Kvaser Memorator 2nd Generation Configuration XML API

Generated by Doxygen 1.7.3

Sun May 22 2016 19:52:14

Contents

Chapter 1

Data Structure Documentation

1.1 KvParseHandle Struct Reference

Handle used when parsing postfix expressions (deprecated):

```
#include <kvamemolibxml.h>
```

Data Fields

- Token ∗ next

1.1.1 Detailed Description

Handle used when parsing postfix expressions (deprecated):

1.1.2 Field Documentation

1.1.2.1 Token∗ next

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

- kvamemolibxml.h

1.2 tag_token Struct Reference

Token used when parsing postfix expressions (deprecated):

```
#include <kvamemolibxml.h>
```

Data Fields

- int type
- char ∗ name
- struct tag_token ∗ left
- struct tag_token ∗ right
- int start_pos
- int end_pos
- struct tag_token ∗ next
- int errCode

1.2.1 Detailed Description

Token used when parsing postfix expressions (deprecated):

1.2.2 Field Documentation

1.2.2.1 int **end_pos**

1.2.2.2 int **errCode**

1.2.2.3 struct **tag_token∗ left**

1.2.2.4 char∗ **name**

1.2.2.5 struct **tag_token∗ next**

1.2.2.6 struct **tag_token∗ right**

1.2.2.7 int **start_pos**

1.2.2.8 int **type**

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

- kvamemolibxml.h

Chapter 2

File Documentation

2.1 kvamemolibxml.h File Reference

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

Data Structures

- struct tag_token

  Token used when parsing postfix expressions (deprecated):

- struct KvParseHandle

  Handle used when parsing postfix expressions (deprecated):

Defines

XML_ERROR_MESSAGE_LENGTH

Maximum length of the xml error message string.

- #define XML_ERROR_MESSAGE_LENGTH 2048

Typedefs

- typedef struct tag_token Token

Enumerations

KvaXmlStatus

Generally, a return code greater than or equal to zero means success. A value less than zero means failure.

- enum KvaXmlStatus {
  KvaXmlStatusOK = 0,
  KvaXmlStatusFail = -1,
  KvaXmlStatusERR_ATTR_NOT_FOUND = -3,
  KvaXmlStatusERR_ATTR_VALUE = -4,
  KvaXmlStatusERR_ELEM_NOT_FOUND = -5,
  KvaXmlStatusERR_VALUE_RANGE = -6,
  KvaXmlStatusERR_VALUE_UNIQUE = -7,
  KvaXmlStatusERR_VALUE_CONSECUTIVE = -8,
  KvaXmlStatusERR_EXPRESSION = -9,
  KvaXmlStatusERR_XML_PARSER = -10,
  KvaXmlStatusERR_DTD_VALIDATION = -11,
  KvaXmlStatusERR_SCRIPT_ERROR = -12,
  KvaXmlStatusERR_INTERNAL = -20 }

KvaXmlValidationStatus

Generally, a return code greater than or equal to zero means success. A value less than zero means failure.

- enum KvaXmlValidationStatus {
  KvaXmlValidationStatusOK = 0,
  KvaXmlValidationStatusFail = -1,
  KvaXmlValidationStatusERR_ABORT = -2,
  KvaXmlValidationStatusERR_SILENT_TRANSMIT = -3,
  KvaXmlValidationStatusERR_UNDEFINED_TRIGGER = -4,
  KvaXmlValidationStatusERR_MULTIPLE_EXT_TRIGGER = -5,
  KvaXmlValidationStatusERR_MULTIPLE_START_TRIGGER = -6,
  KvaXmlValidationStatusERR_DISK_FULL_STARTS_LOG = -7,
  KvaXmlValidationStatusERR_NUM_OUT_OF_RANGE = -8,
  KvaXmlValidationStatusERR_SCRIPT_NOT_FOUND = -9,
  KvaXmlValidationStatusERR_SCRIPT_TOO_LARGE = -10,
  KvaXmlValidationStatusERR_SCRIPT_TOO_MANY = -11,
  KvaXmlValidationStatusERR_SCRIPT_CONFLICT = -12,
  KvaXmlValidationStatusERR_ELEMENT_COUNT = -13,
  KvaXmlValidationStatusWARN_ABORT = -100,
  KvaXmlValidationStatusWARN_NO_ACTIVE_LOG = -101,
  KvaXmlValidationStatusWARN_DISK_FULL_AND_FIFO = -102,
  KvaXmlValidationStatusWARN_IGNORED_ELEMENT = -103 }

Functions

- KvaXmlStatus kvaXmlInitialize (void)
- KvaXmlStatus kvaXmlGetLastError (char ∗buf, unsigned int len, KvaXmlStatus ∗err)
- KvaXmlStatus kvaXmlToBuffer (const char ∗xmlbuf, unsigned int xmllen, char ∗outbuf, unsigned int ∗outlen, long ∗version)
- KvaXmlStatus kvaXmlToFile (const char ∗infile, const char ∗outfile)
- KvaXmlStatus kvaFileToXml (const char ∗parfile, const char ∗xmlfile)
- KvaXmlStatus kvaXmlDebugOutput (int on)
- KvaXmlStatus kvaBufferToXml (const char ∗inbuf, unsigned int inlen, char ∗xmlbuf, unsigned int ∗xmllen, long ∗version, const char ∗scriptpath)
- KvaXmlStatus kvaXmlValidate (const char ∗xmlbuf, unsigned int xmllen)
- KvaXmlStatus kvaXmlGetValidationStatusCount (int ∗countErr, int ∗countWarn)
- KvaXmlStatus kvaXmlGetValidationError (KvaXmlValidationStatus ∗status, char ∗buf, unsigned int len)
- KvaXmlStatus kvaXmlGetValidationWarning (KvaXmlValidationStatus ∗status, char ∗buf, unsigned int len)
- KvaXmlStatus kvaXmlGetErrorText (KvaXmlStatus status, char ∗buf, unsigned int len)
- KvaXmlStatus kvaXmlGetValidationText (KvaXmlValidationStatus status, char ∗buf, unsigned int len)
- unsigned short kvaXmlGetVersion (void)
- KvParseHandle ∗ kvaToolsParseCreate (void)
- void kvaToolsParseDestroy (KvParseHandle ∗h)
- void kvaToolsExprGetErrorString (int errCode, char ∗s, size_t bufsiz)
- int kvaToolsParseExpr (KvParseHandle ∗h, char ∗expr, Token ∗∗t)
- int kvaToolsFreeExpr (KvParseHandle ∗h, Token ∗t)
- int kvaToolsDumpExpr (KvParseHandle ∗h, Token ∗t)
- int kvaToolsExprHasErrors (KvParseHandle ∗h, Token ∗t)
- int kvaToolsExprGetError (KvParseHandle ∗h, Token ∗t, int ∗errCode, int ∗pos)

2.1.1 Define Documentation

2.1.1.1 #define XML_ERROR_MESSAGE_LENGTH 2048

Maximum length of the xml error message string.

2.1.2 Typedef Documentation

2.1.2.1 typedef struct tag_token Token

Token used when parsing postfix expressions (deprecated):

2.1.3 Enumeration Type Documentation

2.1.3.1 enum KvaXmlStatus

Enumerator:

- ***KvaXmlStatusOK*** OK.
- ***KvaXmlStatusFail*** Generic error.
- ***KvaXmlStatusERR_ATTR_NOT_FOUND*** Failed to find an attribute in a node.
- ***KvaXmlStatusERR_ATTR_VALUE*** The attribute value is not correct, e.g. whitespace after a number.
- ***KvaXmlStatusERR_ELEM_NOT_FOUND*** Could not find a required element.
- ***KvaXmlStatusERR_VALUE_RANGE*** The value is outside the allowed range.
- ***KvaXmlStatusERR_VALUE_UNIQUE*** The value is not unique; usually idx attributes.
- ***KvaXmlStatusERR_VALUE_CONSECUTIVE*** The values are not consecutive; usually idx attributes.
- ***KvaXmlStatusERR_EXPRESSION*** The trigger expression could not be parsed.
- ***KvaXmlStatusERR_XML_PARSER*** The XML settings contain syntax errors.
- ***KvaXmlStatusERR_DTD_VALIDATION*** The XML settings do not follow the DTD.
- ***KvaXmlStatusERR_SCRIPT_ERROR*** t-script related errors, e.g. file not found.
- ***KvaXmlStatusERR_INTERNAL*** Internal errors, e.g. null pointers.

2.1.3.2 enum KvaXmlValidationStatus

Enumerator:

- ***KvaXmlValidationStatusOK*** OK.
- ***KvaXmlValidationStatusFail*** Generic error.
- ***KvaXmlValidationStatusERR_ABORT*** Too many errors, validation aborted.
- ***KvaXmlValidationStatusERR_SILENT_TRANSMIT*** Transmit lists used in silent mode.
- ***KvaXmlValidationStatusERR_UNDEFINED_TRIGGER*** An undefined trigger is used in an expression.
- ***KvaXmlValidationStatusERR_MULTIPLE_EXT_TRIGGER*** There are more than one external trigger defined.
- ***KvaXmlValidationStatusERR_MULTIPLE_START_TRIGGER*** There are more than one start up trigger defined.
- ***KvaXmlValidationStatusERR_DISK_FULL_STARTS_LOG*** A trigger on disk full starts the logging.

KvaXmlValidationStatusERR_NUM_OUT_OF_RANGE A numerical value is out of range.

KvaXmlValidationStatusERR_SCRIPT_NOT_FOUND A t-script file could not be opened.

KvaXmlValidationStatusERR_SCRIPT_TOO_LARGE A t-script is too large for the configuration.

KvaXmlValidationStatusERR_SCRIPT_TOO_MANY Too many active t-scripts for selected device.

KvaXmlValidationStatusERR_SCRIPT_CONFLICT More than one active script is set as 'primary'.

KvaXmlValidationStatusERR_ELEMENT_COUNT Too many or too few elements of this type.

KvaXmlValidationStatusWARN_ABORT Too many warnings, validation aborted.

KvaXmlValidationStatusWARN_NO_ACTIVE_LOG No active logging detected.

KvaXmlValidationStatusWARN_DISK_FULL_AND_FIFO A trigger on disk full used with FIFO mode.

KvaXmlValidationStatusWARN_IGNORED_ELEMENT This XML element was ignored.

2.1.4 Function Documentation

2.1.4.1 KvaXmlStatus kvaBufferToXml (const char ∗ *inbuf,* unsigned int *inlen,* char ∗ *xmlbuf,* unsigned int ∗ *xmllen,* long ∗ *version,* const char ∗ *scriptpath*)

Convert a buffer containing param.lif with size inlen to a new XML settings buffer xmlbuf with length xmllen. The version of the XML settings is returned in version (Upper 16 bits: major, lower 16 bits: minor). Scripts from the param.lif will be written to the directory specified in scriptpath.

Parameters

`in`	*inbuf*	Buffer containing the param.lif settings.
`in`	*inlen*	Size of param.lif buffer in bytes.
`out`	*xmlbuf*	Buffer to receive the XML settings.
`out`	*xmllen*	Size of the XML buffer in bytes.
`out`	*version*	XML version.
`in`	*scriptpath*	Path to destination of scripts.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.2 KvaXmlStatus kvaFileToXml (const char * *parfile*, const char * *xmlfile*)

Convert the binary settings from parfile and write the XML settings to xmlfile.

Parameters

in	*parfile*	Path and name of the file containing the param.lif settings.
out	*xmlfile*	Path and name of the file to receive the XML settings.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.3 int kvaToolsDumpExpr (KvParseHandle * *h*, Token * *t*)

Dump Token tree contents if debug is enabled, otherwise do nothing (deprecated).

Parameters

in	*h*	KvParseHandle to parser.
in	*t*	Pointer to Token t.

Returns

0.

2.1.4.4 int kvaToolsExprGetError (KvParseHandle * *h*, Token * *t*, int * *errCode*, int * *pos*)

Returns errorcode of first error found in Token tree (deprecated).

Parameters

in	*h*	KvParseHandle to parser.
in	*t*	Pointer to Token t.
out	*errCode*	Error code of first found error token.
out	*pos*	Position of first found error token.

Returns

TRUE if Token tree contains any error tokens
FALSE if Token tree contains no error tokens

2.1.4.5 void kvaToolsExprGetErrorString (int *errCode*, char * *s*, size_t *bufsiz*)

Get a human readable description of errors that occur when parsing a postfix expression (deprecated).

Parameters

in	*errCode*	Error code from parser.
out	*s*	Buffer to receive error message.
in	*bufsiz*	Buffer size in bytes.

2.1.4.6 int kvaToolsExprHasErrors (KvParseHandle ∗ *h*, Token ∗ *t*)

Returns True if Token tree representation of trigger expression has errors (deprecated).

Parameters

in	*h*	KvParseHandle to parser.
in	*t*	Pointer to Token t.

Returns

TRUE if Token tree contains any error tokens
FALSE if Token tree contains no error tokens

2.1.4.7 int kvaToolsFreeExpr (KvParseHandle ∗ *h*, Token ∗ *t*)

Do nothing (deprecated).

Parameters

in	*h*	KvParseHandle to parser.
in	*t*	Pointer to Token t.

Returns

0.

2.1.4.8 KvParseHandle∗ kvaToolsParseCreate (void)

C#

KvParseHandle kvaToolsParseCreate(void);

Create a parser, the start of a linked list of Tokens (deprecated).

Returns

KvParseHandle of new parser.

2.1.4.9 void kvaToolsParseDestroy (KvParseHandle ∗ *h*)

Destroy a parser with handle h and any linked Tokens (deprecated).

Parameters

in	*h*	KvParseHandle to parser to be destroyed.

2.1.4.10 int kvaToolsParseExpr (KvParseHandle ∗ *h*, char ∗ *expr*, Token ∗∗ *t*)

Parse postifix expression expr and return a Token tree representation (deprecated).

Parameters

in	*h*	KvParseHandle to parser.
out	*expr*	String representation of postfix expression.
in	*t*	Pointer to Token tree.

Returns

0.

2.1.4.11 KvaXmlStatus kvaXmlDebugOutput (int *on*)

Enable detailed information about the XML conversion on standard out. This can be very useful when the error that causes the failure is masked by subsequent errors.

Parameters

in	*on*	Enable debug output if non-zero.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.12 KvaXmlStatus kvaXmlGetErrorText (KvaXmlStatus *status*, char ∗ *buf*, unsigned int *len*)

Get a human readable description of error with supplied error code.

Parameters

in	*status*	KvaXmlStatus error code.
out	*buf*	Buffer to receive error message.
in	*len*	Buffer size in bytes.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.13 KvaXmlStatus kvaXmlGetLastError (char ∗ *buf*, unsigned int *len*, KvaXmlStatus ∗ *err*)

Get the last error message (if any) from the conversion in human readable format. Use the macro XML_ERROR_MESSAGE_LENGTH to allocate the buffer buf.

Parameters

out	*buf*	Buffer to receive error text.
in	*len*	Buffer size in bytes.
in	*err*	The error code to convert.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.14 KvaXmlStatus kvaXmlGetValidationError (KvaXmlValidationStatus ∗ *status*, char ∗ *buf*, unsigned int *len*)

Get the validation errors (if any). Call after kvaXmlValidate() until KvaXmlValidationStatusOK

Parameters

out	*status*	Valdiation status code.
out	*buf*	Buffer containing the validation error message.
out	*len*	Size of the validation message buffer in bytes.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.15 KvaXmlStatus kvaXmlGetValidationStatusCount (int ∗ *countErr*, int ∗ *countWarn*)

Get the number of validation statuses (if any). Call after kvaXmlValidate()

Parameters

out	*countWarn*	Number of XML valditation errors.
out	*countErr*	Number of XML validation warnings.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.16 KvaXmlStatus kvaXmlGetValidationText (KvaXmlValidationStatus *status,* char ∗ *buf,* unsigned int *len*)

Get a human readable description of validation error with supplied error code.

Parameters

`in`	*status*	KvaXmlValidationStatus error code.
`out`	*buf*	Buffer to receive error message.
`in`	*len*	Buffer size in bytes.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.17 KvaXmlStatus kvaXmlGetValidationWarning (KvaXmlValidationStatus ∗ *status,* char ∗ *buf,* unsigned int *len*)

Get the validation warnings (if any). Call after kvaXmlValidate() until KvaXmlValidationStatusOK

Parameters

`out`	*status*	Valdiation status code.
`out`	*buf*	Buffer containing the validation warning message.
`out`	*len*	Size of the validation message buffer in bytes.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.18 unsigned short kvaXmlGetVersion (void)

C#

KvaXmlStatus kvaXmlGetVersion(void);

Return the version of the kvaMemoLibXML DLL. The most significant byte is the major version number and the least significant byte is the minor version number.

Returns

Version of the kvaMemoLibXML DLL.

2.1.4.19 KvaXmlStatus kvaXmlInitialize (void)

C#

KvaXmlStatus kvaXmlInitialize(void);

This function must be called before any other functions are used. It will initialize the kvamemolibxml library.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.20 KvaXmlStatus kvaXmlToBuffer (const char ∗ *xmlbuf*, unsigned int *xmllen*, char ∗ *outbuf*, unsigned int ∗ *outlen*, long ∗ *version*)

Convert the XML settings from buffer xmlbuf with length xmllen. The resulting param.lif is written to the buffer outbuf and has length outlen. Use the macro PARAMLIF_-SIZE to allocate the ouput buffer to ensure that it is sufficiently large. The version of the XML settings is returned in version (Upper 16 bits: major, lower 16 bits: minor).

Parameters

in	*xmlbuf*	Buffer containing the XML settings.
in	*xmllen*	Size of the XML buffer in bytes.
out	*outbuf*	Buffer to receive the param.lif settings.
out	*outlen*	Size of the param.lif buffer in bytes.
out	*version*	XML version.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.21 KvaXmlStatus kvaXmlToFile (const char ∗ *infile*, const char ∗ *outfile*)

Convert the XML settings from infile and write the binary settings to outfile.

Parameters

in	*infile*	Path and name of the file containing the XML settings.
out	*outfile*	Path and name of the file to receive the param.lif settings.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.

2.1.4.22 KvaXmlStatus kvaXmlValidate (const char * *xmlbuf,* unsigned int *xmllen*)

Validate a buffer with XML settings

Parameters

`in`	*xmlbuf*	Buffer containing the XML settings.
`in`	*xmllen*	Size of the XML buffer in bytes.

Returns

KvaXmlStatusOK (zero) if success.
KvaXmlStatusERR_XXX (negative) if failure.