

1 System.Xml.XmlReader Class

```
2 [ILAsm]  
3 .class public abstract XmlReader extends System.Object  
  
4 [C#]  
5 public abstract class 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 non-cached, forward-only access to XML data.

17 Inherits From: System.Object

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 stream of XML data. This class
26 enforces the rules of well-formed XML but does not perform data validation.

27
28 This class conforms to the W3C Extensible Markup Language (XML) 1.0 and the
29 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 reader is advanced to the
33 next node using the System.Xml.XmlReader.Read method. The current node refers to
34 the node on which the reader is positioned. The following table lists the node properties
35 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.
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
14
15

This class does not expand default attributes or 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 might be different than the actual node type of the current node.

[*Note:* This class is abstract and implemented in the `System.Xml.XmlTextReader` class.

1
2]

3

1 XmlReader() Constructor

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

6 Summary

7 Constructs a new instance of the `System.Xml.XmlReader` class.

8

1 XmlReader.Close() Method

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

6 Summary

7 Changes the `System.Xml.XmlReader.ReadState` to `System.Xml.ReadState.Closed`.

8 Behaviors

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

12

13 How and When to Override

14 This method must be overridden in order to provide the functionality described above,
15 as there is no default implementation.

16

17

1 XmlReader.GetAttribute(System.String)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual abstract string GetAttribute(string name)  
5 [C#]  
6 public abstract 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 <code>System.String</code> 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.

14 Behaviors

15 This method does not move the reader.

16

17 How and When to Override

18 This method must be overridden in order to provide the functionality described above,
19 as there is no default implementation.

20

21 Example

22 For an example demonstrating this method, see
23 `System.Xml.XmlTextReader.GetAttribute(System.String, System.String)`.

24

1 XmlReader.GetAttribute(System.String, 2 System.String) Method

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

8 Summary

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

10 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.

11

12 Return Value

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

15 Behaviors

16 This method does not move the reader.

17

18 How and When to Override

19 This method must be overridden in order to provide the functionality described above,
20 as there is no default implementation.

21

22 Example

23 For an example demonstrating this method, see
24 System.Xml.XmlTextReader.GetAttribute(System.String, System.String).

25

1 XmlReader.GetAttribute(System.Int32)

2 Method

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

15 This method does not move the reader.

17 How and When to Override

18 This method must be overridden in order to provide the functionality described above,
19 as there is no default implementation.

21 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	<i>i</i> is less than 0, or greater than or equal to the <code>System.Xml.XmlReader.AttributeCount</code> of

the containing element.

1

2 **Example**

3 For an example demonstrating this method, see
4 `System.Xml.XmlTextReader.GetAttribute(System.String, System.String)`.

5

1 XmlReader.IsName(System.String) Method

```
2 [ILAsm]  
3 .method public hidebysig static bool IsName(string str)  
4 [C#]  
5 public static bool IsName(string str)
```

6 Summary

7 Determines whether the specified string is a valid XML name.

8 Parameters

Parameter	Description
<i>str</i>	A <code>System.String</code> specifying the name to validate.

9 Return Value

11 A `System.Boolean` where `true` indicates the name is valid; otherwise, `false`.

12 Description

13 [Note: This method uses the W3C XML 1.0 Recommendation
14 (<http://www.w3.org/TR/2000/REC-xml-20001006#NT-Name>) to determine whether the
15 name is valid.

17]

18

1 XmlReader.IsNameToken(System.String)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static bool IsNameToken(string str)  
5 [C#]  
6 public static bool IsNameToken(string str)
```

7 Summary

8 Determines whether the specified string is a valid XML name token (Nmtoken).

9 Parameters

Parameter	Description
<i>str</i>	A System.String specifying the name to validate.

10

11 Return Value

12 A System.Boolean where true indicates the name is valid; otherwise false.

13 Description

14 [Note: This method uses the W3C XML 1.0 Recommendation
15 (<http://www.w3.org/TR/2000/REC-xml-20001006#NT-Nmtoken>) to determine whether
16 the name token is valid.

17

18]

19

1 XmlReader.IsStartElement(System.String, 2 System.String) Method

```
3 [ILAsm]  
4 .method public hidebysig virtual bool IsStartElement(string localname,  
5 string ns)  
  
6 [C#]  
7 public virtual bool IsStartElement(string localname, string ns)
```

8 Summary

9 Determines if a node containing content is an Element node with the specified local
10 name and namespace URI.

11 Parameters

Parameter	Description
<i>localname</i>	A System.String specifying the local name of an element.
<i>ns</i>	A System.String specifying the namespace URI associated with the element.

12

13 Return Value

14 A System.Boolean where true indicates the node is an Element node with the specified
15 local name and namespace URI; false otherwise.

16 Behaviors

17 As described above.

18

19 Default

20 This method calls the System.Xml.XmlReader.MoveToContent method, which
21 determines whether the current node can contain content and, if not, moves the reader
22 to the next content node or the end of the input stream. When the reader is positioned
23 on a content node, the node is checked to determine if it is an Element node with
24 System.Xml.XmlReader.LocalName and System.Xml.XmlReader.NamespaceURI
25 properties equal to *localname* and *ns*, respectively.

26

27 How and When to Override

1 Override this method to customize the behavior of this method in types derived from the
2 System.Xml.XmlReader class.

3

4 **Usage**

5 Use this method to determine whether the node returned by the
6 System.Xml.XmlReader.MoveToContent method is an Element node with the specified
7 local name and namespace URI.

8

9 **Exceptions**

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

10

11

1 XmlReader.IsStartElement(System.String) 2 Method

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

7 Summary

8 Determines if a node containing content is an `Element` node with the specified qualified
9 name.

10 Parameters

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

11 12 Return Value

13 A `System.Boolean` where `true` indicates the node is an `Element` node with the specified
14 name; `false` otherwise.

15 Behaviors

16 As described above.

17

18 Default

19 This method calls the `System.Xml.XmlReader.MoveToContent` method, which
20 determines whether the current node can contain content and, if not, moves the reader
21 to the next content node or the end of the input stream. When the reader is positioned
22 on a content node, the node is checked to determine if it is an `Element` node with a
23 `System.Xml.XmlReader.Name` property equal to *name*.

24

25 How and When to Override

26 Override this method to customize the behavior of this method in types derived from the
27 `System.Xml.XmlReader` class.

1

2 Usage

3 Use this method to determine whether the node returned by the
4 `System.Xml.XmlReader.MoveToContent` method is an `Element` node with the specified
5 name.

6

7 Exceptions

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

8

9

1 XmlReader.IsStartElement() Method

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

6 Summary

7 Determines if a node containing content is an Element node.

8 Return Value

9 A System.Boolean where true indicates the node is an Element node; false otherwise.

10 Behaviors

11 As described above.

12

13 Default

14 This method calls the System.Xml.XmlReader.MoveToContent method, which
15 determines whether the current node can contain content and, if not, moves the reader
16 to the next content node or the end of the input stream. When the reader is positioned
17 on a content node, the node is checked to determine if it is an Element node.

18

19 How and When to Override

20 Override this method to customize the behavior of this method in types derived from the
21 System.Xml.XmlReader class.

22

23 Usage

24 Use this method to determine whether the node returned by the
25 System.Xml.XmlReader.MoveToContent method is an Element node.

26

27 Exceptions

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

1

2

1
2 **XmlReader.LookupNamespace(System.String**
3 **) Method**

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

9 **Summary**

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

11 **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.

12
13 **Return Value**

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

17 **Behaviors**

18 As described above.

19

20 **How and When to Override**

21 This method must be overridden in order to provide the functionality described above,
22 as there is no default implementation.

23

24

1 XmlReader.MoveToAttribute(System.Int32)

2 Method

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

7 Summary

8 Moves the position of the current instance to the attribute with the specified index
9 relative to the containing 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

12 Behaviors

13 After calling this method, the `System.Xml.XmlReader.Name`,
14 `System.Xml.XmlReader.NamespaceURI`, and `System.Xml.XmlReader.Prefix` properties
15 reflect the properties of current attribute.

16

17 How and When to Override

18 This method must be overridden in order to provide the functionality described above,
19 as there is no default implementation.

20

21 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	<i>i</i> is less than 0 or greater than or equal to the <code>System.Xml.XmlReader.AttributeCount</code> of the containing element.

22

1 XmlReader.MoveToAttribute(System.String, 2 System.String) Method

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

8 Summary

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

11 Parameters

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

12

13 Return Value

14 A System.Boolean where true indicates the attribute was found; otherwise, false. If
15 false, the position of the current instance does not change.

16 Behaviors

17 After calling this method, the System.Xml.XmlReader.Name,
18 System.Xml.XmlReader.NamespaceURI, and System.Xml.XmlReader.Prefix properties
19 reflect the properties of current attribute.

20

21 How and When to Override

22 This method must be overridden in order to provide the functionality described above,
23 as there is no default implementation.

24

25

1 XmlReader.MoveToAttribute(System.String)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig virtual abstract bool MoveToAttribute(string  
5 name)  
  
6 [C#]  
7 public abstract 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 System.String specifying the qualified name of the attribute.

12 Return Value

14 A System.Boolean where true indicates the attribute was found; otherwise, false. If
15 false, the reader's position does not change.

16 Behaviors

17 After calling this method, the System.Xml.XmlReader.Name,
18 System.Xml.XmlReader.NamespaceURI, and System.Xml.XmlReader.Prefix properties
19 reflect the properties of current attribute.

20 How and When to Override

22 This method must be overridden in order to provide the functionality described above,
23 as there is no default implementation.

24
25

1 XmlReader.MoveToContent() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual valuetype System.Xml.XmlNodeType  
4 MoveToContent()  
  
5 [C#]  
6 public virtual XmlNodeType MoveToContent()
```

7 Summary

8 Determines whether the current node can contain content and, if not, moves the
9 position of the current instance to the next content node or the end of the input stream.

10 Return Value

11 The `System.Xml.XmlNodeType` of the content node, or `System.Xml.XmlNodeType.None`
12 if the position of the reader has reached the end of the input stream.

13 Description

14 [*Note:* The following members of `System.Xml.XmlNodeType` can contain content:
15 `Attribute`, `CDATA`, `Element`, `EndElement`, `EntityReference`, `EndEntity`, and `Text`.]
16
17

18 Behaviors

19 As described above.
20

21 Default

22 If the current node is an `Attribute` node, this method moves the position of the reader
23 back to the `Element` node that owns the attribute.
24

25 How and When to Override

26 Override this method to customize the behavior of this method in types derived from the
27 `System.Xml.XmlReader` class.
28

29 Usage

1 Use this method to determine whether the current node can contain content and, if not,
2 move the position of the reader to the next content node.

3

4 **Exceptions**

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

5

6

1 XmlReader.MoveToElement() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual abstract bool MoveToElement()  
4 [C#]  
5 public abstract 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 position of the reader was moved; false
11 indicates the reader was not positioned on an Attribute node and therefore the
12 position of the reader was not moved.

13 Description

14 [*Note:* The DocumentType, Element, and XmlDeclaration members of
15 System.Xml.XmlNodeType can contain attributes.]
16
17

18 Behaviors

19 As described above.
20

21 How and When to Override

22 This method must be overridden in order to provide the functionality described above,
23 as there is no default implementation.
24
25

1 XmlReader.MoveToFirstAttribute() Method

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

6 Summary

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

9 Return Value

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

12 Behaviors

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

15

16 How and When to Override

17 This method must be overridden in order to provide the functionality described above,
18 as there is no default implementation.

19

20

1 XmlReader.MoveToNextAttribute() Method

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

6 Summary

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

9 Return Value

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

12 Behaviors

13 As described above.

14

15 How and When to Override

16 This method must be overridden in order to provide the functionality described above,
17 as there is no default implementation.

18

19

1 XmlReader.Read() Method

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

13 As described above.

14

15 How and When to Override

16 This method must be overridden in order to provide the functionality as described
17 herein, as there is no default implementation.

18

19 Usage

20 When a reader is first created and initialized, there is no information available. Calling
21 this method is required to read the first node.

22

23 Exceptions

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

24

25

1 XmlReader.ReadAttributeValue() Method

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

6 Summary

7 Parses an attribute value into one or more Text, EntityReference, and EndEntity
8 nodes.

9 Return Value

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

13 Description

14 [*Note:* To parse an EntityReference node, call the
15 System.Xml.XmlReader.ResolveEntity method. After the node is parsed into child
16 nodes, call the System.Xml.XmlReader.ReadAttributeValue method again to read the
17 value of the entity.

18
19 The System.Xml.XmlReader.Depth of an attribute value node is one plus the depth of
20 the attribute node. When general entity references are stepped into or out of, the
21 System.Xml.XmlReader.Depth increments or decrements by one, respectively.

22
23]

24 Behaviors

25 As described above.

26

27 How and When to Override

28 Implementations that cannot expand general entities should return general entities as a
29 single empty (System.Xml.XmlReader.Value equals System.String.Empty)
30 EntityReference node.

31

32 Usage

1 Use this method after calling `System.Xml.XmlReader.MoveToAttribute` to read through
2 the `Text`, `EntityReference`, or `EndElement` nodes that make up the attribute value. Call
3 the `System.Xml.XmlReader.ResolveEntity` method to resolve the `EntityReference`
4 nodes.

5

6

1 2 XmlReader.ReadElementString(System.String 3 , System.String) Method

```
4 [ILAsm]  
5 .method public hidebysig virtual string ReadElementString(string  
6 localname, string ns)  
7 [C#]  
8 public virtual string ReadElementString(string localname, string ns)
```

9 Summary

10 Reads the contents of a text-only element with the specified local name and namespace
11 URI.

12 Parameters

Parameter	Description
<i>localname</i>	A System.String specifying the local name of an element.
<i>ns</i>	A System.String specifying the namespace URI associated with the element.

13 14 Return Value

15 A System.String containing the contents of the element.

16 Behaviors

17 As described above.

18

19 Default

20 This method calls the System.Xml.XmlReader.MoveToContent method. If the returned
21 node is an Element node, this method compares the System.Xml.XmlReader.LocalName
22 and System.Xml.XmlReader.NamespaceURI properties of the node to *localname* and *ns*,
23 respectively. If they are equal, this method calls the
24 System.Xml.XmlReader.ReadString method to read the contents of the element.

25

26 How and When to Override

1 Override this method to customize the behavior of this method in types derived from the
2 System.Xml.XmlReader class.

3

4 **Usage**

5 Use this method to read the contents of a text-only element with the specified local
6 name and namespace URI.

7

8 **Exceptions**

Exception	Condition
System.Xml.XmlException	The node is not an Element node, the System.Xml.XmlReader.LocalName property of the Element node does not equal <i>localname</i> , or the System.Xml.XmlReader.NamespaceURI property of the Element node does not equal <i>ns</i> , the element does not contain a simple text value, or an error occurred while parsing the XML.

9

10

1 2 XmlReader.ReadElementString(System.String 3) Method

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

8 Summary

9 Reads the contents of a text-only element with the specified qualified name.

10 Parameters

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

11 12 Return Value

13 A `System.String` containing the contents of the element.

14 Behaviors

15 As described above.

17 Default

18 This method calls the `System.Xml.XmlReader.MoveToContent` method and, if the
19 returned node is an `Element` node, compares the `System.Xml.XmlReader.Name` property
20 of the node to *name*. If they are equal, this method calls the
21 `System.Xml.XmlReader.ReadString` method to read the contents of the element.

23 How and When to Override

24 Override this method to customize the behavior of this method in types derived from the
25 `System.Xml.XmlReader` class.

26

1 **Usage**

2 Use this method to read the contents of a text-only element with the specified qualified
3 name.

4

5 **Exceptions**

Exception	Condition
System.Xml.XmlException	The node is not an <code>Element</code> node, the <code>System.Xml.XmlReader.Name</code> property of the <code>Element</code> node does not equal <i>name</i> , the element does not contain a simple text value, or an error occurred while parsing the XML.

6

7

1 XmlReader.ReadElementString() Method

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

6 Summary

7 Reads the contents of a text-only element.

8 Return Value

9 A `System.String` containing the contents of the element.

10 Behaviors

11 As described above.

12

13 Default

14 This method calls the `System.Xml.XmlReader.MoveToContent` method and, if the
15 returned node is an `Element` node, calls the `System.Xml.XmlReader.ReadString`
16 method to read the contents.

17

18 How and When to Override

19 Override this method to customize the behavior of this method in types derived from the
20 `System.Xml.XmlReader` class.

21

22 Usage

23 Use this method to read the contents of a text-only element.

24

25 Exceptions

Exception	Condition
-----------	-----------

System.Xml.XmlException

The node is not an `Element` node, the element does not contain a simple text value, or an error occurred while parsing the XML.

1

2

1 XmlReader.ReadEndElement() Method

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

6 Summary

7 Reads an `EndElement` node and advances the reader to the next node.

8 Behaviors

9 As described above.

10

11 Default

12 This method calls the `System.Xml.XmlReader.MoveToContent` method, which
13 determines whether the current node can contain content and, if not, moves the reader
14 to the next content node or the end of the input stream. The node the reader ends up
15 positioned on is checked to determine if it is an `EndElement` node. If so, the node is read
16 and the reader is moved to the next node.

17

18 How and When to Override

19 Override this method to customize the behavior of this method in types derived from the
20 `System.Xml.XmlReader` class.

21

22 Usage

23 Use this method to read an `EndElement` node and advance the reader to the next node.

24

25 Exceptions

Exception	Condition
<code>System.Xml.XmlException</code>	The node is not an <code>EndElement</code> node or an error occurred while parsing the XML.

1

2

1 XmlReader.ReadInnerXml() Method

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

6 Summary

7 Reads the contents of the current node, 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, or has no child nodes.

11 Behaviors

12 The current node and corresponding end node are not 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: For a comparison between this method and the*
20 *System.Xml.XmlReader.ReadOuterXml method, see*
21 *System.Xml.XmlTextReader.ReadInnerXml.*

22
23]

24 How and When to Override

25 This method must be overridden in order to provide the functionality described above,
26 as there is no default implementation.

27

28 Exceptions

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

29

30

1 XmlReader.ReadOuterXml() Method

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

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: For a comparison between this method and the*
20 *System.Xml.XmlReader.ReadOuterXml method, see*
21 *System.Xml.XmlTextReader.ReadInnerXml.*

22
23]

24 How and When to Override

25 This method must be overridden in order to provide the functionality described above,
26 as there is no default implementation.

27

28 Exceptions

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

29

30

1 XmlReader.ReadStartElement() Method

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

6 Summary

7 Reads an `Element` node and advances the reader to the next node.

8 Behaviors

9 As described above.

10

11 Default

12 This method calls the `System.Xml.XmlReader.MoveToContent` method, which
13 determines whether the current node can contain content and, if not, moves the reader
14 to the next content node or the end of the input stream. The node the reader ends up
15 positioned on is checked to determine if it is an `Element` node. If so, the node is read
16 and the reader is moved to the next node.

17

18 How and When to Override

19 Override this method to customize the behavior of this method in types derived from the
20 `System.Xml.XmlReader` class.

21

22 Usage

23 Use this method to read an `Element` node and advance the reader to the next node.

24

25 Exceptions

Exception	Condition
<code>System.Xml.XmlException</code>	The node is not an <code>Element</code> node or an error occurred while parsing the XML.

1

2

1 XmlReader.ReadStartElement(System.String) 2 Method

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

7 Summary

8 Reads an `Element` node with the specified qualified name and advances the reader to
9 the next node.

10 Parameters

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

11

12 Behaviors

13 As described above.

14

15 Default

16 This method calls the `System.Xml.XmlReader.MoveToContent` method and, if the
17 returned node is an `Element` node, compares the `System.Xml.XmlReader.Name` property
18 of the node to *name*. If they are equal, this method calls the
19 `System.Xml.XmlReader.Read` method to read the element and move to the next node.

20

21 How and When to Override

22 Override this method to customize the behavior of this method in types derived from the
23 `System.Xml.XmlReader` class.

24

25 Usage

1 Use this method to read an `Element` node with the specified qualified name, and
2 advance the reader to the next node.

3

4 **Exceptions**

Exception	Condition
System.Xml.XmlException	The node is not an <code>Element</code> node, the <code>System.Xml.XmlReader.Name</code> property of the <code>Element</code> node does not equal <i>name</i> , or an error occurred while parsing the XML.

5

6

1 XmlReader.ReadStartElement(System.String, 2 System.String) Method

```
3 [ILAsm]  
4 .method public hidebysig virtual void ReadStartElement(string localname,  
5 string ns)  
6 [C#]  
7 public virtual void ReadStartElement(string localname, string ns)
```

8 Summary

9 Reads an Element node with the specified local name and namespace URI and advances
10 the reader to the next node.

11 Parameters

Parameter	Description
<i>localname</i>	A System.String specifying the local name of an element.
<i>ns</i>	A System.String specifying the namespace URI associated with the element.

13 Behaviors

14 As described above.

16 Default

17 This method calls the System.Xml.XmlReader.MoveToContent method. If the returned
18 node is an Element node, this method compares the System.Xml.XmlReader.LocalName
19 and System.Xml.XmlReader.NamespaceURI properties of the node to *localname* and *ns*,
20 respectively. If they are equal, this method calls the System.Xml.XmlReader.Read
21 method to read the element and move to the next node.

23 How and When to Override

24 Override this method to customize the behavior of this method in types derived from the
25 System.Xml.XmlReader class.

1 **Usage**

2 Use this method to read an `Element` node with the specified local name and namespace
3 URI, and advance the reader to the next node.

4

5 **Exceptions**

Exception	Condition
System.Xml.XmlException	The node is not an <code>Element</code> node, the <code>System.Xml.XmlReader.LocalName</code> property of the <code>Element</code> node does not equal <i>localname</i> , the <code>System.Xml.XmlReader.NamespaceURI</code> property of the <code>Element</code> node does not equal <i>ns</i> , or an error occurred while parsing the XML.

6

7

1 XmlReader.ReadString() Method

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

6 Summary

7 Reads the contents of an element or 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 Behaviors

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 How and When to Override

24 This method must be overridden in order to provide the functionality described above,
25 as there is no default implementation.

26

27 Exceptions

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

28

29

1 XmlReader.ResolveEntity() Method

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

6 Summary

7 Resolves the entity reference for EntityReference nodes.

8 Behaviors

9 This method parses the entity reference into child nodes. When the parsing is finished a
10 new System.Xml.XmlNodeType.EndEntity node is placed in the stream to close the
11 EntityReference scope. To step into the entity after this method has been called, call
12 the System.Xml.XmlReader.ReadAttributeValue method if the entity is part of an
13 attribute value, or the System.Xml.XmlReader.Read method if the entity is part of
14 element content.

15
16 If this method is not called, the parser moves to the next node past the entity (child
17 nodes are bypassed).

18 How and When to Override

19 This method must be overridden in order to provide the functionality as described in the
20 Behaviors and Usage sections, as there is no default implementation.

21
22 This method is required to throw an exception for implementations that do not support
23 schema or DTD information. In this case, the
24 System.Xml.XmlReader.CanResolveEntity property is required to return false.

25 Usage

26 Use this method to resolve the entity reference for EntityReference nodes. Before
27 calling this method, determine whether the reader can resolve an entity by checking the
28 System.Xml.XmlReader.CanResolveEntity property.

29

30 Exceptions

Exception	Condition
System.InvalidOperationException	The reader is not positioned on a System.Xml.XmlNodeType.EntityReference node.

31

1 XmlReader.Skip() Method

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

6 Summary

7 Skips over the current element and moves the position of the current instance to the
8 next node in the stream.

9 Behaviors

10 If the reader is positioned on a non-empty Element node
11 (System.Xml.XmlReader.IsEmptyElement equals false), the position of the reader is
12 moved to the node following the corresponding EndElement node. The properties of the
13 nodes that are skipped over are not exposed. If the reader is positioned on any other
14 node type, the position of the reader is moved to the next node, in this case behaving
15 like the System.Xml.XmlReader.Read method.

16

17 Default

18 This method calls the System.Xml.XmlReader.MoveToElement method before skipping
19 to the next node.

20

21 How and When to Override

22 Override this method to customize the behavior of this method in types derived from the
23 System.Xml.XmlReader class.

24

25 Usage

26 Use this method to skip over the current node.

27

28 Exceptions

Exception	Condition
-----------	-----------

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

1

2

1 XmlReader.AttributeCount Property

```
2 [ILAsm]  
3 .property int32 AttributeCount { public hidebysig virtual abstract  
4 specialname int32 get_AttributeCount() }  
5 [C#]  
6 public abstract 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 [*Note:* This property is only relevant to the DocumentType, Element, and
14 XmlDeclaration node types of the System.Xml.XmlNodeType enumeration. Other node
15 types do not have attributes.]
16
17

18 Behaviors

19 As described above.

20
21 This property is read-only.

22 How and When to Override

23 This property must be overridden in order to provide the functionality described above,
24 as there is no default implementation.

25

26

1 XmlReader.BaseURI Property

```
2 [ILAsm]  
3 .property string BaseURI { public hidebysig virtual abstract specialname  
4 string get_BaseURI() }  
  
5 [C#]  
6 public abstract 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 [*Note:* A networked XML document is comprised of chunks of data aggregated using
13 various W3C standard inclusion mechanisms and therefore contains nodes that come
14 from different places. DTD entities are an example of this, but this is not limited to
15 DTDs. The base URI tells where these nodes come from. If there is no base URI for the
16 nodes being returned (for example, they were parsed from an in-memory string),
17 System.String.Empty is returned.]
18
19

20 Behaviors

21 As described above.

22 This property is read-only.

24 How and When to Override

25 This property must be overridden in order to provide the functionality described above,
26 as there is no default implementation.

27

28

1 XmlReader.CanResolveEntity Property

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

7 Summary

8 Gets a value indicating whether this reader can parse and resolve entities.

9 Property Value

10 A System.Boolean equal to false.

11 Behaviors

12 This property returns `true` to indicate the reader can parse and resolve entities;
13 otherwise, `false`.

14
15 This property is read-only.

16 Default

17 This property always returns `false`.

18

19 How and When to Override

20 Override this property to return `true` for implementations that support schema or DTD
21 information.

22

23 Usage

24 Use this property to determine whether the reader can parse and resolve entities.

25

26

1 XmlReader.Depth Property

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

12 As described above.

13

14 This property is read-only.

15 How and When to Override

16 This property must be overridden in order to provide the functionality described above,
17 as there is no default implementation.

18

19

1 XmlReader.EOF Property

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

7 Summary

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

11 Property Value

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

14 Behaviors

15 As described above.

16
17 This property is read-only.

18 How and When to Override

19 This property must be overridden in order to provide the functionality described above,
20 as there is no default implementation.

21

22

1 XmlReader.HasAttributes Property

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

7 Summary

8 Gets a value indicating whether the current node has any attributes.

9 Property Value

10 A `System.Boolean` where `true` indicates the current node has attributes; otherwise,
11 `false`.

12 Behaviors

13 As described above.

14

15 This property is read-only.

16 Default

17 This property returns `true` if the `System.Xml.XmlReader.AttributeCount` property of
18 the current node is greater than zero.

19

20 How and When to Override

21 Override this property to customize the behavior of this property in types derived from
22 the `System.Xml.XmlReader` class.

23

24 Usage

25 Use this property to determine whether the current node has any attributes.

26

27

1 XmlReader.HasValue Property

```
2 [ILAsm]  
3 .property bool HasValue { public hidebysig virtual abstract specialname  
4 bool get_HasValue() }  
  
5 [C#]  
6 public abstract 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 [*Note:* The following members of the System.Xml.XmlNodeType enumeration can have
14 an associated value: Attribute, CDATA, Comment, DocumentType,
15 ProcessingInstruction, SignificantWhitespace, Text, Whitespace, and
16 XmlDeclaration.]
17
18

19 Behaviors

20 As described above.

21
22 This property is read-only.

23 How and When to Override

24 This property must be overridden in order to provide the functionality described above,
25 as there is no default implementation.

26

27

1 XmlReader.IsDefault Property

```
2 [ILAsm]  
3 .property bool IsDefault { public hidebysig virtual abstract specialname  
4 bool get_IsDefault() }  
  
5 [C#]  
6 public abstract 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 A `System.Boolean` where `true` indicates the current node is an attribute whose value
12 was generated from the default value defined in the DTD or schema; `false` indicates the
13 attribute value was explicitly set.

14 Behaviors

15 As described above.

16
17 This property is read-only.

18 How and When to Override

19 This property should return `false` for implementations that do not support schema or
20 DTD information.

21

22

1 XmlReader.IsEmptyElement Property

```
2 [ILAsm]  
3 .property bool IsEmptyElement { public hidebysig virtual abstract  
4 specialname bool get_IsEmptyElement() }  
  
5 [C#]  
6 public abstract 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.XmlReader.NodeType equals System.Xml.XmlNodeType.Element) that
13 ends with ">", otherwise, false.

14 Behaviors

15 A corresponding EndElement node is not generated for empty elements.

16
17 This property is read-only.

18 How and When to Override

19 This property must be overridden in order to provide the functionality described above,
20 as there is no default implementation.

21

22

1 XmlReader.Item Property

```
2 [ILAsm]  
3 .property string Item[int32 i] { public hidebysig virtual abstract  
4 specialname string get_Item(int32 i) }  
  
5 [C#]  
6 public abstract 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 <code>System.Int32</code> 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 Behaviors

15 This property does not move the reader.

16

17 How and When to Override

18 This property must be overridden in order to provide the functionality described above,
19 as there is no default implementation.

20

21 Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	<i>i</i> is less than 0 or greater than or equal to the <code>System.Xml.XmlReader.AttributeCount</code> of the containing element.

1

2

1 XmlReader.Item Property

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

15 This property does not move the reader.

16

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

19 How and When to Override

20 This property must be overridden in order to provide the functionality described above,
21 as there is no default implementation.

22

23

1 XmlReader.Item Property

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

8 Summary

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

10 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.

11

12 Property Value

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

15 Behaviors

16 This property does not move the reader.

17

18 How and When to Override

19 This property must be overridden in order to provide the functionality described above,
20 as there is no default implementation.

21

22

1 XmlReader.LocalName Property

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

13 As described above.

14

15 How and When to Override

16 This property must be overridden in order to provide the functionality described above,
17 as there is no default implementation.

18

19

1 XmlReader.Name Property

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

13 The qualified name is equivalent to the System.Xml.XmlReader.LocalName prefixed with
14 System.Xml.XmlReader.Prefix and the ':' character. For example,
15 System.Xml.XmlReader.Name is "bk:book" for the element <bk:book>.

16
17 The name returned is dependent on the System.Xml.XmlReader.NodeType of the node.
18 The following node types return the listed values. All other node types return an empty
19 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".

20
21 This property is read-only.

22 How and When to Override

1 This property must be overridden in order to provide the functionality described above,
2 as there is no default implementation.

3

4

1 XmlReader.NamespaceURI Property

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

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

14
15 Namespaces conform to the W3C "Namespaces in XML" recommendation, REC-xml-
16 names-19990114.

17
18 This property is read-only.

19 How and When to Override

20 This property must be overridden in order to provide the functionality described above,
21 as there is no default implementation.

22

23

1 XmlReader.NameTable Property

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

8 Summary

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

11 Property Value

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

13 Behaviors

14 Element and attribute names, prefixes, and namespaces are stored as individual
15 System.String objects when a document is read.

16
17 This property is read-only.

18 How and When to Override

19 This property must be overridden in order to provide the functionality described above,
20 as there is no default implementation.

21

22

1 XmlReader.NodeType Property

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

8 Summary

9 Gets the type of the current node.

10 Property Value

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

13 Behaviors

14 This property does not return the following `System.Xml.XmlNodeType` members:
15 `Document`, `DocumentFragment`, `Entity`, `EndEntity`, and `Notation`.

16
17 This property is read-only.

18 How and When to Override

19 This property must be overridden in order to provide the functionality described above,
20 as there is no default implementation.

21

22

1 XmlReader.Prefix Property

```
2 [ILAsm]  
3 .property string Prefix { public hidebysig virtual abstract specialname  
4 string get_Prefix() }  
  
5 [C#]  
6 public abstract 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 [*Note:* A namespace prefix is used as a reference for a namespace URI and is defined in
13 an element declaration. For example, <someElement xmlns:bk="someURL">, defines a
14 prefix name "bk".]
15
16

17 Behaviors

18 As described above.

19
20 This property is read-only.

21 How and When to Override

22 This property must be overridden in order to provide the functionality described above,
23 as there is no default implementation.

24

25

1 XmlReader.QuoteChar Property

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

13 As described above.

14

15 This property is read-only.

16 How and When to Override

17 This property must be overridden in order to provide the functionality described above,
18 as there is no default implementation.

19

20

1 XmlReader.ReadState Property

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

8 Summary

9 Gets the read state of the reader.

10 Property Value

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

12 Behaviors

13 As described above.

14
15 This property is read-only.

16 How and When to Override

17 This property must be overridden in order to provide the functionality described above,
18 as there is no default implementation.

19

20

1 XmlReader.Value Property

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

12 The value returned depends on the System.Xml.XmlReader.NodeType. The following
13 table lists node types that have a value to return. All other node types return
14 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 between markup in a mixed content model, or 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.

15
16 This property is read-only.

1 **How and When to Override**

2 This property must be overridden in order to provide the functionality described above,
3 as there is no default implementation.

4

5

1 XmlReader.XmlLang Property

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

12 As described above.

13

14 This property is read-only.

15 How and When to Override

16 This property must be overridden in order to provide the functionality described above,
17 as there is no default implementation.

18

19

1 XmlReader.XmlSpace Property

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

8 Summary

9 Gets the current `xml:space` scope.

10 Property Value

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

13 Behaviors

14 As described above.

15
16 This property is read-only.

17 How and When to Override

18 This property must be overridden in order to provide the functionality described above,
19 as there is no default implementation.

20

21