

# 1 System.Xml.XmlTextReader Class

```
2 [ILAsm]  
3 .class public XmlTextReader extends System.Xml.XmlReader  
4 [C#]  
5 public class XmlTextReader: XmlReader
```

## 6 Assembly Info:

- 7 • Name: System.Xml
- 8 • Public Key: [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- 9 • Version: 2.0.x.x
- 10 • Attributes:
  - 11 ○ CLSCompliantAttribute(true)

## 12 Type Attributes:

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

## 15 Summary

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

## 17 Inherits From: System.Xml.XmlReader

18  
19 Library: XML

20  
21 **Thread Safety:** All public static members of this type are safe for multithreaded operations.  
22 No instance members are guaranteed to be thread safe.

## 24 Description

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

27  
28 This class implements the `System.Xml.XmlReader` class and conforms to the W3C  
29 Extensible Markup Language (XML) 1.0 and the Namespaces in XML recommendations.

30  
31 A given set of XML data is modeled as a tree of nodes. The different types of nodes are  
32 specified in the `System.Xml.XmlNodeType` enumeration. The current node refers to the  
33 node on which the reader is positioned. The reader is advanced using any of the "read"  
34 or "moveto" methods. The following table lists the node properties exposed for the  
35 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 <code>System.Xml.XmlReader</code> )
HasValue	Whether the node can have a text value.
IsDefault	Whether an <code>Attribute</code> node was generated from the default value defined in the DTD or schema.
IsEmptyElement	Whether an <code>Element</code> node is empty.
LocalName	The local name of the node.
Name	The qualified name of the node, equal to <code>Prefix:LocalName</code> .
NamespaceUri	The URI defining the namespace associated with the node.
NodeType	The <code>System.Xml.XmlNodeType</code> 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 <code>xml:lang</code> scope within which the node resides.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13

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 can be different than the actual node type of the current node.

# 1 XmlTextReader(System.String) Constructor

```
2 [ILAsm]  
3 public rtspecialname specialname instance void .ctor(string url)  
4 [C#]  
5 public XmlTextReader(string url)
```

## 6 Summary

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

## 9 Parameters

Parameter	Description
<code>url</code>	A <code>System.String</code> specifying the URL for the file containing the XML data.

10

## 11 Description

12 This constructor is equivalent to `System.Xml.XmlTextReader.XmlTextReader(url, new`  
13 `System.Xml.NameTable())`.

## 14 Exceptions

Exception	Condition
<code>System.Xml.XmlException</code>	<code>url</code> is null.

15

16

# 1 XmlTextReader(System.String, 2 System.Xml.XmlNodeType, 3 System.Xml.XmlParserContext) Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(string xmlFragment,  
6 valuetype System.Xml.XmlNodeType fragType, class  
7 System.Xml.XmlParserContext context)  
  
8 [C#]  
9 public XmlTextReader(string xmlFragment, XmlNodeType fragType,  
10 XmlParserContext context)
```

## 11 Summary

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

## 14 Parameters

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

## 15 16 Description

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

XmlNodeType	Fragment Can 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.
----------	--

1  
2 [Note: If the XML fragment is an element or attribute, root level rules for well-formed  
3 XML documents are not enforced.

4  
5 This constructor can handle strings returned from  
6 `System.Xml.XmlTextReader.ReadInnerXml`.

7  
8 ]

9  
10 This constructor calls `System.Xml.XmlTextReader.XmlTextReader(context.NameTable)`  
11 or, if `context` is null,  
12 `System.Xml.XmlTextReader.XmlTextReader(newSystem.Xml.NameTable())` to initialize  
13 properties of the class. Following this call, if `context` is not null, the following  
14 `System.Xml.XmlTextReader` properties are set to the specified values.

Property	Value
BaseUri	<code>context.BaseURI</code> or, if <code>context</code> is null, <code>System.String.Empty</code> .
Encoding	<code>context.Encoding</code> or, if <code>context</code> or <code>context.Encoding</code> is null, UTF-8.
XmlLang	If <code>context</code> is not null, <code>context.XmlLang</code> . If <code>context</code> is null, this property is not changed.
XmlSpace	If <code>context</code> is not null, <code>context.XmlSpace</code> . If <code>context</code> is null, this property is not changed.

15  
16 **Exceptions**

Exception	Condition
<b>System.ArgumentNullException</b>	<code>xmlFragment</code> is null.
<b>System.Xml.XmlException</b>	<code>fragType</code> is not an Element, Attribute, or DocumentSystem.Xml.XmlNodeType.

17  
18

# 1 XmlTextReader(System.String, 2 System.Xml.XmlNameTable) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(string url, class  
5 System.Xml.XmlNameTable nt)  
  
6 [C#]  
7 public XmlTextReader(string url, XmlNameTable nt)
```

## 8 Summary

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

## 11 Parameters

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

12

## 13 Description

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

## 16 Exceptions

Exception	Condition
System.ArgumentNullException	<i>nt</i> is null.
System.Xml.XmlException	<i>url</i> is null.

17

18

# 1 XmlTextReader(System.IO.Stream, 2 System.Xml.XmlNodeType, 3 System.Xml.XmlParserContext) Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(class  
6 System.IO.Stream xmlFragment, valuetype System.Xml.XmlNodeType fragType,  
7 class System.Xml.XmlParserContext context)  
  
8 [C#]  
9 public XmlTextReader(Stream xmlFragment, XmlNodeType fragType,  
10 XmlParserContext context)
```

## 11 Summary

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

## 14 Parameters

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

## 15 16 Description

17 The following table lists valid values for *fragType*.

XmlNodeType	Fragment Can 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.
--	-----------

1  
2 [Note: If the XML fragment is an element or attribute, the root level rules for well-  
3 formed XML documents are not enforced.

4  
5 ]

6  
7 This constructor calls `System.Xml.XmlTextReader(context.NameTable)` or, if `context` is  
8 null, `System.Xml.XmlTextReader(newSystem.Xml.NameTable())` to initialize properties  
9 of the class. Afterwards, the following `System.Xml.XmlTextReader` properties are set to  
10 the specified values.

Property	Value
BaseUri	<code>context.BaseURI</code> or, if <code>context</code> is null, <code>System.String.Empty</code> .
Encoding	<code>context.Encoding</code> or, if <code>context</code> or <code>context.Encoding</code> 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	true.
XmlLang	If <code>context</code> is not null, <code>context.XmlLang</code> . If <code>context</code> is null, this property is not changed.
XmlSpace	If <code>context</code> is not null, <code>context.XmlSpace</code> . If <code>context</code> is null, this property is not changed.

11

12 **Exceptions**

Exception	Condition
<b>System.ArgumentNullException</b>	<code>xmlFragment</code> is null.
<b>System.Xml.XmlException</b>	<code>fragType</code> is not an Element, Attribute, or DocumentSystem.Xml.XmlNodeType.

13

14

# 1 XmlTextReader() Constructor

```
2  [ILAsm]  
3  family rtspecialname specialname instance void .ctor()  
4  [C#]  
5  protected XmlTextReader()
```

## 6 Summary

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

8

# 1 XmlTextReader(System.Xml.XmlNameTable)

## 2 Constructor

```
3 [ILAsm]  
4 family rtspecialname specialname instance void .ctor(class  
5 System.Xml.XmlNameTable nt)  
  
6 [C#]  
7 protected XmlTextReader(XmlNameTable nt)
```

### 8 Summary

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

### 11 Parameters

Parameter	Description
<i>nt</i>	The System.Xml.XmlNameTable to use.

### 12 13 Description

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

Property	Value
Namespaces	true
NameTable	<i>nt</i>
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()

1

## 2 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>nt</i> is null.

3

4

# 1 XmlTextReader(System.IO.Stream)

## 2 Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class  
5 System.IO.Stream input)  
  
6 [C#]  
7 public XmlTextReader(Stream input)
```

### 8 Summary

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

### 11 Parameters

Parameter	Description
<i>input</i>	The <code>System.IO.Stream</code> containing the XML data to read.

### 12 13 Description

14 This constructor is equivalent to  
15 `System.Xml.XmlTextReader.XmlTextReader(System.String.Empty, input, new`  
16 `System.Xml.NameTable())`.

### 17 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<i>input</i> is null.

18  
19

# 1 XmlTextReader(System.String, 2 System.IO.Stream) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(string url, class  
5 System.IO.Stream input)  
  
6 [C#]  
7 public XmlTextReader(string url, Stream input)
```

## 8 Summary

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

## 11 Parameters

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

12

## 13 Description

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

## 16 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<i>input</i> is null.

17

18

# 1 XmlTextReader(System.IO.Stream, 2 System.Xml.XmlNameTable) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class  
5 System.IO.Stream input, class System.Xml.XmlNameTable nt)  
  
6 [C#]  
7 public XmlTextReader(Stream input, XmlNameTable nt)
```

## 8 Summary

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

## 11 Parameters

Parameter	Description
<i>input</i>	The <code>System.IO.Stream</code> containing the XML data to read.
<i>nt</i>	The <code>System.Xml.XmlNameTable</code> to use.

12

## 13 Description

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

## 16 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>input</i> or <i>nt</i> is null.

17

18

# 1 XmlTextReader(System.String, 2 System.IO.Stream, 3 System.Xml.XmlNameTable) Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(string url, class  
6 System.IO.Stream input, class System.Xml.XmlNameTable nt)  
  
7 [C#]  
8 public XmlTextReader(string url, Stream input, XmlNameTable nt)
```

## 9 Summary

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

## 12 Parameters

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

## 13 14 Description

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

17  
18 System.Xml.XmlTextReader.Encoding is set to the encoding corresponding to the byte-  
19 order mark at the beginning of the stream or, if no byte-order mark is found, UTF-8.

20  
21 System.Xml.XmlTextReader.BaseURI is set to *url* or, if *url* is null, to  
22 System.String.Empty.

## 23 Exceptions

Exception	Condition
System.ArgumentNullException	<i>input</i> or <i>nt</i> is null.

24  
25

# 1 XmlTextReader(System.IO.TextReader)

## 2 Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class  
5 System.IO.TextReader input)  
  
6 [C#]  
7 public XmlTextReader(TextReader input)
```

### 8 Summary

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

### 11 Parameters

Parameter	Description
<i>input</i>	A <code>System.IO.TextReader</code> , set to the correct encoding, containing the XML data to read.

12

### 13 Description

14 This constructor is equivalent to  
15 `System.Xml.XmlTextReader.XmlTextReader(System.String.Empty, input, new`  
16 `System.Xml.NameTable())`.

17

# 1 XmlTextReader(System.String, 2 System.IO.TextReader) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(string url, class  
5 System.IO.TextReader input)  
  
6 [C#]  
7 public XmlTextReader(string url, TextReader input)
```

## 8 Summary

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

## 11 Parameters

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

12

## 13 Description

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

16

# 1 XmlTextReader(System.IO.TextReader, 2 System.Xml.XmlNameTable) Constructor

```
3 [ILAsm]  
4 public rtspecialname specialname instance void .ctor(class  
5 System.IO.TextReader input, class System.Xml.XmlNameTable nt)  
  
6 [C#]  
7 public XmlTextReader(TextReader input, XmlNameTable nt)
```

## 8 Summary

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

## 11 Parameters

Parameter	Description
<i>input</i>	A <code>System.IO.TextReader</code> , set to the correct encoding, containing the XML data to read.
<i>nt</i>	The <code>System.Xml.XmlNameTable</code> to use.

12

## 13 Description

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

## 16 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<i>nt</i> is null.

17

18

# 1 XmlTextReader(System.String, 2 System.IO.TextReader, 3 System.Xml.XmlNameTable) Constructor

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(string url, class  
6 System.IO.TextReader input, class System.Xml.XmlNameTable nt)  
7  
8 [C#]  
9 public XmlTextReader(string url, TextReader input, XmlNameTable nt)
```

## 9 Summary

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

## 12 Parameters

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

13

## 14 Description

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

17

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

20

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

23

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

27

28 ]

## 29 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>nt</i> is null.

1

2

# 1 XmlTextReader.Close() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual void Close()  
4 [C#]  
5 public override void Close()
```

## 6 Summary

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

## 8 Description

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

13  
14 [*Note:* This method overrides System.Xml.XmlReader.Close.

15  
16 ]

17

# 1 XmlTextReader.GetAttribute(System.String)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual string GetAttribute(string name)  
5 [C#]  
6 public override string GetAttribute(string name)
```

### 7 Summary

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

### 9 Parameters

Parameter	Description
<i>name</i>	A System.String specifying the qualified name of the attribute.

10

### 11 Return Value

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

### 14 Description

15 This method does not move the reader.

16

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

19

20 This method overrides System.Xml.XmlReader.GetAttribute.

21

22 ]

### 23 Example

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

26

# 1 XmlTextReader.GetAttribute(System.String, 2 System.String) Method

```
3 [ILAsm]  
4 .method public hidebysig virtual string GetAttribute(string localName,  
5 string namespaceURI)  
  
6 [C#]  
7 public override string GetAttribute(string localName, string namespaceURI)
```

## 8 Summary

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

## 10 Parameters

Parameter	Description
<i>localName</i>	A System.String specifying the local name of the attribute.
<i>namespaceURI</i>	A System.String specifying the namespace URI of the attribute.

11

## 12 Return Value

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

## 15 Description

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

17

18 This method does not move the reader.

19

20 [Note: This method overrides System.Xml.XmlReader.GetAttribute.

21

22 ]

## 23 Example

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

26

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

28

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

30

31 [C#]

```

1  using System;
2  using System.Xml;
3
4  public class Reader {
5
6      public static void Main() {
7
8          string xmlFragment = @"<test xmlns:dt=""urn:datatypes""
9                                  dt:type=""int""/>";
10
11         NameTable nameTable = new NameTable();
12         XmlNamespaceManager xmlNsMan = new
13             XmlNamespaceManager(nameTable);
14         XmlParserContext xmlPContext = new
15             XmlParserContext(null, xmlNsMan,
16                             null, XmlSpace.None);
17         XmlTextReader xmlTReader = new
18             XmlTextReader(xmlFragment, XmlNodeType.Element,
19                           xmlPContext);
20
21         xmlTReader.Read();
22         Console.WriteLine( "{0}", xmlTReader.GetAttribute(0) );
23
24         string str1 = xmlTReader.GetAttribute(1);
25         string str2 = xmlTReader.GetAttribute("dt:type");
26         string str3 = xmlTReader.GetAttribute("type",
27                                             "urn:datatypes");
28         Console.WriteLine("{0} - {1} - {2}",
29                           str1, str2, str3);
30     }
31 }

```

33 The output is

```

34
35 urn:datatypes
36
37 int - int - int

```

38

# 1 XmlTextReader.GetAttribute(System.Int32)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual string GetAttribute(int32 i)  
5 [C#]  
6 public override string GetAttribute(int i)
```

### 7 Summary

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

### 10 Parameters

Parameter	Description
<i>i</i>	A <code>System.Int32</code> specifying the zero-based index of the attribute relative to the containing element.

### 11 Return Value

13 A `System.String` containing the value of the specified attribute.

### 14 Description

15 This method does not move the reader.

16  
17 [*Note:* This method overrides `System.Xml.XmlReader.GetAttribute`.]  
18

### 20 Exceptions

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

--	--

1

2 **Example**

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

5

# 1 XmlTextReader.GetRemainder() Method

```
2 [ILAsm]  
3 .method public hidebysig instance class System.IO.TextReader  
4 GetRemainder()  
  
5 [C#]  
6 public TextReader GetRemainder()
```

## 7 Summary

8 Returns the remainder of the buffered XML.

## 9 Return Value

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

## 11 Description

12 This method calls the System.Xml.XmlTextReader.Close method, and then resets the  
13 System.Xml.XmlTextReader.ReadState to System.Xml.ReadState.EndOfFile.  
14

15 [*Note:* Because System.Xml.XmlTextReader performs a buffered read operation, it must  
16 be able to return the remainder of the unused buffer so that no data is lost. For  
17 example, this allows protocols (such as multi-part MIME) to package XML in the same  
18 stream.  
19

20 ]  
21

# 1 2 XmlTextReader.LookupNamespace(System.St 3 ring) Method

```
4 [ILAsm]  
5 .method public hidebysig virtual string LookupNamespace(string prefix)  
6 [C#]  
7 public override string LookupNamespace(string prefix)
```

## 8 Summary

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

## 10 Parameters

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

## 11 12 Return Value

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

## 17 Description

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

```
21  
22 <root xmlns:a="urn:456">  
23  
24 <item>  
25  
26 <ref href="a:b"/>  
27  
28  
29  
30  
31 </item>  
32  
33  
34 </root>
```

35  
36

1 This method overrides `System.Xml.XmlReader.LookupNamespace`.  
2  
3 ]

#### 4 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The <code>System.Xml.XmlTextReader.Namespaces</code> property of the current instance is <code>true</code> and <i>prefix</i> is <code>null</code> .

5

6

1  
2 **XmlTextReader.MoveToAttribute(System.Int32) Method**  
3

```
4 [ILAsm]  
5 .method public hidebysig virtual void MoveToAttribute(int32 i)  
6 [C#]  
7 public override void MoveToAttribute(int i)
```

8 **Summary**

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

11 **Parameters**

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

12  
13 **Description**

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

17 [Note: This method overrides System.Xml.XmlReader.MoveToAttribute.

18 ]  
19  
20

21 **Exceptions**

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	<i>i</i> is less than 0 or greater than or equal to the System.Xml.XmlTextReader.AttributeCount of the containing element. [Note: System.Xml.XmlTextReader.AttributeCount 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 of these node types.]

--	--

1

2

1  
2 **XmlTextReader.MoveToAttribute(System.String, System.String) Method**  
3

```
4 [ILAsm]  
5 .method public hidebysig virtual bool MoveToAttribute(string localName,  
6 string namespaceURI)  
  
7 [C#]  
8 public override bool MoveToAttribute(string localName, string  
9 namespaceURI)
```

10 **Summary**

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

13 **Parameters**

Parameter	Description
<i>localName</i>	A System.String specifying the local name of the attribute.
<i>namespaceURI</i>	A System.String specifying the namespace URI of the attribute.

14  
15 **Return Value**

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

19 **Description**

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

21  
22 After calling this method, the System.Xml.XmlTextReader.Name,  
23 System.Xml.XmlTextReader.NamespaceURI, and System.Xml.XmlTextReader.Prefix  
24 properties reflect the properties of the new attribute.

25  
26 [Note: This method overrides System.Xml.XmlReader.MoveToAttribute.

27  
28 ]

29

# 1 2 XmlTextReader.MoveToAttribute(System.String) Method 3

```
4 [ILAsm]  
5 .method public hidebysig virtual bool MoveToAttribute(string name)  
6 [C#]  
7 public override bool MoveToAttribute(string name)
```

## 8 Summary

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

## 11 Parameters

Parameter	Description
<i>name</i>	A <code>System.String</code> specifying the qualified name of the attribute.

## 12 13 Return Value

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

## 17 Description

18 After calling this method, the `System.Xml.XmlTextReader.Name`,  
19 `System.Xml.XmlTextReader.NamespaceURI`, and `System.Xml.XmlTextReader.Prefix`  
20 properties reflect the properties of the new attribute.

21 [Note: This method overrides `System.Xml.XmlReader.MoveToAttribute`.  
22  
23  
24 ]

25

# 1 XmlTextReader.MoveToElement() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual bool MoveToElement()  
4 [C#]  
5 public override bool MoveToElement()
```

## 6 Summary

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

## 9 Return Value

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

## 12 Description

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

15 This method overrides System.Xml.XmlReader.MoveToElement.  
16  
17  
18 ]

19

# 1 XmlTextReader.MoveToFirstAttribute() 2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual bool MoveToFirstAttribute()  
5 [C#]  
6 public override bool MoveToFirstAttribute()
```

## 7 Summary

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

## 10 Return Value

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

## 13 Description

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

16  
17 [*Note:* This method overrides System.Xml.XmlReader.MoveToFirstAttribute.  
18  
19 ]

20

# 1 XmlTextReader.MoveNextAttribute() 2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual bool MoveNextAttribute()  
5 [C#]  
6 public override bool MoveNextAttribute()
```

## 7 Summary

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

## 10 Return Value

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

## 13 Description

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

16  
17 [*Note:* This method overrides `System.Xml.XmlReader.MoveNextAttribute`.

18  
19 ]

20

# 1 XmlTextReader.Read() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual bool Read()  
4 [C#]  
5 public override bool Read()
```

## 6 Summary

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

## 9 Return Value

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

## 12 Description

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

15  
16 This method overrides `System.Xml.XmlReader.Read`.  
17  
18 ]

## 19 Exceptions

Exception	Condition
<code>System.Xml.XmlException</code>	An error occurred while parsing the XML.

20

21

# 1 XmlTextReader.ReadAttributeValue() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual bool ReadAttributeValue()  
4 [C#]  
5 public override bool ReadAttributeValue()
```

## 6 Summary

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

## 8 Return Value

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

## 12 Description

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

16  
17 [*Note:* Use this method after calling `System.Xml.XmlTextReader.MoveToAttribute` to  
18 read through the text or entity reference nodes that make up the attribute value. The  
19 `System.Xml.XmlTextReader.Depth` of the attribute value nodes is one plus the depth of  
20 the attribute node. When general entity references are stepped into or out of, the  
21 `System.Xml.XmlTextReader.Depth` is incremented or decremented by one, respectively.

22  
23 This method overrides `System.Xml.XmlReader.ReadAttributeValue`.

24  
25 ]

26

# 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
<i>array</i>	A <code>System.Byte</code> array to store the content.
<i>offset</i>	A <code>System.Int32</code> specifying the zero-based index into <i>array</i> where the method should begin to write.
<i>len</i>	A <code>System.Int32</code> 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
<b>System.ArgumentNullException</b>	<i>array</i> is null.
<b>System.ArgumentOutOfRangeException</b>	<i>offset</i> < 0, or <i>len</i> < 0. - or - <i>len</i> > <i>array.Length</i> - <i>offset</i> .
<b>System.Xml.XmlException</b>	The Base64 sequence is not valid.

1

2

# 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
<i>array</i>	A <code>System.Byte</code> array to store the content.
<i>offset</i>	A <code>System.Int32</code> specifying the zero-based index into <i>array</i> where the method should begin to write.
<i>len</i>	A <code>System.Int32</code> 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
<code>System.ArgumentNullException</code>	<i>array</i> is null.

<b>System.ArgumentOutOfRangeException</b>	<i>offset &lt; 0, or len &lt; 0.</i>  -or-  <i>len &gt; array.Length - offset.</i>
<b>System.Xml.XmlException</b>	The BinHex sequence is not valid.

1

2

# 1 XmlTextReader.ReadChars(System.Char[], 2 System.Int32, System.Int32) Method

```
3 [ILAsm]  
4 .method public hidebysig instance int32 ReadChars(char[] buffer, int32  
5 index, int32 count)  
  
6 [C#]  
7 public int ReadChars(char[] buffer, int index, int count)
```

## 8 Summary

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

## 10 Parameters

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

## 11 12 Return Value

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

## 15 Description

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

18 This method has the following functionality:

- 20 • It is designed to work on element nodes only; it returns zero for other node types.
- 21 • It returns the actual character content including markup. There is no attempt to  
22 resolve entities, CDATA, or any other markup encountered.
- 23 • It ignores XML markup that is not well-formed. For example, when reading the  
24 following XML string `<A>1<A>2</A>`, `1<A>2</A>` is returned. (It returns markup from  
25 the matching element pair and ignores others.)

- 1 • It does not do any normalization.
  - 2 • When it has reached the end of the character stream, the reader is positioned after
  - 3 the end tag.
  - 4 • Attribute read methods are not available.
- 5 [Note: Using this method is the most efficient way to process very large streams of text
- 6 embedded in an XML document. Rather than allocating large string objects, this method
- 7 returns text content a buffer at a time.
- 8
- 9 ]

## 10 Exceptions

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

11

12

# 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/>
```

1  
2 Calling `System.Xml.XmlTextReader.ReadOuterXml` returns

```
3
4 <book1 id="123" cost="39.95">
5     Title1
6     <page1/>
7 </book1>
```

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

9

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

11

12 Calling `System.Xml.XmlTextReader.ReadInnerXml` returns

```
13
14 123
```

15

16

17 Calling `System.Xml.XmlTextReader.ReadOuterXml` returns

```
18
19 id="123"
```

20

21

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

23

24 This method overrides `System.Xml.XmlReader.ReadInnerXml`.

25

26 ]

27 **Exceptions**

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

28  
29

# 1 XmlTextReader.ReadOuterXml() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual string ReadOuterXml()  
4 [C#]  
5 public override string ReadOuterXml()
```

## 6 Summary

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

## 8 Return Value

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

## 11 Description

12 The current node and corresponding end node are returned.

13  
14 If the current node is an element, after the call to this method, the reader is positioned  
15 after the corresponding end element.

16  
17 If the current node is an attribute, the position of the reader is not changed.

18  
19 *[Note:* If the reader is positioned on a `System.Xml.XmlNodeType` other than `Element` or  
20 `Attribute`, calling this method is equivalent to calling the  
21 `System.Xml.XmlTextReader.Read` method.

22  
23 For a comparison between this method and the  
24 `System.Xml.XmlTextReader.ReadInnerXml` method, see  
25 `System.Xml.XmlTextReader.ReadInnerXml`.

26  
27 This method overrides `System.Xml.XmlReader.ReadOuterXml`.

28  
29 ]

## 30 Exceptions

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

31

32

# 1 XmlTextReader.ReadString() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual string ReadString()  
4 [C#]  
5 public override string ReadString()
```

## 6 Summary

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

## 8 Return Value

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

## 11 Description

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

18  
19 If positioned on an element `Text` node, this method performs the same concatenation  
20 from the `Text` node to the element end tag. If the reader is positioned on an attribute  
21 `Text` node, this method has the same functionality as if the reader were position on the  
22 element start tag.

23  
24 [*Note:* This method overrides `System.Xml.XmlReader.ReadString`.  
25  
26 ]

## 27 Exceptions

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

28

29

# 1 XmlTextReader.ResetState() Method

```
2 [ILAsm]  
3 .method public hidebysig instance void ResetState()  
4 [C#]  
5 public void ResetState()
```

## 6 Summary

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

## 8 Description

9 The System.Xml.XmlTextReader.Normalization,  
10 System.Xml.XmlTextReader.WhitespaceHandling,  
11 System.Xml.XmlTextReader.Namespaces, and  
12 System.Xml.XmlTextReader.XmlResolver properties are not changed by this method.

13  
14 [*Note:* This method enables the parsing of multiple XML documents in a single stream.  
15 When the end of an XML document is reached, this method resets the state of the  
16 current instance in preparation for the next XML document.

17 ]  
18 ]

## 19 Exceptions

Exception	Condition
<b>System.InvalidOperationException</b>	The current instance was constructed with a System.Xml.XmlParserContext.

20

21

# 1 XmlTextReader.ResolveEntity() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual void ResolveEntity()  
4 [C#]  
5 public override void ResolveEntity()
```

## 6 Summary

7 Resolves the entity reference for EntityReference nodes.

## 8 Description

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

10 This method overrides System.Xml.XmlReader.ResolveEntity.  
11 ]

12 ]  
13 ]

## 14 Exceptions

Exception	Condition
System.InvalidOperationException	Any call to this method.

15

16

# 1 XmlTextReader.AttributeCount Property

```
2 [ILAsm]  
3 .property int32 AttributeCount { public hidebysig virtual specialname  
4 int32 get_AttributeCount() }  
5 [C#]  
6 public override int AttributeCount { get; }
```

## 7 Summary

8 Gets the number of attributes on the current node.

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 [*Note:* This property is relevant to the DocumentType, Element, and XmlDeclaration  
16 node types of the System.Xml.XmlNodeType enumeration. Other node types do not have  
17 attributes.

18  
19 This property overrides System.Xml.XmlReader.AttributeCount.

20  
21 ]

22

# 1 XmlTextReader.BaseURI Property

```
2 [ILAsm]  
3 .property string BaseURI { public hidebysig virtual specialname string  
4 get_BaseURI() }  
5 [C#]  
6 public override string BaseURI { get; }
```

## 7 Summary

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

## 9 Property Value

10 The base URI of the current node.

## 11 Description

12 This property is read-only.

13

14 This property is set when the reader is instantiated and defaults to

15 `System.String.Empty`.

16

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

21

22 This property overrides `System.Xml.XmlReader.BaseURI`.

23

24 ]

25

# 1 XmlTextReader.Depth Property

```
2 [ILAsm]  
3 .property int32 Depth { public hidebysig virtual specialname int32  
4 get_Depth() }  
5 [C#]  
6 public override int Depth { get; }
```

## 7 Summary

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

## 9 Property Value

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

## 11 Description

12 This property is read-only.

13  
14 [*Note:* This property overrides System.Xml.XmlReader.Depth.

15  
16 ]

17

# 1 XmlTextReader.Encoding Property

```
2 [ILAsm]  
3 .property class System.Text.Encoding Encoding { public hidebyref  
4 specialname instance class System.Text.Encoding get_Encoding() }  
  
5 [C#]  
6 public Encoding Encoding { get; }
```

## 7 Summary

8 Gets the encoding of the document.

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14

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

16

# 1 XmlTextReader.EOF Property

```
2 [ILAsm]  
3 .property bool EOF { public hidebysig virtual specialname bool get_EOF() }  
4 [C#]  
5 public override bool EOF { get; }
```

## 6 Summary

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

## 10 Property Value

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

## 13 Description

14 This property is read-only.

15  
16 [*Note:* This property overrides `System.Xml.XmlReader.EOF`.  
17

18 ]

19

# 1 XmlTextReader.HasValue Property

```
2 [ILAsm]  
3 .property bool HasValue { public hidebysig virtual specialname bool  
4 get_HasValue() }  
  
5 [C#]  
6 public override bool HasValue { get; }
```

## 7 Summary

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

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 The following members of the System.Xml.XmlNodeType enumeration can have an  
16 associated value: Attribute, CDATA, Comment, DocumentType, ProcessingInstruction,  
17 SignificantWhitespace, Text, Whitespace, and XmlDeclaration.

18 [Note: This property overrides System.Xml.XmlReader.HasValue.

19  
20  
21 ]

22

# 1 XmlTextReader.IsDefault Property

```
2 [ILAsm]  
3 .property bool IsDefault { public hidebysig virtual specialname bool  
4 get_IsDefault() }  
5 [C#]  
6 public override bool IsDefault { get; }
```

## 7 Summary

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

## 10 Property Value

11 This property always returns the `System.Boolean` value `false`.

## 12 Description

13 This property is read-only.

14

15 This property applies only to attribute nodes.

16

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

18

19 This property overrides `System.Xml.XmlReader.IsDefault`.

20

21 ]

22

# 1 XmlTextReader.IsEmptyElement Property

```
2 [ILAsm]  
3 .property bool IsEmptyElement { public hidebysig virtual specialname bool  
4 get_IsEmptyElement() }  
  
5 [C#]  
6 public override bool IsEmptyElement { get; }
```

## 7 Summary

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

## 10 Property Value

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

## 14 Description

15 This property is read-only.

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

17  
18 [Note: This property determines the difference between the following:

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

20  
21 <item bar="123"> (IsEmptyElement is false).

22  
23 This property overrides System.Xml.XmlReader.IsEmptyElement.

24  
25 ]  
26  
27  
28

# 1 XmlTextReader.Item Property

```
2 [ILAsm]  
3 .property string Item[int32 i] { public hidebysig virtual specialname  
4 string get_Item(int32 i) }  
  
5 [C#]  
6 public override string this[int i] { get; }
```

## 7 Summary

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

## 10 Parameters

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

## 11 12 Property Value

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

## 14 Description

15 This property is read-only.

16  
17 This property does not move the reader.

18  
19 [*Note:* This property overrides the System.Xml.XmlReader indexer.

20  
21 ]

## 22 Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	<i>i</i> is less than 0 or greater than or equal to the System.Xml.XmlTextReader.AttributeCount of the containing element. [ <i>Note:</i> System.Xml.XmlTextReader.AttributeCount 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 of

	these node types.]
--	--------------------

1

2

# 1 XmlTextReader.Item Property

```
2 [ILAsm]  
3 .property string Item[string name] { public hidebysig virtual specialname  
4 string get_Item(string name) }  
5 [C#]  
6 public override string this[string name] { get; }
```

## 7 Summary

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

## 9 Parameters

Parameter	Description
<i>name</i>	A <code>System.String</code> specifying the qualified name of the attribute.

10

## 11 Property Value

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

## 14 Description

15 This property is read-only.

16 This property does not move the reader.  
17

18  
19 [*Note:* If the reader is positioned on a `DocumentType` node, this method can be used to  
20 get the `PUBLIC` and `SYSTEM` literals.

21  
22 This property overrides the `System.Xml.XmlReader` indexer.  
23

24 ]  
25

# 1 XmlTextReader.Item Property

```
2 [ILAsm]  
3 .property string Item[string name, string namespaceURI] { public hideby sig  
4 virtual specialname string get_Item(string name, string namespaceURI) }  
5 [C#]  
6 public override string this[string name, string namespaceURI] { get; }
```

## 7 Summary

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

## 9 Parameters

Parameter	Description
<i>name</i>	A System.String specifying the local name of the attribute.
<i>namespaceURI</i>	A System.String specifying the namespace URI of the attribute.

10

## 11 Property Value

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

## 14 Description

15 This property is read-only.

16

17 This property does not move the reader.

18

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

20

21 ]

22

# 1 XmlTextReader.LineNumber Property

```
2 [ILAsm]  
3 .property int32 LineNumber { public final hidebysig virtual specialname  
4 int32 get_LineNumber() }  
5 [C#]  
6 public int LineNumber { get; }
```

## 7 Summary

8 Gets the current line number.

## 9 Property Value

10 A System.Int32 containing the current line number.

## 11 Description

12 This property is read-only.

13  
14 The constructors initialize this property to one.

15  
16 [*Note:* This property is most commonly used for error reporting, but can be called at any  
17 time.

18  
19 The start of a document is indicated when this property is 1 and the  
20 System.Xml.XmlTextReader.LinePosition property is 1.

21  
22 ]

23

# 1 XmlTextReader.LinePosition Property

```
2 [ILAsm]  
3 .property int32 LinePosition { public final hidebysig virtual specialname  
4 int32 get_LinePosition() }  
  
5 [C#]  
6 public int LinePosition { get; }
```

## 7 Summary

8 Gets the current position in a line.

## 9 Property Value

10 A System.Int32 containing the current line position.

## 11 Description

12 This property is read-only.

13  
14 The constructors initialize this property to one, which indicates the first character of text  
15 in a line.

16  
17 [*Note:* For example, <root>, contains the character 'r' at  
18 System.Xml.XmlTextReader.LinePosition equal to 2 and the character '>' at  
19 System.Xml.XmlTextReader.LinePosition equal to 6.

20  
21 This property is most commonly used for error reporting, but can be called at any time.

22  
23 The start of a document is indicated when this property is 1 and the  
24 System.Xml.XmlTextReader.LineNumber property is 1.

25  
26 ]

27

# 1 XmlTextReader.LocalName Property

```
2 [ILAsm]  
3 .property string LocalName { public hidebysig virtual specialname string  
4 get_LocalName() }  
5 [C#]  
6 public override string LocalName { get; }
```

## 7 Summary

8 Gets the local name of the current node.

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 The local name is equivalent to System.Xml.XmlTextReader.Name with  
16 System.Xml.XmlTextReader.Prefix and the ':' character removed. For example,  
17 System.Xml.XmlTextReader.LocalName is "book" for the element <bk:book>.

18  
19 [*Note:* This property overrides System.Xml.XmlReader.LocalName.

20  
21 ]

22

# 1 XmlTextReader.Name Property

```
2 [ILAsm]  
3 .property string Name { public hidebysig virtual specialname string  
4 get_Name() }  
  
5 [C#]  
6 public override string Name { get; }
```

## 7 Summary

8 Gets the qualified name of the current node.

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 The name returned is dependent on the System.Xml.XmlTextReader.NodeType of the  
16 node. The following node types return the listed values. All other node types return an  
17 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".

18  
19 *[Note:* The qualified name is equivalent to the System.Xml.XmlTextReader.LocalName  
20 prefixed with System.Xml.XmlTextReader.Prefix and the ':' character. For example,  
21 System.Xml.XmlTextReader.Name is "bk:book" for the element <bk:book>.

22  
23 This property overrides System.Xml.XmlReader.Name.

1  
2 ]  
3

# 1 XmlTextReader.Namespaces Property

```
2 [ILAsm]  
3 .property bool Namespaces { public hidebysig specialname instance bool  
4 get_Namespaces() public hidebysig specialname instance void  
5 set_Namespaces(bool value) }  
  
6 [C#]  
7 public bool Namespaces { get; set; }
```

## 8 Summary

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

## 10 Property Value

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

## 13 Description

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

18  
19 If an attempt is made to set this property after a read operation has occurred, a  
20 `System.InvalidOperationException` is thrown.

## 21 Exceptions

Exception	Condition
<b>System.InvalidOperationException</b>	When attempting to set the property, the <code>System.Xml.XmlTextReader.ReadState</code> was not <code>System.Xml.ReadState.Initial</code> .

22

23

# 1 XmlTextReader.NamespaceURI Property

```
2 [ILAsm]  
3 .property string NamespaceURI { public hidebysig virtual specialname  
4 string get_NamespaceURI() }  
5 [C#]  
6 public override string NamespaceURI { get; }
```

## 7 Summary

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

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 This property is relevant to `Element` and `Attribute` nodes only.

16  
17 [*Note:* Namespaces conform to the W3C "Namespaces in XML" recommendation, REC-  
18 xml-names-19990114.

19  
20 This property overrides `System.Xml.XmlReader.NamespaceURI`.

21  
22 ]

23

# 1 XmlTextReader.NameTable Property

```
2 [ILAsm]  
3 .property class System.Xml.XmlNameTable NameTable { public hidebysig  
4 virtual specialname class System.Xml.XmlNameTable get_NameTable() }  
  
5 [C#]  
6 public override XmlNameTable NameTable { get; }
```

## 7 Summary

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

## 10 Property Value

11 The `System.Xml.XmlNameTable` used by the current instance.

## 12 Description

13 This property is read-only.

14  
15 The `System.Xml.XmlTextReader` class stores element and attribute names, prefixes,  
16 and namespaces as individual `System.String` objects when a document is read.

17  
18 A qualified name is stored as a unique `System.String` instance and separated into its  
19 prefix and local name parts, which are also stored as unique strings instances. For  
20 example, `<somePrefix:someElement>`, is stored as three strings,  
21 "somePrefix:someElement", "somePrefix", and "someElement".

22  
23 [*Note:* This property overrides `System.Xml.XmlReader.NameTable`.  
24  
25 ]

26 ]

# 1 XmlTextReader.NodeType Property

```
2 [ILAsm]  
3 .property valuetype System.Xml.XmlNodeType NodeType { public hidebyref  
4 virtual specialname valuetype System.Xml.XmlNodeType get_NodeType() }  
5 [C#]  
6 public override XmlNodeType NodeType { get; }
```

## 7 Summary

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

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 This property does not return the following System.Xml.XmlNodeType types: Document,  
16 DocumentFragment, Entity, EndEntity, Or Notation.

17  
18 [*Note:* This property overrides System.Xml.XmlReader.NodeType.

19  
20 ]

21

# 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.

# 1 XmlTextReader.Prefix Property

```
2 [ILAsm]  
3 .property string Prefix { public hidebysig virtual specialname string  
4 get_Prefix() }  
5 [C#]  
6 public override string Prefix { get; }
```

## 7 Summary

8 Gets the namespace prefix associated with the current node.

## 9 Property Value

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

## 11 Description

12 This property is read-only.

13  
14 [*Note:* A namespace prefix is used as a reference for a namespace URI and is defined in  
15 an element declaration. For example, `<someElement xmlns:bk='someURL'>`, defines a  
16 prefix name "bk".

17  
18 This property overrides `System.Xml.XmlReader.Prefix`.

19  
20 ]

21

# 1 XmlTextReader.QuoteChar Property

```
2 [ILAsm]  
3 .property valuetype System.Char QuoteChar { public hidebysig virtual  
4 specialname valuetype System.Char get_QuoteChar() }  
5 [C#]  
6 public override char QuoteChar { get; }
```

## 7 Summary

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

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 This property applies only to an `Attribute` node.

16  
17 [*Note:* This property overrides `System.Xml.XmlReader.QuoteChar`.

18  
19 ]

20

# 1 XmlTextReader.ReadState Property

```
2 [ILAsm]  
3 .property valuetype System.Xml.ReadState ReadState { public hidebyvisig  
4 virtual specialname valuetype System.Xml.ReadState get_ReadState() }  
5 [C#]  
6 public override ReadState ReadState { get; }
```

## 7 Summary

8 Gets the read state of the reader.

## 9 Property Value

10 One of the members of the `System.Xml.ReadState` enumeration.

## 11 Description

12 This property is read-only.

13

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

15

16 ]

17

# 1 XmlTextReader.Value Property

```
2 [ILAsm]  
3 .property string Value { public hidebysig virtual specialname string  
4 get_Value() }  
  
5 [C#]  
6 public override string Value { get; }
```

## 7 Summary

8 Gets the text value of the current node.

## 9 Property Value

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

## 11 Description

12 This property is read-only.

13  
14 The value returned depends on the System.Xml.XmlTextReader.NodeType. The  
15 following table lists node types that have a value to return. All other node types return  
16 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.

```
1
2 [Note: This property overrides System.Xml.XmlReader.Value.
3
4 ]
5
```

# 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
<b>System.ArgumentOutOfRangeException</b>	The value to be set is not one of the members of the <code>System.Xml.WhitespaceHandling</code> enumeration.
<b>System.InvalidOperationException</b>	When setting the property, the <code>System.Xml.XmlTextReader.ReadState</code> is <code>System.Xml.ReadState.Closed</code> .

# 1 XmlTextReader.XmlLang Property

```
2 [ILAsm]  
3 .property string XmlLang { public hidebysig virtual specialname string  
4 get_XmlLang() }  
5 [C#]  
6 public override string XmlLang { get; }
```

## 7 Summary

8 Gets the current `xml:lang` scope.

## 9 Property Value

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

## 11 Description

12 This property is read-only.

13  
14 [*Note:* This property represents the `xml:lang` scope within which the current node  
15 resides. For example, the following is an XML fragment with `xml:lang` set to US English:

```
16 <root xml:lang="en-us">  
17  
18  
19  
20 <name>Fred</name>  
21  
22  
23 </root>
```

24  
25  
26 When the reader is positioned on the `name` element, this property returns "en-us".

27  
28 The returned string is also in the `System.Xml.XmlTextReader.NameTable` for the  
29 reader.

30  
31 This property overrides `System.Xml.XmlReader.XmlLang`.

32  
33 ]

34

# 1 XmlTextReader.XmlResolver Property

```
2 [ILAsm]
3 .property class System.Xml.XmlResolver XmlResolver { public hideby sig
4 specialname instance void set_XmlResolver(class System.Xml.XmlResolver
5 value) }
6 [C#]
7 public XmlResolver XmlResolver { set; }
```

## 8 Summary

9 Sets the `System.Xml.XmlResolver` used for resolving DTD references.

## 10 Property Value

11 The `System.Xml.XmlResolver` to use for resolving DTD references.

12  
13 If this property is set to `null`, any external DTD or entities encountered by the reader  
14 are not resolved.

## 15 Description

16 This property is write-only.

17  
18 The `System.Xml.XmlResolver` is used to resolve the location of the file loaded into the  
19 reader and also to resolve DTD references. For example, if the XML included the  
20 DOCTYPE declaration, `<!DOCTYPE book SYSTEM book.dtd>`, the reader resolves this  
21 external file and ensures that the DTD is well-formed. `System.Xml.XmlTextReader` does  
22 not use the DTD for validation.

23  
24 This property can be changed at any time and takes affect on the next read operation.

25

# 1 XmlTextReader.XmlSpace Property

```
2 [ILAsm]  
3 .property valuetype System.Xml.XmlSpace XmlSpace { public hidebysig  
4 virtual specialname valuetype System.Xml.XmlSpace get_XmlSpace() }  
  
5 [C#]  
6 public override XmlSpace XmlSpace { get; }
```

## 7 Summary

8 Gets the current `xml:space` scope.

## 9 Property Value

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

## 12 Description

13 This property is read-only.

14  
15 [*Note:* The `System.Xml.XmlTextReader` class has no DTD information available;  
16 therefore, `SignificantWhitespace` nodes are only returned when inside the scope of  
17 `xml:space = "preserve"`.  
18

19 This property overrides `System.Xml.XmlReader.XmlSpace`.

20  
21 ]

22