System.Xml.XmlNodeType Enum

[ILAsm]
```
.class public sealed serializable XmlNodeType extends System.Enum
```

[C#]
```
public enum XmlNodeType
```

Assembly Info:

- *Name:* System.Xml
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 2.0.x.x
- *Attributes:*
  - CLSCompliantAttribute(true)

Summary

Specifies the type of node.

Inherits From: System.Enum

Library: XML

Description

A given set of XML data is modeled as a tree of nodes. This enumeration specifies the different node types.

XmlNodeType.Attribute Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType Attribute =
2
```

[C#]

```
Attribute = 2
```

Summary

An attribute.

Example XML: id="123"

An `Attribute` node can have the following child node types: `Text` and `EntityReference`. The `Attribute` node does not appear as the child node of any other node type. It is not considered a child node of an `Element`.

XmlNodeType.CDATA Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType CDATA = 4
```

[C#]

```
CDATA = 4
```

Summary

A CDATA section.

Example XML: `<![CDATA[escaped text]]>`

CDATA sections are used to escape blocks of text that would otherwise be recognized as markup. A `CDATA` node cannot have any child nodes. It can appear as the child of the `DocumentFragment`, `EntityReference`, and `Element` nodes.

XmlNodeType.Comment Field

```
[ILAsm]
.field public static literal valuetype System.Xml.XmlNodeType Comment = 8

[C#]
Comment = 8
```

Summary

A comment.

Example XML: `<!-- comment -->`

A `Comment` node cannot have any child nodes. It can appear as the child of the `Document`, `DocumentFragment`, `Element`, and `EntityReference` nodes.

XmlNodeType.Document Field

```
[ILAsm]
.field public static literal valuetype System.Xml.XmlNodeType Document = 9

[C#]
Document = 9
```

Summary

A document object that, as the root of the document tree, provides access to the entire XML document.

A `Document` node can have the following child node types: `XmlDeclaration`, `Element` (maximum of one), `ProcessingInstruction`, `Comment`, and `DocumentType`. It cannot appear as the child of any node types.

XmlNodeType.DocumentFragment Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType
DocumentFragment = 11
```

[C#]

```
DocumentFragment = 11
```

Summary

A document fragment.

The `DocumentFragment` node associates a node or sub-tree with a document without actually being contained within the document. A `DocumentFragment` node can have the following child node types: `Element`, `ProcessingInstruction`, `Comment`, `Text`, `CDATA`, and `EntityReference`. It cannot appear as the child of any node types.

XmlNodeType.DocumentType Field

```
[ILAsm]
.field public static literal valuetype System.Xml.XmlNodeType DocumentType
= 10

[C#]
DocumentType = 10
```

Summary

The document type declaration, indicated by the following tag.

Example XML: `<!DOCTYPE...>`

A `DocumentType` node can have the following child node types: `Notation` and `Entity`. It can appear as the child of the `Document` node.

XmlNodeType.Element Field

```
[ILAsm]
.field public static literal valuetype System.Xml.XmlNodeType Element = 1

[C#]
Element = 1
```

Summary

An element.

Example XML: <name>

An `Element` node can have the following child node types: `Element`, `Text`, `Comment`, `ProcessingInstruction`, `CDATA`, and `EntityReference`. It can be the child of the `Document`, `DocumentFragment`, `EntityReference`, and `Element` nodes.

XmlNodeType.EndElement Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType EndElement =
15
```

[C#]

```
EndElement = 15
```

Summary

An end element.

Example XML: </name>

Returned when `System.Xml.XmlReader` gets to the end of an element.

XmlNodeType.EndEntity Field

```
[ILAsm]
.field public static literal valuetype System.Xml.XmlNodeType EndEntity =
16

[C#]
EndEntity = 16
```

Summary

Returned when `System.Xml.XmlReader` gets to the end of the entity replacement as a result of a call to `System.Xml.XmlReader.ResolveEntity`.

XmlNodeType.Entity Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType Entity = 6
```

[C#]

```
Entity = 6
```

Summary

An entity declaration.

Example XML: <!ENTITY...>

An `Entity` node can have child nodes that represent the expanded entity (for example, `Text` and `EntityReference` nodes). It can appear as the child of the `DocumentType` node.

XmlNodeType.EntityReference Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType
EntityReference = 5
```

[C#]

```
EntityReference = 5
```

Summary

A reference to an entity.

Example XML: &num;

An `EntityReference` node can have the following child node types: `Element`, `ProcessingInstruction`, `Comment`, `Text`, `CDATA`, and `EntityReference`. It can appear as the child of the `Attribute`, `DocumentFragment`, `Element`, and `EntityReference` nodes.

XmlNodeType.None Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType None = 0
```

[C#]

```
None = 0
```

Summary

This is returned by the `System.Xml.XmlReader` if a read method has not been called or if no more nodes are available to be read.

XmlNodeType.Notation Field

```
[ILAsm]
.field public static literal valuetype System.Xml.XmlNodeType Notation =
12

[C#]
Notation = 12
```

Summary

A notation in the document type declaration.

Example XML: `<!NOTATION...>`

A `Notation` node cannot have any child nodes. It can appear as the child of the `DocumentType` node.

XmlNodeType.ProcessingInstruction Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType
ProcessingInstruction = 7
```

[C#]

```
ProcessingInstruction = 7
```

Summary

A processing instruction.

Example XML: `<?pi test?>`

A `ProcessingInstruction` node cannot have any child nodes. It can appear as the child of the `Document`, `DocumentFragment`, `Element`, and `EntityReference` nodes.

XmlNodeType.SignificantWhitespace Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType
SignificantWhitespace = 14
```

[C#]

```
SignificantWhitespace = 14
```

Summary

White space between markup in a mixed content model or white space within the `xml:space="preserve"` scope.

XmlNodeType.Text Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType Text = 3
```

[C#]

```
Text = 3
```

Summary

The text content of a node.

A `Text` node cannot have any child nodes. It can appear as the child node of the `Attribute`, `DocumentFragment`, `Element`, and `EntityReference` nodes.

XmlNodeType.Whitespace Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType Whitespace =
13
```

[C#]

```
Whitespace = 13
```

Summary

White space between markup.

XmlNodeType.XmlDeclaration Field

[ILAsm]

```
.field public static literal valuetype System.Xml.XmlNodeType
XmlDeclaration = 17
```

[C#]

```
XmlDeclaration = 17
```

Summary

The XML declaration.

Example XML: `<?xml version="1.0"?>`

The `XmlDeclaration` node must be the first node in the document. It cannot have children. It is a child of the `Document` node. It can have attributes that provide version and encoding information.