEXI int rtEXIDecInt8Value (OSCTXT * *pctxt*, OSINT8 * *pvalue*)

This function decodes a 8-bit signed integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.43 EXTERNEXI int rtEXIDecIntValue (OSCTXT * *pctxt*, OSINT32 * *pvalue*)

This function decodes a signed integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.44 EXTERNEXI int rtEXIDecLocalName (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppname*)

Returns the local name associated with the current event.

For event type SE and AT, it returns the local name of the element and attribute, respectively. For all other event types, the value returned by this method is undefined.

Parameters:

pctxt Pointer to a context structure.

ppname Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.45 EXTERNEXI int rtEXIDecNamespaceURI (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppNSURI*)

Returns the namespace associated with the current event.

For event type SE and AT, it returns the namespace URI of the element and attribute, respectively. For NS, it returns the URI bound to the prefix in the declaration. For all other event types, the value returned by this function is undefined.

Parameters:

pctxt Pointer to a context structure.

ppNSURI Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.46 EXTERNEXI int rtEXIDecNBitUIntValue (OSCTXT * *pctxt*, OSUINT32 * *pvalue*, OSUINT32 *nbits*)

This function decodes an unsigned integer value from a bit field of the given width.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned integer to receive decoded value.

nbits Size of bit field.

Returns:

Completion status of operation:

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

5.3.1.47 EXTERNEXI OSEXIDecStringTable* rtEXIDecNewStringTable (OSCTXT * *pctxt*, size_t *capacity*)

This function allocates and initializes a new string table structure.

Parameters:

pctxt Pointer to context block structure.

capacity Capacity of the array list or zero to use default.

Returns:

Allocated string table structure.

5.3.1.48 EXTERNEXI int rtEXIDecNextEventType (OSCTXT * *pctxt*, OSEXIEventType * *pEventType*)

Returns the next `OSEXIEventType` read by this decoder.

Attributes and namespaces are reported as separate events.

Parameters:

pctxt Pointer to a context structure.

pEventType Pointer to variable to receive decoded data.

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.49 EXTERNEXI int rtEXIDecoderInit (OSCTXT * *pctxt*)

This function initializes the decoder.

It must be called before any other decode functions when decoding a document or fragment.

Parameters:

pctxt Pointer to a context structure.

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.50 EXTERNEXI int rtEXIDecPrefix (OSCTXT * *pctxt*, const OSUTF8CHAR * *uri*, const OSUTF8CHAR ** *ppPrefix*)

Returns the namespace associated with the current event.

For event type SE and AT, it returns the namespace URI of the element and attribute, respectively. For NS, it returns the URI bound to the prefix in the declaration. For all other event types, the value returned by this function is undefined.

Parameters:

pctxt Pointer to a context structure.

uri Namespace URI that prefix is associated with.

ppPrefix Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.51 EXTERNEXI int rtEXIDecProcessingInstruction (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppTarget*, const OSUTF8CHAR ** *ppData*)

This function decodes an XML processing instruction (PI).

Parameters:

pctxt Pointer to a context structure.

ppTarget Pointer to variable to receive decoded PI target. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

ppData Pointer to variable to receive decoded PI data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.52 EXTERNEXI int rtEXIDecQName (OSCTXT * *pctxt*, OSXMLFullQName * *pqname*)

Returns the qname associated with the current event.

For SE and AT it returns the name of the element and attribute, respectively. For all other event types, the value returned by this method is undefined.

Parameters:

pctxt Pointer to a context structure.

pqname Pointer to variable to receive decoded data. Memory for the decoded name components is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.53 EXTERNEXI int rtEXIDecQNameValue (OSCTXT * *pctxt*, OSXMLFullQName * *pvalue*, const OSXMLFullQName * *pqname*)

Decode the qname value for AT and CH events.

Parameters:

pctxt Pointer to a context structure.

pvalue Pointer to variable to receive decoded data. Memory for the decoded name components is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

pqname QName the value is associated with. This should be the QName that was decoded from the stream prior to calling this function. 0 - decode xsi:type value

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.54 EXTERNEXI int rtEXIDecReset (OSCTXT * *pctxt*)

Resets the decoder for decoding a new message instance.

The decoder's state after calling this method is identical to that of a newly created instance.

Parameters:

pctxt Pointer to a context structure.

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.55 EXTERNEXI int rtEXIDecSimpleTypeEvent (OSCTXT * *pctxt*, OSINT32 * *ppart*, OSUINT32 *flags*)

This function decodes of event codes of simple type grammar.

It will decode and skip undeclared items until a schema-valid event code is received. Funcion use "default" mode encoding.

Parameters:

pctxt Pointer to a context structure.

ppart Pointer to event code part to receive decoded value:

- 0 - CH event
- 1 - EE event (empty element)
- 2 - EE event after AT(xsi:nil) event

flags Element flags:

- OSEXI_NO_SUBTYPES - type has no named subtypes
- OSEXI_NILLABLE - nillable element
- OSEXI_HAS_DEFAULT - element has no fixed or default value

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.56 EXTERNEXI int rtEXIDecSimpleTypeEventCompact (OSCTXT * *pctxt*, OSINT32 * *ppart*, OSUINT32 *flags*)

This function decodes of event codes of simple type grammar.

It will decode and skip undeclared items until a schema-valid event code is received. Funcion use "default" mode encoding, but it is break on undeclared event.

Parameters:

pctxt Pointer to a context structure.

ppart Pointer to event code part to receive decoded value:

- 0 - CH event
- 1 - EE event (empty element)
- 2 - EE event after AT(xsi:nil) event

flags Element flags:

- OSEXI_NO_SUBTYPES - type has no named subtypes
- OSEXI_NILLABLE - nillable element
- OSEXI_HAS_DEFAULT - element has no fixed or default value

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.57 EXTERNEXI int rtEXIDecSimpleTypeEventStrict (OSCTXT * *pctxt*, OSINT32 * *ppart*, OSUINT32 *flags*)

This function decodes of event codes of simple type grammar.

It will decode and skip undeclared items until a schema-valid event code is received. Funcion use "strict" mode encoding.

Parameters:

pctxt Pointer to a context structure.

ppart Pointer to event code part to receive decoded value:

- 0 - CH event
- 1 - EE event (empty element)
- 2 - EE event after AT(xsi:nil) event

flags Element flags:

- OSEXI_NO_SUBTYPES - type has no named subtypes
- OSEXI_NILLABLE - nillable element
- OSEXI_HAS_DEFAULT - element has no fixed or default value

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.58 EXTERNEXI int rtEXIDecString (OSCTXT * *pctxt*, const OSXMLFullQName * *pqname*, const OSUINT16 * *charSet*, const OSUTF8CHAR ** *ppvalue*)

Returns the value associated with the current event.

For character (CH) and comment (CM), it returns the corresponding characters. For attribute (AT) it returns the attribute value. For entity reference (ER), it returns the entity's name. For all other event types, the value returned by this method is undefined.

Parameters:

pctxt Pointer to a context structure.

pqname QName the value is associated with. This should be the QName that was decoded from the stream prior to calling this function.

charSet Pointer to restricted character set to be used for decoding the string or NULL if not restricted.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

5.3.1.59 EXTERNEXI int rtEXIDecStringLength (OSCTXT * *pctxt*)

Returns the length of the string returned by `rtEXIDecString`.

Parameters:

pctxt Pointer to a context structure.

Returns:

The length of the string or a negative status code if decoding failed.

5.3.1.60 EXTERNEXI OSUINT32 rtEXIDecStringTableAdd (OSCTXT * *pctxt*, OSEXIDecStringTable * *pstrtab*, const OSUTF8CHAR * *str*)

This function adds a string to the given string table.

The string is not added if it already exists in the table. Its compact identifier is returned in either case.

Parameters:

pctxt Pointer to context block structure.

pstrtab Pointer to string table structure.

str Pointer to string to be added to table. Memory for the string is assumed to have been allocated by the user using rtxMemAlloc. It will be freed when the table is cleared.

Returns:

Index of string in table. Note that this is an unsigned value and therefore a negative value cannot be used to signal an error condition as in other functions. In this case, the user must check the status value within the error information in the context to determine if an error occurred.

5.3.1.61 EXTERNEXI void rtEXIDecStringTableClear (OSCTXT * *pctxt*, OSEXIDecStringTable * *pstrtab*)

This function clears all strings out of the existing table.

Memory for all strings is freed using rtxMemFreePtr.

Parameters:

pctxt Pointer to context block structure.

pstrtab Pointer to string table structure.

5.3.1.62 EXTERNEXI const OSUTF8CHAR* rtEXIDecStringTableGetString (OSEXIDecStringTable * *pstrtab*, OSUINT32 *index*)

This function gets the string at the given index (i.e.

compact identifier) in the given string table.

Parameters:

pstrtab Pointer to string table structure.

index Index to string in table.

Returns:

Pointer to string in table.

5.3.1.63 EXTERNEXI void rtEXIDecStringTableInit (OSCTXT * *pctxt*, OSEXIDecStringTable * *pstrtab*, size_t *capacity*)

This function initializes the given string table structure.

Parameters:

pctxt Pointer to context block structure.

pstrtab Pointer to string table structure.

capacity Capacity of the array list or zero to use default.

5.3.1.64 EXTERNEXI int rtEXIDecStringToCharArray (OSCTXT * *pctxt*, const OSUTF8CHAR * *target*, size_t *start*, size_t *length*)

Similar to `rtEXIDecString` but the characters are copied into a fixed-size character array.

Parameters:

pctxt Pointer to a context structure.

target Destination array in which to copy the chars.

start Start index in target where the chars are copied.

length Maximum number of chars to copy.

Returns:

The actual number of chars copied into target or a negative error code if decoding failed.

5.3.1.65 EXTERNEXI OSUINT32 rtEXIDecStrTabsAddGlobalValue (OSCTXT * *pctxt*, OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *value*)

This function will add a string value to the global value string table partition.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

value Value to be added.

Returns:

Index (compact identifier) of the added value.

5.3.1.66 EXTERNEXI OSUINT32 rtEXIDecStrTabsAddLocalName (OSCTXT * *pctxt*, OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, const OSUTF8CHAR * *name*)

This function adds the given local name to the local name table partition identified by the given URI.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

uri URI identifier of local name table partition.

name Name to be added.

Returns:

Index (compact identifier) of the added value.

5.3.1.67 EXTERNEXI OSUINT32 rtEXIDecStrTabsAddLocalValue (OSCTXT * *pctxt*, OSEXIDecStringTables * *pstrtabs*, const OSXMLFullQName * *qname*, const OSUTF8CHAR * *value*)

This function adds the given local value to the local value table partition identified by the given QName.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

qname QName identifier of local value table partition.

value Value to be added.

Returns:

Index (compact identifier) of the added value.

5.3.1.68 EXTERNEXI OSUINT32 rtEXIDecStrTabsAddPrefix (OSCTXT * *pctxt*, OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, const OSUTF8CHAR * *prefix*)

This function adds the given prefix to the prefix table partition identified by the given URI.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

uri URI identifier of prefix table partition.

prefix Prefix to be added.

Returns:

Index (compact identifier) of the added value.

5.3.1.69 EXTERNEXI OSUINT32 rtEXIDecStrTabsAddURI (OSCTXT * *pctxt*, OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*)

This function will add a URI to the URI string table partition.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

uri URI to be added.

Returns:

Index (compact identifier) of the added URI.

5.3.1.70 EXTERNEXI void rtEXIDecStrTabsClear (OSCTXT * *pctxt*, OSEXIDecStringTables * *pstrtabs*)

This function clears all EXI string table partitions.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to string table set structure.

5.3.1.71 EXTERNEXI const OSUTF8CHAR* rtEXIDecStrTabsGetGlobalValue (OSEXIDecStringTables * *pstrtabs*, OSUINT32 *index*)

This function will get the compact identifier of the given string value from the global value string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

index Global value table compact identifier.

Returns:

Index (compact identifier) or the null index identifier (OSEXINULLINDEX) if not found.

5.3.1.72 EXTERNEXI OSUINT32 rtEXIDecStrTabsGetGlobalValueTableSize (OSEXIDecStringTables * *pstrtabs*)

This function returns the current number of entries in the global value string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

Returns:

Current number of entries in the partition.

5.3.1.73 EXTERNEXI const OSUTF8CHAR* rtEXIDecStrTabsGetLocalName (OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, OSUINT32 *index*)

This function will get the compact identifier of the given localName from the localName string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of localName table partition.

index Name table compact identifier.

Returns:

Index (compact identifier) of the value.

5.3.1.74 EXTERNEXI OSUINT32 rtEXIDecStrTabsGetLocalNameTableSize (OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*)

This function returns the current number of entries in the localName string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of localName table partition.

Returns:

Current number of entries in the partition.

5.3.1.75 EXTERNEXI const OSUTF8CHAR* rtEXIDecStrTabsGetLocalValue (OSEXIDecStringTables * *pstrtabs*, const OSXMLFullQName * *qname*, OSUINT32 *index*)

This function will get the compact identifier of the given local value from the local value string table partition identified by the given QName.

Parameters:

pstrtabs Pointer to full string table set structure.

qname Identifier of table partition.

index Local value table compact identifier.

Returns:

Index (compact identifier) of the value.

5.3.1.76 EXTERNEXI OSUINT32 rtEXIDecStrTabsGetLocalValueTableSize (OSEXIDecStringTables * *pstrtabs*, const OSXMLFullQName * *qname*)

This function returns the current number of entries in the local value string table partition identified by the given QName.

Parameters:

pstrtabs Pointer to full string table set structure.

qname Identifier of table partition.

Returns:

Current number of entries in the partition.

5.3.1.77 EXTERNEXI const OSUTF8CHAR* rtEXIDecStrTabsGetPrefix (OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, OSUINT32 *index*)

This function will get the compact identifier of the given prefix from the prefix string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of prefix table partition.

index Prefix table compact identifier.

Returns:

Index (compact identifier) of the value.

5.3.1.78 EXTERNEXI OSUINT32 rtEXIDecStrTabsGetPrefixTableSize (OSEXIDecStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*)

This function returns the current number of entries in the prefix string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of prefix table partition.

Returns:

Current number of entries in the partition.

5.3.1.79 EXTERNEXI const OSUTF8CHAR* rtEXIDecStrTabsGetURI (OSEXIDecStringTables * *pstrtabs*, OSUINT32 *index*)

This function will get the compact identifier of the given URI from the URI string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

index URI table compact identifier.

Returns:

Index (compact identifier) or the null index identifier (OSEXINULLINDEX) if not found.

5.3.1.80 EXTERNEXI OSUINT32 rtEXIDecStrTabsGetURITableSize (OSEXIDecStringTables * *pstrtabs*)

This function returns the current number of entries in the URI string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

Returns:

Current number of entries in the partition.

5.3.1.81 EXTERNEXI void rtEXIDecStrTabsInit (OSCTXT * *pctxt*, OSEXIDecStringTables * *pstrtabs*)

This function initializes all EXI string table partitions.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to string table set structure.

5.3.1.82 EXTERNEXI int rtEXIDecTime (OSCTXT * *pctxt*, OSNumDateTime * *pvalue*)

This function decodes a time value into a structured variable.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to structured variable to receive decoded data.

Returns:

Completion status of operation:

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

5.3.1.83 EXTERNEXI int rtEXIDecTimeString (OSCTXT * *pctxt*, const OSUTF8CHAR ** *ppvalue*)

This function decodes a time value into a string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to variable to receive decoded data. Memory for the decoded name is allocated with rtxMemAlloc. It must be freed with rtxMemFreePtr or it will be freed when all memory in the context is freed (rtxMemFreeAll or rtxFreeContext).

Returns:

Completion status of operation:

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

5.3.1.84 EXTERNEXI int rtEXIDecUInt16Value (OSCTXT * *pctxt*, OSUINT16 * *pvalue*)

This function decodes an 16-bit unsigned integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.85 EXTERNEXI int rtEXIDecUInt64Value (OSCTXT * *pctxt*, OSUINT64 * *pvalue*)

This function decodes an 64-bit unsigned integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.86 EXTERNEXI int rtEXIDecUInt8Value (OSCTXT * *pctxt*, OSUINT8 * *pvalue*)

This function decodes an 8-bit unsigned integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.87 EXTERNEXI int rtEXIDecUIntValue (OSCTXT * *pctxt*, OSUINT32 * *pvalue*)

This function decodes an unsigned integer value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to unsigned integer to receive decoded value.

Returns:

Completion status of operation:

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

5.3.1.88 EXTERNEXI int rtEXIDecUTF8Chars (OSCTXT * *pctxt*, OSUTF8CHAR ** *ppvalue*, OSUINT32 *nchars*, const OSUINT16 * *charSet*)

This function reads the given number of characters from the decode stream and creates a UTF-8 string.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to character string pointer to recive decoded string value. Memory for the string is allocated using rtxMemAlloc. It is freed using rtxMemFreePtr or when the context is freed.

nchars Number of characters to read from stream.

charSet Pointer to restricted character set to be used for decoding the string or NULL if not restricted.

Returns:

Completion status of operation:

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

5.3.1.89 EXTERNEXI int rtEXIDecUTF8Str (OSCTXT * *pctxt*, OSUTF8CHAR ** *ppvalue*, const OSUINT16 * *charSet*)

This function decodes a UTF-8 string value.

Parameters:

pctxt Pointer to context block structure.

ppvalue Pointer to character string pointer to recive decoded string value. Memory for the string is allocated using rtxMemAlloc. It is freed using rtxMemFreePtr or when the context is freed.

charSet Pointer to restricted character set to be used for decoding the string or NULL if not restricted.

Returns:

Completion status of operation:

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

5.4 EXI encode structures and functions.

Classes

- struct OSEXIEncStringTable

  This structure defines the structure of the various string table partitions used by the encoder.

- struct OSEXIEncStringTables

  This structure defines the complete set of string table partitions used by the encoder.

Functions

- EXTERNEXI int rtEXIEncAtmAddTransition (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent *pEvent, const OSEXIEventCode *pEventCode)

  This function adds a transition between two states.

- EXTERNEXI OSEXIEventCode * rtEXIEncAutomatonAdvance (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, const OSEXIEvent *pEvent, OSBOOL dynamicItems)

  This function advances the encoder automaton based on event, adding new transitions if `dynamicItems` *is set to* `true` *and either SE(*) or AT(*) are matched instead of SE(qname) or AT(qname), respectively.*

- EXTERNEXI OSEXIAutomaton * rtEXIEncGetDocAutomaton (OSCTXT *pctxt, size_t numGblElems, const OSXMLFullQName *gblElems)

  This functions returns an automaton that accepts the built-in document grammar.

- EXTERNEXI OSEXIAutomaton * rtEXIEncGetElemAutomaton (OSCTXT *pctxt, OSXMLFullQName *pqname)

  This function returns an automaton that accepts the built-in element grammar.

- EXTERNEXI void rtEXIEncStringTableInit (OSCTXT *pctxt, OSEXIEncStringTable *pstrtab, size_t capacity)

  This function initializes the given string table structure.

- EXTERNEXI OSEXIEncStringTable * rtEXIEncNewStringTable (OSCTXT *pctxt, size_t capacity)

  This function allocates and initializes a new string table structure.

- EXTERNEXI void rtEXIEncStringTableClear (OSCTXT *pctxt, OSEXIEncStringTable *pstrtab)

  This function clears all strings out of the existing table.

- EXTERNEXI OSUINT32 rtEXIEncStringTableAdd (OSCTXT *pctxt, OSEXIEncStringTable *pstrtab, const OSUTF8CHAR *str)

  This function adds a string to the given string table.

- EXTERNEXI OSUINT32 rtEXIEncStringTableGetIndex (OSEXIEncStringTable *pstrtab, const OSUTF8CHAR *str)

  This function gets the index (i.e.

- EXTERNEXI void rtEXIEncStrTabsInit (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs)

  This function initializes all EXI string table partitions.

- EXTERNEXI void rtEXIEncStrTabsClear (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs)

  This function clears all EXI string table partitions.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddURI (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function will add a URI to the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetURIID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function will get the compact identifier of the given URI from the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetURITableSize (OSEXIEncStringTables *pstrtabs)

  This function returns the current number of entries in the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddPrefix (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *prefix)

  This function adds the given prefix to the prefix table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetPrefixID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *prefix)

  This function will get the compact identifier of the given prefix from the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetPrefixTableSize (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddLocalName (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *name)

  This function adds the given local name to the local name table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalNameID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *name)

  This function will get the compact identifier of the given localName from the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalNameTableSize (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddLocalValue (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSXMLFullQName *qname, const OSUTF8CHAR *value)

  This function adds the given local value to the local value table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalValueID (OSEXIEncStringTables *pstrtabs, const OSXMLFullQName *qname, const OSUTF8CHAR *value)

  This function will get the compact identifier of the given local value from the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalValueTableSize (OSEXIEncStringTables *pstrtabs, const OSXMLFullQName *qname)

This function returns the current number of entries in the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddGlobalValue (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *value)

  This function will add a string value to the global value string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetGlobalValueID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *value)

  This function will get the compact identifier of the given string value from the global value string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetGlobalValueTableSize (OSEXIEncStringTables *pstrtabs)

  This function returns the current number of entries in the global value string table partition.

5.4.1 Function Documentation

5.4.1.1 EXTERNEXI int rtEXIEncAtmAddTransition (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, OSEXIState *fromState*, OSEXIState *toState*, const OSEXIEvent * *pEvent*, const OSEXIEventCode * *pEventCode*)

This function adds a transition between two states.

The transition is defined by a pair of states, and event and event code.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

fromState The origin state for this transition.

toState The destination state for this transition.

pEvent The event on which this transition occurs.

pEventCode The event code returned after the transition.

Returns:

Zero if operation was successful; a negative status code otherwise.

5.4.1.2 EXTERNEXI OSEXIEventCode* rtEXIEncAutomatonAdvance (OSCTXT * *pctxt*, OSEXIAutomaton * *pAutomaton*, const OSEXIEvent * *pEvent*, OSBOOL *dynamicItems*)

This function advances the encoder automaton based on event, adding new transitions if `dynamicItems` is set to `true` and either SE(*) or AT(*) are matched instead of SE(qname) or AT(qname), respectively.

Dynamic transitions for CH are also added according to the spec.

Parameters:

pctxt Pointer to context block structure.

pAutomaton Pointer to automaton structure.

pEvent Event on which to advance the automaton.

dynamicItems Flag to add dynamic transitions to the automaton.

Returns:

The event code returned as a result of this transition.

5.4.1.3 EXTERNEXI OSEXIAutomaton* rtEXIEncGetDocAutomaton (OSCTXT * *pctxt*, size_t *numGblElems*, const OSXMLFullQName * *gblElems*)

This functions returns an automaton that accepts the built-in document grammar.

This grammar is not extensible, so all the event codes created are immutable.

Parameters:

pctxt Pointer to context block structure.

numGblElems Number of global elements.

gblElems Array of global element objects.

Returns:

Pointer to document automaton.

5.4.1.4 EXTERNEXI OSEXIAutomaton* rtEXIEncGetElemAutomaton (OSCTXT * *pctxt*, OSXMLFullQName * *pqname*)

This function returns an automaton that accepts the built-in element grammar.

The `elementName` is associated with the automaton.

Parameters:

pctxt Pointer to context block structure.

pqname Pointer to element QName.

Returns:

Pointer to element automaton.

5.4.1.5 EXTERNEXI OSEXIEncStringTable* rtEXIEncNewStringTable (OSCTXT * *pctxt*, size_t *capacity*)

This function allocates and initializes a new string table structure.

Parameters:

pctxt Pointer to context block structure.

capacity Capacity of the hash map or zero to use default.

Returns:

Allocated string table structure.

5.4.1.6 EXTERNEXI OSUINT32 rtEXIEncStringTableAdd (OSCTXT * *pctxt*, OSEXIEncStringTable * *pstrtab*, const OSUTF8CHAR * *str*)

This function adds a string to the given string table.

The string is not added if it already exists in the table. Its compact identifier is returned in either case.

Parameters:

pctxt Pointer to context block structure.

pstrtab Pointer to string table structure.

str Pointer to string to be added to table. A copy of the string is not made; the pointer is simply added to the table.

Returns:

Index of string in table. Note that this is an unsigned value and therefore a negative value cannot be used to signal an error condition as in other functions. In this case, the user must check the status value within the error information in the context to determine if an error occurred.

5.4.1.7 EXTERNEXI void rtEXIEncStringTableClear (OSCTXT * *pctxt*, OSEXIEncStringTable * *pstrtab*)

This function clears all strings out of the existing table.

Parameters:

pctxt Pointer to context block structure.

pstrtab Pointer to string table structure.

5.4.1.8 EXTERNEXI OSUINT32 rtEXIEncStringTableGetIndex (OSEXIEncStringTable * *pstrtab*, const OSUTF8CHAR * *str*)

This function gets the index (i.e.

compact identifier) of a string in the given string table.

Parameters:

pstrtab Pointer to string table structure.

str Pointer to string to be added to table. A copy of the string is not made; the pointer is simply added to the table.

Returns:

Index of string in table. Note that this is an unsigned value. If the string is not found or an error occurs, the unsigned integer max value is returned (OSUINT32_MAX).

5.4.1.9 EXTERNEXI void rtEXIEncStringTableInit (OSCTXT * *pctxt*, OSEXIEncStringTable * *pstrtab*, size_t *capacity*)

This function initializes the given string table structure.

Parameters:

pctxt Pointer to context block structure.

pstrtab Pointer to string table structure.

capacity Capacity of the hash map or zero to use default.

5.4.1.10 EXTERNEXI OSUINT32 rtEXIEncStrTabsAddGlobalValue (OSCTXT * *pctxt*, OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *value*)

This function will add a string value to the global value string table partition.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

value Value to be added.

Returns:

Index (compact identifier) of the added value.

5.4.1.11 EXTERNEXI OSUINT32 rtEXIEncStrTabsAddLocalName (OSCTXT * *pctxt*, OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, const OSUTF8CHAR * *name*)

This function adds the given local name to the local name table partition identified by the given URI.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

uri URI identifier of local name table partition.

name Name to be added.

Returns:

Index (compact identifier) of the added value.

5.4.1.12 EXTERNEXI OSUINT32 rtEXIEncStrTabsAddLocalValue (OSCTXT * *pctxt*, OSEXIEncStringTables * *pstrtabs*, const OSXMLFullQName * *qname*, const OSUTF8CHAR * *value*)

This function adds the given local value to the local value table partition identified by the given QName.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

qname QName identifier of local value table partition.

value Value to be added.

Returns:

Index (compact identifier) of the added value.

5.4.1.13 EXTERNEXI OSUINT32 rtEXIEncStrTabsAddPrefix (OSCTXT * *pctxt*, OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, const OSUTF8CHAR * *prefix*)

This function adds the given prefix to the prefix table partition identified by the given URI.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

uri URI identifier of prefix table partition.

prefix Prefix to be added.

Returns:

Index (compact identifier) of the added value.

5.4.1.14 EXTERNEXI OSUINT32 rtEXIEncStrTabsAddURI (OSCTXT * *pctxt*, OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*)

This function will add a URI to the URI string table partition.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to full string table set structure.

uri URI to be added.

Returns:

Index (compact identifier) of the added URI.

5.4.1.15 EXTERNEXI void rtEXIEncStrTabsClear (OSCTXT * *pctxt*, OSEXIEncStringTables * *pstrtabs*)

This function clears all EXI string table partitions.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to string table set structure.

5.4.1.16 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetGlobalValueID (OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *value*)

This function will get the compact identifier of the given string value from the global value string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

value Value to be looked up.

Returns:

Index (compact identifier) or the null index identifier (OSEXINULLINDEX) if not found.

5.4.1.17 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetGlobalValueTableSize (OSEXIEncStringTables * *pstrtabs*)

This function returns the current number of entries in the global value string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

Returns:

Current number of entries in the partition.

5.4.1.18 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalNameID (OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, const OSUTF8CHAR * *name*)

This function will get the compact identifier of the given localName from the localName string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of localName table partition.

name Name to be looked up.

Returns:

Index (compact identifier) of the value.

5.4.1.19 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalNameTableSize (OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*)

This function returns the current number of entries in the localName string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of localName table partition.

Returns:

Current number of entries in the partition.

5.4.1.20 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalValueID (OSEXIEncStringTables * *pstrtabs*, const OSXMLFullQName * *qname*, const OSUTF8CHAR * *value*)

This function will get the compact identifier of the given local value from the local value string table partition identified by the given QName.

Parameters:

pstrtabs Pointer to full string table set structure.

qname Identifier of table partition.

value Value to be looked up.

Returns:

Index (compact identifier) of the value.

5.4.1.21 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalValueTableSize (OSEXIEncStringTables * *pstrtabs*, const OSXMLFullQName * *qname*)

This function returns the current number of entries in the local value string table partition identified by the given QName.

Parameters:

pstrtabs Pointer to full string table set structure.

qname Identifier of table partition.

Returns:

Current number of entries in the partition.

5.4.1.22 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetPrefixID (OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*, const OSUTF8CHAR * *prefix*)

This function will get the compact identifier of the given prefix from the prefix string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of prefix table partition.

prefix Prefix to be looked up.

Returns:

Index (compact identifier) of the value.

5.4.1.23 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetPrefixTableSize (OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*)

This function returns the current number of entries in the prefix string table partition identified by the given URI.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI identifier of prefix table partition.

Returns:

Current number of entries in the partition.

5.4.1.24 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetURIID (OSEXIEncStringTables * *pstrtabs*, const OSUTF8CHAR * *uri*)

This function will get the compact identifier of the given URI from the URI string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

uri URI to be looked up.

Returns:

Index (compact identifier) or the null index identifier (OSEXINULLINDEX) if not found.

5.4.1.25 EXTERNEXI OSUINT32 rtEXIEncStrTabsGetURITableSize (OSEXIEncStringTables * *pstrtabs*)

This function returns the current number of entries in the URI string table partition.

Parameters:

pstrtabs Pointer to full string table set structure.

Returns:

Current number of entries in the partition.

5.4.1.26 EXTERNEXI void rtEXIEncStrTabsInit (OSCTXT * *pctxt*, OSEXIEncStringTables * *pstrtabs*)

This function initializes all EXI string table partitions.

Parameters:

pctxt Pointer to context block structure.

pstrtabs Pointer to string table set structure.

5.5 EXI event code definitions and functions.

Classes

- struct OSEXIEventCode

  A structure representing a production's event code.

- struct OSEXIEventCodeGroup

  An EventCodeGroup is a group of related event codes.

Defines

- #define rtEXISetEventCode1(pEventCode, part1) rtEXISetEventCode3(pEventCode, part1, OSINT32_MIN, OSINT32_MIN)

  This macro sets a one-part event code.

- #define rtEXISetEventCode2(pEventCode, part1, part2) rtEXISetEventCode3(pEventCode, part1, part2, OSINT32_MIN)

  This macro sets a two-part event code.

- #define rtEXIEventCodeGroupHasPart1(pecgrp, part1) rtxDynBitSetTestBit(&pecgrp → part1Set,part1)

  This macro returns true if there is an event code in this group whose length is 1 and whose first part is equal to part1.

- #define rtEXIEventCodeGroupHasPart2(pecgrp, part2) rtxDynBitSetTestBit(&pecgrp → part2Set,part2)

  This macro returns true if there is an event code in this group whose length is 2 and whose first part is equal to part2.

Functions

- EXTERNEXI int rtEXIEventCodeCompare (const OSEXIEventCode *pec1, const OSEXIEventCode *pec2)

  This function compares two event codes.

- EXTERNEXI OSEXIEventCode * rtEXIEventCodeCopy (OSCTXT *pctxt, const OSEXIEventCode *pec)

  This function does a deep-copy of an event code structure.

- EXTERNEXI OSBOOL rtEXIEventCodesEqual (const OSEXIEventCode *pec1, const OSEXIEventCode *pec2)

  This function compares two event codes for equality.

- EXTERNEXI char * rtEXIEventCodeToString (OSCTXT *pctxt, const OSEXIEventCode *pec)

  This function returns a string representation of the given event code in dot notation (part1.part2.part3).

- EXTERNEXI OSUINT32 rtEXIEventCodeLength (const OSEXIEventCode *pec)

  This function returns the length of the given event code.

- EXTERNEXI OSEXIEventCode * rtEXINewEventCode1 (OSCTXT *pctxt, OSINT32 part1)

  This function allocates and initializes a one-part event code.

- EXTERNEXI OSEXIEventCode * rtEXINewEventCode2 (OSCTXT *pctxt, OSINT32 part1, OSINT32 part2)

  This function allocates and initializes a two-part event code.

- EXTERNEXI OSEXIEventCode * rtEXINewEventCode3 (OSCTXT *pctxt, OSINT32 part1, OSINT32 part2, OSINT32 part3)

  This function allocates and initializes a three-part event code.

- EXTERNEXI void rtEXISetEventCode3 (OSEXIEventCode *pEventCode, OSINT32 part1, OSINT32 part2, OSINT32 part3)

  This function sets a three-part event code.

- EXTERNEXI void rtEXIEventCodePrint (const OSEXIEventCode *pEventCode)

  This function prints information on the given event code to stdout.

- EXTERNEXI void rtEXIEventCodeGroupInit (OSCTXT *pctxt, OSEXIEventCodeGroup *pecgrp)

  This function initializes an event code group structure.

- EXTERNEXI OSEXIEventCodeGroup * rtEXINewEventCodeGroup (OSCTXT *pctxt)

  This function allocates and initializes a new event code group structure.

- EXTERNEXI int rtEXIEventCodeGroupAdd (OSEXIEventCodeGroup *pecgrp, const OSEXIEventCode *pec)

  This function adds an event code to an event code group and updates the maximum part variables.

- EXTERNEXI OSEXIEventCodeGroup * rtEXIEventCodeGroupCopy (const OSEXIEventCodeGroup *pecgrp)

  This function performs a deep copy of the given event group.

- EXTERNEXI void rtEXIEventCodeGroupFreeMem (OSEXIEventCodeGroup *pecgrp)

  This function frees all memory associated with an event group.

- EXTERNEXI int rtEXIEventCodeGroupGetBitsPart1 (OSEXIEventCodeGroup *pecgrp)

  This function returns the number of bits necessary to encode the max part1 value in this group.

- EXTERNEXI int rtEXIEventCodeGroupGetBitsPart2 (OSEXIEventCodeGroup *pecgrp)

  This function returns the number of bits necessary to encode the max part2 value in this group.

- EXTERNEXI int rtEXIEventCodeGroupGetBitsPart3 (OSEXIEventCodeGroup *pecgrp)

  This function returns the number of bits necessary to encode the max part3 value in this group.

- EXTERNEXI void rtEXIEventCodeGroupIncrPart1 (OSCTXT *pctxt, OSEXIEventCodeGroup *pecgrp)

  This function increments part1 in all event codes in this group, as well as the max part1 value.

5.5.1 Define Documentation

5.5.1.1 #define rtEXISetEventCode1(pEventCode, part1) rtEXISetEventCode3(pEventCode, part1, OSINT32_MIN, OSINT32_MIN)

This macro sets a one-part event code.

Parameters:

pEventCode Pointer to event code structure.

part1 Part 1 of the event code.

Definition at line 172 of file rtEXIEventCode.h.

5.5.1.2 #define rtEXISetEventCode2(pEventCode, part1, part2) rtEXISetEventCode3(pEventCode, part1, part2, OSINT32_MIN)

This macro sets a two-part event code.

Parameters:

pEventCode Pointer to event code structure.

part1 Part 1 of the event code.

part2 Part 2 of the event code.

Definition at line 182 of file rtEXIEventCode.h.

5.5.2 Function Documentation

5.5.2.1 EXTERNEXI int rtEXIEventCodeCompare (const OSEXIEventCode * *pec1*, const OSEXIEventCode * *pec2*)

This function compares two event codes.

Event codes are first compared based on length and then partwise.

Parameters:

pec1 Pointer to event code to compare.

pec2 Pointer to event code to compare.

Returns:

Comparison result: < 0 = (ec1 $<$ ec2), 0 = (ec1 == ec2), > 0 = (ec1 $>$ ec2)

5.5.2.2 EXTERNEXI OSEXIEventCode* rtEXIEventCodeCopy (OSCTXT * *pctxt*, const OSEXIEventCode * *pec*)

This function does a deep-copy of an event code structure.

Parameters:

pctxt Pointer to context block structure.

pec Pointer to event code to copy.

Returns:

Copied structure. Memory for the new structure is allocated with the rtxMemAlloc run-time functions and thus must be freed with an rtxMemFree* function. If memory allocation fails, NULL is returned.

5.5.2.3 EXTERNEXI int rtEXIEventCodeGroupAdd (OSEXIEventCodeGroup * *pecgrp*, const OSEXIEventCode * *pec*)

This function adds an event code to an event code group and updates the maximum part variables.

Parameters:

pecgrp Pointer to event code group structure.

pec Pointer to event code to add.

Returns:

Status of operation: 0 if success, negative status code if failure.

5.5.2.4 EXTERNEXI OSEXIEventCodeGroup* rtEXIEventCodeGroupCopy (const OSEXIEventCodeGroup * *pecgrp*)

This function performs a deep copy of the given event group.

This includes copying the group list as well as all its members.

Parameters:

pecgrp Pointer to event code group structure.

Returns:

Copied group structure. Memory for the new structure is allocated with the rtxMemAlloc run-time functions and thus must be freed with an rtxMemFree* function. If memory allocation fails, NULL is returned.

5.5.2.5 EXTERNEXI void rtEXIEventCodeGroupFreeMem (OSEXIEventCodeGroup * *pecgrp*)

This function frees all memory associated with an event group.

Parameters:

pecgrp Pointer to event code group structure.

5.5.2.6 EXTERNEXI int rtEXIEventCodeGroupGetBitsPart1 (OSEXIEventCodeGroup * *pecgrp*)

This function returns the number of bits necessary to encode the max part1 value in this group.

Parameters:

pecgrp Pointer to event code group structure.

Returns:

Number of bits.

5.5.2.7 EXTERNEXI int rtEXIEventCodeGroupGetBitsPart2 (OSEXIEventCodeGroup * *pecgrp*)

This function returns the number of bits necessary to encode the max part2 value in this group.

Parameters:

pecgrp Pointer to event code group structure.

Returns:

Number of bits.

5.5.2.8 EXTERNEXI int rtEXIEventCodeGroupGetBitsPart3 (OSEXIEventCodeGroup * *pecgrp*)

This function returns the number of bits necessary to encode the max part3 value in this group.

Parameters:

pecgrp Pointer to event code group structure.

Returns:

Number of bits.

5.5.2.9 EXTERNEXI void rtEXIEventCodeGroupIncrPart1 (OSCTXT * *pctxt*, OSEXIEventCodeGroup * *pecgrp*)

This function increments part1 in all event codes in this group, as well as the max part1 value.

Parameters:

pctxt Pointer to a context block structure.
pecgrp Pointer to event code group structure.

5.5.2.10 EXTERNEXI void rtEXIEventCodeGroupInit (OSCTXT * *pctxt*, OSEXIEventCodeGroup * *pecgrp*)

This function initializes an event code group structure.

Parameters:

pctxt Pointer to a context block structure.
pecgrp Pointer to event code group structure.

5.5.2.11 EXTERNEXI OSUINT32 rtEXIEventCodeLength (const OSEXIEventCode * *pec*)

This function returns the length of the given event code.

Parameters:

pec Pointer to event code.

Returns:

Length of event code (0 - 3)

5.5.2.12 EXTERNEXI void rtEXIEventCodePrint (const OSEXIEventCode * *pEventCode*)

This function prints information on the given event code to stdout.

Parameters:

pEventCode Pointer to event code structure.

5.5.2.13 EXTERNEXI OSBOOL rtEXIEventCodesEqual (const OSEXIEventCode * *pec1*, const OSEXIEventCode * *pec2*)

This function compares two event codes for equality.

A deep compare is done (i.e. will return TRUE if either the pointers are equal or all parts inside are equal).

Parameters:

pec1 Pointer to event code to compare.

pec2 Pointer to event code to compare.

Returns:

Boolean comparison result.

5.5.2.14 EXTERNEXI char* rtEXIEventCodeToString (OSCTXT * *pctxt*, const OSEXIEventCode * *pec*)

This function returns a string representation of the given event code in dot notation (part1.part2.part3).

Memory is allocated for the string using the rtxMemAlloc run-time function and must be freed using one the rtxMemFree functions.

Parameters:

pctxt Pointer to context block structure.

pec Pointer to event code.

Returns:

Pointer to string version of code or NULL if no dynamic memory is available.

5.5.2.15 EXTERNEXI OSEXIEventCode* rtEXINewEventCode1 (OSCTXT * *pctxt*, OSINT32 *part1*)

This function allocates and initializes a one-part event code.

Parameters:

pctxt Pointer to context block structure.

part1 Part 1 of the event code.

Returns:

Pointer to allocated event code structure. NULL if no dynamic memory available.

5.5.2.16 EXTERNEXI OSEXIEventCode* rtEXINewEventCode2 (OSCTXT * *pctxt*, OSINT32 *part1*, OSINT32 *part2*)

This function allocates and initializes a two-part event code.

Parameters:

pctxt Pointer to context block structure.

part1 Part 1 of the event code.

part2 Part 2 of the event code.

Returns:

Pointer to allocated event code structure. NULL if no dynamic memory available.

5.5.2.17 EXTERNEXI OSEXIEventCode* rtEXINewEventCode3 (OSCTXT * *pctxt*, OSINT32 *part1*, OSINT32 *part2*, OSINT32 *part3*)

This function allocates and initializes a three-part event code.

Parameters:

pctxt Pointer to context block structure.

part1 Part 1 of the event code.

part2 Part 2 of the event code.

part3 Part 3 of the event code.

Returns:

Pointer to allocated event code structure. NULL if no dynamic memory available.

5.5.2.18 EXTERNEXI OSEXIEventCodeGroup* rtEXINewEventCodeGroup (OSCTXT * *pctxt*)

This function allocates and initializes a new event code group structure.

Parameters:

pctxt Pointer to a context block structure.

Returns:

Pointer to new event code group structure or NULL if no dynamic memory available.

5.5.2.19 EXTERNEXI void rtEXISetEventCode3 (OSEXIEventCode * *pEventCode*, OSINT32 *part1*, OSINT32 *part2*, OSINT32 *part3*)

This function sets a three-part event code.

Parameters:

pEventCode Pointer to event code structure.

part1 Part 1 of the event code.

part2 Part 2 of the event code.

part3 Part 3 of the event code.

5.6 XML to EXI serialization functions.

Functions

- EXTERNEXI int rtXmlMem2ExiMem (OSCTXT *pctxt, OSOCTET *outbuf, size_t outbufsiz)

  This function serializes XML data to EXI data and stores the result in the given memory buffer.

- EXTERNEXI int rtXmlFile2ExiStream (const char *xmlFileName, OSCTXT *pExiCtxt)

  This function serializes XML data from a file to EXI data and outputs the result to the output stream defined within the given output context.

- EXTERNEXI int xml2ExiStartElement (void *userData, const OSUTF8CHAR *localname, const OSUTF8CHAR *qname, const OSUTF8CHAR *const *attrs)

  SAX start element handler callback function.

- EXTERNEXI int xml2ExiCharacters (void *userData, const OSUTF8CHAR *chars, int length)

  SAX characters handler callback function definition.

- EXTERNEXI int xml2ExiEndElement (void *userData, const OSUTF8CHAR *localname, const OSUTF8CHAR *qname)

  SAX end element handler callback function definition.

5.6.1 Function Documentation

5.6.1.1 EXTERNEXI int rtXmlFile2ExiStream (const char * *xmlFileName*, OSCTXT * *pExiCtxt*)

This function serializes XML data from a file to EXI data and outputs the result to the output stream defined within the given output context.

The input context structure (pctxt) is assumed to contain an encoded XML message in its internal buffer. This will be the case after encoding an XML instance.

Parameters:

xmlFileName Full path to a file containing an XML document.

pExiCtxt Pointer to an output context assumed to contain an output stream descriptor created with an rtxStreamCreateWriter function.

Returns:

Completion status of operation:

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

5.6.1.2 EXTERNEXI int rtXmlMem2ExiMem (OSCTXT * *pctxt*, OSOCTET * *outbuf*, size_t *outbufsiz*)

This function serializes XML data to EXI data and stores the result in the given memory buffer.

The context structure is assumed to contain an encoded XML message in its internal buffer. This will be the case after encoding an XML instance.

Parameters:

pctxt Pointer to a context structure. This is assumed to hold an encoded XML document or fragment.

outbuf Pointer to output buffer to receive EXI data.

outbufsiz Size of static output buffer.

Returns:

Completion status of operation:

- positive value indicates success and is encoded message length.
- negative return value is error.

5.6.1.3 EXTERNEXI int xml2ExiCharacters (void * *userData*, const OSUTF8CHAR * *chars*, int *length*)

SAX characters handler callback function definition.

Parameters:

userData User-defined data structure. In this case, the XML2EXISAXUserData defined above is used.

chars Character data. String is not null-terminated.

length Number of characters in character string.

Returns:

Status of operation.

5.6.1.4 EXTERNEXI int xml2ExiEndElement (void * *userData*, const OSUTF8CHAR * *localname*, const OSUTF8CHAR * *qname*)

SAX end element handler callback function definition.

Parameters:

userData User-defined data structure. In this case, the XML2EXISAXUserData defined above is used.

localname Local name of element.

qname Qualified name of element.

Returns:

Status of operation.

5.6.1.5 EXTERNEXI int xml2ExiStartElement (void * *userData*, const OSUTF8CHAR * *localname*, const OSUTF8CHAR * *qname*, const OSUTF8CHAR *const * *attrs*)

SAX start element handler callback function.

Parameters:

userData User-defined data structure. In this case, the XML2EXISAXUserData defined above is used.

localname Local name of element.

qname Qualified name of element.

attrs Array of attribute name/value pairs, terminated by 0;

Returns:

Status of operation.

Chapter 6

Class Documentation

6.1 OSEXIAtmState Struct Reference

This structure defines state information from an automaton that must be preserved at each stack level.

`#include <rtEXIAutomaton.h>`

Public Attributes

- OSEXIState currentState

  The automaton's current state.

6.1.1 Detailed Description

This structure defines state information from an automaton that must be preserved at each stack level.

Definition at line 143 of file rtEXIAutomaton.h.

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

- rtEXIAutomaton.h

6.2 OSEXIAutomaton Struct Reference

This structure defines a finite state automata for EXI grammars.

`#include <rtEXIAutomaton.h>`

Public Attributes

- OSEXIState numberOfStates

  The number of states in this automaton.

- OSEXIState currentState

  The automaton's current state.

- OSEXIState acceptingState

  The automaton's accepting state.

- const OSXMLFullQName * elementName

  Name of the element's grammar this automaton accepts, or `null` *if unspecified.*

- OSRTArrayList eventCodeGroups

  List of event code groups indexed by state.

- OSBOOL isClosed

  Indicates if this automaton can be extended by adding additional states or transitions.

- OSBOOL matchedBaseEvent

  Flag indicating if the last call to rtEXIDecAutomatonAdvance matched a base event instead.

- OSRTArrayList eventStates

  Mapping defining transitions between (from, event) into (to, eventcode) pairs.

- OSEXIEvent * pDynEvent

  Event of the form SE(null) or AT(null) used in the last dynamic transition added to this automaton in rtEXIDecAutomatonAdvance.

- OSUINT32 valueChannel

  Value channel index.

- OSEXIStateEvent * pAttrEventState

  Attribute value state.

- OSEXIState prevState

  Last state before AT(*) event.

6.2.1 Detailed Description

This structure defines a finite state automata for EXI grammars.

Definition at line 61 of file rtEXIAutomaton.h.

6.2.2 Member Data Documentation

6.2.2.1 OSRTArrayList OSEXIAutomaton::eventCodeGroups

List of event code groups indexed by state.

Event code groups are used to calculate the number of bits needed to encode an event code.

Definition at line 88 of file rtEXIAutomaton.h.

6.2.2.2 OSBOOL OSEXIAutomaton::isClosed

Indicates if this automaton can be extended by adding additional states or transitions.

An automaton that has been *closed* cannot be further extended.

Definition at line 95 of file rtEXIAutomaton.h.

6.2.2.3 OSBOOL OSEXIAutomaton::matchedBaseEvent

Flag indicating if the last call to rtEXIDecAutomatonAdvance matched a base event instead.

The base event of SE(qname) is SE(∗); the base event of AT(qname) is AT(∗).

Definition at line 102 of file rtEXIAutomaton.h.

6.2.2.4 OSRTArrayList OSEXIAutomaton::eventStates

Mapping defining transitions between (from, event) into (to, eventcode) pairs.

These records are maintained in a flat array list of all possible combinations. New relations can be added as knowledge is learned in an EXI encoding.

Definition at line 110 of file rtEXIAutomaton.h.

6.2.2.5 OSEXIEvent∗ OSEXIAutomaton::pDynEvent

Event of the form SE(null) or AT(null) used in the last dynamic transition added to this automaton in rtEXIDecAutomatonAdvance.

This event is returned by calling rtEXIGetDynamicEvent. The decoder is responsible for filling in the qname in this event, as it isn't available when the dynamic production is added.

Definition at line 120 of file rtEXIAutomaton.h.

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

- rtEXIAutomaton.h

6.3 OSEXIDecStringTable Struct Reference

This structure defines the structure of the various string table partitions used by the decoder.

`#include <rtEXIDecStringTable.h>`

Public Attributes

- OSRTArrayList records

  An expandable array relating string compact identifer (index) values to strings.

6.3.1 Detailed Description

This structure defines the structure of the various string table partitions used by the decoder.

In this case, an array list can be used for record storage because the compact identifiers are simply indexes into the array. A different structure using a has map is used by the encoder.

Definition at line 49 of file rtEXIDecStringTable.h.

6.3.2 Member Data Documentation

6.3.2.1 OSRTArrayList OSEXIDecStringTable::records

An expandable array relating string compact identifer (index) values to strings.

The array contains pointers to UTF8 strings (const OSUTF8CHAR∗).

Note that a count variable is not maintained in this structure as it is for the encode string table. That is because count can be obtained from the OSRTArrayList structure (records.size).

Definition at line 59 of file rtEXIDecStringTable.h.

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

- rtEXIDecStringTable.h

6.4 OSEXIDecStringTables Struct Reference

This structure defines the complete set of string table partitions used by the decoder.

`#include <rtEXIDecStringTables.h>`

Public Attributes

- OSEXIDecStringTable uriTable
  The URI table.
- OSRTHashMap prefixTables
  The prefix table set.
- OSRTHashMap localNameTables
  The local name table set.
- OSRTHashMap localValueTables
  The local value table set.
- OSEXIDecStringTable globalValueTable
  The global value table.

6.4.1 Detailed Description

This structure defines the complete set of string table partitions used by the decoder.

A different structure is used by the decoder.

Definition at line 46 of file rtEXIDecStringTables.h.

6.4.2 Member Data Documentation

6.4.2.1 OSEXIDecStringTable OSEXIDecStringTables::uriTable

The URI table.

This holds URI content items. There is one URI table for a given EXI application.

Definition at line 51 of file rtEXIDecStringTables.h.

6.4.2.2 OSRTHashMap OSEXIDecStringTables::prefixTables

The prefix table set.

There is a prefix table (referred to in the spec as a 'table partition') for each namespace URI. These are represented using a HashMap with URI as key and StringTable as value.

Definition at line 59 of file rtEXIDecStringTables.h.

6.4.2.3 OSRTHashMap OSEXIDecStringTables::localNameTables

The local name table set.

There is a local name table (referred to in the spec as a 'table partition') for each namespace URI. These are represented using a HashMap with URI as key and StringTable as value.

Definition at line 67 of file rtEXIDecStringTables.h.

6.4.2.4 OSRTHashMap OSEXIDecStringTables::localValueTables

The local value table set.

This set of tables holds Value content items. They are partitioned based on qname of their associated attribute or element definitions.

Definition at line 74 of file rtEXIDecStringTables.h.

6.4.2.5 OSEXIDecStringTable OSEXIDecStringTables::globalValueTable

The global value table.

This holds Value content items. There is one global value table for a given EXI application.

Definition at line 80 of file rtEXIDecStringTables.h.

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

- rtEXIDecStringTables.h

6.5 OSEXIEncStringTable Struct Reference

This structure defines the structure of the various string table partitions used by the encoder.

`#include <rtEXIEncStringTable.h>`

Public Attributes

- OSUINT32 count

  The count of the number of items in the table.

- OSRTHashMapStr2UInt records

  A hash map relating string to compact identifer (index) values.

6.5.1 Detailed Description

This structure defines the structure of the various string table partitions used by the encoder.

A different structure is used by the decoder.

Definition at line 47 of file rtEXIEncStringTable.h.

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

- rtEXIEncStringTable.h

6.6 OSEXIEncStringTables Struct Reference

This structure defines the complete set of string table partitions used by the encoder.

`#include <rtEXIEncStringTables.h>`

Public Attributes

- OSEXIEncStringTable uriTable
  The URI table.
- OSRTHashMap prefixTables
  The prefix table set.
- OSRTHashMap localNameTables
  The local name table set.
- OSRTHashMap localValueTables
  The local value table set.
- OSEXIEncStringTable globalValueTable
  The global value table.

6.6.1 Detailed Description

This structure defines the complete set of string table partitions used by the encoder.

A different structure is used by the decoder.

Definition at line 46 of file rtEXIEncStringTables.h.

6.6.2 Member Data Documentation

6.6.2.1 OSEXIEncStringTable OSEXIEncStringTables::uriTable

The URI table.

This holds URI content items. There is one URI table for a given EXI application.

Definition at line 51 of file rtEXIEncStringTables.h.

6.6.2.2 OSRTHashMap OSEXIEncStringTables::prefixTables

The prefix table set.

There is a prefix table (referred to in the spec as a 'table partition') for each namespace URI. These are represented using a HashMap with URI as key and StringTable as value.

Definition at line 59 of file rtEXIEncStringTables.h.

6.6.2.3 OSRTHashMap OSEXIEncStringTables::localNameTables

The local name table set.

There is a local name table (referred to in the spec as a 'table partition') for each namespace URI. These are represented using a HashMap with URI as key and StringTable as value.

Definition at line 67 of file rtEXIEncStringTables.h.

6.6.2.4 OSRTHashMap OSEXIEncStringTables::localValueTables

The local value table set.

This set of tables holds Value content items. They are partitioned based on qname of their associated attribute or element definitions.

Definition at line 74 of file rtEXIEncStringTables.h.

6.6.2.5 OSEXIEncStringTable OSEXIEncStringTables::globalValueTable

The global value table.

This holds Value content items. There is one global value table for a given EXI application.

Definition at line 80 of file rtEXIEncStringTables.h.

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

- rtEXIEncStringTables.h

6.7 OSEXIEventCode Struct Reference

A structure representing a production's event code.

`#include <rtEXIEventCode.h>`

6.7.1 Detailed Description

A structure representing a production's event code.

An event code has at least one part and at most three parts. Each part is represented by an integer number. A part that is absent is represented by the minimum integer value.

Definition at line 47 of file rtEXIEventCode.h.

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

- rtEXIEventCode.h

6.8 OSEXIEventCodeGroup Struct Reference

An EventCodeGroup is a group of related event codes.

`#include <rtEXIEventCodeGroup.h>`

6.8.1 Detailed Description

An EventCodeGroup is a group of related event codes.

Event codes are related if they all correspond to grammar productions with the same LHS non-terminal.

Definition at line 47 of file rtEXIEventCodeGroup.h.

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

- rtEXIEventCodeGroup.h

6.9 OSEXIStateEvent Struct Reference

This structure holds state/event information.

`#include <rtEXIAutomaton.h>`

6.9.1 Detailed Description

This structure holds state/event information.

Definition at line 50 of file rtEXIAutomaton.h.

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

- rtEXIAutomaton.h

Chapter 7

File Documentation

7.1 osrtexi.h File Reference

EXI low-level C context and structure definitions.

```
#include "rtxsrc/rtxContext.h"
#include "rtxsrc/rtxStack.h"
#include "rtexisrc/rtEXIEncAutomaton.h"
#include "rtexisrc/rtEXIEncStringTables.h"
#include "rtexisrc/rtEXIDecStringTables.h"
#include "rtexisrc/rtEXIStateTable.h"
#include "rtxsrc/rtxDiagBitTrace.h"
```

Defines

- #define rtEXISetBufPtr(pctxt, bufaddr, bufsiz) rtxCtxtSetBufPtr (pctxt, bufaddr, bufsiz)

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

- #define rtEXIGetMsgPtr(pctxt) (pctxt) → buffer.data

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

- #define rtEXIGetMsgLen(pctxt) (pctxt) → buffer.byteIndex

  This macro returns the length of the encoded XML message.

- #define rtEXISetOption(pctxt, option) EXICTXT(pctxt) → options |= option;

  This macro is used to set a bit flag in the EXI options bit mask.

- #define rtEXIClearOption(pctxt, option) EXICTXT(pctxt) → options &= ∼option;

  This macro is used to clear a bit flag in the EXI options bit mask.

- #define rtEXITestOption(pctxt, option) ((EXICTXT(pctxt) → options & option) != 0)

  This macro tests if the given option is set in the EXI context.

- #define rtEXISetValueChannelId(pctxt, channelId) EXICTXT(pctxt) → curChannelId = channelId

This macro sets the compression channel ID in the EXI context.

- #define rtEXISetFragmentLevel(pctxt, fraglevel) EXICTXT(pctxt) → selfContainedLevel = fraglevel

  This macro sets level of elements, that encoded as self-contained fragment.

- #define rtEXISetFragmentElement(pctxt, fragelement) EXICTXT(pctxt) → selfContainedElemName = (const OSUTF8CHAR∗) (fragelement)

  This macro sets name of element, that encoded as self-contained fragment.

Functions

- EXTERNEXI int rtEXIInitContext (OSCTXT ∗pctxt)

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

- EXTERNEXI int rtEXIInitCtxtAppInfo (OSCTXT ∗pctxt)

  This function initializes the EXI application info section of the given context.

- EXTERNEXI int rtEXICtxtSetStateTable (OSCTXT ∗pctxt, const OSEXIStateTableRecord ∗statetab, OSINT16 nrows)

  This function will set the state table in the EXI context to the given value if it is not already set.

- EXTERNEXI void rtEXIEnableBitFieldTrace (OSCTXT ∗pctxt)

  This function turns on bit field tracing and initializes the bit field trace list.

- EXTERNEXI int rtEXISetUriPrefix (OSCTXT ∗pctxt, const OSUTF8CHAR ∗prefix, const OSUTF8CHAR ∗uri)

  This function assign URI prefix for EXI decoding.

7.1.1 Detailed Description

EXI low-level C context and structure definitions.

Definition in file osrtexi.h.

7.2 rtEXI2SAX.h File Reference

EXI SAX parser interface.

```
#include "rtxsrc/rtxStack.h"
#include "rtxsrc/rtxXmlQName.h"
#include "rtexisrc/rtEXIExternDefs.h"
```

Typedefs

- typedef int(∗ StartElementHandler)(void ∗userData, const OSUTF8CHAR ∗localname, const OSUTF8CHAR ∗qname, const OSUTF8CHAR ∗const ∗attrs)

  SAX start element handler callback function definition.

- typedef int(∗ EndElementHandler)(void ∗userData, const OSUTF8CHAR ∗localname, const OSUTF8CHAR ∗qname)

  SAX end element handler callback function definition.

- typedef int(∗ CharacterDataHandler)(void ∗userData, const OSUTF8CHAR ∗value, int len)

  SAX characters handler callback function definition.

Functions

- EXTERNEXI EXI2SAXReader ∗ rtEXI2SAXCreateReader (OSCTXT ∗pctxt, void ∗pUserData, StartElementHandler startElemFunc, EndElementHandler endElemFunc, CharacterDataHandler charactersFunc)

  This function is used to create a reader structure for use by the main SAX parser function.

- EXTERNEXI int rtEXI2SAXParse (EXI2SAXReader ∗pReader)

  This is the main SAX parser function.

7.2.1 Detailed Description

EXI SAX parser interface.

Definition in file rtEXI2SAX.h.

7.3 rtEXI2XML.h File Reference

Functions for converting EXI encoded data into XML format.

```
#include "rtexisrc/osrtexi.h"
```

Functions

- EXTERNEXI int rtExi2XmlStream (OSCTXT *pctxt, OSCTXT *poutctxt)

  This function transcodes an EXI encoded message to an XML output stream.

7.3.1 Detailed Description

Functions for converting EXI encoded data into XML format.

Definition in file rtEXI2XML.h.

7.4 rtEXIAutomaton.h File Reference

EXI automaton structure and functions.

```
#include "rtexisrc/rtEXIEvent.h"
#include "rtexisrc/rtEXIEventCodeGroup.h"
#include "rtxmlsrc/osrtxml.h"
#include "rtxsrc/rtxArrayList.h"
```

Classes

- struct OSEXIStateEvent

  This structure holds state/event information.

- struct OSEXIAutomaton

  This structure defines a finite state automata for EXI grammars.

- struct OSEXIAtmState

  This structure defines state information from an automaton that must be preserved at each stack level.

Typedefs

- typedef int(∗ OSEXIAtmAddTransFunc)(OSCTXT ∗pctxt, OSEXIAutomaton ∗pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent ∗pEvent, const OSEXIEventCode ∗pEventCode)

  Add transition function definition.

Functions

- EXTERNEXI void rtEXIAutomatonInit (OSCTXT ∗pctxt, OSEXIAutomaton ∗pAutomaton, const OSXMLFullQName ∗pElemName, OSEXIState numStates)

  This function initializes the automaton to its default state.

- EXTERNEXI OSEXIAutomaton ∗ rtEXINewAutomaton (OSCTXT ∗pctxt, const OSXMLFullQName ∗pElemName, OSEXIState numStates)

  This function allocates memory for a new automaton structure and initializes the structure.

- EXTERNEXI OSEXIAutomaton ∗ rtEXIAutomatonCopy (OSCTXT ∗pctxt, OSEXIAutomaton ∗pAutomaton)

  This function copies an automaton structure.

- EXTERNEXI void rtEXIAutomatonFreeMem (OSCTXT ∗pctxt, OSEXIAutomaton ∗pAutomaton, OSBOOL dynamic)

  This function frees all memory within an Automaton structure.

- EXTERNEXI OSEXIAutomaton ∗ rtEXIAutomatonAddTransition (OSCTXT ∗pctxt, OSEXIAutomaton ∗pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEventCode ∗pEventCode)

  This function adds a transition between two states.

- EXTERNEXI int rtEXIAtmAddUndeclaredItems (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, OSEXIEventCode *pEventCode, size_t numDeclAttrs, const OSXMLFullQName *declAttrs, OSEXIAtmAddTransFunc addTransFunc)

  This function adds all undeclared items (end element, start tag, and content) to an element automaton in schema-informed mode.

- EXTERNEXI int rtEXIAtmAddUndeclaredStartTagItems (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, OSEXIEventCode *pEventCode, size_t numDeclAttrs, const OSXMLFullQName *declAttrs, OSEXIAtmAddTransFunc addTransFunc)

  This function adds undeclared start tag items to an element automaton in schema-informed mode.

- EXTERNEXI int rtEXIAtmAddUndeclaredContentItems (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, OSEXIEventCode *pEventCode, OSEXIAtmAddTransFunc addTransFunc)

  This function adds undeclared content items to an element automaton in schema-informed mode.

- EXTERNEXI OSEXIAutomaton * rtEXIAutomatonInitCopy (OSCTXT *pctxt, OSEXIAutomaton *pDestAtm, OSEXIAutomaton *pSrcAtm)

  This function initializes an automaton structure using the data from an existing automaton.

- EXTERNEXI int rtEXIAutomatonPush (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton)

  This function pushes an automaton onto the context stack.

- EXTERNEXI OSEXIAutomaton * rtEXIAutomatonPop (OSCTXT *pctxt)

  This function pops an automaton from the context stack.

- EXTERNEXI OSEXIEventCodeGroup * rtEXIAtmGetCurrentEventCodeGroup (OSEXIAutomaton *pAutomaton)

  This function returns a pointer to the event code group corresponding to the current state.

- EXTERNEXI OSEXIAutomaton * rtEXIGetDocAutomaton (OSCTXT *pctxt, size_t numGblElems, const OSXMLFullQName *gblElems, OSEXIAtmAddTransFunc addTransFunc)

  This functions returns an automaton that accepts the built-in document grammar.

- EXTERNEXI OSEXIAutomaton * rtEXIGetElemAutomaton (OSCTXT *pctxt, OSXMLFullQName *pqname, int(*addTransFunc)(OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent *pEvent, const OSEXIEventCode *pEventCode))

  This function returns an automaton that accepts the built-in element grammar.

7.4.1 Detailed Description

EXI automaton structure and functions.

Definition in file rtEXIAutomaton.h.

7.5 rtEXIDecAutomaton.h File Reference

EXI decoding automaton structure and functions.

`#include "rtexisrc/rtEXIAutomaton.h"`

Functions

- EXTERNEXI int rtEXIDecAtmAddTransition (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent *pEvent, const OSEXIEventCode *pEventCode)

  This function adds a transition between two states.

- EXTERNEXI OSEXIEvent * rtEXIDecAutomatonAdvance (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, const OSEXIEventCode *pEventCode, OSBOOL dynamicItems)

  This function advances the decoder automaton based on event code, adding new transitions if `dynamicItems` *is set to* `true` *and either SE(*) or AT(*) are matched instead of SE(qname) or AT(qname), respectively.*

- EXTERNEXI OSEXIAutomaton * rtEXIDecGetDocAutomaton (OSCTXT *pctxt, size_t numGblElems, const OSXMLFullQName *gblElems)

  This functions returns an automaton that accepts the built-in document grammar.

- EXTERNEXI OSEXIAutomaton * rtEXIDecGetElemAutomaton (OSCTXT *pctxt, OSXMLFullQName *pqname)

  This function returns an automaton that accepts the built-in element grammar.

7.5.1 Detailed Description

EXI decoding automaton structure and functions.

Definition in file rtEXIDecAutomaton.h.

7.6 rtEXIDecBitTrace.h File Reference

Functions for logging bit trace diagnostic events that occur during the decoding of an EXI-encoded document.

```
#include "rtexisrc/osrtexi.h"
```

7.6.1 Detailed Description

Functions for logging bit trace diagnostic events that occur during the decoding of an EXI-encoded document.

Definition in file rtEXIDecBitTrace.h.

7.7 rtEXIDecoder.h File Reference

Interface for EXI low-level decoders.

```
#include "rtexisrc/osrtexi.h"
#include "rtxmlsrc/osrtxml.h"
#include "rtxsrc/rtxArrayList.h"
```

Functions

- EXTERNEXI int rtEXIDecAttribute (OSCTXT *pctxt, OSXMLFullQName *pqname, const OSUTF8CHAR **ppvalue)

  Decodes the attribute at the current position in the decode stream.

- EXTERNEXI int rtEXIDecBinary (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, OSINT32 bufsize)

  This function decodes the contents of a binary value into a static memory structure.

- EXTERNEXI int rtEXIDecBoolValue (OSCTXT *pctxt, OSBOOL *pvalue)

  This function decodes a boolean value.

- EXTERNEXI int rtEXIDecBoolValueWithPattern (OSCTXT *pctxt, OSBOOL *pvalue, OSUINT8 pattern)

  This function decodes a boolean value with pattern facet.

- EXTERNEXI int rtEXIDec_CH_String_EE (OSCTXT *pctxt, const OSXMLFullQName *pqname, const OSUINT16 *charSet, const OSUTF8CHAR **ppvalue)

  This function decodes a CH event (assumed to be a one part code = 0 in a 1 bit field) followed by string content followed by an EE event (assumed to be a one part code = 0 in a 1 bit field).

- EXTERNEXI int rtEXIDecDate (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a date value into a structured variable.

- EXTERNEXI int rtEXIDecDateString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a date value into a string.

- EXTERNEXI int rtEXIDecDateTime (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a date/time value into a structured variable.

- EXTERNEXI int rtEXIDecDateTimeString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a date/time value into a string.

- EXTERNEXI int rtEXIDecDecimalValue (OSCTXT *pctxt, OSREAL *pvalue)

  This function decodes a decimal value.

- EXTERNEXI int rtEXIDecDocumentType (OSCTXT *pctxt, const OSUTF8CHAR **ppName, const OSUTF8CHAR **ppPublic, const OSUTF8CHAR **ppSystem, const OSUTF8CHAR **ppText)

  This function decodes an XML document type declaration (DTD).

- EXTERNEXI int rtEXIDecDoubleValue (OSCTXT *pctxt, OSREAL *pvalue)

  This function decodes a double value.

- EXTERNEXI int rtEXIDecDynBinary (OSCTXT *pctxt, OSDynOctStr *pvalue)

  This function decodes a binary value into a structured variable.

- EXTERNEXI int rtEXIDecEndDocument (OSCTXT *pctxt)

  This function ends decoding of EXI stream.

- EXTERNEXI int rtEXIDecEndEvent (OSCTXT *pctxt)

  This function decodes of event codes of element end grammar.

- EXTERNEXI int rtEXIDecEndEventCompact (OSCTXT *pctxt)

  This function decodes of event codes of element end grammar.

- EXTERNEXI int rtEXIDecEndEventStrict (OSCTXT *pctxt)

  This function decodes of event codes of element end grammar.

- EXTERNEXI int rtEXIDecEventCodePart1 (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 nbits, OSUINT32 maxval)

  This function decodes part 1 of an event code.

- EXTERNEXI int rtEXIDecGetStateIndex (OSCTXT *pctxt, OSUINT32 flags)

  This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

- EXTERNEXI int rtEXIDecGetStateIndexCompact (OSCTXT *pctxt, OSUINT16 stx, const OSEXIStateTableRecord statetab[ ], size_t nrows, OSUINT32 flags, OSUINT32 *prepcnt)

  This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

- EXTERNEXI int rtEXIDecGetStateIndexStrict (OSCTXT *pctxt, OSUINT16 stx, const OSEXIStateTableRecord statetab[ ], size_t nrows, OSUINT32 flags, OSUINT32 *prepcnt)

  This function decodes an event code as specified in the currently indexed state table record and then determine the index of the state corresponding to the decoded value.

- EXTERNEXI int rtEXIDecFloatValue (OSCTXT *pctxt, OSFLOAT *pvalue)

  This function decodes a float value.

- EXTERNEXI int rtEXIDecGDay (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gDay value into a structured variable.

- EXTERNEXI int rtEXIDecGDayString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gDay value into a string.

- EXTERNEXI int rtEXIDecGMonth (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gMonth value into a structured variable.

- EXTERNEXI int rtEXIDecGMonthDay (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gMonthDay value into a structured variable.

- EXTERNEXI int rtEXIDecGMonthDayString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gMonthDay value into a string.

- EXTERNEXI int rtEXIDecGMonthString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gMonth value into a string.

- EXTERNEXI int rtEXIDecGYear (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gYear value into a structured variable.

- EXTERNEXI int rtEXIDecGYearMonth (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a gYearMonth value into a structured variable.

- EXTERNEXI int rtEXIDecGYearMonthString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gYearMonth value into a string.

- EXTERNEXI int rtEXIDecGYearString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a gYear value into a string.

- EXTERNEXI OSBOOL rtEXIDecHasNext (OSCTXT *pctxt)

  This function checks for additonal events in the decode stream.

- EXTERNEXI int rtEXIDecHeader (OSCTXT *pctxt)

  This function decodes the EXI header at the start of a message and uses the results to set internal flags within the context structure.

- EXTERNEXI int rtEXIDecInitCompression (OSCTXT *pctxt, OSUINT32 nmChannelIds)

  This function initialize decoder to use compressed or precompressed EXI stream.

- EXTERNEXI int rtEXIDecInt16Value (OSCTXT *pctxt, OSINT16 *pvalue)

  This function decodes a 16-bit signed integer value.

- EXTERNEXI int rtEXIDecInt64Value (OSCTXT *pctxt, OSINT64 *pvalue)

  This function decodes a 64-bit signed integer value.

- EXTERNEXI int rtEXIDecInt8Value (OSCTXT *pctxt, OSINT8 *pvalue)

  This function decodes a 8-bit signed integer value.

- EXTERNEXI int rtEXIDecIntValue (OSCTXT *pctxt, OSINT32 *pvalue)

  This function decodes a signed integer value.

- EXTERNEXI int rtEXIDecLocalName (OSCTXT *pctxt, const OSUTF8CHAR **ppname)

  Returns the local name associated with the current event.

- EXTERNEXI int rtEXIDecNamespaceURI (OSCTXT *pctxt, const OSUTF8CHAR **ppNSURI)

  Returns the namespace associated with the current event.

- EXTERNEXI int rtEXIDecNBitUIntValue (OSCTXT *pctxt, OSUINT32 *pvalue, OSUINT32 nbits)

  This function decodes an unsigned integer value from a bit field of the given width.

- EXTERNEXI int rtEXIDecNextEventType (OSCTXT *pctxt, OSEXIEventType *pEventType)

  Returns the next `OSEXIEventType` *read by this decoder.*

- EXTERNEXI int rtEXIDecoderInit (OSCTXT *pctxt)

  This function initializes the decoder.

- EXTERNEXI int rtEXIDecPrefix (OSCTXT *pctxt, const OSUTF8CHAR *uri, const OSUTF8CHAR **ppPrefix)

  Returns the namespace associated with the current event.

- EXTERNEXI int rtEXIDecProcessingInstruction (OSCTXT *pctxt, const OSUTF8CHAR **ppTarget, const OSUTF8CHAR **ppData)

  This function decodes an XML processing instruction (PI).

- EXTERNEXI int rtEXIDecQName (OSCTXT *pctxt, OSXMLFullQName *pqname)

  Returns the qname associated with the current event.

- EXTERNEXI int rtEXIDecQNameValue (OSCTXT *pctxt, OSXMLFullQName *pvalue, const OSXMLFullQName *pqname)

  Decode the qname value for AT and CH events.

- EXTERNEXI int rtEXIDecReset (OSCTXT *pctxt)

  Resets the decoder for decoding a new message instance.

- EXTERNEXI int rtEXIDecSimpleTypeEvent (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 flags)

  This function decodes of event codes of simple type grammar.

- EXTERNEXI int rtEXIDecSimpleTypeEventCompact (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 flags)

  This function decodes of event codes of simple type grammar.

- EXTERNEXI int rtEXIDecSimpleTypeEventStrict (OSCTXT *pctxt, OSINT32 *ppart, OSUINT32 flags)

  This function decodes of event codes of simple type grammar.

- EXTERNEXI int rtEXIDecString (OSCTXT *pctxt, const OSXMLFullQName *pqname, const OSUINT16 *charSet, const OSUTF8CHAR **ppvalue)

  Returns the value associated with the current event.

- EXTERNEXI int rtEXIDecStringToCharArray (OSCTXT *pctxt, const OSUTF8CHAR *target, size_t start, size_t length)

  Similar to `rtEXIDecString` *but the characters are copied into a fixed-size character array.*

- EXTERNEXI int rtEXIDecStringLength (OSCTXT *pctxt)

  Returns the length of the string returned by `rtEXIDecString`.

- EXTERNEXI int rtEXIDecTime (OSCTXT *pctxt, OSNumDateTime *pvalue)

  This function decodes a time value into a structured variable.

- EXTERNEXI int rtEXIDecTimeString (OSCTXT *pctxt, const OSUTF8CHAR **ppvalue)

  This function decodes a time value into a string.

- EXTERNEXI int rtEXIDecUInt16Value (OSCTXT *pctxt, OSUINT16 *pvalue)

  This function decodes an 16-bit unsigned integer value.

- EXTERNEXI int rtEXIDecUInt64Value (OSCTXT *pctxt, OSUINT64 *pvalue)

This function decodes an 64-bit unsigned integer value.

- EXTERNEXI int rtEXIDecUInt8Value (OSCTXT *pctxt, OSUINT8 *pvalue)

  This function decodes an 8-bit unsigned integer value.

- EXTERNEXI int rtEXIDecUIntValue (OSCTXT *pctxt, OSUINT32 *pvalue)

  This function decodes an unsigned integer value.

- EXTERNEXI int rtEXIDecUTF8Str (OSCTXT *pctxt, OSUTF8CHAR **ppvalue, const OSUINT16 *charSet)

  This function decodes a UTF-8 string value.

- EXTERNEXI int rtEXIDecUTF8Chars (OSCTXT *pctxt, OSUTF8CHAR **ppvalue, OSUINT32 nchars, const OSUINT16 *charSet)

  This function reads the given number of characters from the decode stream and creates a UTF-8 string.

7.7.1 Detailed Description

Interface for EXI low-level decoders.

The interface is similar to StAX, but uses enumerations for the different event types and reports attributes and namespaces as individual events.

Definition in file rtEXIDecoder.h.

7.8 rtEXIDecStringTable.h File Reference

EXI decoder string table structure and functions.

```
#include "rtxsrc/rtxArrayList.h"
#include "rtxsrc/rtxContext.h"
#include "rtexisrc/rtEXIExternDefs.h"
```

Classes

- struct OSEXIDecStringTable

  This structure defines the structure of the various string table partitions used by the decoder.

Functions

- EXTERNEXI void rtEXIDecStringTableInit (OSCTXT *pctxt, OSEXIDecStringTable *pstrtab, size_t capacity)

  This function initializes the given string table structure.

- EXTERNEXI OSEXIDecStringTable * rtEXIDecNewStringTable (OSCTXT *pctxt, size_t capacity)

  This function allocates and initializes a new string table structure.

- EXTERNEXI void rtEXIDecStringTableClear (OSCTXT *pctxt, OSEXIDecStringTable *pstrtab)

  This function clears all strings out of the existing table.

- EXTERNEXI OSUINT32 rtEXIDecStringTableAdd (OSCTXT *pctxt, OSEXIDecStringTable *pstrtab, const OSUTF8CHAR *str)

  This function adds a string to the given string table.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStringTableGetString (OSEXIDecStringTable *pstrtab, OSUINT32 index)

  This function gets the string at the given index (i.e.

7.8.1 Detailed Description

EXI decoder string table structure and functions.

Definition in file rtEXIDecStringTable.h.

7.9 rtEXIDecStringTables.h File Reference

EXI decoder string table structure and functions.

```
#include "rtexisrc/rtEXIDecStringTable.h"
#include "rtxsrc/rtxHashMap.h"
#include "rtxmlsrc/osrtxml.h"
```

Classes

- struct OSEXIDecStringTables

  This structure defines the complete set of string table partitions used by the decoder.

Functions

- EXTERNEXI void rtEXIDecStrTabsInit (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs)

  This function initializes all EXI string table partitions.

- EXTERNEXI void rtEXIDecStrTabsClear (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs)

  This function clears all EXI string table partitions.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddURI (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function will add a URI to the URI string table partition.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetURI (OSEXIDecStringTables *pstrtabs, OSUINT32 index)

  This function will get the compact identifier of the given URI from the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetURITableSize (OSEXIDecStringTables *pstrtabs)

  This function returns the current number of entries in the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddPrefix (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *prefix)

  This function adds the given prefix to the prefix table partition identified by the given URI.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetPrefix (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, OSUINT32 index)

  This function will get the compact identifier of the given prefix from the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetPrefixTableSize (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddLocalName (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *name)

  This function adds the given local name to the local name table partition identified by the given URI.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetLocalName (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri, OSUINT32 index)

  This function will get the compact identifier of the given localName from the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetLocalNameTableSize (OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddLocalValue (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSXMLFullQName *qname, const OSUTF8CHAR *value)

  This function adds the given local value to the local value table partition identified by the given QName.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetLocalValue (OSEXIDecStringTables *pstrtabs, const OSXMLFullQName *qname, OSUINT32 index)

  This function will get the compact identifier of the given local value from the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetLocalValueTableSize (OSEXIDecStringTables *pstrtabs, const OSXMLFullQName *qname)

  This function returns the current number of entries in the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsAddGlobalValue (OSCTXT *pctxt, OSEXIDecStringTables *pstrtabs, const OSUTF8CHAR *value)

  This function will add a string value to the global value string table partition.

- EXTERNEXI const OSUTF8CHAR * rtEXIDecStrTabsGetGlobalValue (OSEXIDecStringTables *pstrtabs, OSUINT32 index)

  This function will get the compact identifier of the given string value from the global value string table partition.

- EXTERNEXI OSUINT32 rtEXIDecStrTabsGetGlobalValueTableSize (OSEXIDecStringTables *pstrtabs)

  This function returns the current number of entries in the global value string table partition.

7.9.1 Detailed Description

EXI decoder string table structure and functions.

Definition in file rtEXIDecStringTables.h.

7.10 rtEXIEncAutomaton.h File Reference

EXI encoding automaton structure and functions.

`#include "rtexisrc/rtEXIAutomaton.h"`

Functions

- EXTERNEXI int rtEXIEncAtmAddTransition (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, OSEXIState fromState, OSEXIState toState, const OSEXIEvent *pEvent, const OSEXIEventCode *pEventCode)

  This function adds a transition between two states.

- EXTERNEXI OSEXIEventCode * rtEXIEncAutomatonAdvance (OSCTXT *pctxt, OSEXIAutomaton *pAutomaton, const OSEXIEvent *pEvent, OSBOOL dynamicItems)

  This function advances the encoder automaton based on event, adding new transitions if `dynamicItems` *is set to* `true` *and either SE(*) or AT(*) are matched instead of SE(qname) or AT(qname), respectively.*

- EXTERNEXI OSEXIAutomaton * rtEXIEncGetDocAutomaton (OSCTXT *pctxt, size_t numGblElems, const OSXMLFullQName *gblElems)

  This functions returns an automaton that accepts the built-in document grammar.

- EXTERNEXI OSEXIAutomaton * rtEXIEncGetElemAutomaton (OSCTXT *pctxt, OSXMLFullQName *pqname)

  This function returns an automaton that accepts the built-in element grammar.

7.10.1 Detailed Description

EXI encoding automaton structure and functions.

Definition in file rtEXIEncAutomaton.h.

7.11 rtEXIEncoder.h File Reference

EXI low-level C encode functions.

`#include "rtexisrc/osrtexi.h"`

Functions

- EXTERNEXI int rtEXIEncAttribute (OSCTXT *pctxt, const OSUTF8CHAR *prefix, const OSUTF8CHAR *namespaceURI, const OSUTF8CHAR *localName, const OSUTF8CHAR *value)

  This function writes an attribute in the current element.

- EXTERNEXI int rtEXIEncBinary (OSCTXT *pctxt, OSUINT32 nocts, const OSOCTET *value)

  This function encodes a binary (OCTET) string value.

- EXTERNEXI int rtEXIEncBoolValue (OSCTXT *pctxt, OSBOOL value)

  This function encodes a boolean value.

- EXTERNEXI int rtEXIEncBoolValueWithPattern (OSCTXT *pctxt, OSBOOL value, OSUINT8 pattern)

  This function encodes a boolean value with pattern facet.

- EXTERNEXI int rtEXIEncComment (OSCTXT *pctxt, const OSUTF8CHAR *data)

  This function encode an XML comment.

- EXTERNEXI int rtEXIEncCharacters (OSCTXT *pctxt, const OSUTF8CHAR *text)

  This function encodes a string of characters.

- EXTERNEXI int rtEXIEncCharArray (OSCTXT *pctxt, const OSUTF8CHAR *text, OSUINT32 nbytes)

  This function encodes a given number of bytes from a character array or string.

- EXTERNEXI int rtEXIEncDate (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric date value.

- EXTERNEXI int rtEXIEncDateString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a date string value.

- EXTERNEXI int rtEXIEncDateTime (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric date/time value.

- EXTERNEXI int rtEXIEncDateTimeString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a date/time string value.

- EXTERNEXI int rtEXIEncDecimalValue (OSCTXT *pctxt, OSREAL value)

  This function encodes a variable of the XSD decimal type.

- EXTERNEXI int rtEXIEncDoubleValue (OSCTXT *pctxt, OSREAL value)

  This function encodes a variable of the float type.

- EXTERNEXI int rtEXIEncDTD (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *publix, const OSUTF8CHAR *system, const OSUTF8CHAR *text)

This function encodes a DTD declaration.

- EXTERNEXI int rtEXIEncElemGrammarEvent (OSCTXT *pctxt, OSUINT16 eventId)

  This function writes a start element event code (SE) for a specific XML element as specified in an EXI grammar.

- EXTERNEXI int rtEXIEncElemGrammarEE (OSCTXT *pctxt)

  This function writes an end element event code (EE) for a specific XML element as specified in an EXI grammar.

- EXTERNEXI int rtEXIEncEndDocument (OSCTXT *pctxt)

  This function writes an end document event.

- EXTERNEXI int rtEXIEncEndElement (OSCTXT *pctxt)

  This function writes an end element event.

- EXTERNEXI int rtEXIEncEndEvent (OSCTXT *pctxt)

  This function writes EE event code.

- EXTERNEXI int rtEXIEncEntityRef (OSCTXT *pctxt, const OSUTF8CHAR *name)

  This function writes an XML entity reference.

- EXTERNEXI int rtEXIEncEventCode (OSCTXT *pctxt, const char *name, OSINT32 part1, OSINT32 part2, OSINT32 part3, OSUINT32 nbits1, OSUINT32 nbits2, OSUINT32 nbits3)

  This function writes an event code using the given bit field sizes.

- EXTERNEXI int rtEXIEncGDay (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric gDay value.

- EXTERNEXI int rtEXIEncGDayString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a gDay string value.

- EXTERNEXI int rtEXIEncGMonth (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric gMonth value.

- EXTERNEXI int rtEXIEncGMonthDay (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric gMonthDay value.

- EXTERNEXI int rtEXIEncGMonthDayString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a gMonthDay string value.

- EXTERNEXI int rtEXIEncGMonthString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a gMonth string value.

- EXTERNEXI int rtEXIEncGYear (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric gYear value.

- EXTERNEXI int rtEXIEncGYearMonth (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric gYearMonth value.

- EXTERNEXI int rtEXIEncGYearMonthString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a gYearMonth string value.

- EXTERNEXI int rtEXIEncGYearString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a gYear string value.

- EXTERNEXI int rtEXIEncHeader (OSCTXT *pctxt)

  This function encodes the EXI header that is at the start of all EXI messages.

- EXTERNEXI int rtEXIEncInitCompression (OSCTXT *pctxt, OSUINT32 nmChannelIds)

  This function initialize encoder to use compressed or precompressed EXI stream.

- EXTERNEXI int rtEXIEncInt64Value (OSCTXT *pctxt, OSINT64 value)

  This function encodes a variable of the 64-bit integer type.

- EXTERNEXI int rtEXIEncIntCHEvent (OSCTXT *pctxt, OSINT32 value)

  This function encodes a variable of the XSD integer type as a CH content event.

- EXTERNEXI int rtEXIEncIntElem (OSCTXT *pctxt, OSINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

  This function encodes a complete XML element (<tag>content</tag>) in which the content is a signed integer.

- EXTERNEXI int rtEXIEncIntValue (OSCTXT *pctxt, OSINT32 value)

  This function encodes a variable of the XSD integer type.

- EXTERNEXI int rtEXIEncNBitUIntValue (OSCTXT *pctxt, OSUINT32 value, OSUINT32 nbits)

  This function encodes an unsigned integer value in a bit field of the given width.

- EXTERNEXI int rtEXIEncNamespace (OSCTXT *pctxt, const OSUTF8CHAR *prefix, const OSUTF8CHAR *namespaceURI, OSBOOL indicator)

  This function encodes a namespace declaration.

- EXTERNEXI int rtEXIEncoderInit (OSCTXT *pctxt)

  This function initializes the encoder.

- EXTERNEXI int rtEXIEncProcessingInstruction (OSCTXT *pctxt, const OSUTF8CHAR *target, const OSUTF8CHAR *data)

  This function writes an XML processing instruction.

- EXTERNEXI int rtEXIEncQNameValue (OSCTXT *pctxt, const OSXMLFullQName *pvalue, const OSXMLFullQName *pqname)

  This function encodes a QName value for AT and CH events.

- EXTERNEXI int rtEXIEncSimpleTypeEvent (OSCTXT *pctxt, const char *name, OSINT32 part1, OSBOOL hasXsiAttrs)

  This function writes an event code for simple type elements.

- EXTERNEXI int rtEXIEncStartDocument (OSCTXT *pctxt)

  This function writes a start document event.

- EXTERNEXI int rtEXIEncStartElement (OSCTXT *pctxt, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

This function writes a start element event.

- EXTERNEXI int rtEXIEncString (OSCTXT *pctxt, const OSUTF8CHAR *value, const OSXMLFullQName *pqname, const OSUINT16 *charSet)

  This function encodes a character string value.

- EXTERNEXI int rtEXIEncString2 (OSCTXT *pctxt, const OSUTF8CHAR *value, const OSUTF8CHAR *elemName, struct OSXMLNamespace *pNS, const OSUINT16 *charSet)

  This version of the rtEXIEncString function uses standard generated encode function arguments.

- EXTERNEXI int rtEXIEncTime (OSCTXT *pctxt, const OSNumDateTime *pvalue)

  This function encodes a numeric time value.

- EXTERNEXI int rtEXIEncTimeString (OSCTXT *pctxt, const OSUTF8CHAR *pvalue)

  This function encodes a time string value.

- EXTERNEXI int rtEXIEncUInt64Value (OSCTXT *pctxt, OSUINT64 value)

  This function encodes an 64-bit unsigned integer value.

- EXTERNEXI int rtEXIEncUIntCHEvent (OSCTXT *pctxt, OSUINT32 value)

  This function encodes a variable of the XSD unsigned integer type as a CH content event.

- EXTERNEXI int rtEXIEncUIntElem (OSCTXT *pctxt, OSUINT32 value, const OSUTF8CHAR *elemName, OSXMLNamespace *pNS)

  This function encodes a complete XML element (<tag>content</tag>) in which the content is an unsigned integer.

- EXTERNEXI int rtEXIEncUIntValue (OSCTXT *pctxt, OSUINT32 value)

  This function encodes an unsigned integer value.

- EXTERNEXI int rtEXIEncUTF8Str (OSCTXT *pctxt, const OSUTF8CHAR *value, size_t lengthIncr, const OSUINT16 *charSet)

  This function encodes a UTF-8 character string value.

7.11.1 Detailed Description

EXI low-level C encode functions.

Definition in file rtEXIEncoder.h.

7.11.2 Function Documentation

7.11.2.1 EXTERNEXI int rtEXIEncAttribute (OSCTXT * *pctxt*, const OSUTF8CHAR * *prefix*, const OSUTF8CHAR * *namespaceURI*, const OSUTF8CHAR * *localName*, const OSUTF8CHAR * *value*)

This function writes an attribute in the current element.

Parameters:

pctxt Pointer to OSCTXT structure

prefix The attribute's prefix or empty (null or "") for no prefix.

namespaceURI The attribute's namespace URI or empty for the default namespace.

localName The attribute's local name which must be a non-empty string.

value The attributes's value which must be non-null.

Returns:

Completion status of operation:

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

7.11.2.2 EXTERNEXI int rtEXIEncBinary (OSCTXT * *pctxt*, OSUINT32 *nocts*, const OSOCTET * *value*)

This function encodes a binary (OCTET) string value.

Parameters:

pctxt Pointer to context block structure.

nocts Number of octets in value

value Pointer to string of octets to be encoded.

Returns:

Completion status of operation:

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

7.11.2.3 EXTERNEXI int rtEXIEncBoolValue (OSCTXT * *pctxt*, OSBOOL *value*)

This function encodes a boolean value.

Parameters:

pctxt Pointer to context block structure.

value OSBOOL value.

Returns:

Completion status of operation:

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

7.11.2.4 EXTERNEXI int rtEXIEncBoolValueWithPattern (OSCTXT * *pctxt*, OSBOOL *value*, OSUINT8 *pattern*)

This function encodes a boolean value with pattern facet.

Parameters:

pctxt Pointer to context block structure.

value OSBOOL value.

pattern Mask of enabled values:

- 0 - "false"
- 1 - "0"
- 2 - "true"
- 3 - "1"

Returns:

Completion status of operation:

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

7.11.2.5 EXTERNEXI int rtEXIEncCharacters (OSCTXT * *pctxt*, const OSUTF8CHAR * *text*)

This function encodes a string of characters.

Parameters:

pctxt Pointer to context block structure.

text Pointer to text to be encoded.

Returns:

Completion status of operation:

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

7.11.2.6 EXTERNEXI int rtEXIEncCharArray (OSCTXT * *pctxt*, const OSUTF8CHAR * *text*, OSUINT32 *nbytes*)

This function encodes a given number of bytes from a character array or string.

This function may be used from within a SAX characters handler when the string is not null-terminated.

Parameters:

pctxt Pointer to context block structure.

text Pointer to text to be encoded.

nbytes Number of bytes to encode.

Returns:

Completion status of operation:

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

7.11.2.7 EXTERNEXI int rtEXIEncComment (OSCTXT * *pctxt*, const OSUTF8CHAR * *data*)

This function encode an XML comment.

Parameters:

pctxt Pointer to context block structure.

data The comment to be encoded

Returns:

Completion status of operation:

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

7.11.2.8 EXTERNEXI int rtEXIEncDate (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric date value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.9 EXTERNEXI int rtEXIEncDateString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a date string value.

Parameters:

pctxt Pointer to context block structure.

pvalue Date string to be encoded

- The format of date is CCYY-MM-DD
- The value of CCYY is from 0000-9999
- The value of MM is 01 - 12
- The value of DD is 01 - XX (where XX is the Days in MM month in CCYY year)

Returns:

Completion status of operation:

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

7.11.2.10 EXTERNEXI int rtEXIEncDateTime (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric date/time value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.11 EXTERNEXI int rtEXIEncDateTimeString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a date/time string value.

Parameters:

pctxt Pointer to context block structure.

pvalue Date/time string to be encoded

- The format is CCYY-MM-DDTHH:MM:SS[.frac][TZ]
- The value of CCYY is from 0000-9999
- The value of MM is 01 - 12
- The value of DD is 01 - XX (where XX is the Days in MM month in CCYY year)
- The value of HH is 00 - 23
- The value of MM is 00 - 59
- The value of SS is 00 - 59

Returns:

Completion status of operation:

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

7.11.2.12 EXTERNEXI int rtEXIEncDecimalValue (OSCTXT * *pctxt*, OSREAL *value*)

This function encodes a variable of the XSD decimal type.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.13 EXTERNEXI int rtEXIEncDoubleValue (OSCTXT * *pctxt*, OSREAL *value*)

This function encodes a variable of the float type.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.14 EXTERNEXI int rtEXIEncDTD (OSCTXT * *pctxt*, const OSUTF8CHAR * *name*, const OSUTF8CHAR * *publix*, const OSUTF8CHAR * *system*, const OSUTF8CHAR * *text*)

This function encodes a DTD declaration.

Parameters:

pctxt Pointer to context block structure.

name The DTD's root element.

publix The DTD's public identifier.

system The DTD's system identifier.

text The DTD's textual representation.

Returns:

Completion status of operation:

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

7.11.2.15 EXTERNEXI int rtEXIEncElemGrammarEE (OSCTXT * *pctxt*)

This function writes an end element event code (EE) for a specific XML element as specified in an EXI grammar.

It is assumed state table information in the EXI context is set to point at the first state record in a state group containing the EE event type to be encoded.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

7.11.2.16 EXTERNEXI int rtEXIEncElemGrammarEvent (OSCTXT * *pctxt*, OSUINT16 *eventId*)

This function writes a start element event code (SE) for a specific XML element as specified in an EXI grammar. It is assumed state table information in the EXI context is set to point at the first state record in a state group.

Parameters:

pctxt Pointer to context block structure.

eventId ID of event to be encoded.

Returns:

Completion status of operation:

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

7.11.2.17 EXTERNEXI int rtEXIEncEndDocument (OSCTXT * *pctxt*)

This function writes an end document event.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

7.11.2.18 EXTERNEXI int rtEXIEncEndElement (OSCTXT * *pctxt*)

This function writes an end element event.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

7.11.2.19 EXTERNEXI int rtEXIEncEndEvent (OSCTXT * *pctxt*)

This function writes EE event code.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

7.11.2.20 EXTERNEXI int rtEXIEncEntityRef (OSCTXT * *pctxt*, const OSUTF8CHAR * *name*)

This function writes an XML entity reference.

Parameters:

pctxt Pointer to context block structure.

name The entity's name.

Returns:

Completion status of operation:

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

7.11.2.21 EXTERNEXI int rtEXIEncEventCode (OSCTXT * *pctxt*, const char * *name*, OSINT32 *part1*, OSINT32 *part2*, OSINT32 *part3*, OSUINT32 *nbits1*, OSUINT32 *nbits2*, OSUINT32 *nbits3*)

This function writes an event code using the given bit field sizes.

Parameters:

pctxt Pointer to context block structure.

name Textual name of event (for example, "SE(*)")

part1 Part 1 of event code. -1 if not used.

part2 Part 2 of event code. -1 if not used.

part3 Part 3 of event code. -1 if not used.

nbits1 Length of bit field for part 1.

nbits2 Length of bit field for part 2.

nbits3 Length of bit field for part 3.

Returns:

Completion status of operation:

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

7.11.2.22 EXTERNEXI int rtEXIEncGDay (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric gDay value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.23 EXTERNEXI int rtEXIEncGDayString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a gDay string value.

Parameters:

pctxt Pointer to context block structure.

pvalue gDay string to be encoded

- The format of gDay is —DD
- The value of DD is 01 - 31

Returns:

Completion status of operation:

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

7.11.2.24 EXTERNEXI int rtEXIEncGMonth (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric gMonth value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.25 EXTERNEXI int rtEXIEncGMonthDay (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric gMonthDay value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.26 EXTERNEXI int rtEXIEncGMonthDayString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a gMonthDay string value.

Parameters:

pctxt Pointer to context block structure.

pvalue gMonthDay string to be encoded

- The format of gMonthDay is –MM-DD
- The value of MM is 01 - 12
- The value of DD is 01 - XX (where XX is the Days in MM month)

Returns:

Completion status of operation:

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

7.11.2.27 EXTERNEXI int rtEXIEncGMonthString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a gMonth string value.

Parameters:

pctxt Pointer to context block structure.

pvalue gMonth string to be encoded

- The format of gMonth is –MM
- The value of MM is 01 - 12

Returns:

Completion status of operation:

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

7.11.2.28 EXTERNEXI int rtEXIEncGYear (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric gYear value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.29 EXTERNEXI int rtEXIEncGYearMonth (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric gYearMonth value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.30 EXTERNEXI int rtEXIEncGYearMonthString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a gYearMonth string value.

Parameters:

pctxt Pointer to context block structure.

pvalue gYearMonth string to be encoded

- The format of gYearMonth is CCYY-MM
- The value of CCYY is from 0000-9999
- The value of MM is 01 - 12

Returns:

Completion status of operation:

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

7.11.2.31 EXTERNEXI int rtEXIEncGYearString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a gYear string value.

Parameters:

pctxt Pointer to context block structure.

pvalue gYear string to be encoded

- The format of gYear is CCYY
- The value of CCYY is from 0000-9999

Returns:

Completion status of operation:

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

7.11.2.32 EXTERNEXI int rtEXIEncHeader (OSCTXT * *pctxt*)

This function encodes the EXI header that is at the start of all EXI messages.

Parameters:

pctxt Pointer to context block structure. Header options are determined from flags within the EXI info block of the context.

Returns:

Completion status of operation:

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

7.11.2.33 EXTERNEXI int rtEXIEncInitCompression (OSCTXT * *pctxt*, OSUINT32 *nmChannelIds*)

This function initialize encoder to use compressed or precompressed EXI stream.

Parameters:

pctxt Pointer to context block structure.

nmChannelIds Number of preallocated value channels.

Returns:

Completion status of operation:

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

7.11.2.34 EXTERNEXI int rtEXIEncInt64Value (OSCTXT * *pctxt*, OSINT64 *value*)

This function encodes a variable of the 64-bit integer type.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.35 EXTERNEXI int rtEXIEncIntCHEvent (OSCTXT * *pctxt*, OSINT32 *value*)

This function encodes a variable of the XSD integer type as a CH content event.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.36 EXTERNEXI int rtEXIEncIntElem (OSCTXT * *pctxt*, OSINT32 *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a complete XML element (<tag>content</tag>) in which the content is a signed integer.

Parameters:

pctxt Pointer to context block structure.

value Integer value to be encoded.

elemName Local name of element.

pNS Pointer to namespace structure containing prefix, and namespace URI.

Returns:

Completion status of operation:

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

7.11.2.37 EXTERNEXI int rtEXIEncIntValue (OSCTXT * *pctxt*, OSINT32 *value*)

This function encodes a variable of the XSD integer type.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.38 EXTERNEXI int rtEXIEncNamespace (OSCTXT * *pctxt*, const OSUTF8CHAR * *prefix*, const OSUTF8CHAR * *namespaceURI*, OSBOOL *indicator*)

This function encodes a namespace declaration.

If the "preserve prefixes" option is set to false, nothing is written.

Parameters:

pctxt Pointer to context block structure.

prefix The non-empty prefix being declared.

namespaceURI The namespace associated with the prefix.

indicator Indicator bit used to indicate this namespace prefix is associated with last encoded element QName.

Returns:

Completion status of operation:

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

7.11.2.39 EXTERNEXI int rtEXIEncNBitUIntValue (OSCTXT * *pctxt*, OSUINT32 *value*, OSUINT32 *nbits*)

This function encodes an unsigned integer value in a bit field of the given width.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

nbits Size of bit field.

Returns:

Completion status of operation:

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

7.11.2.40 EXTERNEXI int rtEXIEncoderInit (OSCTXT * *pctxt*)

This function initializes the encoder.

It must be called before any other encode functions when encoding a document or fragment.

Parameters:

pctxt Pointer to a context structure.

Returns:

Status of the operation:

- 0 if success
- a negative status code if failure

7.11.2.41 EXTERNEXI int rtEXIEncProcessingInstruction (OSCTXT * *pctxt*, const OSUTF8CHAR * *target*, const OSUTF8CHAR * *data*)

This function writes an XML processing instruction.

Parameters:

pctxt Pointer to context block structure.

target The PI's target.

data The PI's data.

Returns:

Completion status of operation:

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

7.11.2.42 EXTERNEXI int rtEXIEncQNameValue (OSCTXT * *pctxt*, const OSXMLFullQName * *pvalue*, const OSXMLFullQName * *pqname*)

This function encodes a QName value for AT and CH events.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to QName value to be encoded.

pqname Pointer to QName of attribute or element.

- 0 - encode xsi:type.

Returns:

Completion status of operation:

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

7.11.2.43 EXTERNEXI int rtEXIEncSimpleTypeEvent (OSCTXT * *pctxt*, const char * *name*, OSINT32 *part1*, OSBOOL *hasXsiAttrs*)

This function writes an event code for simple type elements.

Parameters:

pctxt Pointer to context block structure.

name Textual name of event (for example, "SE(*)")

part1 Part 1 of event code.

hasXsiAttrs Is element nillable or it type has named subtypes.

Returns:

Completion status of operation:

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

7.11.2.44 EXTERNEXI int rtEXIEncStartDocument (OSCTXT * *pctxt*)

This function writes a start document event.

This method must be called before writing a document or a fragment.

Parameters:

pctxt Pointer to context block structure.

Returns:

Completion status of operation:

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

7.11.2.45 EXTERNEXI int rtEXIEncStartElement (OSCTXT * *pctxt*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function writes a start element event.

If this method is called before rtEXIEncStartDocument, then the encoder assumes a fragment is being written.

Parameters:

pctxt Pointer to context block structure.

elemName The non-empty element name.

pNS Pointer to namespace structure containing prefix, and namespace URI.

Returns:

Completion status of operation:

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

7.11.2.46 EXTERNEXI int rtEXIEncString (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*, const OSXMLFullQName * *pqname*, const OSUINT16 * *charSet*)

This function encodes a character string value.

It uses the string tables to determine if the actual string value should be encoded or not.

Parameters:

pctxt Pointer to context block structure.

value Pointer to string value to be encoded.

pqname Pointer to a QName structure containing localName, prefix, and namespace URI.

charSet Pointer to restricted character set to be used for encoding the string or NULL if not restricted.

Returns:

Completion status of operation:

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

7.11.2.47 EXTERNEXI int rtEXIEncString2 (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*, const OSUTF8CHAR * *elemName*, struct OSXMLNamespace * *pNS*, const OSUINT16 * *charSet*)

This version of the rtEXIEncString function uses standard generated encode function arguments.

Parameters:

pctxt Pointer to context block structure.

value Pointer to string value to be encoded.

elemName The non-empty element name.

pNS Pointer to namespace structure containing prefix, and namespace URI.

charSet Pointer to restricted character set to be used for encoding the string or NULL if not restricted.

Returns:

Completion status of operation:

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

7.11.2.48 EXTERNEXI int rtEXIEncTime (OSCTXT * *pctxt*, const OSNumDateTime * *pvalue*)

This function encodes a numeric time value.

Parameters:

pctxt Pointer to context block structure.

pvalue Pointer to numeric date/time stricture.

Returns:

Completion status of operation:

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

7.11.2.49 EXTERNEXI int rtEXIEncTimeString (OSCTXT * *pctxt*, const OSUTF8CHAR * *pvalue*)

This function encodes a time string value.

Parameters:

pctxt Pointer to context block structure.

pvalue Time string to be encoded

- The format of time is HH:MM:SS[.frac][TZ]
- The value of HH is 00 - 23
- The value of MM is 00 - 59
- The value of SS is 00 - 59

Returns:

Completion status of operation:

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

7.11.2.50 EXTERNEXI int rtEXIEncUInt64Value (OSCTXT * *pctxt*, OSUINT64 *value*)

This function encodes an 64-bit unsigned integer value.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.51 EXTERNEXI int rtEXIEncUIntCHEvent (OSCTXT * *pctxt*, OSUINT32 *value*)

This function encodes a variable of the XSD unsigned integer type as a CH content event.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.52 EXTERNEXI int rtEXIEncUIntElem (OSCTXT * *pctxt*, OSUINT32 *value*, const OSUTF8CHAR * *elemName*, OSXMLNamespace * *pNS*)

This function encodes a complete XML element (<tag>content</tag>) in which the content is an unsigned integer.

Parameters:

pctxt Pointer to context block structure.

value Integer value to be encoded.

elemName Local name of element.

pNS Pointer to namespace structure containing prefix, and namespace URI.

Returns:

Completion status of operation:

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

7.11.2.53 EXTERNEXI int rtEXIEncUIntValue (OSCTXT * *pctxt*, OSUINT32 *value*)

This function encodes an unsigned integer value.

Parameters:

pctxt Pointer to context block structure.

value Value to be encoded.

Returns:

Completion status of operation:

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

7.11.2.54 EXTERNEXI int rtEXIEncUTF8Str (OSCTXT * *pctxt*, const OSUTF8CHAR * *value*, size_t *lengthIncr*, const OSUINT16 * *charSet*)

This function encodes a UTF-8 character string value.

It does not use the string tables to determine if the actual string value should be encoded or not (i.e. it directly encodes the content).

Parameters:

pctxt Pointer to context block structure.

value Pointer to string value to be encoded.

lengthIncr Value to be added to string length before encoding. This value is normally zero; however, some string table operations require an increment of 1 or 2.

Returns:

Completion status of operation:

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

7.12 rtEXIEncStringTable.h File Reference

EXI encoder string table structure and functions.

```
#include "rtxsrc/rtxContext.h"
#include "rtxsrc/rtxHashMapStr2UInt.h"
#include "rtexisrc/rtEXIExternDefs.h"
#include "rtxsrc/rtxHashMapUndef.h"
```

Classes

- struct OSEXIEncStringTable

  This structure defines the structure of the various string table partitions used by the encoder.

Functions

- EXTERNEXI void rtEXIEncStringTableInit (OSCTXT *pctxt, OSEXIEncStringTable *pstrtab, size_t capacity)

  This function initializes the given string table structure.

- EXTERNEXI OSEXIEncStringTable * rtEXIEncNewStringTable (OSCTXT *pctxt, size_t capacity)

  This function allocates and initializes a new string table structure.

- EXTERNEXI void rtEXIEncStringTableClear (OSCTXT *pctxt, OSEXIEncStringTable *pstrtab)

  This function clears all strings out of the existing table.

- EXTERNEXI OSUINT32 rtEXIEncStringTableAdd (OSCTXT *pctxt, OSEXIEncStringTable *pstrtab, const OSUTF8CHAR *str)

  This function adds a string to the given string table.

- EXTERNEXI OSUINT32 rtEXIEncStringTableGetIndex (OSEXIEncStringTable *pstrtab, const OSUTF8CHAR *str)

  This function gets the index (i.e.

7.12.1 Detailed Description

EXI encoder string table structure and functions.

Definition in file rtEXIEncStringTable.h.

7.13 rtEXIEncStringTables.h File Reference

EXI encoder string table structure and functions.

```
#include "rtexisrc/rtEXIEncStringTable.h"
#include "rtxsrc/rtxHashMap.h"
#include "rtxmlsrc/osrtxml.h"
```

Classes

- struct OSEXIEncStringTables

  This structure defines the complete set of string table partitions used by the encoder.

Functions

- EXTERNEXI void rtEXIEncStrTabsInit (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs)

  This function initializes all EXI string table partitions.

- EXTERNEXI void rtEXIEncStrTabsClear (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs)

  This function clears all EXI string table partitions.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddURI (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function will add a URI to the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetURIID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function will get the compact identifier of the given URI from the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetURITableSize (OSEXIEncStringTables *pstrtabs)

  This function returns the current number of entries in the URI string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddPrefix (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *prefix)

  This function adds the given prefix to the prefix table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetPrefixID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *prefix)

  This function will get the compact identifier of the given prefix from the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetPrefixTableSize (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the prefix string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddLocalName (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *name)

  This function adds the given local name to the local name table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalNameID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri, const OSUTF8CHAR *name)

  This function will get the compact identifier of the given localName from the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalNameTableSize (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *uri)

  This function returns the current number of entries in the localName string table partition identified by the given URI.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddLocalValue (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSXMLFullQName *qname, const OSUTF8CHAR *value)

  This function adds the given local value to the local value table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalValueID (OSEXIEncStringTables *pstrtabs, const OSXMLFullQName *qname, const OSUTF8CHAR *value)

  This function will get the compact identifier of the given local value from the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetLocalValueTableSize (OSEXIEncStringTables *pstrtabs, const OSXMLFullQName *qname)

  This function returns the current number of entries in the local value string table partition identified by the given QName.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsAddGlobalValue (OSCTXT *pctxt, OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *value)

  This function will add a string value to the global value string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetGlobalValueID (OSEXIEncStringTables *pstrtabs, const OSUTF8CHAR *value)

  This function will get the compact identifier of the given string value from the global value string table partition.

- EXTERNEXI OSUINT32 rtEXIEncStrTabsGetGlobalValueTableSize (OSEXIEncStringTables *pstrtabs)

  This function returns the current number of entries in the global value string table partition.

7.13.1 Detailed Description

EXI encoder string table structure and functions.

Definition in file rtEXIEncStringTables.h.

7.14 rtEXIEvent.h File Reference

EXI event definitions and functions.

```
#include "rtexisrc/rtEXIExternDefs.h"
#include "rtxsrc/rtxXmlQName.h"
```

Enumerations

- enum OSEXIEventType

  A structure representing an EXI event.

Functions

- EXTERNEXI void rtEXIEventInit (OSCTXT *pctxt, OSEXIEvent *pEvent, OSEXIEventType type, const OSXMLFullQName *qname)

  This function initializes an event structure.

- EXTERNEXI OSEXIEvent * rtEXINewEvent (OSCTXT *pctxt, OSEXIEventType type, const OSXMLFullQName *qname)

  This function allocates a new event structure and initializes it.

- EXTERNEXI OSEXIEvent * rtEXINewEventDeepCopy (OSCTXT *pctxt, const OSEXIEvent *pEvent)

  This function creates a deep copy of the given event record.

- EXTERNEXI void rtEXIEventDeepCopy (OSCTXT *pctxt, OSEXIEvent *pdest, const OSEXIEvent *psrc)

  This function makes a deep copy of the given Event including the strings inside.

- EXTERNEXI void rtEXIEventFreeMem (OSCTXT *pctxt, OSEXIEvent *pEvent, OSBOOL dynamic)

  This function frees all memory within an Event structure.

- EXTERNEXI OSUINT32 rtEXIEventHash (const OSEXIEvent *pEvent)

  This function computes a hash code for an event structure.

- EXTERNEXI OSBOOL rtEXIEventsEqual (const OSEXIEvent *pEvent1, const OSEXIEvent *pEvent2)

  This function tests if two event structures are equal.

- EXTERNEXI const OSEXIEvent * rtEXIGetBaseEvent (const OSEXIEvent *pEvent)

  This function gets the base event corresponding to a given event.

- EXTERNEXI const char * rtEXIEventTypeToString (OSEXIEventType type)

  This function returns the textual name for the given event type (for example, "SD", "SE", etc.

- EXTERNEXI OSUTF8CHAR * rtEXIEventToString (OSCTXT *pctxt, const OSEXIEvent *pEvent)

  This function returns the full textual description for the given event.

- EXTERNEXI void rtEXIEventPrint (const OSEXIEvent *pEvent)

  This function prints information on the given event to stdout.

7.14.1 Detailed Description

EXI event definitions and functions.

Definition in file rtEXIEvent.h.

7.14.2 Enumeration Type Documentation

7.14.2.1 enum OSEXIEventType

A structure representing an EXI event.

An event is a pair of an event type and a possibly null qname. Only SE and AT events accept a qname parameter. For all other events, the static constants defined in this class must be used. Note that SE(∗) and AT(∗) are represented by SE and AT, respectively.

Definition at line 45 of file rtEXIEvent.h.

7.14.3 Function Documentation

7.14.3.1 EXTERNEXI void rtEXIEventDeepCopy (OSCTXT ∗ *pctxt*, OSEXIEvent ∗ *pdest*, const OSEXIEvent ∗ *psrc*)

This function makes a deep copy of the given Event including the strings inside.

Parameters:

pctxt Pointer to a context structure.

pdest Pointer to Event to receive copied data.

psrc Pointer to Event to be copied.

7.14.3.2 EXTERNEXI void rtEXIEventFreeMem (OSCTXT ∗ *pctxt*, OSEXIEvent ∗ *pEvent*, OSBOOL *dynamic*)

This function frees all memory within an Event structure.

Parameters:

pctxt Pointer to a context structure.

pEvent Pointer to Event in which memory will be freed.

dynamic Boolean indicating if pEvent is dynamic. If true, the memory for pEvent is freed.

7.14.3.3 EXTERNEXI OSUINT32 rtEXIEventHash (const OSEXIEvent ∗ *pEvent*)

This function computes a hash code for an event structure.

Parameters:

pEvent Pointer to event structure.

Returns:

Computed hash code.

7.14.3.4 EXTERNEXI void rtEXIEventInit (OSCTXT * *pctxt*, OSEXIEvent * *pEvent*, OSEXIEventType *type*, const OSXMLFullQName * *qname*)

This function initializes an event structure.

Parameters:

pctxt Pointer to a context block structure.

pEvent Pointer to event structure.

type Type of the event.

qname QName associated with the event (possibly NULL).

7.14.3.5 EXTERNEXI void rtEXIEventPrint (const OSEXIEvent * *pEvent*)

This function prints information on the given event to stdout.

Parameters:

pEvent Pointer to event structure.

7.14.3.6 EXTERNEXI OSBOOL rtEXIEventsEqual (const OSEXIEvent * *pEvent1*, const OSEXIEvent * *pEvent2*)

This function tests if two event structures are equal.

Parameters:

pEvent1 Pointer to event structure.

pEvent2 Pointer to event structure.

Returns:

True if events are equal.

7.14.3.7 EXTERNEXI OSUTF8CHAR* rtEXIEventToString (OSCTXT * *pctxt*, const OSEXIEvent * *pEvent*)

This function returns the full textual description for the given event.

This includes the type (for example, "SD", "SE", etc.) and, if SE or AT, the local name or '*'.

Parameters:

pctxt Pointer to a context structure.

pEvent Pointer to Event structure.

Returns:

Text corresponding to the type.

7.14.3.8 EXTERNEXI const char* rtEXIEventTypeToString (OSEXIEventType *type*)

This function returns the textual name for the given event type (for example, "SD", "SE", etc.

). The code is returned as string literal text.

Parameters:

type Enumerated event type.

Returns:

Text corresponding to the type.

7.14.3.9 EXTERNEXI const OSEXIEvent* rtEXIGetBaseEvent (const OSEXIEvent * *pEvent*)

This function gets the base event corresponding to a given event.

For SE or AT events, this is SE(*) or AT(*) respectively. For all other events, it is the event itself.

Parameters:

pEvent Pointer to event structure.

Returns:

Pointer to base event.

7.14.3.10 EXTERNEXI OSEXIEvent* rtEXINewEvent (OSCTXT * *pctxt*, OSEXIEventType *type*, const OSXMLFullQName * *qname*)

This function allocates a new event structure and initializes it.

Parameters:

pctxt Pointer to a context block structure.

type Type of the event.

qname QName associated with the event (possibly NULL).

Returns:

Pointer to allocated event structure or NULL if no dynamic memory available.

7.14.3.11 EXTERNEXI OSEXIEvent* rtEXINewEventDeepCopy (OSCTXT * *pctxt*, const OSEXIEvent * *pEvent*)

This function creates a deep copy of the given event record.

Memory for the record is allocated using rtxMemAlloc. The record should be freed using rtEXIEventFreeMem.

Parameters:

pctxt Pointer to a context block structure.

pEvent Pointer to event record to be copied.

Returns:

Pointer to allocated event structure or NULL if no dynamic memory available.

7.15 rtEXIEventCode.h File Reference

EXI event code definitions and functions.

```
#include "rtexisrc/rtEXIExternDefs.h"
#include "rtxsrc/rtxContext.h"
```

Classes

- struct OSEXIEventCode

  A structure representing a production's event code.

Defines

- #define rtEXISetEventCode1(pEventCode, part1) rtEXISetEventCode3(pEventCode, part1, OSINT32_MIN, OSINT32_MIN)

  This macro sets a one-part event code.

- #define rtEXISetEventCode2(pEventCode, part1, part2) rtEXISetEventCode3(pEventCode, part1, part2, OSINT32_MIN)

  This macro sets a two-part event code.

Functions

- EXTERNEXI int rtEXIEventCodeCompare (const OSEXIEventCode *pec1, const OSEXIEventCode *pec2)

  This function compares two event codes.

- EXTERNEXI OSEXIEventCode * rtEXIEventCodeCopy (OSCTXT *pctxt, const OSEXIEventCode *pec)

  This function does a deep-copy of an event code structure.

- EXTERNEXI OSBOOL rtEXIEventCodesEqual (const OSEXIEventCode *pec1, const OSEXIEventCode *pec2)

  This function compares two event codes for equality.

- EXTERNEXI char * rtEXIEventCodeToString (OSCTXT *pctxt, const OSEXIEventCode *pec)

  This function returns a string representation of the given event code in dot notation (part1.part2.part3).

- EXTERNEXI OSUINT32 rtEXIEventCodeLength (const OSEXIEventCode *pec)

  This function returns the length of the given event code.

- EXTERNEXI OSEXIEventCode * rtEXINewEventCode1 (OSCTXT *pctxt, OSINT32 part1)

  This function allocates and initializes a one-part event code.

- EXTERNEXI OSEXIEventCode * rtEXINewEventCode2 (OSCTXT *pctxt, OSINT32 part1, OSINT32 part2)

  This function allocates and initializes a two-part event code.

- EXTERNEXI OSEXIEventCode * rtEXINewEventCode3 (OSCTXT *pctxt, OSINT32 part1, OSINT32 part2, OSINT32 part3)

  This function allocates and initializes a three-part event code.

- EXTERNEXI void rtEXISetEventCode3 (OSEXIEventCode *pEventCode, OSINT32 part1, OSINT32 part2, OSINT32 part3)

  This function sets a three-part event code.

- EXTERNEXI void rtEXIEventCodePrint (const OSEXIEventCode *pEventCode)

  This function prints information on the given event code to stdout.

7.15.1 Detailed Description

EXI event code definitions and functions.

Definition in file rtEXIEventCode.h.

7.16 rtEXIEventCodeGroup.h File Reference

EXI event code definitions and functions.

```
#include "rtexisrc/rtEXIEventCode.h"
#include "rtxsrc/rtxDynBitSet.h"
#include "rtxsrc/rtxSList.h"
```

Classes

- struct OSEXIEventCodeGroup

  An EventCodeGroup is a group of related event codes.

Defines

- #define rtEXIEventCodeGroupHasPart1(pecgrp, part1) rtxDynBitSetTestBit(&pecgrp → part1Set,part1)

  This macro returns true if there is an event code in this group whose length is 1 and whose first part is equal to part1.

- #define rtEXIEventCodeGroupHasPart2(pecgrp, part2) rtxDynBitSetTestBit(&pecgrp → part2Set,part2)

  This macro returns true if there is an event code in this group whose length is 2 and whose first part is equal to part2.

Functions

- EXTERNEXI void rtEXIEventCodeGroupInit (OSCTXT *pctxt, OSEXIEventCodeGroup *pecgrp)

  This function initializes an event code group structure.

- EXTERNEXI OSEXIEventCodeGroup * rtEXINewEventCodeGroup (OSCTXT *pctxt)

  This function allocates and initializes a new event code group structure.

- EXTERNEXI int rtEXIEventCodeGroupAdd (OSEXIEventCodeGroup *pecgrp, const OSEXIEventCode *pec)

  This function adds an event code to an event code group and updates the maximum part variables.

- EXTERNEXI OSEXIEventCodeGroup * rtEXIEventCodeGroupCopy (const OSEXIEventCodeGroup *pecgrp)

  This function performs a deep copy of the given event group.

- EXTERNEXI void rtEXIEventCodeGroupFreeMem (OSEXIEventCodeGroup *pecgrp)

  This function frees all memory associated with an event group.

- EXTERNEXI int rtEXIEventCodeGroupGetBitsPart1 (OSEXIEventCodeGroup *pecgrp)

  This function returns the number of bits necessary to encode the max part1 value in this group.

- EXTERNEXI int rtEXIEventCodeGroupGetBitsPart2 (OSEXIEventCodeGroup *pecgrp)

  This function returns the number of bits necessary to encode the max part2 value in this group.

- EXTERNEXI int rtEXIEventCodeGroupGetBitsPart3 (OSEXIEventCodeGroup *pecgrp)

This function returns the number of bits necessary to encode the max part3 value in this group.

- EXTERNEXI void rtEXIEventCodeGroupIncrPart1 (OSCTXT *pctxt, OSEXIEventCodeGroup *pecgrp)

  This function increments part1 in all event codes in this group, as well as the max part1 value.

7.16.1 Detailed Description

EXI event code definitions and functions.

Definition in file rtEXIEventCodeGroup.h.

7.17 rtEXIExternDefs.h File Reference

EXI external function declaration macros for building Windows DLL's.

7.17.1 Detailed Description

EXI external function declaration macros for building Windows DLL's.

Definition in file rtEXIExternDefs.h.

7.18 rtEXIStateTable.h File Reference

EXI state table for schema-informed decoding.

`#include "rtexisrc/rtEXIEvent.h"`

Functions

- EXTERNEXI OSEXIStateTable * rtEXINewStateTable (OSCTXT *pctxt, const OSEXIStateTableRecord *pStabRecArray, OSUINT16 numStabRecs)

  This allocates and initializes a new state table structure.

- EXTERNEXI void rtEXIInitStateTableRecord (OSEXIStateTableRecord *pRecord, OSUINT16 eventId, OSUINT16 stateId, OSINT16 nextStateIdx, OSINT16 repStateIdx, const OSUTF8CHAR *localName, OSEXIEventType eventType, OSUINT8 eventCode, OSUINT8 eventNBits, OSUINT8 maxEvtCode, OSUINT32 maxRepeats)

  This function sets all items in the given state table record to the given values.

- EXTERNEXI OSINT16 rtEXIStateTransition (OSCTXT *pctxt)

  This function transitions from the current encode state in the state table to the next encode state.

- EXTERNEXI OSINT16 rtEXILookupEventInState (OSCTXT *pctxt, OSUINT16 eventId)

  This function is used to lookup the given event ID in the state table.

- EXTERNEXI OSINT16 rtEXILookupEventTypeInState (OSCTXT *pctxt, OSEXIEventType eventType)

  This function is used to lookup the given event type in the state table.

7.18.1 Detailed Description

EXI state table for schema-informed decoding.

Definition in file rtEXIStateTable.h.

7.18.2 Function Documentation

7.18.2.1 EXTERNEXI void rtEXIInitStateTableRecord (OSEXIStateTableRecord * *pRecord*, OSUINT16 *eventId*, OSUINT16 *stateId*, OSINT16 *nextStateIdx*, OSINT16 *repStateIdx*, const OSUTF8CHAR * *localName*, OSEXIEventType *eventType*, OSUINT8 *eventCode*, OSUINT8 *eventNBits*, OSUINT8 *maxEvtCode*, OSUINT32 *maxRepeats*)

This function sets all items in the given state table record to the given values.

Parameters:

pRecord Pointer to state table record structure.

eventId A unique event identifier.

stateId A unique state identifier.

nextStateIdx Next state index.

repStateIdx Repeating state index.

localName Local name for start element or attribute event.

eventType Event type (SE, AT, EE, etc.)

eventCode Part 1 of event code (other parts are not used).

eventNBits Number of bits eventCode to be encoded into.

maxEvtCode Maximum event code for this state.

maxRepeats Maximum number of times this state can repeat.

7.18.2.2 EXTERNEXI OSINT16 rtEXILookupEventInState (OSCTXT * *pctxt*, OSUINT16 *eventId*)

This function is used to lookup the given event ID in the state table.

It is assumed state table information in the EXI context is set to point at the first state record in a group.

Parameters:

pctxt Pointer to OSCTXT structure

eventId ID of event to be searched for.

Returns:

Index to row containing event or -1 if not found.

7.18.2.3 EXTERNEXI OSINT16 rtEXILookupEventTypeInState (OSCTXT * *pctxt*, OSEXIEventType *eventType*)

This function is used to lookup the given event type in the state table.

It is assumed state table information in the EXI context is set to point at the first state record in a group.

Parameters:

pctxt Pointer to OSCTXT structure

eventType Type of event (SE, AT, EE, etc.)

Returns:

Index to row containing event or -1 if not found.

7.18.2.4 EXTERNEXI OSEXIStateTable* rtEXINewStateTable (OSCTXT * *pctxt*, const OSEXIStateTableRecord * *pStabRecArray*, OSUINT16 *numStabRecs*)

This allocates and initializes a new state table structure.

Parameters:

pctxt Pointer to OSCTXT structure

pStabRecArray Pointer to array of state table records.

numStabRecs Number of records in state table array.

Returns:

Pointer to allocated array or NULL if no memory.

7.18.2.5 EXTERNEXI OSINT16 rtEXIStateTransition (OSCTXT * *pctxt*)

This function transitions from the current encode state in the state table to the next encode state.

For encoding, a repeating state is handled as part of the transition.

Parameters:

pctxt Pointer to OSCTXT structure

Returns:

Index to next state or -1 if end or error.

7.19 rtXML2EXI.h File Reference

Functions for serializing XML data into EXI format.

```
#include "rtexisrc/osrtexi.h"
```

Functions

- EXTERNEXI int rtXmlMem2ExiMem (OSCTXT *pctxt, OSOCTET *outbuf, size_t outbufsiz)

  This function serializes XML data to EXI data and stores the result in the given memory buffer.

- EXTERNEXI int rtXmlFile2ExiStream (const char *xmlFileName, OSCTXT *pExiCtxt)

  This function serializes XML data from a file to EXI data and outputs the result to the output stream defined within the given output context.

7.19.1 Detailed Description

Functions for serializing XML data into EXI format.

Definition in file rtXML2EXI.h.

7.20 rtXML2EXISAX.h File Reference

SAX callback functions for serializing XML data into EXI format.

```
#include "rtexisrc/osrtexi.h"
```

Functions

- EXTERNEXI int xml2ExiStartElement (void *userData, const OSUTF8CHAR *localname, const OSUTF8CHAR *qname, const OSUTF8CHAR *const *attrs)

  SAX start element handler callback function.

- EXTERNEXI int xml2ExiCharacters (void *userData, const OSUTF8CHAR *chars, int length)

  SAX characters handler callback function definition.

- EXTERNEXI int xml2ExiEndElement (void *userData, const OSUTF8CHAR *localname, const OSUTF8CHAR *qname)

  SAX end element handler callback function definition.

7.20.1 Detailed Description

SAX callback functions for serializing XML data into EXI format.

Definition in file rtXML2EXISAX.h.

Index