ASN1C C# Common Runtime

Version 7.7

Objective Systems, Inc.

April 2023

ASN1C C# Common Runtime

Author's Contact Information. Comments, suggestions, and inquiries regarding ASN1C or this document may be sent by electronic mail to <info@obj-sys.com>.

Table of Contents

List of Tables

Chapter 1. Namespace Documentation

Com

Namespaces

- struct Com::Objsys

Com::Objsys

Namespaces

- struct Com::Objsys::Asn1

Com::Objsys::Asn1

Namespaces

- struct Com::Objsys::Asn1::Runtime

Com::Objsys::Asn1::Runtime

Classes

- struct Com::Objsys::Asn1::Runtime::Asn18BitCharString
- struct Com::Objsys::Asn1::Runtime::Asn1BigInteger
- struct Com::Objsys::Asn1::Runtime::Asn1BitString
- struct Com::Objsys::Asn1::Runtime::Asn1BMPString
- struct Com::Objsys::Asn1::Runtime::Asn1BMPStringCharset
- struct Com::Objsys::Asn1::Runtime::Asn1Boolean
- struct Com::Objsys::Asn1::Runtime::Asn1CharRange
- struct Com::Objsys::Asn1::Runtime::Asn1CharSet
- struct Com::Objsys::Asn1::Runtime::Asn1CharString
- struct Com::Objsys::Asn1::Runtime::Asn1Choice
- struct Com::Objsys::Asn1::Runtime::Asn1ChoiceExt
- struct Com::Objsys::Asn1::Runtime::Asn1ConsVioException
- struct Com::Objsys::Asn1::Runtime::Asn1DecodeBuffer

- struct Com::Objsys::Asn1::Runtime::Asn1DiscreteCharSet
- struct Com::Objsys::Asn1::Runtime::Asn1EncodeBuffer
- struct Com::Objsys::Asn1::Runtime::Asn1EndOfBufferException
- struct Com::Objsys::Asn1::Runtime::Asn1Enumerated
- struct Com::Objsys::Asn1::Runtime::Asn1Exception
- struct Com::Objsys::Asn1::Runtime::Asn1GeneralizedTime
- struct Com::Objsys::Asn1::Runtime::Asn1GeneralString
- struct Com::Objsys::Asn1::Runtime::Asn1GraphicString
- struct Com::Objsys::Asn1::Runtime::Asn1IA5String
- struct Com::Objsys::Asn1::Runtime::Asn1IA5StringCharset
- struct Com::Objsys::Asn1::Runtime::Asn1InputStream
- struct Com::Objsys::Asn1::Runtime::Asn1Integer
- struct Com::Objsys::Asn1::Runtime::Asn1InvalidArgException
- struct Com::Objsys::Asn1::Runtime::Asn1InvalidChoiceOptionException
- struct Com::Objsys::Asn1::Runtime::Asn1InvalidEnumException
- struct Com::Objsys::Asn1::Runtime::Asn1InvalidLengthException
- struct Com::Objsys::Asn1::Runtime::Asn1InvalidObjectIDException
- struct Com::Objsys::Asn1::Runtime::Asn1MessageBuffer
- struct Com::Objsys::Asn1::Runtime::Asn1MissingRequiredException
- struct Com::Objsys::Asn1::Runtime::Asn1NamedEventHandler
- struct Com::Objsys::Asn1::Runtime::Asn1Null
- struct Com::Objsys::Asn1::Runtime::Asn1NumericString
- struct Com::Objsys::Asn1::Runtime::Asn1NumericStringCharset
- struct Com::Objsys::Asn1::Runtime::Asn1ObjectDescriptor
- struct Com::Objsys::Asn1::Runtime::Asn1ObjectIdentifier
- struct Com::Objsys::Asn1::Runtime::Asn1OctetString
- struct Com::Objsys::Asn1::Runtime::Asn1OpenExt
- struct Com::Objsys::Asn1::Runtime::Asn1OpenType
- struct Com::Objsys::Asn1::Runtime::Asn1OutputStream

- struct Com::Objsys::Asn1::Runtime::Asn1PrintableString
- struct Com::Objsys::Asn1::Runtime::Asn1PrintableStringCharset
- struct Com::Objsys::Asn1::Runtime::Asn1Real
- struct Com::Objsys::Asn1::Runtime::Asn1Real10
- struct Com::Objsys::Asn1::Runtime::Asn1RelativeOID
- struct Com::Objsys::Asn1::Runtime::Asn1SeqOrderException
- struct Com::Objsys::Asn1::Runtime::Asn1Status
- struct Com::Objsys::Asn1::Runtime::Asn1T61String
- struct Com::Objsys::Asn1::Runtime::Asn1Tag
- struct Com::Objsys::Asn1::Runtime::Asn1Time

  <summary>
- struct Com::Objsys::Asn1::Runtime::Asn1TraceHandler
- struct Com::Objsys::Asn1::Runtime::Asn1Type
- struct Com::Objsys::Asn1::Runtime::Asn1TypeIF
- struct Com::Objsys::Asn1::Runtime::Asn1UniversalString
- struct Com::Objsys::Asn1::Runtime::Asn1UTCTime
- struct Com::Objsys::Asn1::Runtime::Asn1UTF8String
- struct Com::Objsys::Asn1::Runtime::Asn1Util
- struct Com::Objsys::Asn1::Runtime::Asn1Value
- struct Com::Objsys::Asn1::Runtime::Asn1ValueParseException
- struct Com::Objsys::Asn1::Runtime::Asn1VarWidthCharString
- struct Com::Objsys::Asn1::Runtime::Asn1VideotexString
- struct Com::Objsys::Asn1::Runtime::Asn1VisibleString
- struct Com::Objsys::Asn1::Runtime::Asn1VisibleStringCharset
- struct Com::Objsys::Asn1::Runtime::BigInteger
- struct Com::Objsys::Asn1::Runtime::BooleanHolder
- struct Com::Objsys::Asn1::Runtime::Diag
- struct Com::Objsys::Asn1::Runtime::IntHolder
- struct Com::Objsys::Asn1::Runtime::StringBufferExt
- struct Com::Objsys::Asn1::Runtime::Tokenizer

Detailed Description

The ASN.1 C# runtime library uses the `Com.Objsys.Asn1.Runtime` namespace. This namespace contains the implementation of the following rules:

- BER (As per ITU-T X.690 standard)
- CER (As per ITU-T X.690 standard)
- DER (As per ITU-T X.690 standard)
- MDER (As per ISO/IEEE 11073-2101:2004 standard)
- OER (As per ITU-T X.696 standard)
- PER (As per ITU-T X.691 standard)
- XER (As per ITU-T X.693 standard)
- XML (As per asn2xsd converter)

Definition at line 26 of file Asn18BitCharString.cs

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

- Asn18BitCharString.cs

System

System::Collections::Generic

System::IO

System::Reflection

System::Runtime::CompilerServices

System::Runtime::InteropServices

System::Text

Chapter 2. ASN1C C# Runtime Library

The ASN.1 C# runtime library uses the `Com.Objsys.Asn1.Runtime` namespace. This namespace contains the implementation of the following rules:

- BER (As per ITU-T X.690 standard)
- CER (As per ITU-T X.690 standard)
- DER (As per ITU-T X.690 standard)
- MDER (As per ISO/IEEE 11073-2101:2004 standard)
- OER (As per ITU-T X.696 standard)
- PER (As per ITU-T X.691 standard)
- XER (As per ITU-T X.693 standard)
- XML (As per asn2xsd converter)

Chapter 3. Class Documentation

Com::Objsys::Asn1::Runtime::Asn18BitCharString class Reference

Public Attributes

- const int BITSPERCHAR_A
- const int BITSPERCHAR_U

- Asn18BitCharString (short typeCode)
- Asn18BitCharString (System.String data, short typeCode)

- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- void Decode (Asn1OerDecodeBuffer buffer, int length)

  <summmary> Decode the value in accordance with OER.
- override void Encode (Asn1PerEncodeBuffer buffer)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet)
- virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet, long lower, long upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- virtual void Encode (Asn1OerEncodeBuffer buffer, bool withLength)

- static void Skip (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)

Detailed Description

This is an abstract base class for holding the ASN.1 8-bit character string types (IA5String, VisibleString, PrintableString, etc.).

Definition at line 36 of file Asn18BitCharString.cs

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

- Asn18BitCharString.cs

Member Data Documentation

const int BITSPERCHAR_A

The `BITSPERCHAR_A` constant specifies the number of bits per character for PER (aligned).

Definition at line 41 of file Asn18BitCharString.cs

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

- Asn18BitCharString.cs

const int BITSPERCHAR_U

The `BITSPERCHAR_U` constant specifies the number of bits per character for PER (unaligned).

Definition at line 46 of file Asn18BitCharString.cs

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

- Asn18BitCharString.cs

Asn18BitCharString (short typeCode)

The default constructor creates an empty string object.

Table 3.1. Parameters

typeCode	Universal ID code for ASN.1 character string

Asn18BitCharString (System.String data, short typeCode)

This version of the constructor can be used to set the string `mValue` member variable to the given string.

Table 3.2. Parameters

data	Character string
typeCode	Universal ID code for ASN.1 character string

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.3. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.4. Parameters

buffer	Decode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)

virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the Decode method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present and an optional permitted alphabet constraint.

The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.5. Parameters

buffer	Decode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Decode (Asn1OerDecodeBuffer buffer)

Decode the value in accordance with OER.

This class's implementation decodes the string with a length determinant. If a subclass should be decoded without a length determinant, it should override this method to invoke Decode(buffer, length). Subclasses may override this to add constraint checks after invoking this method to decode the string.

void Decode (Asn1OerDecodeBuffer buffer, int length)

This class's implementation decodes a string of the given length. This method is not virtual as I don't see any reason for overriding it. </summmary>

Table 3.6. Parameters

length	Length of string to decode

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.7. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.8. Parameters

buffer	Encode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)

virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present and an optional permitted alphabet constraint.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.9. Parameters

buffer	Encode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER) directly into the stream. This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.10. Parameters

outs	PER Encode message stream object

Table 3.11. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER) directly into the stream. This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.12. Parameters

outs	PER Encode message stream object
charSet	Object representing permitted alphabet constraint character set (optional)

Table 3.13. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 character string value directly into the stream, in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present and an optional permitted alphabet constraint.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.14. Parameters

outs	PER Encode message stream object
charSet	Object representing permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

Table 3.15. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1OerEncodeBuffer buffer)

Encode the value in accordance with OER.

This class's implementation invokes Encode(buffer, true) to encode the string with a length determinant. If a subclass should be encoded without a length determinant, it should override this to invoke Encode(buffer, false).

virtual void Encode (Asn1OerEncodeBuffer buffer, bool withLength)

Encode the string, with or without a length determinant.

Subclasses may override this (e.g. to add constraint checks) and invoke this method to perform the encoding.

Table 3.16. Parameters

withLength	true if a length determinant should be encoded.

static void Skip (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `value` member variable in the **Asn1CharString** base class.

Table 3.17. Parameters

buffer	Decode message buffer object

charSet	Object representing permitted alphabet constraint character set (optional)

Asn1AbstractTime class Reference

Com::Objsys::Asn1::Runtime::Asn1BigInteger class Reference

- static new readonly Asn1Tag _TAG
- static readonly BigInteger ZERO
- static readonly BigInteger MINUSONE
- static readonly BigInteger SIXTY_FOUR_K
- static readonly BigInteger TWOFIVESIX
- const int MAX_BIG_INT_LEN

Public Attributes

- BigInteger mValue
- Asn1BigInteger ( )
- Asn1BigInteger (BigInteger value)
- Asn1BigInteger (System.String value)
- Asn1BigInteger (System.String value, int radix)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- void Decode (Asn1PerDecodeBuffer buffer, BigInteger lower, BigInteger upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- void DecodeSigned (Asn1OerDecodeBuffer buffer)
- void DecodeSigned (Asn1OerDecodeBuffer buffer, int octets)
- void DecodeUnsigned (Asn1OerDecodeBuffer buffer)
- void DecodeUnsigned (Asn1OerDecodeBuffer buffer, int octets)
- BigInteger DecodeValue (Asn1DecodeBuffer buffer, int length)
- virtual void DecodeXER (System.String buffer, System.String attrs)

- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- void Encode (Asn1PerEncodeBuffer buffer, BigInteger lower, BigInteger upper)

  This method encodes an ASN.1 integer value using Packed Encoding Rules (PER).
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- override void EncodeAttribute (Asn1XmlEncoder buffer, String attrName)
- void EncodeSigned (Asn1OerEncodeBuffer buffer, int octets)
- void EncodeSigned (Asn1OerEncodeBuffer buffer)
- void EncodeUnsigned (Asn1OerEncodeBuffer buffer)
- void EncodeUnsigned (Asn1OerEncodeBuffer buffer, int octets)
- virtual bool Equals (long value)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- override System.String ToString ( )

- static int EncodeValue (Asn1EncodeBuffer buffer, BigInteger ivalue, bool doCopy)

Detailed Description

This class represents an ASN.1 INTEGER built-in type. In this case, the values can be greater than 64 bits in size. This class is used in generated source code if the <bigInteger> qualifier is specified in a compiler configuration file.

Definition at line 38 of file Asn1BigInteger.cs

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

- Asn1BigInteger.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 2).

Definition at line 43 of file Asn1BigInteger.cs

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

- Asn1BigInteger.cs

Member Data Documentation

BigInteger mValue

The magnitude of this Asn1BigInteger, in *big-endian* order: the zeroth element of this array is the most-significant int of the magnitude. The magnitude must be "minimal" in that the most-significant int (`mValue[0]`) must be non-zero. This is necessary to ensure that there is exactly one representation for each Asn1BigInteger value. Note that this implies that the Asn1BigInteger zero has a zero-length mValue array.

Definition at line 64 of file Asn1BigInteger.cs

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

- Asn1BigInteger.cs

Asn1BigInteger ()

The default constructor sets the big integer value object reference to null.

Asn1BigInteger (BigInteger value)

This constructor assigns a big integer object.

Table 3.18. Parameters

value	BigInteger value

Asn1BigInteger (System.String value)

This constructor creates a new big integer object and sets it to the string value passed in. String value may contain the prefix that describes the radix: 0x - hexadecimal, 0o - octal, 0b - binary. The string value without prefix assumes decimal value. The optional sign '-' may be specified at the beginning of the string to specify the negative value.

Table 3.19. Parameters

value	String value

Asn1BigInteger (System.String value, int radix)

This constructor creates a new big integer object and sets it to the string value passed in. String value may contain the prefix that describes the radix: 0x - hexadecimal, 0o - octal, 0b - binary. The string value without prefix assumes decimal value. The optional sign '-' may be specified at the beginning of the string to specify the negative value.

Table 3.20. Parameters

value	String value
radix	Can be 16 for hexadecimal, 8 for octal, 2 for binary or 10 for decimal

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 integer value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.21. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is stored in the public `mValue` member variable.

Table 3.22. Parameters

buffer	PER Decode message buffer object

void Decode (Asn1PerDecodeBuffer buffer, BigInteger lower, BigInteger upper)

This method decodes an ASN.1 integer value using Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is stored in the public `mValue` member variable.

Table 3.23. Parameters

buffer	PER Decode message buffer object
lower	The PER-visible lower bound; null if there is not a lower bound.
upper	The PER-visible upper bound; null if there is not an upper bound.

override void Decode (Asn1OerDecodeBuffer buffer)

Decode this value as if unconstrained. Subclasses may override this method to decode according to the constraints on the respective ASN.1 type.

void DecodeSigned (Asn1OerDecodeBuffer buffer)

Decode a variable-size signed integer, encoded with a length, into this object, following OER.

void DecodeSigned (Asn1OerDecodeBuffer buffer, int octets)

Decode a signed integer (2's complement form), of the given length, into this object.

Table 3.24. Parameters

octets	The number of octets in which the value was encoded.

void DecodeUnsigned (Asn1OerDecodeBuffer buffer)

Decode a variable-size unsigned integer, encoded with a length, into this object, following OER.

void DecodeUnsigned (Asn1OerDecodeBuffer buffer, int octets)

Decode an unsigned integer (a binary integer, not 2's complement form), of the given length, into this object.

Table 3.25. Parameters

octets	The number of octets in which the value was encoded.

BigInteger DecodeValue (Asn1DecodeBuffer buffer, int length)

This method decodes the contents of an ASN.1 integer value using either the Basic Encoding Rules (BER) or the Packed Encoding Rules (PER).

Table 3.26. Parameters

buffer	Decode message buffer object
length	Length of encoded contents

Returns: . Decoded integer value

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes an ASN.1 integer value using the XML encoding rules (XER).

Table 3.27. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 integer value using the XML schema encoding rules(asn2xsd).

Table 3.28. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 integer value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.29. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.30. Parameters

buffer	PER Encode message buffer object

Returns: . Length of component or negative status value

void Encode (Asn1PerEncodeBuffer buffer, BigInteger lower, BigInteger upper)

The length and contents components of the message are encoded.

Table 3.31. Parameters

buffer	PER Encode message buffer object
lower	The PER-visible lower bound; null if there is not a lower bound.
upper	The PER-visible upper bound; null if there is not an upper bound.

Returns: . Length of component or negative status value

override void Encode (Asn1OerEncodeBuffer buffer)

Encode this value as if unconstrained. Subclasses may override this method to encode it according to the constraints on the respective ASN.1 type.

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 integer value using the XML encoding rules (XER).

Table 3.32. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 integer value using the XML Encoding as specified in the XML schema standard.

Table 3.33. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace prefix name

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 integer value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, exception thrown by the underlying System.IO.Stream object.

Table 3.34. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.35. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.36. Parameters

outs	PER Output Stream object

Table 3.37. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void EncodeAttribute (Asn1XmlEncoder buffer, String attrName)

This method encodes an ASN.1 integer value using the XML Encoding as specified in the XML schema standard.

Table 3.38. Parameters

buffer	Encode message buffer object
elemName	Element name
attribute	Attribute name

void EncodeSigned (Asn1OerEncodeBuffer buffer, int octets)

Encode integer value as a signed value (2's complement) in the given number of octets.

The value must fit in the given number of octets.

Table 3.39. Parameters

buffer	The buffer.
octets	The number of octets to encode in; 0 < octets <= 8

void EncodeSigned (Asn1OerEncodeBuffer buffer)

Encode this value as a variable-size signed integer, with length, according to OER.

void EncodeUnsigned (Asn1OerEncodeBuffer buffer)

Encode this value as a variable-size unsigned integer, with length, according to OER.

void EncodeUnsigned (Asn1OerEncodeBuffer buffer, int octets)

Encode integer value as an unsigned value (binary integer) in the given number of octets.

The value must be non-negative and fit in the given number of octets.

Table 3.40. Parameters

buffer	The buffer.
octets	The number of octets to encode in; 0 < octets <= 8.

virtual bool Equals (long value)

This method compares this value to the given integer value for equality.

Table 3.41. Parameters

value	The long value to compare with the current Object.

Returns: . `true` if the specified long value is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this value to the given Asn1BigInteger value for equality.

Table 3.42. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1BigInteger.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

override System.String ToString ()

This method will return a string representation of the integer value. The format is the ASN.1 value format for this type..

Returns: . Stringified representation of the value

static int EncodeValue (Asn1EncodeBuffer buffer, BigInteger ivalue, bool doCopy)

This method encodes an ASN.1 integer value's contents in accordance with either the ASN.1 Basic Encoding Rules (BER) or Packed Encoding Rules (PER).

Table 3.43. Parameters

buffer	Encode message buffer object
ivalue	Integer value to encode
doCopy	Encode the copy of the value

Returns: . Length of encoded component

Com::Objsys::Asn1::Runtime::Asn1BitString class Reference

- enum StringFormat {
  HEX,
  BITS,
  ASN1VALUE,
  HEXBIN,
  NAMEDBITS
  }

- static new readonly Asn1Tag _TAG

Private Attributes

- Asn1RunTime rt

Protected Attributes

- Dictionary< int, string > mPositionsToNames

Public Attributes

- StringFormat mStringFormat
- byte [] mValue
- int numbits
- bool trimZeroBits
-
-
- Asn1BitString ( )
- Asn1BitString (int numbits_, byte [] data)
- Asn1BitString (byte [] data)
- Asn1BitString (bool [] bitValues)
- Asn1BitString (System.String value_)
- Asn1BitString (System.Collections.BitArray bitArray)
- void BaseDecode (Asn1PerDecodeBuffer buffer, long lower, long upper)
- virtual void Clear (int bitno)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)
- void Decode (Asn1MderDecodeBuffer buffer, int length)
- override void Decode (Asn1OerDecodeBuffer buffer)

  <summary> This method decodes an ASN.1 BIT STRING that was encoded according to OER.
- void DecodeContent (Asn1OerDecodeBuffer buffer, int numOctets, int unusedBits)

  <summmary> This method decodes an ASN.1 BIT STRING's content that was encoded according to OER, given the total number of octets and the number of unused bits in the last octet.
- virtual void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

- override void Encode (Asn1PerEncodeBuffer buffer)
- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- void Encode (Asn1MderOutputStream outs, int length)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- virtual void Encode (Asn1XerEncoder buffer, System.String elemName, System.String [] namedbits, int [] namedbitindex)
- virtual void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix, System.String [] namedbits, int [] namedbitindex)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)
- void EncodeContent (Asn1OerEncodeBuffer buffer)
- virtual bool Equals (int nbits, byte [] value)

  <summary> This method compares this bit string value to the given value for equality.
- override bool Equals (System.Object value)
- virtual bool Get (int bitno)
- override int GetHashCode ( )
- virtual bool IsNamedBitStr (System.String buffer)
- virtual void Set (int bitno, bool value)
- virtual void Set (int bitno)
- virtual bool [] ToBoolArray ( )
- virtual System.String ToHexString ( )
- virtual System.IO.Stream toInputStream ( )
- override string ToString ( )

- void BaseDecode (Asn1PerDecodeBuffer buffer)
- void BaseEncode (Asn1PerEncodeBuffer buffer, long minEncLen)
- void BaseEncode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- virtual int GetOerEffectiveMin ( )

- static void Skip (Asn1PerDecodeBuffer buffer)
- static void Skip (Asn1PerDecodeBuffer buffer, long lower, long upper)
- void AllocBitArray (int numbits_)
- void ReallocBitArray (int numbits_)
- void SetBits (String bits)
- int Trim (int minLength)

  This method will trim trailing zero bits from this BIT STRING.

- static Asn1BitString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 bit string value.

Definition at line 40 of file Asn1BitString.cs

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

- Asn1BitString.cs

enum StringFormat

Defines possible string formats.

The `HEX` constant describes the string format as hex digit value (e.g. 0123456789ABCDEF).

The `BITS` constant describes the string format as binary digit value (e.g. only 0 and 1 digits).

The `ASN1VALUE` constant describes the string format as hex or binary digit value. The binary string value will end with letter 'B' and hex string value will end with letter 'H' (e.g. '0101'B or '11'H). If the number of bits is less than or equal to 16, than it will be printed as Binary digits, else as Hex digits.

The `NAMEDBITS` constant describes the string format as a list of bit position names (where applicable) followed by 1 or 0 in parentheses.

Enumerator:

HEX

BITS

ASN1VALUE

HEXBIN

NAMED-
BITS

Definition at line 73 of file Asn1BitString.cs

```
{
HEX,
BITS,
ASN1VALUE,
HEXBIN,
NAMEDBITS,
}StringFormat;
```

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 3).

Definition at line 44 of file Asn1BitString.cs

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

- Asn1BitString.cs

Member Data Documentation

StringFormat mStringFormat

The **mStringFormat** variable can be used to set the string format for print() or event handler calls or toString() functions. The possible options are: HEX, HEXBIN, BITS, NAMEDBITS, and ASN1VALUE. HEXBIN is the default format.

Definition at line 81 of file Asn1BitString.cs

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

- Asn1BitString.cs

byte [] mValue

This variable holds the bit string value. These are the bits that are encoded when encode is invoked. It is also where the decoded bit string is stored after a Decode operation.

Definition at line 102 of file Asn1BitString.cs

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

- Asn1BitString.cs

int numbits

This variable contains the number of bits in the bit string value.

Definition at line 87 of file Asn1BitString.cs

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

- Asn1BitString.cs

bool trimZeroBits

This variable describes whether trailing zero bits should be truncated. This is required when encoding a named bit string using CER, DER, PER, or C-OER. By default, no trimming is done.

Definition at line 95 of file Asn1BitString.cs

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

- Asn1BitString.cs

Asn1BitString ()

This constructor creates an empty bit string that can be used in a Decode method call to receive a bit string value.

Asn1BitString (int numbits_, byte[] data)

This constructor initializes a bit string with the given number of bits and data.

Table 3.44. Parameters

numbits_	Number of bits
data	Binary bit string contents

Asn1BitString (byte[] data)

This constructor initializes a bit string with the given bytes, using all 8 bits of each byte.

Table 3.45. Parameters

data	Binary bit string contents

Asn1BitString (bool[] bitValues)

This constructor initializes a bit string from the given boolean array. Each array position corresponds to a bit in the bit string.

Table 3.46. Parameters

bitValues	The boolean array

Asn1BitString (System.String value_)

This constructor parses the given ASN.1 value text (either a binary or hex data string) and assigns the values to the internal bit string.

Examples of valid value formats are as follows:

Binary string: '11010010111001'B

Hex string: '0fa56920014abc'H

Table 3.47. Parameters

value_	The ASN.1 value specification text

Asn1BitString (System.Collections.BitArray bitArray)

This constructor initializes a bit string from the given BitSet object.

Table 3.48. Parameters

bitArray	C# BitArray object

void BaseDecode (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method decodes a sized ASN.1 bit string value using the packed encoding rules (PER).

This method is guaranteed non-reentrant.

Table 3.49. Parameters

buffer	Decode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

virtual void Clear (int bitno)

This method clears the given bit in the bit string.

Table 3.50. Parameters

bitno	The zero-based index of the bit to clear. The bit numbers start at zero and with the MSB of the first byte and progress from left to right.

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 bit string value using the BER or DER encoding rules. The UNIVERSAL tag value and length are decoded if explicit tagging is specified.

Table 3.51. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 bit string value using the packed encoding rules (PER). The string is assumed to not contain a size constraint.

Table 3.52. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method decodes a sized ASN.1 bit string value using the packed encoding rules (PER).

Table 3.53. Parameters

buffer	Decode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

void Decode (Asn1MderDecodeBuffer buffer, int length)

Decode a BIT STRING from the MDER encoding into this object. Exactly the given length of bits will be decoded.

Table 3.54. Parameters

length	This should be 8, 16, or 32, as these are the only lengths MDER supports. However, this is not checked here as it should be able to be checked at code generation time.

override void Decode (Asn1OerDecodeBuffer buffer)

This assumes the BIT STRING was not fixed-size constrained, so that the length and unused bits are in the encoding. Subclasses can override this method to simply call DecodeContent for fixed-size constrained BIT STRINGs.

Table 3.55. Parameters

buffer	Decode message buffer object

void DecodeContent (Asn1OerDecodeBuffer buffer, int numOctets, int unusedBits)

</summmary>

Table 3.56. Parameters

buffer	Decode message buffer object
numOctets	Total number of octets encoding the content, including the final, partial octet (if unusedBits > 0).
unusedBits	# of unused bits in last octet.

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes ASN.1 bit string type using the XML encoding rules (XER).

Table 3.57. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes ASN.1 bit string type using the XML decoding as specified in the XML Schema standard.

Table 3.58. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 bit string value using the BER or DER encoding rules. The UNIVERSAL tag value and length are encoded if explicit tagging is specified.

Table 3.59. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an unconstrained ASN.1 bit string value using the packed encoding rules (PER). The value to be encoded is stored in the 'numbits' and 'mValue' public member variables within this class.

Table 3.60. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This method encodes a size-constrained ASN.1 bit string value using the packed encoding rules (PER). The value to be encoded is stored in the 'numbits' and 'mValue' public member variables within this class.

Table 3.61. Parameters

buffer	Encode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

void Encode (Asn1MderOutputStream outs, int length)

Encode this BIT STRING into the MDER encoding. The length of this BIT STRING must match the given length.

Table 3.62. Parameters

length	This should be 8, 16, or 32, as these are the only lengths MDER supports. However, this is not checked here as it should be able to be checked at code generation time. We only check here that the actual and given lengths match.

override void Encode (Asn1OerEncodeBuffer buffer)

This method encodes this ASN.1 BIT STRING value, according to OER. This encodes the length determinant and # of unused bits, assuming that the BIT STRING is not fixed-size constrained. Subclasses may override this to simply call EncodeContent for fixed-size constrained strings.

Table 3.63. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes ASN.1 bit string type using the XML encoding rules (XER).

Table 3.64. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string

virtual void Encode (Asn1XerEncoder buffer, System.String elemName, System.String[] namedbits, int[] namedbitindex)

This method encodes ASN.1 bit string type using the XML Encoding as specified in the itu-t XER standard.

Table 3.65. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string
namedbits	Array of named bits
namedbitindex	Arrat of named bits index values

virtual void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix, System.String[] namedbits, int[] namedbitindex)

This method encodes ASN.1 bit string type using the XML Encoding as specified in the XML Schema standard.

Table 3.66. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string
nsPrefix	XML element namespace name

namedbits	Array of named bits
namedbitindex	Arrat of named bits index values

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes ASN.1 bit string type using the XML Encoding as specified in the XML Schema standard.

Table 3.67. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string
nsPrefix	XML element namespace name

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 bit string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, exception thrown by the underlying System.IO.Stream object.

Table 3.68. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.69. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an unconstrained ASN.1 bit string value using the packed encoding rules (PER)into the stream. The value to be encoded is stored in the 'numbits' and 'mValue' public member variables within this class.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.70. Parameters

outs	PER Output Stream object

Table 3.71. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)

This method encodes a size-constrained ASN.1 bit string value using the packed encoding rules (PER) into the stream. The value to be encoded is stored in the 'numbits' and 'mValue' public member variables within this class.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.72. Parameters

outs	PER Output Stream object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

Table 3.73. Exceptions

Asn1Exception	Thrown, if operation is failed.

void EncodeContent (Asn1OerEncodeBuffer buffer)

This method encodes the content of an ASN.1 bit string value, according to OER. (The length determinant and # of unused bits is not encoded.)

Table 3.74. Parameters

buffer	Encode message buffer object

virtual bool Equals (int nbits, byte[] value)

This method assumes all unused bits in the last byte are set to zero.

Table 3.75. Parameters

nbits	The number of bits to compare from the byte array.
value	The byte array to compare with the current Object.

Returns: . `true` if the specified bit array is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this bit string value to the given value for equality. This method assumes all unused bits in the last byte are set to zero.

Table 3.76. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1BitString.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

virtual bool Get (int bitno)

Gets the value of the bit at a specific position in the bit array.

Table 3.77. Parameters

bitno	The zero-based index of the bit to get. The bit numbers start at zero and with the MSB of the first byte and progress from left to right.

Returns: . `true` if bit is set; otherwise `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

virtual bool IsNamedBitStr (System.String buffer)

This method determines is the input character string represented as named bit string or as bits sequence. It is used for XML decoding.

Table 3.78. Parameters

buffer	Bit string as string to be tested.

Returns: . true, if bit string is represented as sequnece of identifiers.

virtual void Set (int bitno, bool value)

This method sets the given bit number in the bit string with given value. It will expand the existing bit array if it needs to, , setting any added bits, except bitno, to 0.

Table 3.79. Parameters

bitno	The zero-based index of the bit to set. The bit numbers start at zero and with the MSB of the first byte and progress from left to right.
value	The Boolean value to assign to the bit.

virtual void Set (int bitno)

This method will set the given bit number to one (1). It will expand the existing bit array if it needs to.

Table 3.80. Parameters

bitno	The zero-based index of the bit to set. The bit numbers start at zero and with the MSB of the first byte and progress from left to right.

virtual bool [] ToBoolArray ()

This method converts the bit string stored in this object to a boolean array.

Returns: . Boolean array value

virtual System.String ToHexString ()

This method will return a hex string representation of the bit string value. The output format is a string of hex bytes with no extra delimiting characters (ex. 0D56EF).

Returns: . Stringified representation of the value

virtual System.IO.Stream toInputStream ()

This method will return a byte array input stream representation of the bit string value.

Returns: . Reference to System.IO.MemoryStream object

override string ToString ()

This method will return a string representation of the bit string value. The output format is a string of hex digits/binary digits according to the value set for **mStringFormat** variable.

Returns: . String representation of the value

void BaseDecode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 bit string value using the packed encoding rules (PER). The string is assumed to not contain a size constraint.

This method is guaranteed non-reentrant.

Table 3.81. Parameters

buffer	Decode message buffer object

void BaseEncode (Asn1PerEncodeBuffer buffer, long minEncLen)

This method encodes the bit string as an unconstrained ASN.1 bit string value using the packed encoding rules (PER). The value to be encoded is stored in the 'numbits' and 'mValue' public member variables within this class. It does not trim trailing zeros. It will add trailing zeros if necessary, in accordance with minEncLen.

This method is guaranteed non-reentrant.

Table 3.82. Parameters

buffer	Encode message buffer object
minEncLen	Minimum number of bits to encode. Trailing zero bits are added, if necessary. Pass 0 if no trailing zero bits should ever be added.

void BaseEncode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This method encodes a size-constrained ASN.1 bit string value using the packed encoding rules (PER). The value to be encoded is stored in the 'numbits' and 'mValue' public member variables within this class.

This method is guaranteed non-reentrant.

Table 3.83. Parameters

buffer	Encode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

virtual int GetOerEffectiveMin ()

Return the lower bound for the OER effective size constraint. When trimZeroBits is set, this controls the trimming done for OER encoding.

Subclasses must override this to return the correct value if the BIT STRING has named bits and the effective size constraint has a lower bound other than zero.

static void Skip (Asn1PerDecodeBuffer buffer)

This method skips an ASN.1 BIT STRING value using the packed encoding rules (PER). The string is assumed to not be size constrained.

Table 3.84. Parameters

buffer	Decode message buffer object

static void Skip (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method skips a sized ASN.1 bit string value using the packed encoding rules (PER).

Table 3.85. Parameters

buffer	Decode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

void SetBits (String bits)

Assign this Asn1BitString's bytes based on the given bit string.

Table 3.86. Parameters

bits	An xmlbstring ('0', '1', and whitespace chars); the string representation of the bit string.

int Trim (int minLength)

To trim all trailing zero bits, pass 0 for minLength;

Table 3.87. Parameters

bitstr	An ASN.1 BIT STRING object.
minLength	The minimum length to trim down to; do not trim the string to be shorter than this.

Returns: . Adjusted bit count.

Com::Objsys::Asn1::Runtime::Asn1BMPString class Reference

- static new readonly Asn1Tag _TAG
- static readonly Asn1CharSet CHARSET

Public Attributes

- const int BITSPERCHAR

- Asn1BMPString ( )
- Asn1BMPString (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- void Decode (Asn1OerDecodeBuffer buffer, int length)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- virtual void Encode (Asn1OerEncodeBuffer buffer, bool withLength)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet)
- virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet, long lower, long upper)
- void ReadSegment (Asn1BerDecodeBuffer buffer, System.Text.StringBuilder sb, int len)
- static Asn1BMPString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 BMP string value.

Definition at line 36 of file Asn1BMPString.cs

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

- Asn1BMPString.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 30).

Definition at line 47 of file Asn1BMPString.cs

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

- Asn1BMPString.cs

Member Data Documentation

const int BITSPERCHAR

The `BITSPERCHAR` constant specifies the number of bits per character for PER (aligned or unaligned).

Definition at line 42 of file Asn1BMPString.cs

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

- Asn1BMPString.cs

Asn1BMPString ()

The default constructor creates an empty string object.

Asn1BMPString (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string value.

Table 3.88. Parameters

data	string representation of BMPString

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 BMP string value including the UNIVERSAL tag value and length if explicit tagging is specified. This string type uses 16-bit characters.

Table 3.89. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.90. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but allows a permitted alphabet character set to be specified. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.91. Parameters

buffer	Decode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)

virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the Decode method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes both a size constraint and permitted alphabet constraint are present.

The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.92. Parameters

buffer	Decode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Decode (Asn1OerDecodeBuffer buffer)

Decode the value in accordance with OER.

This class's implementation decodes the string with a length determinant. If a subclass should be decoded without a length determinant, it should override this method to invoke decode(buffer, length). Subclasses may override this to add constraint checks after invoking this method to decode the string.

void Decode (Asn1OerDecodeBuffer buffer, int length)

Decode the value in accordance with OER.

This class's implementation decodes a string of the given length. This method is non-virtual as I don't see any reason for overriding it.

Table 3.93. Parameters

length	Length of string to decode, in characters.

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.94. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.95. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.96. Parameters

buffer	Encode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)

virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes both a size constraint and permitted alphabet constraint are present.

The value to encode is stored in the public `mValue` member variable.

Table 3.97. Parameters

buffer	Encode message buffer object
charSet	Object representing the permitted alphabet constraint character set
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Encode (Asn1OerEncodeBuffer buffer)

Encode the value in accordance with OER.

This class's implementation invokes encode(buffer, true) to encode the string with a length determinant. If a subclass should be encoded without a length determinant, it should override this to invoke encode(buffer, false).

virtual void Encode (Asn1OerEncodeBuffer buffer, bool withLength)

Encode the string, with or without a length determinant.

Subclasses may override this (e.g. to add constraint checks) and invoke this method to perform the encoding.

Table 3.98. Parameters

withLength	true if a length determinant should be encoded.

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 BMP string including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, exception thrown by the underlying System.IO.Stream object.

Table 3.99. Parameters

outs	BER Output Stream object

explicitTagging	Flag indicating explicit tagging should be done

Table 3.100. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 BMP string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.101. Parameters

outs	PER Output Stream object

Table 3.102. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet)

This method encodes an ASN.1 BMP string value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.103. Parameters

outs	PER Output Stream object
charSet	Object representing permitted alphabet constraint character set (optional)

Table 3.104. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 BMP string value in accordance with the packed encoding rules (PER). This version of the method assumes both a size constraint and permitted alphabet constraint are present.

The value to encode is stored in the public `mValue` member variable.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.105. Parameters

outs	PER Output Stream object
charSet	Object representing the permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

Table 3.106. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1BMPStringCharset class Reference

•

- Asn1BMPStringCharset ( )
- override int GetCharAtIndex (int index)
- override int GetCharIndex (int charValue)
- override bool validate (String s)

Com::Objsys::Asn1::Runtime::Asn1Boolean class Reference

- static new readonly Asn1Tag _TAG
- static readonly Asn1Boolean FALSE_VALUE
- static readonly Asn1Boolean TRUE_VALUE

Public Attributes

- bool mValue
- static byte trueEncodedByte
- Asn1Boolean ( )
- Asn1Boolean (bool value_)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override void Decode (Asn1OerDecodeBuffer buffer)
- virtual void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- virtual void Encode (Asn1XerEncodeBuffer buffer)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix, bool asText)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)
- virtual bool Equals (bool value)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- override System.String ToString ( )
- static void setTrueEncodedByte (byte b)
- static Asn1Boolean ( )

Detailed Description

This class represents the ASN.1 BOOLEAN built-in type.

Definition at line 34 of file Asn1Boolean.cs

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

- Asn1Boolean.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 1).

Definition at line 39 of file Asn1Boolean.cs

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

- Asn1Boolean.cs

readonly Asn1Boolean FALSE_VALUE

The FALSE_VALUE constant represents a boolean FALSE value.

Definition at line 45 of file Asn1Boolean.cs

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

- Asn1Boolean.cs

readonly Asn1Boolean TRUE_VALUE

The TRUE_VALUE constant represents a boolean TRUE value.

Definition at line 42 of file Asn1Boolean.cs

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

- Asn1Boolean.cs

Member Data Documentation

bool mValue

This public member variable is where the boolean value is stored. This is the value that is encoded when one of the encode methods is called. It is also where the decoded result is stored when a Decode method is called.

Definition at line 53 of file Asn1Boolean.cs

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

- Asn1Boolean.cs

byte trueEncodedByte

This field contains the byte value that will be encoded to represent BER's TRUE value. This field may not be set to 0x00.

Definition at line 112 of file Asn1Boolean.cs

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

- Asn1Boolean.cs

Asn1Boolean ()

The default constructor sets the boolean value to false.

Asn1Boolean (bool value_)

This constructor creates a boolean object from a boolean value.

Table 3.107. Parameters

value_	Boolean value

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 boolean value including the UNIVERSAL. tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER) or the Distinguished Encoding Rules (DER).

The decoded result is stored in the public member mValue. in this object.

Table 3.108. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

Returns: . Decoded boolean value

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 boolean value using. the Packed Encoding Rules (PER).

The decoded result is stored in the public member mValue. in this object.

Table 3.109. Parameters

buffer	PER Decode message buffer object

override void Decode (Asn1OerDecodeBuffer buffer)

Decode BOOLEAN value according to OER.

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes an ASN.1 boolean value using the XML encoding rules (XER).

Table 3.110. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 boolean value using the XML Schema encoding rules(asn2xsd).

Table 3.111. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 boolean value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER) or the Distinguished Encoding Rules (DER).

Table 3.112. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length (in octets) of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 boolean value using the Packed Encoding Rules (PER).

Table 3.113. Parameters

buffer	PER Encode message buffer object

Returns: . Length of component or negative status value

override void Encode (Asn1OerEncodeBuffer buffer)

Encode BOOLEAN value according to OER.

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 boolean value using the XML encoding rules (XER).

Table 3.114. Parameters

buffer	Encode message buffer object
elemName	Element name

virtual void Encode (Asn1XerEncodeBuffer buffer)

This method encodes an ASN.1 boolean value using the XML encoding rules (XER). This method does not add start and end tags (<tag> and </tag>), only value is encoded (<true/> or <false/>).

Table 3.115. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 boolean value according to the Obj-Sys XML encoding rules. It encodes the value as XML text.

Table 3.116. Parameters

buffer	Encode message buffer object
elemName	Element name for optional surrounding element.
nsPrefix	Element namespace prefix value

void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix, bool asText)

This method encodes an ASN.1 boolean value. It is for use with extended-XER.

Table 3.117. Parameters

buffer	Encode message buffer object
elemName	Element name for optional surrounding element.
nsPrefix	XML element name space prefix
asText	If true, encode as text. Otherwise, encode as an empty element.

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 boolean value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, exception thrown by the underlying System.IO.Stream object.

Table 3.118. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.119. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 boolean value using the Packed Encoding Rules (PER).

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.120. Parameters

outs	PER Encode message stream object

Table 3.121. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)

This method encodes an ASN.1 boolean value using the XML Encoding as specified in the W3C XML schema standard(asn2xsd).

Table 3.122. Parameters

buffer	Encode message buffer object
attrName	Element name

virtual bool Equals (bool value)

This method compares this boolean value to the given value for equality.

Table 3.123. Parameters

value	The bool value to compare with the current Object.

Returns: . `true` if the specified bool value is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this boolean value to the given value for equality.

Table 3.124. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1Boolean.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

override System.String ToString ()

This method will return a string representation of the boolean value. The format is the ASN.1 value format for this type.

Returns: . Stringified representation of the value

static void setTrueEncodedByte (byte b)

This method sets the byte value that represents the boolean value TRUE. If a zero byte (0x00) is passed, the value is transparently set to 0xFF, the valid representation for BER, CER, and DER.

Table 3.125. Parameters

b	The byte value to set.

Com::Objsys::Asn1::Runtime::Asn1CharRange class Reference

- int mLower
- int mUpper
-
- Asn1CharRange (int lower, int upper)
- override int GetCharAtIndex (int index)
- override int GetCharIndex (int charValue)
- override bool validate (String s)

Detailed Description

This class is used to represent a permitted alphabet that is specified. as a large, continuous range of characters. An example of this can be found in the following extract from T.124:

simpleTextFirstCharacter UniversalString ::= {0, 0, 0, 0}.

simpleTextLastCharacter UniversalString ::= {0, 0, 0, 255}.

SimpleTextString ::= BMPString (SIZE (0..255)). (FROM (simpleTextFirstCharacter..simpleTextLastCharacter))

This class is mainly for internal use by the compiler when generating. methods that encode/Decode PER charater string components containing permitted alphabet constraints.

Definition at line 46 of file Asn1CharRange.cs

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

- Asn1CharRange.cs

int mLower

This variable represents the lower value of the range.

Definition at line 48 of file Asn1CharRange.cs

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

- Asn1CharRange.cs

int mUpper

This variable represents the upper value of the range.

Definition at line 51 of file Asn1CharRange.cs

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

- Asn1CharRange.cs

Asn1CharRange (int lower, int upper)

This constructor sets the range values.

Table 3.126. Parameters

lower	Range lower value
upper	Range upper value

override int GetCharAtIndex (int index)

This method will fetch the character from the permitted alphabet at the given index.

Table 3.127. Parameters

index	Index of character within the character set

Returns: . Character at given index

Table 3.128. Exceptions

Asn1ConsVioException	Index not within define range

override int GetCharIndex (int charValue)

This method will determine the index of the given character within the permitted alphabet character set.

Table 3.129. Parameters

charValue	Character value to search for

Returns: . Index of character

Table 3.130. Exceptions

Asn1ConsVioException	Character not found in set

override bool validate (String s)

This method will validate a character string by comparing its contents to the character range. Each character in the string is checked against the range. If it exceeds the upper or lower limit of the range, false is returned. Otherwise true is returned.

Table 3.131. Parameters

s	The string to be validated.

Returns: . False if the string contains invalid characters; true otherwise.

Com::Objsys::Asn1::Runtime::Asn1CharSet class Reference

- int mABitsPerChar
- int mUBitsPerChar
-
- Asn1CharSet (int nchars)
- abstract int GetCharAtIndex (int index)
- abstract int GetCharIndex (int charValue)
- virtual int GetNumBitsPerChar (bool aligned)
- abstract bool validate (String s)

Detailed Description

This is the base class for representing character sets that are defined in ASN.1 permitted alphabet constraints.

Definition at line 34 of file Asn1CharSet.cs

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

- Asn1CharSet.cs

int mABitsPerChar

This variable holds number of bits-per-character (PER aligned).

Definition at line 36 of file Asn1CharSet.cs

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

- Asn1CharSet.cs

int mUBitsPerChar

This variable holds number of bits-per-character (PER unaligned).

Definition at line 39 of file Asn1CharSet.cs

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

- Asn1CharSet.cs

Asn1CharSet (int nchars)

This constructor sets the number of bits-per-character values based on the given number of characters in the character set.

Table 3.132. Parameters

nchars	Number of characters in the character set

abstract int GetCharAtIndex (int index)

This method will fetch the character from the permitted alphabet at the given index.

Table 3.133. Parameters

index	Index of character within the character set

Returns: . Character at given index

Table 3.134. Exceptions

Asn1ConsVioException	Index not within define range

abstract int GetCharIndex (int charValue)

This method will determine the index of the given character within the permitted alphabet character set.

Table 3.135. Parameters

charValue	Character value to search for

Returns: . Index of character

Table 3.136. Exceptions

Asn1ConsVioException	Character not found in set

virtual int GetNumBitsPerChar (bool aligned)

This method will return the number of bits-per-character.

Table 3.137. Parameters

aligned	Boolean value indicating whether number of aligned (true) or unaligned (false) characters should be returned.

Returns: . Number of bits-per-character

abstract bool validate (String s)

This method will validate a character string by comparing its contents to the character set. If a character string contains characters that are not in the character set, this method will return false. Otherwise it returns true.

Table 3.138. Parameters

s	The string to be validated.

Returns: . False if the string contains invalid characters; true otherwise.

Com::Objsys::Asn1::Runtime::Asn1CharString class Reference

Public Attributes

- System.String mValue

- System.Text.StringBuilder mStringBuffer

Private Attributes

- short mTypeCode
- Asn1RunTime rt

-

- Asn1CharString (short typeCode)
- Asn1CharString (System.String data, short typeCode)
- virtual void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength, Asn1Tag tag)
- virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)
- virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)
- virtual int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging, Asn1Tag tag)
- virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)
- virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)

- void DecodeByteToChar (Asn1OerDecodeBuffer buffer, int length)
- virtual void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- bool Equals (System.String value)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- override System.String ToString ( )
- bool validate (Asn1CharSet charSet)

- static void DecodeInternal (Asn1PerDecodeBuffer buffer, System.Text.StringBuilder stringBuffer, int abpc, int ubpc, Asn1CharSet charSet)
- static int Encode (Asn1BerEncodeBuffer buffer, String val, bool explicitTagging, Asn1Tag tag)

- static void ByteAlign (Asn1PerMessageBuffer buffer, int nbitsPerChar, long lower, long upper)

Detailed Description

This is a container class for holding the components of an ASN.1 character string value. Subclasses are defined for all of the different string types.

Definition at line 38 of file Asn1CharString.cs

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

- Asn1CharString.cs

Member Data Documentation

System.String mValue

The `mValue` public member variable is used to hold the string value to be encoded or the results of a Decode operation.

Definition at line 43 of file Asn1CharString.cs

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

- Asn1CharString.cs

System.Text.StringBuilder mStringBuffer

The `mStringBuffer` member variable is used to do internal operations on a string being encoded or decoded. Users should create it before using if it is null.

Definition at line 50 of file Asn1CharString.cs

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

- Asn1CharString.cs

Member Data Documentation

short mTypeCode

The `mTypeCode` member variable holds the type code (universal ID value) for a given character string type.

Definition at line 56 of file Asn1CharString.cs

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

- Asn1CharString.cs

Asn1CharString (short typeCode)

This constructor creates an empty string that can be used in a Decode method call to receive a string value.

Table 3.139. Parameters

typeCode	Universal ID code for ASN.1 character string

Asn1CharString (System.String data, short typeCode)

This constructor initializes a character string from the given string data.

Table 3.140. Parameters

data	Character string
typeCode	Universal ID code for ASN.1 character string

virtual void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength, Asn1Tag tag)

This method decodes an ASN.1 character string value including the UNIVERSAL tag value and length if explicit tagging is specified. It is a protected method that can only be accessed by objects subclassed from this type.

Table 3.141. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit
tag	Universal tag to apply

virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes that a permitted alphabet constraint has been specified that would reduce the the number of bits-per-character from the default character set. It also assumes a general length determinant is present (i.e. there is not size constraint). The decoded result is stored in the public `mValue` member variable.

Table 3.142. Parameters

buffer	Decode message buffer object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)

virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the Decode method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The decoded result is stored in the public `mValue` member variable.

Table 3.143. Parameters

buffer	Decode message buffer object
abpc	Number of bits per character (aligned)

ubpc	Number of bits per character (unaligned)
charSet	Object representing permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

virtual int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging, Asn1Tag tag)

This method encodes ASN.1 8-bit character string types including IA5String, PrintableString, NumericString, etc. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified (the universal identifier must be provided by the caller).

Table 3.144. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done
tag	Universal tag to apply

Returns: . Length in octets of encoded component

virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable.

Table 3.145. Parameters

buffer	Encode message buffer object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)

virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The value to encode is stored in the public `mValue` member variable.

Table 3.146. Parameters

buffer	Encode message buffer object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

void DecodeByteToChar (Asn1OerDecodeBuffer buffer, int length)

This method decodes a given number of bytes and converts each byte to a char having the same value.

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes ASN.1 8-bit character string types including IA5String, PrintableString, NumericString, etc. using the XML encoding rules (XER).

Table 3.147. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes ASN.1 8-bit character string types including IA5String, PrintableString, NumericString, etc. using the XML Schema encoding rules(asn2xsd).

Table 3.148. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes ASN.1 8-bit character string types including IA5String, PrintableString, NumericString, etc. using the XML encoding rules (XER).

Table 3.149. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes ASN.1 8-bit character string types including IA5String, PrintableString, NumericString, etc. using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.150. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string
nsPrefix	XML element namespace value

bool Equals (System.String value)

This method compares this character string value to the given value for equality.

Table 3.151. Parameters

value	The String value to compare with the current Object.

Returns: . `true` if the specified string is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this character string value to the given value for equality.

Table 3.152. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1CharString.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

override System.String ToString ()

This method will return a string representation of the value. The format is the ASN.1 value format for this type..

Returns: . Stringified representation of the value

bool validate (Asn1CharSet charSet)

This method will attempt to validate a string against its internal character set.

Returns: . True or False.

static void DecodeInternal (Asn1PerDecodeBuffer buffer, System.Text.StringBuilder stringBuffer, int abpc, int ubpc, Asn1CharSet charSet)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes that a permitted alphabet constraint has been specified that would reduce the the number of bits-per-character from the default character set. It also assumes a general length determinant is present (i.e. there is not size constraint). The decoded result is stored in the given StringBuilder, which may be null.

Table 3.153. Parameters

buffer	Decode message buffer object
stringBuffer	If not null, receives the decoded result.
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)

static int Encode (Asn1BerEncodeBuffer buffer, String val, bool explicitTagging, Asn1Tag tag)

This method encodes ASN.1 8-bit character string types including IA5String, PrintableString, NumericString, etc. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified (the universal identifier must be provided by the caller).

Table 3.154. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done
tag	Universal tag to apply

Returns: . Length in octets of encoded component

Com::Objsys::Asn1::Runtime::Asn1Choice class Reference

- int choiceID
- Asn1Type element
-
-
- Asn1Choice ( )
- override bool Equals (System.Object value)
- virtual Asn1Type GetElement ( )
- override int GetHashCode ( )
- virtual void SetElement (int choiceID, Asn1Type element)

Detailed Description

This class represents the ASN.1 CHOICE built-in type.

Definition at line 34 of file Asn1Choice.cs

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

- Asn1Choice.cs

int choiceID

This member variable is where the selected choice option identifier is stored. This selects the choice option to be used. It is populated with one of the generated choice ID constants in a compiler-generated derived class.

Definition at line 41 of file Asn1Choice.cs

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

- Asn1Choice.cs

Asn1Type element

This member variable is where the selected choice option value is stored. It can be accessed via the get and set methods in this class and in compiler-generated derived classes.

Definition at line 49 of file Asn1Choice.cs

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

- Asn1Choice.cs

Asn1Choice ()

The default constructor initializes the choiceID and value.

override bool Equals (System.Object value)

This method compares this type element with the passed type element.

Table 3.155. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1Choice.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

virtual Asn1Type GetElement ()

This method returns the element object.

Returns: . `element` data member.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

virtual void SetElement (int choiceID, Asn1Type element)

This protected method sets the choice ID and value to the given values. Refer the Set_<element>() methods in compiler generated inherited classes.

Table 3.156. Parameters

choiceID	The identifier for element value. The identifier value can be defined by compiler-generated derived class.
element	The element value. The possible value types can be defined by compiler-generated derived class.

Com::Objsys::Asn1::Runtime::Asn1ChoiceExt class Reference

Public Attributes

- short choiceIndex

- Asn1Tag tag

- Asn1ChoiceExt ( )
- Asn1ChoiceExt (byte [] data)
- Asn1ChoiceExt (byte [] data, int encoding)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- override void Encode (Asn1OerEncodeBuffer buffer)

Detailed Description

This is a container class for holding a CHOICE open type extension element. This class is used for an open type extension (i.e. a ... at the end of a constructed type or a ..., ... at some other point in a constructed type).

For BER, data is expected to be a full TLV. For PER, data is the encoding of the actual type; choiceIndex is the PER choice index. For OER, data is the encoding of the actual type; tag is the OER tag.

Definition at line 42 of file Asn1ChoiceExt.cs

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

- Asn1ChoiceExt.cs

Member Data Documentation

short choiceIndex

The choice index value is used with the packed encoding rules (PER) when this object is used to encode/Decode a choice extension.

Definition at line 112 of file Asn1ChoiceExt.cs

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

- Asn1ChoiceExt.cs

Asn1Tag tag

tag is used with OER. When decoding, it should be set to hold the decoded tag. When encoding, it is encoded as the tag for the unknown choice extension. The form (tag.mForm) is irrelevant.

Definition at line 189 of file Asn1ChoiceExt.cs

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

- Asn1ChoiceExt.cs

Asn1ChoiceExt ()

This constructor creates an empty type that can be used in a Decode method call to receive an encoded value. The data encoding is UNKNOWN.

Asn1ChoiceExt (byte[] data)

This constructor initializes an open type from the given byte array. The array is assumed to contain a previously encoded message component. The data encoding is UNKNOWN.

Table 3.157. Parameters

data	Byte array containing a previously encoded message component.

Asn1ChoiceExt (byte[] data, int encoding)

This constructor initializes an open type from the given byte array. The array is assumed to contain a previously encoded message component.

Table 3.158. Parameters

data	Byte array containing a previously encoded value.
encoding	The encoding that describes the meaning of data. Any of the encodings other than JSON.

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an extension field using the Basic Encoding Rules (BER).

Table 3.159. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an open type extension in a CHOICE construct using the packed encoding rules (PER). This method will capture the extension item in an open type object and store it in the `mValue` public member list variable. The public member variable `choiceIndex` will be populated with the decoded choice index value.

Table 3.160. Parameters

buffer	Decode message buffer object

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 open type extension value using the Basic Encoding Rules (BER).

Table 3.161. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating element is explicitly tagged

Returns: . Length of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 open type extension value using the Packed Encoding Rules (PER).

Table 3.162. Parameters

buffer	Encode message buffer object

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes an ASN.1 open type extension value using the Basic Encoding Rules (BER).

Table 3.163. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating element is explicitly tagged

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 open type value using the Packed Encoding Rules (PER).

Table 3.164. Parameters

outs	PER Output Stream object

override void Encode (Asn1OerEncodeBuffer buffer)

Encode this unknown choice extension using the Octet Encoding Rules (OER). This uses this object's tag as the tag, and then encodes the value field as open type content.

Com::Objsys::Asn1::Runtime::Asn1ConsVioException class Reference

- Asn1ConsVioException (System.String varname, long value)
- Asn1ConsVioException (System.String varname, double value)
- Asn1ConsVioException (System.String varname, System.String value)

Detailed Description

This class defines the 'ASN.1 constraint violation' exception that is thrown when an element is parsed that is outside the bounds to a defined constraint..

Definition at line 37 of file Asn1ConsVioException.cs

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

- Asn1ConsVioException.cs

Asn1ConsVioException (System.String varname, long value)

This constructor creates an exception object with a standard message based on the given variable name and value. The form of the message is "Element 'varname' with value 'value' violates defined constraint".

Table 3.165. Parameters

varname	Name of variable that violates constraint
value	Value of variable

Asn1ConsVioException (System.String varname, double value)

This constructor creates an exception object with a standard message based on the given variable name and value. The form of the message is "Element 'varname' with value 'value' violates defined constraint".

Table 3.166. Parameters

varname	Name of variable that violates constraint
value	Value of variable

Asn1ConsVioException (System.String varname, System.String value)

This constructor creates an exception object with a standard message based on the given variable name and value. The form of the message is "Element 'varname' with value 'value' violates defined constraint".

Table 3.167. Parameters

varname	Name of variable that violates constraint
value	Value of variable

Com::Objsys::Asn1::Runtime::Asn1DecodeBuffer class Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1DecodeBuffer::BufferPos
- int mByteCount

Private Attributes

- System.Collections.ArrayList mCaptureBufferList
- System.IO.Stream mInputStream
- bool mLazyOpenTypeDecode
- long mMarkedPosition
- int mSavedByteCount
- int [] oidBuffer
-
-
- Asn1DecodeBuffer (byte [] msgdata)
- Asn1DecodeBuffer (System.IO.Stream istream)
- virtual void Init ( )
- virtual void AddCaptureBuffer (System.IO.MemoryStream buffer)
- virtual void Capture (int nbytes)

- virtual long DecodeIntValue (int length, bool signExtend)
- int [] DecodeOIDContents (int llen)
- int [] DecodeRelOIDContents (int llen)
- override System.IO.Stream GetInputStream ( )
- virtual void HexDump ( )
- virtual void Mark ( )
- virtual void MarkPos (ref BufferPos bufferPos)
- virtual int Read ( )
- virtual void Read (byte [] buffer, int offset, int nbytes)
- virtual void Read (byte [] buffer)
- int Read2Bytes ( )
- int Read4Bytes ( )
- abstract int ReadByte ( )
- virtual void RemoveCaptureBuffer (System.IO.MemoryStream buffer)
- virtual void Reset ( )
- virtual void ResetPos (ref BufferPos bufferPos)
- virtual void SetInputStream (byte [] msgdata, int offset, int length)
- virtual long Skip (long nbytes)
- void SkipOIDContents (int llen)
- int [] SkipRelOIDContents (int llen)

- int [] DecodeOIDContents (int llen, bool retval)
- int [] DecodeRelOIDContents (int llen, bool retval)

Detailed Description

This is the base class to specific Decode buffer classes for the different types of encoding rules (BER, DER and PER).

Definition at line 32 of file Asn1DecodeBuffer.cs

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

- Asn1DecodeBuffer.cs

int mByteCount

This member variable holds the count of bytes currently read from the message being decoded or input stream.

Definition at line 36 of file Asn1DecodeBuffer.cs

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

- Asn1DecodeBuffer.cs

Asn1DecodeBuffer (byte[] msgdata)

This constructor creates a Decode buffer from the given byte array. The array is assumed to contain an encoded ASN.1 message.

Table 3.168. Parameters

msgdata	Byte array containing an encoded ASN.1 message.

Asn1DecodeBuffer (System.IO.Stream istream)

This constructor creates a Decode buffer from the given input stream object. The stream is assumed to contain an encoded ASN.1 message.

Table 3.169. Parameters

istream	Stream from which to read encoded ASN.1 message.

virtual void Init ()

This method initializes the input stream for decoding.

virtual void AddCaptureBuffer (System.IO.MemoryStream buffer)

This method is used to add a capture buffer to the internal capture buffer list. A capture buffer is used to capture all bytes read from this position forward from the input stream.

Table 3.170. Parameters

buffer	Buffer into which captured bytes are to be stored

virtual void Capture (int nbytes)

This method captures bytes from the input stream to a separate object for later analysis.

Table 3.171. Parameters

nbytes	Number of bytes to capture

virtual long DecodeIntValue (int length, bool signExtend)

This method decodes the contents of an ASN.1 integer value. It can be used for either BER or PER decoding.

Table 3.172. Parameters

length	Length of encoded contents
signExtend	Sign-extend the decoded value to form a 2's comp result

Returns: . Decoded long integer value

int [] DecodeOIDContents (int llen)

This method decodes the contents of an ASN.1 object identifier value. It can be used for either BER or PER decoding.

Table 3.173. Parameters

llen	Length of encoded contents

Returns: . Decoded object identifier value

int [] DecodeRelOIDContents (int llen)

This method decodes the contents of an ASN.1 relative object identifier value. It can be used for either BER or PER decoding.

Table 3.174. Parameters

llen	Length of encoded contents

Returns: . Decoded object identifier value

override System.IO.Stream GetInputStream ()

This method returns a reference to the current current Decode input stream object.

Returns: . New input stream object containing encoded message

virtual void HexDump ()

This method provides a hex dump of the bytes in the message being decoded.

virtual void Mark ()

This method is used to mark the current position in the input stream for retry processing.

virtual void MarkPos (ref BufferPos bufferPos)

This method marks the current position and stores it in a user-supplied structure.

Table 3.175. Parameters

bufferPos	A BufferPos structure to hold the position information.

virtual int Read ()

The read method reads a single byte from the current input stream and returns it to the caller. It will also write the byte out to all registered capture buffers.

Throws, Exception thrown by C# System.IO.Stream for I/O error for Stream as input data

Returns: . byte that was read from the input stream

Table 3.176. Exceptions

Asn1EndOfBufferException	Thrown if at end-of-stream

virtual void Read (byte[] buffer, int offset, int nbytes)

This version of the read method reads the given number of bytes from the current input stream and writes them to the specified byte array at the given offset. It also writes the data to all registered capture buffers.

Throws, Exception thrown by C# System.IO.Stream for I/O error for Stream as input data

Table 3.177. Parameters

buffer	the buffer into which the data is read
offset	the start offset of the data
nbytes	number of bytes to read

Table 3.178. Exceptions

Asn1EndOfBufferException	Thrown if at end-of-stream

virtual void Read (byte[] buffer)

This version of the read method reads the number of bytes equal to the length of the given input buffer.

Throws, Exception thrown by C# System.IO.Stream for I/O error, for Stream as input data

Table 3.179. Parameters

buffer	the buffer into which the data is read

Table 3.180. Exceptions

Asn1EndOfBufferException	thrown if at end-of-stream

int Read2Bytes ()

Read the next two bytes from the current input stream into an int, and return that int. The bytes of the int, from lowest to highest, will correspond to the bytes read from the stream, from last to first. The highest two bytes will be 0.

Each byte read will be written to all registered capture buffers.

Returns: . an int representing the 2 bytes read, as described above.

Table 3.181. Exceptions

Asn1EndOfBufferException	if at end-of-stream

int Read4Bytes ()

Read the next four bytes from the current input stream into an int, and return that int. The bytes of the int, from lowest to highest, will correspond to the bytes read from the stream, from last to first.

Each byte read will be written to all registered capture buffers.

Returns: . an int representing the 4 bytes read, as described above.

Table 3.182. Exceptions

Asn1EndOfBufferException	if at end-of-stream

abstract int ReadByte ()

This abstract method returns the next available 8-bit value from the input stream. It is implemented differently for BER/DER and PER to take into account odd alignments in PER.

Returns: . Next 8-bit byte value from input stream

virtual void RemoveCaptureBuffer (System.IO.MemoryStream buffer)

This method is used to remove a capture buffer from the internal capture buffer list. The add and remove methods can be used to get a set of raw bytes from the input stream for further processing.

Table 3.183. Parameters

buffer	Buffer in which captured bytes stored

virtual void Reset ()

This method is used to reset the current position in the decode buffer back to the location of the last 'mark' call.

virtual void ResetPos (ref BufferPos bufferPos)

This method resets the current position using the information in the passed BufferPos structure.

Table 3.184. Parameters

bufferPos	BufferPos structure containing information for the reset.

virtual void SetInputStream (byte[] msgdata, int offset, int length)

This method will set the input stream from which data is read. This version of the method allows a byte array containing encoded data to be specified.

Table 3.185. Parameters

msgdata	Byte array containing encoded message data
offset	Starting offset of data in the byte array
length	Length (in bytes) of the encoded data

virtual long Skip (long nbytes)

This method will skip over the requested number of bytes in the input stream.

Table 3.186. Parameters

nbytes	Number of bytes to skip

Returns: . Skipped number of bytes

void SkipOIDContents (int llen)

This method skips the contents of an ASN.1 object identifier value. It can be used for either BER or PER decoding.

Table 3.187. Parameters

llen	Length of encoded contents

int [] SkipRelOIDContents (int llen)

This method decodes the contents of an ASN.1 relative object identifier value. It can be used for either BER or PER decoding.

Table 3.188. Parameters

llen	Length of encoded contents

Returns: . Decoded object identifier value

int [] DecodeOIDContents (int llen, bool retval)

This method decodes the contents of an ASN.1 object identifier value. It can be used for either BER or PER decoding.

Table 3.189. Parameters

llen	Length of encoded contents
retval	If true, return the decoded value; else return null.

Returns: . Decoded object identifier value or null.

int [] DecodeRelOIDContents (int llen, bool retval)

This method decodes the contents of an ASN.1 relative object identifier value. It can be used for either BER or PER decoding.

Table 3.190. Parameters

llen	Length of encoded contents
retval	If true, return decoded value; else return null.

Returns: . Decoded object identifier value

Com::Objsys::Asn1::Runtime::Asn1DiscreteCharSet class Reference

.

Private Attributes

- int [] mCharSet

- Asn1DiscreteCharSet (System.String charSet)
- Asn1DiscreteCharSet (int [] charSet)
- override int GetCharAtIndex (int index)
- override int GetCharIndex (int charValue)
- override bool validate (String s)
- bool helpValidate (char c)

Detailed Description

This class is used to represent a discrete set of characters from a permitted alphabet.

Definition at line 35 of file Asn1DiscreteCharSet.cs

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

- Asn1DiscreteCharSet.cs

Asn1DiscreteCharSet (System.String charSet)

This constructor sets the permitted alphabet chracter set

Table 3.191. Parameters

charSet	Permitted alphabet character set

Asn1DiscreteCharSet (int[] charSet)

This constructor sets the permitted alphabet chracter set

Table 3.192. Parameters

charSet	Permitted alphabet character set

override int GetCharAtIndex (int index)

This method will fetch the character from the permitted alphabet at the given index.

Table 3.193. Parameters

index	Index of character within the character set

Returns: . Character at given index

Table 3.194. Exceptions

Asn1ConsVioException	Thrown if index not within define range

override int GetCharIndex (int charValue)

This method will determine the index of the given character within the permitted alphabet character set.

Table 3.195. Parameters

charValue	Character value to search for

Returns: . Index of character

Table 3.196. Exceptions

Asn1ConsVioException	Thrown if 'charValue' not found in set

override bool validate (String s)

This method will validate a character string by comparing its contents to the character set. If a character string contains characters that are not in the character set, this method will return false. Otherwise it returns true.

Table 3.197. Parameters

s	The string to be validated.

Returns: . False if the string contains invalid characters; true otherwise.

bool helpValidate (char c)

This function helps validate a character string by checking a character to see if it is in the character set. It returns false if a character is not contained in the set and true otherwise.

Com::Objsys::Asn1::Runtime::Asn1EncodeBuffer class Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1EncodeBuffer::InternalOutputStream

- int mSizeIncrement

Public Attributes

- const int SIZE_INCREMENT

-

-

-

- Asn1EncodeBuffer ( )
- Asn1EncodeBuffer (int sizeIncrement)

- virtual void BinDump (System.String varName)
- abstract void BinDump (System.IO.StreamWriter outs, System.String varName)
- abstract void Copy (byte value)
- void Copy (byte [] value)
- abstract void Copy (byte [] value, int offset, int len)
- override Stream GetInputStream ( )
- Stream GetOutputStream ( )
- virtual void HexDump ( )
- virtual void HexDump (System.IO.StreamWriter outs)
- abstract void Reset ( )
- abstract void Write (System.IO.Stream outs)

- static int EncodeIntSigned (long value, byte [] dest, int offset)
- static int EncodeIntUnsigned (long value, byte [] dest, int offset)

  Encode an integer value as an unsigned value (binary integer), in the minimum number of octets possible (>=1), into the given array.

- static int GetMinimalOctetsSigned (long value)
- static int GetMinimalOctetsUnsigned (long value)

Detailed Description

This is the base class to specific encode buffer classes for the different types of encoding rules (BER, DER and PER).

Definition at line 34 of file Asn1EncodeBuffer.cs

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

- Asn1EncodeBuffer.cs

int mSizeIncrement

This variable holds the user defined buffer increament size. It defines intial size and size it will be incremented by each time the buffer expands.

Definition at line 39 of file Asn1EncodeBuffer.cs

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

- Asn1EncodeBuffer.cs

Member Data Documentation

const int SIZE_INCREMENT

This constant specifies the default size of the encode buffer and size it will be incremented by each time the buffer expands. It is currently set to 1024 bytes.

Definition at line 45 of file Asn1EncodeBuffer.cs

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

- Asn1EncodeBuffer.cs

Asn1EncodeBuffer ()

The default constructor initializes the encode buffer to the default buffer size (SIZE_INCREMENT bytes).

Asn1EncodeBuffer (int sizeIncrement)

This constructor allows specification of the size increment value. This overrides the SIZE_INCREMENT constant value.

Table 3.198. Parameters

sizeIncrement	Buffer size increment in bytes

virtual void BinDump (System.String varName)

This method invokes an overloaded version of BinDump to dump the encoded message to standard output.

Table 3.199. Parameters

varName	Name of the Decoded ASN1 Type

abstract void BinDump (System.IO.StreamWriter outs, System.String varName)

This method dumps the encoded message in a human-readable format showing a bit trace of all fields to the given print output stream.

Table 3.200. Parameters

outs	Output will be written to this stream
varName	Name of the Decoded ASN1 Type

abstract void Copy (byte value)

This abstract method is used to copy a single byte to the encode buffer.

Table 3.201. Parameters

value	The byte value to copy

void Copy (byte[] value)

This method copies multiple bytes to the encode buffer

Table 3.202. Parameters

value	Array of bytes to copy to the encode buffer

abstract void Copy (byte[] value, int offset, int len)

This method copies multiple bytes from the given array to the encode buffer.

Table 3.203. Parameters

value	Array of bytes to copy to the encode buffer
offset	The first byte copied is value[offset]
len	The last byte copied is value[offset + len - 1]

override Stream GetInputStream ()

This method returns an input stream representing the encoded message.

Returns: . Input stream containing encoded message

Stream GetOutputStream ()

Return an output stream associated with this object. Anything written to the Stream will be written to this buffer using copy(byte) or copy(byte[]).

virtual void HexDump ()

This method dumps the encoded message in hex/ascii format to the standard output stream.

virtual void HexDump (System.IO.StreamWriter outs)

This method dumps the encoded message in hex/ascii format to the given print output stream.

Table 3.204. Parameters

outs	Output stream object reference

abstract void Reset ()

This method resets the buffer to allow a new record to be encoded into it. Any previously encoded data is lost.

abstract void Write (System.IO.Stream outs)

This method writes the encoded record to the given output stream.

Table 3.205. Parameters

outs	Output stream to which record is to be written

static int EncodeIntSigned (long value, byte[] dest, int offset)

Encode an integer value as a signed value (2's complement), in the minimum number of octets possible (>=1), into the given array. /p>

Table 3.206. Parameters

value	The value to encode.
dest	The array to but the encoding into. It must be large enough to hold the result, taking into account the given offset.
offset	The index where the first byte should go.

Returns: . The number of bytes.

static int EncodeIntUnsigned (long value, byte[] dest, int offset)

Table 3.207. Parameters

value	The value to encode. It must be non-negative.
dest	The array to but the encoding into. It must be large enough to hold the result, taking into account the given offset.
offset	The index where the first byte should go.

Returns: . The number of bytes.

static int GetMinimalOctetsSigned (long value)

Return the minimal number of octets required to represent the given value, when a signed representation is being used.

Returns: . The number of octets, <= 8.

static int GetMinimalOctetsUnsigned (long value)

Return the minimal number of octets required to represent the given value, when an unsigned representation is being used.

Table 3.208. Parameters

value	Must be >=0

Returns: . The number of octets, <= 8.

Com::Objsys::Asn1::Runtime::Asn1EndOfBufferExcep class Reference

- Asn1EndOfBufferException (Asn1DecodeBuffer buffer)

Detailed Description

This class defines the 'ASN.1 end of buffer' exception that is thrown when an unexpected end-of-buffer condition is encountered when decoding a message..

Definition at line 37 of file Asn1EndOfBufferException.cs

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

- Asn1EndOfBufferException.cs

Asn1EndOfBufferException (Asn1DecodeBuffer buffer)

This constructor creates an exception object with a textual message describing the tag of the duplicate element..

Table 3.209. Parameters

buffer	Decode buffer object reference

Com::Objsys::Asn1::Runtime::Asn1Enumerated class Reference

- static new readonly Asn1Tag _TAG

Public Attributes

- int mValue
- const int UNDEFINED
-
- Asn1Enumerated ( )
- Asn1Enumerated (int value_)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- virtual void Encode (Asn1XerEncodeBuffer buffer)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- virtual void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix, bool asText)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)
- virtual bool Equals (int value)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- override System.String ToString ( )

- static int ParseValue (System.String value)

- static Asn1Enumerated ( )

Detailed Description

This class represents the ASN.1 ENUMERATED built-in type. It is declared to be an abstract class and therefore cannot be used on its own. It must be extended by a specific enumerated type class.

Definition at line 37 of file Asn1Enumerated.cs

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

- Asn1Enumerated.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 10).

Definition at line 42 of file Asn1Enumerated.cs

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

- Asn1Enumerated.cs

Member Data Documentation

int mValue

This public member variable is where the enumerated value is stored. This is the value that is encoded when one of the encode methods is called. It is also where the decoded result is stored when a Decode method is called.

Definition at line 56 of file Asn1Enumerated.cs

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

- Asn1Enumerated.cs

const int UNDEFINED

The `UNDEFINED` constant is stored in the `mValue` member variable when the value of this enumerated type is undetermined.

Definition at line 48 of file Asn1Enumerated.cs

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

- Asn1Enumerated.cs

Asn1Enumerated ()

The default constructor sets the enumerated value to undefined.

Asn1Enumerated (int value_)

This constructor creates an enumerated object from a integer value.

Table 3.210. Parameters

value_	Integer value

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 enumerated value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.211. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This method encodes an ASN.1 enumerated value using the Packed Encoding Rules (PER).

Table 3.212. Parameters

buffer	PER Encode message buffer object
lower	Smallest enumerated value in the set
upper	Largest enumerated value in the set

override void Encode (Asn1OerEncodeBuffer buffer)

Encode enumerated value according to OER.

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 enumerated value using the XML encoding rules (XER).

Table 3.213. Parameters

buffer	Encode message buffer object
elemName	Element name

virtual void Encode (Asn1XerEncodeBuffer buffer)

This method encodes an ASN.1 enumerated value using the XML encoding rules (XER). This method does not add start and end tags (<tag> and </tag>), only value is encoded (<val1/> or <val2/>).

Table 3.214. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 enumerated value using the Obj-Sys XML Encoding rules. It encodes the value as XML text.

Table 3.215. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace value

virtual void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix, bool asText)

This method encodes an ASN.1 enumerated value. It is for use with extended-XER.

Table 3.216. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	XML element name space prefix
asText	If true, encode the value as XML text, otherwise encode as an empty element.

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 enumerated value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.217. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.218. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)

This method encodes an ASN.1 enumerated value using the Packed Encoding Rules (PER).

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.219. Parameters

outs	PER Output Stream object
lower	Smallest enumerated value in the set
upper	Largest enumerated value in the set

Table 3.220. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual bool Equals (int value)

This method compares this enumerated value to the given value for equality.

Table 3.221. Parameters

value	The int or enumerated value to compare with the current Object.

Returns: . `true` if the specified int value is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this enumerated value to the given value for equality.

Table 3.222. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1Enumerated.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

override System.String ToString ()

This method will return the enumeration text for a given enumerated value.

Returns: . Stringified representation of the value

static int ParseValue (System.String value)

This method will parse the given enumeration text and set the enumerated value. This method is implemented by the extending class for XER or XML code generation ONLY.

Table 3.223. Parameters

value	enumeration text

Returns: . Stringified representation of the value
AB: don't make it abstract (it is only for XER)

Com::Objsys::Asn1::Runtime::Asn1Exception class Reference

- Asn1Exception (System.String message)
- Asn1Exception (System.String message, System.Exception innerException)
- Asn1Exception (Asn1DecodeBuffer buffer, System.String message)
- string GetMessage ( )

Detailed Description

This class defines a generic ASN.1 exception for use as a base class for exceptions common to all encode/decode operations. Specific exceptions for BER, DER, and PER encoding and decoding are subclassed from this base class..

Definition at line 38 of file Asn1Exception.cs

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

- Asn1Exception.cs

Asn1Exception (System.String message)

This constructor passes the given message text to the superclass.

Table 3.224. Parameters

message	Error message text

Asn1Exception (System.String message, System.Exception innerException)

This constructor passes the given message text to the superclass.

Table 3.225. Parameters

message	Error message text
innerException	The exception that is the cause of the current exception.

Asn1Exception (Asn1DecodeBuffer buffer, System.String message)

This constructor creates the base exception object and captures the current buffer offset from the Decode buffer..

Table 3.226. Parameters

buffer	ASN.1 Decode buffer object reference
message	Error message text

Com::Objsys::Asn1::Runtime::Asn1GeneralizedTime class Reference

- static new readonly Asn1Tag _TAG
-
- Asn1GeneralizedTime ( )

- Asn1GeneralizedTime (bool useDerRules)
- Asn1GeneralizedTime (System.String data)
- Asn1GeneralizedTime (System.String data, bool useDerRules)
- override System.Int32 CompareTo (System.Object obj)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void ParseString (System.String data)
- override bool CompileString ( )
- static Asn1GeneralizedTime ( )

Detailed Description

This is a container class for holding the components of an ASN.1 generalized time string value.

Definition at line 36 of file Asn1GeneralizedTime.cs

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

- Asn1GeneralizedTime.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 24).

Definition at line 41 of file Asn1GeneralizedTime.cs

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

- Asn1GeneralizedTime.cs

Asn1GeneralizedTime ()

The default constructor creates an empty time string object.

Asn1GeneralizedTime (bool useDerRules)

This constructor creates an empty time string object and allows DER encoding rules to be specified.

Table 3.227. Parameters

useDerRules	'true' if time string should be encoded with DER/PER.

Asn1GeneralizedTime (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given time string. The format of a GeneralizedTime string is YYYYMMDDHHMMSS.n[Z][-HHMM].

Table 3.228. Parameters

data	Character string

Asn1GeneralizedTime (System.String data, bool useDer-Rules)

This version of the constructor can be used to set the string `mValue` member variable to the given time string and specify DER encoding rules be used to construct the string.

Table 3.229. Parameters

data	Character string
useDerRules	'true' if time string should be encoded with DER/PER.

override System.Int32 CompareTo (System.Object obj)

This method compares this object with Asn1Time class instance or with System.DateTime instance. Returns a negative integer, zero, or a positive integer as this object is less than, equal to, or greater than the specified object. Note that the action of this method may differentiate for different inherited Asn1Time classes.

Table 3.230. Parameters

obj	the Object to be compared.

Returns: . The difference in Ticks with the specified object.

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.231. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.232. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to stream an ASN.1 generalized time string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.233. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

override void ParseString (System.String data)

This method parses the given time string value. It will throw an exception if the string is not in the valid time format. The valid format of a GeneralizedTime string is YYYYMMDDHHMMSS.n[Z][-HHMM].

Table 3.234. Parameters

data	The time string value to be parsed.

Table 3.235. Exceptions

Asn1Exception	Thrown, if operation is failed.

override bool CompileString ()

Compiles new time string accoring X.680 (clause 41) and ISO 8601.

Returns: . true, if succeed, or false code, if error.

Com::Objsys::Asn1::Runtime::Asn1GeneralString class Reference

- static new readonly Asn1Tag _TAG

- Asn1GeneralString ( )
- Asn1GeneralString (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

- static Asn1GeneralString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 general string value.

Definition at line 36 of file Asn1GeneralString.cs

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

- Asn1GeneralString.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 27).

Definition at line 41 of file Asn1GeneralString.cs

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

- Asn1GeneralString.cs

Asn1GeneralString ()

The default constructor creates an empty string object.

Asn1GeneralString (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string value.

Table 3.236. Parameters

data	Character string

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.237. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.238. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 general string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.239. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.240. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1GraphicString class Reference

- static new readonly Asn1Tag _TAG

- Asn1GraphicString ( )
- Asn1GraphicString (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

- static Asn1GraphicString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 graphic string value.

Definition at line 36 of file Asn1GraphicString.cs

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

- Asn1GraphicString.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 25).

Definition at line 41 of file Asn1GraphicString.cs

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

- Asn1GraphicString.cs

Asn1GraphicString ()

The default constructor creates an empty string object.

Asn1GraphicString (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string value.

Table 3.241. Parameters

data	string representation of GraphicString

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 graphic string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.242. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 graphic string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.243. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 graphic string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.244. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.245. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1IA5String class Reference

- static new readonly Asn1Tag _TAG
- static readonly Asn1CharSet CHARSET

- Asn1IA5String ( )
- Asn1IA5String (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

- static Asn1IA5String ( )

Detailed Description

This is a container class for holding the components of an ASN.1 IA5 string value.

Definition at line 35 of file Asn1IA5String.cs

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

- Asn1IA5String.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 22).

Definition at line 41 of file Asn1IA5String.cs

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

- Asn1IA5String.cs

Asn1IA5String ()

The default constructor creates an empty string object.

Asn1IA5String (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string.

Table 3.246. Parameters

data	string representation of IA5String

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 IA5 string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.247. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 IA5 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.248. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 IA5 string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.249. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.250. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1IA5StringCharset class Reference

•

- Asn1IA5StringCharset ( )
- override int GetCharAtIndex (int index)
- override int GetCharIndex (int charValue)
- override bool validate (String s)

Com::Objsys::Asn1::Runtime::Asn1InputStream interface Reference

- int Available ( )
- void Close ( )
- void Mark ( )
- bool MarkSupported ( )
- void Reset ( )
- long Skip (long nbytes)

Detailed Description

This interface is a base interface for all classes, which implement an input stream functionality for decoding.

Definition at line 32 of file Asn1InputStream.cs

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

- Asn1InputStream.cs

int Available ()

Returns the number of bytes that can be read (or skipped over) from this input stream without blocking by the next caller of a method for this input stream. The next caller might be the same thread or or another thread.

Returns: . the number of bytes that can be read from this input stream without blocking.

Table 3.251. Exceptions

System.SystemException	if an I/O error occurs.

void Close ()

Closes this input stream and releases any system resources associated with the stream.

Table 3.252. Exceptions

System.SystemException	an I/O error occurs.

void Mark ()

This method is used to mark the current position in the input stream for retry processing or reseting the input stream position to current position.

bool MarkSupported ()

Tests if this input stream supports the seeking. This method is equivalent to C# `CanSeek` method of `System.IO.Stream`.

Returns: . `true` if input stream supports seeking; Otherwise `false`.

void Reset ()

This method is used to reset the current position in the input stream back to the location of the last 'mark' call. It is equivalent to calling 'Stream.Position' to marked location.

long Skip (long nbytes)

This method will skip over the requested number of bytes in the input stream.

Table 3.253. Parameters

nbytes	Number of bytes to skip

Table 3.254. Exceptions

System.SystemException	an I/O error occurs.

Returns: . Skipped number of bytes

Com::Objsys::Asn1::Runtime::Asn1Integer class Reference

- const int SIZEOF_INT

- const int SIZEOF_LONG

Public Attributes

- const Object MAX
- const Object MIN
- long mValue

- static new readonly Asn1Tag _TAG

Private Attributes

- Asn1RunTime rt

- Asn1Integer ( )
- Asn1Integer (long value)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Object lower, long upper)
- virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, Object upper)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Object lower, Object upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- void Decode16Bit (Asn1MderDecodeBuffer buffer, bool signed)
- void Decode32Bit (Asn1MderDecodeBuffer buffer, bool signed)
- void Decode8Bit (Asn1MderDecodeBuffer buffer, bool signed)
- void DecodeSigned (Asn1OerDecodeBuffer buffer, int octets)
- void DecodeSigned (Asn1OerDecodeBuffer buffer)
- void DecodeUnsigned (Asn1OerDecodeBuffer buffer, int octets)
- void DecodeUnsigned (Asn1OerDecodeBuffer buffer)
- virtual void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)

- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Object lower, long upper)
- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, Object upper)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Object lower, Object upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)
- virtual void Encode (Asn1PerOutputStream outs, Object lower, long upper)
- virtual void Encode (Asn1PerOutputStream outs, long lower, Object upper)
- virtual void Encode (Asn1PerOutputStream outs, Object lower, Object upper)
- void Encode16Bit (Asn1MderOutputStream outs, bool signed)
- void Encode32Bit (Asn1MderOutputStream outs, bool signed)
- void Encode8Bit (Asn1MderOutputStream outs, bool signed)
- override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)
- void EncodeSigned (Asn1OerEncodeBuffer buffer)
- void EncodeSigned (Asn1OerEncodeBuffer buffer, int octets)
- void EncodeUnsigned (Asn1OerEncodeBuffer buffer)
- void EncodeUnsigned (Asn1OerEncodeBuffer buffer, int octets)
- virtual bool Equals (long value)
- override bool Equals (System.Object value)
- virtual int GetBitCount ( )
- override int GetHashCode ( )
- virtual int GetUnsignedBitCount ( )
- override System.String ToString ( )

- static long DecodeValue (Asn1PerDecodeBuffer buffer)
- static long DecodeValue (Asn1PerDecodeBuffer buffer, bool retval)

- static long DecodeValue (Asn1PerDecodeBuffer buffer, long lower, long upper)
- static long DecodeValue (Asn1PerDecodeBuffer buffer, long lower, long upper, bool retval)
- static void EncodeValue (Asn1PerEncoder encoder, long val, long lower, long upper)
- static int GetBitCount (long ivalue)
- static int GetUnsignedBitCount (long ivalue)
- static void Skip (Asn1PerDecodeBuffer buffer)
- static void Skip (Asn1PerDecodeBuffer buffer, long lower, long upper)
- static Asn1Integer ( )
- void CheckRange (long lower, long upper)

Detailed Description

This class represents the ASN.1 INTEGER built-in type.

Definition at line 34 of file Asn1Integer.cs

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

- Asn1Integer.cs

Member Data Documentation

long mValue

This public member variable is where the integer value is stored. This is the value that is encoded when one of the encode methods is called. It is also where the decoded result is stored when a Decode method is called.

Definition at line 52 of file Asn1Integer.cs

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

- Asn1Integer.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 2).

Definition at line 44 of file Asn1Integer.cs

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

- Asn1Integer.cs

Asn1Integer ()

The default constructor sets the integer value to zero.

Asn1Integer (long value)

This constructor creates an integer object from a integer value.

Table 3.255. Parameters

value	Integer value

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 integer value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.256. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is stored in the public member 'mValue' in this object.

Table 3.257. Parameters

buffer	PER Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method decodes a constrained ASN.1 integer value using the Packed Encoding Rules (PER). The decoded result is stored in the public member 'mValue' in this object.

Table 3.258. Parameters

buffer	PER Decode message buffer object
lower	Lower bound of the integer range

upper	Upper bound of the integer range

virtual void Decode (Asn1PerDecodeBuffer buffer, Object lower, long upper)

This method decodes a unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The decoded result is stored in the public member 'mValue' in this object.

Table 3.259. Parameters

buffer	PER Decode message buffer object
lower	Lower bound equal MIN
upper	Upper bound of the integer range

virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, Object upper)

This method decodes a semi-constrained ASN.1 integer value using the Packed Encoding Rules (PER). The decoded result is stored in the public member 'mValue' in this object.

Table 3.260. Parameters

buffer	PER Decode message buffer object
lower	Lower bound of the integer range
upper	Upper bound equal MAX

virtual void Decode (Asn1PerDecodeBuffer buffer, Object lower, Object upper)

This method decodes a unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The decoded result is stored in the public member 'mValue' in this object.

Table 3.261. Parameters

buffer	PER Decode message buffer object
lower	Lower bound equal MIN
upper	Upper bound equal MAX

override void Decode (Asn1OerDecodeBuffer buffer)

Decode this value as if unconstrained. Subclasses may override this method to decode according to the constraints on the respective ASN.1 type.

void Decode16Bit (Asn1MderDecodeBuffer buffer, bool signed)

Decode a signed or unsigned 16-bit integer from an MDER encoding. This should be used to decode integer types that are constrained to exactly the value space of a signed/unsigned 16 bit integer.

void Decode32Bit (Asn1MderDecodeBuffer buffer, bool signed)

Decode a signed or unsigned 32-bit integer from an MDER encoding. This should be used to decode integer types that are constrained to exactly the value space of a signed/unsigned 32 bit integer.

void Decode8Bit (Asn1MderDecodeBuffer buffer, bool signed)

Decode a signed or unsigned 8-bit integer from an MDER encoding. This should be used to decode integer types that are constrained to exactly the value space of a signed/unsigned 8 bit integer.

void DecodeSigned (Asn1OerDecodeBuffer buffer, int octets)

Decode a signed integer (2's complement) of the given length into this object.

Table 3.262. Parameters

octets	The number of octets in which the value was encoded. 0 < octets <= 8

void DecodeSigned (Asn1OerDecodeBuffer buffer)

Decode a variable-size signed integer, encoded with a length, into this object, according to OER.

void DecodeUnsigned (Asn1OerDecodeBuffer buffer, int octets)

Decode an unsigned integer (binary integer) of the given length into this object.

Table 3.263. Parameters

octets	The number of octets in which the value was encoded. o < octets <= 8

void DecodeUnsigned (Asn1OerDecodeBuffer buffer)

Decode a variable-size unsigned integer, encoded with a length, into this object, according to OER.

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes an ASN.1 integer value using the XML encoding rules (XER).

Table 3.264. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 integer value using the XML schema encoding rules(asn2xsd).

Table 3.265. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 integer value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.266. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.267. Parameters

buffer	PER Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This method encodes a constrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.268. Parameters

buffer	PER Encode message buffer object
lower	Lower bound (inclusive) of integer being encoded
upper	Upper bound (inclusive) of integer being encoded

virtual void Encode (Asn1PerEncodeBuffer buffer, Object lower, long upper)

This method encodes a unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.269. Parameters

buffer	PER Encode message buffer object
lower	Lower bound equal MIN
upper	Upper bound (inclusive) of integer being encoded

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, Object upper)

This method encodes a semi-constrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.270. Parameters

buffer	PER Encode message buffer object
lower	Lower bound (inclusive) of integer being encoded
upper	Upper bound equal MAX

virtual void Encode (Asn1PerEncodeBuffer buffer, Object lower, Object upper)

This method encodes a unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.271. Parameters

buffer	PER Encode message buffer object
lower	Lower bound equal MIN
upper	Upper bound equal MAX

override void Encode (Asn1OerEncodeBuffer buffer)

Encode this value as if unconstrained. Subclasses may override this method to encode it according to the constraints on the respective ASN.1 type.

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 integer value using the XML encoding rules (XER).

Table 3.272. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 integer value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.273. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace value

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 integer value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.274. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.275. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.276. Parameters

outs	PER Encode message buffer object

Table 3.277. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)

This method encodes a constrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.278. Parameters

outs	PER Encode message buffer object
lower	Lower bound (inclusive) of integer being encoded
upper	Upper bound (inclusive) of integer being encoded

Table 3.279. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, Object lower, long upper)

This method encodes a unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.280. Parameters

outs	PER Encode message buffer object
lower	Lower bound equal MIN
upper	Upper bound (inclusive) of integer being encoded

virtual void Encode (Asn1PerOutputStream outs, long lower, Object upper)

This method encodes a semi-constrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.281. Parameters

outs	PER Encode message buffer object
lower	Lower bound (inclusive) of integer being encoded
upper	Upper bound equal MAX

virtual void Encode (Asn1PerOutputStream outs, Object lower, Object upper)

This method encodes a unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.282. Parameters

outs	PER Encode message buffer object
lower	Lower bound equal MIN
upper	Upper bound equal MAX

void Encode16Bit (Asn1MderOutputStream outs, bool signed)

This method encodes this ASN.1 INTEGER value to the MDER encoding. The value must fall in the set of 16-bit unsigned integers (if signed is false) or 16-bit signed integers (if signed is true).

void Encode32Bit (Asn1MderOutputStream outs, bool signed)

This method encodes this ASN.1 INTEGER value to the MDER encoding. The value must fall in the set of 32-bit unsigned integers (if signed is false) or 32-bit signed integers (if signed is true).

void Encode8Bit (Asn1MderOutputStream outs, bool signed)

This method encodes this ASN.1 INTEGER value to the MDER encoding. The value must fall in the set of 8-bit unsigned integers (if signed is false) or 8-bit signed integers (if signed is true).

override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)

This method encodes an ASN.1 integer value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.283. Parameters

buffer	Encode message buffer object
elemName	Element name
attribute	Element attribute value

void EncodeSigned (Asn1OerEncodeBuffer buffer)

Encode this value as a variable-size signed integer, with length, according to OER.

void EncodeSigned (Asn1OerEncodeBuffer buffer, int octets)

Encode integer value as a signed value (2's complement) in the given number of octets.

The value must fit in the given number of octets.

void EncodeUnsigned (Asn1OerEncodeBuffer buffer)

Encode this value as a variable-size unsigned integer, with length, according to OER.

void EncodeUnsigned (Asn1OerEncodeBuffer buffer, int octets)

Encode integer value as an unsigned value (binary integer) in the given number of octets, according to OER.

The value must be non-negative and fit in the given number of octets.

Table 3.284. Parameters

buffer	The buffer
octets	The number of octets to encode in: 1, 2, 4, or 8.

virtual bool Equals (long value)

This method compares this integer value to the given value for equality.

Table 3.285. Parameters

value	The long value to compare with the current Object.

Returns: . `true` if the specified long value is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this integer value to the given value for equality.

Table 3.286. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1Integer.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

virtual int GetBitCount ()

This method calculates the count of bits in the contained integer value.

Returns: . Bit count.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

virtual int GetUnsignedBitCount ()

This method calculates the count of bits in the contained unsigned integer value.

Returns: . Bit count.

override System.String ToString ()

This method will return a string representation of the integer value. The format is the ASN.1 value format for this type.

Returns: . Stringified representation of the value

static long DecodeValue (Asn1PerDecodeBuffer buffer)

This method decodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is returned.

Table 3.287. Parameters

buffer	PER Decode message buffer object

static long DecodeValue (Asn1PerDecodeBuffer buffer, bool retval)

This method decodes an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is returned.

Table 3.288. Parameters

buffer	PER Decode message buffer object
retval	If true, return the decoded value; else return 0.

static long DecodeValue (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method decodes a constrained ASN.1 integer value using the Packed Encoding Rules (PER). The decoded result is returned.

Table 3.289. Parameters

buffer	PER Decode message buffer object
lower	Lower bound of the integer range
upper	Upper bound of the integer range

static long DecodeValue (Asn1PerDecodeBuffer buffer, long lower, long upper, bool retval)

This method decodes a constrained ASN.1 integer value using the Packed Encoding Rules (PER). The decoded result is returned.

Table 3.290. Parameters

buffer	PER Decode message buffer object
lower	Lower bound of the integer range
upper	Upper bound of the integer range
retval	If true, return the decoded result; else return 0.

static void EncodeValue (Asn1PerEncoder encoder, long val, long lower, long upper)

This method encodes a constrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.291. Parameters

buffer	PER Encode message buffer object
lower	Lower bound (inclusive) of integer being encoded
upper	Upper bound (inclusive) of integer being encoded

static int GetBitCount (long ivalue)

This method calculates the count of bits in an integer value.

Table 3.292. Parameters

ivalue	Integer value in which to count bits.

Returns: . Bit count.

static int GetUnsignedBitCount (long ivalue)

This method calculates the count of bits in an unsigned integer value.

Table 3.293. Parameters

ivalue	Integer value in which to count bits.

Returns: . Bit count.

static void Skip (Asn1PerDecodeBuffer buffer)

This method skips an unconstrained ASN.1 integer value using the Packed Encoding Rules (PER). The length and contents components of the message are skipped.

Table 3.294. Parameters

buffer	PER Decode message buffer object

static void Skip (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method skips a constrained ASN.1 integer value using the Packed Encoding Rules (PER).

Table 3.295. Parameters

buffer	PER Decode message buffer object
lower	Lower bound of the integer range
upper	Upper bound of the integer range

void CheckRange (long lower, long upper)

Throw an exception if this integer's value is not in the range given by lower and upper.

Com::Objsys::Asn1::Runtime::Asn1InvalidArgExceptio class Reference

- Asn1InvalidArgException (System.String argName, System.String argValue)

Detailed Description

This class defines the 'ASN.1 invalid argument' exception that is thrown when as argument that is passed to a method is determined to be invalid (for example, not within a defined range)..

Definition at line 37 of file Asn1InvalidArgException.cs

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

- Asn1InvalidArgException.cs

Asn1InvalidArgException (System.String argName, System.String argValue)

This constructor creates an exception object with a textual message describing the argument that was invalid..

Table 3.296. Parameters

argName	Name of invalid argument

argValue	Value of invalid argument

Com::Objsys::Asn1::Runtime::Asn1InvalidChoiceOptic class Reference

- Asn1InvalidChoiceOptionException (Asn1BerDecodeBuffer buffer, Asn1Tag tag)
- Asn1InvalidChoiceOptionException (Asn1PerDecodeBuffer buffer, int index)
- Asn1InvalidChoiceOptionException ( )

Detailed Description

This class defines the 'ASN.1 invalid choice option' exception that is thrown when a CHOICE construct is detected to contain an element that is not within the given set.

Definition at line 37 of file Asn1InvalidChoiceOptionException.cs

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

- Asn1InvalidChoiceOptionException.cs

Asn1InvalidChoiceOptionException (Asn1BerDecodeBuffer buffer, Asn1Tag tag)

This constructor creates an exception object with a textual message describing the tag of the invalid element..

Table 3.297. Parameters

buffer	BER Decode buffer object reference
tag	Tag value of duplicate element

Asn1InvalidChoiceOptionException (Asn1PerDecodeBuffer buffer, int index)

This constructor creates an exception object with a textual message describing the PER choice index of the invalid element..

Table 3.298. Parameters

buffer	PER Decode buffer object reference
index	Parsed choice index value

Asn1InvalidChoiceOptionException ()

The default constructor is invoked in the encode logic if the object assigned to the choice item is not in the allowed set..

Com::Objsys::Asn1::Runtime::Asn1InvalidEnumExcep class Reference

- Asn1InvalidEnumException (long data)
- Asn1InvalidEnumException (System.String data)

Detailed Description

This class defines the 'ASN.1 invalid enum' exception that is thrown when an enumerated value is not within the defined set of values.

Definition at line 36 of file Asn1InvalidEnumException.cs

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

- Asn1InvalidEnumException.cs

Asn1InvalidEnumException (long data)

This constructor creates an exception object with a default textual message with integer value.

Table 3.299. Parameters

data	Invalid enumerated value

Asn1InvalidEnumException (System.String data)

This constructor creates an exception object with a default textual message with textual enumerated value.

Table 3.300. Parameters

data	Invalid enumerated value

Com::Objsys::Asn1::Runtime::Asn1InvalidLengthExce class Reference

- Asn1InvalidLengthException ( )

Detailed Description

This class defines the 'ASN.1 invalid length' exception that is thrown when a length is determined to be invalid.

Things that can cause this to be thrown are:

- Constructor length field is not sum of parts.
- Object identifier length is not sum of sub ID lengths.

Definition at line 42 of file Asn1InvalidLengthException.cs

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

- Asn1InvalidLengthException.cs

Asn1InvalidLengthException ()

This constructor creates an exception object with a default textual message.

Com::Objsys::Asn1::Runtime::Asn1InvalidObjectIDExc class Reference

- Asn1InvalidObjectIDException ( )

Detailed Description

This class defines the 'ASN.1 invalid object identifier' exception that is thrown when an Object Identifier is determined to be invalid.

Things that can cause this to be thrown are:

- Max subID's (128) exceeded.
- Not enough subID's in the OID (must contain at least 2).
- First subID > 2 and/or second subID > 40.

Definition at line 42 of file Asn1InvalidObjectIDException.cs

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

- Asn1InvalidObjectIDException.cs

Asn1InvalidObjectIDException ()

This constructor creates an exception object with a default textual message.

Com::Objsys::Asn1::Runtime::Asn1MessageBuffer class Reference

-

- virtual void AddNamedEventHandler (Asn1NamedEventHandler handler)
- abstract System.IO.Stream GetInputStream ( )

- virtual void InvokeCharacters (System.String svalue)
- virtual void InvokeEndElement (System.String name, int index)
- virtual void InvokeStartElement (System.String name, int index)

Detailed Description

This is the base class for all of the different message buffer types. This includes the BER and PER encode and decode message buffer classes.

Definition at line 36 of file Asn1MessageBuffer.cs

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

- Asn1MessageBuffer.cs

virtual void AddNamedEventHandler (Asn1NamedEventHandler handler)

This method adds a named event handler to the named event handler list for this buffer.

Table 3.301. Parameters

handler	Asn1NamedEventHandler object to be added

abstract System.IO.Stream GetInputStream ()

This abstract method must be implemented by all of the derived classes. It returns an input stream object reference to the message buffer contents (i.e. the encoded data).

Returns: . Input stream object reference

virtual void InvokeCharacters (System.String svalue)

This method is used by the event handling logic to invoke the 'characters' event handling method when message contents are parsed. The TypeCode property is used for the event's type code.

Table 3.302. Parameters

svalue	Stringified representation of a parsed value field

virtual void InvokeEndElement (System.String name, int index)

This method is used by the event handling logic to invoke the 'endElement' event handling method when parsing of an element within a message is completed.

Table 3.303. Parameters

name	Name of the element
index	Index of element if SEQUENCE OF or SET OF element

virtual void InvokeStartElement (System.String name, int index)

This method is used by the event handling logic to invoke the 'StartElement' event handling method when parsing of an element within a message is started.

Table 3.304. Parameters

name	Name of the element
index	Index of element if SEQUENCE OF or SET OF element

Asn1MessageBufferBase class Reference

Com::Objsys::Asn1::Runtime::Asn1MissingRequiredEx class Reference

- Asn1MissingRequiredException (Asn1BerDecodeBuffer buffer)
- Asn1MissingRequiredException (System.String elemName)

Detailed Description

This class defines the 'ASN.1 set missing required element' exception that is thrown from Decode methods when a SET construct is decoded and found to be missing a required element..

Definition at line 37 of file Asn1MissingRequiredException.cs

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

- Asn1MissingRequiredException.cs

Asn1MissingRequiredException (Asn1BerDecodeBuffer buffer)

This constructor creates an exception object with a textual message describing the error.

Table 3.305. Parameters

buffer	BER decode buffer object reference

Asn1MissingRequiredException (System.String elem-Name)

This constructor creates an exception object with a textual message describing the error including the name of the required element that is missing.

Table 3.306. Parameters

elemName	Name of missing required element

Com::Objsys::Asn1::Runtime::Asn1NamedEventHandl interface Reference

- void Characters (System.String svalue, short typeCode)
- void EndElement (System.String name, int index)
- void StartElement (System.String name, int index)

Detailed Description

This interface defines the methods that must be implemented to define a SAX-like event handler. These methods are invoked from within the generated C# decode logic when significant events occur during the parsing of an ASN.1 message.

Definition at line 34 of file Asn1NamedEventHandler.cs

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

- Asn1NamedEventHandler.cs

void Characters (System.String svalue, short typeCode)

The Characters callback method is invoked when content (primitive data) is encountered. A stringified representation of the parsed value is returned.

Table 3.307. Parameters

svalue	Stringified representation of the parsed value. The representation will be in ASN.1 value format.
typeCode	Identifier specifying the type of the parsed data variable. The enumerated list of values that might appear here is provided in the the Asn1Type class (see the documentation on this class for a full list of the names).

See also: . <seealso cref=Asn1Type The type codes are member of Asn1Type class

void EndElement (System.String name, int index)

The EndElement callback method is invoked when the end of an element within a constructed type (SEQUENCE, SET, SEQUENCE OF, SET OF, or CHOICE) is detected.

Table 3.308. Parameters

name	Name of the parsed element.
index	Index of element in array. Only used for SEQUENCE OF or SET OF elements. Set to -1 for all others.

void StartElement (System.String name, int index)

The StartElement callback method is invoked when the start of an element within a constructed type (SEQUENCE, SET, SEQUENCE OF, SET OF, or CHOICE) is encountered.

Table 3.309. Parameters

name	Name of the parsed element.
index	Index of element in array. Only used for SEQUENCE OF or SET OF elements. Set to -1 for all others.

Com::Objsys::Asn1::Runtime::Asn1Null class Reference

- static new readonly Asn1Tag _TAG
- static readonly Asn1Null NULL_VALUE

- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override void Decode (Asn1OerDecodeBuffer buffer)
- virtual void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- override bool Equals (object o)
- override int GetHashCode ( )
- override System.String ToString ( )
- static Asn1Null ( )

Detailed Description

This class represents the ASN.1 NULL built-in type.

Definition at line 35 of file Asn1Null.cs

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

- Asn1Null.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 5).

Definition at line 39 of file Asn1Null.cs

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

- Asn1Null.cs

readonly Asn1Null NULL_VALUE

The NULL_VALUE constant represents a NULL value.

Definition at line 42 of file Asn1Null.cs

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

- Asn1Null.cs

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 null value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.310. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 null value in accordance with the Packed Encoding Rules (PER).

Table 3.311. Parameters

buffer	Decode message buffer object

override void Decode (Asn1OerDecodeBuffer buffer)

Decode ASN.1 NULL type, using Octet Encoding Rules (OER).

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes an ASN.1 null value using the XML encoding rules (XER).

Table 3.312. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 null value using the XML schema encoding rules(asn2xsd).

Table 3.313. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 null value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version used the Basic Encoding Rules (BER).

Table 3.314. Parameters

buffer	Encode message buffer object

explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length (in octets) of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 null value in accordance with the Packed Encoding Rules (PER).

Table 3.315. Parameters

buffer	Encode message buffer object

override void Encode (Asn1OerEncodeBuffer buffer)

Encode ASN.1 NULL type, using Octet Encoding Rules (OER).

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 null value using the XML encoding rules (XER).

Table 3.316. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 null value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.317. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace value

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 NULL value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.318. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.319. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 null value in accordance with the Packed Encoding Rules (PER).

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.320. Parameters

outs	PER Output Stream object

Table 3.321. Exceptions

Asn1Exception	Thrown, if operation is failed.

override bool Equals (object o)

Tests for equality with any other object. Returns true if the input type is an Asn1Null object and false otherwise.

override System.String ToString ()

This method will return a string representation of the null value. The format is the ASN.1 value format for this type..

Returns: . Stringified representation of the value

Com::Objsys::Asn1::Runtime::Asn1NumericString class Reference

- static new readonly Asn1Tag _TAG
- static readonly Asn1CharSet CHARSET

- Asn1NumericString ( )
- Asn1NumericString (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- static Asn1NumericString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 numeric string value.

Definition at line 35 of file Asn1NumericString.cs

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

- Asn1NumericString.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 18).

Definition at line 42 of file Asn1NumericString.cs

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

- Asn1NumericString.cs

Asn1NumericString ()

The default constructor creates an empty string object.

Asn1NumericString (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string value.

Table 3.322. Parameters

data	string representation of numeric string

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.323. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.324. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)

This overloaded version of the Decode method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.325. Parameters

buffer	Decode message buffer object
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.326. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.327. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.328. Parameters

buffer	Encode message buffer object
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 numeric string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.329. Parameters

outs	BER Output Stream object

explicitTagging	Flag indicating explicit tagging should be done

Table 3.330. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1NumericStringCha class Reference

•

- Asn1NumericStringCharset ( )
- override int GetCharAtIndex (int index)
- override int GetCharIndex (int charValue)
- override bool validate (String s)

Com::Objsys::Asn1::Runtime::Asn1ObjectDescriptor class Reference

- static new readonly Asn1Tag _TAG

- Asn1ObjectDescriptor ( )
- Asn1ObjectDescriptor (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

- static Asn1ObjectDescriptor ( )

Detailed Description

This is a container class for holding the components of an ASN.1 object descriptor value.

Definition at line 35 of file Asn1ObjectDescriptor.cs

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

- Asn1ObjectDescriptor.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 7).

Definition at line 40 of file Asn1ObjectDescriptor.cs

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

- Asn1ObjectDescriptor.cs

Asn1ObjectDescriptor ()

The default constructor creates an empty string object.

Asn1ObjectDescriptor (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string.

Table 3.331. Parameters

data	string representation of ObjectDescriptor

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.332. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.333. Parameters

buffer	Encode message buffer object

explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 object descriptor value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.334. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.335. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1ObjectIdentifier class Reference

- static Asn1RunTime rt
- static new readonly Asn1Tag _TAG

Public Attributes

- const int MAXSUBIDS
- int [] mValue
- virtual void Append (int [] value2)
- Asn1ObjectIdentifier ( )
- Asn1ObjectIdentifier (int [] value)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override void Decode (Asn1OerDecodeBuffer buffer)

- virtual void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- override System.String ToString ( )
- String ToXMLValue ( )
- static void Skip (Asn1PerDecodeBuffer buffer)
- void DecodeString (String buffer)
- virtual void Validate ( )
- static Asn1ObjectIdentifier ( )

Detailed Description

This is a container class for holding the components of an ASN.1 object identifier value.

Definition at line 36 of file Asn1ObjectIdentifier.cs

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

- Asn1ObjectIdentifier.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 6).

Definition at line 45 of file Asn1ObjectIdentifier.cs

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

- Asn1ObjectIdentifier.cs

Member Data Documentation

const int MAXSUBIDS

The `MAXSUBIDS` constant specifies the maximum number of subidentifiers that can appear in an OID value (128).

Definition at line 51 of file Asn1ObjectIdentifier.cs

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

- Asn1ObjectIdentifier.cs

int [] mValue

The `mValue` public member variable is where the object identifier value is stored.

Definition at line 57 of file Asn1ObjectIdentifier.cs

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

- Asn1ObjectIdentifier.cs

virtual void Append (int[] value2)

This method appends an object identifier value onto the existing value. A typical use of this method would be for SNMP objects to create the base and index parts.

Table 3.336. Parameters

value2	Array of subidentifiers to append

Asn1ObjectIdentifier ()

This constructor creates an empty object identifier that can be used in a Decode method call to receive an OID value.

Asn1ObjectIdentifier (int[] value)

This constructor initializes the object identifier from the given array of integer subidentifier values.

Table 3.337. Parameters

value	Array of subidentifiers

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 object identifier value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.338. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 object identifier value using the packed encoding rules (PER).

Table 3.339. Parameters

buffer	Decode message buffer object

override void Decode (Asn1OerDecodeBuffer buffer)

Decode an ASN.1 OBJECT IDENTIFIER that was encoded according to the Octet Encoding Rules (OER).

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes an ASN.1 object identifier value using the XML encoding rules (XER).

Table 3.340. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 object identifier value using the XML schema encoding rules(asn2xsd).

Table 3.341. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 object identifier value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.342. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of encoded component in octets

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 object identifier value using the packed encoding rules (PER).

The value to be encoded is stored in the `mValue` public member variable within this class.

Table 3.343. Parameters

buffer	Encode message buffer object

override void Encode (Asn1OerEncodeBuffer buffer)

This method encodes an ASN.1 OBJECT IDENTIFIER according to Octet Encoding Rules (OER).

Table 3.344. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 object identifier value using the XML encoding rules (XER).

Table 3.345. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 object identifier value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.346. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace value

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 object identifier value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.347. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.348. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 object identifier value using the packed encoding rules (PER).

The value to be encoded is stored in the `mValue` public member variable within this class.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.349. Parameters

outs	PER Output Stream object

Table 3.350. Exceptions

Asn1Exception	Thrown, if operation is failed.

override bool Equals (System.Object value)

This method compares this object identifier to the given one for equality.

Table 3.351. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1ObjectIdentifier.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

override System.String ToString ()

This method will return a string representation of the OID value. The format is the ASN.1 value format for this type. Note that textual subidentifiers are not used, only numeric values..

Returns: . Stringified representation of the value

String ToXMLValue ()

Return the XML value representation (defined in X.680) for this object identifier.

Table 3.352. Exceptions

Asn1InvalidObjectIDException	if this is not a valid value.

static void Skip (Asn1PerDecodeBuffer buffer)

This method skips an ASN.1 object identifier value using the packed encoding rules (PER).

Table 3.353. Parameters

buffer	Decode message buffer object

void DecodeString (String buffer)

This method decodes an XMLObjectIdentifierValue into this object. Note that JSON uses this same representation.

virtual void Validate ()

Do some minimal validation. Subclasses may override this to adjust for their validation rules (e.g. Asn1RelativeOID overrides this to be more lax). Minimally, the implementation should enforce that value is not null and that there is at least one arc.

Table 3.354. Exceptions

Asn1InvalidObjectIDException	if validation fails

Com::Objsys::Asn1::Runtime::Asn1OctetString class Reference

- static Asn1RunTime rt
- static new readonly Asn1Tag _TAG

Public Attributes

- byte [] mValue
-
- static readonly char [] encodeTable
- Asn1OctetString ( )
- Asn1OctetString (byte [] data)
- Asn1OctetString (byte [] data, int offset, int nbytes)
- Asn1OctetString (System.String value)
- virtual int CompareTo (System.Object octstr)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- override void Decode (Asn1MderDecodeBuffer buffer)
- void Decode (Asn1MderDecodeBuffer buffer, int constrainedLength)
- void DecodeContent (Asn1OerDecodeBuffer buffer, int numOctets)
- virtual void DecodeXER (System.String buffer, System.String attrs, bool base64)

  </p>
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)

- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1MderOutputStream outs)
- void Encode (Asn1MderOutputStream outs, int constrainedLength)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- virtual void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix, bool base64)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)
- override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)
- void EncodeContent (Asn1OerEncodeBuffer buffer)
- bool Equals (byte [] value)
- bool Equals (String s)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- int GetMderLength ( )
- virtual System.IO.Stream toInputStream ( )
- override System.String ToString ( )

- static System.String EncodeBase64Binary (byte [] data)
- static void Skip (Asn1PerDecodeBuffer buffer)
- static void Skip (Asn1PerDecodeBuffer buffer, long lower, long upper)

- void AllocByteArray (int nbytes)
- void ReAllocByteArray (int nbytes)

- static Asn1OctetString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 octet string value.

Definition at line 37 of file Asn1OctetString.cs

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

- Asn1OctetString.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 4).

Definition at line 45 of file Asn1OctetString.cs

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

- Asn1OctetString.cs

Member Data Documentation

byte [] mValue

This variable holds the octet string value. These are the octets that are encoded when encode is invoked. It is also where the decoded octet string is stored after a Decode operation.

Definition at line 52 of file Asn1OctetString.cs

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

- Asn1OctetString.cs

Asn1OctetString ()

This constructor creates an empty octet string that can be used in a Decode method call to receive an octet string value.

Asn1OctetString (byte[] data)

This constructor initializes an octet string from the given byte array.

Table 3.355. Parameters

data	Byte array containing an octet string in binary form.

Asn1OctetString (byte[] data, int offset, int nbytes)

This constructor initializes an octet string from a portion of the given byte array. A new byte array is created starting at the given offset and consisting of the given number of bytes.

Table 3.356. Parameters

data	Byte array containing an octet string in binary form.
offset	The offset in array at which to begin copy.
nbytes	Number of bytes to copy from target array

Asn1OctetString (System.String value)

This constructor parses the given ASN.1 value text (either a binary or hex data string) and assigns the values to the internal bit string.

Examples of valid value formats are as follows:

Binary string: '11010010111001'B

Hex string: '0fa56920014abc'H

Char string: 'abcdefg'

Table 3.357. Parameters

value	The ASN.1 value specification text

virtual int CompareTo (System.Object octstr)

This method compares two Asn1OctetString objects for equality. The OCTET STRING's are equal if a) all octets are equal, and b) the lengths are the same.

This method is required to implement the Comparable interface used for sorting.

Table 3.358. Parameters

octstr	Asn1OctetString to compare

Returns: . 0 if equal, 1 if this string is greater than supplied string, -1 if this string is less than supplied string.

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 octet string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.359. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 octet string value using the packed encoding rules (PER). The string is assumed to not contain a size constraint.

Table 3.360. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method decodes a sized ASN.1 octet string value using the packed encoding rules (PER).

Table 3.361. Parameters

buffer	Decode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

override void Decode (Asn1OerDecodeBuffer buffer)

This method decodes an ASN.1 octet string using the Octet Encoding Rules (OER).

This implementation expects a length determinant in the encoding. This method may be overridden for ASN.1 types that have a fixed length to invoke DecodeContent with the predetermined length.

override void Decode (Asn1MderDecodeBuffer buffer)

Decode an unconstrained octet string from the MDER encoding into this object.

void Decode (Asn1MderDecodeBuffer buffer, int constrainedLength)

Decode an octet string from the MDER encoding into this object.

Table 3.362. Parameters

constrainedLength	The constrained length of the type being encoded. Pass -1 if the type is unconstrained. For a constrained length octet string, exactly that many octets will be read.

void DecodeContent (Asn1OerDecodeBuffer buffer, int numOctets)

This method decodes an ASN.1 OCTET STRING's content that was encoded according to OER, given the total number of octets. This does not decode a length determinant.

Table 3.363. Parameters

buffer	Decode message buffer object

numOctets	Total number of octets encoding the content.

virtual void DecodeXER (System.String buffer, System.String attrs, bool base64)

This method decodes ASN.1 octet string type using the XML encoding rules (XER). Extended-XER is supported by the base64

Table 3.364. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag
base64	pass true if encoding is base64 (extended-XER only)

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 octet string type using the XML schema encoding rules(asn2xsd).

Table 3.365. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 octet string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.366. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an unconstrained ASN.1 octet string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Table 3.367. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This method encodes a size-constrained ASN.1 octet string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Table 3.368. Parameters

buffer	Encode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

override void Encode (Asn1OerEncodeBuffer buffer)

This method encodes this ASN.1 OCTET STRING value, according to OER. This encodes the length determinant, assuming that the OCTET STRING is not fixed-size constrained. Subclasses may override this to simply call Encode-Content for fixed-size constrained strings.

override void Encode (Asn1MderOutputStream outs)

Encode this octet string into the MDER encoding. This should be used for octet string types that are not of fixed length.

Table 3.369. Exceptions

Asn1MderUnsupported	if the actual length > 65535 (maximum allowed by MDER).

void Encode (Asn1MderOutputStream outs, int constrainedLength)

Encode this octet string into the MDER encoding.

Table 3.370. Parameters

constrainedLength	The constrained length of the type being encoded. Pass -1 if the type is unconstrained.

Table 3.371. Exceptions

Asn1ConsVioException	if a constrained length is given and the value is not exactly that length.
Asn1MderUnsupported	if the length is unconstrained and the actual length > 65535 (maximum allowed by MDER).

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes ASN.1 octet string type using the XML encoding rules (XER).

Table 3.372. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes ASN.1 octet string type as an xmlhstring.

Table 3.373. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string
nsPrefix	Element namespace value

virtual void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix, bool base64)

This method encodes ASN.1 octet string type using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.374. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string
nsPrefix	Element namespace value
base64	Pass true to encode as base64 (extended-XER only, including XSD compilation)

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 octet string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.375. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.376. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an unconstrained ASN.1 octet string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.377. Parameters

outs	PER Output Stream object

Table 3.378. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)

This method encodes a size-constrained ASN.1 octet string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.379. Parameters

outs	PER Output Stream object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

Table 3.380. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)

This method encodes ASN.1 octet string type using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.381. Parameters

buffer	Encode message buffer object
elemName	XML element name used to wrap string
attribute	Element attribute value

void EncodeContent (Asn1OerEncodeBuffer buffer)

This method encodes the content of an ASN.1 OCTET STRING, according to OER. (The length determinant is not encoded.)

Table 3.382. Parameters

buffer	Encode message buffer object

bool Equals (byte[] value)

This method compares this octet string value to the given value for equality.

Table 3.383. Parameters

value	The byte array to compare with the current Object.

Returns: . `true` if the specified byte array is equal to the current Object; otherwise, `false`.

bool Equals (String s)

This method compares the given ASN.1 OCTET STRING value to this OCTET STRING value for equality.

Table 3.384. Parameters

value	The ASN.1 OCTET STRING value to compare with the current Object. Examples of valid value formats are as follows: Binary string: '11010010111001'B Hex string: '0fa56920014abc'H

Returns: . `true` if the specified value is equal to this value; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this octet string value to the given value for equality.

Table 3.385. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1OctetString.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

int GetMderLength ()

Return the length of the MDER encoding for this type, assuming that the type is NOT a fixed length OCTET STRING (if it is fixed length, you need not call this method!).

virtual System.IO.Stream toInputStream ()

This method will return a byte array input stream representation of the octet string value.

Returns: . Reference to System.IO.MemoryStream object

override System.String ToString ()

This method will return a string representation of the octet string value. The format is the ASN.1 value format for this type..

Returns: . Stringified representation of the value

static System.String EncodeBase64Binary (byte[] data)

Encodes xsd:Base64Binary data into ASCII character using the algorithm defined in RFC2045, as defined in w3c stadard http://www.w3.org/tr/2001/rec-xmlschema-2-20010502#base64Binary

Table 3.386. Parameters

base64binary	Array containing base64binary

Returns: . Encoded ASCII string

static void Skip (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 octet string value using the packed encoding rules (PER). The string is assumed to not contain a size constraint.

Table 3.387. Parameters

buffer	Decode message buffer object

static void Skip (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method decodes a sized ASN.1 octet string value using the packed encoding rules (PER).

Table 3.388. Parameters

buffer	Decode message buffer object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

Com::Objsys::Asn1::Runtime::Asn1OpenExt class Reference

Public Attributes

- System.Collections.ArrayList mValue

•

Private Attributes

- bool [] mOptBits

- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void DecodeComponent (Asn1BerDecodeBuffer buffer)
- virtual void DecodeEventComponent (Asn1BerDecodeBuffer buffer)
- virtual Asn1OpenType DecodeOpenType (Asn1PerDecodeBuffer buffer, bool present, int index)
- Asn1OpenType DecodeOpenType (Asn1OerDecodeBuffer buffer, bool present, int index)

- override void Encode (Asn1OerEncodeBuffer buffer)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer)
- override void Encode (Asn1XmlEncoder buffer)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- void EncodeExtBits (Asn1OerEncodeBuffer buffer)
- virtual void EncodeExtBits (Asn1PerEncodeBuffer buffer)
- bool HasPresentExtensions ( )
- virtual void SetOpenType (Asn1OpenType obj, int index)
- virtual void ShrinkArray (int numrecs)
- override System.String ToString ( )

Detailed Description

This is a container class for holding open type elements that may occur within an open type extension (i.e. a ... at the end of a constructed type or a ..., ... at some other point in a constructed type).

Definition at line 39 of file Asn1OpenExt.cs

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

- Asn1OpenExt.cs

Member Data Documentation

System.Collections.ArrayList mValue

The value is a list of Asn1OpenType objects. Each of these objects contains a fully encoded extension item.

Definition at line 44 of file Asn1OpenExt.cs

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

- Asn1OpenExt.cs

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 open type extension value using the Basic Encoding Rules (BER).

Table 3.389. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length if implicit element

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an open type extension in a SEQUENCE or SET construct using the packed encoding rules (PER). This method will capture each extension item in a separate open type object and store it in the `mValue` public member list variable. If optional items are absent, null placeholders will be inserted in the list.

Table 3.390. Parameters

buffer	Decode message buffer object

virtual void DecodeComponent (Asn1BerDecodeBuffer buffer)

This method decodes a single component of a BER open type extension by decoding an open type value and appending it to the list of open type objects.

Table 3.391. Parameters

buffer	Decode message buffer object

virtual void DecodeEventComponent (Asn1BerDecodeBuffer buffer)

This method decodes a single component of a BER open type extension by decoding an open type value and appending it to the list of open type objects, this function also triggers event handler code, with element name "..."

Table 3.392. Parameters

buffer	Decode message buffer object

virtual Asn1OpenType DecodeOpenType (Asn1PerDecodeBuffer buffer, bool present, int index)

This method decodes a single open type extension item in a SEQUENCE or SET construct using the packed encoding rules (PER). It will then add the item to the open extension element list.

Table 3.393. Parameters

buffer	Decode message buffer object
present	Flag indicating whether element is present
index	Index of element in the object array

Returns: . Decoded open type

Asn1OpenType DecodeOpenType (Asn1OerDecodeBuffer buffer, bool present, int index)

This method decodes a single open type extension item in a SEQUENCE or SET construct using the octet encoding rules (OER). It will then add the item to the open extension element list.

Table 3.394. Parameters

buffer	Decode message buffer object
present	Flag indicating whether element is present.
index	Index of element in the object array. If index >= value.Count, the item is appended to the list (all such values of index thus have the same behavior).

override void Encode (Asn1OerEncodeBuffer buffer)

This method encodes the ASN.1 open type extensions using Octet Encoding Rules (OER).

Table 3.395. Parameters

buffer	Encode message buffer object

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 open type extension value using the Basic Encoding Rules (BER).

Table 3.396. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating element is explicitly tagged

Returns: . Length of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 open type extension value using the Packed Encoding Rules (PER).

Table 3.397. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XerEncoder buffer)

This method encodes an ASN.1 open type extension value using the XML Encoding Rules (XER).

Table 3.398. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XmlEncoder buffer)

This method encodes an ASN.1 open type extension value using the XML Encoding as specified in the XML schema standard (asn2xsd).

Table 3.399. Parameters

buffer	Encode message buffer object

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes an ASN.1 open type extension value using the Basic Encoding Rules (BER) and writes it into the stream.

Also throws any exception thrown by the underlying Asn1BerOutputStream.

Table 3.400. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating element is explicitly tagged

Table 3.401. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 open type extension value using the Packed Encoding Rules (PER).

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.402. Parameters

outs	PER Output Stream object

Table 3.403. Exceptions

Asn1Exception	Thrown, if operation is failed.

void EncodeExtBits (Asn1OerEncodeBuffer buffer)

This method encodes an ASN.1 open type extension value bits using the Packed Encoding Rules (PER).

Table 3.404. Parameters

buffer	Encode message buffer object

virtual void EncodeExtBits (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 open type extension value bits using the Packed Encoding Rules (PER).

Table 3.405. Parameters

buffer	Encode message buffer object

bool HasPresentExtensions ()

Return true if this contains present extensions. Null values in the value list represent absent extensions.

virtual void SetOpenType (Asn1OpenType obj, int index)

This method will add the given open type object to the open extension element list at the given index.

Table 3.406. Parameters

obj	Open type object
index	Index in open type list where element is to be placed

virtual void ShrinkArray (int numrecs)

This method adjusts the size of the open type component array downward to the given size value.

Table 3.407. Parameters

numrecs	Number of entries the array should hold

override System.String ToString ()

This method will return a string representation of the open extension value. The format is the ASN.1 value format for each open type in the extension.

Returns: . Stringified representation of the value

Com::Objsys::Asn1::Runtime::Asn1OpenType class Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1OpenType::SaxHandler

Public Attributes

- const int
- const int JSON
- const int UNKNOWN

  The possible data encoding values.

- Asn1EncodeBuffer mEncodeBuffer
- int mLength

Protected Attributes

- int dataEncoding
- const System.String EDATAMSG

Private Attributes

- char [] charData
- Asn1XerSaxHandler mSaxHandler
-
- Asn1OpenType ( )
- Asn1OpenType (byte [] data)

- Asn1OpenType (byte [] data, int encoding)
- Asn1OpenType (byte [] data, int offset, int nbytes)
- Asn1OpenType (byte [] data, int offset, int nbytes, int encoding)
- Asn1OpenType (Asn1EncodeBuffer buffer)
- Asn1OpenType (string data, int encoding)
- Asn1OpenType (char [] data, int encoding)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override void Decode (Asn1OerDecodeBuffer buffer)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix)
- override void Encode (Asn1XmlEncoder buffer)
- override void Encode (Asn1XerEncoder buffer, String elemName)
- override void Encode (Asn1XerEncoder buffer)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- void EncodeAsExtension (Asn1XmlEncoder buffer)
- void EncodeAsExtension (Asn1XerEncoder buffer)
- char [] GetCharData ( )
- int GetDataEncoding ( )
- virtual Asn1XerSaxHandler GetSaxHandler ( )
- virtual Asn1XerSaxHandler GetSaxHandler (bool captureOuterElem)
- void SetBinaryData (byte [] data, int encoding)
- void SetCharData (String data, int encoding)
- void SetCharData (char [] data, int encoding)
- override System.String ToString ( )
- void EncodeUnknownXml (Asn1XmlXerEncoder buffer)
- byte [] GetBytes ( )

Detailed Description

This is a container class for holding an ASN.1 open type value.

Where the data is internally stored and how it is interpreted is controlled by a "dataEncoding" field. You can specify the data encoding when creating the object. It will also be set when decoding. The following explains the possible data encodings:

- UNKNOWN: "mValue" is used to hold byte data. It is interpreted as the encoding of the actual value according to some unknown encoding rules (BER, PER, OER, XER, JSON, or some other).
- BER, PER, OER: "mValue" is used to hold byte data. It is interpreted as the encoding of the actual value according to BER, PER, or OER, respectively.
- XER: "mValue" is used to hold byte data. It is interpreted as the the encoding of the actual value in XML (whether X.693 or Obj-Sys rules), where the XML characters are encoded to bytes using the UTF-8 character encoding.
- JSON: a private field is used to hold character data. The character data is the encoding of the actual value in JSON.

Note especially that the data encoding indicates the encoding rules used to encode the actual value. That result is typically encoded as part of some larger encoding. The data encoding does NOT indicate the rules used for that larger encoding. For example, using an xmlhstring representation, XER and JSON will encode an open type that was previously encoded using any arbitrary encoding rules. When decoding such data, the data encoding will be UNKNOWN, not XER nor JSON.

Definition at line 72 of file Asn1OpenType.cs

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

- Asn1OpenType.cs

Asn1EncodeBuffer mEncodeBuffer

The encode buffer into which component type will be encoded. /p>

Definition at line 89 of file Asn1OpenType.cs

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

- Asn1OpenType.cs

int mLength

Length of the pre-encoded component. /p>

Definition at line 85 of file Asn1OpenType.cs

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

- Asn1OpenType.cs

Member Data Documentation

int dataEncoding

Specifies the nature of the data held by this object. /p>

Definition at line 93 of file Asn1OpenType.cs

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

- Asn1OpenType.cs

Member Data Documentation

char [] charData

The character data for a character-based encoding. At present, used only for dataEncoding == JSON /p>

Definition at line 99 of file Asn1OpenType.cs

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

- Asn1OpenType.cs

Asn1OpenType ()

This constructor creates an empty type that can be used in a Decode method call to receive an encoded value. The data encoding is UNKNOWN.

Asn1OpenType (byte[] data)

This constructor initializes an open type from the given byte array. The array is assumed to contain a previously encoded message component. The data encoding is UNKNOWN.

Table 3.408. Parameters

data	Byte array containing a previously encoded message component.

Asn1OpenType (byte[] data, int encoding)

This constructor initializes an open type from the given byte array. The array is assumed to contain a previously encoded message component.

Table 3.409. Parameters

data	Byte array containing a previously encoded value.
encoding	The encoding that describes the meaning of data. Any of the encodings other than JSON.

Asn1OpenType (byte[] data, int offset, int nbytes)

This constructor initializes the open type from a portion of the given byte array. A new byte array is created starting at the given offset and consisting of the given number of bytes. The encoding is UNKNOWN.

Table 3.410. Parameters

data	Byte array containing a previously encoded value.
offset	The offset in array at which to begin copy.
nbytes	Number of bytes to copy from target array

Asn1OpenType (byte[] data, int offset, int nbytes, int encoding)

This constructor initializes the open type from a portion of the given byte array. A new byte array is created starting at the given offset and consisting of the given number of bytes. /p>

Table 3.411. Parameters

data	Byte array containing a previously encoded value.
encoding	Starting offset in data from which to copy bytes
nbytes	Number of bytes to copy from target array
offset	The encoding that describes the meaning of data. Any of the encodings other than JSON.

Asn1OpenType (Asn1EncodeBuffer buffer)

This constructor initializes an open type using an encoded component. This can be used if a header (for example, a ROSE header) is being prepended to a pre-encoded component. The data encoding is derived from the type of Asn1EncodeBuffer given, and is possibly UNKNOWN.

Table 3.412. Parameters

buffer	Reference to encode buffer into which component type was encoded.

Asn1OpenType (string data, int encoding)

Convenience constructor. Same as Asn1OpenType(data.ToCharArray(), encoding)

Asn1OpenType (char[] data, int encoding)

Create Asn1OpenType on the given character data. /p>

Table 3.413. Parameters

data	The character data array.
encoding	The encoding. Either XER or JSON. If JSON, data is taken to be the JSON encoding of the actual value. The array is not copied, so beware using it after passing it here. The "mValue" field will be assigned null. If XER, data is taken to be the XER encoding of the actual value. The "mValue" field will be assigned the UTF-8 character encoding of the given characters.

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 open type value. The data encoding will be set to BER.

Table 3.414. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 open type value using the packed encoding rules (PER). The data encoding will be set to PER.

Table 3.415. Parameters

buffer	Decode message buffer object

override void Decode (Asn1OerDecodeBuffer buffer)

This method decodes an ASN.1 open type value using the Octet Encoding Rules (OER). This object will subsequently contain the encoding of the open type (which should be OER).

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 open type value. The value is assumed to be an already-encoded BER message component and will be copied to the encoded buffer. An optimization is available in which no copy will performed if the encoded component is already present in the encode buffer. This is done if the form of constructor specifiying only a component length is used.

Table 3.416. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating element is explicitly tagged

Returns: . Length of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 open type value using the Packed Encoding Rules (PER). The data should be the value pre-encoded encoded in PER.

Table 3.417. Parameters

buffer	Encode message buffer object

override void Encode (Asn1OerEncodeBuffer buffer)

This method encodes an ASN.1 open type value using the Octet Encoding Rules (OER). The data should be the value pre-encoded in OER.

Table 3.418. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix)

This method encodes an ASN.1 open type value using the XML Encoding as specified in the XML schema standard(asn2xsd). If the data encoding is XER, the data is written as-is. Otherwise, the data's hexadecimal representation is encoded (i.e. xmlhstring).

Table 3.419. Parameters

buffer	Encode message buffer object
elemName	Ignored
nsPrefix	Ignored

override void Encode (Asn1XmlEncoder buffer)

This method encodes an ASN.1 open type value using the XML Encoding as specified in the XML schema standard(asn2xsd).

If the data encoding is XER, the data is written as-is. Otherwise, the data's hexadecimal representation is encoded (i.e. xmlhstring).

Table 3.420. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XerEncoder buffer, String elemName)

This method encodes an ASN.1 open type value using the XML Encoding as specified in the XML schema standard(asn2xsd).

If the data encoding is XER, the data is written as-is. Otherwise, the data's hexadecimal representation is encoded (i.e. xmlhstring).

Table 3.421. Parameters

buffer	Encode message buffer object
elemName	Ignored

override void Encode (Asn1XerEncoder buffer)

This method encodes an ASN.1 open type value using the XML Encoding Rules (XER).

If the data encoding is XER, the data is written as-is. Otherwise, the data's hexadecimal representation is encoded (i.e. xmlhstring).

Table 3.422. Parameters

buffer	Encode message buffer object

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 open type value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER). The data should be the value pre-encoded in BER.

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.423. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.424. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 open type value using the Packed Encoding Rules (PER). The data should be the value pre-encoded in PER.

Also throws any exception thrown by the underlying Asn1PerOutputStream.

Table 3.425. Parameters

outs	PER Output Stream object

Table 3.426. Exceptions

Asn1Exception	Thrown, if operation is failed.

void EncodeAsExtension (Asn1XmlEncoder buffer)

This method encodes an extension element to XML. If the data encoding is other than XER, the hexadecimal representation of the data is encoded inside an XML comment.

Table 3.427. Parameters

buffer	Encode message buffer object

void EncodeAsExtension (Asn1XerEncoder buffer)

This method encodes an extension element to XML. If the data encoding is other than XER, its hexadecimal representation is encoded inside an XML comment.

Table 3.428. Parameters

buffer	

char [] GetCharData ()

Returns the character data for the open type. Currently, this is only supported when getDataEncoding() returns JSON.

Returns: . The char data. The actual internal array is returned; a copy is not made.

Table 3.429. Exceptions

RuntimeException	if data encoding indicates there is no char data

int GetDataEncoding ()

Return the data encoding of the data.

virtual Asn1XerSaxHandler GetSaxHandler ()

This method returns the Asn1XerOpenType.SaxHandler class instance used for ASN.1 XER encoding. If this is the first invocation of any of the GetSaxHandler methods on this object, the returned handler will capture the outer element start/end tags. Otherwise, this will simply return the same SAX handler as previously returned.

Returns: . Asn1XerOpenType.SaxHandler object

virtual Asn1XerSaxHandler GetSaxHandler (bool captureOuterElem)

This method returns the Asn1XerOpenType.SaxHandler class instance used for ASN.1 XER encoding.

Table 3.430. Parameters

captureOuterElem	Pass true if the outer element start and end tags (if present) should be captured. The outer element is the element for which startElement is called when the level is the start level. Note that this parameter is ignored if you have previously invoked one of the getSaxHandler methods on this object.

Returns: . Asn1XerOpenType.SaxHandler object

void SetBinaryData (byte[] data, int encoding)

Sets the binary data for the open type and assigns the data encoding to the given encoding.

Table 3.431. Parameters

data	The binary data that make up an encoding of the actual value.
encoding	Identifies the encoding rules for which data is an encoding. This can be any of the encodings except JSON.

void SetCharData (String data, int encoding)

Convenience method; same as `SetCharData(mValue.toCharArray(), encoding)`

void SetCharData (char[] data, int encoding)

Sets the character data for the open type and assigns the data encoding to the given encoding.

Table 3.432. Parameters

data	The character data that make up an encoding of the actual value.
encoding	Permissible values are JSON and XER. If XER, the given data will be converted to bytes using UTF-8 and held in the "mValue" field. If JSON, the given data will be held as-is. The array will not be copied. The "mValue" field will be assigned null.

override System.String ToString ()

This method will return a string representation of the open type value. If the data encoding is XER or JSON, the string will consist of the corresponding characters (in the case of XER, the byte data is converted to characters using UTF-8). In all other cases, the hexadecimal representation of the byte data is returned.

Returns: . Stringified representation of the value

void EncodeUnknownXml (Asn1XmlXerEncoder buffer)

Encodes binary data to XML inside a comment. The data is encoded into a comment because we won't know the element to encode to, since we don't know how to decode the binary content.

Table 3.433. Parameters

buffer	

byte [] GetBytes ()

Return the array of bytes representing the open type data. Generally, this will simply return "mValue", but for cases where the data is held in charData, it will convert the character data to byte data using an appropriate character encoding.

Com::Objsys::Asn1::Runtime::Asn1OutputStream class Reference

Private Attributes

- Asn1Context context
- System.IO.Stream os
-
-
-
-
-
-

- Asn1OutputStream (System.IO.Stream os)
- override void Close ( )
- override void Flush ( )
- override int Read (byte [] buffer, int offset, int count)
- override long Seek (long offset, System.IO.SeekOrigin origin)
- override void SetLength (long value)
- virtual void Write (byte [] b)

- override void Write (System.Byte [] b, int off, int len)
- void Write2Bytes (int value)
- void Write4Bytes (int value)
- virtual void WriteByte (int b)
- override void WriteByte (byte b)

Detailed Description

This abstract class implements the base output stream to encode ASN.1 messages.

Definition at line 35 of file Asn1OutputStream.cs

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

- Asn1OutputStream.cs

System.IO.Stream os

C# Stream object

Definition at line 39 of file Asn1OutputStream.cs

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

- Asn1OutputStream.cs

Asn1OutputStream (System.IO.Stream os)

This constructor creates an output stream object.

Table 3.434. Parameters

os	The underlying System.IO.Stream object.

override void Close ()

Closes this output stream and releases any system resources associated with this stream. The general contract of `close` is that it closes the output stream. A closed stream cannot perform output operations and cannot be reopened.

Table 3.435. Exceptions

System.IO.IOException	An error occurred while trying to close the stream.

override void Flush ()

Flushes this output stream and forces any buffered output bytes to be written out. The general contract of `flush` is that calling it is an indication that, if any bytes previously written have been buffered by the implementation of the output stream, such bytes should immediately be written to their intended destination.

Table 3.436. Exceptions

System.IO.IOException	An I/O error occurs.
System.ObjectDisposedException	The stream is closed.

override int Read (byte[] buffer, int offset, int count)

This method always throws NotSupportedException. Asn1OutputStream doesn't support reading.

Table 3.437. Parameters

buffer	When this method returns, contains the specified byte array with the values between offset and (offset + count - 1) replaced by the bytes read from the current source.
offset	The byte offset in array at which to begin reading.
count	The maximum number of bytes to read.

Returns: . The total number of bytes read into the buffer.

Table 3.438. Exceptions

System.NotSupportedException	The stream does not support reading.

override long Seek (long offset, System.IO.SeekOrigin origin)

Sets the current position of this stream to the given value.

Table 3.439. Parameters

offset	The point relative to origin from which to begin seeking.
origin	Specifies the beginning, the end, or the current position as a reference point for origin, using a value of type SeekOrigin.

Returns: . The new position in the stream.

Table 3.440. Exceptions

System.IO.IOException	An I/O error occurs.
System.NotSupportedException	The stream does not support seeking, such as if the FileStream is constructed from a pipe or console output.
System.ArgumentException	Attempted seeking before the beginning of the stream.

System.ObjectDisposedException	Methods were called after the stream was closed.

override void SetLength (long value)

Sets the length of this stream to the given value.

Table 3.441. Parameters

value	The new length of the stream.

Table 3.442. Exceptions

System.IO.IOException	An I/O error has occurred.
System.NotSupportedException	The stream does not support both writing and seeking.
System.ArgumentOutOfRangeException	Attempted to set the value parameter to less than 0.

virtual void Write (byte[] b)

Writes `b.length` bytes from the specified byte array to this output stream. The general contract for `write(b)` is that it should have exactly the same effect as the call `Write(b, 0, b.length)`.

Table 3.443. Parameters

b	the data.

Table 3.444. Exceptions

System.ArgumentNullException	array is a null reference
System.ArgumentException	offset and count describe an invalid range in array.
System.ArgumentOutOfRangeException	offset or count is negative.
System.IO.IOException	An I/O error occurs.
System.ObjectDisposedException	The stream is closed.
System.NotSupportedException	The current stream instance does not support writing.

override void Write (System.Byte[] b, int off, int len)

Writes `len` bytes from the specified byte array starting at offset `off` to this output stream.

Table 3.445. Parameters

b	The byte array data to be written.

off	The offset in array at which to begin write.
len	the number of bytes to write.

Table 3.446. Exceptions

System.ArgumentNullException	array is a null reference
System.ArgumentException	offset and count describe an invalid range in array.
System.ArgumentOutOfRangeException	offset or count is negative.
System.IO.IOException	An I/O error occurs.
System.ObjectDisposedException	The stream is closed.
System.NotSupportedException	The current stream instance does not support writing.

void Write2Bytes (int value)

Write the lowest two bytes of value to the output stream. The lowest byte is written last.

Table 3.447. Parameters

value	

void Write4Bytes (int value)

Write the four bytes of value to the output stream. The lowest byte is written last.

Table 3.448. Parameters

value	

virtual void WriteByte (int b)

Writes the specified byte to this output stream. The general contract for `Write` is that one byte is written to the output stream. The byte to be written is the eight low-order bits of the argument `b`. The 24 high-order bits of `b` are ignored.

Table 3.449. Parameters

b	the `byte`.

Table 3.450. Exceptions

System.ObjectDisposedException	The stream is closed.

System.NotSupportedException	The stream does not support writing.

override void WriteByte (byte b)

Writes the specified byte to this output stream. The general contract for `Write` is that one byte is written to the output stream. The byte to be written is the eight low-order bits of the argument `b`. The 24 high-order bits of `b` are ignored.

Table 3.451. Parameters

b	the `byte`.

Table 3.452. Exceptions

System.ObjectDisposedException	The stream is closed.
System.NotSupportedException	The stream does not support writing.

Com::Objsys::Asn1::Runtime::Asn1PrintableString class Reference

- static new readonly Asn1Tag _TAG
- static readonly Asn1CharSet CHARSET

- Asn1PrintableString ( )
- Asn1PrintableString (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

- static Asn1PrintableString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 printable string value.

Definition at line 35 of file Asn1PrintableString.cs

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

- Asn1PrintableString.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 19).

Definition at line 43 of file Asn1PrintableString.cs

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

- Asn1PrintableString.cs

Asn1PrintableString ()

The default constructor creates an empty string object.

Asn1PrintableString (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string.

Table 3.453. Parameters

data	value string

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.454. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.455. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to stream an ASN.1 printable string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws Exception, if any exception thrown by the underlying System.IO.Stream.

Table 3.456. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Com::Objsys::Asn1::Runtime::Asn1PrintableStringCha class Reference

•

- Asn1PrintableStringCharset ( )
- override int GetCharAtIndex (int index)
- override int GetCharIndex (int charValue)
- override bool validate (String s)

Com::Objsys::Asn1::Runtime::Asn1Real class Reference

- static new readonly Asn1Tag _TAG

Private Attributes

- const int MINUS_INFINITY
- const int MINUS_ZERO
- const int NOT_A_NUMBER
- const int PLUS_INFINITY
- const int REAL_BASE_16
- const int REAL_BASE_2

- const int REAL_BASE_8
- const int REAL_BASE_MASK
- const int REAL_BINARY
- const int REAL_EXPLEN_1
- const int REAL_EXPLEN_2
- const int REAL_EXPLEN_3
- const int REAL_EXPLEN_LONG
- const int REAL_EXPLEN_MASK
- const int REAL_FACTOR_MASK
- const int REAL_ISO6093_MASK
- const int REAL_SIGN

Public Attributes

- double mValue

Protected Attributes

- const int ANY_BASE
- const int ANY_BINARY
- const int BASE_2_ONLY

- Asn1Real ( )
- Asn1Real (double value)
- Asn1Real (bool floatType)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override void Decode (Asn1OerDecodeBuffer buffer)

  Decode a REAL value, encoded according to OER, into this object.
- void DecodeDouble (Asn1OerDecodeBuffer buffer)

  Decode a REAL value, encoded according to OER in double precision format.
- void DecodeSingle (Asn1OerDecodeBuffer buffer)

  Decode a REAL value, encoded according to OER in single precision format.

- virtual void DecodeXER (System.String buffer, System.String attrs, bool decodingElemName, bool modifiedEncodings)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix, bool asText)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)

  <summary> This method encodes an ASN.1 real value using the XML Encoding as specified in the W3C XML schema standard(asn2xsd).
- void EncodeDouble (Asn1OerEncodeBuffer buffer)

  Encode this REAL value, according to OER, in double precision format, into the buffer.
- void EncodeSingle (Asn1OerEncodeBuffer buffer)
- virtual void EncodeValue (Asn1XmlEncoder buffer)
- virtual bool Equals (double value)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- override System.String ToString ( )

- void Decode (Asn1DecodeBuffer buffer, int length, int baseflag)

  This method decodes the content octets of an ASN.1 REAL value into this object, where the REAL was encoded as for BER and the length is taken to be as given.
- void Decode (Asn1PerDecodeBuffer buffer, int baseflag)

- void DecodeXER (String buffer, bool modifiedEncodings)
- void SetNumericReal (String numericValue)
- byte [] ToBER ( )

- static System.String NormalizedRealValueToString (double value)

- static void Skip (Asn1PerDecodeBuffer buffer)

- static Asn1Real ( )
- static int TrailingZerosCnt (long w)

Detailed Description

This class represents the ASN.1 REAL built-in type.

Definition at line 36 of file Asn1Real.cs

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

- Asn1Real.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 9).

Definition at line 42 of file Asn1Real.cs

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

- Asn1Real.cs

Member Data Documentation

double mValue

This public member variable is where the double value is stored. This is the value that is encoded when one of the encode methods is called. It is also where the decoded result is stored when a Decode method is called.

Definition at line 70 of file Asn1Real.cs

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

- Asn1Real.cs

Asn1Real ()

The default constructor sets the double value to zero.

Asn1Real (double value)

This constructor creates an REAL object from a double value.

Table 3.457. Parameters

value	double value

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 REAL value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.458. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes ASN.1 REAL value using the Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is stored in the public member 'mValue' in this object.

Table 3.459. Parameters

buffer	PER Decode message buffer object

override void Decode (Asn1OerDecodeBuffer buffer)

This method applies to unconstrained REAL values and REAL values that are constrained but not meeting the requirements for encoding using IEEE 754 single or double precision format.

Table 3.460. Parameters

buffer	

Table 3.461. Exceptions

IOException	

void DecodeDouble (Asn1OerDecodeBuffer buffer)

Table 3.462. Parameters

buffer	

Table 3.463. Exceptions

IOException	

void DecodeSingle (Asn1OerDecodeBuffer buffer)

Table 3.464. Parameters

buffer	

Table 3.465. Exceptions

IOException	

virtual void DecodeXER (System.String buffer, System.String attrs, bool decodingElemName, bool modifiedEncodings)

This method decodes an ASN.1 real value using XER.

Table 3.466. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag
decodingElemName	Pass true if you the ASN.1 value being decoded was encoded as an empty element and buffer is the element name. Such an encoding occurs for the special real values under basic-XER, canonical-XER, and extended-XER without GLOBAL-DEFAULTS MODIFIED-ENCODINGS present.
modifiedEncodings	Pass TRUE if decoding under extended-XER with GLOBAL-DEFAULTS MODIFIED-ENCODINGS present.

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 real value using the XML schema encoding rules(asn2xsd).

Table 3.467. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 REAL value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.468. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes ASN.1 REAL value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.469. Parameters

buffer	PER Encode message buffer object

override void Encode (Asn1OerEncodeBuffer buffer)

Encode this REAL value, according to OER, into the buffer. This method applies to unconstrained REAL values and REAL values that are constrained but not meeting the requirements for encoding using IEEE 754 single or double precision format.

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 real value using the XML encoding rules (XER).

Table 3.470. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 real value according to Obj-Sys encoding rules. The value is encoded as text.

Table 3.471. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace value

void Encode (Asn1XmlEncoder buffer, String elemName, String nsPrefix, bool asText)

This method encodes an ASN.1 real value according to XER encoding rules. It is for use with extended-XER.

Table 3.472. Parameters

buffer	Encode message buffer object
elemName	Element name
asText	If TRUE, encode special values as text. Otherwise, special values are encoded as empty elements.

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 real value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.473. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.474. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes ASN.1 REAL value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.475. Parameters

outs	PER Output Stream object

Table 3.476. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)

Table 3.477. Parameters

buffer	Encode message buffer object
attrName	Attribute name

void EncodeDouble (Asn1OerEncodeBuffer buffer)

Table 3.478. Parameters

buffer	

Table 3.479. Exceptions

IOException	

void EncodeSingle (Asn1OerEncodeBuffer buffer)

Encode this REAL value, according to OER, in single precision format, into the buffer.

virtual void EncodeValue (Asn1XmlEncoder buffer)

This method encodes an ASN.1 real value using the XML encoding (non-XER).

Table 3.480. Parameters

buffer	Encode message buffer object

virtual bool Equals (double value)

This method compares this REAL value to the given value for equality.

Table 3.481. Parameters

value	The double value to compare with the current Object.

Returns: . `true` if the specified double value is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

Determines whether the specified Object is equal to the current Object.

Table 3.482. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1Real.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

override System.String ToString ()

This method will return a string representation of the REAL value. The format is the ASN.1 value format for this type..

Returns: . Stringified representation of the value

void Decode (Asn1DecodeBuffer buffer, int length, int baseflag)

Note that this method is used for OER also, since OER uses the same content octets as BER, at least in certain cases.

Table 3.483. Parameters

buffer	Decode message buffer object
length	Length of contents
baseflag	ANY_BASE: encoding may use any base ANY_BINARY: encoding may any of bases 2, 8, 16 BASE_2_ONLY: encoding may only use base 2

void Decode (Asn1PerDecodeBuffer buffer, int baseflag)

This method decodes ASN.1 REAL value using the Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is stored in the public member 'mValue' in this object.

Table 3.484. Parameters

buffer	PER Decode message buffer object
baseflag	ANY_BASE: encoding may use base 2 or base 10 BASE_2_ONLY: encoding may only use base 2

void DecodeXER (String buffer, bool modifiedEncodings)

Decode an ASN.1 real value. This method implements behavior that is common to XER and OSys-XER. basic-XER: Only invoke this method to decode non-special reals. Decoding is lax in that we allow any number of leading zeros in the exponent. canonical-XER: Decoding is lax. We do not enforce the restrictions on real value representations given for canonical-XER. extended-XER: Use the modifiedEncodings parameter to indicate whether special reals may appear as text or not. Decoding is lax in that regardless of the value passed for modifiedEncodings, we allow leading zeros in the exponent. OSys-XER: pass true for modifiedEncodings

Table 3.485. Parameters

modifiedEncodings	true if special real values are encoded as text

void SetNumericReal (String numericValue)

Assign the value of this object based on the given numeric real value. The method may be lax depending on the encoding rules being used.

- we allow any number of leading zeros in the exponent (useful for EXER which allows that in some cases and for OSys-XER), which should not be allowed for JSON or basic-XER
- If used for canonical-XER, which places some restrictions on real value representation, those restrictions will not be enforced here.

byte [] ToBER ()

This method returns the BER contents octets for the encoding of this ASN.1 REAL value. This treats the real value as being a value having base 2, and encode using base 2.

Returns: . the contents octets

static System.String NormalizedRealValueToString (double value)

This method will return a string representation of the normalized REAL value.

The output format is value format [-]X.XXXXE[-]XXX.

The format is the ASN.1 value format for this type. This means it is a "NumericRealValue" as defined in X.680. Additionally, if there is a leading minus sign, there will be no whitespace between it and the first digit of the integer part, making it also an "XMLNumericRealValue".

Table 3.486. Parameters

value	value to be normalized and stringified.

Returns: . the string as described above

static void Skip (Asn1PerDecodeBuffer buffer)

This method skips a REAL value using the Packed Encoding Rules (PER). The length and contents components are skipped.

Table 3.487. Parameters

buffer	PER Decode message buffer object

Com::Objsys::Asn1::Runtime::Asn1Real10 class Reference

- static new readonly Asn1Tag _TAG

Private Attributes

- Asn1RunTime rt

- Asn1Real10 ( )
- Asn1Real10 (System.String data)
- void ConvertToDecimal ( )
- void ConvertToNR3Form (bool cxerForm)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override void Decode (Asn1OerDecodeBuffer buffer)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- virtual int Encode (Asn1DerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1OerEncodeBuffer buffer)

  This method encodes an ASN.1 REAL value, whose base is 10, according to OER.

- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1CerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)
- byte GetNumberForm ( )
- static byte GetNumberForm (System.String value)
- static void Skip (Asn1PerDecodeBuffer buffer)
- static Asn1Real10 ( )

Detailed Description

This class represents the ASN.1 REAL BASE 10 built-in type.

Definition at line 33 of file Asn1Real10.cs

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

- Asn1Real10.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 9).

Definition at line 37 of file Asn1Real10.cs

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

- Asn1Real10.cs

Asn1Real10 ()

The default constructor sets the real10 value to zero.

Asn1Real10 (System.String data)

This constructor creates an real10 object from a string value.

Table 3.488. Parameters

data	String value

void ConvertToDecimal ()

This method convert the contained real10 value to XML decimal. Result number placed in mStringBuffer field.

void ConvertToNR3Form (bool cxerForm)

This method convert the contained real10 value to NR3 form. NR3 form number placed in mStringBuffer field.

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 real10 value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.489. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an real10 value using the Packed Encoding Rules (PER). The length and contents components of the message are decoded. The decoded result is stored in the public member 'value' in this object.

Table 3.490. Parameters

buffer	PER Decode message buffer object

Returns: . void. Decoded result is stored in the 'value' member variable and can be accessed using '<object>.value'.

override void Decode (Asn1OerDecodeBuffer buffer)

This method decodes an ASN.1 REAL value, having base 10, that was encoded according to OER.

Table 3.491. Parameters

buffer	Decode message buffer object

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 real10 value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.492. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

virtual int Encode (Asn1DerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 real10 value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Distinguished Encoding Rules (DER).

Table 3.493. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an real10 value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.494. Parameters

buffer	PER Encode message buffer object

Returns: . Length of component or negative status value

override void Encode (Asn1OerEncodeBuffer buffer)

Table 3.495. Parameters

buffer	Encode message buffer object
explicit	Flag indicating explicit tagging should be done

Returns: . Length of component or negative status value

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 real10 value using the XML encoding rules (XER).

Table 3.496. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 real10 value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.497. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace value

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 real10 value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.498. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

<throws> IOException Any exception thrown by the underlying OutputStream. </throws> <throws> Asn1Exception Thrown, if operation is failed. </throws>

override void Encode (Asn1CerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 real10 value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Canonical Encoding Rules (CER).

Table 3.499. Parameters

outs	CER Output Stream object

explicitTagging	Flag indicating explicit tagging should be done

<throws> IOException Any exception thrown by the underlying OutputStream. </throws> <throws> Asn1Exception Thrown, if operation is failed. </throws>

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 real10 value using the Packed Encoding Rules (PER). The length and contents components of the message are encoded.

Table 3.500. Parameters

outs	PER Encode message buffer object

<throws> IOException Any exception thrown by the Asn1PerOutputStream. </throws> <throws> Asn1Exception Thrown, if operation is failed. </throws>

override void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)

This method encodes an ASN.1 real10 value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.501. Parameters

buffer	Encode message buffer object
elemName	Element name
attribute	Element attribute value

byte GetNumberForm ()

This method calculates the number form of the contained real10 value.

Table 3.502. Parameters

value	Real10 value in which to count bits.

Returns: . Number form.

static byte GetNumberForm (System.String value)

This method calculates the number form of an real10 value.

Table 3.503. Parameters

value	Real10 value in which to count bits.

Returns: . Number form.

static void Skip (Asn1PerDecodeBuffer buffer)

This method skips a REAL value using the Packed Encoding Rules (PER). The length and contents components are skipped.

Table 3.504. Parameters

buffer	PER Decode message buffer object

Com::Objsys::Asn1::Runtime::Asn1RelativeOID class Reference

- static Asn1RunTime rt
- new static readonly Asn1Tag _TAG
- Asn1RelativeOID ( )
- Asn1RelativeOID (int [] value)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- override void Decode (Asn1OerDecodeBuffer buffer)
- override void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- static new void Skip (Asn1PerDecodeBuffer buffer)

- static Asn1RelativeOID ( )
- override void Validate ( )

Detailed Description

This is a container class for holding the components of an ASN.1 relative object identifier value.

Definition at line 35 of file Asn1RelativeOID.cs

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

- Asn1RelativeOID.cs

new static readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 13).

Definition at line 45 of file Asn1RelativeOID.cs

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

- Asn1RelativeOID.cs

Asn1RelativeOID ()

This constructor creates an empty object identifier that can be used in a Decode method call to receive an OID value.

Asn1RelativeOID (int[] value)

This constructor initializes the object identifier from the given array of integer subidentifier values.

Table 3.505. Parameters

value	Array of subidentifiers

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 relative object identifier value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.506. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 relative object identifier value using the packed encoding rules (PER).

Table 3.507. Parameters

buffer	Decode message buffer object

override void Decode (Asn1OerDecodeBuffer buffer)

Decode an ASN.1 RELATIVE-OID that was encoded according to the Octet Encoding Rules (OER).

override void DecodeXER (System.String buffer, System.String attrs)

This method decodes an ASN.1 RELATIVE-OID value using the XML encoding rules (XER).

Table 3.508. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 RELATIVE-OID value using the XML schema encoding rules(asn2xsd).

Table 3.509. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 relative object identifier value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.510. Parameters

buffer	Encode message buffer object

explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length of encoded component in octets

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 relative object identifier value using the packed encoding rules (PER).

The value to be encoded is stored in the `mValue` public member variable within this class.

Table 3.511. Parameters

buffer	Encode message buffer object

override void Encode (Asn1OerEncodeBuffer buffer)

This method encodes an ASN.1 RELATIVE-OID according to Octet Encoding Rules (OER).

Table 3.512. Parameters

buffer	Encode message buffer object

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 RELATIVE-OID value using the XML encoding rules (XER).

Table 3.513. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 RELATIVE-OID value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.514. Parameters

buffer	Encode message buffer object
elemName	Element name

nsPrefix	Element namespace value

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 object identifier value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.515. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.516. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 relative object identifier value using the packed encoding rules (PER).

The value to be encoded is stored in the `mValue` public member variable within this class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.517. Parameters

outs	PER Output Stream object

Table 3.518. Exceptions

Asn1Exception	Thrown, if operation is failed.

static new void Skip (Asn1PerDecodeBuffer buffer)

This method skips an ASN.1 relative object identifier value using the packed encoding rules (PER).

Table 3.519. Parameters

buffer	Decode message buffer object

override void Validate ()

Do some minimal validation. Subclasses may override this to adjust for their validation rules (e.g. Asn1RelativeOID overrides this to be more lax). Minimally, the implementation should enforce that value is not null and that there is at least one arc.

Table 3.520. Exceptions

Asn1InvalidObjectIDException	if validation fails

Com::Objsys::Asn1::Runtime::Asn1SeqOrderException class Reference

- Asn1SeqOrderException ( )

Detailed Description

This class defines the 'ASN.1 sequence order' exception that is thrown when an element is received in a SEQUENCE construct that is not in the correct order..

Definition at line 36 of file Asn1SeqOrderException.cs

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

- Asn1SeqOrderException.cs

Asn1SeqOrderException ()

This constructor creates an exception object with a textual message describing the element that was received out-of-order.

Com::Objsys::Asn1::Runtime::Asn1Status class Reference

Public Attributes

- const int INDEFLEN

Detailed Description

This class defines common constants used in the run-time and generated code. Note that all error reporting in the C# version is done via exceptions. Therefore, there are very few status values defined in the class.

Definition at line 34 of file Asn1Status.cs

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

- Asn1Status.cs

Member Data Documentation

const int INDEFLEN

This constant indicates an indefinite length field was parsed

Definition at line 36 of file Asn1Status.cs

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

- Asn1Status.cs

Com::Objsys::Asn1::Runtime::Asn1T61String class Reference

- static new readonly Asn1Tag _TAG

- Asn1T61String ( )
- Asn1T61String (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

- static Asn1T61String ( )

Detailed Description

This is a container class for holding the components of an ASN.1 teletex (T61) string value.

Definition at line 35 of file Asn1T61String.cs

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

- Asn1T61String.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 20).

Definition at line 41 of file Asn1T61String.cs

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

- Asn1T61String.cs

Asn1T61String ()

The default constructor creates an empty string object.

Asn1T61String (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string.

Table 3.521. Parameters

data	String value of T61String

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 T61String value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.522. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 T61String type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.523. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 T61String value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.524. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.525. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1Tag class Reference

Public Attributes

- const short APPL
- const short Bit8Mask
- const short ClassMask
- const short CONS
- const short CTXT
- const bool EXPL
- const short EXTIDCODE
- const short FormMask
- const short IDMask
- const bool IMPL
- const short L7BitsMask
- short mClass
- short mForm
- int mIDCode
- const short PRIM
- const short PRIV
- const short UNIV

- static readonly Asn1Tag ENUM
- static readonly Asn1Tag EOC
- static readonly Asn1Tag SEQUENCE
- static readonly Asn1Tag SET
-
- Asn1Tag ( )
- Asn1Tag (short tagclass, short form, int idCode)
- virtual bool Equals (short tagclass, short form, int idCode)
- bool Equals (Asn1Tag tag)
- virtual bool IsEOC ( )
- override System.String ToString ( )
- static Asn1Tag ( )

Detailed Description

This is a container class for holding the components of an ASN.1 tag value.

Definition at line 36 of file Asn1Tag.cs

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

- Asn1Tag.cs

Member Data Documentation

const short APPL

Mask value for an APPLICATION tag

Definition at line 51 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short Bit8Mask

Bit 8 (MSB) octet mask value

Definition at line 107 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short ClassMask

Mask value to mask the class bits from a tag

Definition at line 63 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short CONS

Mask value for CONSTRUCTED form

Definition at line 71 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short CTXT

Mask value for a context-specific tag

Definition at line 55 of file Asn1Tag.cs

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

- Asn1Tag.cs

const bool EXPL

This specifies that explicit tagging should be used.

Definition at line 39 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short EXTIDCODE

Mask value for extended tag identifier indicator

Definition at line 79 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short FormMask

Mask value to mask the form bit from a tag

Definition at line 75 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short IDMask

Mask value to mask the tag ID bits from a tag

Definition at line 83 of file Asn1Tag.cs

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

- Asn1Tag.cs

const bool IMPL

This specifies that implicit tagging should be used.

Definition at line 43 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short L7BitsMask

Lower 7 bits octet mask value

Definition at line 111 of file Asn1Tag.cs

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

- Asn1Tag.cs

short mClass

Tag class value (UNIV, APPL, CTXT, or PRIV)

Definition at line 116 of file Asn1Tag.cs

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

- Asn1Tag.cs

short mForm

Tag form value (PRIM or CONS)

Definition at line 121 of file Asn1Tag.cs

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

- Asn1Tag.cs

int mIDCode

Tag ID code

Definition at line 126 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short PRIM

Mask value for PRIMITIVE form

Definition at line 67 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short PRIV

Mask value for a PRIVATE tag

Definition at line 59 of file Asn1Tag.cs

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

- Asn1Tag.cs

const short UNIV

Mask value for a UNIVERSAL tag

Definition at line 47 of file Asn1Tag.cs

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

- Asn1Tag.cs

readonly Asn1Tag ENUM

ASN.1 ENUMERATED type tag value

Definition at line 89 of file Asn1Tag.cs

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

- Asn1Tag.cs

readonly Asn1Tag EOC

ASN.1 end-of-contents (EOC) type tag value

Definition at line 93 of file Asn1Tag.cs

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

- Asn1Tag.cs

readonly Asn1Tag SEQUENCE

ASN.1 SEQUENCE type tag value

Definition at line 97 of file Asn1Tag.cs

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

- Asn1Tag.cs

readonly Asn1Tag SET

ASN.1 SET type tag value

Definition at line 101 of file Asn1Tag.cs

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

- Asn1Tag.cs

Asn1Tag ()

The default constructor initializes all fields to zero

Asn1Tag (short tagclass, short form, int idCode)

This constructor initializes all fields to the given values

Table 3.526. Parameters

tagclass	Tag class value (UNIV, APPL, CTXT, or PRIV)
form	Tag form value (PRIM or CONS)
idCode	Tag identifier code

virtual bool Equals (short tagclass, short form, int idCode)

This method compares this tag with the given tag value for equality.

Table 3.527. Parameters

tagclass	Tag class value (UNIV, APPL, CTXT, or PRIV)
form	Tag form value (PRIM or CONS)

idCode	Tag identifier code

Returns: . `true` if the specified Tags are equal; otherwise, `false`.

bool Equals (Asn1Tag tag)

This method compares this tag with the given tag value for equality.

Table 3.528. Parameters

tag	Asn1Tag object to which this tag is to be compared

Returns: . `true` if the specified Tags are equal; otherwise, `false`.

virtual bool IsEOC ()

This method tests if the tag is an end-of-contents (EOC) tag.

Returns: . True if tag is an EOC.

override System.String ToString ()

This method will return a formatted string representing the tag value. The form is "[<class> <ID>]" (i.e. the ASN.1 standard syntax for a tag value).

Returns: . Formatted tag string

Com::Objsys::Asn1::Runtime::Asn1Time class Reference

Public Attributes

- const int Apr
- const int April
- const int Aug
- const int August
- const int Dec
- const int December
- const int Feb
- const int February
- const int Jan
- const int January

Month constants.

- const int Jul
- const int July
- const int Jun
- const int June
- const int Mar
- const int March
- const int May
- const int Nov
- const int November
- const int Oct
- const int October
- const int Sep
- const int September

- static readonly Asn1Tag TAG

  <summary> The TAG constant describes the universal tag for this data type (UNIVERSAL 14).

- Asn1Time ( )

  <summary> The default constructor creates an empty string object.</summary>

- Asn1Time (System.String data)

  <summary> This version of the constructor can be used to set the string <B>value</B> member variable to the given string.

- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

  <summary> This method decodes an ASN.1 TIME string value including the UNIVERSAL tag value and length if explicit tagging is specified.

- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

  <summary> This method encodes an ASN.1 TIME string type.

- override void Encode (Asn1BerOutputStream outstr, bool explicitTagging)

  <summary> This method encodes and writes to the stream an ASN.1 TIME

- static Asn1Time ( )

Detailed Description

This class is used for all TIME types that are not one of the useful time types (viz. DATE, DATE-TIME, TIME-OF-DAY, DURATION).

Definition at line 38 of file Asn1Time.cs

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

- Asn1Time.cs

Member Data Documentation

const int January

These were defined in what used to be Asn1Time and was renamed to Asn1AbstractTime. They are here to allow backward compatibility so users don't have to modify their code to use Asn1AbstractTime to refer to these constants.

Definition at line 45 of file Asn1Time.cs

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

- Asn1Time.cs

readonly Asn1Tag TAG

Definition at line 72 of file Asn1Time.cs

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

- Asn1Time.cs

Asn1Time ()

Asn1Time (System.String data)

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

Table 3.529. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.530. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outstr, bool explicitTagging)

value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Table 3.531. Parameters

outstr	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Com::Objsys::Asn1::Runtime::Asn1TraceHandler class Reference

- System.IO.StreamWriter mPrintStream

- Asn1TraceHandler ( )
- Asn1TraceHandler (System.IO.StreamWriter ps)
- virtual void Characters (System.String svalue, short typeCode)
- virtual void EndElement (System.String name, int index)
- virtual void StartElement (System.String name, int index)

Detailed Description

This class is a standard named event handler for printing the data in an encoded message in a human-readable format. Note that this handler will work with data encoded using any of the encoding rules (BER, DER, or PER).

Definition at line 34 of file Asn1TraceHandler.cs

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

- Asn1TraceHandler.cs

Asn1TraceHandler ()

This constructor sets the output stream to standard output.

Asn1TraceHandler (System.IO.StreamWriter ps)

This constructor sets the output stream to the given StreamWriter.

Table 3.532. Parameters

ps	StreamWriter object

virtual void Characters (System.String svalue, short typeCode)

The characters callback method is invoked when content (primitive data) is encountered. A stringified representation of the parsed value is returned.

Table 3.533. Parameters

svalue	Stringified representation of the parsed value. The representation will be in ASN.1 value format.
typeCode	Identifier specifying the type of the parsed data variable. The enumerated list of values that might appear here is provided in the the Asn1Type class (see the documentation on this class for a full list of the names).

virtual void EndElement (System.String name, int index)

The endElement callback method is invoked when the end of an element within a constructed type (SEQUENCE, SET, SEQUENCE OF, SET OF, or CHOICE) is detected.

Table 3.534. Parameters

name	Name of the parsed element.
index	Index of element in array. Only used for SEQUENCE OF or SET OF elements. Set to -1 for all others.

virtual void StartElement (System.String name, int index)

The StartElement callback method is invoked when the start of an element within a constructed type (SEQUENCE, SET, SEQUENCE OF, SET OF, or CHOICE) is encountered.

Table 3.535. Parameters

name	Name of the parsed element.
index	Index of element in array. Only used for SEQUENCE OF or SET OF elements. Set to -1 for all others.

Com::Objsys::Asn1::Runtime::Asn1Type class Reference

Public Attributes

- const short BIT_STRING
- const short BMPString
- const short BOOLEAN
- const short DATE
- const short DATE_TIME

  <summary> DATE-TIME type code = 33
- const short DURATION

  <summary> DURATION type code = 34
- const short ENUMERATED
- const short EOC
- const short EXTERNAL
- const short GeneralString
- const short GeneralTime
- const short GraphicString
- const short IA5String
- const short INTEGER
- const short NULL
- const short NumericString
- const short OBJECT_IDENTIFIER
- const short ObjectDescriptor
- const short OCTET_STRING
- const short OID_IRI

  <summary> OID-IRI type code = 35 </p>
- const short OpenType
- const short PrintableString
- const short REAL

- const short RELATIVE_OID_IRI
- const short RelativeOID
- const short SEQUENCE
- const short SET
- const short T61String
- const short TeletexString
- const short TIME
- const short TIME_OF_DAY

  <summary> TIME-OF-DAY type code = 32
- const short UniversalString
- const short UTCTime
- const short UTF8String
- const short VideotexString
- const short VisibleString

- static readonly Asn1Tag _TAG

-
-

Private Attributes

- String mNonParameterizedTypeName
- bool mOpenType

- void _SetKey (byte [] rtkey)
- virtual void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- virtual void Decode (Asn1BerDecodeBuffer buffer)
- virtual void Decode (Asn1OerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (System.Object reader, System.String xmlURI)
- virtual void Decode (System.Object reader, System.IO.Stream byteStream)
- virtual void Decode (Asn1MderDecodeBuffer buffer)
- virtual void DecodeXML (String buffer, String attrs)

- virtual int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- virtual int Encode (Asn1BerEncodeBuffer buffer)
- virtual void Encode (Asn1OerEncodeBuffer buffer)
- virtual void Encode (Asn1PerEncodeBuffer buffer)
- virtual void Encode (Asn1XerEncoder buffer)
- virtual void Encode (Asn1XerEncoder buffer, System.String elemName)
- virtual void Encode (Asn1XmlEncoder buffer)
- virtual void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- virtual void Encode (Asn1XmlEncodeBuffer buffer)
- virtual void Encode (Asn1MderOutputStream buffer)
- virtual void Encode (Asn1MderOutputStream buffer, bool useCachedLength)
- virtual void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- virtual void Encode (Asn1CerOutputStream outs, bool explicitTagging)
- virtual void Encode (Asn1PerOutputStream outs)
- void EncodeAsOpenType (Asn1OerEncodeBuffer buffer)
- virtual void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)
- virtual bool Equals (Asn1Type obj)
- String GetNonParameterizedTypeName ( )
- virtual void Indent (System.IO.TextWriter outs, int level)
- virtual bool IsOpenType ( )
- virtual bool MatchTypeName (System.String typeName)
- virtual void Pdiag (System.String s)
- virtual void Print (System.IO.TextWriter outs, System.String varName, int level)
- virtual void Print (System.String varName)
- void SetNonParameterizedTypeName (String value)
- virtual void SetOpenType ( )

- static void _SetKey2 (byte [] rtkey)
- static void _SetLicLocation (String path)
- static Asn1Type Decode (Asn1BerDecodeBuffer buffer, Asn1OpenTypeField openTypeField, bool explicitTag, int implicitLength)

- static Asn1Type Decode (Asn1OerDecodeBuffer buffer, Asn1OpenTypeField openTypeField)
- static Asn1Type Decode (Asn1PerDecodeBuffer buffer, Asn1OpenTypeField openTypeField)
- static System.String GetTypeName (short typeCode)
- static int MatchTag (Asn1BerDecodeBuffer buffer, short tagClass, short tagForm, int tagIDCode)
- static int MatchTag (Asn1BerDecodeBuffer buffer, Asn1Tag tag)

Detailed Description

This is the base class for all ASN.1 built-in types.

Definition at line 34 of file Asn1Type.cs

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

- Asn1Type.cs

Member Data Documentation

const short BIT_STRING

BIT_STRING type code = 3

Definition at line 48 of file Asn1Type.cs

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

- Asn1Type.cs

const short BMPString

BMPString type code = 30

Definition at line 123 of file Asn1Type.cs

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

- Asn1Type.cs

const short BOOLEAN

BOOLEAN type code = 1

Definition at line 42 of file Asn1Type.cs

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

- Asn1Type.cs

const short DATE

DATE type code = 31

Definition at line 126 of file Asn1Type.cs

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

- Asn1Type.cs

const short ENUMERATED

ENUMERATED type code = 10

Definition at line 69 of file Asn1Type.cs

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

- Asn1Type.cs

const short EOC

EOC type code = 0

Definition at line 39 of file Asn1Type.cs

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

- Asn1Type.cs

const short EXTERNAL

EXTERNAL type code = 8

Definition at line 63 of file Asn1Type.cs

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

- Asn1Type.cs

const short GeneralString

GeneralString type code = 27

Definition at line 117 of file Asn1Type.cs

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

- Asn1Type.cs

const short GeneralTime

GeneralTime type code = 24

Definition at line 108 of file Asn1Type.cs

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

- Asn1Type.cs

const short GraphicString

GraphicString type code = 25

Definition at line 111 of file Asn1Type.cs

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

- Asn1Type.cs

const short IA5String

IA5String type code = 22

Definition at line 102 of file Asn1Type.cs

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

- Asn1Type.cs

const short INTEGER

INTEGER type code = 2

Definition at line 45 of file Asn1Type.cs

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

- Asn1Type.cs

const short NULL

NULL type code = 5

Definition at line 54 of file Asn1Type.cs

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

- Asn1Type.cs

const short NumericString

NumericString type code = 18

Definition at line 87 of file Asn1Type.cs

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

- Asn1Type.cs

const short OBJECT_IDENTIFIER

OBJECT_IDENTIFIER type code = 6

Definition at line 57 of file Asn1Type.cs

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

- Asn1Type.cs

const short ObjectDescriptor

ObjectDescriptor type code = 7

Definition at line 60 of file Asn1Type.cs

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

- Asn1Type.cs

const short OCTET_STRING

OCTET_STRING type code = 4

Definition at line 51 of file Asn1Type.cs

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

- Asn1Type.cs

const short OpenType

OpenType type code = 99

Definition at line 144 of file Asn1Type.cs

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

- Asn1Type.cs

const short PrintableString

PrintableString type code = 19

Definition at line 90 of file Asn1Type.cs

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

- Asn1Type.cs

const short REAL

REAL type code = 9

Definition at line 66 of file Asn1Type.cs

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

- Asn1Type.cs

const short RELATIVE_OID_IRI

RELATIVE-OID-IRI type code = 35

Definition at line 141 of file Asn1Type.cs

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

- Asn1Type.cs

const short RelativeOID

RELATIVE_OID type code = 13

Definition at line 75 of file Asn1Type.cs

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

- Asn1Type.cs

const short SEQUENCE

SEQUENCE type code = 16

Definition at line 81 of file Asn1Type.cs

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

- Asn1Type.cs

const short SET

SET type code = 17

Definition at line 84 of file Asn1Type.cs

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

- Asn1Type.cs

const short T61String

T61String type code = TeletexString

Definition at line 96 of file Asn1Type.cs

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

- Asn1Type.cs

const short TeletexString

TeletexString type code = 20

Definition at line 93 of file Asn1Type.cs

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

- Asn1Type.cs

const short TIME

TIME type code = 14

Definition at line 78 of file Asn1Type.cs

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

- Asn1Type.cs

const short UniversalString

UniversalString type code = 28

Definition at line 120 of file Asn1Type.cs

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

- Asn1Type.cs

const short UTCTime

UTCTime type code = 23

Definition at line 105 of file Asn1Type.cs

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

- Asn1Type.cs

const short UTF8String

UTF8String type code = 12

Definition at line 72 of file Asn1Type.cs

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

- Asn1Type.cs

const short VideotexString

VideotexString type code = 21

Definition at line 99 of file Asn1Type.cs

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

- Asn1Type.cs

const short VisibleString

VisibleString type code = 26

Definition at line 114 of file Asn1Type.cs

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

- Asn1Type.cs

readonly Asn1Tag _TAG

Will hold tag for possible definitions

Definition at line 147 of file Asn1Type.cs

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

- Asn1Type.cs

Member Data Documentation

String mNonParameterizedTypeName

This type's NonParameterizedTypeName, if set.

Definition at line 710 of file Asn1Type.cs

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

- Asn1Type.cs

void _SetKey (byte[] rtkey)

This method is used with the limited run-time to set a run-time key value generated by the compiler to allow the run-time to operate on the licensed hosts. This is not used in the unlimited redistribution versions.

Table 3.536. Parameters

rtkey	Run-time key generated by ASN1C

virtual void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method is used to Decode a message that is encoded in BER or DER format.

Table 3.537. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating explicit tag should be parsed from the encoded type.
implicitLength	Length of the contents field (only required if explicitTagging is false).

virtual void Decode (Asn1BerDecodeBuffer buffer)

This method is used to Decode a message that is encoded in BER or DER format. This version of the method sets tagging to explicit (Asn1Tag.EXPL) and implicit length to zero.

Table 3.538. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1OerDecodeBuffer buffer)

This method decodes an ASN.1 value into this object, following the Octet Encoding Rules (OER).

This method MUST be overridden by all subclasses, except for Asn1Enumerated and subclasses thereof, in order to implement the correct decode behavior for the corresponding ASN.1 type.

This method MUST NOT be invoked on an Asn1Enumerated object. Such objects are immutable and cannot be decoded into.

This method is used polymorphically when table constraint code is generated and Asn1Type is used as the declared type for an open type.

virtual void Decode (Asn1PerDecodeBuffer buffer)

This method is the base implementation of the standard Packed Encoding Rules (PER) Decode method. It throws an exception because it should never be invoked. Inherited class implementes this method in Compiler generated code.

Table 3.539. Parameters

buffer	PER Encode message buffer object

Table 3.540. Exceptions

Asn1Exception	if invoked directly

virtual void Decode (System.Object reader, System.String xmlURI)

This method declaration is the signature of the standard XML Encoding Rules (XER) Decode method.

Also throws any IO exception from the parser, possibly from a byte stream or character stream supplied by the application.

Table 3.541. Parameters

reader	XML reader object
xmlURI	URI of a source

Table 3.542. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Decode (System.Object reader, System.IO.Stream byteStream)

This method declaration is the signature of the standard XML Encoding Rules (XER) Decode method.

Also throws any IO exception from the parser, possibly from a byte stream or character stream supplied by the application.

Table 3.543. Parameters

reader	XML reader object
byteStream	Input byte stream object

Table 3.544. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Decode (Asn1MderDecodeBuffer buffer)

This method decodes this object's data from an MDER encoding. The implementation for this class throws an exception. When generating MDER code, subclasses will override this implementation.

Table 3.545. Parameters

buffer	MDER Decode message buffer object

<throws> IOException Any exception thrown by the underlying stream. </throws> <throws> Asn1Exception Thrown, if operation is failed. </throws>

virtual void DecodeXML (String buffer, String attrs)

This method decodes the XML content of a simple type.

Table 3.546. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

virtual int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method is used to encode this data type in BER or DER format.

Table 3.547. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tag should be added to the encoded type.

Returns: . Length of component or negative status value

virtual int Encode (Asn1BerEncodeBuffer buffer)

This method is used to encode a message in BER or DER format. This version of the method sets tagging to explicit (Asn1Tag.EXPL).

Table 3.548. Parameters

buffer	Decode message buffer object

Returns: . Length of component or negative status value

virtual void Encode (Asn1OerEncodeBuffer buffer)

This method encodes the ASN.1 value represented by this object, following the Octet Encoding Rules (OER).

This method MUST be overridden by all subclasses in order to implement the correct encode behavior for the corresponding type.

This method is used polymorphically when table constraint code is generated and Asn1Type is used as the declared type for an open type.

virtual void Encode (Asn1PerEncodeBuffer buffer)

This method is the base implementation of the standard Packed Encoding Rules (PER) encode method. It throws an exception because it should never be invoked. Inherited class implementes this method in Compiler generated code.

Table 3.549. Parameters

buffer	PER Encode message buffer object

Table 3.550. Exceptions

Asn1Exception	if invoked directly

virtual void Encode (Asn1XerEncoder buffer)

This method is the base implementation of the standard XML Encoding Rules (XER) encode method. This method invokes the generated method with element name set to null. This will cause the ASN.1 type name to be used as the top-level element name.

Also throws any exception thrown by the underlying stream.

Table 3.551. Parameters

buffer	XER Encode message buffer object

Table 3.552. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1XerEncoder buffer, System.String elemName)

This method is the base implementation of the standard XML Encoding Rules (XER) encode method. It throws an exception because it should never be invoked by compiler generated code.

Also throws any exception thrown by the underlying stream.

Table 3.553. Parameters

buffer	XER Encode message buffer object
elemName	XML element name of item

Table 3.554. Exceptions

Asn1Exception	if invoked directly

virtual void Encode (Asn1XmlEncoder buffer)

This method is the base implementation of the standard XML Encoding as specified in the XML schema standard(asn2xsd). This method invokes the generated method with element name and attribute name set to null. This will cause the ASN.1 type name to be used as the top-level element name.

Also throws any exception thrown by the underlying stream.

Table 3.555. Parameters

buffer	XML Encode message buffer object

Table 3.556. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method is the base implementation of the standard XML Encoding as specified in the XML schema standard(asn2xsd). It throws an exception because it should never be invoked. Inherited class implementes this method in Compiler generated code.

Also throws any exception thrown by the underlying stream.

Table 3.557. Parameters

buffer	XML Encode message buffer object
elemName	XML element name of item
nsPrefix	Element namespace value

Table 3.558. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1XmlEncodeBuffer buffer)

This method is the base implementation of the standard XML Encoding as specified in the XML schema standard(asn2xsd). It invokes the Encode() signature that accepts an implementation of the Asn1XmlEncoder interface.

Also throws any exception thrown by the underlying stream.

Table 3.559. Parameters

buffer	XML Encode message buffer object

Table 3.560. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1MderOutputStream buffer)

This method encodes this object's data to an MDER encoding. The implementation for this class invokes Encode(buffer, false). When generating MDER code, subclasses will override this implementation or else will override the Encode(Asn1MderOutputStream, bool) overload. Invoke this method if you are not certain which overload has been overridden by the subclass.

Table 3.561. Parameters

buffer	MDER Encode message buffer object

<throws> IOException Any exception thrown by the underlying stream. </throws> <throws> Asn1Exception Thrown, if operation is failed. </throws>

virtual void Encode (Asn1MderOutputStream buffer, bool useCachedLength)

This method encodes this object's data to an MDER encoding. The implementation for this class throws an exception. When generating MDER code, subclasses will override this implementation or else will override the Encode(Asn1MderOutputStream) overload. Do not invoke this function unless you are certain the subclass has overridden it.

Table 3.562. Parameters

buffer	MDER Encode message buffer object
useCachedLength	Indicates whether cached length of encoding should be used. In general, only generated code should pass true.

<throws> IOException Any exception thrown by the underlying stream. </throws> <throws> Asn1Exception Thrown, if operation is failed. </throws>

virtual void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method writes to the stream an encoded ASN.1 type value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.563. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.564. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1CerOutputStream outs, bool explicitTagging)

This method writes to the stream an encoded ASN.1 type value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Cer Encoding Rules (CER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.565. Parameters

outs	CER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.566. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs)

This method is the base implementation of the standard Packed Encoding Rules (PER) encode method using output stream. It throws an exception because it should never be invoked by compiler generated code.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.567. Parameters

outs	PER Output Stream object

Table 3.568. Exceptions

Asn1Exception	Thrown, if operation is failed.

void EncodeAsOpenType (Asn1OerEncodeBuffer buffer)

This method encodes the ASN.1 value represented by this object, following the Octet Encoding Rules (OER), as the value for the actual type of an open type (i.e., it precedes its encoding with a length).

virtual void EncodeAttribute (Asn1XmlEncoder buffer, System.String attrName)

This method is the base implementation of the standard XML Encoding as specified in the XML schema standard(asn2xsd). It throws an exception because it should never be invoked by compiler generated code.

Table 3.569. Parameters

buffer	XML Encode message buffer object
attrName	XML attribute name of item

attribute	Element attribute value

<throws> IOException Any exception thrown by the underlying stream. </throws> <throws> Asn1Exception Thrown, if operation is failed. </throws>

virtual bool Equals (Asn1Type obj)

Return true if the two objects are equal. /p> Note: in generated code, we will implement Equals(Asn1Type) rather than Equals(Object) to avoid the need to also implement HashCode. The default implementation here is to invoke Equals(Object). This is acceptable for the runtime classes, which have overridden Equals(Object).

String GetNonParameterizedTypeName ()

Return this type's NonParameterizedTypeName, if set.

virtual void Indent (System.IO.TextWriter outs, int level)

This method will indent three spaces in the given print stream. It is used by the print methods to provide a formatted output of an encoded element value.

Table 3.570. Parameters

outs	Print stream
level	Indentation level (no of spaces is 3 x this number)

virtual bool IsOpenType ()

Returns open type mode for XML encoding/decoding.

Returns: . `true` if open type mode is on.

virtual bool MatchTypeName (System.String typeName)

This method is used to check the outer level tag in an XER message to verify it matches the expected value. This method is overridden by generated code. The default implementation always returns true.

Table 3.571. Parameters

typeName	Type name to compare.

Returns: . True if name matches internal name.

virtual void Pdiag (System.String s)

This is a diagnostics print method. It is a shorthand way to invoke the Diag object's println method.

Table 3.572. Parameters

s	diagnostics message to be printed.

virtual void Print (System.IO.TextWriter outs, System.String varName, int level)

This method will format and output a primitive value to the given print stream.

Table 3.573. Parameters

outs	Print output stream
varName	Name of variable
level	Indentation level

virtual void Print (System.String varName)

This method will format and output a primitive value to the standard console output.

Table 3.574. Parameters

varName	Name of variable

void SetNonParameterizedTypeName (String value)

Set this type's NonParameterizedTypeName. The value is specified, based on the ASN.1 type, by X.680.

virtual void SetOpenType ()

Sets open type mode for XML encoding/decoding.

static void _SetKey2 (byte[] rtkey)

This method is used with the limited run-time to set a run-time key value generated by the compiler to allow the run-time to operate on the licensed hosts. This is not used in the unlimited redistribution versions. This is the static version of _SetKey.

Table 3.575. Parameters

rtkey	Run-time key generated by ASN1C

static void _SetLicLocation (String path)

This method sets a location that the runtime should check for a license file.

Table 3.576. Parameters

path	Path to check

static Asn1Type Decode (Asn1BerDecodeBuffer buffer, Asn1OpenTypeField openTypeField, bool explicitTag, int implicitLength)

Decode an open type field value from the given buffer.

Table 3.577. Parameters

buffer	Decode message buffer object
openTypeField	Provides the actual type.
explicitTag	if true, the value to be decoded is preceded by a tag and length, which must first be decoded. Otherwise, implicitLength provides the length of the value to be decoded.
implicitLength	The length of the value to be decoded if explicitTag was given as false.

static Asn1Type Decode (Asn1OerDecodeBuffer buffer, Asn1OpenTypeField openTypeField)

Decode an open type field value from the given buffer.

Table 3.578. Parameters

openTypeField	Describes the open type being decoded.

static Asn1Type Decode (Asn1PerDecodeBuffer buffer, Asn1OpenTypeField openTypeField)

Decode an open type field value from the given buffer.

Table 3.579. Parameters

buffer	Decode message buffer object
openTypeField	Describes the actual type.

static System.String GetTypeName (short typeCode)

This method will convert a type code into a type name as defined in the X.680 standard..

Table 3.580. Parameters

typeCode	Type code to be converted

Returns: . ASN.1 type name

static int MatchTag (Asn1BerDecodeBuffer buffer, short tagClass, short tagForm, int tagIDCode)

This method will compare the next parsed tag with the given tag value. If they do not match, an exception will be thrown.

Table 3.581. Parameters

buffer	Decode message buffer object
tagClass	Tag class value (UNIV, APPL, CTXT, or PRIV)
tagForm	Tag form value (PRIM or CONS)
tagIDCode	Tag identifier code

Returns: . Decoded length value

Table 3.582. Exceptions

Asn1TagMatchFailedException	Tag is not equal to expected value

static int MatchTag (Asn1BerDecodeBuffer buffer, Asn1Tag tag)

This method will compare the next parsed tag with the given tag value. If they do not match, an exception will be thrown.

Table 3.583. Parameters

buffer	Decode message buffer object
tag	Tag value to compare

Returns: . Decoded length value

Table 3.584. Exceptions

Asn1TagMatchFailedException	Tag is not equal to expected value

Com::Objsys::Asn1::Runtime::Asn1TypeIF interface Reference

- void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- void Decode (Asn1PerDecodeBuffer buffer)
- void Decode (System.Object reader, System.String xmlURI)
- void Decode (System.Object reader, System.IO.Stream byteStream)
- void Decode (Asn1MderDecodeBuffer buffer)
- int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- void Encode (Asn1PerEncodeBuffer buffer)
- void Encode (Asn1XerEncoder buffer)
- void Encode (Asn1XerEncoder buffer, System.String elemName)
- void Encode (Asn1XmlEncoder buffer)
- void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- void Encode (Asn1MderOutputStream buffer)
- void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- void Encode (Asn1PerOutputStream outs)
- bool IsOpenType ( )
- void Print (System.IO.TextWriter outs, System.String varName, int level)
- void SetOpenType ( )

Detailed Description

This is the base interface for all ASN.1 built-in types.

Definition at line 64 of file Asn1TypeIF.cs

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

- Asn1TypeIF.cs

void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method declaration is the signature of the standard Basic Encoding Rules (BER) or Distinguished Encoding Rules (DER) Decode method.

Table 3.585. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating explicit tag should be parsed from the encoded type.
implicitLength	Length of the contents field (only required if explicit is false).

void Decode (Asn1PerDecodeBuffer buffer)

This method declaration is the signature of the standard Packed Encoding Rules (PER) Decode method.

Table 3.586. Parameters

buffer	PER Encode message buffer object

void Decode (System.Object reader, System.String xmlURI)

This method declaration is the signature of the standard XML Encoding Rules (XER) Decode method.

Throws an IO exception from the parser, possibly from a byte stream or character stream supplied by the application.

Table 3.587. Parameters

reader	XML reader object
xmlURI	URI of a source

Table 3.588. Exceptions

Asn1Exception	Thrown, if operation is failed.

void Decode (System.Object reader, System.IO.Stream byteStream)

This method declaration is the signature of the standard XML Encoding Rules (XER) Decode method.

Throws an IO exception from the parser, possibly from a byte stream or character stream supplied by the application.

Table 3.589. Parameters

reader	XML reader object
byteStream	Input byte stream object

Table 3.590. Exceptions

Asn1Exception	Thrown, if operation is failed.

void Decode (Asn1MderDecodeBuffer buffer)

This method decodes this object's data from an MDER encoding. When MDER code has not been generated, classes implementing this interface may simply throw an exception.

int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method declaration is the signature of the standard Basic Encoding Rules (BER) or Distinguished Encoding Rules (DER) encode method.

Table 3.591. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tag should be added to the encoded type.

Returns: . Length of component or negative status value

void Encode (Asn1PerEncodeBuffer buffer)

This method declaration is the signature of the standard Packed Encoding Rules (PER) encode method.

Table 3.592. Parameters

buffer	PER Encode message buffer object

void Encode (Asn1XerEncoder buffer)

This method declaration is the signature of the standard XML Encoding Rules (XER) encode method. This method invokes the generated method with element name set to null. This will cause the ASN.1 type name to be used as the top-level element name.

Also throws any exception thrown by the underlying stream.

Table 3.593. Parameters

buffer	XER Encode message buffer object

Table 3.594. Exceptions

Asn1Exception	Thrown, if operation is failed.

void Encode (Asn1XerEncoder buffer, System.String elemName)

This method declaration is the signature of the standard XML Encoding Rules (XER) encode method.

Also throws any exception thrown by the underlying stream.

Table 3.595. Parameters

buffer	XER Encode message buffer object
elemName	XML element name of item

Table 3.596. Exceptions

Asn1Exception	Thrown, if operation is failed.

void Encode (Asn1XmlEncoder buffer)

This method declaration is the signature of the standard XML Encoding as specified in the XML schema standard(asn2xsd). This method invokes the generated method with element name and attribute name set to null. This will cause the ASN.1 type name to be used as the top-level element name.

Also throws any exception thrown by the underlying stream.

Table 3.597. Parameters

buffer	XML Encode message buffer object

Table 3.598. Exceptions

Asn1Exception	Thrown, if operation is failed.

void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method declaration is the signature of the standard XML Encoding as specified in the XML schema standard(asn2xsd).

Also throws any exception thrown by the underlying stream.

Table 3.599. Parameters

buffer	XML Encode message buffer object
elemName	XML element name of item
nsPrefix	Element namespace value

Table 3.600. Exceptions

Asn1Exception	Thrown, if operation is failed.

void Encode (Asn1MderOutputStream buffer)

This method encodes this object's data to an MDER encoding. When MDER code has not been generated, classes implementing this interface may simply throw an exception.

void Encode (Asn1BerOutputStream outs, bool explicit-Tagging)

This method declaration is the signature of the streaming oriented BER encode method.

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.601. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.602. Exceptions

Asn1Exception	Thrown, if operation is failed.

void Encode (Asn1PerOutputStream outs)

This method declaration is the signature of the streaming oriented PER encode method.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.603. Parameters

outs	PER Output Stream object

Table 3.604. Exceptions

Asn1Exception	Thrown, if operation is failed.

bool IsOpenType ()

Returns open type mode for XML encoding/decoding.

Returns: . `true` if open type mode is on.

void Print (System.IO.TextWriter outs, System.String varName, int level)

This method declaration is the signature of the standard print method used to print the contents of the object representing the ASN.1 type.

Table 3.605. Parameters

outs	Output print stream
varName	Name of the variable being printed
level	Indentation level

void SetOpenType ()

Sets open type mode for XML encoding/decoding.

Com::Objsys::Asn1::Runtime::Asn1UniversalString class Reference

Public Attributes

- const int BITSPERCHAR
- int [] mValue
- static new readonly Asn1Tag _TAG
- System.Text.StringBuilder mStringBuffer
-
- static readonly int [] encoding_byte
- static readonly int [] encoding_mask

- Asn1UniversalString ( )
- Asn1UniversalString (int [] value)
- Asn1UniversalString (System.String value)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)
- virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- void Decode (Asn1OerDecodeBuffer buffer, int length)
- virtual void DecodeXER (System.String buffer, System.String attrs)
- override void DecodeXML (System.String buffer, System.String attrs)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet)
- virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- virtual void Encode (Asn1OerEncodeBuffer buffer, bool withLength)
- override void Encode (Asn1XerEncoder buffer, System.String elemName)
- override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet)
- virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet, long lower, long upper)
- virtual void EncodeData (Asn1XmlXerEncoder buffer)
- override bool Equals (System.Object value)
- override int GetHashCode ( )
- override System.String ToString ( )
- bool validate (Asn1CharSet charSet)

- void ByteAlign (Asn1PerMessageBuffer buffer, int nbitsPerChar, long lower, long upper)

- void ReadSegment (Asn1BerDecodeBuffer buffer, int llen, IntHolder idx)
- void ReallocIntArray (int nint)
- void SetValue (System.String value)

- virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)
- virtual void Decode (Asn1PerDecodeBuffer buffer, int nchars, int abpc, int ubpc, Asn1CharSet charSet, int startIdx)
- virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)
- virtual void Encode (Asn1PerEncodeBuffer buffer, int nchars, int offset, int abpc, int ubpc, Asn1CharSet charSet)
- virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)
- virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)
- virtual void Encode (Asn1PerOutputStream outs, int nchars, int offset, int abpc, int ubpc, Asn1CharSet charSet)
- virtual void Encode (Asn1PerOutputStream outs, int abpc, int ubpc, Asn1CharSet charSet)
- virtual void Encode (Asn1PerOutputStream outs, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)

- static Asn1UniversalString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 Universal string value.

Definition at line 37 of file Asn1UniversalString.cs

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

- Asn1UniversalString.cs

Member Data Documentation

const int BITSPERCHAR

The `BITSPERCHAR` constant specifies the number of bits per character for PER (aligned or unaligned).

Definition at line 42 of file Asn1UniversalString.cs

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

- Asn1UniversalString.cs

int [] mValue

The `mValue` public member variable is used to hold the string value to be encoded or the results of a Decode operation. For UniversalString, the characters are stored in an array of 32-bit integer values.

Definition at line 59 of file Asn1UniversalString.cs

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

- Asn1UniversalString.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 28).

Definition at line 49 of file Asn1UniversalString.cs

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

- Asn1UniversalString.cs

System.Text.StringBuilder mStringBuffer

Variable holds the string representation of the value

Definition at line 63 of file Asn1UniversalString.cs

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

- Asn1UniversalString.cs

Asn1UniversalString ()

This constructor creates an empty string that can be used in a Decode method call to receive a string value.

Asn1UniversalString (int[] value)

This constructor initializes the universal string from the given an array of 32-bit integer value.

Table 3.606. Parameters

value	universal string value as array of 32-bit int

Asn1UniversalString (System.String value)

This constructor converts a standard C# string value into a universal string.

Table 3.607. Parameters

value	universal string value as string

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 universal string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.608. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable.

Table 3.609. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet)

This method decodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but allows a permitted alphabet character set to be specified. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable.

Table 3.610. Parameters

buffer	Decode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)

virtual void Decode (Asn1PerDecodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the Decode method decodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The decoded result is stored in the public `mValue` member variable.

Table 3.611. Parameters

buffer	Decode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Decode (Asn1OerDecodeBuffer buffer)

Decode the value in accordance with OER.

This class's implementation decodes the string with a length determinant. If a subclass should be decoded without a length determinant, it should override this method to invoke Decode(buffer, length). Subclasses may override this to add constraint checks after invoking this method to decode the string.

void Decode (Asn1OerDecodeBuffer buffer, int length)

Decode the value in accordance with OER.

This class's implementation decodes a string of the given length. This method is final as I don't see any reason for overriding it.

Table 3.612. Parameters

length	Length of string to decode, in characters.

virtual void DecodeXER (System.String buffer, System.String attrs)

This method decodes an ASN.1 Universal String value using the XML encoding rules (XER).

Table 3.613. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override void DecodeXML (System.String buffer, System.String attrs)

This method decodes an ASN.1 Universal String value using the XML schema encoding rules(asn2xsd).

Table 3.614. Parameters

buffer	String containing data to be decoded
attrs	Attributes string from element tag

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 universal string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.615. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable.

Table 3.616. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet)

This method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified.

The value to encode is stored in the public `mValue` member variable.

Table 3.617. Parameters

buffer	Encode message buffer object
charSet	Object representing permitted alphabet constraint character set (optional)

virtual void Encode (Asn1PerEncodeBuffer buffer, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes both a size constraint and permitted alphabet constraint are present.

The value to encode is stored in the public `mValue` member variable.

Table 3.618. Parameters

buffer	Encode message buffer object

charSet	Object representing the permitted alphabet constraint character set
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Encode (Asn1OerEncodeBuffer buffer)

Encode the value in accordance with OER.

This class's implementation invokes encode(buffer, true) to encode the string with a length determinant. If a subclass should be encoded without a length determinant, it should override this to invoke Encode(buffer, false).

virtual void Encode (Asn1OerEncodeBuffer buffer, bool withLength)

Encode the string, with or without a length determinant.

Subclasses may override this (e.g. to add constraint checks) and invoke this method to perform the encoding.

Table 3.619. Parameters

withLength	true if a length determinant should be encoded.

override void Encode (Asn1XerEncoder buffer, System.String elemName)

This method encodes an ASN.1 Universal String value using the XML encoding rules (XER).

Table 3.620. Parameters

buffer	Encode message buffer object
elemName	Element name

override void Encode (Asn1XmlEncoder buffer, System.String elemName, System.String nsPrefix)

This method encodes an ASN.1 Universal String value with element and namespace prefix name tag using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.621. Parameters

buffer	Encode message buffer object
elemName	Element name
nsPrefix	Element namespace value

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 universal string including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.622. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.623. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.624. Parameters

outs	PER Output Stream object

Table 3.625. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet)

This method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no size constraints but does allow a permitted alphabet character set to be specified.

The value to encode is stored in the public `mValue` member variable.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.626. Parameters

outs	PER Output Stream object
charSet	Object representing permitted alphabet constraint character set (optional)

Table 3.627. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes both a size constraint and permitted alphabet constraint are present.

The value to encode is stored in the public `mValue` member variable.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.628. Parameters

outs	PER Output Stream object
charSet	Object representing the permitted alphabet constraint character set
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

Table 3.629. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void EncodeData (Asn1XmlXerEncoder buffer)

This method encodes an ASN.1 Universal String value using the XML Encoding as specified in the XML schema standard(asn2xsd).

Table 3.630. Parameters

buffer	Encode message buffer object

override bool Equals (System.Object value)

This method compares this character string value to the given value for equality.

Table 3.631. Parameters

value	The Object to compare with the current Object. Object should be instance of Asn1UniversalString.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

override System.String ToString ()

This method will return a string representation of the value. The format is the ASN.1 value format for this type.

Returns: . Stringified representation of the value

bool validate (Asn1CharSet charSet)

This method will attempt to validate a string against its internal character set.

Returns: . True or False.

void SetValue (System.String value)

Assign this object's data based on the given String value. This version is for standard .NET framework.

Table 3.632. Parameters

value	universal string value as string

virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)

This method decodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes that a permitted alphabet constraint has been specified that would reduce the the number of bits-per-character from the default character set. It also assumes a general length determinant is present (i.e. there is not size constraint). The decoded result is stored in the public `mValue` member variable.

Table 3.633. Parameters

buffer	Decode message buffer object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)

virtual void Decode (Asn1PerDecodeBuffer buffer, int nchars, int abpc, int ubpc, Asn1CharSet charSet, int startIdx)

This method decodes the contents of a UniversalString. This version of the method assumes a permitted alphabet constraint is in place.

Table 3.634. Parameters

buffer	Decode message buffer object
nchars	Number of characters
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)
startIdx	Start index to fill in value array

virtual void Decode (Asn1PerDecodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the Decode method decodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The decoded result is stored in the public `mValue` member variable.

Table 3.635. Parameters

buffer	Decode message buffer object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

virtual void Encode (Asn1PerEncodeBuffer buffer, int nchars, int offset, int abpc, int ubpc, Asn1CharSet charSet)

This method encodes the contents of a UniversalString type. This version assumes a permitted alphabet constraint was specified.

Table 3.636. Parameters

buffer	Encode message buffer object
nchars	Number of characters from string to encode
offset	Offset to first char in string to encode
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing permitted alphabet constraint character set (optional)

virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet)

This method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable.

Table 3.637. Parameters

buffer	Encode message buffer object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)

virtual void Encode (Asn1PerEncodeBuffer buffer, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The value to encode is stored in the public `mValue` member variable.

Table 3.638. Parameters

buffer	Encode message buffer object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound

upper	Effective size constraint upper bound

virtual void Encode (Asn1PerOutputStream outs, int nchars, int offset, int abpc, int ubpc, Asn1CharSet charSet)

This method encodes the contents of a UniversalString type. This version assumes a permitted alphabet constraint was specified.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.639. Parameters

outs	PER Output Stream object
nchars	Number of characters from string to encode
offset	Offset to first char in string to encode
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing permitted alphabet constraint character set (optional)

Table 3.640. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, int abpc, int ubpc, Asn1CharSet charSet)

This method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.641. Parameters

outs	PER Output Stream object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)

Table 3.642. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, int abpc, int ubpc, Asn1CharSet charSet, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 UniversalString value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present but no permitted alphabet constraint.

The value to encode is stored in the public `mValue` member variable.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.643. Parameters

outs	PER Output Stream object
abpc	Number of bits per character (aligned)
ubpc	Number of bits per character (unaligned)
charSet	Object representing the permitted alphabet constraint character set (optional)
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

Table 3.644. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1UTCTime class Reference

- static new readonly Asn1Tag _TAG
-
-
- Asn1UTCTime ( )
- Asn1UTCTime (bool useDerRules)
- Asn1UTCTime (System.String data)

- Asn1UTCTime (System.String data, bool useDerRules)
- override void Clear ( )
- override System.Int32 CompareTo (System.Object obj)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void ParseString (System.String data)
- override void SetTime (System.DateTime time)

- override bool CompileString ( )
- override void Init ( )

- static Asn1UTCTime ( )

Detailed Description

This is a container class for holding the components of an ASN.1 UTC time string value.

Definition at line 36 of file Asn1UTCTime.cs

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

- Asn1UTCTime.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 23).

Definition at line 41 of file Asn1UTCTime.cs

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

- Asn1UTCTime.cs

Asn1UTCTime ()

The default constructor creates an empty time string object.

Asn1UTCTime (bool useDerRules)

This constructor creates an empty UTCTime string object and allows DER encoding rules to be specified.

Table 3.645. Parameters

useDerRules	'true' if time string should be encoded with DER/PER.

Asn1UTCTime (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given time string.

Table 3.646. Parameters

data	UTCTime as string

Asn1UTCTime (System.String data, bool useDerRules)

This version of the constructor can be used to set the string `mValue` member variable to the given time string and specify DER encoding rules be used to construct the string.

Table 3.647. Parameters

data	UTCTime as string
useDerRules	'true' if time string should be encoded with DER/PER.

override void Clear ()

Clears out time string.

override System.Int32 CompareTo (System.Object obj)

This method compares this object with Asn1Time class instance or with System.DateTime instance. Returns a negative integer, zero, or a positive integer as this object is less than, equal to, or greater than the specified object. Note that the action of this method may differentiate for different inherited Asn1Time classes.

Table 3.648. Parameters

obj	the Object to be compared.

Returns: . The difference in Ticks with the specified object.

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 UTCTime value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.649. Parameters

buffer	Decode message buffer object

explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 UTCTime type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.650. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 UTC time string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.651. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.652. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void ParseString (System.String data)

This method parses passed UTCTime string.

Table 3.653. Parameters

data	The UTCTime string value to be parsed.

Table 3.654. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void SetTime (System.DateTime time)

This method converts the System.DateTime value to UTCTime string.

Table 3.655. Parameters

time	The System.DateTime value.

Table 3.656. Exceptions

Asn1Exception	Thrown, if operation is failed.

override bool CompileString ()

Compiles new time string accoring X.680 (clause 42) and ISO 8601.

Returns: . `true` if successful; otherwise `false`.

override void Init ()

This method initializes the Asn1UTCTime class member variables.

Com::Objsys::Asn1::Runtime::Asn1UTF8String class Reference

- static new readonly Asn1Tag _TAG

- Asn1UTF8String ( )
- Asn1UTF8String (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1PerEncodeBuffer buffer)

- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)
- void SetAnyAttribute (String qname, String val)

- static string Decode (Asn1BerDecodeBuffer buffer, Asn1Tag explicitTag, int implicitLength)
- static string DecodeUTF8 (Asn1PerDecodeBuffer buffer)
- static String DecodeUTF8 (Asn1OerDecodeBuffer buffer)
- static int Encode (Asn1BerEncodeBuffer buffer, Asn1Tag explicitTag, string value)
- static void Encode (Asn1PerEncodeBuffer buffer, string value)
- static void Encode (Asn1OerEncodeBuffer buffer, String value)
- static void Encode (Asn1BerOutputStream outs, Asn1Tag explicitTag, string value)

- virtual void Decode (Asn1OerDecodeBuffer buffer, int length)

- static byte [] AllocByteArray (int nbytes)
- static Asn1UTF8String ( )
- static String DecodeUTF8 (Asn1OerDecodeBuffer buffer, int length)
- static byte [] ReAllocByteArray (byte [] ba1, int nbytes)

Detailed Description

This is a container class for holding the components of an ASN.1 UTF-8 string value.

Definition at line 35 of file Asn1UTF8String.cs

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

- Asn1UTF8String.cs

new readonly Asn1Tag _TAG

The _TAG constant describes the universal tag for this data type (UNIVERSAL 12).

Definition at line 40 of file Asn1UTF8String.cs

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

- Asn1UTF8String.cs

Asn1UTF8String ()

The default constructor creates an empty time string object.

Asn1UTF8String (System.String data)

This constructor can be used to set the UTF8 String `mValue` member variable to the given string value.

Table 3.657. Parameters

data	UTF8 String value

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 UTF-8 string value including the UNIVERSAL tag value and length if explicit tagging is specified. This string type uses variable length character encodings.

Table 3.658. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 UTF-8 string value using the packed encoding rules (PER). The string is assumed to not contain a size constraint.

Table 3.659. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)

This method decodes a sized ASN.1 UTF-8 string value using the packed encoding rules (PER).

Table 3.660. Parameters

buffer	Decode message buffer object

lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

override void Decode (Asn1OerDecodeBuffer buffer)

This method decodes an ASN.1 UTF8String string value that was encoded according to OER.

Table 3.661. Parameters

buffer	Decode message buffer object

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 UTF8 string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.662. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an unconstrained ASN.1 UTF-8 string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Table 3.663. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This method encodes a size-constrained ASN.1 UTF-8 string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Table 3.664. Parameters

buffer	Encode message buffer object

lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

override void Encode (Asn1OerEncodeBuffer buffer)

This method encodes a ASN.1 UTF8String according to OER.

Table 3.665. Parameters

buffer	Encode message buffer object

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 UTF8 string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.666. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.667. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1PerOutputStream outs)

This method encodes an unconstrained ASN.1 UTF-8 string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.668. Parameters

outs	PER Output Stream object

Table 3.669. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)

This method encodes a size-constrained ASN.1 UTF-8 string value using the packed encoding rules (PER). The value to be encoded is stored in the 'mValue' public member variable within this class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.670. Parameters

outs	PER Output Stream object
lower	Lower bound (inclusive) of size constraint
upper	Upper bound (inclusive) of size constraint

Table 3.671. Exceptions

Asn1Exception	Thrown, if operation is failed.

void SetAnyAttribute (String qname, String val)

This method will set the anyAttribute type value for given qname and value of XML attribute

Table 3.672. Parameters

qname	The qualified (prefixed) name
value	The attribute value

static string Decode (Asn1BerDecodeBuffer buffer, Asn1Tag explicitTag, int implicitLength)

This method decodes an ASN.1 UTF-8 string value including the UNIVERSAL tag value and length if explicit tagging is specified. This string type uses variable length character encodings.

BER encodes UTF8String, OID-IRI, and RELATIVE-OID-IRI in essentially the same way; this permits sharing of the implementation.

Table 3.673. Parameters

buffer	Decode message buffer object
explicitTag	Tag to explicitly match, or null if none
implicitLength	Length of contents if implicit

static string DecodeUTF8 (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 UTF-8 string value using the packed encoding rules (PER).

Table 3.674. Parameters

buffer	Decode message buffer object

static String DecodeUTF8 (Asn1OerDecodeBuffer buffer)

This method decodes the content of an ASN.1 UTF8String string value that was encoded according to OER.

Table 3.675. Parameters

buffer	Decode message buffer object

Returns: . The decoded string.

static int Encode (Asn1BerEncodeBuffer buffer, Asn1Tag explicitTag, string value)

This method encodes an ASN.1 UTF8 string type. Nearly the same encoding is shared by OID-IRI and RELATIVE-OID_IRI; this method facilitates sharing the implementation.

The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.676. Parameters

buffer	Encode message buffer object
explicitTag	Tag to explicitly encode, or null for none.
value	The value to encode. For OID-IRI and RELATIVE-OID-IRI, whitespace should already have been removed.

Returns: . Length in octets of encoded component

static void Encode (Asn1PerEncodeBuffer buffer, string value)

This method encodes a string using the packed encoding rules (PER) specifie for ASN1. UTF8String. These rules are essentially shared with OID-IRI and RELATIVE-OID-IRI.

Table 3.677. Parameters

buffer	Encode message buffer object

value	The value to be encoded. In the case of OID-IRI and RELATIVE-OID-IRI, should not contain whitespace.

static void Encode (Asn1OerEncodeBuffer buffer, String value)

This method encodes an ASN.1 UTF8String according to OER.

Table 3.678. Parameters

buffer	Encode message buffer object
value	The value to be encoded.

static void Encode (Asn1BerOutputStream outs, Asn1Tag explicitTag, string value)

This method encodes the given string using the BER encoding rules for UTF8String, including the given tag, if provided.

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.679. Parameters

outs	BER Output Stream object
explicitTag	Tag to encode, or null for none.
value	The value to encode. For OID-IRI and RELATIVE-OID-IRI, should not contain whitespace.

Table 3.680. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Decode (Asn1OerDecodeBuffer buffer, int length)

This method decodes the content of an ASN.1 UTF8String string value that was encoded according to OER. The length is not decoded but is taken to be as given.

Table 3.681. Parameters

buffer	Decode message buffer object

static String DecodeUTF8 (Asn1OerDecodeBuffer buffer, int length)

This method decodes the content of an ASN.1 UTF8String string value that was encoded according to OER. The length is not decoded but is taken to be as given.

Table 3.682. Parameters

buffer	Decode message buffer object

Returns: . The decoded string.

Com::Objsys::Asn1::Runtime::Asn1Util class Reference

- static readonly byte [] base64DecodeTable
- static readonly char [] base64EncodeTable

- static System.String BCDToString (byte [] bcd)
- static void CloseRuntime ( )
- static byte [] DecodeBase64Array (char [] srcArray)
- static string EncodeBase64Array (byte [] srcArray)
- static byte [] GetAddressBytes (string ipaddress)
- static int GetBytesCount (long val)
- static int GetUlongBytesCount (long val)
- static int HexToByte (char c)
- static bool IsLimited ( )
- static string [] PLMNidentityToString (byte [] plmnIdentity)
- static byte [] StringToBCD (System.String str)
- static byte [] StringToPLMNidentity (string mcc, string mnc)
- static byte [] StringToTBCD (System.String str)
- static String StripWhitespace (String value)
- static char TbcdBinToChar (byte tbcdDigit)
- static byte TbcdCharToBin (char digit)
- static System.String TBCDToString (byte [] bcd)

- static void ToArray (System.Collections.ICollection c, System.Object [] objects)
- static byte [] ToByteArray (System.String sourceString)
- static char [] ToCharArray (byte [] byteArray)
- static char ToHexChar (int nibble)
- static System.String ToHexString (byte b)
- static System.String ToHexString (byte [] b, int offset, int nbytes)
- static System.String ToHexString (byte [] b, int offset, int nbytes, bool spaces)
- static int URShift (int number, int bits)
- static int URShift (int number, long bits)
- static long URShift (long number, int bits)
- static long URShift (long number, long bits)
- static void WriteStackTrace (System.Exception throwable, System.IO.TextWriter stream)
- static int DecodeBase64Char (char c)

Detailed Description

This class contains some general purpose static utility functions.

Definition at line 34 of file Asn1Util.cs

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

- Asn1Util.cs

static System.String BCDToString (byte[] bcd)

Translates a BCD string to an ASCII string. The returned string will consist of characters in [0..9] and [A-E]. All half-bytes in the input shall be less than 0xF, except that in the final byte, the low half-byte may be 0xF, to act as filler for a string of odd length.

Table 3.683. Parameters

bcd	the source BCD string

Returns: . the ASCII string.

static void CloseRuntime ()

Signal that you are finished with the ASN1C runtime and release internal resources. Do not use the runtime classes, on any thread, after calling this.

This may be called from any thread.

static byte [] DecodeBase64Array (char[] srcArray)

Translates the specified Base64 array into byte array. The resulting array could be converted to the String by `new String (byte[])` constructor.

Table 3.684. Parameters

srcArray	Base64 byte array to be translated

Returns: . decoded byte array

static string EncodeBase64Array (byte[] srcArray)

Translates the specified byte array to Base64 string.

Table 3.685. Parameters

srcArray	byte array to be translated

Returns: . Base64 encoded string

static byte [] GetAddressBytes (string ipaddress)

Converts an IPaddress to byte array

Table 3.686. Parameters

ipaddress	String representation of IP Address

Returns: . The byte array(size 4) represenation of IP address

static int GetBytesCount (long val)

Calculate the number of bytes necessary to represent a signed long value.

Table 3.687. Parameters

val	singed long value.

Returns: . the number of bytes.

static int GetUlongBytesCount (long val)

Calculate the number of bytes necessary to represent an unsigned long value.

Table 3.688. Parameters

val	unsinged long value.

Returns: . the number of bytes.

static int HexToByte (char c)

Return the value of a single hexadecimal character. /p>

Table 3.689. Parameters

c	The hex character (0-9,A-F,a-f)

Returns: . Value of hex character, 0-15.

Table 3.690. Exceptions

Asn1Exception	Thrown if character is not a valid hexadecimal character.

static bool IsLimited ()

Indicates whether the run-time is limited or unlimited.

Returns: . true if limited, false if unlimited

static string [] PLMNidentityToString (byte[] plmnIdentity)

Translates a PLMNidentity string (i.e., byte array) to ASCII strings for the MCC and the MNC. Refer to the description of PLMNidentity in StringToPLMNIdentity().

Table 3.691. Parameters

plmnIdentity	The source PLMNidentity string (byte array)

Returns: . Array of 2 strings where element 0 is the MCC and element 1 is the MNC.

static byte [] StringToBCD (System.String str)

Translates an ASCII string to a BCD string. The ASCII string must contain only the following characters: [0..9A-Ea-e]. If the length of the source string is not even, the unused part of the last byte will be set to 0xF.

Table 3.692. Parameters

str	the source ASCII string

Returns: . the BCD string as a byte array.

Table 3.693. Exceptions

Asn1ValueParseException	If invalid characters are in the source string.

static byte [] StringToPLMNidentity (string mcc, string mnc)

Translates ASCII Strings for MCC and MNC to a PLMNidentity. The PLMNidentity format is described in (for example) 3GPP TS 25.413. Each nibble represents a digit from 0 through 9 inclusive. The first 3 nibbles are the digits of the MCC. If the MNC is 2 digits long, the fourth nibble is a filler of 0xf. Otherwise, the fourth nibble is the first digit of the MNC. Then the remaining 2 nibbles are the remaining digits of the MNC.

An ASCII string of "123" for the MCC and "45" for the MNC would be encoded as 0x21F354.

Table 3.694. Parameters

mcc	The source ASCII string for the MCC.
mnc	The source ASCII string for the MNC.

Returns: . The PLMNidentity string as a byte array.

static byte [] StringToTBCD (System.String str)

Translates an ASCII string to a TBCD string. The TBCD encoding format is as described in 3GPP 29.002:

- the characters used are [0-9*#abc].
- each nibble represents a digit, and the upper nibble represents the digit which follows the digit represented in the lower nibble.
- 0xF is used as filler in the final high nibble when the input string has and odd length. The string "12345" would be encoded 0x2143f5.

Table 3.695. Parameters

str	the source ASCII string

Returns: . the TBCD string as a byte array.

Table 3.696. Exceptions

Asn1ValueParseException	If invalid characters are in the source string.

static String StripWhitespace (String value)

Return the given string with all whitespace characters removed. Here, "whitespace characters" means those characters defined by X.680 12.1.6 as whitespace: HT, LF, VT, FF, CR, SPACE.

Returns: . value, with all whitespace removed; if the input string has no whitespace, it is returned itself.

static char TbcdBinToChar (byte tbcdDigit)

This function converts a TBCD binary character into its ASCII equivalent. /p>

Table 3.697. Parameters

tbcdDigit	TBCD digit

Returns: . the converted character

static byte TbcdCharToBin (char digit)

This function converts a TBCD character ('0'-'9',"*#abc") into its binary equivalent. /p>

Table 3.698. Parameters

digit	TBCD digit character ('0'-'9',"*#abc")

Returns: . binary representation of given char

static System.String TBCDToString (byte[] bcd)

Translates a TBCD string to an ASCII string. Refer to the description of TBCD in stringToTBCD.

Table 3.699. Parameters

bcd	the source TBCD string

Returns: . the ASCII string.

static void ToArray (System.Collections.ICollection c, System.Object[] objects)

Obtains an array containing all the elements of the collection.

Table 3.700. Parameters

c	The Collection instance, which contains the elements.

objects	The array into which the elements of the collection will be stored.

Returns: . The array containing all the elements of the collection.

static byte [] ToByteArray (System.String sourceString)

Converts a string to an array of bytes

Table 3.701. Parameters

sourceString	The string to be converted

Returns: . The new array of bytes

static char [] ToCharArray (byte[] byteArray)

Converts an array of bytes to an array of chars

Table 3.702. Parameters

byteArray	The array of bytes to convert

Returns: . The new array of chars

static char ToHexChar (int nibble)

Convert a value from 0x0 to 0xF to the corresponding hex char. Lowercase letters are used for the hex characters. /p>

static System.String ToHexString (byte b)

Convert a byte value to a hex string. Unlike the C# built-in function, this will:

a. not sign extend the byte value out to 32 bits if the MSB is set, and b. put a zero padding byte in front if less than 0xf

In other words, a character string of length 2 is always returned.

Table 3.703. Parameters

b	byte value

Returns: . Hex String value

static System.String ToHexString (byte[] b, int offset, int nbytes)

Convert a array of bytes into a hex string.

Table 3.704. Parameters

b	byte array to be converted to hex string
offset	start position in byte array
nbytes	no. of bytes to be converted

Returns: . Hex String value

static System.String ToHexString (byte[] b, int offset, int nbytes, bool spaces)

Convert a array of bytes into a hex string.

Table 3.705. Parameters

b	byte array to be converted to hex string
offset	start position in byte array
nbytes	no. of bytes to be converted
spaces	Pass true if each byte's hex digits should be followed by a space character.

Returns: . Hex String value

static int URShift (int number, int bits)

Performs an unsigned bitwise right shift with the specified number. The low-order bits of number are discarded, the remaining bits are shifted right, and the high-order empty bit positions are set to zero.

Table 3.706. Parameters

number	Number to operate on
bits	Ammount of bits to shift

Returns: . The resulting number from the shift operation

static int URShift (int number, long bits)

Performs an unsigned bitwise right shift with the specified number The low-order bits of number are discarded, the remaining bits are shifted right, and the high-order empty bit positions are set to zero.

Table 3.707. Parameters

number	Number to operate on
bits	Ammount of bits to shift

Returns: . The resulting number from the shift operation

static long URShift (long number, int bits)

Performs an unsigned bitwise right shift with the specified number The low-order bits of number are discarded, the remaining bits are shifted right, and the high-order empty bit positions are set to zero.

Table 3.708. Parameters

number	Number to operate on
bits	Ammount of bits to shift

Returns: . The resulting number from the shift operation

static long URShift (long number, long bits)

Performs an unsigned bitwise right shift with the specified number The low-order bits of number are discarded, the remaining bits are shifted right, and the high-order empty bit positions are set to zero.

Table 3.709. Parameters

number	Number to operate on
bits	Ammount of bits to shift

Returns: . The resulting number from the shift operation

static void WriteStackTrace (System.Exception throwable, System.IO.TextWriter stream)

Writes the exception stack trace to the received stream

Table 3.710. Parameters

throwable	Exception to obtain information from
stream	Output sream used to write to

static int DecodeBase64Char (char c)

Translates the specified character, which is assumed to be in the "Base 64 Alphabet" into its equivalent 6-bit positive integer.

Table 3.711. Exceptions

ArgumentException	if c is not in the Base64 Alphabet.

IndexOutOfRangeException	if c is not in the Base64 Alphabet.

Com::Objsys::Asn1::Runtime::Asn1Value class Reference

- static byte [] AllocBitArray (int numbits)
- static byte HexCharsToByte (char c1, char c2)
- static byte HexCharToByte (char c)
- static byte [] ParseString (System.String data, IntHolder numbits)
- static byte [] ParseString (System.String data)
- static bool StringEqualsBytes (String value, byte [] data)
- static bool StringEqualsBytes (String value, byte [] data, int nbits)

Detailed Description

This class provides methods for parsing and formatting text in ASN.1 value notation.

Definition at line 33 of file Asn1Value.cs

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

- Asn1Value.cs

static byte HexCharsToByte (char c1, char c2)

Return the value of a hexadecimal number whose two characters are c1, followed by c2.

static byte HexCharToByte (char c)

Return the integer value of the given hexadecimal character. /p>

static byte [] ParseString (System.String data, IntHolder numbits)

This static method parses the given ASN.1 value text (either a binary or hex data string) and returns the value in a binary byte array. The number of bits is also returned.

Examples of valid value formats are as follows:

Binary string: '11010010111001'B

Hex string: '0fa56920014abc'H

Char string: 'abcdefg'

Table 3.712. Parameters

data	The ASN.1 value specification text
numbits	Holder to receive number of bits in string

Returns: . byte array value

static byte [] ParseString (System.String data)

This overloaded version of the ParseString method sets the numbits holder value to null.

Table 3.713. Parameters

data	The ASN.1 value specification text

Returns: . byte array value

static bool StringEqualsBytes (String value, byte[] data)

This static method parses the given ASN.1 value text (either a binary or hex data string) and compares it with the given byte array. Examples of valid value formats are as follows: Binary string: '11010010111001'B Hex string: '0fa56920014abc'H Char string: 'abcdefg'

Table 3.714. Parameters

value	The ASN.1 value specification text
data	Array of bytes to compare string with.

static bool StringEqualsBytes (String value, byte[] data, int nbits)

This static method parses the given ASN.1 value text (either a binary or hex data string) and compares it with the given byte array. This version of the method allows you to specify a number of bits to compare, which is useful when dealing with bit strings, as opposed to octet strings.

Examples of valid value formats are as follows: Binary string: '11010010111001'B Hex string: '0fa56920014abc'H Char string: 'abcdefg' /p>

Table 3.715. Parameters

value	The ASN.1 value specification text
data	Array of bytes to compare string with.
nbits	Number of bits in data to compare.

Com::Objsys::Asn1::Runtime::Asn1ValueParseExcepti class Reference

- Asn1ValueParseException (System.String text)
- Asn1ValueParseException (System.String text, int offset)

Detailed Description

This class defines the 'ASN.1 value Parse' exception that is thrown when a string containing an ASN.1 value cannot be parsed.

Definition at line 36 of file Asn1ValueParseException.cs

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

- Asn1ValueParseException.cs

Asn1ValueParseException (System.String text)

This constructor creates an exception object with a textual message describing the expected and parsed tag values.

Table 3.716. Parameters

text	The value string that could not be parsed

Asn1ValueParseException (System.String text, int offset)

This constructor creates an exception object with a textual message describing the expected and parsed tag values. This version allows the offset in the string to also be specified.

Table 3.717. Parameters

text	The value string that could not be parsed
offset	Offset to error location in string

Com::Objsys::Asn1::Runtime::Asn1VarWidthCharStrin class Reference

Public Attributes

- const int BITSPERCHAR_A

- const int BITSPERCHAR_U

- Asn1VarWidthCharString (short typeCode)
- Asn1VarWidthCharString (System.String data, short typeCode)

- override void Decode (Asn1PerDecodeBuffer buffer)
- virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)
- override void Decode (Asn1OerDecodeBuffer buffer)
- override void Encode (Asn1PerEncodeBuffer buffer)
- virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)
- override void Encode (Asn1OerEncodeBuffer buffer)
- override void Encode (Asn1PerOutputStream outs)
- virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)

Detailed Description

This is an abstract base class for holding the ASN.1 variable width character string types (GraphicString, GeneralString, TeletexString, T61String, VideotexString, ObjectDescriptor).

Definition at line 37 of file Asn1VarWidthCharString.cs

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

- Asn1VarWidthCharString.cs

Member Data Documentation

const int BITSPERCHAR_A

The `BITSPERCHAR_A` constant specifies the number of bits per character for PER (aligned).

Definition at line 42 of file Asn1VarWidthCharString.cs

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

- Asn1VarWidthCharString.cs

const int BITSPERCHAR_U

The `BITSPERCHAR_U` constant specifies the number of bits per character for PER (unaligned).

Definition at line 47 of file Asn1VarWidthCharString.cs

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

- Asn1VarWidthCharString.cs

Asn1VarWidthCharString (short typeCode)

The default constructor creates an empty string object.

Table 3.718. Parameters

typeCode	Universal ID code for ASN.1 character string

Asn1VarWidthCharString (System.String data, short typeCode)

This version of the constructor can be used to set the string `mValue` member variable to the given string.

Table 3.719. Parameters

data	Character string
typeCode	Universal ID code for ASN.1 character string

override void Decode (Asn1PerDecodeBuffer buffer)

This method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints. Decoded characters are assigned as-is to the output string. The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.720. Parameters

buffer	Decode message buffer object

virtual void Decode (Asn1PerDecodeBuffer buffer, long lower, long upper)

This overloaded version of the Decode method decodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present. A permitted alphabet may be passed, but it will be ignored.

The decoded result is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.721. Parameters

buffer	Decode message buffer object
lower	Effective size constraint lower bound

upper	Effective size constraint upper bound

override void Decode (Asn1OerDecodeBuffer buffer)

Decode the value in accordance with OER.

Subclasses may override this to add constraint checks after invoking this method to decode the string.

override void Encode (Asn1PerEncodeBuffer buffer)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.722. Parameters

buffer	Encode message buffer object

virtual void Encode (Asn1PerEncodeBuffer buffer, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present. A permitted alphabet may be passed, but it will be ignored.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Table 3.723. Parameters

buffer	Encode message buffer object
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

override void Encode (Asn1OerEncodeBuffer buffer)

Encode the value in accordance with OER.

We're assuming each char is encoded in a single byte of the same value as the char.

Subclasses may override this (e.g. to add constraint checks) and invoke this method to perform the encoding.

override void Encode (Asn1PerOutputStream outs)

This method encodes an ASN.1 character string value in accordance with the packed encoding rules (PER) directly into the stream. This version of the method assumes no permitted alphabet or size constraints.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.724. Parameters

outs	PER Encode message stream object

Table 3.725. Exceptions

Asn1Exception	Thrown, if operation is failed.

virtual void Encode (Asn1PerOutputStream outs, long lower, long upper)

This overloaded version of the encode method encodes an ASN.1 character string value directly into the stream, in accordance with the packed encoding rules (PER). This version of the method assumes a size constraint is present. A permitted alphabet may be passed, but it will be ignored.

The value to encode is stored in the public `mValue` member variable in the `Asn1CharString` base class.

Also throws any exception thrown by the Asn1PerOutputStream.

Table 3.726. Parameters

outs	PER Encode message stream object
lower	Effective size constraint lower bound
upper	Effective size constraint upper bound

Table 3.727. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1VideotexString class Reference

- static new readonly Asn1Tag _TAG
- Asn1VideotexString ( )
- Asn1VideotexString (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)
- static Asn1VideotexString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 videotex string value.

Definition at line 35 of file Asn1VideotexString.cs

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

- Asn1VideotexString.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 21).

Definition at line 40 of file Asn1VideotexString.cs

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

- Asn1VideotexString.cs

Asn1VideotexString ()

The default constructor creates an empty string object.

Asn1VideotexString (System.String data)

This version of the constructor can be used to set the string `mValue` member variable to the given string.

Table 3.728. Parameters

data	string representation of videotex string

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 Videotex string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.729. Parameters

buffer	Decode message buffer object

explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

Table 3.730. Exceptions

Asn1Exception	Thrown, if operation is failed.

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 Videotex String type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.731. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

Table 3.732. Exceptions

Asn1Exception	Thrown, if operation is failed.

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 videotex string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.733. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Com::Objsys::Asn1::Runtime::Asn1VisibleString class Reference

- static new readonly Asn1Tag _TAG

- static readonly Asn1CharSet CHARSET

- Asn1VisibleString ( )
- Asn1VisibleString (System.String data)
- override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)
- override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)
- override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

- static Asn1VisibleString ( )

Detailed Description

This is a container class for holding the components of an ASN.1 Visible string value.

Definition at line 35 of file Asn1VisibleString.cs

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

- Asn1VisibleString.cs

new readonly Asn1Tag _TAG

The `_TAG` constant describes the universal tag for this data type (UNIVERSAL 26).

Definition at line 42 of file Asn1VisibleString.cs

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

- Asn1VisibleString.cs

Asn1VisibleString ()

The default constructor creates an empty string object.

Asn1VisibleString (System.String data)

This version of the constructor can be used to set the Visible string `mValue` member variable to the given string.

Table 3.734. Parameters

data	string representation of visible string

override void Decode (Asn1BerDecodeBuffer buffer, bool explicitTagging, int implicitLength)

This method decodes an ASN.1 Visible string value including the UNIVERSAL tag value and length if explicit tagging is specified.

Table 3.735. Parameters

buffer	Decode message buffer object
explicitTagging	Flag indicating element is explicitly tagged
implicitLength	Length of contents if implicit

override int Encode (Asn1BerEncodeBuffer buffer, bool explicitTagging)

This method encodes an ASN.1 Visible string type. The UNIVERSAL tag value and length is also encoded if explicit tagging is specified.

Table 3.736. Parameters

buffer	Encode message buffer object
explicitTagging	Flag indicating explicit tagging should be done

Returns: . Length in octets of encoded component

override void Encode (Asn1BerOutputStream outs, bool explicitTagging)

This method encodes and writes to the stream an ASN.1 visible string value including the UNIVERSAL tag value and length if explicit tagging is specified. This overloaded version uses the Basic Encoding Rules (BER).

Throws, Exception thrown by C# System.IO.Stream for I/O error

Table 3.737. Parameters

outs	BER Output Stream object
explicitTagging	Flag indicating explicit tagging should be done

Table 3.738. Exceptions

Asn1Exception	Thrown, if operation is failed.

Com::Objsys::Asn1::Runtime::Asn1VisibleStringChars class Reference

.

- Asn1VisibleStringCharset ( )
- override int GetCharAtIndex (int index)
- override int GetCharIndex (int charValue)
- override bool validate (String s)

Asn1XerSaxHandler class Reference

Com::Objsys::Asn1::Runtime::BigInteger class Reference

- const int MAX_BIG_INT_LEN
- int mSign
- byte [] mValue

- static int [] bitsPerDigit
- static int [] byteRadix
- static int [] digitsPerByte

Private Attributes

- const int ADDRESS_BITS
- const int BIT_INDEX_MASK
- const int BITS_PER_UNIT
- const int UNIT_MASK

- static readonly BigInteger ONE
- static readonly BigInteger TEN
- static readonly BigInteger ZERO

- BigInteger Add (BigInteger op)
- BigInteger ( )
- BigInteger (byte [] value, int sign)
- BigInteger (System.String value)
- BigInteger (System.Int64 value)

- BigInteger (System.String value, int radix)
- int BitLength ( )
- virtual int CompareTo (BigInteger value)
- virtual bool Equals (long value)
- override bool Equals (System.Object value)
- byte [] GetData ( )
- override int GetHashCode ( )
- void Init (System.String val, int radix)
- bool IsNegative ( )
- long LongValue ( )
- void SecureDelete ( )
- void SetData (byte [] ivalue)
- BigInteger Subtract (BigInteger op)
- System.String ToString (int radix)
- override System.String ToString ( )

- static implicit operator BigInteger (long value)

- static int BitsLeftOf (int x)
- static byte [] bitwiseAdd (byte [] a, byte [] b)
- static byte [] bitwiseSubtract (byte [] a, byte [] b)
- static void DestructiveMulAdd (byte [] x, int y, byte z)
- static int DivideByInt (ref BigInteger divident, int divisor, ref BigInteger quotient, ref int reminder)
- static string IntToStr (long value, int radix)
- static byte [] RemoveLeadingZeroBytes (byte [] data)
- static int ShiftLeft (BigInteger data, uint shift)
- static byte [] TrustedStripLeadingZeroInts (byte [] val)

- void FastCopy (ref BigInteger src, ref BigInteger dst)
- BigInteger GetCopy ( )
- BigInteger GetCopyAndInverse ( )
- int GetDataLen ( )

- char NibbleToHexChar (int b)

Detailed Description

This class represents an ASN.1 INTEGER built-in type. In this case, the values can be greater than 64 bits in size. This class is used in generated source code if the <bigInteger> qualifier is specified in a compiler configuration file.

Definition at line 38 of file BigInteger.cs

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

- BigInteger.cs

int mSign

The signum of this BigInteger: -1 for negative, 0 for zero, or 1 for positive. Note that the BigInteger zero *must* have a signum of 0. This is necessary to ensures that there is exactly one representation for each BigInteger value.

<serial> </serial>

Definition at line 54 of file BigInteger.cs

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

- BigInteger.cs

byte [] mValue

The magnitude of this BigInteger, in *big-endian* order: the zeroth element of this array is the most-significant byte of the magnitude. The magnitude must be "minimal" in that the most-significant byte (`mValue[0]`) must be non-zero. This is necessary to ensure that there is exactly one representation for each BigInteger value. Note that this implies that the BigInteger zero has a zero-length mValue array. Instances can share an array, since the array should be treated as immutable; this may be done for values that are identical except for the sign.

Definition at line 67 of file BigInteger.cs

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

- BigInteger.cs

BigInteger Add (BigInteger op)

Return the result of adding op to this BigInteger. Does not modify this object.

Table 3.739. Parameters

op	

Returns: .

BigInteger ()

The default constructor sets the big integer value object reference to null.

BigInteger (byte[] value, int sign)

This constructor creates a new big integer object and sets it to the byte[] value passed in.

Table 3.740. Parameters

value	String value
sign	Can be -1 for negative, 0 for zero, or 1 for positive.

BigInteger (System.String value)

This constructor creates a new big integer object and sets it to the string value passed in. String value may contain the prefix that describes the radix: 0x - hexadecimal, 0o - octal, 0b - binary. The string value without prefix assumes decimal value. The optional sign '-' may be specified at the beginning of the string to specify the negative value.

Table 3.741. Parameters

value	String value

BigInteger (System.Int64 value)

This constructor creates a new big integer object and sets it to the int value passed in.

Table 3.742. Parameters

value	Integer value

BigInteger (System.String value, int radix)

This constructor creates a new big integer object and sets it to the string value passed in. String value may contain the prefix that describes the radix: 0x - hexadecimal, 0o - octal, 0b - binary. The string value without prefix assumes decimal value. The optional sign '-' may be specified at the beginning of the string to specify the negative value.

Table 3.743. Parameters

value	String value
radix	Can be 16 for hexadecimal, 8 for octal, 2 for binary or 10 for decimal

int BitLength ()

Returns the number of bits in the minimal two's-complement representation of this BigInteger, excluding a sign bit. For positive BigIntegers, this is equivalent to the number of bits in the ordinary binary representation. (Computes (ceil(log2(this < 0 ? -this : this+1))).)

Returns: . number of bits in the minimal two's-complement representation of this BigInteger, excluding a sign bit.

virtual int CompareTo (BigInteger value)

This method compares this integer value to the given value.

Table 3.744. Parameters

value	The value to compare with the current object.

Returns: . -1 if this object is less than value, 0 if this object is equal to value 1 if this object is greater than value

virtual bool Equals (long value)

This method compares this integer value to the given value for equality.

Table 3.745. Parameters

value	The long value to compare with the current Object.

Returns: . `true` if the specified long value is equal to the current Object; otherwise, `false`.

override bool Equals (System.Object value)

This method compares this integer value to the given value for equality.

Table 3.746. Parameters

value	The Object to compare with the current Object. Object should be instance of BigInteger.

Returns: . `true` if the specified Object is equal to the current Object; otherwise, `false`.

byte [] GetData ()

This method provides the byte array representation of the integer value, in 2's complement form. The most significant byte is at index 0.

Returns: . byte array for integer value

override int GetHashCode ()

Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.

Returns: . A hash code for the current Object.

void Init (System.String val, int radix)

Translates the String representation of a Integer in the specified radix into a BigInteger. The String representation consists of an optional minus sign followed by a sequence of one or more digits in the specified radix. The String may not contain any extraneous Characters (whitespace, for example).

Table 3.747. Parameters

val	String representation of Integer.
radix	radix to be used in interpreting string value.

Table 3.748. Exceptions

System.FormatException	`val` is not a valid representation of a BigInteger in the specified radix, or invalid value of radix.

bool IsNegative ()

This method checks the Integer value is negative.

Returns: . `true` if negative; otherwise `false`

long LongValue ()

Converts this BigInteger to a long. This conversion is analogous to the conversion from long to int: if this BigInteger is too big to fit in a long, only the low-order 64 bits are returned. Note that this conversion can lose information about the overall magnitude of the BigInteger value as well as return a result with the opposite sign.

Returns: . this BigInteger converted to a long.

void SecureDelete ()

This function clears the current value (by overwritting with zeros). Than sets the value to zero, and passes the old value to garbage collector.

void SetData (byte[] ivalue)

This method sets the Integer value from byte array.

Table 3.749. Parameters

ivalue	byte array of the integer value

Returns: . Decoded integer value

See also: . <seealso cref=GetData To retrieve Byte Array

BigInteger Subtract (BigInteger op)

Subtract given op from this value and return the result. This object is not modified.

Table 3.750. Parameters

op	

Returns: .

System.String ToString (int radix)

Returns the String representation of this BigInteger in the given radix. If the radix is invalid, it will default to 10 (as is the case for `Int32.ToString`). The a minus sign is prepended if appropriate. This method is compatible with `BigInteger(String, int)` constructor.

Table 3.751. Parameters

radix	radix of the String representation.

Returns: . String representation of this BigInteger in the given radix.

See also: . <seealso cref=System.Int32.ToString Integer ToString methods

override System.String ToString ()

This method will return a string representation of the integer value. The format is the ASN.1 value format for this type..

Returns: . Stringified representation of the value

static implicit operator BigInteger (long value)

Overloaded implicit conversion operator for long to BigInteger value

static byte [] bitwiseAdd (byte[] a, byte[] b)

Return the result of a bitwise addition of a and b. The first byte is the most significant. The arrays may not have the same size.

Table 3.752. Parameters

a	
b	

Returns: .

static byte [] bitwiseSubtract (byte[] a, byte[] b)

Return the result of bitwise subtracting b from a. An imaginary borrow will be made, if necessary. You can be sure to avoid this by arranging for a > b. The resulting array may have leading zero bytes.

Table 3.753. Parameters

a	
b	

Returns: .

Com::Objsys::Asn1::Runtime::BooleanHolder class Reference

Public Attributes

- bool mValue

- BooleanHolder ( )
- BooleanHolder (bool value)

Detailed Description

A Holder class for a `Boolean` that is used to store "out" and "inout" parameters in methods. If a method has a `boolean` as an "out" or "inout" parameter, the programmer must pass an instance of `BooleanHolder` as the corresponding parameter in the method invocation; for "inout" parameters, the programmer must also fill the "in" value.

If myBooleanHolder is an instance of BooleanHolder,. the value stored in its value field can be accessed with `myBooleanHolder.mValue`.

Definition at line 43 of file BooleanHolder.cs

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

- BooleanHolder.cs

Member Data Documentation

bool mValue

This member variable is where the boolean value is stored.

Definition at line 47 of file BooleanHolder.cs

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

- BooleanHolder.cs

BooleanHolder ()

The default constructor for BooleanHolder class

BooleanHolder (bool value)

This constructor will intialize BooleanHolder class with specified boolean value.

Table 3.754. Parameters

value	`Boolean` value

Com::Objsys::Asn1::Runtime::Asn1DecodeBuffer::Buf struct Reference

Private Attributes

- int byteCount
- long offset
-
-

Detailed Description

Defines information for marking a buffer at a specific position. The usage pattern is to call MarkPos() first and then eventually ResetPos().

Definition at line 50 of file Asn1DecodeBuffer.cs

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

- Asn1DecodeBuffer.cs

Com::Objsys::Asn1::Runtime::Diag class Reference

-

- static Diag mInstance

Private Attributes

- System.IO.StreamWriter mPrintStream
- int mTraceLevel

- Diag ( )
- Diag (System.IO.StreamWriter ps)

- virtual bool IsEnabled ( )
- virtual bool IsEnabled (int traceLevel)
- virtual void Println (System.String s)
- virtual void Println (System.String s, int traceLevel)
- virtual bool SetEnabled (bool data)
- virtual int SetTraceLevel2 (int level)

- static void HexDump (byte [] bytes)
- static void HexDump (byte [] bytes, int traceLevel)
- static void HexDump (System.IO.Stream istrm, System.IO.StreamWriter ostrm)
- static Diag Instance ( )
- static void Prtln (System.String s)
- static void Prtln (System.String s, int traceLevel)
- static void Prtln (byte [] b, int offset, int nbytes, int tl)
- static void Prtln (byte [] b, int offset, int nbytes)
- static int SetTraceLevel (int level)

Detailed Description

This class is used for printing diagnostic messages for debugging the run-time components. It allows messages to be easily switched on and off.

Definition at line 32 of file Diag.cs

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

- Diag.cs

virtual bool IsEnabled ()

This method will enable the diagnostic message printing.

Returns: . `true` if enabled, else `false`

virtual bool IsEnabled (int traceLevel)

This method checks that given trace level message will be printed.

Table 3.755. Parameters

traceLevel	Trace Level

Returns: . `true` if enabled, else `false`

virtual void Println (System.String s)

This method prints a the diagnsotic message

Table 3.756. Parameters

s	diagnsotic message

virtual void Println (System.String s, int traceLevel)

This method prints a the diagnsotic message, if given trace level is enabled

Table 3.757. Parameters

s	diagnsotic message
traceLevel	Trace Level

virtual bool SetEnabled (bool data)

This method enables or disables the diagnostic message priting.

Table 3.758. Parameters

data	`true` for enabling printing; otherwise `false`

Returns: . The stat before setting this stat

virtual int SetTraceLevel2 (int level)

This method sets the trace level for this class

Table 3.759. Parameters

level	Trace Level

Returns: . Set trace level

static void HexDump (byte[] bytes)

This method prints a formatted hex dump for the contents of the given byte array to the standard output stream.

Table 3.760. Parameters

bytes	Byte array containg data to be dumped

static void HexDump (byte[] bytes, int traceLevel)

This method prints a formatted hex dump for the contents of the given byte array to the standard output stream, if the given trace level is enabled.

Table 3.761. Parameters

bytes	Byte array containg data to be dumped
traceLevel	Trace level

static void HexDump (System.IO.Stream istrm, System.IO.StreamWriter ostrm)

This method prints a formatted hex dump for the contents of the given input stream to the given output stream.

Table 3.762. Parameters

istrm	Input Stream containg data to be dumped
ostrm	Output Stream to which formatted data is to be written

static Diag Instance ()

This method provides the current instance of the Diag Class

Returns: . Current instance of the Diag class

static void Prtln (System.String s)

This method prints a the diagnsotic message to current Diag class instance.

Table 3.763. Parameters

s	diagnsotic message

static void Prtln (System.String s, int traceLevel)

This method prints a the diagnsotic message to current Diag class instance, if given trace level is enabled

Table 3.764. Parameters

s	diagnsotic message
traceLevel	Trace Level

static void Prtln (byte[] b, int offset, int nbytes, int tl)

This method prints a the hex dump of the given byte array to current Diag class instance, if given trace level is enabled

Table 3.765. Parameters

b	byte array containing data
offset	start offset in the byte array
nbytes	no of bytes to be printed
tl	trace level

static void Prtln (byte[] b, int offset, int nbytes)

This method prints a the hex dump of the given byte array to current Diag class instance

Table 3.766. Parameters

b	byte array containing data
offset	start offset in the byte array
nbytes	no of bytes to be printed

static int SetTraceLevel (int level)

This method sets the trace level for the current instance of the Diag Class

Table 3.767. Parameters

level	Trace Level

Returns: . Set trace level

System::Exception class Reference

System::IComparable class Reference

System::Collections::IEnumerator class Reference

Com::Objsys::Asn1::Runtime::Asn1EncodeBuffer::Inte class Reference

Private Attributes

- Asn1EncodeBuffer encodeBuffer

-
-
-
-
-

- override void Flush ( )
- InternalOutputStream (Asn1EncodeBuffer buffer)
- override int Read (byte [] buffer, int offset, int count)
- override long Seek (long offset, SeekOrigin origin)
- override void SetLength (long value)
- override void Write (byte [] buffer, int offset, int count)
- override void WriteByte (byte value)

Com::Objsys::Asn1::Runtime::IntHolder class Reference

Public Attributes

- int mValue

- IntHolder ( )
- IntHolder (int value)

Detailed Description

A Holder class for an `int` that is used to store "out" and "inout" parameters in methods. If a method has an `int` as an "out" or "inout" parameter, the programmer must pass an instance of `IntHolder` as the corresponding parameter in the method invocation; for "inout" parameters, the programmer must also fill the "in" value.

If myIntHolder is an instance of IntHolder,. the value stored in its value field can be accessed with `myIntHolder.mValue`.

Definition at line 42 of file IntHolder.cs

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

- IntHolder.cs

Member Data Documentation

int mValue

This member variable is where the int value is stored.

Definition at line 45 of file IntHolder.cs

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

- IntHolder.cs

IntHolder ()

The default constructor for IntHolder class

IntHolder (int value)

This constructor will intialize IntHolder class with specified int value.

Table 3.768. Parameters

value	`int` value

Com::Objsys::Asn1::Runtime::Asn1OpenType::SaxHar class Reference

Private Attributes

- bool mCaptureOuterElem

- Asn1OpenType mParentClass
- Asn1XerEncodeBuffer mEncodeBuffer
- void InitBlock (Asn1OpenType mParentClass)
- bool IsEmptyElement (System.String qname)
- SaxHandler (Asn1OpenType mParentClass)
- SaxHandler (Asn1OpenType mParentClass, bool captureOuterElem)
- override void Characters (System.Char [] ch, int start, int length)
- override void EndElement (System.String namespaceURI, System.String localName, System.String qName)
- override void StartElement (System.String namespaceURI, System.String localName, System.String qName, XmlAttributes atts)

Detailed Description

This class extends the Asn1XerSaxHandler class to add items specific to ASN.1 XER encoding.

Definition at line 912 of file Asn1OpenType.cs

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

- Asn1OpenType.cs

override void Characters (System.Char[] ch, int start, int length)

This method manage the notification when Characters element were found.

Table 3.769. Parameters

ch	The array with the characters founds
start	The index of the first position of the characters found
length	Specify how many characters must be read from the array

override void EndElement (System.String namespaceURI, System.String localName, System.String qName)

This method manage the notification when the end element node were found

Table 3.770. Parameters

namespaceURI	The namespace URI of the element
localName	The local name of the element
qName	The long name (qualify name) of the element

override void StartElement (System.String name-spaceURI, System.String localName, System.String qName, XmlAttributes atts)

This method manage the event when a start element node were found

Table 3.771. Parameters

namespaceURI	The namespace uri of the element tag
localName	The local name of the element
qName	The Qualify (long) name of the element
atts	The list of attributes of the element

System::IO::Stream class Reference

Com::Objsys::Asn1::Runtime::StringBufferExt class Reference

- static StringBuilder Replace (StringBuilder sbuf, int start, int end, String str)

Detailed Description

This class provides the additional functionality to StringBuilder class

Definition at line 9 of file StringBufferExt.cs

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

- StringBufferExt.cs

static StringBuilder Replace (StringBuilder sbuf, int start, int end, String str)

Replaces the characters in a substring of given StringBuilder with characters in the specified String. The substring begins at the specified start and extends to the character at index end - 1 or to the end of the StringBuilder if no such character exists. First the characters in the substring are removed and then the specified String is inserted at start. (The specifed StringBuilder will be lengthened to accommodate the specified String if necessary.)

Table 3.772. Parameters

sbuf	StringBuilder that will have contents.
start	The beginning index, inclusive.
end	The ending index, exclusive.
str	String that will replace previous contents.

Returns: . The replaced string builder.

Com::Objsys::Asn1::Runtime::Tokenizer class Reference

Private Attributes

- char [] chars

  Char representation of the String to tokenize.

- long currentPos

  Position over the string.

- string delimiters
- bool includeDelims

  Include demiliters in the results.

-
-

- bool HasMoreTokens ( )
- bool MoveNext ( )
- System.String NextToken ( )
- System.String NextToken (System.String delimiters)
- string RemainingString ( )
- void Reset ( )
- Tokenizer (System.String source)
- Tokenizer (System.String source, System.String delimiters)
- Tokenizer (System.String source, System.String delimiters, bool includeDelims)

- System.String NextToken (char [] delimiters)

Detailed Description

The class performs token processing in strings

Definition at line 8 of file Tokenizer.cs

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

- Tokenizer.cs

bool HasMoreTokens ()

Determines if there are more tokens to return from the source string

Returns: . True or false, depending if there are more tokens

bool MoveNext ()

Performs the same action as HasMoreTokens.

Returns: . True or false, depending if there are more tokens

System.String NextToken ()

Returns the next token from the token list

Returns: . The string value of the token

System.String NextToken (System.String delimiters)

Returns the next token from the source string, using the provided token delimiters

Table 3.773. Parameters

delimiters	String containing the delimiters to use

Returns: . The string value of the token

string RemainingString ()

Returns the rest of the string from current position.

Returns: . rest of the string

void Reset ()

Does nothing.

Tokenizer (System.String source)

Initializes a new class instance with a specified string to process

Table 3.774. Parameters

source	String to tokenize

Tokenizer (System.String source, System.String delimiters)

Initializes a new class instance with a specified string to process and the specified token delimiters to use

Table 3.775. Parameters

source	String to tokenize
delimiters	String containing the delimiters

Tokenizer (System.String source, System.String delimiters, bool includeDelims)

Initializes a new class instance with a specified string to process, the specified token delimiters to use, and whether the delimiters must be included in the results.

Table 3.776. Parameters

source	String to tokenize
delimiters	String containing the delimiters
includeDelims	Determines if delimiters are included in the results.

Chapter 4. File Documentation

Asn18BitCharString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn18BitCharString

Namespaces

- struct Com::Objsys::Asn1::Runtime
- struct System

Detailed Description

Definition in file Asn18BitCharString.cs

Asn1BigInteger.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1BigInteger

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1BigInteger.cs

Asn1BitString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1BitString

Namespaces

- struct Com::Objsys::Asn1::Runtime
- struct System::Collections::Generic
- struct System::Runtime::InteropServices
- struct System::Text

Detailed Description

Definition in file Asn1BitString.cs

Asn1BMPString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1BMPString
- struct Com::Objsys::Asn1::Runtime::Asn1BMPStringCharset

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1BMPString.cs

Asn1Boolean.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Boolean

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Boolean.cs

Asn1CharRange.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1CharRange

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1CharRange.cs

Asn1CharSet.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1CharSet

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1CharSet.cs

Asn1CharString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1CharString

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1CharString.cs

Asn1Choice.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Choice

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Choice.cs

Asn1ChoiceExt.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1ChoiceExt

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1ChoiceExt.cs

Asn1ConsVioException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1ConsVioException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1ConsVioException.cs

Asn1DecodeBuffer.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1DecodeBuffer
- struct Com::Objsys::Asn1::Runtime::Asn1DecodeBuffer::BufferPos

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1DecodeBuffer.cs

Asn1DiscreteCharSet.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1DiscreteCharSet

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1DiscreteCharSet.cs

Asn1EncodeBuffer.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1EncodeBuffer
- struct Com::Objsys::Asn1::Runtime::Asn1EncodeBuffer::InternalOutputStream

Namespaces

- struct Com::Objsys::Asn1::Runtime
- struct System::IO

Detailed Description

Definition in file Asn1EncodeBuffer.cs

Asn1EndOfBufferException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1EndOfBufferException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1EndOfBufferException.cs

Asn1Enumerated.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Enumerated

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Enumerated.cs

Asn1Exception.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Exception

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Exception.cs

Asn1GeneralizedTime.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1GeneralizedTime

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1GeneralizedTime.cs

Asn1GeneralString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1GeneralString

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1GeneralString.cs

Asn1GraphicString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1GraphicString

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1GraphicString.cs

Asn1IA5String.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1IA5String
- struct Com::Objsys::Asn1::Runtime::Asn1IA5StringCharset

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1IA5String.cs

Asn1InputStream.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1InputStream

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1InputStream.cs

Asn1Integer.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Integer

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Integer.cs

Asn1InvalidArgException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1InvalidArgException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1InvalidArgException.cs

Asn1InvalidChoiceOptionException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1InvalidChoiceOptionException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1InvalidChoiceOptionException.cs

Asn1InvalidEnumException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1InvalidEnumException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1InvalidEnumException.cs

Asn1InvalidLengthException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1InvalidLengthException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1InvalidLengthException.cs

Asn1InvalidObjectIDException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1InvalidObjectIDException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1InvalidObjectIDException.cs

Asn1MessageBuffer.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1MessageBuffer

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1MessageBuffer.cs

Asn1MissingRequiredException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1MissingRequiredException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1MissingRequiredException.cs

Asn1NamedEventHandler.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1NamedEventHandler

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1NamedEventHandler.cs

Asn1Null.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Null

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Null.cs

Asn1NumericString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1NumericString

- struct Com::Objsys::Asn1::Runtime::Asn1NumericStringCharset

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1NumericString.cs

Asn1ObjectDescriptor.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1ObjectDescriptor

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1ObjectDescriptor.cs

Asn1ObjectIdentifier.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1ObjectIdentifier

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1ObjectIdentifier.cs

Asn1OctetString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1OctetString

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1OctetString.cs

Asn1OpenExt.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1OpenExt

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1OpenExt.cs

Asn1OpenType.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1OpenType
- struct Com::Objsys::Asn1::Runtime::Asn1OpenType::SaxHandler

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1OpenType.cs

Asn1OutputStream.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1OutputStream

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1OutputStream.cs

Asn1PrintableString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1PrintableString
- struct Com::Objsys::Asn1::Runtime::Asn1PrintableStringCharset

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1PrintableString.cs

Asn1Real.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Real

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Real.cs

Asn1Real10.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Real10

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Real10.cs

Asn1RelativeOID.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1RelativeOID

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1RelativeOID.cs

Asn1SeqOrderException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1SeqOrderException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1SeqOrderException.cs

Asn1Status.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Status

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Status.cs

Asn1T61String.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1T61String

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1T61String.cs

Asn1Tag.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Tag

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Tag.cs

Asn1Time.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Time

  <summary>

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Time.cs

Asn1TraceHandler.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1TraceHandler

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1TraceHandler.cs

Asn1Type.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Type

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Type.cs

Asn1TypeIF.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1TypeIF

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1TypeIF.cs

Asn1UniversalString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1UniversalString

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1UniversalString.cs

Asn1UTCTime.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1UTCTime

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1UTCTime.cs

Asn1UTF8String.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1UTF8String

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1UTF8String.cs

Asn1Util.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Util

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Util.cs

Asn1Value.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1Value

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1Value.cs

Asn1ValueParseException.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1ValueParseException

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1ValueParseException.cs

Asn1VarWidthCharString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1VarWidthCharString

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1VarWidthCharString.cs

Asn1VideotexString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1VideotexString

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1VideotexString.cs

Asn1VisibleString.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Asn1VisibleString

- struct Com::Objsys::Asn1::Runtime::Asn1VisibleStringCharset

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Asn1VisibleString.cs

AssemblyInfo.cs File Reference

Namespaces

- struct System::Reflection
- struct System::Runtime::CompilerServices

Detailed Description

Definition in file AssemblyInfo.cs

BigInteger.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::BigInteger

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file BigInteger.cs

BooleanHolder.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::BooleanHolder

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file BooleanHolder.cs

Diag.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Diag

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Diag.cs

IntHolder.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::IntHolder

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file IntHolder.cs

StringBufferExt.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::StringBufferExt

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file StringBufferExt.cs

Tokenizer.cs File Reference

Classes

- struct Com::Objsys::Asn1::Runtime::Tokenizer

Namespaces

- struct Com::Objsys::Asn1::Runtime

Detailed Description

Definition in file Tokenizer.cs