System.Xml.XmlTextReader Class

```
[ILASM]
.class public XmlTextReader extends System.Xml.XmlReader

[C#]
public class XmlTextReader: XmlReader
```

Assembly Info:

- *Name:* System.Xml
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 1.0.x.x
- *Attributes:*
  - CLSCompliantAttribute(true)

Type Attributes:

- DefaultMemberAttribute("Item") [*Note:* This attribute requires the RuntimeInfrastructure library.]

Summary

Represents a reader that provides fast, non-cached, forward-only access to XML data.

Inherits From: System.Xml.XmlReader

Library: XML

Thread Safety: All public static members of this type are safe for multithreaded operations. No instance members are guaranteed to be thread safe.

Description

This class provides forward-only, read-only access to a character stream of XML data. This class enforces the rules of well-formed XML but does not perform data validation.

This class implements the **System.Xml.XmlReader** class and conforms to the W3C Extensible Markup Language (XML) 1.0 and the Namespaces in XML recommendations.

A given set of XML data is modeled as a tree of nodes. The different types of nodes are specified in the **System.Xml.XmlNodeType** enumeration. The current node refers to the node on which the reader is positioned. The reader is advanced using any of the "read" or

"moveto" methods. The following table lists the node properties exposed for the current node.

Property	Description
AttributeCount	The number of attributes on the node.
BaseUri	The base URI of the node.
Depth	The depth of the node in the tree.
HasAttributes	Whether the node has attributes. (Inherited from **System.Xml.XmlReader**)
HasValue	Whether the node can have a text value.
IsDefault	Whether an **Attribute** node was generated from the default value defined in the DTD or schema.
IsEmptyElement	Whether an **Element** node is empty.
LocalName	The local name of the node.
Name	The qualified name of the node, equal to **Prefix**:**LocalName**.
NamespaceUri	The URI defining the namespace associated with the node.
NodeType	The **System.Xml.XmlNodeType** of the node.
Prefix	A shorthand reference to the namespace associated with the node.
QuoteChar	The quotation mark character used to enclose the value of an attribute.
Value	The text value of the node.
XmlLang	The `xml:lang` scope within which the node resides.

This class does not expand default attributes or resolve general entities. Any general entities encountered are returned as a single empty **EntityReference** node.

This class checks that a Document Type Definition (DTD) is well-formed, but does not validate using the DTD.

To read strongly typed data, use the **System.Xml.XmlConvert** class.

This class throws a **System.Xml.XmlException** on XML parse errors. After an exception is thrown, the state of the reader is not predictable. For example, the reported node type may be different than the actual node type of the current node.

XmlTextReader() Constructor

[ILASM]
```
family rtspecialname specialname instance void .ctor()
```

[C#]
```
protected XmlTextReader()
```

Summary

Constructs a new instance of the **System.Xml.XmlTextReader** class.

XmlTextReader(System.Xml.XmlNameTable) Constructor

```
[ILASM]
family rtspecialname specialname instance void .ctor(class
System.Xml.XmlNameTable nt)

[C#]
protected XmlTextReader(XmlNameTable nt)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified name table.

Parameters

Parameter	Description
nt	The **System.Xml.XmlNameTable** to use.

Description

The **System.Xml.XmlTextReader** public constructors call this constructor to initialize the following **System.Xml.XmlTextReader** properties to the specified values. Derived classes can call this constructor to incorporate this behavior.

Property	Value
Namespaces	**true**
NameTable	*nt*
Normalization	**false**
ReadState	**System.Xml.ReadState.Initial**
WhitespaceHandling	**System.Xml.WhitespaceHandling.All**
XmlLang	**System.String.Empty**
XmlSpace	**System.Xml.XmlSpace.None**
XmlResolver	new **System.Xml.XmlUrlResolver**()

Exceptions

Exception	Condition

System.ArgumentNullException	*nt* is **null**.

1
2
3

XmlTextReader(System.IO.Stream) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(class
System.IO.Stream input)

[C#]
public XmlTextReader(Stream input)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified stream.

Parameters

Parameter	Description
input	The **System.IO.Stream** containing the XML data to read.

Description

This constructor is equivalent to **System.Xml.XmlTextReader**(**System.String.Empty**, *input*, new **System.Xml.NameTable**()).

Exceptions

Exception	Condition
System.ArgumentNullException	*input* is **null**.

XmlTextReader(System.String, System.IO.Stream) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(string
url, class System.IO.Stream input)

[C#]
public XmlTextReader(string url, Stream input)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified URL and stream.

Parameters

Parameter	Description
url	A **System.String** specifying the URL to use for resolving external resources.
input	The **System.IO.Stream** containing the XML data to read.

Description

This constructor is equivalent to **System.Xml.XmlTextReader**(*url*, *input*, new **System.Xml.NameTable**()).

Exceptions

Exception	Condition
System.ArgumentNullException	*input* is **null**.

XmlTextReader(System.IO.Stream, System.Xml.XmlNameTable) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(class
System.IO.Stream input, class System.Xml.XmlNameTable nt)

[C#]
public XmlTextReader(Stream input, XmlNameTable nt)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified stream and name table.

Parameters

Parameter	Description
input	The **System.IO.Stream** containing the XML data to read.
nt	The **System.Xml.XmlNameTable** to use.

Description

This constructor is equivalent to **System.Xml.XmlTextReader**(**System.String.Empty**, *input*, *nt*).

Exceptions

Exception	Condition
System.ArgumentNullException	*input* or *nt* is **null**.

XmlTextReader(System.String, System.IO.Stream, System.Xml.XmlNameTable) Constructor

[ILASM]
```
public rtspecialname specialname instance void .ctor(string
url, class System.IO.Stream input, class
System.Xml.XmlNameTable nt)
```

[C#]
```
public XmlTextReader(string url, Stream input, XmlNameTable
nt)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified URL, stream, and name table.

Parameters

Parameter	Description
url	A **System.String** specifying the URL to use for resolving external resources.
input	The **System.IO.Stream** containing the XML data to read.
nt	The **System.Xml.XmlNameTable** to use.

Description

This constructor calls **System.Xml.XmlTextReader**(*nt*) to initialize properties of the class.

System.Xml.XmlTextReader.Encoding is set to the encoding corresponding to the byte-order mark at the beginning of the stream or, if no byte-order mark is found, UTF-8.

System.Xml.XmlTextReader.BaseURI is set to *url* or, if *url* is **null**, to **System.String.Empty**.

Exceptions

Exception	Condition
System.ArgumentNullException	*input* or *nt* is **null**.

1
2
3

XmlTextReader(System.IO.TextReader) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(class
System.IO.TextReader input)

[C#]
public XmlTextReader(TextReader input)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified **System.IO.TextReader**.

Parameters

Parameter	Description
input	A **System.IO.TextReader**, set to the correct encoding, containing the XML data to read.

Description

This constructor is equivalent to **System.Xml.XmlTextReader**(**System.String.Empty**, *input*, new **System.Xml.NameTable**()).

XmlTextReader(System.String, System.IO.TextReader) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(string
url, class System.IO.TextReader input)

[C#]
public XmlTextReader(string url, TextReader input)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified URL and **System.IO.TextReader**.

Parameters

Parameter	Description
url	A **System.String** specifying the URL to use for resolving external resources.
input	A **System.IO.TextReader**, set to the correct encoding, containing the XML data to read.

Description

This constructor is equivalent to **System.Xml.XmlTextReader**(*url*, *input*, new **System.Xml.NameTable**()).

XmlTextReader(System.IO.TextReader, System.Xml.XmlNameTable) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(class
System.IO.TextReader input, class System.Xml.XmlNameTable
nt)

[C#]
public XmlTextReader(TextReader input, XmlNameTable nt)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified **System.IO.TextReader**, and name table.

Parameters

Parameter	Description
input	A **System.IO.TextReader**, set to the correct encoding, containing the XML data to read.
nt	The **System.Xml.XmlNameTable** to use.

Description

This constructor is equivalent to **System.Xml.XmlTextReader**(**System.String.Empty**, *input*, *nt*).

Exceptions

Exception	Condition
System.ArgumentNullException	*nt* is **null**.

XmlTextReader(System.String, System.IO.TextReader, System.Xml.XmlNameTable) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(string
url, class System.IO.TextReader input, class
System.Xml.XmlNameTable nt)

[C#]
public XmlTextReader(string url, TextReader input,
XmlNameTable nt)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified URL, **System.IO.TextReader**, and name table.

Parameters

Parameter	Description
url	A **System.String** specifying the URL to use for resolving external resources.
input	A **System.IO.TextReader**, set to the correct encoding, containing the XML data to read.
nt	The **System.Xml.XmlNameTable** to use.

Description

If *input* is **null**, a **System.Xml.XmlException** is thrown when the **System.Xml.XmlTextReader.Read** method is called.

This constructor calls **System.Xml.XmlTextReader**(*nt*) to initialize properties of the class.

System.Xml.XmlTextReader.BaseURI is set to *url* or, if *url* is **null**, to **System.String.Empty**.

[*Note:* To pass a user defined string that represents full, well-formed XML data, create a **System.IO.StringReader** with the string and pass the **System.IO.StringReader** to this constructor.]

Exceptions

Exception	Condition
System.ArgumentNullException	*nt* is **null**.

1
2
3

XmlTextReader(System.IO.Stream, System.Xml.XmlNodeType, System.Xml.XmlParserContext) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(class
System.IO.Stream xmlFragment, valuetype
System.Xml.XmlNodeType fragType, class
System.Xml.XmlParserContext context)

[C#]
public XmlTextReader(Stream xmlFragment, XmlNodeType
fragType, XmlParserContext context)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified stream containing an XML fragment.

Parameters

Parameter	Description
xmlFragment	The **System.IO.Stream** containing the XML fragment to parse.
fragType	The **System.Xml.XmlNodeType** of the XML fragment. This also determines what the fragment string can contain. (See table below.)
context	The **System.Xml.XmlParserContext** in which the *xmlFragment* is to be parsed, or **null**.

Description

The following table lists valid values for *fragType*.

XmlNodeType	Fragment May Contain
Element	Any valid element content (for example, any combination of elements, comments, processing instructions, CDATA sections, text, and entity references).
Attribute	The value of an attribute (the part inside the quotes).
Document	The contents of an entire XML document; document level rules are enforced.

[*Note:* If the XML fragment is an element or attribute, the root level rules for well-formed XML documents are not enforced.] This constructor calls **System.Xml.XmlTextReader**(*context*.**NameTable**) or, if *context* is **null**, **System.Xml.XmlTextReader**(**new System.Xml.NameTable**()) to initialize properties of the class. Afterwards, the following **System.Xml.XmlTextReader** properties are set to the specified values.

Property	Value
BaseUri	*context*.**BaseURI** or, if *context* is **null**, **System.String.Empty**.
Encoding	*context*.**Encoding** or, if *context* or *context*.**Encoding** is **null**, the encoding corresponding to the byte-order mark at the beginning of the stream or, if no byte-order mark is found, UTF-8.
Namespaces	**false**.
XmlLang	If *context* is not **null**, *context*.**XmlLang**. If *context* is **null**, this property is not changed.
XmlSpace	If *context* is not **null**, *context*.**XmlSpace**. If *context* is **null**, this property is not changed.

Exceptions

Exception	Condition
System.ArgumentNullException	*xmlFragment* is **null**.
System.Xml.XmlException	*fragType* is not an **Element**, **Attribute**, or **Document System.Xml.XmlNodeType**.

XmlTextReader(System.String, System.Xml.XmlNodeType, System.Xml.XmlParserContext) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(string
xmlFragment, valuetype System.Xml.XmlNodeType fragType,
class System.Xml.XmlParserContext context)

[C#]
public XmlTextReader(string xmlFragment, XmlNodeType
fragType, XmlParserContext context)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified XML fragment.

Parameters

Parameter	Description
xmlFragment	A **System.String** containing the XML fragment to parse.
fragType	The **System.Xml.XmlNodeType** of the XML fragment. This also determines what the fragment string can contain. (See table below.)
context	The **System.Xml.XmlParserContext** in which the *xmlFragment* is to be parsed, or **null**.

Description

The following table lists valid values for *fragType* and how the reader will parse each of the different node types.

XmlNodeType	Fragment May Contain
Element	Any valid element content (for example, any combination of elements, comments, processing instructions, CDATA sections, text, and entity references).
Attribute	The value of an attribute (the part inside the quotes).
Document	The contents of an entire XML document; document level rules are enforced.

[*Note:* If the XML fragment is an element or attribute, root level rules

for well-formed XML documents are not enforced.

This constructor can handle strings returned from **System.Xml.XmlTextReader.ReadInnerXml**.] This constructor calls **System.Xml.XmlTextReader**(*context*.**NameTable**) or, if *context* is **null**, **System.Xml.XmlTextReader**(**new System.Xml.NameTable**()) to initialize properties of the class. Following this call, **System.Xml.XmlTextReader.Namespaces** is set to **false** and, if *context* is not **null**, the following **System.Xml.XmlTextReader** properties are set to the specified values.

Property	Value
BaseUri	*context*.**BaseURI** or, if *context* is **null**, **System.String.Empty**.
Encoding	*context*.**Encoding** or, if *context* or *context*.**Encoding** is **null**, UTF-8.
Namespaces	**false**.
XmlLang	If *context* is not **null**, *context*.**XmlLang**. If *context* is **null**, this property is not changed.
XmlSpace	If *context* is not **null**, *context*.**XmlSpace**. If *context* is **null**, this property is not changed.

Exceptions

Exception	Condition
System.ArgumentNullException	*xmlFragment* is **null**.
System.Xml.XmlException	*fragType* is not an **Element**, **Attribute**, or **Document System.Xml.XmlNodeType**.

XmlTextReader(System.String) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(string
url)

[C#]
public XmlTextReader(string url)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified file.

Parameters

Parameter	Description
url	A **System.String** specifying the URL for the file containing the XML data.

Description

This constructor is equivalent to **System.Xml.XmlTextReader**(*url*, new **System.Xml.NameTable**()).

Exceptions

Exception	Condition
System.Xml.XmlException	*url* is **null**.

XmlTextReader(System.String, System.Xml.XmlNameTable) Constructor

```
[ILASM]
public rtspecialname specialname instance void .ctor(string
url, class System.Xml.XmlNameTable nt)

[C#]
public XmlTextReader(string url, XmlNameTable nt)
```

Summary

Constructs and initializes a new instance of the **System.Xml.XmlTextReader** class with the specified file and name table.

Parameters

Parameter	Description
url	A **System.String** specifying the URL for the file containing the XML data to read.
nt	The **System.Xml.XmlNameTable** to use.

Description

This constructor calls **System.Xml.XmlTextReader**(*nt*) to initialize properties of the class.

Exceptions

Exception	Condition
System.ArgumentNullException	*nt* is **null**.
System.Xml.XmlException	*url* is **null**.

XmlTextReader.Close() Method

```
[ILASM]
.method public hidebysig virtual void Close()

[C#]
public override void Close()
```

Summary

Changes the **System.Xml.XmlTextReader.ReadState** to **Closed**.

Description

This method releases any resources allocated by the current instance, changes the **System.Xml.XmlTextReader.ReadState** to **System.Xml.ReadState.Closed**, and calls the **Close** method of any underlying **System.IO.Stream** or **System.IO.TextReader** instance.

[*Note:* This method overrides **System.Xml.XmlReader.Close**.]

XmlTextReader.GetAttribute(System.Int32) Method

```
[ILASM]
.method public hidebysig virtual string GetAttribute(int32
i)

[C#]
public override string GetAttribute(int i)
```

Summary

Returns the value of the attribute with the specified index relative to the containing element.

Parameters

Parameter	Description
i	A **System.Int32** specifying the zero-based index of the attribute relative to the containing element.

Return Value

A **System.String** containing the value of the specified attribute.

Description

This method does not move the reader.

[*Note:* This method overrides **System.Xml.XmlReader.GetAttribute**.]

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	*i* is less than 0, or greater than or equal to the **System.Xml.XmlTextReader.AttributeCoun** of the containing element.[*Note:* **System.Xml.XmlTextReader.AttributeCoun** returns zero for all node types except **Attribute**, **DocumentType**, **Element**, and **XmlDeclaration**. Therefore, this exception is thrown if the reader is not positioned on one o these node types.]

Example

See the **System.Xml.XmlTextReader.GetAttribute**(**String**, **String**) method for an example using all three overloads of this method.

XmlTextReader.GetAttribute(System.String, System.String) Method

```
[ILASM]
.method public hidebysig virtual string GetAttribute(string
localName, string namespaceURI)

[C#]
public override string GetAttribute(string localName,
string namespaceURI)
```

Summary

Returns the value of the attribute with the specified local name and namespace URI.

Parameters

Parameter	Description
localName	A **System.String** specifying the local name of the attribute.
namespaceURI	A **System.String** specifying the namespace URI of the attribute.

Return Value

A **System.String** containing the value of the specified attribute, or **null** if the attribute is not found. If *localname* is **null**, **null** is returned.

Description

If *namespaceURI* is **null**, the local namespace is searched for *localName*.

This method does not move the reader.

[*Note:* This method overrides **System.Xml.XmlReader.GetAttribute**.]

Example

This example writes the value of the attributes from the following XML fragment to the console:

```
<test xmlns:dt="urn:datatypes" dt:type="int"/>
```

The second attribute value is retrieved using all three overloads of this method.

[C#]

```
using System;
using System.Xml;

public class Reader {

  public static void Main() {

    string xmlFragment = @"<test xmlns:dt=""urn:datatypes""
                            dt:type=""int""/>";

    NameTable nameTable = new NameTable();
    XmlNamespaceManager xmlNsMan = new
         XmlNamespaceManager(nameTable);
    XmlParserContext xmlPContext = new
         XmlParserContext(null, xmlNsMan,
                          null, XmlSpace.None);
    XmlTextReader xmlTReader = new
         XmlTextReader(xmlFragment,XmlNodeType.Element,
                       xmlPContext);

    xmlTReader.Read();
    Console.WriteLine("{0}", xmlTReader.GetAttribute(0));

    string str1 = xmlTReader.GetAttribute(1);
    string str2 = xmlTReader.GetAttribute("dt:type");
    string str3 = xmlTReader.GetAttribute("type",
                                          "urn:datatypes");
    Console.WriteLine("{0} - {1} - {2}",
                      str1, str2, str3);
  }
}
```

The output is

```
urn:datatypes

int - int - int
```

XmlTextReader.GetAttribute(System.String) Method

```
[ILASM]
.method public hidebysig virtual string GetAttribute(string
name)

[C#]
public override string GetAttribute(string name)
```

Summary

Returns the value of the attribute with the specified qualified name.

Parameters

Parameter	Description
name	A **System.String** specifying the qualified name of the attribute.

Return Value

A **System.String** containing the value of the specified attribute, or **null** if the attribute is not found. If *name* is **null**, **null** is returned.

Description

This method does not move the reader.

[*Note:* If the reader is positioned on a **DocumentType** node, this method can be used to get the PUBLIC and SYSTEM literals.

This method overrides **System.Xml.XmlReader.GetAttribute**.]

Example

See the **System.Xml.XmlTextReader.GetAttribute**(**String**, **String**) method for an example using all three overloads of this method.

XmlTextReader.GetRemainder() Method

```
[ILASM]
.method public hidebysig instance class
System.IO.TextReader GetRemainder()

[C#]
public TextReader GetRemainder()
```

Summary

Returns the remainder of the buffered XML.

Return Value

The **System.IO.StringReader** attached to the XML.

Description

This method calls the **System.Xml.XmlTextReader.Close** method, and then resets the **System.Xml.XmlTextReader.ReadState** to **System.Xml.ReadState.EndOfFile**.

[*Note:* Because **System.Xml.XmlTextReader** performs a buffered read operation, it must be able to return the remainder of the unused buffer so that no data is lost. For example, this allows protocols (such as multi-part MIME) to package XML in the same stream.]

XmlTextReader.LookupNamespace(System.String) Method

```
[ILASM]
.method public hidebysig virtual string
LookupNamespace(string prefix)

[C#]
public override string LookupNamespace(string prefix)
```

Summary

Resolves a namespace prefix in the scope of the current element.

Parameters

Parameter	Description
prefix	A **System.String** specifying the prefix whose namespace URI is to be resolved. To return the default namespace, specify **System.String.Empty**.

Return Value

A **System.String** containing the namespace URI to which the prefix maps. If **System.Xml.XmlTextReader.Namespaces** is **false**, *prefix* is not in **System.Xml.XmlTextReader.NameTable**, or no matching namespace is found, **null** is returned.

Description

[*Note:* In the following XML, if the reader is positioned on the `href` attribute, the prefix "a" is resolved by calling **System.Xml.XmlTextReader.LookupNamespace**("a"). The returned string is "urn:456".

```
<root xmlns:a="urn:456">


<item>


<ref href="a:b"/>


</item>


</root>
```

This method overrides
System.Xml.XmlReader.LookupNamespace.]

Exceptions

Exception	Condition
System.ArgumentNullException	The **System.Xml.XmlTextReader.Namespaces** property of the current instance is **true** and *prefix* is **null**.

XmlTextReader.MoveToAttribute(System.Int32) Method

```
[ILASM]
.method public hidebysig virtual void MoveToAttribute(int32
i)

[C#]
public override void MoveToAttribute(int i)
```

Summary

Moves the position of the current instance to the attribute with the specified index relative to the containing element.

Parameters

Parameter	Description
i	A **System.Int32** specifying the zero-based index of the attribute relative to the containing element.

Description

After calling this method, the **System.Xml.XmlTextReader.Name**, **System.Xml.XmlTextReader.NamespaceURI**, and **System.Xml.XmlTextReader.Prefix** properties reflect the properties of the new attribute.

[*Note:* This method overrides **System.Xml.XmlReader.MoveToAttribute**.]

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	*i* is less than 0 or greater than or equal to the **System.Xml.XmlTextReader.AttributeCoun** of the containing element.[*Note:* **System.Xml.XmlTextReader.AttributeCoun** returns zero for all node types except **Attribute**, **DocumentType**, **Element**, and **XmlDeclaration**. Therefore, this exception is thrown if the reader is not positioned on one o these node types.]

1
2
3

XmlTextReader.MoveToAttribute(System.String, System.String) Method

```
[ILASM]
.method public hidebysig virtual bool
MoveToAttribute(string localName, string namespaceURI)

[C#]
public override bool MoveToAttribute(string localName,
string namespaceURI)
```

Summary

Moves the position of the current instance to the attribute with the specified local name and namespace URI.

Parameters

Parameter	Description
localName	A **System.String** specifying the local name of the attribute.
namespaceURI	A **System.String** specifying the namespace URI of the attribute.

Return Value

A **System.Boolean** where **true** indicates the attribute was found. If *localname* is **null**, or the attribute was not found, **false** is returned and the position of the reader does not change.

Description

If *namespaceURI* is **null**, the local namespace is searched for *localName*.

After calling this method, the **System.Xml.XmlTextReader.Name**, **System.Xml.XmlTextReader.NamespaceURI**, and **System.Xml.XmlTextReader.Prefix** properties reflect the properties of the new attribute.

[*Note:* This method overrides **System.Xml.XmlReader.MoveToAttribute**.]

XmlTextReader.MoveToAttribute(System.String) Method

```
[ILASM]
.method public hidebysig virtual bool
MoveToAttribute(string name)

[C#]
public override bool MoveToAttribute(string name)
```

Summary

Moves the position of the current instance to the attribute with the specified qualified name.

Parameters

Parameter	Description
name	A **System.String** specifying the qualified name of the attribute.

Return Value

A **System.Boolean** where **true** indicates the attribute was found. If *name* is **null**, or the attribute was not found, **false** is returned and the position of the reader does not change.

Description

After calling this method, the **System.Xml.XmlTextReader.Name**, **System.Xml.XmlTextReader.NamespaceURI**, and **System.Xml.XmlTextReader.Prefix** properties reflect the properties of the new attribute.

[*Note:* This method overrides **System.Xml.XmlReader.MoveToAttribute**.]

XmlTextReader.MoveToElement() Method

```
[ILASM]
.method public hidebysig virtual bool MoveToElement()

[C#]
public override bool MoveToElement()
```

Summary

Moves the position of the current instance to the node that contains the current **Attribute** node.

Return Value

A **System.Boolean** where **true** indicates the reader was moved; **false** indicates the reader was not positioned on an **Attribute** node and therefore was not moved.

Description

[*Note:* The **DocumentType**, **Element**, and **XmlDeclaration** node types can contain attributes.

This method overrides **System.Xml.XmlReader.MoveToElement**.]

XmlTextReader.MoveToFirstAttribute() Method

```
[ILASM]
.method public hidebysig virtual bool
MoveToFirstAttribute()

[C#]
public override bool MoveToFirstAttribute()
```

Summary

Moves the position of the current instance to the first attribute associated with the current node.

Return Value

A **System.Boolean** where **true** indicates the current node contains at least one attribute; otherwise, **false**.

Description

If **System.Xml.XmlTextReader.AttributeCount** is non-zero, the reader moves to the first attribute; otherwise, the position of the reader does not change.

[*Note:* This method overrides **System.Xml.XmlReader.MoveToFirstAttribute**.]

XmlTextReader.MoveToNextAttribute() Method

```
[ILASM]
.method public hidebysig virtual bool MoveToNextAttribute()

[C#]
public override bool MoveToNextAttribute()
```

Summary

Moves the position of the current instance to the next attribute associated with the current node.

Return Value

A **System.Boolean** where **true** indicates the reader moved to the next attribute; **false** if there were no more attributes.

Description

If the current node is an element node, this method is equivalent to **System.Xml.XmlTextReader.MoveToFirstAttribute**.

[*Note:* This method overrides **System.Xml.XmlReader.MoveToNextAttribute**.]

XmlTextReader.Read() Method

```
[ILASM]
.method public hidebysig virtual bool Read()

[C#]
public override bool Read()
```

Summary

Moves the position of the current instance to the next node in the stream, exposing its properties.

Return Value

A **System.Boolean** where **true** indicates the node was read successfully, and **false** indicates there were no more nodes to read.

Description

[*Note:* When a reader is first created and initialized, there is no information available. Calling this method is required to read the first node.

This method overrides **System.Xml.XmlReader.Read**.]

Exceptions

Exception	Condition
System.Xml.XmlException	An error occurred while parsing the XML.

XmlTextReader.ReadAttributeValue() Method

```
[ILASM]
.method public hidebysig virtual bool ReadAttributeValue()

[C#]
public override bool ReadAttributeValue()
```

Summary

Parses an attribute value into one or more **Text** and **EntityReference** nodes.

Return Value

A **System.Boolean** where **true** indicates the attribute value was parsed, and **false** indicates the reader was not positioned on an attribute node or all the attribute values have been read.

Description

The **System.Xml.XmlTextReader** class does not expand general entities; any encountered are returned as a single empty **EntityReference** node (**System.Xml.XmlTextReader.Value** is **System.String.Empty**).

[*Note:* Use this method after calling **System.Xml.XmlTextReader.MoveToAttribute** to read through the text or entity reference nodes that make up the attribute value. The **System.Xml.XmlTextReader.Depth** of the attribute value nodes is one plus the depth of the attribute node. When general entity references are stepped into or out of, the **System.Xml.XmlTextReader.Depth** is incremented of decremented by one, respectively.

This method overrides **System.Xml.XmlReader.ReadAttributeValue**.]

XmlTextReader.ReadBase64(System.Byte[], System.Int32, System.Int32) Method

```
[ILASM]
.method public hidebysig instance int32 ReadBase64(class
System.Byte[] array, int32 offset, int32 len)

[C#]
public int ReadBase64(byte[] array, int offset, int len)
```

Summary

Reads and decodes the Base64 encoded contents of an element and stores the result in a byte buffer.

Parameters

Parameter	Description
array	A **System.Byte** array to store the content.
offset	A **System.Int32** specifying the zero-based index into *array* where the method should begin to write.
len	A **System.Int32** specifying the number of bytes to write.

Return Value

A **System.Int32** containing the number of bytes written to *array*, or zero if the current instance is not positioned on an element.

Description

[*Note:* This method can be called successively to read large streams of embedded text.

Base64 encoding represents byte sequences in a text form comprised of the 65 US-ASCII characters (A-Z, a-z, 0-9, +, /, =) where each character encodes 6 bits of the binary data.

For more information on Base64 encoding, see RFC 2045 (http://www.ietf.org/rfc/2045).]

Exceptions

Exception	Condition
System.ArgumentNullException	*array* is **null**.

System.ArgumentOutOfRangeException	*offset* < 0, or *len* < 0. - or - *len* > *array*.**Length** - *offset*.
System.Xml.XmlException	The Base64 sequence is not valid.

1
2
3

XmlTextReader.ReadBinHex(System.Byte[], System.Int32, System.Int32) Method

```
[ILASM]
.method public hidebysig instance int32 ReadBinHex(class
System.Byte[] array, int32 offset, int32 len)

[C#]
public int ReadBinHex(byte[] array, int offset, int len)
```

Summary

Reads and decodes the BinHex encoded contents of an element and stores the result in a byte buffer.

Parameters

Parameter	Description
array	A **System.Byte** array to store the content.
offset	A **System.Int32** specifying the zero-based index into *array* where the method should begin to write.
len	A **System.Int32** specifying the number of bytes to write.

Return Value

A **System.Int32** containing the number of bytes written to *array*, or zero if the current instance is not positioned on an element.

Description

[*Note:* This method can be called successively to read large streams of embedded text.

For information on BinHex encoding, see RFC 1741 (http://www.ietf.org/rfc/rfc1741).]

Exceptions

Exception	Condition
System.ArgumentNullException	*array* is **null**.
System.ArgumentOutOfRangeException	*offset* < 0, or *len* < 0. The Base64 sequence is not valid. - or -

	len > *array*.**Length** - *offset*.
System.Xml.XmlException	The BinHex sequence is not valid.

1
2
3

XmlTextReader.ReadChars(System.Char[], System.Int32, System.Int32) Method

```
[ILASM]
.method public hidebysig instance int32 ReadChars(class
System.Char[] buffer, int32 index, int32 count)

[C#]
public int ReadChars(char[] buffer, int index, int count)
```

Summary

Reads the text contents of an element into a character buffer.

Parameters

Parameter	Description
buffer	A **System.Char** array to store the content.
index	A **System.Int32** specifying the zero-based index into *buffer* where the method should begin to write.
count	A **System.Int32** specifying the number of characters to read and store in *buffer*.

Return Value

A **System.Int32** containing the number of characters written to *buffer*, or zero if the current instance is not positioned on an element.

Description

If the end of the character stream in the element is reached before the specified number of characters is read, the return value will be less than *count*.

This method has the following functionality:

- It is designed to work on element nodes only; it returns zero for other node types.
- It returns the actual character content including markup. There is no attempt to resolve entities, CDATA, or any other markup encountered.
- It ignores XML markup that is not well-formed. For example, when reading the following XML string <A>1<A>2</A>,

1<A>2</A> is returned. (It returns markup from the matching element pair and ignores others.)

- It does not do any normalization.
- When it has reached the end of the character stream, the reader is positioned after the end tag.
- Attribute read methods are not available.

[*Note:* Using this method is the most efficient way to process very large streams of text embedded in an XML document. Rather than allocating large string objects, this method returns text content a buffer at a time.]

Exceptions

Exception	Condition
System.ArgumentException	*count* > *buffer*.**Length** - *index*.
System.ArgumentNullException	*buffer* is **null**.
System.ArgumentOutOfRangeException	*index* < 0, or *count* < 0.

XmlTextReader.ReadInnerXml() Method

```
[ILASM]
.method public hidebysig virtual string ReadInnerXml()

[C#]
public override string ReadInnerXml()
```

Summary

Reads the contents of the current node, including child nodes and markup.

Return Value

A **System.String** containing the XML content, or **System.String.Empty** if the current node is neither an element nor attribute, or has no child nodes.

Description

The current node and corresponding end node are not returned.

If the current node is an element, after the call to this method, the reader is positioned after the corresponding end element.

If the current node is an attribute, the position of the reader is not changed.

[*Note:* If the reader is positioned on a **System.Xml.XmlNodeType** other than **Element** or **Attribute**, calling this method is equivalent to calling the **System.Xml.XmlTextReader.Read** method.

A comparison between calling the **System.Xml.XmlTextReader.ReadInnerXml** and **System.Xml.XmlTextReader.ReadOuterXml** methods on a XML fragment is shown below.

Assume the reader is positioned on <book1 in the following XML fragment.

```
<books>
   <book1 id="123" cost="39.95">
      Title1
      <page1/>
   </book1>
</books>
```

Calling **System.Xml.XmlTextReader.ReadInnerXml** returns

```
Title1


<page1/>
```

Calling **System.Xml.XmlTextReader.ReadOuterXml** returns

```
<book1 id="123" cost="39.95">
   Title1
   <page1/>
</book1>
```

After either method returns, the reader is positioned on </books>.

Assume the reader is positioned on id in the previous XML fragment.

Calling **System.Xml.XmlTextReader.ReadInnerXml** returns

```
123
```

Calling **System.Xml.XmlTextReader.ReadOuterXml** returns

```
id="123"
```

After either method returns, the reader is still positioned on `id`.

This method overrides **System.Xml.XmlReader.ReadInnerXml**.]

Exceptions

Exception	Condition
System.Xml.XmlException	The XML was not well-formed, or an error occurred while parsing the XML.

XmlTextReader.ReadOuterXml() Method

```
[ILASM]
.method public hidebysig virtual string ReadOuterXml()

[C#]
public override string ReadOuterXml()
```

Summary

Reads the current node and its contents, including child nodes and markup.

Return Value

A **System.String** containing the XML content, or **System.String.Empty** if the current node is neither an element nor attribute.

Description

The current node and corresponding end node are returned.

If the current node is an element, after the call to this method, the reader is positioned after the corresponding end element.

If the current node is an attribute, the position of the reader is not changed.

[*Note:* If the reader is positioned on a **System.Xml.XmlNodeType** other than **Element** or **Attribute**, calling this method is equivalent to calling the **System.Xml.XmlTextReader.Read** method.

For a comparison between this method and the **System.Xml.XmlTextReader.ReadInnerXml** method, see **System.Xml.XmlTextReader.ReadInnerXml**.

This method overrides **System.Xml.XmlReader.ReadOuterXml**.]

Exceptions

Exception	Condition
System.Xml.XmlException	The XML was not well-formed, or an error occurred while parsing the XML.

XmlTextReader.ReadString() Method

```
[ILASM]
.method public hidebysig virtual string ReadString()

[C#]
public override string ReadString()
```

Summary

Reads the contents of an element or a text node as a string.

Return Value

A **System.String** containing the contents of the **Element** or **Text** node, or **System.String.Empty** if the reader is positioned on any other type of node.

Description

If positioned on an **Element** node, this method concatenates all **Text**, **SignificantWhitespace**, **Whitespace**, and **CDATA** node types, and returns the concatenated data as the element content. If none of these node types exist, **System.String.Empty** is returned. Concatenation stops when any markup is encountered, which can occur in a mixed content model or when an element end tag is read.

If positioned on an element **Text** node, this method performs the same concatenation from the **Text** node to the element end tag. If the reader is positioned on an attribute **Text** node, this method has the same functionality as if the reader were position on the element start tag.

[*Note:* This method overrides **System.Xml.XmlReader.ReadString**.]

Exceptions

Exception	Condition
System.InvalidOperationException	An invalid operation was attempted.
System.Xml.XmlException	An error occurred while parsing the XML.

XmlTextReader.ResetState() Method

```
[ILASM]
.method public hidebysig instance void ResetState()

[C#]
public void ResetState()
```

Summary

Resets the **System.Xml.XmlTextReader.ReadState** to **System.Xml.ReadState.Initial**.

Description

The **System.Xml.XmlTextReader.Normalization**, **System.Xml.XmlTextReader.WhitespaceHandling**, **System.Xml.XmlTextReader.Namespaces**, and **System.Xml.XmlTextReader.XmlResolver** properties are not changed by this method.

[*Note:* This method enables the parsing of multiple XML documents in a single stream. When the end of an XML document is reached, this method resets the state of the current instance in preparation for the next XML document.]

Exceptions

Exception	Condition
System.InvalidOperationException	The current instance was constructed with a **System.Xml.XmlParserContext**.

XmlTextReader.ResolveEntity() Method

```
[ILASM]
.method public hidebysig virtual void ResolveEntity()

[C#]
public override void ResolveEntity()
```

Summary

Resolves the entity reference for **EntityReference** nodes.

Description

[*Note:* **System.Xml.XmlTextReader** does not support entity resolution.

This method overrides **System.Xml.XmlReader.ResolveEntity**.]

Exceptions

Exception	Condition
System.InvalidOperationException	Any call to this method.

XmlTextReader.AttributeCount Property

```
[ILASM]
.property int32 AttributeCount { public hidebysig virtual
specialname int32 get_AttributeCount() }

[C#]
public override int AttributeCount { get; }
```

Summary

Gets the number of attributes on the current node.

Property Value

A **System.Int32** containing the number of attributes on the current node, or zero if the current node does not support attributes.

Description

This property is read-only.

[*Note:* This property is relevant to the **DocumentType**, **Element**, and **XmlDeclaration** node types of the **System.Xml.XmlNodeType** enumeration. Other node types do not have attributes.

This property overrides **System.Xml.XmlReader.AttributeCount**.]

XmlTextReader.BaseURI Property

```
[ILASM]
.property string BaseURI { public hidebysig virtual
specialname string get_BaseURI() }

[C#]
public override string BaseURI { get; }
```

Summary

Gets the base Uniform Resource Identifier (URI) of the current node.

Property Value

The base URI of the current node.

Description

This property is read-only.

This property is set when the reader is instantiated and defaults to **System.String.Empty**.

[*Note:* A networked XML document is comprised of chunks of data aggregated using various W3C standard inclusion mechanisms and therefore contains nodes that come from different places. Document Type Definition (DTD) entities are an example of this, but this is not limited to DTDs. The base URI tells where these nodes come from.

This property overrides **System.Xml.XmlReader.BaseURI**.]

XmlTextReader.Depth Property

```
[ILASM]
.property int32 Depth { public hidebysig virtual
specialname int32 get_Depth() }

[C#]
public override int Depth { get; }
```

Summary

Gets the depth of the current node in the XML document.

Property Value

A **System.Int32** containing the depth of the current node in the XML document.

Description

This property is read-only.

[*Note:* This property overrides **System.Xml.XmlReader.Depth**.]

XmlTextReader.Encoding Property

```
[ILASM]
.property class System.Text.Encoding Encoding { public
hidebysig specialname instance class System.Text.Encoding
get_Encoding() }

[C#]
public Encoding Encoding { get; }
```

Summary

Gets the encoding of the document.

Property Value

If the **System.Xml.XmlTextReader.ReadState** is **System.Xml.ReadState.Interactive**, a **System.Text.Encoding**; otherwise **null**.

Description

This property is read-only.

If no encoding attribute exists, this property defaults to UTF-8.

XmlTextReader.EOF Property

```
[ILASM]
.property bool EOF { public hidebysig virtual specialname
bool get_EOF() }

[C#]
public override bool EOF { get; }
```

Summary

Gets a value indicating whether the **System.Xml.XmlTextReader.ReadState** is **System.Xml.ReadState.EndOfFile**, signifying the reader is positioned at the end of the stream.

Property Value

A **System.Boolean** where **true** indicates the reader is positioned at the end of the stream; otherwise, **false**.

Description

This property is read-only.

[*Note:* This property overrides **System.Xml.XmlReader.EOF**.]

XmlTextReader.HasValue Property

```
[ILASM]
.property bool HasValue { public hidebysig virtual
specialname bool get_HasValue() }

[C#]
public override bool HasValue { get; }
```

Summary

Gets a value indicating whether the current node can have an associated text value.

Property Value

A **System.Boolean** where **true** indicates the node on which the reader is currently positioned can have an associated text value; otherwise, **false**.

Description

This property is read-only.

The following members of the **System.Xml.XmlNodeType** enumeration can have an associated value: **Attribute**, **CDATA**, **Comment**, **DocumentType**, **ProcessingInstruction**, **SignificantWhitespace**, **Text**, **Whitespace**, and **XmlDeclaration**.

[*Note:* This property overrides **System.Xml.XmlReader.HasValue**.]

XmlTextReader.IsDefault Property

```
[ILASM]
.property bool IsDefault { public hidebysig virtual
specialname bool get_IsDefault() }

[C#]
public override bool IsDefault { get; }
```

Summary

Gets a value indicating whether the current node is an attribute that was generated from the default value defined in the DTD or schema.

Property Value

This property always returns the **System.Boolean** value **false**.

Description

This property is read-only.

This property applies only to attribute nodes.

[*Note:* **System.Xml.XmlTextReader** does not expand default attributes.

This property overrides **System.Xml.XmlReader.IsDefault**.]

XmlTextReader.IsEmptyElement Property

```
[ILASM]
.property bool IsEmptyElement { public hidebysig virtual
specialname bool get_IsEmptyElement() }

[C#]
public override bool IsEmptyElement { get; }
```

Summary

Gets a value indicating whether the current node is an empty element (for example, `<MyElement />`).

Property Value

A **System.Boolean** where **true** indicates the current node is an element (**System.Xml.XmlTextReader.NodeType** equals **System.Xml.XmlNodeType.Element**) that ends with "/>"; otherwise, **false**.

Description

This property is read-only.

A **System.Xml.XmlNodeType.EndElement** node is not generated for empty elements.

[*Note:* This property determines the difference between the following:

`<item bar="123"/>` (**IsEmptyElement** is **true**).

`<item bar="123">` (**IsEmptyElement** is **false**).

This property overrides **System.Xml.XmlReader.IsEmptyElement**.]

XmlTextReader.Item Property

```
[ILASM]
.property string Item[int32 i] { public hidebysig virtual
specialname string get_Item(int32 i) }

[C#]
public override string this[int i] { get; }
```

Summary

Retrieves the value of the attribute with the specified index relative to the containing element.

Parameters

Parameter	Description
i	A **System.Int32** specifying the zero-based index of the attribute relative to the containing element.

Property Value

A **System.String** containing the value of the attribute.

Description

This property is read-only.

This property does not move the reader.

[*Note:* This property overrides the **System.Xml.XmlReader** indexer.]

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	*i* is less than 0 or greater than or equal to the **System.Xml.XmlTextReader.AttributeCoun** of the containing element.[*Note:* **System.Xml.XmlTextReader.AttributeCoun** returns zero for all node types except **Attribute**, **DocumentType**, **Element**, and **XmlDeclaration**. Therefore, this exception is thrown if the reader is not positioned on one o these node types.]

1
2
3

XmlTextReader.Item Property

```
[ILASM]
.property string Item[string name] { public hidebysig
virtual specialname string get_Item(string name) }

[C#]
public override string this[string name] { get; }
```

Summary

Retrieves the value of the attribute with the specified qualified name.

Parameters

Parameter	Description
name	A **System.String** specifying the qualified name of the attribute.

Property Value

A **System.String** containing the value of the specified attribute, or **null** if the attribute is not found.

Description

This property is read-only.

This property does not move the reader.

[*Note:* If the reader is positioned on a **DocumentType** node, this method can be used to get the PUBLIC and SYSTEM literals.

This property overrides the **System.Xml.XmlReader** indexer.]

XmlTextReader.Item Property

```
[ILASM]
.property string Item[string name, string namespaceURI] {
public hidebysig virtual specialname string get_Item(string
name, string namespaceURI) }

[C#]
public override string this[string name, string
namespaceURI] { get; }
```

Summary

Retrieves the value of the attribute with the specified local name and namespace URI.

Parameters

Parameter	Description
name	A **System.String** specifying the local name of the attribute.
namespaceURI	A **System.String** specifying the namespace URI of the attribute.

Property Value

A **System.String** containing the value of the specified attribute, or **null** if the attribute is not found.

Description

This property is read-only.

This property does not move the reader.

[*Note:* This property overrides the **System.Xml.XmlReader** indexer.]

XmlTextReader.LineNumber Property

```
[ILASM]
.property int32 LineNumber { public final hidebysig virtual
specialname int32 get_LineNumber() }

[C#]
public int LineNumber { get; }
```

Summary

Gets the current line number.

Property Value

A **System.Int32** containing the current line number.

Description

This property is read-only.

The constructors initialize this property to one.

[*Note:* This property is most commonly used for error reporting, but can be called at any time.

The start of a document is indicated when this property is 1 and the **System.Xml.XmlTextReader.LinePosition** property is 1.]

XmlTextReader.LinePosition Property

```
[ILASM]
.property int32 LinePosition { public final hidebysig
virtual specialname int32 get_LinePosition() }

[C#]
public int LinePosition { get; }
```

Summary

Gets the current position in a line.

Property Value

A **System.Int32** containing the current line position.

Description

This property is read-only.

The constructors initialize this property to one, which indicates the first character of text in a line.

[*Note:* For example, <root>, contains the character 'r' at **System.Xml.XmlTextReader.LinePosition** equal to 2 and the character '>' at **System.Xml.XmlTextReader.LinePosition** equal to 6.

This property is most commonly used for error reporting, but can be called at any time.

The start of a document is indicated when this property is 1 and the **System.Xml.XmlTextReader.LineNumber** property is 1.]

XmlTextReader.LocalName Property

```
[ILASM]
.property string LocalName { public hidebysig virtual
specialname string get_LocalName() }

[C#]
public override string LocalName { get; }
```

Summary

Gets the local name of the current node.

Property Value

A **System.String** containing the local name of the current node or, for node types that do not have a name (like **Text**, **Comment**, and so on), **System.String.Empty**.

Description

This property is read-only.

The local name is equivalent to **System.Xml.XmlTextReader.Name** with **System.Xml.XmlTextReader.Prefix** and the ':' character removed. For example, **System.Xml.XmlTextReader.LocalName** is "book" for the element <bk:book>.

[*Note:* This property overrides **System.Xml.XmlReader.LocalName**.]

XmlTextReader.Name Property

```
[ILASM]
.property string Name { public hidebysig virtual
specialname string get_Name() }

[C#]
public override string Name { get; }
```

Summary

Gets the qualified name of the current node.

Property Value

A **System.String** containing the qualified name of the current node or, for node types that do not have a name (like **Text**, **Comment**, and so on), **System.String.Empty**.

Description

This property is read-only.

The name returned is dependent on the **System.Xml.XmlTextReader.NodeType** of the node. The following node types return the listed values. All other node types return an empty string.

Node Type	Name
Attribute	The name of the attribute.
DocumentType	The document type name.
Element	The tag name.
EntityReference	The name of the entity referenced.
ProcessingInstruction	The target of the processing instruction.
XmlDeclaration	The literal string "xml".

[*Note:* The qualified name is equivalent to the **System.Xml.XmlTextReader.LocalName** prefixed with **System.Xml.XmlTextReader.Prefix** and the ':' character. For example, **System.Xml.XmlTextReader.Name** is "bk:book" for the element <bk:book>.

This property overrides **System.Xml.XmlReader.Name**.]

XmlTextReader.Namespaces Property

```
[ILASM]
.property bool Namespaces { public hidebysig specialname
instance bool get_Namespaces() public hidebysig specialname
instance void set_Namespaces(bool value) }

[C#]
public bool Namespaces { get; set; }
```

Summary

Gets or sets a value indicating whether the reader supports namespaces.

Property Value

A **System.Boolean** where **true** indicates the reader supports namespaces; otherwise, **false**. The default is **true**.

Description

This property determines whether the reader supports the XML Namespaces specification (http://www.w3.org/TR/REC-xml-names). If this property is **false**, namespaces are ignored and the reader allows names to contain multiple colon characters.

If an attempt is made to set this property after a read operation has occurred, a **System.InvalidOperationException** is thrown.

Exceptions

Exception	Condition
System.InvalidOperationException	When attempting to set the property, the **System.Xml.XmlTextReader.ReadState** was not **System.Xml.ReadState.Initial**.

XmlTextReader.NamespaceURI Property

```
[ILASM]
.property string NamespaceURI { public hidebysig virtual
specialname string get_NamespaceURI() }

[C#]
public override string NamespaceURI { get; }
```

Summary

Gets the namespace URI associated with the node on which the reader is positioned.

Property Value

A **System.String** containing the namespace URI of the current node or, if no namespace URI is associated with the current node, **System.String.Empty**.

Description

This property is read-only.

This property is relevant to **Element** and **Attribute** nodes only.

[*Note:* Namespaces conform to the W3C "Namespaces in XML" recommendation, REC-xml-names-19990114.

This property overrides **System.Xml.XmlReader.NamespaceURI**.]

XmlTextReader.NameTable Property

```
[ILASM]
.property class System.Xml.XmlNameTable NameTable { public
hidebysig virtual specialname class System.Xml.XmlNameTable
get_NameTable() }

[C#]
public override XmlNameTable NameTable { get; }
```

Summary

Gets the name table used by the current instance to store and look up element and attribute names, prefixes, and namespaces.

Property Value

The **System.Xml.XmlNameTable** used by the current instance.

Description

This property is read-only.

The **System.Xml.XmlTextReader** class stores element and attribute names, prefixes, and namespaces as individual **System.String** objects when a document is read.

A qualified name is stored as a unique **System.String** instance and separated into its prefix and local name parts, which are also stored as unique strings instances. For example, <somePrefix:someElement>, is stored as three strings, "somePrefix:someElement", "somePrefix", and "someElement".

[*Note:* This property overrides **System.Xml.XmlReader.NameTable**.]

XmlTextReader.NodeType Property

```
[ILASM]
.property valuetype System.Xml.XmlNodeType NodeType {
public hidebysig virtual specialname valuetype
System.Xml.XmlNodeType get_NodeType() }

[C#]
public override XmlNodeType NodeType { get; }
```

Summary

Gets the **System.Xml.XmlNodeType** of the current node.

Property Value

One of the members of the **System.Xml.XmlNodeType** enumeration representing the type of the current node.

Description

This property is read-only.

This property does not return the following **System.Xml.XmlNodeType** types: **Document**, **DocumentFragment**, **Entity**, **EndEntity**, or **Notation**.

[*Note:* This property overrides **System.Xml.XmlReader.NodeType**.]

XmlTextReader.Normalization Property

```
[ILASM]
.property bool Normalization { public hidebysig specialname
instance bool get_Normalization() public hidebysig
specialname instance void set_Normalization(bool value) }

[C#]
public bool Normalization { get; set; }
```

Summary

Gets or sets a value indicating whether to normalize white space and attribute values.

Property Value

A **System.Boolean** where **true** indicates to normalize; otherwise, **false**. The default is **false**.

Description

This property can be changed at any time before the current instance has been closed and takes affect on the next read operation.

If **System.Xml.XmlTextReader.Normalization** is set to **false**, this also disables character range checking for numeric entities. As a result, character entities, such as " &#0", are allowed.

[*Note:* See "Attribute-Value Normalization" in the W3C XML 1.0 recommendation, REC-xml-19980210.]

Exceptions

Exception	Condition
System.InvalidOperationException	When attempting to set the property, the current instance has been closed.

XmlTextReader.Prefix Property

```
[ILASM]
.property string Prefix { public hidebysig virtual
specialname string get_Prefix() }

[C#]
public override string Prefix { get; }
```

Summary

Gets the namespace prefix associated with the current node.

Property Value

A **System.String** containing the namespace prefix associated with the current node.

Description

This property is read-only.

[*Note:* A namespace prefix is used as a reference for a namespace URI and is defined in an element declaration. For example, `<someElement xmlns:bk='someURL'>`, defines a prefix name "bk".

This property overrides **System.Xml.XmlReader.Prefix**.]

XmlTextReader.QuoteChar Property

```
[ILASM]
.property valuetype System.Char QuoteChar { public
hidebysig virtual specialname valuetype System.Char
get_QuoteChar() }

[C#]
public override char QuoteChar { get; }
```

Summary

Gets the quotation mark character used to enclose the value of an attribute.

Property Value

A **System.Char** specifying the quotation mark character (" or ') used to enclose the value of an attribute.

Description

This property is read-only.

This property applies only to an **Attribute** node.

[*Note:* This property overrides **System.Xml.XmlReader.QuoteChar**.]

XmlTextReader.ReadState Property

```
[ILASM]
.property valuetype System.Xml.ReadState ReadState { public
hidebysig virtual specialname valuetype
System.Xml.ReadState get_ReadState() }

[C#]
public override ReadState ReadState { get; }
```

Summary

Gets the read state of the reader.

Property Value

One of the members of the **System.Xml.ReadState** enumeration.

Description

This property is read-only.

[*Note:* This property overrides **System.Xml.XmlReader.ReadState**.]

XmlTextReader.Value Property

```
[ILASM]
.property string Value { public hidebysig virtual
specialname string get_Value() }

[C#]
public override string Value { get; }
```

Summary

Gets the text value of the current node.

Property Value

A **System.String** containing the text value of the current node.

Description

This property is read-only.

The value returned depends on the **System.Xml.XmlTextReader.NodeType**. The following table lists node types that have a value to return. All other node types return **System.String.Empty**.

Node Type	Value
Attribute	The value of the attribute.
CDATA	The content of the CDATA section.
Comment	The content of the comment.
DocumentType	The internal subset.
ProcessingInstruction	The entire content, excluding the target.
SignificantWhitespace	The white space in the scope of `xml:space = "preserve"`.
Text	The content of the text node.
Whitespace	The white space between markup.
XmlDeclaration	The content of the declaration.

[*Note:* This property overrides **System.Xml.XmlReader.Value**.]

XmlTextReader.WhitespaceHandling Property

```
[ILASM]
.property valuetype System.Xml.WhitespaceHandling
WhitespaceHandling { public hidebysig specialname instance
valuetype System.Xml.WhitespaceHandling
get_WhitespaceHandling() public hidebysig specialname
instance void set_WhitespaceHandling(valuetype
System.Xml.WhitespaceHandling value) }

[C#]
public WhitespaceHandling WhitespaceHandling { get; set; }
```

Summary

Gets or sets a value that specifies the type of white space returned by the reader.

Property Value

One of the members of the **System.Xml.WhitespaceHandling** enumeration. The default is **System.Xml.WhitespaceHandling.All** (returns both significant and insignificant white space).

Description

This property can be changed at any time before the current instance is closed and takes affect on the next read operation.

[*Note:* Because an instance of the **System.Xml.XmlTextReader** class does not have DTD information available to it, **SignificantWhitespace** nodes are only returned within the `xml:space="preserve"` scope.]

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The value to be set is not one of the members of the **System.Xml.WhitespaceHandling** enumeration.
System.InvalidOperationException	When setting the property, the **System.Xml.XmlTextReader.ReadState** is **System.Xml.ReadState.Closed**.

XmlTextReader.XmlLang Property

```
[ILASM]
.property string XmlLang { public hidebysig virtual
specialname string get_XmlLang() }

[C#]
public override string XmlLang { get; }
```

Summary

Gets the current `xml:lang` scope.

Property Value

A **System.String** containing the current `xml:lang` scope.

Description

This property is read-only.

[*Note:* This property represents the `xml:lang` scope within which the current node resides. For example, the following is an XML fragment with `xml:lang` set to US English:

```
<root xml:lang="en-us">


<name>Fred</name>


</root>
```

When the reader is positioned on the `name` element, this property returns "en-us".

The returned string is also in the **System.Xml.XmlTextReader.NameTable** for the reader.

This property overrides **System.Xml.XmlReader.XmlLang**.]

XmlTextReader.XmlResolver Property

```
[ILASM]
.property class System.Xml.XmlResolver XmlResolver { public
hidebysig specialname instance void set_XmlResolver(class
System.Xml.XmlResolver value) }

[C#]
public XmlResolver XmlResolver { set; }
```

Summary

Sets the **System.Xml.XmlResolver** used for resolving DTD references.

Property Value

The **System.Xml.XmlResolver** to use for resolving DTD references.

If this property is not set, the current instance uses a new instance of the **System.Xml.XmlUrlResolver** class with default credentials. If this property is set to **null**, any external DTD or entities encountered by the reader are not resolved.

Description

This property is write-only.

The **System.Xml.XmlResolver** is used to resolve the location of the file loaded into the reader and also to resolve DTD references. For example, if the XML included the DOCTYPE declaration, `<!DOCTYPE book SYSTEM book.dtd>`, the reader resolves this external file and ensures that the DTD is well-formed. **System.Xml.XmlTextReader** does not use the DTD for validation.

This property can be changed at any time and takes affect on the next read operation.

XmlTextReader.XmlSpace Property

```
[ILASM]
.property valuetype System.Xml.XmlSpace XmlSpace { public
hidebysig virtual specialname valuetype System.Xml.XmlSpace
get_XmlSpace() }

[C#]
public override XmlSpace XmlSpace { get; }
```

Summary

Gets the current `xml:space` scope.

Property Value

One of the members of the **System.Xml.XmlSpace** enumeration. If no `xml:space` scope exists, this property defaults to **System.Xml.XmlSpace.None**.

Description

This property is read-only.

[*Note:* The **System.Xml.XmlTextReader** class has no DTD information available; therefore, **SignificantWhitespace** nodes are only returned when inside the scope of `xml:space = "preserve"`.

This property overrides **System.Xml.XmlReader.XmlSpace**.]