XBinder

XML Schema Compiler
Version 2.7
C JSON Runtime
Reference Manual

Objective Systems, Inc. — August 2021

Author's Contact Information

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

Contents

Chapter 1

C JSON Runtime Library Functions

The **C run-time JSON library** contains functions used to encode/decode data in Javascript object notation (JSON). These functions are identified by their *rtJson* prefixes.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

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

Chapter 6

Module Documentation

6.1 JSON encode functions.

Macros

- #define rtJsonEncIntValue rtxTxtWriteInt

  This function encodes a variable of the XSD integer type.
- #define rtJsonEncInt64Value rtxTxtWriteInt64

  This function encodes a variable of the XSD integer type.
- #define rtJsonEncDecrIndent rtxIndentDecr

  This decreases the indentation level set in the given context by updating the indent member.
- #define rtJsonEncIncrIndent rtxIndentIncr

  This increases the indentation level set in the given context by updating the indent member.
- #define rtJsonEncResetIndent rtxIndentReset

  This resets the indentation level in the given context to zero.
- #define rtJsonGetIndentLevels rtxGetIndentLevels

  This returns the number of levels of indentation rtJsonEncIndent is using.
- #define rtJsonEncUIntValue rtxTxtWriteUInt

  This function encodes a variable of the XSD unsigned integer type.
- #define rtJsonEncUInt64Value rtxTxtWriteUInt64

  This function encodes a variable of the XSD integer type.

Functions

- EXTERNJSON int rtJsonEncAnyAttr (OSCTXT ∗pctxt, const OSRTDList ∗pvalue)

  This function encodes a list of OSAnyAttr attributes in which the name and value are given as a UTF-8 string.
- EXTERNJSON int rtJsonEncBase64StrValue (OSCTXT ∗pctxt, OSSIZE nocts, const OSOCTET ∗value)

  This function encodes a variable of the XSD base64Binary type.
- EXTERNJSON int rtJsonEncBoolValue (OSCTXT ∗pctxt, OSBOOL value)

  This function encodes a variable of the XSD boolean type.
- EXTERNJSON int rtJsonEncGYear (OSCTXT ∗pctxt, const OSXSDDateTime ∗pvalue)

This function encodes a numeric gYear value into a JSON string representation.

- EXTERNJSON int rtJsonEncGYearMonth (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gYearMonth value into a JSON string representation.
- EXTERNJSON int rtJsonEncGMonth (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gMonth value into a JSON string representation.
- EXTERNJSON int rtJsonEncGMonthDay (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gMonthDay value into a JSON string representation.
- EXTERNJSON int rtJsonEncGDay (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gDay value into a JSON string representation.
- EXTERNJSON int rtJsonEncDate (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a variable of the XSD 'date' type as a string.
- EXTERNJSON int rtJsonEncTime (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a variable of the XSD 'time' type as a JSON string.
- EXTERNJSON int rtJsonEncDateTime (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric date/time value into a string representation.
- EXTERNJSON int rtJsonEncDecimalValue (OSCTXT *pctxt, OSREAL value, const OSDecimalFmt *pFmtSpec)

  This function encodes a value of the XSD decimal type.
- EXTERNJSON int rtJsonEncDoubleValue (OSCTXT *pctxt, OSREAL value, const OSDoubleFmt *pFmtSpec)

  This function encodes a value of the XSD double or float type.
- EXTERNJSON int rtJsonEncFloatValue (OSCTXT *pctxt, OSREAL value, const OSDoubleFmt *pFmtSpec)

  This function encodes a variable of the XSD float type.
- EXTERNJSON int rtJsonEncHexStr (OSCTXT *pctxt, OSSIZE nocts, const OSOCTET *data)

  This function encodes the given data as a JSON string, using a hexadecimal representation of the data.
- EXTERNJSON int rtJsonEncHexValue (OSCTXT *pctxt, OSSIZE nocts, const OSOCTET *data)

  This function encodes the given data as a sequence of hexadecimal characters.
- EXTERNJSON int rtJsonEncBitStrValueV72 (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data)

  This function encodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonEncBitStrValueExtV72 (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data, OSS↩ IZE dataSize, const OSOCTET *extData)

  This function encodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonEncBitStrValue (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data)

  This function encodes a variable of the ASN.1 Bit string type as a JSON object, as required by X.697 for a BIT STRING without a fixed length.
- EXTERNJSON int rtJsonEncFixedBitStrValue (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data)

  This function encodes a variable of the ASN.1 Bit string type as a JSON string, as required by X.697 for a BIT STRING with a fixed length.
- EXTERNJSON int rtJsonEncBitStrValueExt (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data, OSSIZE dataSize, const OSOCTET *extData)

  This function encodes a variable of the ASN.1 Bit string type as a JSON object, as required by X.697 for a BIT STRING without a fixed length.
- EXTERNJSON int rtJsonEncIndent (OSCTXT *pctxt)

  This function:
- EXTERNJSON int rtJsonEncStringObject (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value)

  This function encodes a JSON object containing a string value.
- EXTERNJSON int rtJsonEncStringObject2 (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const OSUTF8CHAR *value, size_t valueLen)

  This function encodes a JSON object containing a string value.

- EXTERNJSON int rtJsonEncStringPair (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value)

  This function encodes a name/value pair.
- EXTERNJSON int rtJsonEncStringPair2 (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const OSUTF8CHAR *value, size_t valueLen)

  This function encodes a name/value pair.
- EXTERNJSON int rtJsonEncStringValue (OSCTXT *pctxt, const OSUTF8CHAR *value)

  This function encodes a UTF8 string value.
- EXTERNJSON int rtJsonEncStringValue2 (OSCTXT *pctxt, const OSUTF8CHAR *value, size_t valueLen)

  This function encodes a UTF8 string value.
- EXTERNJSON int rtJsonEncChars (OSCTXT *pctxt, const char *value, OSSIZE valueLen)

  This function encodes the given number of characters from a character string value as a JSON string.
- EXTERNJSON int rtJsonEncCharStr (OSCTXT *pctxt, const char *value)

  This function encodes a character string value as a JSON string.
- EXTERNJSON int rtJsonEncStringNull (OSCTXT *pctxt)

  This function encodes an asn.1 NULL type as string.
- EXTERNJSON int rtJsonEncStringRaw (OSCTXT *pctxt, const OSUTF8CHAR *value)

  This function encodes a raw string without any quotation.
- EXTERNJSON int rtJsonEncUnicodeData (OSCTXT *pctxt, const OSUNICHAR *value, OSSIZE nchars)

  This function encodes a variable that contains UCS-2 / UTF-16 characters.
- EXTERNJSON int rtJsonEncUCS4Data (OSCTXT *pctxt, const OS32BITCHAR *value, OSSIZE nchars)

  This function encodes a variable that contains UCS-4 / UTF-32 characters.
- EXTERNJSON int rtJsonEncStartObject (OSCTXT *pctxt, const OSUTF8CHAR *name, OSBOOL noComma)

  This function encodes the beginning of a JSON object.
- EXTERNJSON int rtJsonEncEndObject (OSCTXT *pctxt)

  This function encodes the end of a JSON object.
- EXTERNJSON int rtJsonEncBetweenObject (OSCTXT *pctxt)

  This function encodes the characters separating the JSON name and value.

6.1.1 Detailed Description

6.1.2 Macro Definition Documentation

6.1.2.1 rtJsonEncDecrIndent

```
#define rtJsonEncDecrIndent rtxIndentDecr
```

This decreases the indentation level set in the given context by updating the indent member.

Parameters

pctxt	Pointer to context block structure.

Definition at line 423 of file osrtjson.h.

6.1.2.2 rtJsonEncIncrIndent

```
#define rtJsonEncIncrIndent rtxIndentIncr
```

This increases the indentation level set in the given context by updating the indent member.

Parameters

pctxt	Pointer to context block structure.

Definition at line 431 of file osrtjson.h.

6.1.2.3 rtJsonEncInt64Value

```
#define rtJsonEncInt64Value rtxTxtWriteInt64
```

This function encodes a variable of the XSD integer type.

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

Parameters

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

Returns

Completion status of operation:

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

Definition at line 130 of file osrtjson.h.

6.1.2.4 rtJsonEncIntValue

```
#define rtJsonEncIntValue rtxTxtWriteInt
```

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.

Definition at line 118 of file osrtjson.h.

6.1.2.5 rtJsonEncResetIndent

```
#define rtJsonEncResetIndent rtxIndentReset
```

This resets the indentation level in the given context to zero.

Parameters

pctxt	Pointer to context block structure.

Definition at line 438 of file osrtjson.h.

6.1.2.6 rtJsonEncUInt64Value

```
#define rtJsonEncUInt64Value rtxTxtWriteUInt64
```

This function encodes a variable of the XSD integer type.

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

Parameters

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

Returns

Completion status of operation:

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

Definition at line 627 of file osrtjson.h.

6.1.2.7 rtJsonEncUIntValue

```
#define rtJsonEncUIntValue rtxTxtWriteUInt
```

This function encodes a variable of the XSD unsigned 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.

Definition at line 614 of file osrtjson.h.

6.1.2.8 rtJsonGetIndentLevels

```
#define rtJsonGetIndentLevels rtxGetIndentLevels
```

This returns the number of levels of indentation rtJsonEncIndent is using.

Parameters

pctxt	Pointer to context block structure.

Definition at line 447 of file osrtjson.h.

6.1.3 Function Documentation

6.1.3.1 rtJsonEncAnyAttr()

```
EXTERNJSON int rtJsonEncAnyAttr (
             OSCTXT * pctxt,
             const OSRTDList * pvalue )
```

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

Parameters

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

Returns

Completion status of operation:

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

6.1.3.2 rtJsonEncBase64StrValue()

```
EXTERNJSON int rtJsonEncBase64StrValue (
             OSCTXT * pctxt,
             OSSIZE nocts,
             const OSOCTET * value )
```

This function encodes a variable of the XSD base64Binary type.

Parameters

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

Returns

Completion status of operation:

- 0 = success,

- negative return value is error.

6.1.3.3 rtJsonEncBetweenObject()

```
EXTERNJSON int rtJsonEncBetweenObject (
             OSCTXT * pctxt )
```

This function encodes the characters separating the JSON name and value.

Parameters

pctxt	Pointer to context block structure.

Returns

Completion status of operation:

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

6.1.3.4 rtJsonEncBitStrValue()

```
EXTERNJSON int rtJsonEncBitStrValue (
             OSCTXT * pctxt,
             OSSIZE nbits,
             const OSOCTET * data )
```

This function encodes a variable of the ASN.1 Bit string type as a JSON object, as required by X.697 for a BIT STRING without a fixed length.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.5 rtJsonEncBitStrValueExt()

```
EXTERNJSON int rtJsonEncBitStrValueExt (
             OSCTXT * pctxt,
             OSSIZE nbits,
             const OSOCTET * data,
             OSSIZE dataSize,
             const OSOCTET * extData )
```

This function encodes a variable of the ASN.1 Bit string type as a JSON object, as required by X.697 for a BIT STRING without a fixed length.

It handles bit strings with extdata member present.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.6 rtJsonEncBitStrValueExtV72()

```
EXTERNJSON int rtJsonEncBitStrValueExtV72 (
             OSCTXT * pctxt,
             OSSIZE nbits,
             const OSOCTET * data,
             OSSIZE dataSize,
             const OSOCTET * extData )
```

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

It handles bit strings with extdata member present. This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.7 rtJsonEncBitStrValueV72()

```
EXTERNJSON int rtJsonEncBitStrValueV72 (
             OSCTXT * pctxt,
             OSSIZE nbits,
             const OSOCTET * data )
```

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

This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.8 rtJsonEncBoolValue()

```
EXTERNJSON int rtJsonEncBoolValue (
             OSCTXT * pctxt,
             OSBOOL value )
```

This function encodes a variable of the XSD boolean type.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.9 rtJsonEncChars()

```
EXTERNJSON int rtJsonEncChars (
             OSCTXT * pctxt,
             const char * value,
             OSSIZE valueLen )
```

This function encodes the given number of characters from a character string value as a JSON string.

Parameters

pctxt	Pointer to context block structure.
value	Character string value to be encoded.
valueLen	Length of the XML string to be encoded.

Returns

Completion status of operation:

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

6.1.3.10 rtJsonEncCharStr()

```
EXTERNJSON int rtJsonEncCharStr (
             OSCTXT * pctxt,
             const char * value )
```

This function encodes a character string value as a JSON string.

Parameters

pctxt	Pointer to context block structure.
value	Character string value to be encoded.

Returns

Completion status of operation:

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

6.1.3.11 rtJsonEncDate()

```
EXTERNJSON int rtJsonEncDate (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to be encoded.

Returns

Completion status of operation:

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

6.1.3.12 rtJsonEncDateTime()

```
EXTERNJSON int rtJsonEncDateTime (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

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

Parameters

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

Returns

Completion status of operation:

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

6.1.3.13 rtJsonEncDecimalValue()

```
EXTERNJSON int rtJsonEncDecimalValue (
             OSCTXT * pctxt,
             OSREAL value,
             const OSDecimalFmt * pFmtSpec )
```

This function encodes a value of the XSD decimal type.

Parameters

pctxt	Pointer to context block structure.
value	Value to be encoded.
pFmtSpec	Pointer to format specification structure.

Returns

Completion status of operation:

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

6.1.3.14 rtJsonEncDoubleValue()

```
EXTERNJSON int rtJsonEncDoubleValue (
             OSCTXT * pctxt,
             OSREAL value,
             const OSDoubleFmt * pFmtSpec )
```

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

Parameters

pctxt	Pointer to context block structure.
value	Value to be encoded.
pFmtSpec	Pointer to format specification structure.

Returns

Completion status of operation:

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

6.1.3.15 rtJsonEncEndObject()

```
EXTERNJSON int rtJsonEncEndObject (
             OSCTXT * pctxt )
```

This function encodes the end of a JSON object.

Parameters

pctxt	Pointer to context block structure.

Returns

Completion status of operation:

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

6.1.3.16 rtJsonEncFixedBitStrValue()

```
EXTERNJSON int rtJsonEncFixedBitStrValue (
             OSCTXT * pctxt,
             OSSIZE nbits,
             const OSOCTET * data )
```

This function encodes a variable of the ASN.1 Bit string type as a JSON string, as required by X.697 for a BIT STRING with a fixed length.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.17 rtJsonEncFloatValue()

```
EXTERNJSON int rtJsonEncFloatValue (
             OSCTXT * pctxt,
             OSREAL value,
             const OSDoubleFmt * pFmtSpec )
```

This function encodes a variable of the XSD float type.

Parameters

pctxt	Pointer to context block structure.
value	Value to be encoded.
pFmtSpec	Pointer to format specification structure.

Returns

Completion status of operation:

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

6.1.3.18 rtJsonEncGDay()

```
EXTERNJSON int rtJsonEncGDay (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

This function encodes a numeric gDay value into a JSON string representation.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.19 rtJsonEncGMonth()

```
EXTERNJSON int rtJsonEncGMonth (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

This function encodes a numeric gMonth value into a JSON string representation.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.20 rtJsonEncGMonthDay()

```
EXTERNJSON int rtJsonEncGMonthDay (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

This function encodes a numeric gMonthDay value into a JSON string representation.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.21 rtJsonEncGYear()

```
EXTERNJSON int rtJsonEncGYear (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

This function encodes a numeric gYear value into a JSON string representation.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.22 rtJsonEncGYearMonth()

```
EXTERNJSON int rtJsonEncGYearMonth (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

This function encodes a numeric gYearMonth value into a JSON string representation.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.23 rtJsonEncHexStr()

```
EXTERNJSON int rtJsonEncHexStr (
             OSCTXT * pctxt,
             OSSIZE nocts,
             const OSOCTET * data )
```

This function encodes the given data as a JSON string, using a hexadecimal representation of the data.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.24 rtJsonEncHexValue()

```
EXTERNJSON int rtJsonEncHexValue (
             OSCTXT * pctxt,
             OSSIZE nocts,
             const OSOCTET * data )
```

This function encodes the given data as a sequence of hexadecimal characters.

Unlike rtJsonHexStr, it does not encode quotation marks.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.25 rtJsonEncIndent()

```
EXTERNJSON int rtJsonEncIndent (
             OSCTXT * pctxt )
```

This function:

- Writes a comma unless OSJSONNOCOMMA is set or the previous character is a comma, null terminator, {, or [character.
- Writes a newline and indentation unless OSNOWHITEPSACE is set.

The amount of indentation to add is determined by the indent member variable in the context structure.

Parameters

pctxt	Pointer to context block structure.

Returns

Completion status of operation:

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

6.1.3.26 rtJsonEncStartObject()

```
EXTERNJSON int rtJsonEncStartObject (
             OSCTXT * pctxt,
             const OSUTF8CHAR * name,
             OSBOOL noComma )
```

This function encodes the beginning of a JSON object.

Parameters

pctxt	Pointer to context block structure.
name	Object name token to be encoded.
noComma	If TRUE do not print comma at end of line in output.

Returns

Completion status of operation:

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

6.1.3.27 rtJsonEncStringNull()

```
EXTERNJSON int rtJsonEncStringNull (
             OSCTXT * pctxt )
```

This function encodes an asn.1 NULL type as string.

Parameters

pctxt	Pointer to context block structure.

Returns

Completion status of operation:

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

6.1.3.28 rtJsonEncStringObject()

```
EXTERNJSON int rtJsonEncStringObject (
             OSCTXT * pctxt,
             const OSUTF8CHAR * name,
             const OSUTF8CHAR * value )
```

This function encodes a JSON object containing a string value.

Parameters

pctxt	Pointer to context block structure.
name	Name token to be encoded.
value	Value as a character string to be encoded.

Returns

Completion status of operation:

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

6.1.3.29 rtJsonEncStringObject2()

```
EXTERNJSON int rtJsonEncStringObject2 (
             OSCTXT * pctxt,
             const OSUTF8CHAR * name,
             size_t nameLen,
             const OSUTF8CHAR * value,
             size_t valueLen )
```

This function encodes a JSON object containing a string value.

Parameters

pctxt	Pointer to context block structure.
name	Name token to be encoded.
nameLen	Length of the name token to be encoded.
value	Value as a character string to be encoded.
valueLen	Length of the value to be encoded.

Returns

Completion status of operation:

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

6.1.3.30 rtJsonEncStringPair()

```
EXTERNJSON int rtJsonEncStringPair (
             OSCTXT * pctxt,
             const OSUTF8CHAR * name,
             const OSUTF8CHAR * value )
```

This function encodes a name/value pair.

The value is a character string.

Parameters

pctxt	Pointer to context block structure.
name	Name token to be encoded.
value	Value as a character string to be encoded.

Returns

Completion status of operation:

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

6.1.3.31 rtJsonEncStringPair2()

```
EXTERNJSON int rtJsonEncStringPair2 (
             OSCTXT * pctxt,
             const OSUTF8CHAR * name,
             size_t nameLen,
             const OSUTF8CHAR * value,
             size_t valueLen )
```

This function encodes a name/value pair.

The value is a character string.

Parameters

pctxt	Pointer to context block structure.
name	Name token to be encoded.
nameLen	Length of the name token to be encoded.
value	Value as a character string to be encoded.
valueLen	Length of the value to be encoded.

Returns

Completion status of operation:

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

6.1.3.32 rtJsonEncStringRaw()

```
EXTERNJSON int rtJsonEncStringRaw (
             OSCTXT * pctxt,
             const OSUTF8CHAR * value )
```

This function encodes a raw string without any quotation.

Parameters

pctxt	Pointer to context block structure.
value	String value to be written.

Returns

Completion status of operation:

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

6.1.3.33 rtJsonEncStringValue()

```
EXTERNJSON int rtJsonEncStringValue (
             OSCTXT * pctxt,
             const OSUTF8CHAR * value )
```

This function encodes a UTF8 string value.

Parameters

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

Returns

Completion status of operation:

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

6.1.3.34 rtJsonEncStringValue2()

```
EXTERNJSON int rtJsonEncStringValue2 (
             OSCTXT * pctxt,
             const OSUTF8CHAR * value,
             size_t valueLen )
```

This function encodes a UTF8 string value.

Parameters

pctxt	Pointer to context block structure.
value	XML string value to be encoded.
valueLen	Length of the XML string to be encoded.

Returns

Completion status of operation:

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

6.1.3.35 rtJsonEncTime()

```
EXTERNJSON int rtJsonEncTime (
             OSCTXT * pctxt,
             const OSXSDDateTime * pvalue )
```

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to be encoded.

Returns

Completion status of operation:

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

6.1.3.36 rtJsonEncUCS4Data()

```
EXTERNJSON int rtJsonEncUCS4Data (
             OSCTXT * pctxt,
             const OS32BITCHAR * value,
             OSSIZE nchars )
```

This function encodes a variable that contains UCS-4 / UTF-32 characters.

Parameters

pctxt	Pointer to context block structure.
value	UCS-4 characters to be encoded.
nchars	Number of characters to be encoded.

Returns

Completion status of operation:

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

6.1.3.37 rtJsonEncUnicodeData()

```
EXTERNJSON int rtJsonEncUnicodeData (
             OSCTXT * pctxt,
             const OSUNICHAR * value,
             OSSIZE nchars )
```

This function encodes a variable that contains UCS-2 / UTF-16 characters.

Parameters

pctxt	Pointer to context block structure.
value	UCS-2 characters to be encoded.
nchars	Number of Unicode characters to be encoded.

Returns

Completion status of operation:

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

6.2 JSON decode functions.

Macros

- #define rtJsonDecPeekChar(pctxt, pch) rtxTxtPeekChar(pctxt, pch, TRUE)

  This function determines the next non-whitespace character in the input.
- #define rtJsonDecPeekChar2(pctxt) rtxTxtPeekChar2(pctxt, TRUE)

  This function determines the next non-whitespace character in the input.

Functions

- EXTERNJSON int rtJsonDecAnyElem (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes an arbitrary block of JSON-encoded data into a string variable.
- EXTERNJSON int rtJsonDecAnyElem2 (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This version of rtJsonDecAnyElem assumes the element name has been pushed on the element name stack in the context.
- EXTERNJSON int rtJsonDecAnyType (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes an arbitrary block of JSON-encoded data into a string variable.
- EXTERNJSON int rtJsonDecBase64Str (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, size_t bufsize)

  This function decodes a contents of a Base64-encode binary string into a static memory structure.
- EXTERNJSON int rtJsonDecBase64Str64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, size_t bufsize)

  This function is identical to rtJsonDecBase64Str except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecDynBase64Str (OSCTXT *pctxt, OSDynOctStr *pvalue)

  This function decodes a contents of a Base64-encode binary string.
- EXTERNJSON int rtJsonDecDynBase64Str64 (OSCTXT *pctxt, OSDynOctStr64 *pvalue)

  This function is identical to rtJsonDecDynBase64Str64 except that it supports 64-bit integer lengths on 64-bit systems.
- EXTERNJSON int rtJsonDecBool (OSCTXT *pctxt, OSBOOL *pvalue)

  This function decodes a variable of the boolean type.
- EXTERNJSON int rtJsonTryDecBool (OSCTXT *pctxt, OSBOOL *pvalue)

  This function will attempt to decode a boolean variable at the current buffer or stream position.
- EXTERNJSON int rtJsonDecDate (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'date' type.
- EXTERNJSON int rtJsonDecTime (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'time' type.
- EXTERNJSON int rtJsonDecDateTime (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'dateTime' type.
- EXTERNJSON int rtJsonDecGYear (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gYear' type.
- EXTERNJSON int rtJsonDecGYearMonth (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gYearMonth' type.
- EXTERNJSON int rtJsonDecGMonth (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gMonth' type.
- EXTERNJSON int rtJsonDecGMonthDay (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gMonthDay' type.
- EXTERNJSON int rtJsonDecGDay (OSCTXT *pctxt, OSXSDDateTime *pvalue)

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

- EXTERNJSON int rtJsonDecDecimal (OSCTXT *pctxt, OSREAL *pvalue, int totalDigits, int fractionDigits)

  This function decodes the contents of a decimal data type.
- EXTERNJSON int rtJsonDecDouble (OSCTXT *pctxt, OSREAL *pvalue)

  This function decodes the contents of a float or double data type.
- EXTERNJSON int rtJsonDecHexToCharStr (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes a JSON string, interpreting the content as the hexadecimal represention of the bytes for a character string, which it returns.
- EXTERNJSON int rtJsonDecHexStr (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, size_t bufsize)

  This function decodes a JSON string consisting of hexadecimal characters into a static memory structure.
- EXTERNJSON int rtJsonDecHexStr64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, size_t bufsize)

  This function is identical to rtJsonDecHexStr except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecHexData64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, size_t bufsize)

  This function deocdes a sequence of heaxadecimal characters until a non-hexadecimal character (which is not consumed) is found OR the preallocated array is filled.
- EXTERNJSON int rtJsonDecDynHexStr (OSCTXT *pctxt, OSDynOctStr *pvalue)

  This function decodes a JSON string consisting of hexadecimal characters.
- EXTERNJSON int rtJsonDecDynHexStr64 (OSCTXT *pctxt, OSDynOctStr64 *pvalue)

  This function is identical to rtJsonDecDynHexStr except that it supports 64-bit integer lengths on 64-bit systems.
- EXTERNJSON int rtJsonDecDynHexData64 (OSCTXT *pctxt, OSDynOctStr64 *pvalue)

  This function decodes a sequence of hexadecimal characters until a non-hexadecimal character (which is not consumed) is found.
- EXTERNJSON int rtJsonDecDynBitStrV72 (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET **data)

  This function decodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonDecDynBitStr64V72 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET **data)

  This function is identical to rtJsonDecDynBitStrV72 except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecBitStrValueV72 (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize)

  This function decodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonDecBitStrValue64V72 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize)

  This function is identical to rtJsonDecBitStrValueV72 except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecBitStrValueExtV72 (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function decodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonDecBitStrValueExt64V72 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function is identical to rtJsonDecBitStrValueExtV72 except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecDynBitStr (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET **data)

  This function decodes a value of an ASN.1 BIT STRING type, not constrained to a fixed length, encoded according to X.697 as a JSON object.
- EXTERNJSON int rtJsonDecDynBitStr64 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET **data)

  This function is identical to rtJsonDecDynBitStr except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecFixedDynBitStr (OSCTXT *pctxt, OSSIZE nbits, OSOCTET **data)

  This function decodes a value of an ASN.1 BIT STRING type constrained to a fixed length, encoded according to X.697 as a JSON string.
- EXTERNJSON int rtJsonDecBitStrValue (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize)

This function decodes a value of an ASN.1 BIT STRING type that is not constrained to a fixed length, encoded according to X.697 as a JSON object.

- EXTERNJSON int rtJsonDecBitStrValue64 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize)

  This function is identical to rtJsonDecBitStrValue except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecFixedBitStrValue (OSCTXT *pctxt, OSSIZE nbits, OSOCTET *data, OSSIZE bufsize)

  This function decodes a value of an ASN.1 BIT STRING type constrained to have the given fixed length, encoded according to X.697 as a JSON string.
- EXTERNJSON int rtJsonDecBitStrValueExt (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function decodes a value of a ASN.1 BIT STRING type without a fixed-length constraint, encoded according to X.697 as JSON object.
- EXTERNJSON int rtJsonDecBitStrValueExt64 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function is identical to rtJsonDecBitStrValueExt except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecMatchChar (OSCTXT *pctxt, OSUTF8CHAR ch)

  This function attempts to match the given character, skipping over any whitesapce, if necessary.
- EXTERNJSON int rtJsonDecMatchCharStr (OSCTXT *pctxt, const char *token)

  This function decodes a JSON string and matches with a given token.
- EXTERNJSON int rtJsonDecMatchObjectStart (OSCTXT *pctxt, const OSUTF8NameAndLen *nameArray, size_t numNames)

  This function matches the start of a JSON object.
- EXTERNJSON int rtJsonDecMatchToken (OSCTXT *pctxt, const OSUTF8CHAR *token)

  This function decodes a JSON string and matches with a given token.
- EXTERNJSON int rtJsonDecMatchToken2 (OSCTXT *pctxt, const OSUTF8CHAR *token, size_t tokenLen)

  This function decodes a JSON string and matches with a given token.
- EXTERNJSON int rtJsonDecNameValuePair (OSCTXT *pctxt, OSUTF8NVP *pvalue)

  This function decodes a name/value pair.
- EXTERNJSON int rtJsonDecNull (OSCTXT *pctxt)

  This function decodes a JSON null.
- EXTERNJSON int rtJsonTryDecNull (OSCTXT *pctxt)

  This function will attempt to decode a null value at the current buffer or stream position.
- EXTERNJSON int rtJsonDecNumberString (OSCTXT *pctxt, char **ppCharStr)

  This function decodes a JSON number into a character string variable.
- EXTERNJSON int rtJsonDecStringObject (OSCTXT *pctxt, const OSUTF8CHAR *name, OSUTF8CHAR **ppvalue)

  This function decodes a JSON object containing a single entry with the given key (name), and returns the key's associated value, which must be a JSON string, via ppvalue.
- EXTERNJSON int rtJsonDecStringValue (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes the contents of a string data type.
- EXTERNJSON int rtJsonDecStringValueArray (OSCTXT *pctxt, OSUTF8CHAR *pvalue, OSSIZE bufsize)

  This function is the same as rtJsonDecStringValue except that it decodes into a character array.
- EXTERNJSON int rtJsonDecXmlStringValue (OSCTXT *pctxt, OSXMLSTRING *pvalue)

  This function decodes the contents of an XML string data type.
- EXTERNJSON int rtJsonDecUCS2String (OSCTXT *pctxt, OSUNICHAR **ppstr, OSSIZE *pnchars)

  This function is used to decode input UTF-8 data into a UCS-2 / UTF-16 character string.
- EXTERNJSON int rtJsonDecUCS4String (OSCTXT *pctxt, OS32BITCHAR **ppstr, OSSIZE *pnchars)

  This function is used to decode input UTF-8 data into a UCS-4 / UTF-32 character string.
- EXTERNJSON size_t rtJsonGetElemIdx (OSCTXT *pctxt, const OSUTF8NameAndLen nameArray[ ], size_t nrows)

  This function determines which of several possible JSON strings appears next in the input.

6.2.1 Detailed Description

6.2.2 Macro Definition Documentation

6.2.2.1 rtJsonDecPeekChar

```
#define rtJsonDecPeekChar(
              pctxt,
              pch ) rtxTxtPeekChar(pctxt, pch, TRUE)
```

This function determines the next non-whitespace character in the input.

The non-whitespace character is not consumed.

Parameters

pctxt	Pointer to OSCTXT structure
pch	A pointer to a variable to receive the next character.

Returns

Completion status of operation:

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

Definition at line 1589 of file osrtjson.h.

6.2.2.2 rtJsonDecPeekChar2

```
#define rtJsonDecPeekChar2(
              pctxt ) rtxTxtPeekChar2(pctxt, TRUE)
```

This function determines the next non-whitespace character in the input.

The non-whitespace character is not consumed.

Parameters

pctxt	Pointer to OSCTXT structure

Returns

The peeked character, or null if there is a failure. The error will be logged in the context.

Definition at line 1599 of file osrtjson.h.

6.2.3 Function Documentation

6.2.3.1 rtJsonDecAnyElem()

```
EXTERNJSON int rtJsonDecAnyElem (
             OSCTXT * pctxt,
             OSUTF8CHAR ** ppvalue )
```

This function decodes an arbitrary block of JSON-encoded data into a string variable.

In this case, the expected format is element name : JSON encoded data.

Parameters

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

Returns

Completion status of operation:

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

6.2.3.2 rtJsonDecAnyElem2()

```
EXTERNJSON int rtJsonDecAnyElem2 (
             OSCTXT * pctxt,
             OSUTF8CHAR ** ppvalue )
```

This version of rtJsonDecAnyElem assumes the element name has been pushed on the element name stack in the context.

This will be the case if rtJsonGetElemIdx is called prior to calling this function.

Parameters

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

Returns

Completion status of operation:

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

6.2.3.3 rtJsonDecAnyType()

```
EXTERNJSON int rtJsonDecAnyType (
             OSCTXT * pctxt,
             OSUTF8CHAR ** ppvalue )
```

This function decodes an arbitrary block of JSON-encoded data into a string variable.

In this case, the expected format is a complete JSON encoded data fragment.

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
ppvalue	A pointer to a variable to receive the decoded JSON text. You may pass null if not interested in receiving the decoded text.

Returns

Completion status of operation:

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

6.2.3.4 rtJsonDecBase64Str()

```
EXTERNJSON int rtJsonDecBase64Str (
             OSCTXT * pctxt,
             OSOCTET * pvalue,
```

```
OSUINT32 * pnocts,
size_t bufsize )
```

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

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

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the bufsize input parameter.
pnocts	A pointer to an integer value to receive the decoded number of octets.
bufsize	The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns

Completion status of operation:

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

6.2.3.5 rtJsonDecBase64Str64()

```
EXTERNJSON int rtJsonDecBase64Str64 (
             OSCTXT * pctxt,
             OSOCTET * pvalue,
             OSSIZE * pnocts,
             size_t bufsize )
```

This function is identical to rtJsonDecBase64Str except that it supports lengths up to 64-bits in size on 64-bit machines.

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the bufsize input parameter.
pnocts	A pointer to an integer value to receive the decoded number of octets.
bufsize	The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns

Completion status of operation:

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

See also

rtJsonDecBase64Str

6.2.3.6 rtJsonDecBitStrValue()

```
EXTERNJSON int rtJsonDecBitStrValue (
             OSCTXT * pctxt,
             OSUINT32 * nbits,
             OSOCTET * data,
             OSSIZE bufsize )
```

This function decodes a value of an ASN.1 BIT STRING type that is not constrained to a fixed length, encoded according to X.697 as a JSON object.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.

Returns

Completion status of operation:

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

6.2.3.7 rtJsonDecBitStrValue64()

```
EXTERNJSON int rtJsonDecBitStrValue64 (
             OSCTXT * pctxt,
             OSSIZE * nbits,
             OSOCTET * data,
             OSSIZE bufsize )
```

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

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.

Returns

Completion status of operation:

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

See also

rtJsonDecBitStrValue

6.2.3.8 rtJsonDecBitStrValue64V72()

```
EXTERNJSON int rtJsonDecBitStrValue64V72 (
             OSCTXT * pctxt,
             OSSIZE * nbits,
             OSOCTET * data,
             OSSIZE bufsize )
```

This function is identical to rtJsonDecBitStrValueV72 except that it supports lengths up to 64-bits in size on 64-bit machines.

This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.

Returns

Completion status of operation:

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

See also

rtJsonDecBitStrValueV72

6.2.3.9 rtJsonDecBitStrValueExt()

```
EXTERNJSON int rtJsonDecBitStrValueExt (
             OSCTXT * pctxt,
             OSUINT32 * nbits,
             OSOCTET * data,
             OSSIZE bufsize,
             OSOCTET ** extdata )
```

This function decodes a value of a ASN.1 BIT STRING type without a fixed-length constraint, encoded according to X.697 as JSON object.

It handles bit strings with extdata member present.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.
extdata	A pointer to an OSOCTET array that will receive the decoded extdata value.

Returns

Completion status of operation:

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

6.2.3.10 rtJsonDecBitStrValueExt64()

```
EXTERNJSON int rtJsonDecBitStrValueExt64 (
             OSCTXT * pctxt,
             OSSIZE * nbits,
             OSOCTET * data,
             OSSIZE bufsize,
             OSOCTET ** extdata )
```

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

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.
extdata	A pointer to an OSOCTET array that will receive the decoded extdata value.

Returns

Completion status of operation:

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

See also

rtJsonDecBitStrValueExt

6.2.3.11 rtJsonDecBitStrValueExt64V72()

```
EXTERNJSON int rtJsonDecBitStrValueExt64V72 (
             OSCTXT * pctxt,
             OSSIZE * nbits,
             OSOCTET * data,
             OSSIZE bufsize,
             OSOCTET ** extdata )
```

This function is identical to rtJsonDecBitStrValueExtV72 except that it supports lengths up to 64-bits in size on 64-bit machines.

This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.
extdata	A pointer to an OSOCTET array that will receive the decoded extdata value.

Returns

Completion status of operation:

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

See also

rtJsonDecBitStrValueExtV72

6.2.3.12 rtJsonDecBitStrValueExtV72()

```
EXTERNJSON int rtJsonDecBitStrValueExtV72 (
             OSCTXT * pctxt,
             OSUINT32 * nbits,
             OSOCTET * data,
             OSSIZE bufsize,
             OSOCTET ** extdata )
```

This function decodes a variable of the ASN.1 Bit string type.

It handles bit strings with extdata member present. This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.
extdata	A pointer to an OSOCTET array that will receive the decoded extdata value.

Returns

Completion status of operation:

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

6.2.3.13 rtJsonDecBitStrValueV72()

```
EXTERNJSON int rtJsonDecBitStrValueV72 (
             OSCTXT * pctxt,
             OSUINT32 * nbits,
             OSOCTET * data,
             OSSIZE bufsize )
```

This function decodes a variable of the ASN.1 Bit string type.

This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.

Returns

Completion status of operation:

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

6.2.3.14 rtJsonDecBool()

```
EXTERNJSON int rtJsonDecBool (
             OSCTXT * pctxt,
             OSBOOL * pvalue )
```

This function decodes a variable of the boolean type.

Parameters

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

Returns

Completion status of operation:

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

6.2.3.15 rtJsonDecDate()

```
EXTERNJSON int rtJsonDecDate (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.16 rtJsonDecDateTime()

```
EXTERNJSON int rtJsonDecDateTime (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.17 rtJsonDecDecimal()

```
EXTERNJSON int rtJsonDecDecimal (
             OSCTXT * pctxt,
             OSREAL * pvalue,
             int totalDigits,
             int fractionDigits )
```

This function decodes the contents of a decimal data type.

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

Parameters

pctxt	Pointer to context block structure.
pvalue	Pointer to 64-bit double value to receive decoded result.
totalDigits	The total number of digits in the decimal value.
fractionDigits	The number of fractional digits in the decimal value.

Returns

Completion status of operation:

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

6.2.3.18 rtJsonDecDouble()

```
EXTERNJSON int rtJsonDecDouble (
             OSCTXT * pctxt,
             OSREAL * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	Pointer to 64-bit double value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.19 rtJsonDecDynBase64Str()

```
EXTERNJSON int rtJsonDecDynBase64Str (
             OSCTXT * pctxt,
             OSDynOctStr * pvalue )
```

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

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

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated for the string using the ::rtxMemAlloc function.

Returns

Completion status of operation:

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

6.2.3.20 rtJsonDecDynBase64Str64()

```
EXTERNJSON int rtJsonDecDynBase64Str64 (
             OSCTXT * pctxt,
             OSDynOctStr64 * pvalue )
```

This function is identical to rtJsonDecDynBase64Str64 except that it supports 64-bit integer lengths on 64-bit systems.

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated for the string using the ::rtxMemAlloc function.

Returns

Completion status of operation:

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

6.2.3.21 rtJsonDecDynBitStr()

```
EXTERNJSON int rtJsonDecDynBitStr (
             OSCTXT * pctxt,
             OSUINT32 * nbits,
             OSOCTET ** data )
```

This function decodes a value of an ASN.1 BIT STRING type, not constrained to a fixed length, encoded according to X.697 as a JSON object.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string; the array will be allocated by the decoding function.

Returns

Completion status of operation:

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

6.2.3.22 rtJsonDecDynBitStr64()

```
EXTERNJSON int rtJsonDecDynBitStr64 (
             OSCTXT * pctxt,
             OSSIZE * nbits,
             OSOCTET ** data )
```

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

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string; the array will be allocated by the decoding function.

Returns

Completion status of operation:

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

See also

rtJsonDecDynBitStr

6.2.3.23 rtJsonDecDynBitStr64V72()

```
EXTERNJSON int rtJsonDecDynBitStr64V72 (
             OSCTXT * pctxt,
             OSSIZE * nbits,
             OSOCTET ** data )
```

This function is identical to rtJsonDecDynBitStrV72 except that it supports lengths up to 64-bits in size on 64-bit machines.

This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string; the array will be allocated by the decoding function.

Returns

Completion status of operation:

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

See also

rtJsonDecDynBitStrV72

6.2.3.24 rtJsonDecDynBitStrV72()

```
EXTERNJSON int rtJsonDecDynBitStrV72 (
             OSCTXT * pctxt,
             OSUINT32 * nbits,
             OSOCTET ** data )
```

This function decodes a variable of the ASN.1 Bit string type.

This provides ObjSys-specific behavior that predates ITU-T X.697.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string; the array will be allocated by the decoding function.

Returns

Completion status of operation:

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

6.2.3.25 rtJsonDecDynHexData64()

```
EXTERNJSON int rtJsonDecDynHexData64 (
             OSCTXT * pctxt,
             OSDynOctStr64 * pvalue )
```

This function decodes a sequence of hexadecimal characters until a non-hexadecimal character (which is not consumed) is found.

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated to hold the string using the ::rtxMemAlloc function.

Returns

Completion status of operation:

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

6.2.3.26 rtJsonDecDynHexStr()

```
EXTERNJSON int rtJsonDecDynHexStr (
             OSCTXT * pctxt,
             OSDynOctStr * pvalue )
```

This function decodes a JSON string consisting of hexadecimal characters.

This function will allocate dynamic memory to store the decoded result. Input is expected to be a string of OSUTF8CHAR characters returned by a JSON parser.

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated to hold the string using the ::rtxMemAlloc function.

Returns

Completion status of operation:

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

6.2.3.27 rtJsonDecDynHexStr64()

```
EXTERNJSON int rtJsonDecDynHexStr64 (
             OSCTXT * pctxt,
             OSDynOctStr64 * pvalue )
```

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

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a dynamic octet string structure to receive the decoded octet string. Dynamic memory is allocated to hold the string using the ::rtxMemAlloc function.

Returns

Completion status of operation:

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

6.2.3.28 rtJsonDecFixedBitStrValue()

```
EXTERNJSON int rtJsonDecFixedBitStrValue (
             OSCTXT * pctxt,
             OSSIZE nbits,
             OSOCTET * data,
             OSSIZE bufsize )
```

This function decodes a value of an ASN.1 BIT STRING type constrained to have the given fixed length, encoded according to X.697 as a JSON string.

This ensures the encoded data is the given number of bits and that unused bits are zeros.

Parameters

pctxt	Pointer to context block structure.
nbits	The number of bits the BIT STRING is fixed to.
data	A pointer to an OSOCTET array that will receive the decoded bit string. The array must be preallocated.
bufsize	Size of the static buffer in bytes into which the data is to be decoded.

Returns

Completion status of operation:

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

6.2.3.29 rtJsonDecFixedDynBitStr()

```
EXTERNJSON int rtJsonDecFixedDynBitStr (
             OSCTXT * pctxt,
             OSSIZE nbits,
             OSOCTET ** data )
```

This function decodes a value of an ASN.1 BIT STRING type constrained to a fixed length, encoded according to X.697 as a JSON string.

This ensures the encoded data is the given number of bits and that unused bits are zeros.

Parameters

pctxt	Pointer to context block structure.
nbits	A pointer to an unsigned integer to receive the number of bits in the bit string.
data	A pointer to an OSOCTET array that will receive the decoded bit string; the array will be allocated by the decoding function.

Returns

Completion status of operation:

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

6.2.3.30 rtJsonDecGDay()

```
EXTERNJSON int rtJsonDecGDay (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.31 rtJsonDecGMonth()

```
EXTERNJSON int rtJsonDecGMonth (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.32 rtJsonDecGMonthDay()

```
EXTERNJSON int rtJsonDecGMonthDay (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.33 rtJsonDecGYear()

```
EXTERNJSON int rtJsonDecGYear (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.34 rtJsonDecGYearMonth()

```
EXTERNJSON int rtJsonDecGYearMonth (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.35 rtJsonDecHexData64()

```
EXTERNJSON int rtJsonDecHexData64 (
             OSCTXT * pctxt,
             OSOCTET * pvalue,
             OSSIZE * pnocts,
             size_t bufsize )
```

This function deocdes a sequence of heaxadecimal characters until a non-hexadecimal character (which is not consumed) is found OR the preallocated array is filled.

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a variable to receive the decoded bit string. This is assumed to be a preallocated 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 pvalue.

Returns

Completion status of operation:

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

6.2.3.36 rtJsonDecHexStr()

```
EXTERNJSON int rtJsonDecHexStr (
             OSCTXT * pctxt,
```

```
            OSOCTET * pvalue,
            OSUINT32 * pnocts,
            size_t bufsize )
```

This function decodes a JSON string consisting of hexadecimal characters into a static memory structure.

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

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the bufsize input parameter.
pnocts	A pointer to an integer value to receive the decoded number of octets.
bufsize	The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns

Completion status of operation:

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

6.2.3.37 rtJsonDecHexStr64()

```
EXTERNJSON int rtJsonDecHexStr64 (
            OSCTXT * pctxt,
            OSOCTET * pvalue,
            OSSIZE * pnocts,
            size_t bufsize )
```

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

Parameters

pctxt	A pointer to a context structure. This provides a storage area for the function to store all working variables that must be maintained between function calls.
pvalue	A pointer to a variable to receive the decoded bit string. This is assumed to be a static array large enough to hold the number of octets specified in the bufsize input parameter.
pnocts	A pointer to an integer value to receive the decoded number of octets.
bufsize	The size (in octets) of the sized octet string. An error will occur if the number of octets in the decoded string is larger than this value.

Returns

Completion status of operation:

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

See also

rtJsonDecHexStr

6.2.3.38 rtJsonDecHexToCharStr()

```
EXTERNJSON int rtJsonDecHexToCharStr (
             OSCTXT * pctxt,
             OSUTF8CHAR ** ppvalue )
```

This function decodes a JSON string, interpreting the content as the hexadecimal represention of the bytes for a character string, which it returns.

This function is used to support the JSON encoding specified in X.697 for the following ASN.1 types: TeletexString, T61String, VideotexString, GraphicString, and GeneralString.

Parameters

pctxt	Pointer to context block structure.
ppvalue	Pointer to an string structure to receive the decoded string. Memory is allocated for the string using the run-time memory manager. If null, the JSON string is decoded but not retained.

Returns

Completion status of operation:

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

6.2.3.39 rtJsonDecMatchChar()

```
EXTERNJSON int rtJsonDecMatchChar (
             OSCTXT * pctxt,
             OSUTF8CHAR ch )
```

This function attempts to match the given character, skipping over any whitesapce, if necessary.

If a different character is found, this function returns RTERR_INVCHAR and does not consume the non-matching character.

Parameters

pctxt	Pointer to context block structure.
ch	The character to be matched.

Returns

Completion status of operation:

- 0 = success,
- RTERR_INVCHAR = different character found
- negative return value is error.

6.2.3.40 rtJsonDecMatchCharStr()

```
EXTERNJSON int rtJsonDecMatchCharStr (
             OSCTXT * pctxt,
             const char * token )
```

This function decodes a JSON string and matches with a given token.

This is equivalent to rtJsonDecMatchToken.

Parameters

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

Returns

Completion status of operation:

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

6.2.3.41 rtJsonDecMatchObjectStart()

```
EXTERNJSON int rtJsonDecMatchObjectStart (
             OSCTXT * pctxt,
             const OSUTF8NameAndLen * nameArray,
             size_t numNames )
```

This function matches the start of a JSON object.

This will skip leading whitespace, then match the opening '{'. It will then match a key that matches any of the values in nameArray, and, if successful, it then matches the subsequent ':' character after the key.

There is no indication of which name was matched, making this function not very useful. See also rtJsonDecString↩Object.

It is an error if there is not an opening '{', if the key does not match any of the given names, or if the ':' character is not found.

Parameters

pctxt	Pointer to context block structure.
nameArray	Array of names to be matched.
numNames	Number of names in the name array

Returns

Completion status of operation:

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

6.2.3.42 rtJsonDecMatchToken()

```
EXTERNJSON int rtJsonDecMatchToken (
             OSCTXT * pctxt,
             const OSUTF8CHAR * token )
```

This function decodes a JSON string and matches with a given token.

Parameters

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

Returns

Completion status of operation:

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

6.2.3.43 rtJsonDecMatchToken2()

```
EXTERNJSON int rtJsonDecMatchToken2 (
             OSCTXT * pctxt,
             const OSUTF8CHAR * token,
             size_t tokenLen )
```

This function decodes a JSON string and matches with a given token.

Parameters

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

Returns

Completion status of operation:

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

6.2.3.44 rtJsonDecNameValuePair()

```
EXTERNJSON int rtJsonDecNameValuePair (
             OSCTXT * pctxt,
             OSUTF8NVP * pvalue )
```

This function decodes a name/value pair.

Parameters

pctxt	Pointer to context block structure.
pvalue	Pointer to an structure to receive the decoded name and value.

Returns

Completion status of operation:

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

6.2.3.45 rtJsonDecNull()

```
EXTERNJSON int rtJsonDecNull (
             OSCTXT * pctxt )
```

This function decodes a JSON null.

Parameters

pctxt	A pointer to a context data structure.

Returns

0 on success, less than zero otherwise.

6.2.3.46 rtJsonDecNumberString()

```
EXTERNJSON int rtJsonDecNumberString (
             OSCTXT * pctxt,
             char ** ppCharStr )
```

This function decodes a JSON number into a character string variable.

Parameters

pctxt	Pointer to context block structure.
ppCharStr	Pointer to character string pointer to receive decoded value. Dynamic memory is allocated for the string using the rtxMemAlloc function.

Returns

Completion status of operation:

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

6.2.3.47 rtJsonDecStringObject()

```
EXTERNJSON int rtJsonDecStringObject (
             OSCTXT * pctxt,
             const OSUTF8CHAR * name,
             OSUTF8CHAR ** ppvalue )
```

This function decodes a JSON object containing a single entry with the given key (name), and returns the key's associated value, which must be a JSON string, via ppvalue.

Parameters

pctxt	Pointer to context block structure.
name	The name token.
ppvalue	Pointer to an string structure to receive the decoded string. Memory is allocated for the string using the run-time memory manager.

Returns

Completion status of operation:

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

6.2.3.48 rtJsonDecStringValue()

```
EXTERNJSON int rtJsonDecStringValue (
             OSCTXT * pctxt,
             OSUTF8CHAR ** ppvalue )
```

This function decodes the contents of a string data type.

This type contains a pointer to a UTF-8 characer string. Input is expected to be a string of UTF-8 characters returned by a JSON parser.

Parameters

pctxt	Pointer to context block structure.
ppvalue	Pointer to an string structure to receive the decoded string. Memory is allocated for the string using the run-time memory manager.

Returns

Completion status of operation:

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

6.2.3.49 rtJsonDecStringValueArray()

```
EXTERNJSON int rtJsonDecStringValueArray (
             OSCTXT * pctxt,
             OSUTF8CHAR * pvalue,
             OSSIZE bufsize )
```

This function is the same as rtJsonDecStringValue except that it decodes into a character array.

Parameters

pctxt	Pointer to context block structure.
pvalue	Pointer to character array to decode into.
bufize	Size of the given array.

Returns

Completion status of operation:

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

6.2.3.50 rtJsonDecTime()

```
EXTERNJSON int rtJsonDecTime (
             OSCTXT * pctxt,
             OSXSDDateTime * pvalue )
```

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

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

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

Parameters

pctxt	Pointer to context block structure.
pvalue	OSXSDDateTime type pointer points to a OSXSDDateTime value to receive decoded result.

Returns

Completion status of operation:

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

6.2.3.51 rtJsonDecUCS2String()

```
EXTERNJSON int rtJsonDecUCS2String (
             OSCTXT * pctxt,
             OSUNICHAR ** ppstr,
             OSSIZE * pnchars )
```

This function is used to decode input UTF-8 data into a UCS-2 / UTF-16 character string.

Parameters

pctxt	A pointer to the context block structure.
ppstr	A pointer to a UTF-16 string; memory will be allocated to hold the string using the run-time memory manager.
pnchars	A pointer to an integer to hold the number of characters in the string. (The number of octets may be found by multiplying by two.)

Returns

0 on success; less than zero on error.

6.2.3.52 rtJsonDecUCS4String()

```
EXTERNJSON int rtJsonDecUCS4String (
             OSCTXT * pctxt,
             OS32BITCHAR ** ppstr,
             OSSIZE * pnchars )
```

This function is used to decode input UTF-8 data into a UCS-4 / UTF-32 character string.

Parameters

pctxt	A pointer to the context block structure.
ppstr	A pointer to a UTF-32 string; memory will be allocated to hold the string using the run-time memory manager.
pnchars	A pointer to an integer to hold the number of characters in the string. (The number of octets may be found by multiplying by two.)

Returns

0 on success; less than zero on error.

6.2.3.53 rtJsonDecXmlStringValue()

```
EXTERNJSON int rtJsonDecXmlStringValue (
             OSCTXT * pctxt,
             OSXMLSTRING * pvalue )
```

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

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

Parameters

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

Returns

Completion status of operation:

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

6.2.3.54 rtJsonGetElemIdx()

```
EXTERNJSON size_t rtJsonGetElemIdx (
             OSCTXT * pctxt,
             const OSUTF8NameAndLen nameArray[],
             size_t nrows )
```

This function determines which of several possible JSON strings appears next in the input.

This will skip any leading whitespace and then parses a JSON string. It is an error if the input does not have a JSON string. The value of the JSON string is then matched against one of the values in nameArray and the corresponding index is returned.

Parameters

pctxt	Pointer to context block structure.
nameArray	Elements descriptor table.
nrows	Number of descriptors in table.

Returns

Completion status of operation:

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

6.2.3.55 rtJsonTryDecBool()

```
EXTERNJSON int rtJsonTryDecBool (
             OSCTXT * pctxt,
             OSBOOL * pvalue )
```

This function will attempt to decode a boolean variable at the current buffer or stream position.

If the attempt fails, the decode cursor is reset to the position it was at when the function was called and an RTERR_I↩ DNOTFOU status is returned. The error is not logged in the context.

Parameters

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

Returns

Completion status of operation:

- 0 = success,
- RTERR_IDNOTFOU = not boolean value (not logged)
- negative return value if other error (logged)

6.2.3.56 rtJsonTryDecNull()

```
EXTERNJSON int rtJsonTryDecNull (
             OSCTXT * pctxt )
```

This function will attempt to decode a null value at the current buffer or stream position.

If the attempt fails, the decode cursor is reset to the position it was at when the function was called and an RTERR_I↩ DNOTFOU status is returned. The error is not logged in the context.

Parameters

pctxt	Pointer to context block structure.

Returns

Completion status of operation:

- 0 = success,
- RTERR_IDNOTFOU = not boolean value (not logged)
- negative return value if other error (logged)

Chapter 7

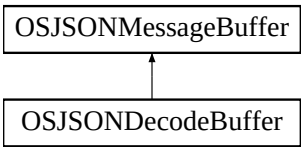
Class Documentation

7.1 OSJSONDecodeBuffer Class Reference

The OSJSONDecodeBuffer class is derived from the OSJSONMessageBuffer base class.

`#include <OSJSONDecodeBuffer.h>`

Inheritance diagram for OSJSONDecodeBuffer:



Public Member Functions

- OSJSONDecodeBuffer (const char *jsonFile)

  This version of the OSJSONDecodeBuffer constructor takes a name of a file that contains JSON data to be decoded and constructs a buffer.
- OSJSONDecodeBuffer (const OSOCTET *msgbuf, size_t bufsiz)

  This version of the OSJSONDecodeBuffer constructor takes parameters describing a message in memory to be decoded and constructs a buffer.
- OSJSONDecodeBuffer (OSRTInputStream &inputStream)

  This version of the OSJSONDecodeBuffer constructor takes a reference to the OSInputStream object.
- virtual EXTJSONMETHOD int init ()

  This method initializes the decode message buffer.
- virtual OSBOOL isA (Type bufferType)

  This is a virtual method that must be overridden by derived classes to allow identification of the class.

Protected Attributes

- OSRTInputStream * mpInputStream

  Input source for message to be decoded.
- OSBOOL mbOwnStream

  This is set to true if this object creates the underlying stream object.

Additional Inherited Members

7.1.1 Detailed Description

The OSJSONDecodeBuffer class is derived from the OSJSONMessageBuffer base class.

It contains variables and methods specific to decoding JSON messages. It is used to manage an input buffer or stream containing a message to be decoded.

Definition at line 40 of file OSJSONDecodeBuffer.h.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 OSJSONDecodeBuffer() [1/3]

```
OSJSONDecodeBuffer::OSJSONDecodeBuffer (
            const char * jsonFile )
```

This version of the OSJSONDecodeBuffer constructor takes a name of a file that contains JSON data to be decoded and constructs a buffer.

Parameters

jsonFile	A pointer to name of file to be decoded.

7.1.2.2 OSJSONDecodeBuffer() [2/3]

```
OSJSONDecodeBuffer::OSJSONDecodeBuffer (
            const OSOCTET * msgbuf,
            size_t bufsiz )
```

This version of the OSJSONDecodeBuffer constructor takes parameters describing a message in memory to be decoded and constructs a buffer.

Parameters

msgbuf	A pointer to a buffer containing an JSON message.
bufsiz	Size of the message buffer.

7.1.2.3 OSJSONDecodeBuffer() [3/3]

```
OSJSONDecodeBuffer::OSJSONDecodeBuffer (
            OSRTInputStream & inputStream )
```

This version of the OSJSONDecodeBuffer constructor takes a reference to the OSInputStream object.

The stream is assumed to have been previuously initialized to point at an encoded JSON message.

Parameters

inputStream	reference to the OSInputStream object

7.1.3 Member Function Documentation

7.1.3.1 init()

```
virtual EXTJSONMETHOD int OSJSONDecodeBuffer::init ( )  [virtual]
```

This method initializes the decode message buffer.

Returns

Completion status of operation:

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

7.1.3.2 isA()

```
virtual OSBOOL OSJSONDecodeBuffer::isA (
            Type bufferType )  [inline], [virtual]
```

This is a virtual method that must be overridden by derived classes to allow identification of the class.

The base class variant is abstract. This method matches an enumerated identifier defined in the base class. One identifier is declared for each of the derived classes.

Parameters

bufferType	Enumerated identifier specifying a derived class. This type is defined as a public access type in the OSRTMessageBufferIF base interface. Possible values include BEREncode, BERDecode, PEREncode, PERDecode, JSONEncode, and JSONDecode.

Returns

Boolean result of the match operation. True if the `bufferType` argument is `JSONDecode`. argument.

Definition at line 108 of file OSJSONDecodeBuffer.h.

7.1.4 Member Data Documentation

7.1.4.1 mbOwnStream

```
OSBOOL OSJSONDecodeBuffer::mbOwnStream  [protected]
```

This is set to true if this object creates the underlying stream object.

In this case, the stream will be deleted in the object's destructor.

Definition at line 52 of file OSJSONDecodeBuffer.h.

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

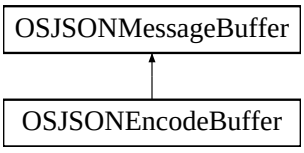
- OSJSONDecodeBuffer.h

7.2 OSJSONEncodeBuffer Class Reference

The OSJSONEncodeBuffer class is derived from the OSJSONMessageBuffer base class.

```
#include <OSJSONEncodeBuffer.h>
```

Inheritance diagram for OSJSONEncodeBuffer:



Public Member Functions

- EXTJSONMETHOD OSJSONEncodeBuffer ()

  Default constructor.
- EXTJSONMETHOD OSJSONEncodeBuffer (OSOCTET *pMsgBuf, size_t msgBufLen)

  This constructor allows a static message buffer to be specified to receive the encoded message.
- virtual size_t getMsgLen ()

  This method returns the length of a previously encoded JSON message.
- virtual EXTJSONMETHOD int init ()

  This method reinitializes the encode buffer to allow a new message to be encoded.
- virtual OSBOOL isA (Type bufferType)

  This is a virtual method that must be overridden by derived classes to allow identification of the class.
- void nullTerminate ()

  This method adds a null-terminator character ('\0') at the current buffer position.
- virtual EXTJSONMETHOD long write (const char *filename)

  This method writes the encoded message to the given file.
- virtual EXTJSONMETHOD long write (FILE *fp)

  This version of the write method writes to a file that is specified by a FILE pointer.

Additional Inherited Members

7.2.1 Detailed Description

The OSJSONEncodeBuffer class is derived from the OSJSONMessageBuffer base class.

It contains variables and methods specific to encoding JSON messages. It is used to manage the buffer into which a message is to be encoded.

Definition at line 38 of file OSJSONEncodeBuffer.h.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 OSJSONEncodeBuffer()

```
EXTJSONMETHOD OSJSONEncodeBuffer::OSJSONEncodeBuffer (
            OSOCTET * pMsgBuf,
            size_t msgBufLen )
```

This constructor allows a static message buffer to be specified to receive the encoded message.

Parameters

pMsgBuf	A pointer to a fixed size message buffer to receive the encoded message.
msgBufLen	Size of the fixed-size message buffer.

7.2.3 Member Function Documentation

7.2.3.1 getMsgLen()

```
virtual size_t OSJSONEncodeBuffer::getMsgLen ( )  [inline], [virtual]
```

This method returns the length of a previously encoded JSON message.

Returns

Length of the JSON message encapsulated within this buffer object.

Definition at line 67 of file OSJSONEncodeBuffer.h.

7.2.3.2 init()

```
virtual EXTJSONMETHOD int OSJSONEncodeBuffer::init ( )  [virtual]
```

This method reinitializes the encode buffer to allow a new message to be encoded.

This makes it possible to reuse one message buffer object in a loop to encode multiple messages. After this method is called, any previously encoded message in the buffer will be overwritten on the next encode call.

7.2.3.3 isA()

```
virtual OSBOOL OSJSONEncodeBuffer::isA (
            Type bufferType )  [inline], [virtual]
```

This is a virtual method that must be overridden by derived classes to allow identification of the class.

The base class variant is abstract. This method matches an enumerated identifier defined in the base class. One identifier is declared for each of the derived classes.

Parameters

bufferType	Enumerated identifier specifying a derived class. This type is defined as a public access type in the OSRTMessageBufferIF base interface. Possible values include BEREncode, BERDecode, PEREncode, PERDecode, JSONEncode, and JSONDecode.

Returns

Boolean result of the match operation. True if the `bufferType` argument is `JSONEncode`. argument.

Definition at line 95 of file OSJSONEncodeBuffer.h.

7.2.3.4 write() [1/2]

```
virtual EXTJSONMETHOD long OSJSONEncodeBuffer::write (
            const char * filename )  [virtual]
```

This method writes the encoded message to the given file.

Parameters

filename	The name of file to which the encoded message will be written.

Returns

Number of octets actually written. This value may be less than the actual message length if an error occurs.

7.2.3.5 write() [2/2]

```
virtual EXTJSONMETHOD long OSJSONEncodeBuffer::write (
            FILE * fp )  [virtual]
```

This version of the write method writes to a file that is specified by a FILE pointer.

Parameters

fp	Pointer to FILE structure to which the encoded message will be written.

Returns

Number of octets actually written. This value may be less than the actual message length if an error occurs.

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

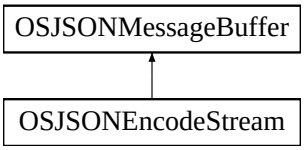
- OSJSONEncodeBuffer.h

7.3 OSJSONEncodeStream Class Reference

The OSJSONEncodeStream class is derived from the OSJSONMessageBuffer base class.

```
#include <OSJSONEncodeStream.h>
```

Inheritance diagram for OSJSONEncodeStream:



Public Member Functions

- EXTJSONMETHOD OSJSONEncodeStream (OSRTOutputStream &outputStream)

  This version of the OSJSONEncodeStream constructor takes a reference to the OSOutputStream object.
- OSJSONEncodeStream (OSRTOutputStream *pOutputStream, OSBOOL ownStream=TRUE)

  This version of the OSJSONEncodeStream constructor takes a pointer to the OSRTOutputStream object.
- EXTJSONMETHOD int encodeAttr (const OSUTF8CHAR *name, const OSUTF8CHAR *value)

  This function encodes an attribute in which the name and value are given as null-terminated UTF-8 strings.
- EXTJSONMETHOD int encodeText (const OSUTF8CHAR *value)

  This method encodes JSON textual content.
- virtual EXTJSONMETHOD int init ()

  This method reinitializes the encode stream to allow a new message to be encoded.
- virtual OSBOOL isA (Type bufferType)

  This is a virtual method that must be overridden by derived classes to allow identification of the class.
- virtual const OSOCTET * getMsgPtr ()

  This is a virtual method that must be overridden by derived classes to allow access to the stored message.
- OSRTOutputStream * getStream () const

  This method returns the output stream associated with the object.

Protected Attributes

- OSRTOutputStream * mpStream

  A pointer to an OSRTOutputStream object.
- OSBOOL mbOwnStream

  TRUE if the OSJSONEncodeStream object will close and free the stream in the destructor.
- OSCTXT * mpCtxt

  Internal pointer to the context structure associated with the stream for making C function calls.

Additional Inherited Members

7.3.1 Detailed Description

The OSJSONEncodeStream class is derived from the OSJSONMessageBuffer base class.

It contains variables and methods specific to streaming encoding JSON messages. It is used to manage the stream into which a message is to be encoded.

Definition at line 40 of file OSJSONEncodeStream.h.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 OSJSONEncodeStream() [1/2]

```
EXTJSONMETHOD OSJSONEncodeStream::OSJSONEncodeStream (
            OSRTOutputStream & outputStream )
```

This version of the OSJSONEncodeStream constructor takes a reference to the OSOutputStream object.

The stream is assumed to have been previously initialized.

Parameters

outputStream	reference to the OSOutputStream object

7.3.2.2 OSJSONEncodeStream() [2/2]

```
OSJSONEncodeStream::OSJSONEncodeStream (
            OSRTOutputStream * pOutputStream,
            OSBOOL ownStream = TRUE )
```

This version of the OSJSONEncodeStream constructor takes a pointer to the OSRTOutputStream object.

The stream is assumed to have been previously initialized. If ownStream is set to TRUE, then stream will be closed and freed in the destructor.

Parameters

pOutputStream	reference to the OSOutputStream object
ownStream	set ownership for the passed stream object.

7.3.3 Member Function Documentation

7.3.3.1 encodeAttr()

```
EXTJSONMETHOD int OSJSONEncodeStream::encodeAttr (
            const OSUTF8CHAR * name,
            const OSUTF8CHAR * value )
```

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

Parameters

name	Attribute name.
value	UTF-8 string value to be encoded.

Returns

Completion status of operation:

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

7.3.3.2 encodeText()

```
EXTJSONMETHOD int OSJSONEncodeStream::encodeText (
            const OSUTF8CHAR * value )
```

This method encodes JSON textual content.

JSON metadata characters are escaped. The input value is specified in UTF-8 character format.

Parameters

value	UTF-8 string value to be encoded.

Returns

Completion status of operation:

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

7.3.3.3 getMsgPtr()

```
virtual const OSOCTET* OSJSONEncodeStream::getMsgPtr ( )  [inline], [virtual]
```

This is a virtual method that must be overridden by derived classes to allow access to the stored message.

The base class implementation returns a null value.

Returns

A pointer to the stored message.

Definition at line 135 of file OSJSONEncodeStream.h.

7.3.3.4 getStream()

```
OSRTOutputStream* OSJSONEncodeStream::getStream ( ) const  [inline]
```

This method returns the output stream associated with the object.

Returns

A pointer to the output stream.

Definition at line 142 of file OSJSONEncodeStream.h.

7.3.3.5 init()

```
virtual EXTJSONMETHOD int OSJSONEncodeStream::init ( )  [virtual]
```

This method reinitializes the encode stream to allow a new message to be encoded.

This makes it possible to reuse one stream object in a loop to encode multiple messages.

7.3.3.6 isA()

```
virtual OSBOOL OSJSONEncodeStream::isA (
            Type bufferType )  [inline], [virtual]
```

This is a virtual method that must be overridden by derived classes to allow identification of the class.

The base class variant is abstract. This method matches an enumerated identifier defined in the base class. One identifier is declared for each of the derived classes.

Parameters

bufferType	Enumerated identifier specifying a derived class. This type is defined as a public access type in the OSRTMessageBufferIF base interface. Possible values include BEREncode, BERDecode, PEREncode, PERDecode, JSONEncode, and JSONDecode.

Returns

Boolean result of the match operation. True if the `bufferType` argument is `JSONEncode`. argument.

Definition at line 124 of file OSJSONEncodeStream.h.

7.3.4 Member Data Documentation

7.3.4.1 mbOwnStream

```
OSBOOL OSJSONEncodeStream::mbOwnStream  [protected]
```

TRUE if the OSJSONEncodeStream object will close and free the stream in the destructor.

Definition at line 47 of file OSJSONEncodeStream.h.

7.3.4.2 mpCtxt

```
OSCTXT* OSJSONEncodeStream::mpCtxt  [protected]
```

Internal pointer to the context structure associated with the stream for making C function calls.

Definition at line 51 of file OSJSONEncodeStream.h.

7.3.4.3 mpStream

```
OSRTOutputStream* OSJSONEncodeStream::mpStream  [protected]
```

A pointer to an OSRTOutputStream object.

Definition at line 43 of file OSJSONEncodeStream.h.

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

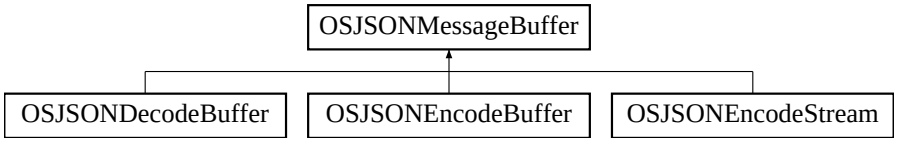
- OSJSONEncodeStream.h

7.4 OSJSONMessageBuffer Class Reference

The JSON message buffer class is derived from the OSMessageBuffer base class.

```
#include <OSJSONMessageBuffer.h>
```

Inheritance diagram for OSJSONMessageBuffer:



Protected Member Functions

- EXTJSONMETHOD OSJSONMessageBuffer (Type bufferType, OSRTContext *pContext=0)

  The protected constructor creates a new context and sets the buffer class type.

7.4.1 Detailed Description

The JSON message buffer class is derived from the OSMessageBuffer base class.

It is the base class for the OSJSONEncodeBuffer and OSJSONDecodeBuffer classes. It contains variables and methods specific to encoding or decoding JSON messages. It is used to manage the buffer into which a message is to be encoded or decoded.

Definition at line 42 of file OSJSONMessageBuffer.h.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 OSJSONMessageBuffer()

```
EXTJSONMETHOD OSJSONMessageBuffer::OSJSONMessageBuffer (
            Type bufferType,
            OSRTContext * pContext = 0 )  [protected]
```

The protected constructor creates a new context and sets the buffer class type.

Parameters

bufferType	Type of message buffer that is being created (for example, JSONEncode or JSONDecode).
pContext	Pointer to a context to use. If NULL, new context will be allocated.

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

- OSJSONMessageBuffer.h

Chapter 8

File Documentation

8.1 OSJSONDecodeBuffer.h File Reference

JSON decode buffer or stream class definition.

```
#include "rtxsrc/OSRTInputStream.h"
#include "rtjsonsrc/OSJSONMessageBuffer.h"
```

Classes

- class OSJSONDecodeBuffer

  The OSJSONDecodeBuffer class is derived from the OSJSONMessageBuffer base class.

8.1.1 Detailed Description

JSON decode buffer or stream class definition.

8.2 OSJSONEncodeBuffer.h File Reference

JSON encode message buffer class definition.

```
#include "rtjsonsrc/OSJSONMessageBuffer.h"
```

Classes

- class OSJSONEncodeBuffer

  The OSJSONEncodeBuffer class is derived from the OSJSONMessageBuffer base class.

8.2.1 Detailed Description

JSON encode message buffer class definition.

8.3 OSJSONEncodeStream.h File Reference

JSON encode stream class definition.

```
#include "rtxsrc/OSRTOutputStream.h"
#include "rtjsonsrc/OSJSONMessageBuffer.h"
```

Classes

- class OSJSONEncodeStream

  The OSJSONEncodeStream class is derived from the OSJSONMessageBuffer base class.

8.3.1 Detailed Description

JSON encode stream class definition.

8.4 OSJSONMessageBuffer.h File Reference

JSON encode/decode buffer and stream base class.

```
#include "rtxsrc/OSRTMsgBuf.h"
#include "rtjsonsrc/osrtjson.h"
```

Classes

- class OSJSONMessageBuffer

  The JSON message buffer class is derived from the OSMessageBuffer base class.

8.4.1 Detailed Description

JSON encode/decode buffer and stream base class.

8.5 osrtjson.h File Reference

JSON low-level C encode/decode functions.

```
#include "rtxsrc/osMacros.h"
#include "rtxsrc/osSysTypes.h"
#include "rtxsrc/rtxCommon.h"
#include "rtxsrc/rtxError.h"
#include "rtxsrc/rtxBuffer.h"
#include "rtxsrc/rtxMemory.h"
#include "rtxsrc/rtxText.h"
#include "rtjsonsrc/rtJsonExternDefs.h"
```

Macros

- #define OSUPCASE 0x00008000

  The upper-case flag: if set, hex strings will be encoded in upper case.
- #define rtJsonEncIntValue rtxTxtWriteInt

  This function encodes a variable of the XSD integer type.
- #define rtJsonEncInt64Value rtxTxtWriteInt64

  This function encodes a variable of the XSD integer type.
- #define rtJsonEncDecrIndent rtxIndentDecr

  This decreases the indentation level set in the given context by updating the indent member.
- #define rtJsonEncIncrIndent rtxIndentIncr

  This increases the indentation level set in the given context by updating the indent member.
- #define rtJsonEncResetIndent rtxIndentReset

  This resets the indentation level in the given context to zero.
- #define rtJsonGetIndentLevels rtxGetIndentLevels

  This returns the number of levels of indentation rtJsonEncIndent is using.
- #define rtJsonEncUIntValue rtxTxtWriteUInt

  This function encodes a variable of the XSD unsigned integer type.
- #define rtJsonEncUInt64Value rtxTxtWriteUInt64

  This function encodes a variable of the XSD integer type.
- #define rtJsonDecPeekChar(pctxt, pch) rtxTxtPeekChar(pctxt, pch, TRUE)

  This function determines the next non-whitespace character in the input.
- #define rtJsonDecPeekChar2(pctxt) rtxTxtPeekChar2(pctxt, TRUE)

  This function determines the next non-whitespace character in the input.

Functions

- EXTERNJSON int rtJsonEncAnyAttr (OSCTXT *pctxt, const OSRTDList *pvalue)

  This function encodes a list of OSAnyAttr attributes in which the name and value are given as a UTF-8 string.
- EXTERNJSON int rtJsonEncBase64StrValue (OSCTXT *pctxt, OSSIZE nocts, const OSOCTET *value)

  This function encodes a variable of the XSD base64Binary type.
- EXTERNJSON int rtJsonEncBoolValue (OSCTXT *pctxt, OSBOOL value)

This function encodes a variable of the XSD boolean type.

- EXTERNJSON int rtJsonEncGYear (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gYear value into a JSON string representation.
- EXTERNJSON int rtJsonEncGYearMonth (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gYearMonth value into a JSON string representation.
- EXTERNJSON int rtJsonEncGMonth (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gMonth value into a JSON string representation.
- EXTERNJSON int rtJsonEncGMonthDay (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gMonthDay value into a JSON string representation.
- EXTERNJSON int rtJsonEncGDay (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric gDay value into a JSON string representation.
- EXTERNJSON int rtJsonEncDate (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a variable of the XSD 'date' type as a string.
- EXTERNJSON int rtJsonEncTime (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a variable of the XSD 'time' type as a JSON string.
- EXTERNJSON int rtJsonEncDateTime (OSCTXT *pctxt, const OSXSDDateTime *pvalue)

  This function encodes a numeric date/time value into a string representation.
- EXTERNJSON int rtJsonEncDecimalValue (OSCTXT *pctxt, OSREAL value, const OSDecimalFmt *pFmtSpec)

  This function encodes a value of the XSD decimal type.
- EXTERNJSON int rtJsonEncDoubleValue (OSCTXT *pctxt, OSREAL value, const OSDoubleFmt *pFmtSpec)

  This function encodes a value of the XSD double or float type.
- EXTERNJSON int rtJsonEncFloatValue (OSCTXT *pctxt, OSREAL value, const OSDoubleFmt *pFmtSpec)

  This function encodes a variable of the XSD float type.
- EXTERNJSON int rtJsonEncHexStr (OSCTXT *pctxt, OSSIZE nocts, const OSOCTET *data)

  This function encodes the given data as a JSON string, using a hexadecimal representation of the data.
- EXTERNJSON int rtJsonEncHexValue (OSCTXT *pctxt, OSSIZE nocts, const OSOCTET *data)

  This function encodes the given data as a sequence of hexadecimal characters.
- EXTERNJSON int rtJsonEncBitStrValueV72 (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data)

  This function encodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonEncBitStrValueExtV72 (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data, OSS↩ IZE dataSize, const OSOCTET *extData)

  This function encodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonEncBitStrValue (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data)

  This function encodes a variable of the ASN.1 Bit string type as a JSON object, as required by X.697 for a BIT STRING without a fixed length.
- EXTERNJSON int rtJsonEncFixedBitStrValue (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data)

  This function encodes a variable of the ASN.1 Bit string type as a JSON string, as required by X.697 for a BIT STRING with a fixed length.
- EXTERNJSON int rtJsonEncBitStrValueExt (OSCTXT *pctxt, OSSIZE nbits, const OSOCTET *data, OSSIZE dataSize, const OSOCTET *extData)

  This function encodes a variable of the ASN.1 Bit string type as a JSON object, as required by X.697 for a BIT STRING without a fixed length.
- EXTERNJSON int rtJsonEncIndent (OSCTXT *pctxt)

  This function:
- EXTERNJSON int rtJsonEncStringObject (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value)

  This function encodes a JSON object containing a string value.

- EXTERNJSON int rtJsonEncStringObject2 (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const OSUTF8CHAR *value, size_t valueLen)

  This function encodes a JSON object containing a string value.
- EXTERNJSON int rtJsonEncStringPair (OSCTXT *pctxt, const OSUTF8CHAR *name, const OSUTF8CHAR *value)

  This function encodes a name/value pair.
- EXTERNJSON int rtJsonEncStringPair2 (OSCTXT *pctxt, const OSUTF8CHAR *name, size_t nameLen, const OSUTF8CHAR *value, size_t valueLen)

  This function encodes a name/value pair.
- EXTERNJSON int rtJsonEncStringValue (OSCTXT *pctxt, const OSUTF8CHAR *value)

  This function encodes a UTF8 string value.
- EXTERNJSON int rtJsonEncStringValue2 (OSCTXT *pctxt, const OSUTF8CHAR *value, size_t valueLen)

  This function encodes a UTF8 string value.
- EXTERNJSON int rtJsonEncChars (OSCTXT *pctxt, const char *value, OSSIZE valueLen)

  This function encodes the given number of characters from a character string value as a JSON string.
- EXTERNJSON int rtJsonEncCharStr (OSCTXT *pctxt, const char *value)

  This function encodes a character string value as a JSON string.
- EXTERNJSON int rtJsonEncStringNull (OSCTXT *pctxt)

  This function encodes an asn.1 NULL type as string.
- EXTERNJSON int rtJsonEncStringRaw (OSCTXT *pctxt, const OSUTF8CHAR *value)

  This function encodes a raw string without any quotation.
- EXTERNJSON int rtJsonEncUnicodeData (OSCTXT *pctxt, const OSUNICHAR *value, OSSIZE nchars)

  This function encodes a variable that contains UCS-2 / UTF-16 characters.
- EXTERNJSON int rtJsonEncUCS4Data (OSCTXT *pctxt, const OS32BITCHAR *value, OSSIZE nchars)

  This function encodes a variable that contains UCS-4 / UTF-32 characters.
- EXTERNJSON int rtJsonEncStartObject (OSCTXT *pctxt, const OSUTF8CHAR *name, OSBOOL noComma)

  This function encodes the beginning of a JSON object.
- EXTERNJSON int rtJsonEncEndObject (OSCTXT *pctxt)

  This function encodes the end of a JSON object.
- EXTERNJSON int rtJsonEncBetweenObject (OSCTXT *pctxt)

  This function encodes the characters separating the JSON name and value.
- EXTERNJSON int rtJsonDecAnyElem (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes an arbitrary block of JSON-encoded data into a string variable.
- EXTERNJSON int rtJsonDecAnyElem2 (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This version of rtJsonDecAnyElem assumes the element name has been pushed on the element name stack in the context.
- EXTERNJSON int rtJsonDecAnyType (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes an arbitrary block of JSON-encoded data into a string variable.
- EXTERNJSON int rtJsonDecBase64Str (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, size_t bufsize)

  This function decodes a contents of a Base64-encode binary string into a static memory structure.
- EXTERNJSON int rtJsonDecBase64Str64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, size_t bufsize)

  This function is identical to rtJsonDecBase64Str except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecDynBase64Str (OSCTXT *pctxt, OSDynOctStr *pvalue)

  This function decodes a contents of a Base64-encode binary string.
- EXTERNJSON int rtJsonDecDynBase64Str64 (OSCTXT *pctxt, OSDynOctStr64 *pvalue)

  This function is identical to rtJsonDecDynBase64Str64 except that it supports 64-bit integer lengths on 64-bit systems.
- EXTERNJSON int rtJsonDecBool (OSCTXT *pctxt, OSBOOL *pvalue)

This function decodes a variable of the boolean type.

- EXTERNJSON int rtJsonTryDecBool (OSCTXT *pctxt, OSBOOL *pvalue)

  This function will attempt to decode a boolean variable at the current buffer or stream position.
- EXTERNJSON int rtJsonDecDate (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'date' type.
- EXTERNJSON int rtJsonDecTime (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'time' type.
- EXTERNJSON int rtJsonDecDateTime (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'dateTime' type.
- EXTERNJSON int rtJsonDecGYear (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gYear' type.
- EXTERNJSON int rtJsonDecGYearMonth (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gYearMonth' type.
- EXTERNJSON int rtJsonDecGMonth (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gMonth' type.
- EXTERNJSON int rtJsonDecGMonthDay (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gMonthDay' type.
- EXTERNJSON int rtJsonDecGDay (OSCTXT *pctxt, OSXSDDateTime *pvalue)

  This function decodes a variable of the XSD 'gDay' type.
- EXTERNJSON int rtJsonDecDecimal (OSCTXT *pctxt, OSREAL *pvalue, int totalDigits, int fractionDigits)

  This function decodes the contents of a decimal data type.
- EXTERNJSON int rtJsonDecDouble (OSCTXT *pctxt, OSREAL *pvalue)

  This function decodes the contents of a float or double data type.
- EXTERNJSON int rtJsonDecHexToCharStr (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes a JSON string, interpreting the content as the hexadecimal represention of the bytes for a character string, which it returns.
- EXTERNJSON int rtJsonDecHexStr (OSCTXT *pctxt, OSOCTET *pvalue, OSUINT32 *pnocts, size_t bufsize)

  This function decodes a JSON string consisting of hexadecimal characters into a static memory structure.
- EXTERNJSON int rtJsonDecHexStr64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, size_t bufsize)

  This function is identical to rtJsonDecHexStr except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecHexData64 (OSCTXT *pctxt, OSOCTET *pvalue, OSSIZE *pnocts, size_t bufsize)

  This function deocdes a sequence of heaxadecimal characters until a non-hexadecimal character (which is not consumed) is found OR the preallocated array is filled.
- EXTERNJSON int rtJsonDecDynHexStr (OSCTXT *pctxt, OSDynOctStr *pvalue)

  This function decodes a JSON string consisting of hexadecimal characters.
- EXTERNJSON int rtJsonDecDynHexStr64 (OSCTXT *pctxt, OSDynOctStr64 *pvalue)

  This function is identical to rtJsonDecDynHexStr except that it supports 64-bit integer lengths on 64-bit systems.
- EXTERNJSON int rtJsonDecDynHexData64 (OSCTXT *pctxt, OSDynOctStr64 *pvalue)

  This function decodes a sequence of hexadecimal characters until a non-hexadecimal character (which is not consumed) is found.
- EXTERNJSON int rtJsonDecDynBitStrV72 (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET **data)

  This function decodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonDecDynBitStr64V72 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET **data)

  This function is identical to rtJsonDecDynBitStrV72 except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecBitStrValueV72 (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize)

  This function decodes a variable of the ASN.1 Bit string type.

- EXTERNJSON int rtJsonDecBitStrValue64V72 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize)

  This function is identical to rtJsonDecBitStrValueV72 except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecBitStrValueExtV72 (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function decodes a variable of the ASN.1 Bit string type.
- EXTERNJSON int rtJsonDecBitStrValueExt64V72 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function is identical to rtJsonDecBitStrValueExtV72 except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecDynBitStr (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET **data)

  This function decodes a value of an ASN.1 BIT STRING type, not constrained to a fixed length, encoded according to X.697 as a JSON object.
- EXTERNJSON int rtJsonDecDynBitStr64 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET **data)

  This function is identical to rtJsonDecDynBitStr except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecFixedDynBitStr (OSCTXT *pctxt, OSSIZE nbits, OSOCTET **data)

  This function decodes a value of an ASN.1 BIT STRING type constrained to a fixed length, encoded according to X.697 as a JSON string.
- EXTERNJSON int rtJsonDecBitStrValue (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize)

  This function decodes a value of an ASN.1 BIT STRING type that is not constrained to a fixed length, encoded according to X.697 as a JSON object.
- EXTERNJSON int rtJsonDecBitStrValue64 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize)

  This function is identical to rtJsonDecBitStrValue except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecFixedBitStrValue (OSCTXT *pctxt, OSSIZE nbits, OSOCTET *data, OSSIZE bufsize)

  This function decodes a value of an ASN.1 BIT STRING type constrained to have the given fixed length, encoded according to X.697 as a JSON string.
- EXTERNJSON int rtJsonDecBitStrValueExt (OSCTXT *pctxt, OSUINT32 *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function decodes a value of a ASN.1 BIT STRING type without a fixed-length constraint, encoded according to X.697 as JSON object.
- EXTERNJSON int rtJsonDecBitStrValueExt64 (OSCTXT *pctxt, OSSIZE *nbits, OSOCTET *data, OSSIZE bufsize, OSOCTET **extdata)

  This function is identical to rtJsonDecBitStrValueExt except that it supports lengths up to 64-bits in size on 64-bit machines.
- EXTERNJSON int rtJsonDecMatchChar (OSCTXT *pctxt, OSUTF8CHAR ch)

  This function attempts to match the given character, skipping over any whitesapce, if necessary.
- EXTERNJSON int rtJsonDecMatchCharStr (OSCTXT *pctxt, const char *token)

  This function decodes a JSON string and matches with a given token.
- EXTERNJSON int rtJsonDecMatchObjectStart (OSCTXT *pctxt, const OSUTF8NameAndLen *nameArray, size_t numNames)

  This function matches the start of a JSON object.
- EXTERNJSON int rtJsonDecMatchToken (OSCTXT *pctxt, const OSUTF8CHAR *token)

  This function decodes a JSON string and matches with a given token.
- EXTERNJSON int rtJsonDecMatchToken2 (OSCTXT *pctxt, const OSUTF8CHAR *token, size_t tokenLen)

  This function decodes a JSON string and matches with a given token.
- EXTERNJSON int rtJsonDecNameValuePair (OSCTXT *pctxt, OSUTF8NVP *pvalue)

  This function decodes a name/value pair.
- EXTERNJSON int rtJsonDecNull (OSCTXT *pctxt)

This function decodes a JSON null.

- EXTERNJSON int rtJsonTryDecNull (OSCTXT *pctxt)

  This function will attempt to decode a null value at the current buffer or stream position.
- EXTERNJSON int rtJsonDecNumberString (OSCTXT *pctxt, char **ppCharStr)

  This function decodes a JSON number into a character string variable.
- EXTERNJSON int rtJsonDecStringObject (OSCTXT *pctxt, const OSUTF8CHAR *name, OSUTF8CHAR **ppvalue)

  This function decodes a JSON object containing a single entry with the given key (name), and returns the key's associated value, which must be a JSON string, via ppvalue.
- EXTERNJSON int rtJsonDecStringValue (OSCTXT *pctxt, OSUTF8CHAR **ppvalue)

  This function decodes the contents of a string data type.
- EXTERNJSON int rtJsonDecStringValueArray (OSCTXT *pctxt, OSUTF8CHAR *pvalue, OSSIZE bufsize)

  This function is the same as rtJsonDecStringValue except that it decodes into a character array.
- EXTERNJSON int rtJsonDecXmlStringValue (OSCTXT *pctxt, OSXMLSTRING *pvalue)

  This function decodes the contents of an XML string data type.
- EXTERNJSON int rtJsonDecUCS2String (OSCTXT *pctxt, OSUNICHAR **ppstr, OSSIZE *pnchars)

  This function is used to decode input UTF-8 data into a UCS-2 / UTF-16 character string.
- EXTERNJSON int rtJsonDecUCS4String (OSCTXT *pctxt, OS32BITCHAR **ppstr, OSSIZE *pnchars)

  This function is used to decode input UTF-8 data into a UCS-4 / UTF-32 character string.
- EXTERNJSON size_t rtJsonGetElemIdx (OSCTXT *pctxt, const OSUTF8NameAndLen nameArray[ ], size_t nrows)

  This function determines which of several possible JSON strings appears next in the input.

8.5.1 Detailed Description

JSON low-level C encode/decode functions.

8.5.2 Macro Definition Documentation

8.5.2.1 OSUPCASE

```
#define OSUPCASE 0x00008000
```

The upper-case flag: if set, hex strings will be encoded in upper case.

Definition at line 86 of file osrtjson.h.

8.6 rtJsonCppMsgBuf.h File Reference

This file is deprecated.

```
#include "rtsrc/asn1CppTypes.h"
#include "rtjsonsrc/OSJSONEncodeBuffer.h"
#include "rtjsonsrc/OSJSONEncodeStream.h"
#include "rtjsonsrc/OSJSONDecodeBuffer.h"
```

8.6.1 Detailed Description

This file is deprecated.

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

8.7 rtJsonExternDefs.h File Reference

JSON external definitions macro.

8.7.1 Detailed Description

JSON external definitions macro.

This is used for Windows to properly declare function scope within DLL's.

Index